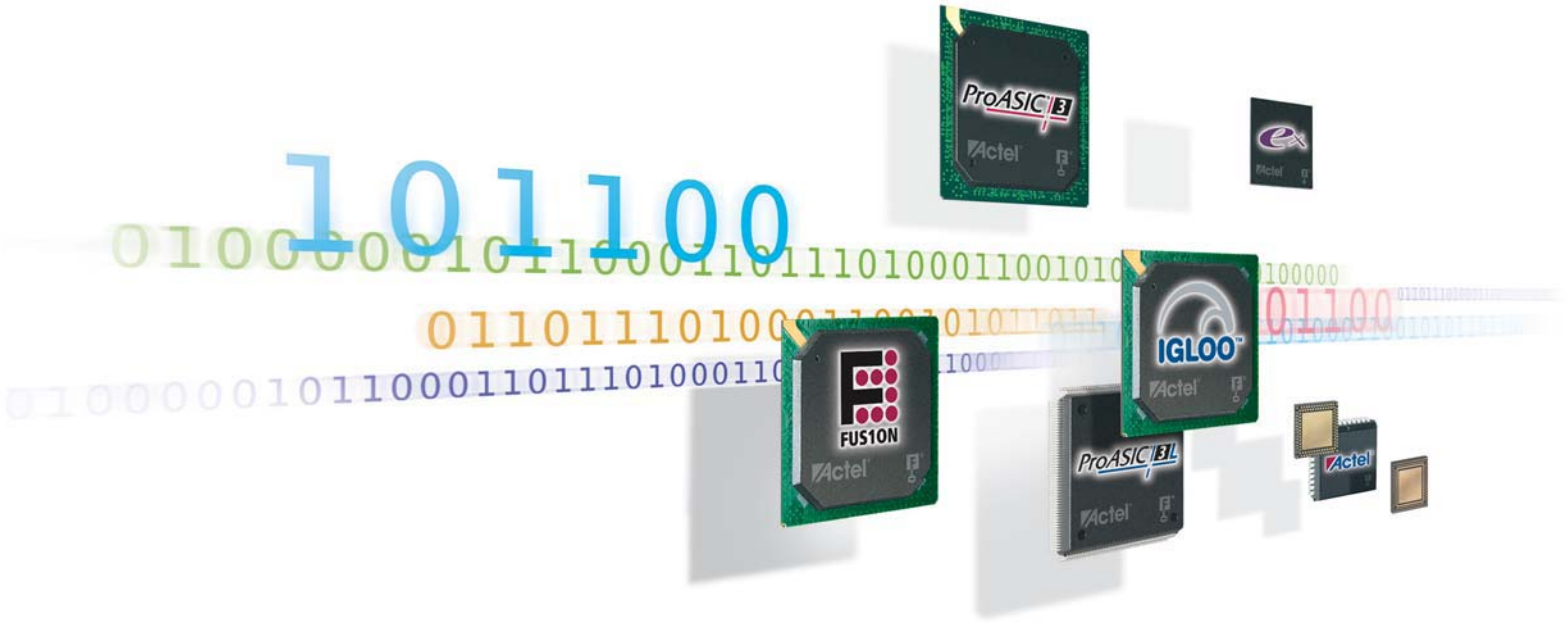
Rate This Document

Actel Fusion® Handbook



Actel® POWER MATTERS

FUS1ON®

Table of Contents

Low-Power Flash Device Handbooks Introduction i

Section I – Actel Fusion® Datasheet

Fusion Device Family Overview 1-1
Device Architecture 2-1
DC and Power Characteristics 3-1
Package Pin Assignments. 4-1

Section II – User's Guide

Low-Power Flash Technology

FPGA Array Architecture in Low-Power Flash Devices. 1-1

Global Resources and Clock Conditioning

Global Resources in Actel Low-Power Flash Devices 2-1
Clock Conditioning Circuits in Low-Power Flash Devices and Mixed-Signal FPGAs 3-1
Fusion Clock Resources 4-1

Embedded Memories

Fusion Embedded Flash Memory Blocks 5-1
FlashROM in Actel's Low-Power Flash Devices 6-1
SRAM and FIFO Memories in Actel's Low-Power Flash Devices 7-1

Analog System

Designing the Fusion Analog System 8-1
Fusion Design Solutions and Methodologies 9-1
Interfacing with the Fusion Analog System: Processor/Microcontroller Interface 10-1
Interfacing with the Fusion Analog System: IP Interface 11-1
Temperature, Voltage, and Current Calibration in Fusion FPGAs 12-1

I/O Descriptions and Usage

I/O Software Control in Low-Power Flash Devices 13-1
DDR for Actel's Low-Power Flash Devices. 14-1

Design Migration

Prototyping With AFS600 for Smaller Devices 15-1

Programming and Security

Programming Flash Devices 16-1

Security in Low-Power Flash Devices 17-1

In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3 18-1

Microprocessor Programming of Actel's Low-Power Flash Devices 19-1

Boundary Scan and UJTAG

Boundary Scan in Low-Power Flash Devices 20-1

UJTAG Applications in Actel's Low-Power Flash Devices 21-1

Board-Level Requirements

Fusion Board-Level Design Guidelines 22-1

Software Design Tutorials and Examples

Fusion Solutions, Design Examples, and Reference Designs 23-1

Fusion Glossary i

Rate This Document

Low-Power Flash Device Handbooks Introduction

Device Handbooks contain all the information available to help designers understand and use Actel's devices. Handbook chapters are grouped into sections on the website to simplify navigation. Each chapter of the handbook can be viewed as an individual PDF file.

At the top of the handbook web page, you will see a PDF file for each product family. This file contains the complete device handbook. Please register for product updates to be notified when a section of the handbook changes.

Table 1 • **Differences between Former Datasheets and Device Handbooks**

Description of Change	Comparison between Handbook and Datasheet
The silicon datasheet in the handbook does not contain the same chapters as the previous versions of the datasheets.	The former version of the silicon datasheet contained the following: • General Description • Device Architecture • DC and Switching Characteristics • Packaging The current datasheet now contains: • Product Brief (same information as the General Description) • DC and Switching Characteristics • Packaging The information previously contained in the Device Architecture chapter has been separated into individual chapters and merged with relevant application note content to provide one location for information on each architectural feature. The Device Architecture section still exists in the Fusion Handbook only.
The General Description section no longer exists in datasheets.	The Product Brief and the General Description consisted of basically the same information but with different titles. To eliminate the duplicated information, we changed the document name to Product Brief.
Change tables were carried forward through this process; they contain information from the old datasheets and the new datasheets.	It is important that all earlier technical changes from the datasheets are listed so customers can determine if any of the changes affect their designs. Changes are listed chronologically, with the most current at the top and the earlier changes listed below them.
Version numbers were restarted.	The version numbers were restarted when the handbooks were created. For example, a datasheet may have been v2.1 and is now v1.0. The category (i.e., advance or production) of the datasheet did not change. The only change occurred to the actual numbering of the datasheet. All version numbers are located in the footer of the page.
Publication date	The publication date indicates when the document was published and posted to the Actel website.
The former datasheets were two columns and the current datasheets and handbooks are formatted with one column.	We changed the datasheet format to accommodate the large graphics and to improve readability.

Table 1 • **Differences between Former Datasheets and Device Handbooks (continued)**

Description of Change	Comparison between Handbook and Datasheet
Each chapter within a handbook can have a different version number.	Each chapter in the handbook has its own version number. Since the chapters are independent documents and can be updated at different times, the version numbers for each chapter increment only when a change is made to that chapter. For example, in the ProASIC3L handbook, the DC and Switching section is Advance v0.2, the Packaging Pin Assignments section is v1.0, and the Global Resources section is v1.1. The DC and Switching section is advance because the data has not been fully characterized. The Packaging and Global Resources chapters are both production versions because the data is final. They have different version numbers because they were updated at different times.
Chapters can be numbered differently in each of the handbooks, but the version number for a chapter should be consistent throughout all handbooks. Chapters are also published individually without chapter numbers.	Chapters with shared content are reused across multiple handbooks. The number of chapters in each handbook varies depending on content, so the Pin Descriptions chapter could be chapter 7 in one handbook and chapter 9 in another. This is shown only in the combined handbook version of the document. The content within the chapters and version numbers is the same across each handbook for that chapter. If the chapter is updated, it will be reposted for all associated handbooks.
There are multiple versions of the I/O Structures chapter.	There are several major differences between the families regarding I/O structures, so the information has been grouped by similar families. As a result, we have multiple I/O Structures chapters to describe the features for each group of families in detail.
The part number of the datasheet has changed.	Part numbers are used internally to track documents. Each document has its own unique part number. The number after the second dash indicates the revision of the document. For example, in the part number 51700094-006-1, 51700094-006 is the part number and 1 is the revision of the document. We start with 0 for all new documents. There is a part number for the current version of the datasheet on the back page with the addresses. In addition, all chapters in the datasheet and handbooks have their own unique part numbers. When we implemented the current handbook format, existing documents were assigned new part numbers.
The information contained in the Core Architecture section of the silicon handbook includes information previously found in application notes.	The chapters in the silicon handbook combine application notes that previously had very detailed information about an architectural feature and information from the former datasheets. In addition, because the information was very similar among several of Actel's low-power flash devices, we combined the information into one document. The Supported Families tables describe which devices are supported in the document. The application notes that contained specific architecture information were combined into the handbook and many no longer exist as standalone application notes. Those that do exist in standalone version have been assigned an AC number (top right of first page) to help identify them. The AC number appears in the standalone version and in the handbook chapters where they occur.

Table 1 • **Differences between Former Datasheets and Device Handbooks (continued)**

Description of Change	Comparison between Handbook and Datasheet	
The location of the following information has changed in the current handbook format:		
	Former Datasheet Location	**Current Handbook Location**
CCC/PLL Specification Table	Core Architecture	DC and Switching section > Clock Conditioning Circuits
Peak-to-Peak Jitter Waveform	Core Architecture	DC and Switching section > Clock Conditioning Circuits

Versions

Device handbook chapters may have different version numbers. Actel's goal is to provide customers with the latest information in a timely matter. As a result, the handbook chapters will be updated independently of the handbook.

Categories

In order to provide the latest information to designers, some datasheets are published before data has been fully characterized. Datasheets are designated as "Product Brief," "Advance," "Preliminary," and "Production." The definitions of these categories are as follows:

Product Brief

The product brief is a summarized version of a datasheet (advance or production) and contains general product information. This document gives an overview of specific device and family information.

Advance

This version contains initial estimated information based on simulation, other products, devices, or speed grades. This information can be used as estimates, but not for production. This label only applies to the DC and Switching Characteristics chapter of the datasheet and will only be used when the data has not been fully characterized.

Preliminary

The datasheet contains information based on simulation and/or initial characterization. The information is believed to be correct, but changes are possible.

Unmarked (production)

This version contains information that is considered to be final.

Export Administration Regulations (EAR)

The products described in this document are subject to the Export Administration Regulations (EAR). They could require an approved export license prior to export from the United States. An export includes release of product or disclosure of technology to a foreign national inside or outside the United States.

Part Number and Revision Date

Part Number 51700094-001-2
Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.2)	Page
v1.1 (November 2008)	Table 1 · Differences between Former Datasheets and Device Handbooks was revised to indicate that only the Fusion Handbook retains a Device Architecture section. The information on the I/O Structures chapters was revised, as an I/O Structures chapter has been added for nano devices.	i
v1.0 (January 2008)	This document was rewritten to address the differences between the former datasheets and new device handbooks.	N/A

Rate This Document

Section I – Actel Fusion® Datasheet

Actel
POWER MATTERS

Actel POWER MATTERS

Rate This Document

v2.0

Actel Fusion Family of Mixed-Signal FPGAs

FlashLock

Features and Benefits

High-Performance Reprogrammable Flash Technology

- Advanced 130-nm, 7-Layer Metal, Flash-Based CMOS Process
- Nonvolatile, Retains Program when Powered Off
- Live at Power-Up (LAPU) Single-Chip Solution
- 350 MHz System Performance

Embedded Flash Memory

- User Flash Memory – 2 Mbits to 8 Mbits
  - Configurable 8-, 16-, or 32-Bit Datapath
  - 10 ns Access in Read-Ahead Mode
- 1 Kbit of Additional FlashROM

Integrated A/D Converter (ADC) and Analog I/O

- Up to 12-Bit Resolution and up to 600 Ksps
- Internal 2.56 V or External Reference Voltage
- ADC: Up to 30 Scalable Analog Input Channels
- High-Voltage Input Tolerance: –10.5 V to +12 V
- Current Monitor and Temperature Monitor Blocks
- Up to 10 MOSFET Gate Driver Outputs
  - P- and N-Channel Power MOSFET Support
  - Programmable 1, 3, 10, 30 µA, and 20 mA Drive Strengths
- ADC Accuracy is Better than 1%

On-Chip Clocking Support

- Internal 100 MHz RC Oscillator (accurate to 1%)
- Crystal Oscillator Support (32 KHz to 20 MHz)
- Programmable Real-Time Counter (RTC)
- 6 Clock Conditioning Circuits (CCCs) with 1 or 2 Integrated PLLs
  - Phase Shift, Multiply/Divide, and Delay Capabilities
  - Frequency: Input 1.5–350 MHz, Output 0.75–350 MHz

Low Power Consumption

- Single 3.3 V Power Supply with On-Chip 1.5 V Regulator
- Sleep and Standby Low-Power Modes

In-System Programming (ISP) and Security

- Secure ISP with 128-Bit AES via JTAG
- FlashLock® to Secure FPGA Contents

Advanced Digital I/O

- 1.5 V, 1.8 V, 2.5 V, and 3.3 V Mixed-Voltage Operation
- Bank-Selectable I/O Voltages – Up to 5 Banks per Chip
- Single-Ended I/O Standards: LVTTL, LVCMOS 3.3 V / 2.5 V /1.8 V / 1.5 V, 3.3 V PCI / 3.3 V PCI-X, and LVCMOS 2.5 V / 5.0 V Input
- Differential I/O Standards: LVPECL, LVDS, B-LVDS, M-LVDS
  - Built-In I/O Registers
  - 700 Mbps DDR Operation
- Hot-Swappable I/Os
- Programmable Output Slew Rate, Drive Strength, and Weak Pull-Up/Down Resistor
- Pin-Compatible Packages across the Fusion Family

SRAMs and FIFOs

- Variable-Aspect-Ratio 4,608-Bit SRAM Blocks (×1, ×2, ×4, ×9, and ×18 organizations available)
- True Dual-Port SRAM (except ×18)
- Programmable Embedded FIFO Control Logic

Soft ARM® Cortex™-M1 Fusion Devices (M1)

- ARM Cortex-M1–Enabled (without debug)

Pigeon Point ATCA IP Support (P1)

- Targeted to Actel's Pigeon Point® Board Management Reference (BMR) Starter Kits
- In Partnership with Pigeon Point Systems
- ARM Cortex-M1 Enabled

MicroBlade Advanced Mezzanine Card Support (U1)

- Targeted to Advanced Mezzanine Card (AdvancedMC Designs)
- Designed in Partnership with MicroBlade
- 8051-Based Module Management Controller (MMC)

Fusion Family

Fusion Devices		AFS090	AFS250	AFS600	AFS1500
ARM Cortex-M1* Devices			M1AFS250	M1AFS600	M1AFS1500
Pigeon Point Devices				P1AFS600	P1AFS1500
MicroBlade Devices				U1AFS600	
General Information	System Gates	90,000	250,000	600,000	1,500,000
	Tiles (D-flip-flops)	2,304	6,144	13,824	38,400
	Secure (AES) ISP	Yes	Yes	Yes	Yes
	PLLs	1	1	2	2
	Globals	18	18	18	18
Memory	Flash Memory Blocks (2 Mbits)	1	1	2	4
	Total Flash Memory Bits	2M	2M	4M	8M
	FlashROM Bits	1,024	1,024	1,024	1,024
	RAM Blocks (4,608 bits)	6	8	24	60
	RAM kbits	27	36	108	270
Analog and I/Os	Analog Quads	5	6	10	10
	Analog Input Channels	15	18	30	30
	Gate Driver Outputs	5	6	10	10
	I/O Banks (+ JTAG)	4	4	5	5
	Maximum Digital I/Os	75	114	172	252
	Analog I/Os	20	24	40	40

*Note: *Refer to the Cortex-M1 product brief for more information.*

© 2009 Actel Corporation

Fusion Device Architecture Overview

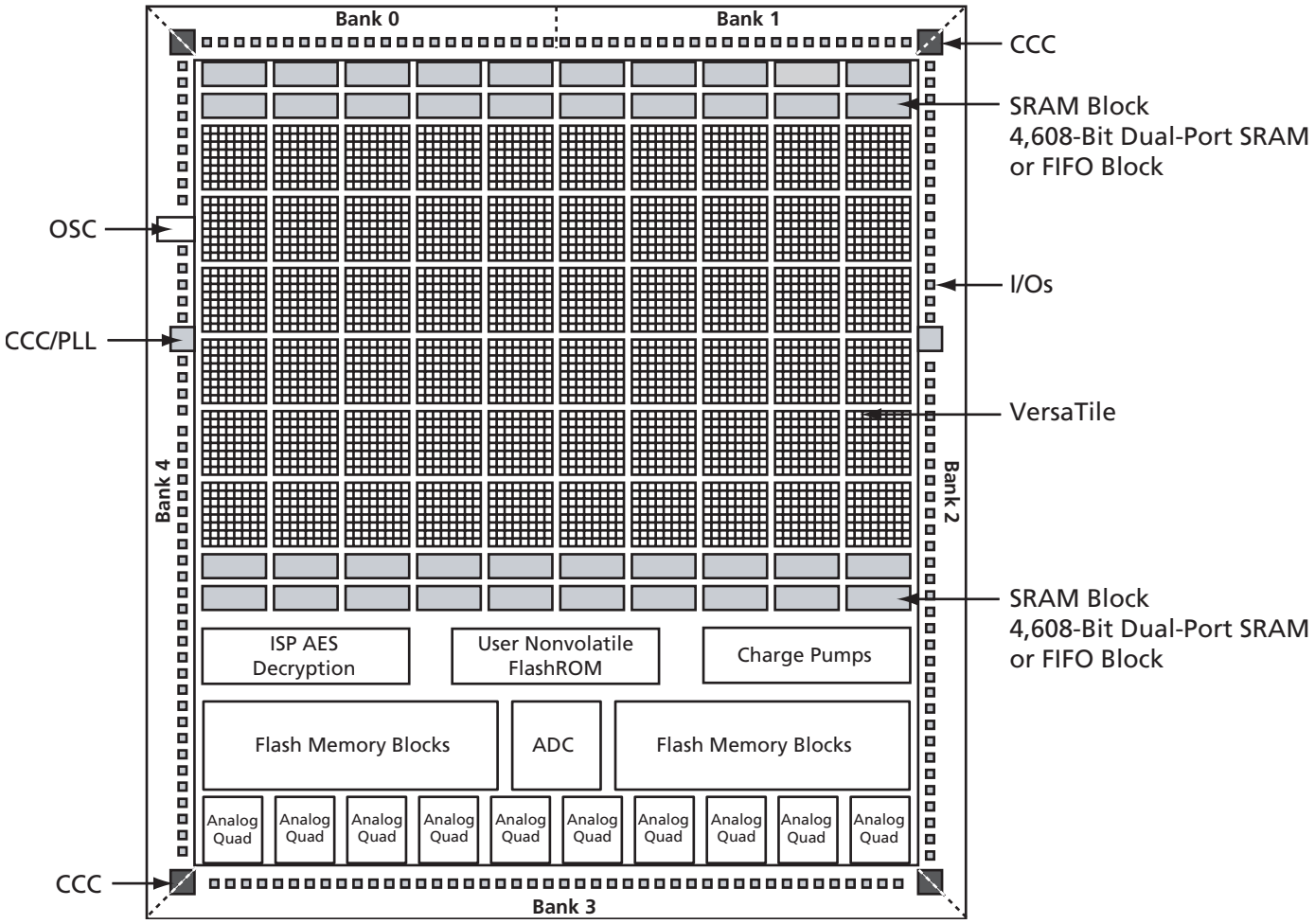


Figure 1-1 • **Fusion Device Architecture Overview (AFS600)**

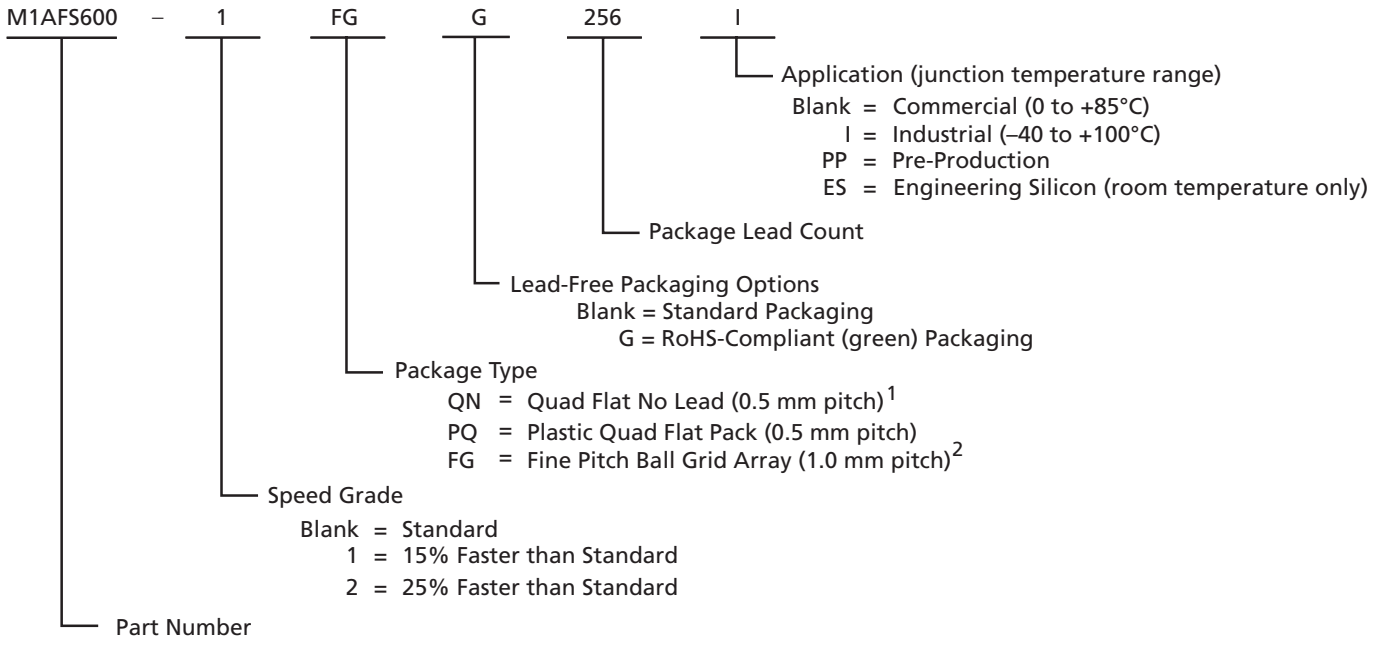
Package I/Os: Single-/Double-Ended (Analog)

Fusion Devices	AFS090	AFS250	AFS600	AFS1500
ARM Cortex-M1 Devices		**M1AFS250**	**M1AFS600**	**M1AFS1500**
Pigeon Point Devices			**P1AFS600 [2, 3]**	**P1AFS1500 [2, 3]**
MicroBlade Devices			**U1AFS600 [2]**	
QN108	37/9 (16)			
QN180	60/16 (20)	65/15 (24)		
PQ208 [1]		93/26 (24)	95/46 (40)	
FG256	75/22 (20)	114/37 (24)	119/58 (40)	119/58 (40)
FG484			172/86 (40)	223/109 (40)
FG676				252/126 (40)

Notes:

1. *Fusion devices in the same package are pin compatible with the exception of the PQ208 package (AFS250 and AFS600).*
2. *MicroBlade devices are only offered in FG256.*
3. *Pigeon Point devices are only offered in FG484 and FG256.*

Product Ordering Codes



Fusion Devices

AFS090 = 90,000 System Gates
AFS250 = 250,000 System Gates
AFS600 = 600,000 System Gates
AFS1500 = 1,500,000 System Gates

ARM-Enabled Fusion Devices

M1AFS250 = 250,000 System Gates
M1AFS600 = 600,000 System Gates
M1AFS1500 = 1,500,000 System Gates

Pigeon Point Devices

P1AFS600 = 600,000 System Gates
P1AFS1500 = 1,500,000 System Gates

MicroBlade Devices

U1AFS600 = 600,000 System Gates

Notes:

1. *For Fusion devices, Quad Flat No Lead packages are only offered as RoHS compliant, QNG packages.*
2. *MicroBlade and Pigeon Point devices only support FG packages.*

Temperature Grade Offerings

Fusion Devices	AFS090	AFS250	AFS600	AFS1500
ARM Cortex-M1 Devices		M1AFS250	M1AFS600	M1AFS1500
Pigeon Point Devices			P1AFS600 [3]	P1AFS1500 [3]
MicroBlade Devices			U1AFS600 [4]	
QN108	C, I	–	–	–
QN180	C, I	C, I	–	–
PQ208	–	C, I	C, I	–
FG256	C, I	C, I	C, I	C, I
FG484	–	–	C, I	C, I
FG676	–	–	–	C, I

Notes:
1. *C = Commercial Temperature Range: 0°C to 85°C Junction*
2. *I = Industrial Temperature Range: –40°C to 100°C Junction*
3. *Pigeon Point devices are only offered in FG484 and FG256.*
4. *MicroBlade devices are only offered in FG256.*

Speed Grade and Temperature Grade Matrix

	Std.[1]	–1	–2[2]
C[3]	✓	✓	✓
I[4]	✓	✓	✓

Notes:
1. *MicroBlade devices are only offered in standard speed grade.*
2. *Pigeon Point devices are only offered in –2 speed grade.*
3. *C = Commercial Temperature Range: 0°C to 85°C Junction*
4. *I = Industrial Temperature Range: –40°C to 100°C Junction*

Contact your local Actel representative for device availability (http://www.actel.com/contact/offices/index.html).

Cortex-M1, Pigeon Point, and MicroBlade Fusion Device Information

This datasheet provides information for all Fusion (AFS), Cortex-M1 (M1), Pigeon Point (P1), and MicroBlade (U1) devices. The remainder of the document will only list the Fusion (AFS) devices. Please apply relevant information to M1, P1, and U1 devices when appropriate. Please note the following:

- Cortex-M1 devices are offered in the same speed grades and packages as basic Fusion devices.
- Pigeon Point devices are only offered in –2 speed grade and FG484 and FG256 packages.
- MicroBlade devices are only offered in standard speed grade and the FG256 package.

1 – Fusion Device Family Overview

Introduction

The Actel Fusion® mixed-signal FPGA satisfies the demand from system architects for a device that simplifies design and unleashes their creativity. As the world's first mixed-signal programmable logic family, Fusion integrates mixed-signal analog, flash memory, and FPGA fabric in a monolithic device. Actel Fusion devices enable designers to quickly move from concept to completed design and then deliver feature-rich systems to market. This new technology takes advantage of the unique properties of Actel flash-based FPGAs, including a high-isolation, triple-well process and the ability to support high-voltage transistors to meet the demanding requirements of mixed-signal system design.

Actel Fusion mixed-signal FPGAs bring the benefits of programmable logic to many application areas, including power management, smart battery charging, clock generation and management, and motor control. Until now, these applications have only been implemented with costly and space-consuming discrete analog components or mixed-signal ASIC solutions. Actel Fusion mixed-signal FPGAs present new capabilities for system development by allowing designers to integrate a wide range of functionality into a single device, while at the same time offering the flexibility of upgrades late in the manufacturing process or after the device is in the field. Actel Fusion devices provide an excellent alternative to costly and time-consuming mixed-signal ASIC designs. In addition, when used in conjunction with the Actel Cortex-M1, Actel Fusion technology represents the definitive mixed-signal FPGA platform.

Flash-based Fusion devices are live at power-up. As soon as system power is applied and within normal operating specifications, Fusion devices are working. Fusion devices have a 128-bit flash-based lock and industry-leading AES decryption, used to secure programmed intellectual property (IP) and configuration data. Actel Fusion devices are the most comprehensive single-chip analog and digital programmable logic solution available today.

To support this new ground-breaking technology, Actel has developed a series of major tool innovations to help maximize designer productivity. Implemented as extensions to the popular Actel Libero® Integrated Design Environment (IDE), these new tools allow designers to easily instantiate and configure peripherals within a design, establish links between peripherals, create or import building blocks or reference designs, and perform hardware verification. This tool suite will also add comprehensive hardware/software debug capability as well as a suite of utilities to simplify development of embedded soft-processor-based solutions.

General Description

The Actel Fusion family, based on the highly successful ProASIC®3 and ProASIC3E flash FPGA architecture, has been designed as a high-performance, programmable, mixed-signal platform. By combining an advanced flash FPGA core with flash memory blocks and analog peripherals, Fusion devices dramatically simplify system design and, as a result, dramatically reduce overall system cost and board space.

The state-of-the-art flash memory technology offers high-density integrated flash memory blocks, enabling savings in cost, power, and board area relative to external flash solutions, while providing increased flexibility and performance. The flash memory blocks and integrated analog peripherals enable true mixed-mode programmable logic designs. Two examples are using an on-chip soft processor to implement a fully functional flash MCU and using high-speed FPGA logic to offer system and power supervisory capabilities. Live at power-up and capable of operating from a single 3.3 V supply, the Fusion family is ideally suited for system management and control applications.

The devices in the Fusion family are categorized by FPGA core density. Each family member contains many peripherals, including flash memory blocks, an analog-to-digital-converter (ADC), high-drive outputs, both RC and crystal oscillators, and a real-time counter (RTC). This provides the

user with a high level of flexibility and integration to support a wide variety of mixed-signal applications. The flash memory block capacity ranges from 2 Mbits to 8 Mbits. The integrated 12-bit ADC supports up to 30 independently configurable input channels. The on-chip crystal and RC oscillators work in conjunction with the integrated phase-locked loops (PLLs) to provide clocking support to the FPGA array and on-chip resources. In addition to supporting typical RTC uses such as watchdog timer, the Fusion RTC can control the on-chip voltage regulator to power down the device (FPGA fabric, flash memory block, and ADC), enabling a low-power standby mode.

The Actel Fusion family offers revolutionary features, never before available in an FPGA. The nonvolatile flash technology gives the Fusion solution the advantage of being a secure, low-power, single-chip solution that is live at power-up. Fusion is reprogrammable and offers time-to-market benefits at an ASIC-level unit cost. These features enable designers to create high-density systems using existing ASIC or FPGA design flows and tools.

Flash Advantages

Reduced Cost of Ownership

Advantages to the designer extend beyond low unit cost, high performance, and ease of use. Flash-based Fusion devices are live at power-up and do not need to be loaded from an external boot PROM. On-board security mechanisms prevent access to the programming information and enable secure remote updates of the FPGA logic. Designers can perform secure remote in-system reprogramming to support future design iterations and field upgrades, with confidence that valuable IP cannot be compromised or copied. Secure ISP can be performed using the industry-standard AES algorithm with MAC data authentication on the device. The Fusion family device architecture mitigates the need for ASIC migration at higher user volumes. This makes the Fusion family a cost-effective ASIC replacement solution for applications in the consumer, networking and communications, computing, and avionics markets.

Security

As the nonvolatile, flash-based Fusion family requires no boot PROM, there is no vulnerable external bitstream. Fusion devices incorporate FlashLock, which provides a unique combination of reprogrammability and design security without external overhead, advantages that only an FPGA with nonvolatile flash programming can offer.

Fusion devices utilize a 128-bit flash-based key lock and a separate AES key to secure programmed IP and configuration data. The FlashROM data in Fusion devices can also be encrypted prior to loading. Additionally, the flash memory blocks can be programmed during runtime using the industry-leading AES-128 block cipher encryption standard (FIPS Publication 192). The AES standard was adopted by the National Institute of Standards and Technology (NIST) in 2000 and replaces the DES standard, which was adopted in 1977. Fusion devices have a built-in AES decryption engine and a flash-based AES key that make Fusion devices the most comprehensive programmable logic device security solution available today. Fusion devices with AES-based security allow for secure remote field updates over public networks, such as the Internet, and ensure that valuable IP remains out of the hands of system overbuilders, system cloners, and IP thieves. As an additional security measure, the FPGA configuration data of a programmed Fusion device cannot be read back, although secure design verification is possible. During design, the user controls and defines both internal and external access to the flash memory blocks.

Security, built into the FPGA fabric, is an inherent component of the Fusion family. The flash cells are located beneath seven metal layers, and many device design and layout techniques have been used to make invasive attacks extremely difficult. Fusion with FlashLock and AES security is unique in being highly resistant to both invasive and noninvasive attacks. Your valuable IP is protected, making secure remote ISP possible. A Fusion device provides the most impenetrable security for programmable logic designs.

Single Chip

Flash-based FPGAs store their configuration information in on-chip flash cells. Once programmed, the configuration data is an inherent part of the FPGA structure, and no external configuration data needs to be loaded at system power-up (unlike SRAM-based FPGAs). Therefore, flash-based Fusion FPGAs do not require system configuration components such as EEPROMs or

microcontrollers to load device configuration data. This reduces bill-of-materials costs and PCB area, and increases security and system reliability.

Live at Power-Up

Flash-based Fusion devices are Level 0 live at power-up (LAPU). LAPU Fusion devices greatly simplify total system design and reduce total system cost by eliminating the need for CPLDs. The Fusion LAPU clocking (PLLs) replaces off-chip clocking resources. The Fusion mix of LAPU clocking and analog resources makes these devices an excellent choice for both system supervisor and system management functions. LAPU from a single 3.3 V source enables Fusion devices to initiate, control, and monitor multiple voltage supplies while also providing system clocks. In addition, glitches and brownouts in system power will not corrupt the Fusion device flash configuration. Unlike SRAM-based FPGAs, the device will not have to be reloaded when system power is restored. This enables reduction or complete removal of expensive voltage monitor and brownout detection devices from the PCB design. Flash-based Fusion devices simplify total system design and reduce cost and design risk, while increasing system reliability.

Firm Errors

Firm errors occur most commonly when high-energy neutrons, generated in the upper atmosphere, strike a configuration cell of an SRAM FPGA. The energy of the collision can change the state of the configuration cell and thus change the logic, routing, or I/O behavior in an unpredictable way. Another source of radiation-induced firm errors is alpha particles. For an alpha to cause a soft or firm error, its source must be in very close proximity to the affected circuit. The alpha source must be in the package molding compound or in the die itself. While low-alpha molding compounds are being used increasingly, this helps reduce but does not entirely eliminate alpha-induced firm errors.

Firm errors are impossible to prevent in SRAM FPGAs. The consequence of this type of error can be a complete system failure. Firm errors do not occur in Fusion flash-based FPGAs. Once it is programmed, the flash cell configuration element of Fusion FPGAs cannot be altered by high-energy neutrons and is therefore immune to errors from them.

Recoverable (or soft) errors occur in the user data SRAMs of all FPGA devices. These can easily be mitigated by using error detection and correction (EDAC) circuitry built into the FPGA fabric.

Low Power

Flash-based Fusion devices exhibit power characteristics similar to those of an ASIC, making them an ideal choice for power-sensitive applications. With Fusion devices, there is no power-on current surge and no high current transition, both of which occur on many FPGAs.

Fusion devices also have low dynamic power consumption and support both low power standby mode and very low power sleep mode, offering further power savings.

Advanced Flash Technology

The Fusion family offers many benefits, including nonvolatility and reprogrammability through an advanced flash-based, 130-nm LVCMOS process with seven layers of metal. Standard CMOS design techniques are used to implement logic and control functions. The combination of fine granularity, enhanced flexible routing resources, and abundant flash switches allows very high logic utilization (much higher than competing SRAM technologies) without compromising device routability or performance. Logic functions within the device are interconnected through a four-level routing hierarchy.

Advanced Architecture

The proprietary Fusion architecture provides granularity comparable to standard-cell ASICs. The Fusion device consists of several distinct and programmable architectural features, including the following (Figure 1-1 on page 1-5):

- Embedded memories
  - Flash memory blocks
  - FlashROM

  - SRAM and FIFO
- Clocking resources
  - PLL and CCC
  - RC oscillator
  - Crystal oscillator
  - No-Glitch MUX (NGMUX)
- Digital I/Os with advanced I/O standards
- FPGA VersaTiles
- Analog components
  - ADC
  - Analog I/Os supporting voltage, current, and temperature monitoring
  - 1.5 V on-board voltage regulator
  - Real-time counter

The FPGA core consists of a sea of VersaTiles. Each VersaTile can be configured as a three-input logic lookup table (LUT) equivalent or a D-flip-flop or latch (with or without enable) by programming the appropriate flash switch interconnections. This versatility allows efficient use of the FPGA fabric. The VersaTile capability is unique to the Actel families of flash-based FPGAs. VersaTiles and larger functions are connected with any of the four levels of routing hierarchy. Flash switches are distributed throughout the device to provide nonvolatile, reconfigurable interconnect programming. Maximum core utilization is possible for virtually any design.

In addition, extensive on-chip programming circuitry allows for rapid (3.3 V) single-voltage programming of Fusion devices via an IEEE 1532 JTAG interface.

Unprecedented Integration

Integrated Analog Blocks and Analog I/Os

Fusion devices offer robust and flexible analog mixed-signal capability in addition to the high-performance flash FPGA fabric and flash memory block. The many built-in analog peripherals include a configurable 32:1 input analog MUX, up to 10 independent MOSFET gate driver outputs, and a configurable ADC. The ADC supports 8-, 10-, and 12-bit modes of operation with a cumulative sample rate up to 600 k samples per second (Ksps), differential nonlinearity (DNL) < 1.0 LSB, and Total Unadjusted Error (TUE) of 0.72 LSB in 10-bit mode. The TUE is used for characterization of the conversion error and includes errors from all sources, such as offset and linearity. Internal bandgap circuitry offers 1% voltage reference accuracy with the flexibility of utilizing an external reference voltage. The ADC channel sampling sequence and sampling rate are programmable and implemented in the FPGA logic using Designer and Libero IDE software tool support.

Two channels of the 32-channel ADCMUX are dedicated. Channel 0 is connected internally to V_{CC} and can be used to monitor core power supply. Channel 31 is connected to an internal temperature diode which can be used to monitor device temperature. The 30 remaining channels can be connected to external analog signals. The exact number of I/Os available for external connection signals is device-dependent (refer to the "Fusion Family" table on page I for details).

With Fusion, Actel also introduces the Analog Quad I/O structure (Figure 1-1 on page 1-5). Each quad consists of three analog inputs and one gate driver. Each quad can be configured in various built-in circuit combinations, such as three prescaler circuits, three digital input circuits, a current monitor circuit, or a temperature monitor circuit. Each prescaler has multiple scaling factors programmed by FPGA signals to support a large range of analog inputs with positive or negative polarity. When the current monitor circuit is selected, two adjacent analog inputs measure the voltage drop across a small external sense resistor. For more information, refer to the "Analog System Characteristics" section in the *Device Architecture* chapter of the datasheet for more information. Built-in operational amplifiers amplify small voltage signals for accurate current measurement. One analog input in each quad can be connected to an external temperature

monitor diode. In addition to the external temperature monitor diode(s), a Fusion device can monitor an internal temperature diode using dedicated channel 31 of the ADCMUX.

Figure 1-1 on page 1-5 illustrates a typical use of the Analog Quad I/O structure. The Analog Quad shown is configured to monitor and control an external power supply. The AV pad measures the source of the power supply. The AC pad measures the voltage drop across an external sense resistor to calculate current. The AG MOSFET gate driver pad turns the external MOSFET on and off. The AT pad measures the load-side voltage level.

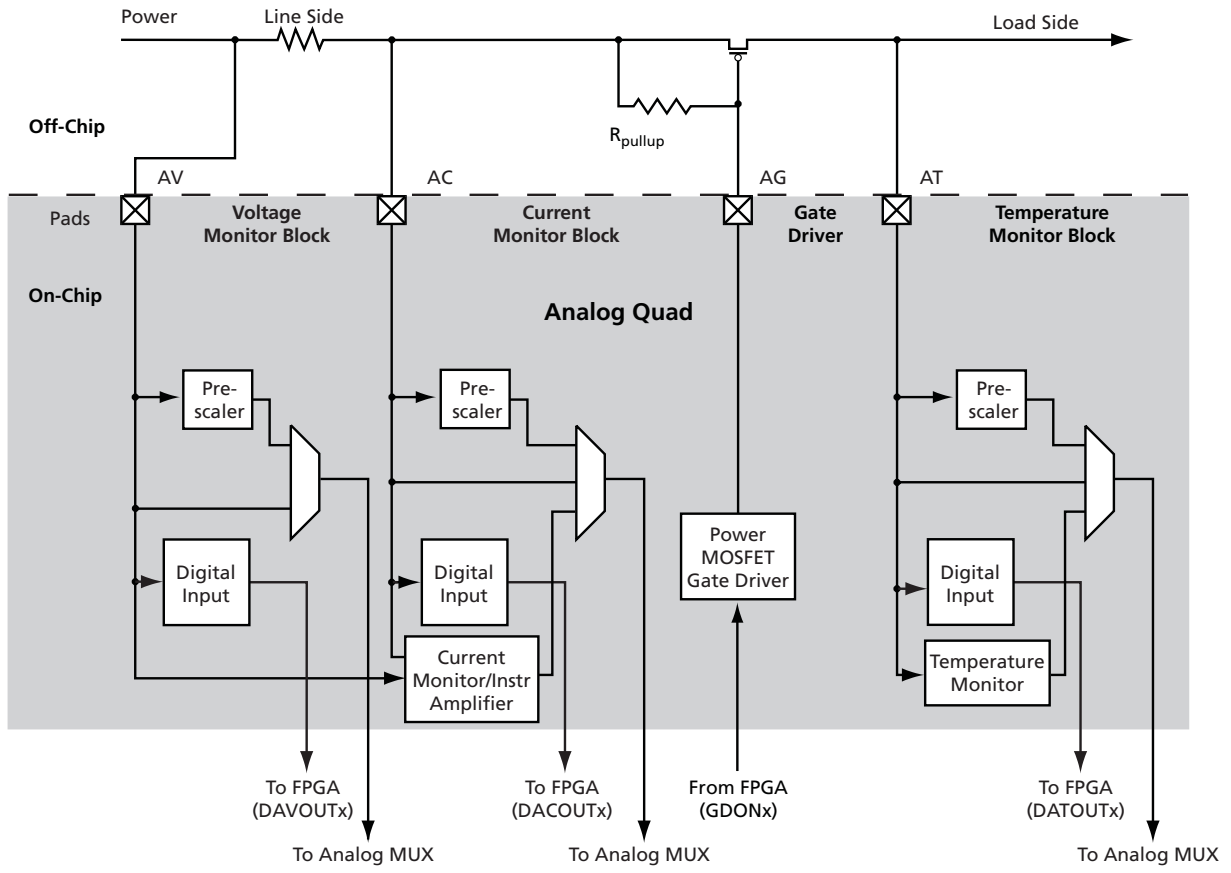


Figure 1-1 • **Analog Quad**

Embedded Memories

Flash Memory Blocks

The flash memory available in each Fusion device is composed of one to four flash blocks, each 2 Mbits in density. Each block operates independently with a dedicated flash controller and interface. Fusion flash memory blocks combine fast access times (60 ns random access and 10 ns access in Read-Ahead mode) with a configurable 8-, 16-, or 32-bit datapath, enabling high-speed flash operation without wait states. The memory block is organized in pages and sectors. Each page has 128 bytes, with 33 pages comprising one sector and 64 sectors per block. The flash block can support multiple partitions. The only constraint on size is that partition boundaries must coincide with page boundaries. The flexibility and granularity enable many use models and allow added granularity in programming updates.

Fusion devices support two methods of external access to the flash memory blocks. The first method is a serial interface that features a built-in JTAG-compliant port, which allows in-system programmability during user or monitor/test modes. This serial interface supports programming of

an AES-encrypted stream. Secure data can be passed through the JTAG interface, decrypted, and then programmed in the flash block. The second method is a soft parallel interface.

FPGA logic or an on-chip soft microprocessor can access flash memory through the parallel interface. Since the flash parallel interface is implemented in the FPGA fabric, it can potentially be customized to meet special user requirements. For more information, refer to the *CoreCFI Handbook*. The flash memory parallel interface provides configurable byte-wide (×8), word-wide (×16), or dual-word-wide (×32) data port options. Through the programmable flash parallel interface, the on-chip and off-chip memories can be cascaded for wider or deeper configurations.

The flash memory has built-in security. The user can configure either the entire flash block or the small blocks to prevent unintentional or intrusive attempts to change or destroy the storage contents. Each on-chip flash memory block has a dedicated controller, enabling each block to operate independently.

The flash block logic consists of the following sub-blocks:

- Flash block – Contains all stored data. The flash block contains 64 sectors and each sector contains 33 pages of data.
- Page Buffer – Contains the contents of the current page being modified. A page contains 8 blocks of data.
- Block Buffer – Contains the contents of the last block accessed. A block contains 128 data bits.
- ECC Logic – The flash memory stores error correction information with each block to perform single-bit error correction and double-bit error detection on all data blocks.

User Nonvolatile FlashROM

In addition to the flash blocks, Actel Fusion devices have 1 Kbit of user-accessible, nonvolatile FlashROM on-chip. The FlashROM is organized as 8×128-bit pages. The FlashROM can be used in diverse system applications:

- Internet protocol addressing (wireless or fixed)
- System calibration settings
- Device serialization and/or inventory control
- Subscription-based business models (for example, set-top boxes)
- Secure key storage for secure communications algorithms
- Asset management/tracking
- Date stamping
- Version management

The FlashROM is written using the standard IEEE 1532 JTAG programming interface. Pages can be individually programmed (erased and written). On-chip AES decryption can be used selectively over public networks to securely load data such as security keys stored in the FlashROM for a user design.

The FlashROM can be programmed (erased and written) via the JTAG programming interface, and its contents can be read back either through the JTAG programming interface or via direct FPGA core addressing.

The FlashPoint tool in the Actel Fusion development software solutions, Libero IDE and Designer, has extensive support for flash memory blocks and FlashROM. One such feature is auto-generation of sequential programming files for applications requiring a unique serial number in each part. Another feature allows the inclusion of static data for system version control. Data for the FlashROM can be generated quickly and easily using the Actel Libero IDE and Designer software tools. Comprehensive programming file support is also included to allow for easy programming of large numbers of parts with differing FlashROM contents.

SRAM and FIFO

Fusion devices have embedded SRAM blocks along the north and south sides of the device. Each variable-aspect-ratio SRAM block is 4,608 bits in size. Available memory configurations are 256×18, 512×9, 1k×4, 2k×2, and 4k×1 bits. The individual blocks have independent read and write ports that can be configured with different bit widths on each port. For example, data can be written

through a 4-bit port and read as a single bitstream. The SRAM blocks can be initialized from the flash memory blocks or via the device JTAG port (ROM emulation mode), using the UJTAG macro.

In addition, every SRAM block has an embedded FIFO control unit. The control unit allows the SRAM block to be configured as a synchronous FIFO without using additional core VersaTiles. The FIFO width and depth are programmable. The FIFO also features programmable Almost Empty (AEMPTY) and Almost Full (AFULL) flags in addition to the normal EMPTY and FULL flags. The embedded FIFO control unit contains the counters necessary for the generation of the read and write address pointers. The SRAM/FIFO blocks can be cascaded to create larger configurations.

Clock Resources

PLLs and Clock Conditioning Circuits (CCCs)

Fusion devices provide designers with very flexible clock conditioning capabilities. Each member of the Fusion family contains six CCCs. In the two larger family members, two of these CCCs also include a PLL; the smaller devices support one PLL.

The inputs of the CCC blocks are accessible from the FPGA core or from one of several inputs with dedicated CCC block connections.

The CCC block has the following key features:

- Wide input frequency range (f_{IN_CCC}) = 1.5 MHz to 350 MHz
- Output frequency range (f_{OUT_CCC}) = 0.75 MHz to 350 MHz
- Clock phase adjustment via programmable and fixed delays from –6.275 ns to +8.75 ns
- Clock skew minimization (PLL)
- Clock frequency synthesis (PLL)
- On-chip analog clocking resources usable as inputs:
  - 100 MHz on-chip RC oscillator
  - Crystal oscillator

Additional CCC specifications:

- Internal phase shift = 0°, 90°, 180°, and 270°
- Output duty cycle = 50% ± 1.5%
- Low output jitter. Samples of peak-to-peak period jitter when a single global network is used:
  - 70 ps at 350 MHz
  - 90 ps at 100 MHz
  - 180 ps at 24 MHz
  - Worst case < 2.5% × clock period
- Maximum acquisition time = 150 µs
- Low power consumption of 5 mW

Global Clocking

Fusion devices have extensive support for multiple clocking domains. In addition to the CCC and PLL support described above, there are on-chip oscillators as well as a comprehensive global clock distribution network.

The integrated RC oscillator generates a 100 MHz clock. It is used internally to provide a known clock source to the flash memory read and write control. It can also be used as a source for the PLLs.

The crystal oscillator supports the following operating modes:

- Crystal (32.768 KHz to 20 MHz)
- Ceramic (500 KHz to 8 MHz)
- RC (32.768 KHz to 4 MHz)

Each VersaTile input and output port has access to nine VersaNets: six main and three quadrant global networks. The VersaNets can be driven by the CCC or directly accessed from the core via

MUXes. The VersaNets can be used to distribute low-skew clock signals or for rapid distribution of high-fanout nets.

Digital I/Os with Advanced I/O Standards

The Fusion family of FPGAs features a flexible digital I/O structure, supporting a range of voltages (1.5 V, 1.8 V, 2.5 V, and 3.3 V). Fusion FPGAs support many different digital I/O standards, both single-ended and differential.

The I/Os are organized into banks, with four or five banks per device. The configuration of these banks determines the I/O standards supported. The banks along the east and west sides of the device support the full range of I/O standards (single-ended and differential). The south bank supports the Analog Quads (analog I/O). In the family's two smaller devices, the north bank supports multiple single-ended digital I/O standards. In the family's larger devices, the north bank is divided into two banks of digital Pro I/Os, supporting a wide variety of single-ended, differential, and voltage-referenced I/O standards.

Each I/O module contains several input, output, and enable registers. These registers allow the implementation of the following applications:

- Single-Data-Rate (SDR) applications
- Double-Data-Rate (DDR) applications—DDR LVDS I/O for chip-to-chip communications
- Fusion banks support LVPECL, LVDS, BLVDS, and M-LVDS with 20 multi-drop points.

VersaTiles

The Fusion core consists of VersaTiles, which are also used in the successful Actel ProASIC3 family. The Fusion VersaTile supports the following:

- All 3-input logic functions—LUT-3 equivalent
- Latch with clear or set
- D-flip-flop with clear or set and optional enable

Refer to Figure 1-2 for the VersaTile configuration arrangement.

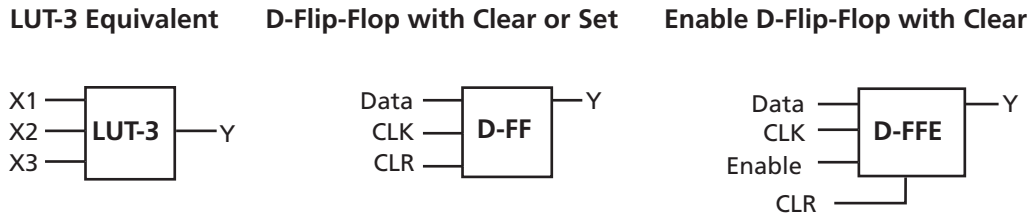


Figure 1-2 • **VersaTile Configurations**

Related Documents

Datasheet

Core8051
www.actel.com/ipdocs/Core8051_DS.pdf

Application Notes

Fusion FlashROM
http://www.actel.com/documents/Fusion_FROM_AN.pdf
Fusion SRAM/FIFO Blocks
http://www.actel.com/documents/Fusion_RAM_FIFO_AN.pdf
Using DDR in Fusion Devices
http://www.actel.com/documents/Fusion_DDR_AN.pdf
Fusion Security
http://www.actel.com/documents/Fusion_Security_AN.pdf
Using Fusion RAM as Multipliers
http://www.actel.com/documents/Fusion_Multipliers_AN.pdf

Handbook

Cortex-M1 Handbook
www.actel.com/documents/CortexM1_HB.pdf
Fusion Handbook
http://www.actel.com/documents/Fusion_HB.pdf
Prototyping with AFS600 for Smaller Devices
http://www.actel.com/documents/Fusion_Prototyp_HBs.pdf
UJTAG Applications in Actel's Low-Power Flash Devices
http://www.actel.com/documents/LPD_UJTAG_HBs.pdf
In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3
http://www.actel.com/documents/LPD_ISP_HBs.pdf

User's Guides

Designer User's Guide
http://www.actel.com/documents/designer_UG.pdf
Fusion, IGLOOe and ProASIC3/E Macro Library Guide
http://www.actel.com/documents/pa3_libguide_ug.pdf
SmartGen, FlashROM, Flash Memory System Builder, and Analog System Builder User's Guide
http://www.actel.com/documents/genguide_ug.pdf

White Papers

Fusion Technology
http://www.actel.com/documents/Fusion_Tech_WP.pdf

Part Number and Revision Date

Part Number 51700092-013-1
Revised July 2009

List of Changes

The following table lists critical changes that were made in the current version of the document.

Previous Version	Changes in Current Version (v2.0)	Page
Preliminary v1.7 (October 2008)	The MicroBlade and Fusion datasheets have been combined. Pigeon Point information is new. CoreMP7 support was removed since it is no longer offered. –F was removed from the datasheet since it is no longer offered. The operating temperature was changed from ambient to junction to better reflect actual conditions of operations. Commercial: 0°C to 85°C Industrial: –40°C to 100°C The version number category was changed from Preliminary to Production, which means the datasheet contains information based on final characterization. The version number changed from Preliminary v1.7 to v2.0.	N/A
	The "Integrated Analog Blocks and Analog I/Os" section was updated to include a reference to the "Analog System Characteristics" section in the *Device Architecture* chapter of the datasheet, which includes Table 2-46 • Analog Channel Specifications and specific voltage data.	1-4
Advance v1.6 (August 2008)	The version number category was changed from Advance to Preliminary, which means the datasheet contains information based on simulation and/or initial characterization. The information is believed to be correct, but changes are possible.	
Advance v1.4 (July 2008)	The title of the datasheet changed from Actel Programmable System Chips to Actel Fusion Mixed-Signal FPGAs. In addition, all instances of programmable system chip were changed to mixed-signal FPGA.	N/A
Advance v0.9 (October 2007)	The following bullet was updated from High-Voltage Input Tolerance: ±12 V to High-Voltage Input Tolerance: 10.5 V to 12 V.	I
	The following bullet was updated from Programmable 1, 3, 10, 30 µA and 25 mA Drive Strengths to Programmable 1, 3, 10, 30 µA and 20 mA Drive Strengths.	I
	This bullet was added to the "Integrated A/D Converter (ADC) and Analog I/O" section: ADC Accuracy is Better than 1%	I
	In the "Integrated Analog Blocks and Analog I/Os" section, ±4 LSB was changed to 0.72. The following sentence was deleted: The input range for voltage signals is from –12 V to +12 V with full-scale output values from 0.125 V to 16 V. In addition, 2°C was changed to 3°C: "One analog input in each quad can be connected to an external temperature monitor diode and achieves detection accuracy of ±3°C." The following sentence was deleted: The input range for voltage signals is from –12 V to +12 V with full-scale output values from 0.125 V to 16 V.	1-4

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.7 (January 2007)	In the "Package I/Os: Single-/Double-Ended (Analog)" table, the AFS1500/M7AFS1500 I/O counts were updated for the following devices: FG484: 223/109 FG676: 252/126	II
Advance v0.4 (April 2006)	The AFS1500 digital I/O count was updated in the "Fusion Family" table.	I
	The AFS1500 digital I/O count was updated in the "Package I/Os: Single-/Double-Ended (Analog)" table.	II
Advance v0.3 (April 2006)	The G was moved in the "Product Ordering Codes" section.	III
Advance v0.2 (April 2006)	The "Features and Benefits" section was updated.	I
	The "Fusion Family" table was updated.	I
	The "Package I/Os: Single-/Double-Ended (Analog)" table was updated.	II
Advance v0.2 (continued)	The "Product Ordering Codes" table was updated.	III
	The "Temperature Grade Offerings" table was updated.	IV
	The "General Description" section was updated to include ARM information.	1-1

Datasheet Categories

Categories

In order to provide the latest information to designers, some datasheets are published before data has been fully characterized. Datasheets are designated as "Product Brief," "Advance," "Preliminary," and "Production." The definitions of these categories are as follows:

Product Brief

The product brief is a summarized version of a datasheet (advance or production) and contains general product information. This document gives an overview of specific device and family information.

Advance

This version contains initial estimated information based on simulation, other products, devices, or speed grades. This information can be used as estimates, but not for production. This label only applies to the DC and Switching Characteristics chapter of the datasheet and will only be used when the data has not been fully characterized.

Preliminary

The datasheet contains information based on simulation and/or initial characterization. The information is believed to be correct, but changes are possible.

Unmarked (production)

This version contains information that is considered to be final.

Export Administration Regulations (EAR)

The products described in this document are subject to the Export Administration Regulations (EAR). They could require an approved export license prior to export from the United States. An export includes release of product or disclosure of technology to a foreign national inside or outside the United States.

Actel Safety Critical, Life Support, and High-Reliability Applications Policy

The Actel products described in this advance status document may not have completed Actel's qualification process. Actel may amend or enhance products during the product introduction and qualification process, resulting in changes in device functionality or performance. It is the responsibility of each customer to ensure the fitness of any Actel product (but especially a new product) for a particular purpose, including appropriateness for safety-critical, life-support, and other high-reliability applications. Consult Actel's Terms and Conditions for specific liability exclusions relating to life-support applications. A reliability report covering all of Actel's products is available on the Actel website at http://www.actel.com/documents/ORT_Report.pdf. Actel also offers a variety of enhanced qualification and lot acceptance screening procedures. Contact your local Actel sales office for additional reliability information.

2 – Device Architecture

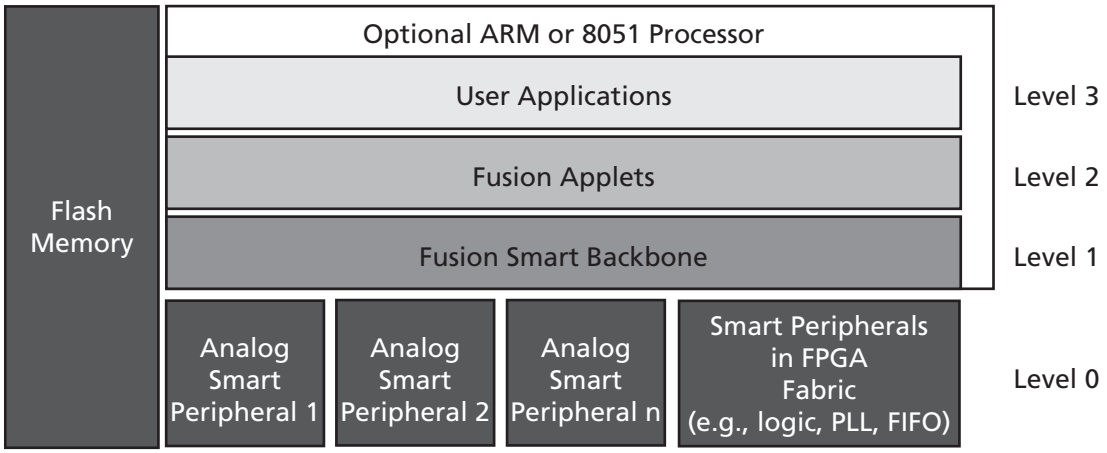
Fusion Stack Architecture

To manage the unprecedented level of integration in Fusion devices, Actel developed the Fusion technology stack (Figure 2-1). This layered model offers a flexible design environment, enabling design at very high and very low levels of abstraction. Fusion peripherals include hard analog IP and hard and soft digital IP. Peripherals communicate across the FPGA fabric via a layer of soft gates—the Fusion backbone. Much more than a common bus interface, this Fusion backbone integrates a micro-sequencer within the FPGA fabric and configures the individual peripherals and supports low-level processing of peripheral data. Fusion applets are application building blocks that can control and respond to peripherals and other system signals. Applets can be rapidly combined to create large applications. The technology is scalable across devices, families, design types, and user expertise, and supports a well-defined interface for external IP and tool integration.

At the lowest level, Level 0, are Fusion peripherals. These are configurable functional blocks that can be hardwired structures such as a PLL or analog input channel, or soft (FPGA gate) blocks such as a UART or two-wire serial interface. The Fusion peripherals are configurable and support a standard interface to facilitate communication and implementation.

Connecting and controlling access to the peripherals is the Fusion backbone, Level 1. The backbone is a soft-gate structure, scalable to any number of peripherals. The backbone is a bus and much more; it manages peripheral configuration to ensure proper operation. Leveraging the common peripheral interface and a low-level state machine, the backbone efficiently offloads peripheral management from the system design. The backbone can set and clear flags based upon peripheral behavior and can define performance criteria. The flexibility of the stack enables a designer to configure the silicon, directly bypassing the backbone if that level of control is desired.

One step up from the backbone is the Fusion applet, Level 2. The applet is an application building block that implements a specific function in FPGA gates. It can react to stimuli and board-level events coming through the backbone or from other sources, and responds to these stimuli by accessing and manipulating peripherals via the backbone or initiating some other action. An applet controls or responds to the peripheral(s). Applets can be easily imported or exported from the design environment. The applet structure is open and well-defined, enabling users to import applets from Actel, system developers, third parties, and user groups.



Note: Levels 1, 2, and 3 are implemented in FPGA logic gates.

Figure 2-1 • **Fusion Architecture Stack**

The system application, Level 3, is the larger user application that utilizes one or more applets. Designing at the highest level of abstraction supported by the Actel Fusion technology stack, the application can be easily created in FPGA gates by importing and configuring multiple applets.

In fact, in some cases an entire FPGA system design can be created without any HDL coding.

An optional MCU enables a combination of software and HDL-based design methodologies. The MCU can be on-chip or off-chip as system requirements dictate. System portioning is very flexible, allowing the MCU to reside above the applets or to absorb applets, or applets and backbone, if desired.

The Actel Fusion technology stack enables a very flexible design environment. Users can engage in design across a continuum of abstraction from very low to very high.

Core Architecture

VersaTile

Based upon successful Actel ProASIC3/E logic architecture, Fusion devices provide granularity comparable to gate arrays. The Fusion device core consists of a sea-of-VersaTiles architecture.

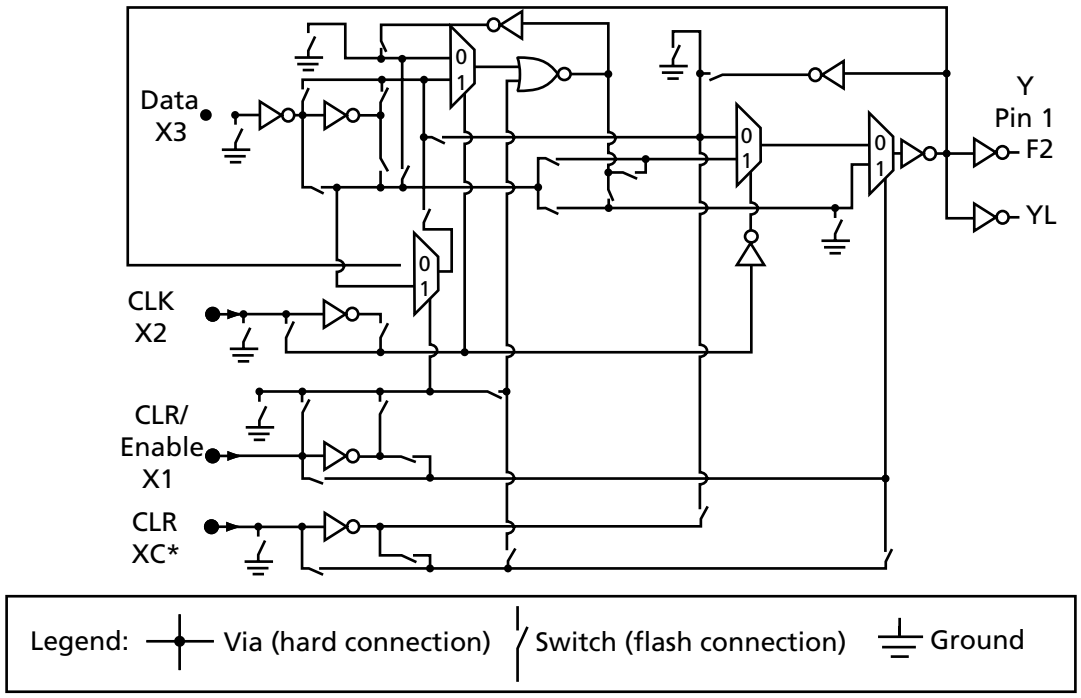
As illustrated in Figure 2-2, there are four inputs in a logic VersaTile cell, and each VersaTile can be configured using the appropriate flash switch connections:

- Any 3-input logic function
- Latch with clear or set
- D-flip-flop with clear or set
- Enable D-flip-flop with clear or set (on a 4th input)

VersaTiles can flexibly map the logic and sequential gates of a design. The inputs of the VersaTile can be inverted (allowing bubble pushing), and the output of the tile can connect to high-speed, very-long-line routing resources. VersaTiles and larger functions are connected with any of the four levels of routing hierarchy.

When the VersaTile is used as an enable D-flip-flop, the SET/CLR signal is supported by a fourth input, which can only be routed to the core cell over the VersaNet (global) network.

The output of the VersaTile is F2 when the connection is to the ultra-fast local lines, or YL when the connection is to the efficient long-line or very-long-line resources (Figure 2-2).



*Note: *This input can only be connected to the global clock distribution network.*

Figure 2-2 • **Fusion Core VersaTile**

VersaTile Characteristics

Sample VersaTile Specifications—Combinatorial Module

The Fusion library offers all combinations of LUT-3 combinatorial functions. In this section, timing characteristics are presented for a sample of the library (Figure 2-3). For more details, refer to the *Fusion, IGLOOe and ProASIC3E Macro Library Guide*.

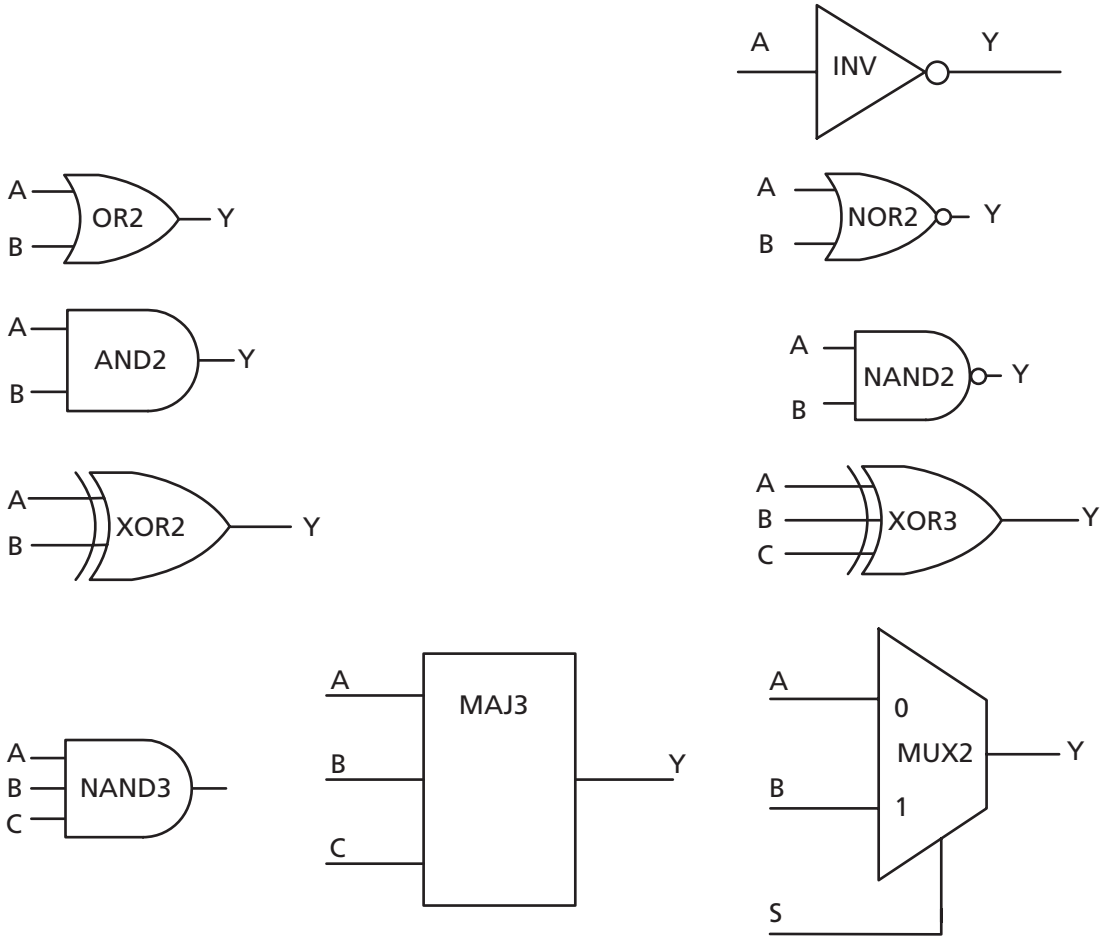


Figure 2-3 • **Sample of Combinatorial Cells**

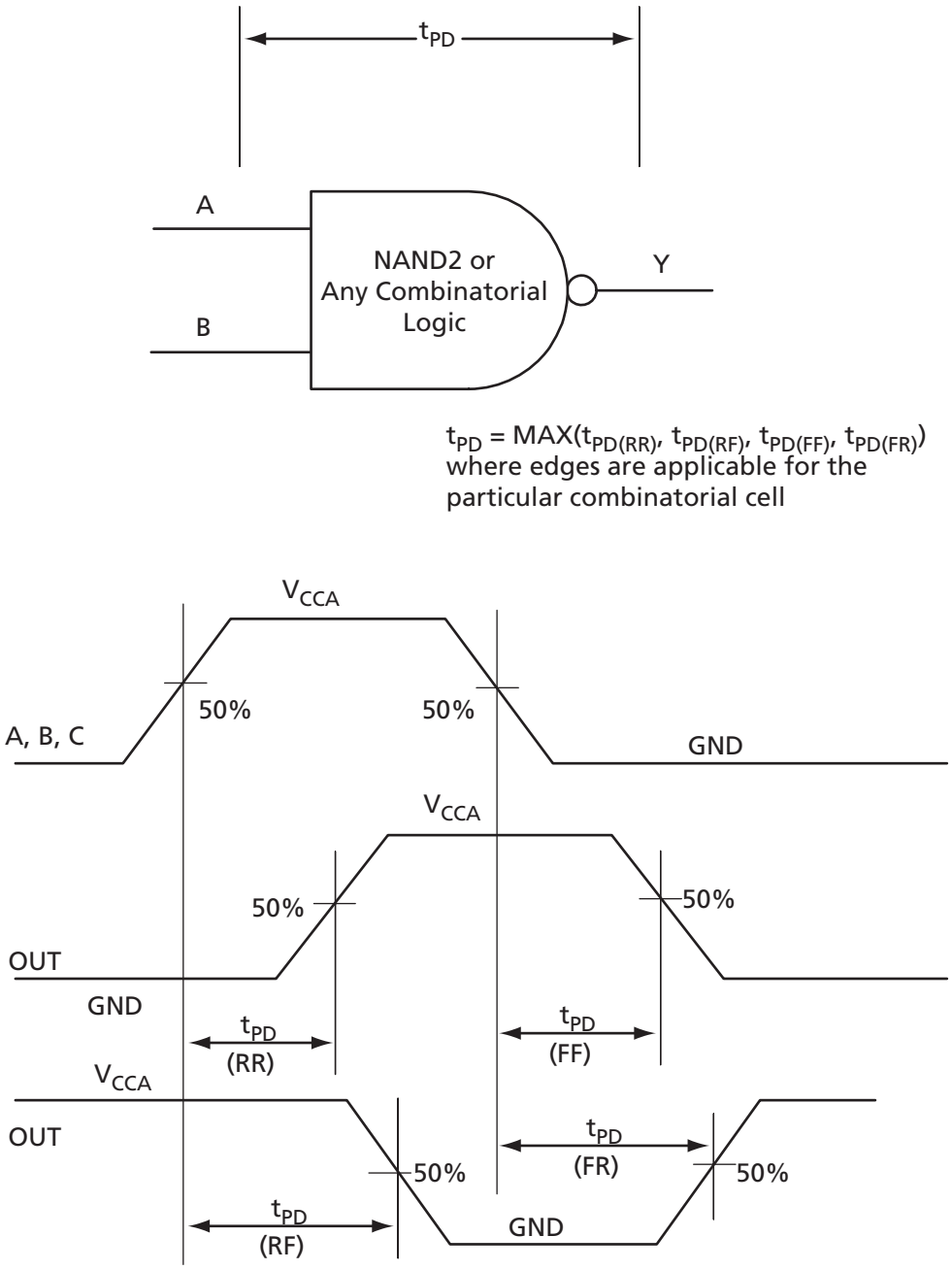


Figure 2-4 • **Combinatorial Timing Model and Waveforms**

Timing Characteristics

Table 2-1 • Combinatorial Cell Propagation Delays
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Combinatorial Cell	Equation	Parameter	–2	–1	Std.	Units
INV	Y = !A	t_{PD}	0.40	0.46	0.54	ns
AND2	Y = A · B	t_{PD}	0.47	0.54	0.63	ns
NAND2	Y = !(A · B)	t_{PD}	0.47	0.54	0.63	ns
OR2	Y = A + B	t_{PD}	0.49	0.55	0.65	ns
NOR2	Y = !(A + B)	t_{PD}	0.49	0.55	0.65	ns
XOR2	Y = A ⊕ B	t_{PD}	0.74	0.84	0.99	ns
MAJ3	Y = MAJ(A, B, C)	t_{PD}	0.70	0.79	0.93	ns
XOR3	Y = A ⊕ B ⊕ C	t_{PD}	0.87	1.00	1.17	ns
MUX2	Y = A !S + B S	t_{PD}	0.51	0.58	0.68	ns
AND3	Y = A · B · C	t_{PD}	0.56	0.64	0.75	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Sample VersaTile Specifications—Sequential Module

The Fusion library offers a wide variety of sequential cells, including flip-flops and latches. Each has a data input and optional enable, clear, or preset. In this section, timing characteristics are presented for a representative sample from the library (Figure 2-5). For more details, refer to the *Fusion, IGLOO/e and ProASIC3/E Macro Library Guide*.

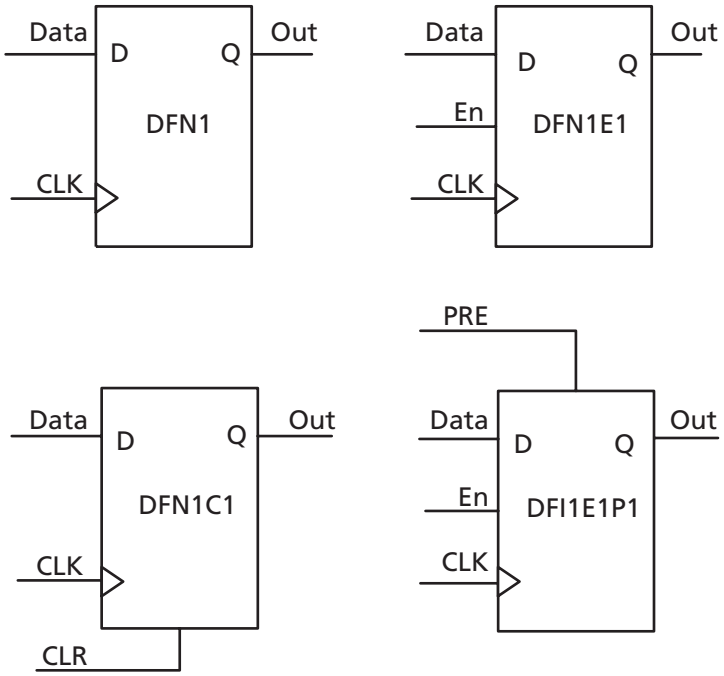


Figure 2-5 • **Sample of Sequential Cells**

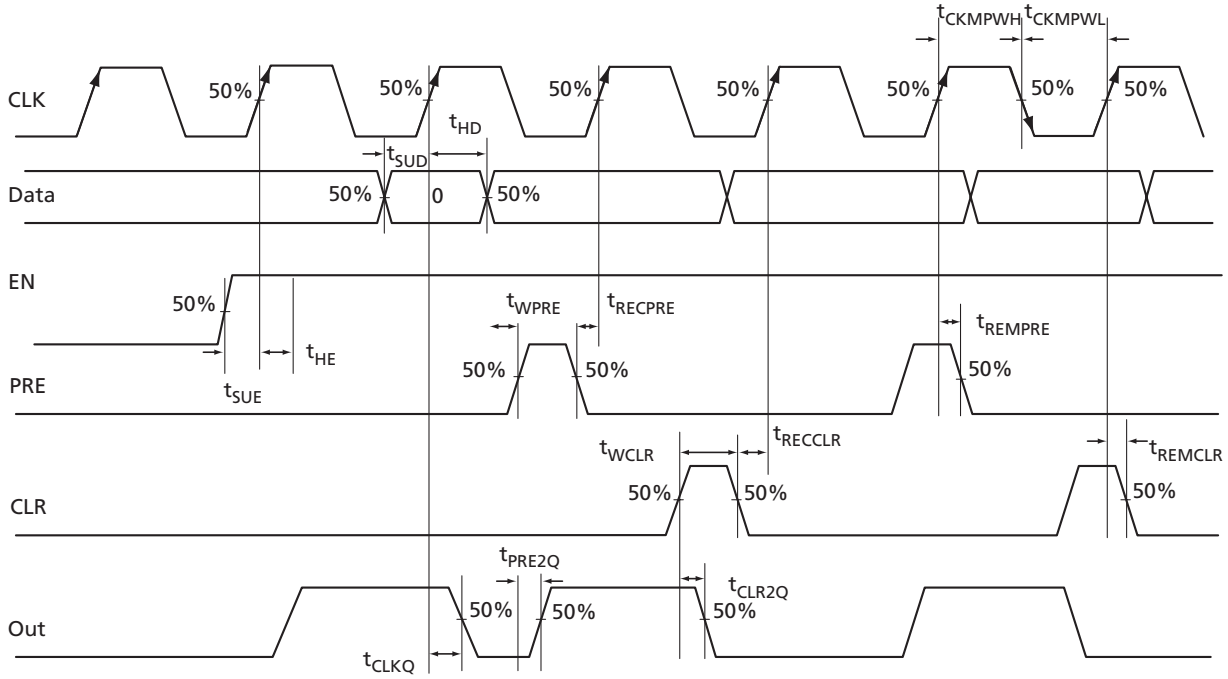


Figure 2-6 • **Sequential Timing Model and Waveforms**

Sequential Timing Characteristics

Table 2-2 • Register Delays
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{CLKQ}	Clock-to-Q of the Core Register	0.55	0.63	0.74	ns
t_{SUD}	Data Setup Time for the Core Register	0.43	0.49	0.57	ns
t_{HD}	Data Hold Time for the Core Register	0.00	0.00	0.00	ns
t_{SUE}	Enable Setup Time for the Core Register	0.45	0.52	0.61	ns
t_{HE}	Enable Hold Time for the Core Register	0.00	0.00	0.00	ns
t_{CLR2Q}	Asynchronous Clear-to-Q of the Core Register	0.40	0.45	0.53	ns
t_{PRE2Q}	Asynchronous Preset-to-Q of the Core Register	0.40	0.45	0.53	ns
t_{REMCLR}	Asynchronous Clear Removal Time for the Core Register	0.00	0.00	0.00	ns
t_{RECCLR}	Asynchronous Clear Recovery Time for the Core Register	0.22	0.25	0.30	ns
t_{REMPRE}	Asynchronous Preset Removal Time for the Core Register	0.00	0.00	0.00	ns
t_{RECPRE}	Asynchronous Preset Recovery Time for the Core Register	0.22	0.25	0.30	ns
t_{WCLR}	Asynchronous Clear Minimum Pulse Width for the Core Register	0.22	0.25	0.30	ns
t_{WPRE}	Asynchronous Preset Minimum Pulse Width for the Core Register	0.22	0.25	0.30	ns
t_{CKMPWH}	Clock Minimum Pulse Width HIGH for the Core Register	0.32	0.37	0.43	ns
t_{CKMPWL}	Clock Minimum Pulse Width LOW for the Core Register	0.36	0.41	0.48	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Array Coordinates

During many place-and-route operations in the Actel Designer software tool, it is possible to set constraints that require array coordinates. Table 2-3 is provided as a reference. The array coordinates are measured from the lower left (0, 0). They can be used in region constraints for specific logic groups/blocks, designated by a wildcard, and can contain core cells, memories, and I/Os.

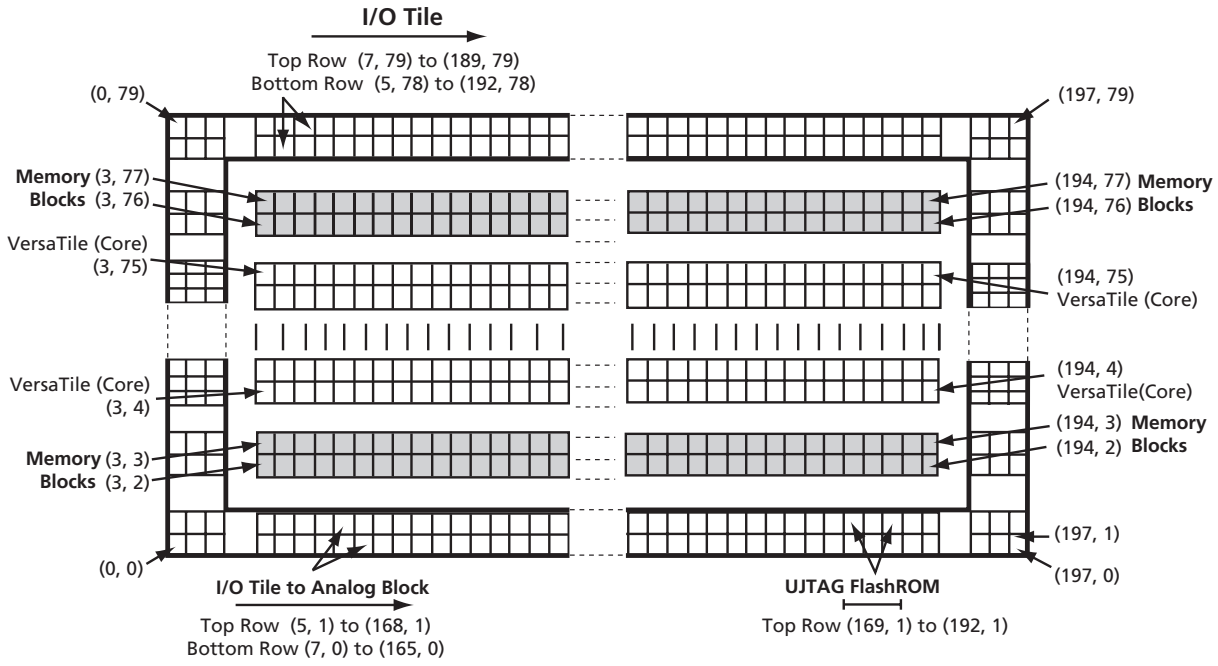
Table 2-3 provides array coordinates of core cells and memory blocks.

I/O and cell coordinates are used for placement constraints. Two coordinate systems are needed because there is not a one-to-one correspondence between I/O cells and edge core cells. In addition, the I/O coordinate system changes depending on the die/package combination. It is not listed in Table 2-3. The Designer ChipPlanner tool provides array coordinates of all I/O locations. I/O and cell coordinates are used for placement constraints. However, I/O placement is easier by package pin assignment.

Figure 2-7 illustrates the array coordinates of an AFS600 device. For more information on how to use array coordinates for region/placement constraints, see the *Designer User's Guide* or online help (available in the software) for Fusion software tools.

Table 2-3 • **Array Coordinates**

	VersaTiles				Memory Rows		All	
	Min.		Max.		Bottom	Top	Min.	Max.
Device	x	y	x	y	(x, y)	(x, y)	(x, y)	(x, y)
AFS090	3	2	98	25	None	(3, 26)	(0, 0)	(101, 29)
AFS250	3	2	130	49	None	(3, 50)	(0, 0)	(133, 53)
AFS600	3	4	194	75	(3, 2)	(3, 76)	(0, 0)	(197, 79)
AFS1500	3	4	322	123	(3, 2)	(3, 124)	(0, 0)	(325, 129)



Note: The vertical I/O tile coordinates are not shown. West side coordinates are {(0, 2) to (2, 2)} to {(0, 77) to (2, 77)}; east side coordinates are {(195, 2) to (197, 2)} to {(195, 77) to (197, 77)}.

Figure 2-7 • **Array Coordinates for AFS600**

Routing Architecture

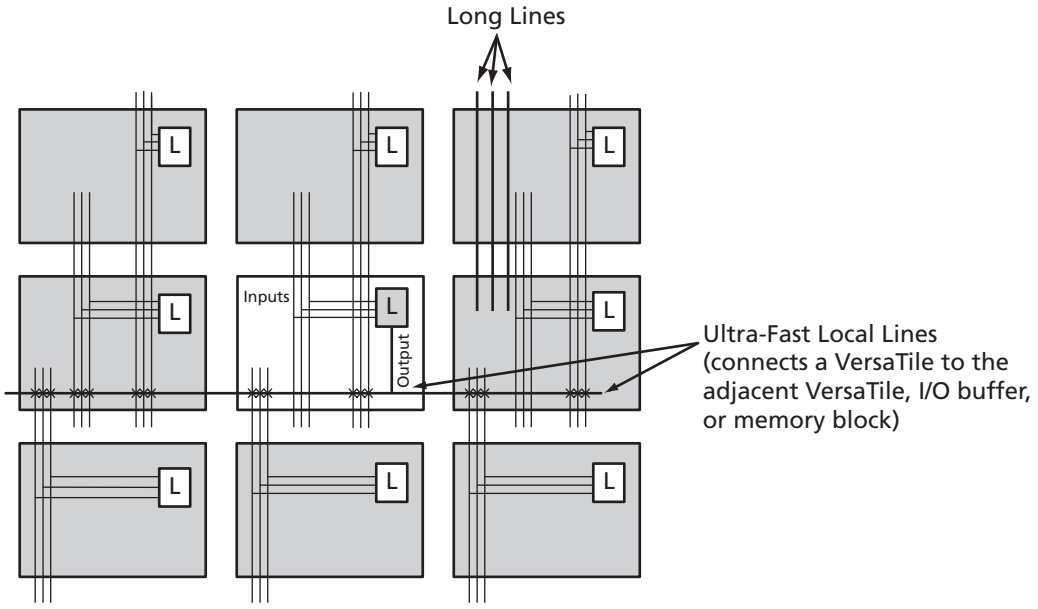
The routing structure of Fusion devices is designed to provide high performance through a flexible four-level hierarchy of routing resources: ultra-fast local resources; efficient long-line resources; high-speed very-long-line resources; and the high-performance VersaNet networks.

The ultra-fast local resources are dedicated lines that allow the output of each VersaTile to connect directly to every input of the eight surrounding VersaTiles (Figure 2-8). The exception to this is that the SET/CLR input of a VersaTile configured as a D-flip-flop is driven only by the VersaNet global network.

The efficient long-line resources provide routing for longer distances and higher-fanout connections. These resources vary in length (spanning one, two, or four VersaTiles), run both vertically and horizontally, and cover the entire Fusion device (Figure 2-9 on page 2-11). Each VersaTile can drive signals onto the efficient long-line resources, which can access every input of every VersaTile. Active buffers are inserted automatically by routing software to limit loading effects.

The high-speed very-long-line resources, which span the entire device with minimal delay, are used to route very long or high-fanout nets: length +/–12 VersaTiles in the vertical direction and length +/–16 in the horizontal direction from a given core VersaTile (Figure 2-10 on page 2-12). Very long lines in Fusion devices, like those in ProASIC3 devices, have been enhanced. This provides a significant performance boost for long-reach signals.

The high-performance VersaNet global networks are low-skew, high-fanout nets that are accessible from external pins or from internal logic (Figure 2-11 on page 2-13). These nets are typically used to distribute clocks, reset signals, and other high-fanout nets requiring minimum skew. The VersaNet networks are implemented as clock trees, and signals can be introduced at any junction. These can be employed hierarchically, with signals accessing every input on all VersaTiles.



Note: Input to the core cell for the D-flip-flop set and reset is only available via the VersaNet global network connection.

Figure 2-8 • **Ultra-Fast Local Lines Connected to the Eight Nearest Neighbors**

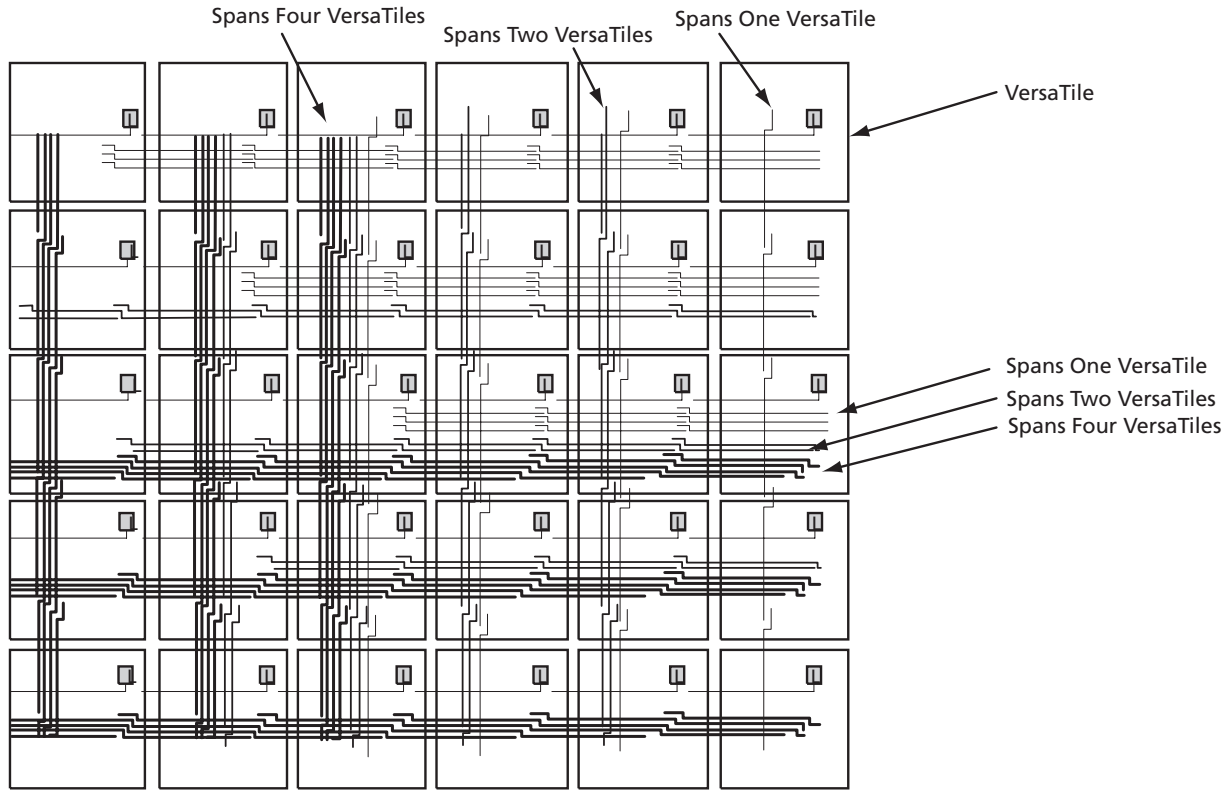


Figure 2-9 • **Efficient Long-Line Resources**

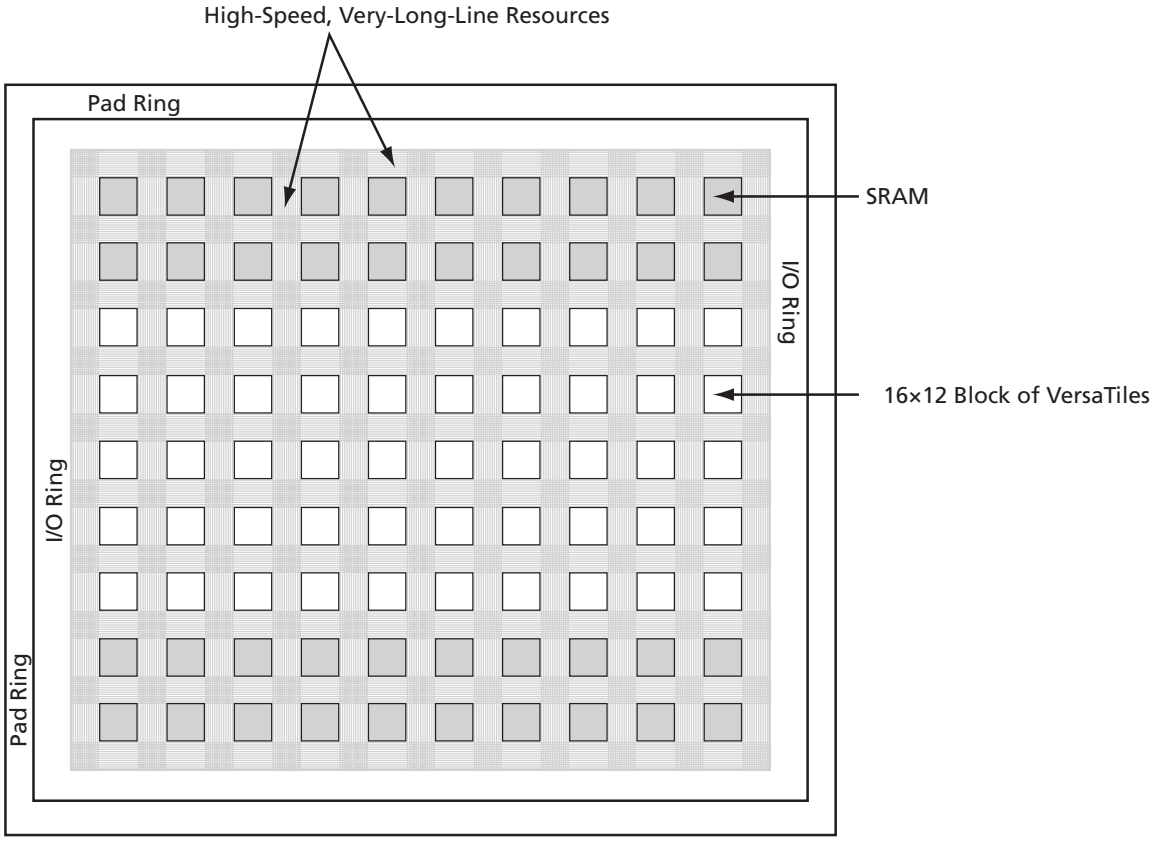


Figure 2-10 • **Very-Long-Line Resources**

Global Resources (VersaNets)

Fusion devices offer powerful and flexible control of circuit timing through the use of analog circuitry. Each chip has six CCCs. The west CCC also contains a PLL core. In the two larger devices (AFS600 and AFS1500), the west and the east CCCs each contain a PLL. The PLLs include delay lines, a phase shifter (0°, 90°, 180°, 270°), and clock multipliers/dividers. Each CCC has all the circuitry needed for the selection and interconnection of inputs to the VersaNet global network. The east and west CCCs each have access to three VersaNet global lines on each side of the chip (six lines total). The CCCs at the four corners each have access to three quadrant global lines on each quadrant of the chip.

Advantages of the VersaNet Approach

One of the architectural benefits of Fusion is the set of powerful and low-delay VersaNet global networks. Fusion offers six chip (main) global networks that are distributed from the center of the FPGA array (Figure 2-11). In addition, Fusion devices have three regional globals (quadrant globals) in each of the four chip quadrants. Each core VersaTile has access to nine global network resources: three quadrant and six chip (main) global networks. There are a total of 18 global networks on the device. Each of these networks contains spines and ribs that reach all VersaTiles in all quadrants (Figure 2-12 on page 2-14). This flexible VersaNet global network architecture allows users to map up to 180 different internal/external clocks in a Fusion device. Details on the VersaNet networks are given in Table 2-4 on page 2-14. The flexibility of the Fusion VersaNet global network allows the designer to address several design requirements. User applications that are clock-resource-intensive can easily route external or gated internal clocks using VersaNet global routing networks. Designers can also drastically reduce delay penalties and minimize resource usage by mapping critical, high-fanout nets to the VersaNet global network.

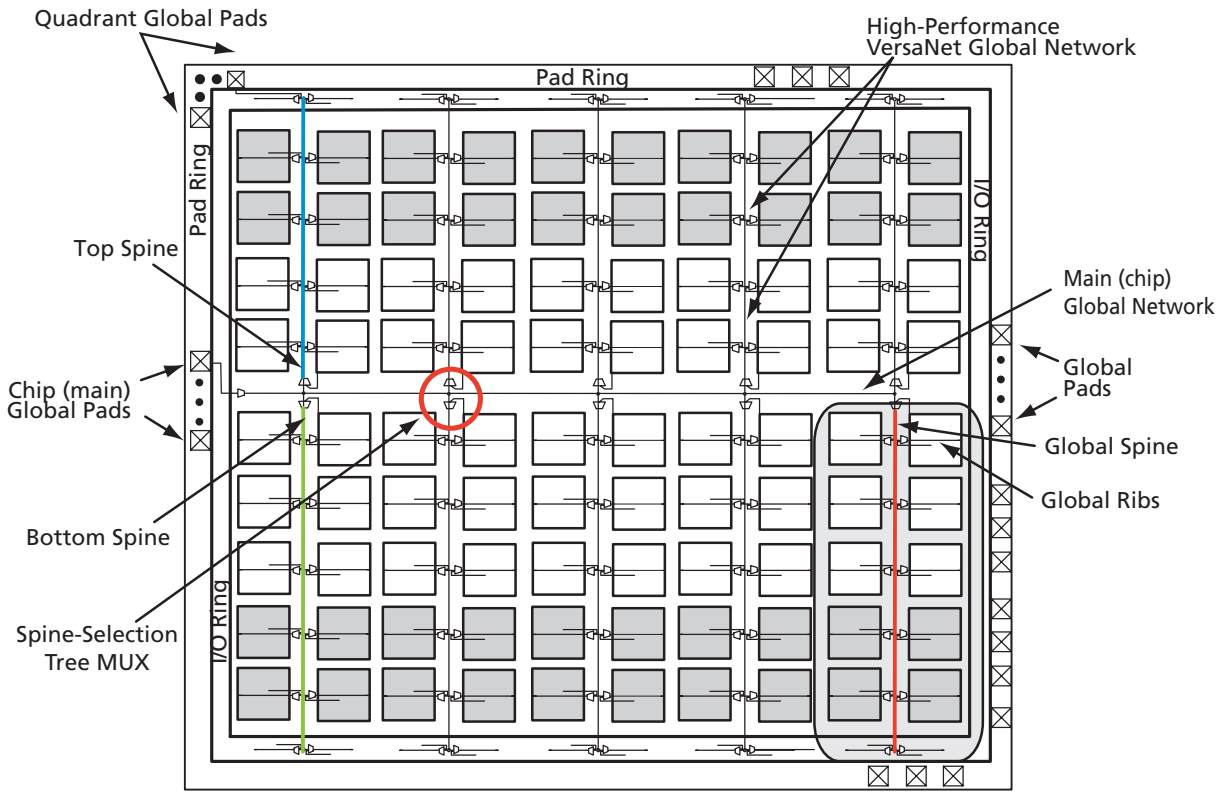


Figure 2-11 • **Overview of Fusion VersaNet Global Network**

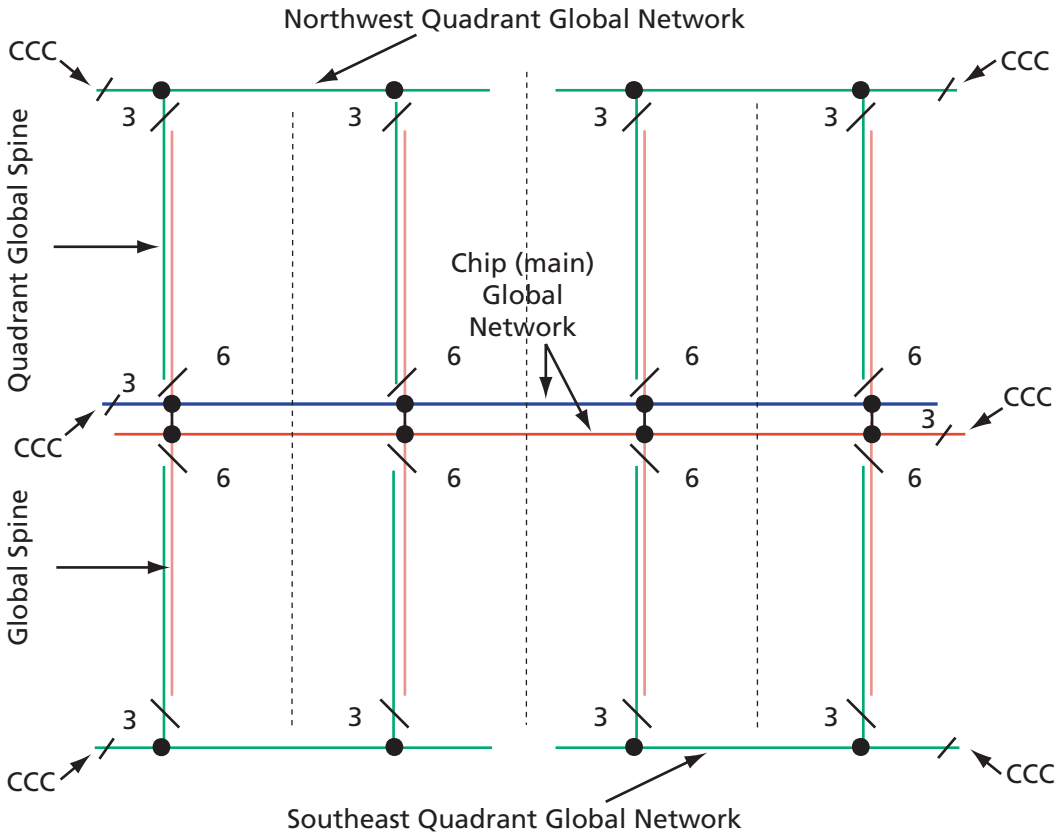


Figure 2-12 • **Global Network Architecture**

Table 2-4 • **Globals/Spines/Rows by Device**

	AFS090	AFS250	AFS600	AFS1500
Global VersaNets (trees)*	9	9	9	9
VersaNet Spines/Tree	4	8	12	20
Total Spines	36	72	108	180
VersaTiles in Each Top or Bottom Spine	384	768	1,152	1,920
Total VersaTiles	2,304	6,144	13,824	38,400

*Note: *There are six chip (main) globals and three globals per quadrant.*

VersaNet Global Networks and Spine Access

The Fusion architecture contains a total of 18 segmented global networks that can access the VersaTiles, SRAM, and I/O tiles on the Fusion device. There are 6 chip (main) global networks that access the entire device and 12 quadrant networks (3 in each quadrant). Each device has a total of 18 globals. These VersaNet global networks offer fast, low-skew routing resources for high-fanout nets, including clock signals. In addition, these highly segmented global networks offer users the flexibility to create low-skew local networks using spines for up to 180 internal/external clocks (in an AFS1500 device) or other high-fanout nets in Fusion devices. Optimal usage of these low-skew networks can result in significant improvement in design performance on Fusion devices.

The nine spines available in a vertical column reside in global networks with two separate regions of scope: the quadrant global network, which has three spines, and the chip (main) global network, which has six spines. Note that there are three quadrant spines in each quadrant of the device. There are four quadrant global network regions per device (Figure 2-12 on page 2-14).

The spines are the vertical branches of the global network tree, shown in Figure 2-11 on page 2-13. Each spine in a vertical column of a chip (main) global network is further divided into two equal-length spine segments: one in the top and one in the bottom half of the die.

Each spine and its associated ribs cover a certain area of the Fusion device (the "scope" of the spine; see Figure 2-11 on page 2-13). Each spine is accessed by the dedicated global network MUX tree architecture, which defines how a particular spine is driven—either by the signal on the global network from a CCC, for example, or another net defined by the user (Figure 2-13). Quadrant spines can be driven from user I/Os on the north and south sides of the die, via analog I/Os configured as direct digital inputs. The ability to drive spines in the quadrant global networks can have a significant effect on system performance for high-fanout inputs to a design.

Details of the chip (main) global network spine-selection MUX are presented in Figure 2-13. The spine drivers for each spine are located in the middle of the die.

Quadrant spines are driven from a north or south rib. Access to the top and bottom ribs is from the corner CCC or from the I/Os on the north and south sides of the device. For details on using spines in Fusion devices, see the Actel application note *Using Global Resources in Actel Fusion Devices*.

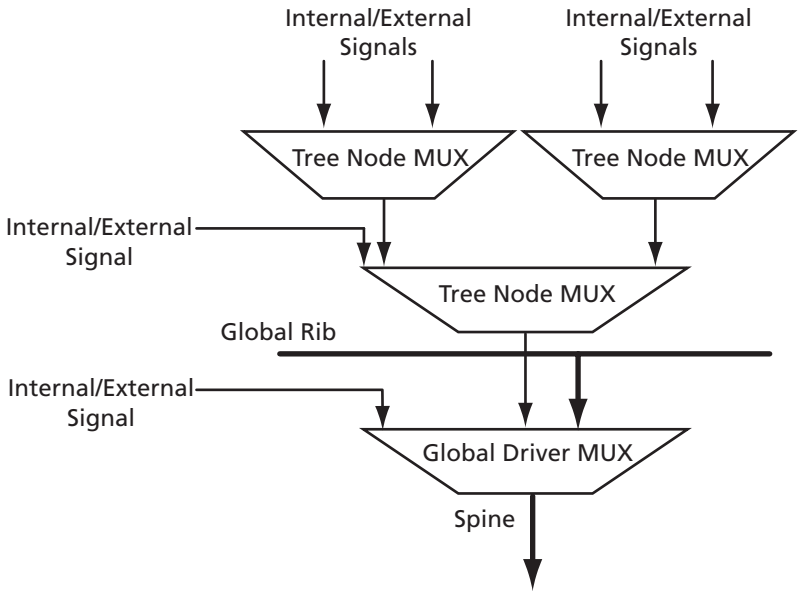


Figure 2-13 • **Spine-Selection MUX of Global Tree**

Clock Aggregation

Clock aggregation allows for multi-spine clock domains. A MUX tree provides the necessary flexibility to allow long lines or I/Os to access domains of one, two, or four global spines. Signal access to the clock aggregation system is achieved through long-line resources in the central rib, and also through local resources in the north and south ribs, allowing I/Os to feed directly into the clock system. As Figure 2-14 indicates, this access system is contiguous.

There is no break in the middle of the chip for north and south I/O VersaNet access. This is different from the quadrant clocks, located in these ribs, which only reach the middle of the rib.Refer to the *Using Global Resources in Actel Fusion Devices* application note.

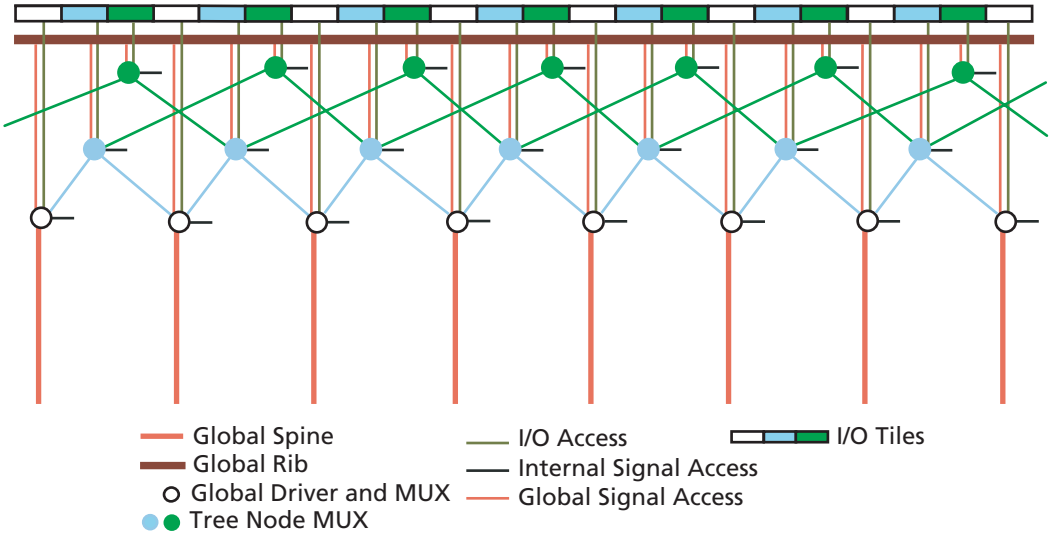


Figure 2-14 • **Clock Aggregation Tree Architecture**

Global Resource Characteristics

AFS600 VersaNet Topology

Clock delays are device-specific. Figure 2-15 is an example of a global tree used for clock routing. The global tree presented in Figure 2-15 is driven by a CCC located on the west side of the AFS600 device. It is used to drive all D-flip-flops in the device.

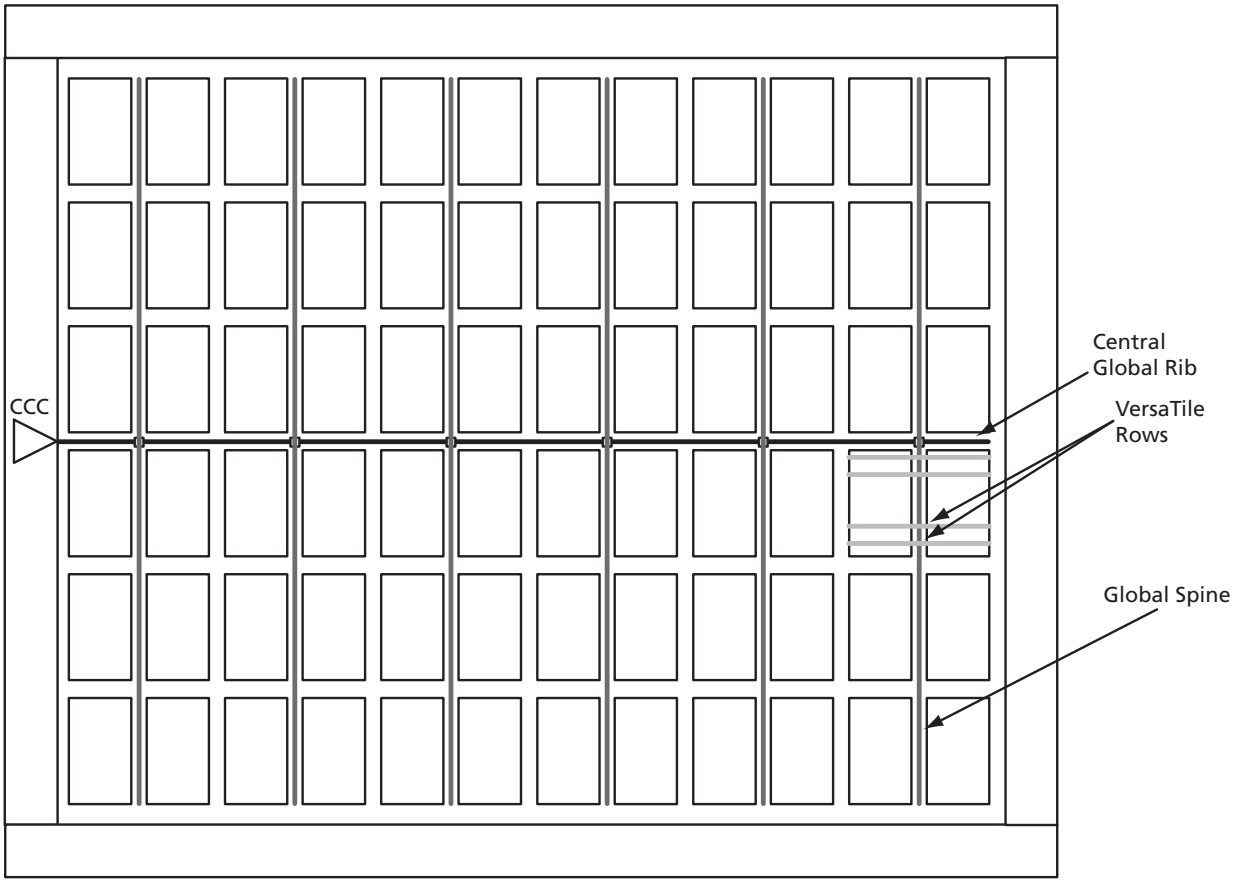


Figure 2-15 • **Example of Global Tree Use in an AFS600 Device for Clock Routing**

VersaNet Timing Characteristics

Global clock delays include the central rib delay, the spine delay, and the row delay. Delays do not include I/O input buffer clock delays, as these are dependent upon I/O standard, and the clock may be driven and conditioned internally by the CCC module. Table 2-5, Table 2-6, Table 2-7, and Table 2-8 on page 2-19 present minimum and maximum global clock delays within the device Minimum and maximum delays are measured with minimum and maximum loading, respectively.

Timing Characteristics

Table 2-5 • **AFS1500 Global Resource Timing**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2		–1		Std.		Units
		Min.[1]	Max.[2]	Min.[1]	Max.[2]	Min.[1]	Max.[2]	
t_{RCKL}	Input LOW Delay for Global Clock	1.53	1.75	1.74	1.99	2.05	2.34	ns
t_{RCKH}	Input HIGH Delay for Global Clock	1.53	1.79	1.75	2.04	2.05	2.40	ns
t_{RCKSW}	Maximum Skew for Global Clock		0.26		0.29		0.34	ns
F_{RMAX}	Maximum Frequency for Global Clock							MHz

Notes:

1. *Value reflects minimum load. The delay is measured from the CCC output to the clock pin of a sequential element located in a lightly loaded row (single element is connected to the global net).*
2. *Value reflects maximum load. The delay is measured on the clock pin of the farthest sequential element located in a fully loaded row (all available flip-flops are connected to the global net in the row).*
3. *For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.*

Table 2-6 • **AFS600 Global Resource Timing**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2		–1		Std.		Units
		Min.[1]	Max.[2]	Min.[1]	Max.[2]	Min.[1]	Max.[2]	
t_{RCKL}	Input LOW Delay for Global Clock	1.27	1.49	1.44	1.70	1.69	2.00	ns
t_{RCKH}	Input HIGH Delay for Global Clock	1.26	1.54	1.44	1.75	1.69	2.06	ns
t_{RCKSW}	Maximum Skew for Global Clock		0.27		0.31		0.36	ns
F_{RMAX}	Maximum Frequency for Global Clock							MHz

Notes:

1. *Value reflects minimum load. The delay is measured from the CCC output to the clock pin of a sequential element located in a lightly loaded row (single element is connected to the global net).*
2. *Value reflects maximum load. The delay is measured on the clock pin of the farthest sequential element located in a fully loaded row (all available flip-flops are connected to the global net in the row).*
3. *For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.*

Table 2-7 • AFS250 Global Resource Timing
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2		–1		Std.		Units
		Min.[1]	Max.[2]	Min.[1]	Max.[2]	Min.[1]	Max.[2]	
t_{RCKL}	Input LOW Delay for Global Clock	0.89	1.12	1.02	1.27	1.20	1.50	ns
t_{RCKH}	Input HIGH Delay for Global Clock	0.88	1.14	1.00	1.30	1.17	1.53	ns
t_{RCKSW}	Maximum Skew for Global Clock		0.26		0.30		0.35	ns
F_{RMAX}	Maximum Frequency for Global Clock							MHz

Notes:

1. *Value reflects minimum load. The delay is measured from the CCC output to the clock pin of a sequential element located in a lightly loaded row (single element is connected to the global net).*
2. *Value reflects maximum load. The delay is measured on the clock pin of the farthest sequential element located in a fully loaded row (all available flip-flops are connected to the global net in the row).*
3. *For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.*

Table 2-8 • AFS090 Global Resource Timing
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2		–1		Std.		Units
		Min.[1]	Max.[2]	Min.[1]	Max.[2]	Min.[1]	Max.[2]	
t_{RCKL}	Input LOW Delay for Global Clock	0.84	1.07	0.96	1.21	1.13	1.43	ns
t_{RCKH}	Input HIGH Delay for Global Clock	0.83	1.10	0.95	1.25	1.12	1.47	ns
t_{RCKSW}	Maximum Skew for Global Clock		0.27		0.30		0.36	ns
F_{RMAX}	Maximum Frequency for Global Clock							MHz

Notes:

1. *Value reflects minimum load. The delay is measured from the CCC output to the clock pin of a sequential element located in a lightly loaded row (single element is connected to the global net).*
2. *Value reflects maximum load. The delay is measured on the clock pin of the farthest sequential element located in a fully loaded row (all available flip-flops are connected to the global net in the row).*
3. *For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.*

Clocking Resources

The Fusion family has a robust collection of clocking peripherals, as shown in the block diagram in Figure 2-16. These on-chip resources enable the creation, manipulation, and distribution of many clock signals. The Fusion integrated RC oscillator produces a 100 MHz clock source with no external components. For systems requiring more precise clock signals, the Actel Fusion family supports an on-chip crystal oscillator circuit. The integrated PLLs in each Fusion device can use the RC oscillator, crystal oscillator, or another on-chip clock signal as a source. These PLLs offer a variety of capabilities to modify the clock source (multiply, divide, synchronize, advance, or delay). Utilizing the CCC found in the popular Actel ProASIC3 family, Fusion incorporates six CCC blocks. The CCCs allow access to Fusion global and local clock distribution nets, as described in the "Global Resources (VersaNets)" section on page 2-13.

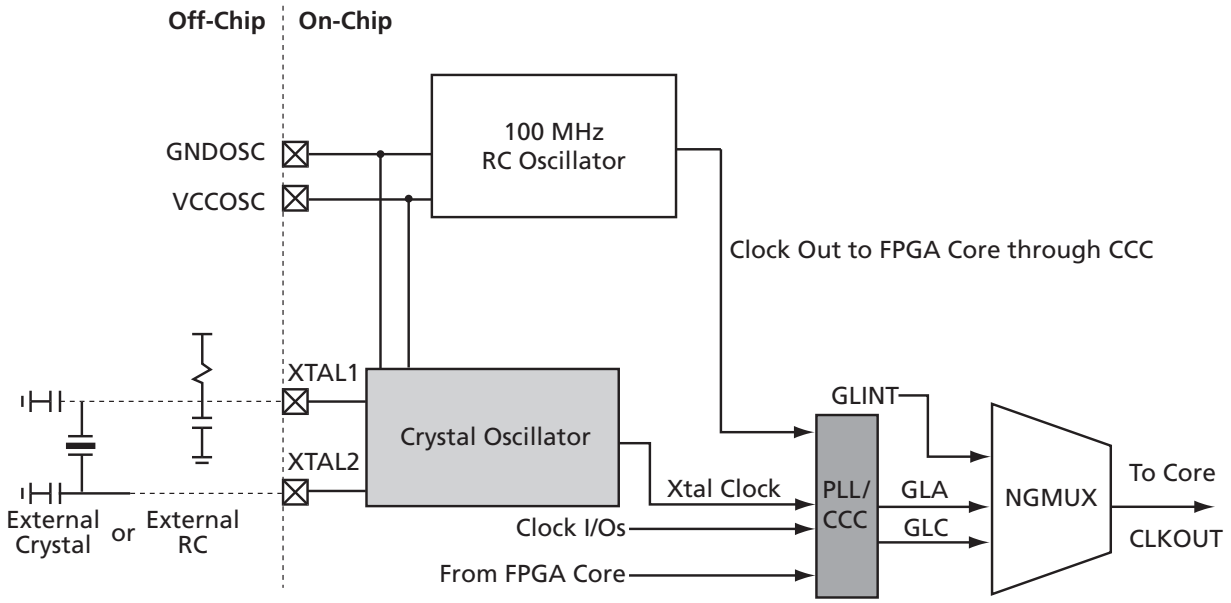


Figure 2-16 • **Fusion Clocking Options**

RC Oscillator

The RC oscillator is an on-chip free-running clock source generating a 100 MHz clock. It can be used as a source clock for both on-chip and off-chip resources. When used in conjunction with the Fusion PLL and CCC circuits, the RC oscillator clock source can be used to generate clocks of varying frequency and phase.

The Fusion RC oscillator is very accurate at ±1% over commercial and industrial temperature ranges. It is an automated clock, requiring no setup or configuration by the user. It requires only that the power and GNDOSC pins be connected; no external components are required. The RC oscillator can be used to drive either a PLL or another internal signal.

RC Oscillator Characteristics

Table 2-9 • **Electrical Characteristics of RC Oscillator**

Parameter	Description	Conditions	Min.	Typ.	Max.	Units
F_{RC}	Operating Frequency			100		MHz
	Accuracy	Temperature: 0°C to 85°C Voltage: 3.3 V ± 5%		1		%
		Temperature: –40°C to 125°C Voltage: 3.3 V ± 5%		3		%
	Output Jitter	Period Jitter (at 5 k cycles)		100		ps
		Cycle–Cycle Jitter (at 5 k cycles)		100		ps
		Period Jitter (at 5 k cycles) with 1 KHz / 300 mV peak-to-peak noise on power supply		150		ps
		Cycle–Cycle Jitter (at 5 k cycles) with 1 KHz / 300 mV peak-to-peak noise on power supply		150		ps
	Output Duty Cycle			50		%
I_{DYNRC}	Operating Current			1		mA

Crystal Oscillator

The Crystal Oscillator (XTLOSC) is source that generates the clock from an external crystal. The output of XTLOSC CLKOUT signal can be selected as an input to the PLL. Refer to "Clock Conditioning Circuits" section for more details. The XTLOSC can operate in normal operations and Standby mode (RTC is running and 1.5 V is not present).

In normal operation, the internal FPGA_EN signal is '1' as long as 1.5 V is present for V_{CC}. As such, the internal enable signal, XTL_EN, for Crystal Oscillator is enabled since FPGA_EN is asserted. The XTL_MODE has the option of using MODE or RTC_MODE, depending on SELMODE.

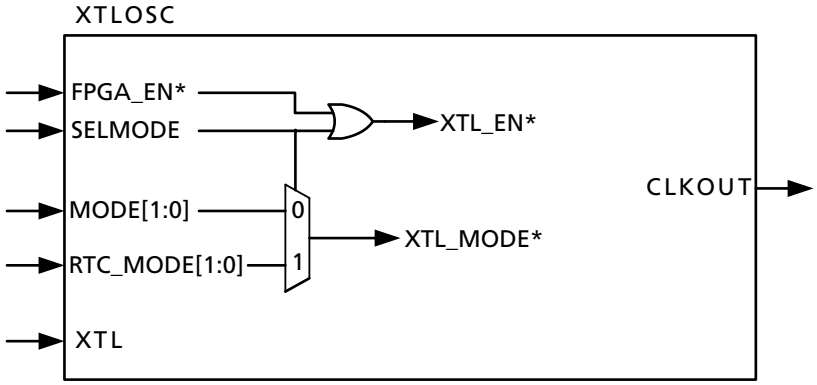
During Standby, 1.5 V is not available, as such, and FPGA_EN is '0'. SELMODE must be asserted in order for XTL_EN to be enabled; hence XTL_MODE relies on RTC_MODE. SELMODE and RTC_MODE must be connected to RTCXTLSEL and RTCXTLMODE from the AB respectively for correct operation during Standby (refer to the "Real-Time Counter System" section on page 2-34 for a detailed description).

The Crystal Oscillator can be configured in one of four modes:

- RC network, 32 KHz to 4 MHz
- Low gain, 32 to 200 KHz
- Medium gain, 0.20 to 2.0 MHz
- High gain, 2.0 to 20.0 MHz

In RC network mode, the XTAL1 pin is connected to an RC circuit, as shown in Figure 2-17. The XTAL2 pin should be left floating. The RC value can be chosen based on Figure 2-18 for any desired frequency between 32 KHz and 4 MHz. The RC network mode can also accommodate an external clock source on XTAL1 instead of an RC circuit.

In Low gain, Medium gain, and High gain, an external crystal component or ceramic resonator can be added onto XTAL1 and XTAL2, as shown in Figure 2-17.



*Note: *Internal signal—does not exist in macro.*

Figure 2-17 • **XTLOSC Macro**

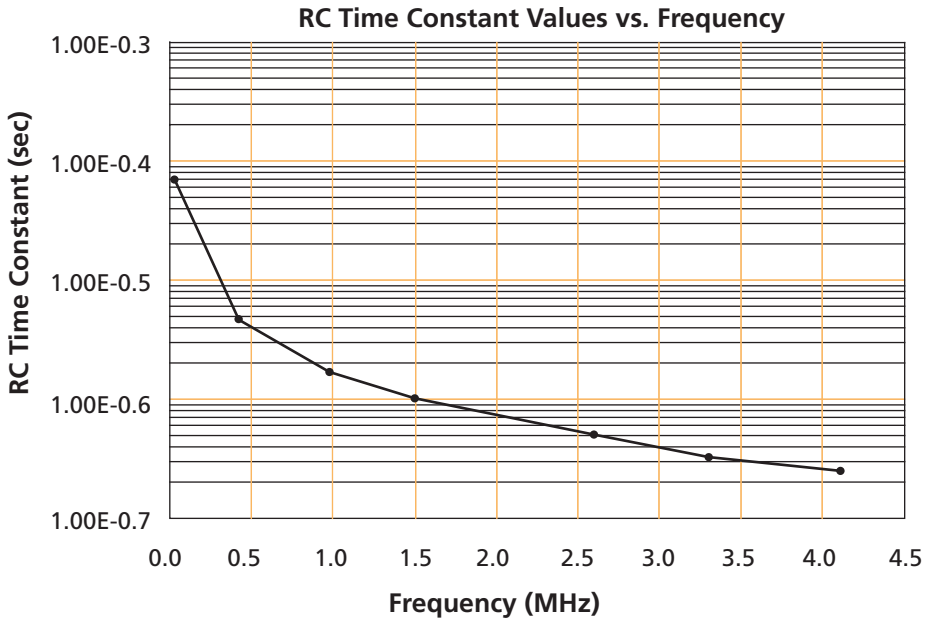


Figure 2-18 • **Crystal Oscillator: RC Time Constant Values vs. Frequency (typical)**

Table 2-10 • **XTLOSC Signals Descriptions**

Signal Name	Width	Direction	Function
XTL_EN*	1		Enables the crystal. Active high.
XTL_MODE*	2		Settings for the crystal clock for different frequency.

Value	Modes	Frequency Range
b'00	RC network	32 KHz to 4 MHz
b'01	Low gain	32 to 200 KHz
b'10	Medium gain	0.20 to 2.0 MHz
b'11	High gain	2.0 to 20.0 MHz

SELMODE	1	IN	Selects the source of XTL_MODE and also enables the XTL_EN. Connect from RTCXTLSEL from AB. 0 For normal operation or sleep mode, XTL_EN depends on FPGA_EN, XTL_MODE depends on MODE 1 For Standby mode, XTL_EN is enabled, XTL_MODE depends on RTC_MODE
RTC_MODE[1:0]	2	IN	Settings for the crystal clock for different frequency ranges. XTL_MODE uses RTC_MODE when SELMODE is '1'.
MODE[1:0]	2	IN	Settings for the crystal clock for different frequency ranges. XTL_MODE uses MODE when SELMODE is '0'. In Standby, MODE inputs will be 0's.
FPGA_EN*	1	IN	0 when 1.5 V is not present for V_{CC} 1 when 1.5 V is present for V_{CC}
XTL	1	IN	Crystal Clock source
CLKOUT	1	OUT	Crystal Clock output

*Note: *Internal signal—does not exist in macro.*

Clock Conditioning Circuits

In Fusion devices, the CCCs are used to implement frequency division, frequency multiplication, phase shifting, and delay operations.

The CCCs are available in six chip locations—each of the four chip corners and the middle of the east and west chip sides.

Each CCC can implement up to three independent global buffers (with or without programmable delay), or a PLL function (programmable frequency division/multiplication, phase shift, and delays) with up to three global outputs. Unused global outputs of a PLL can be used to implement independent global buffers, up to a maximum of three global outputs for a given CCC.

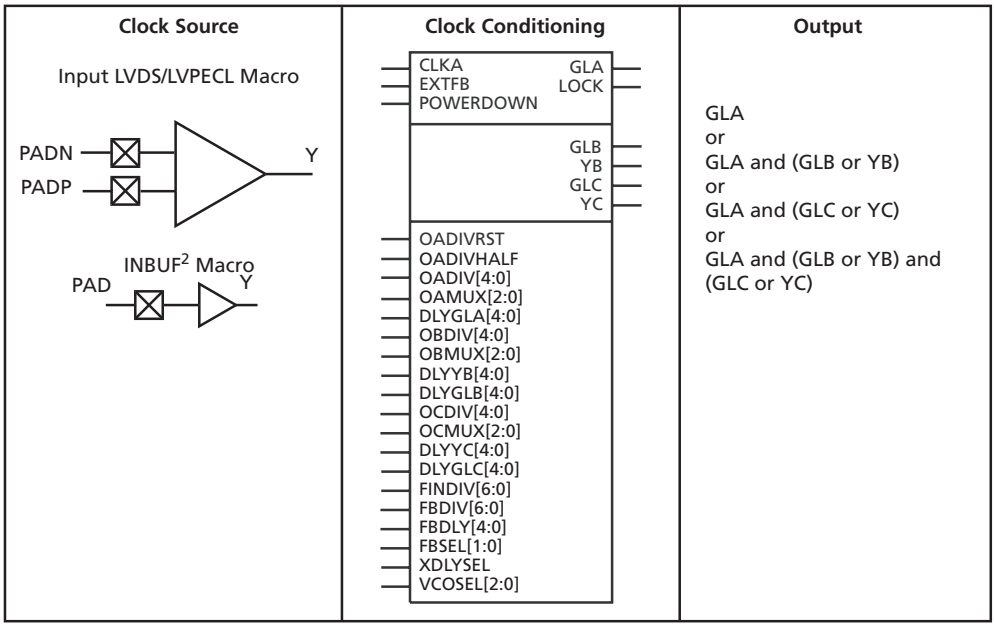
A global buffer can be placed in any of the three global locations (CLKA-GLA, CLKB-GLB, and CLKC-GLC) of a given CCC.

A PLL macro uses the CLKA CCC input to drive its reference clock. It uses the GLA and, optionally, the GLB and GLC global outputs to drive the global networks. A PLL macro can also drive the YB and YC regular core outputs. The GLB (or GLC) global output cannot be reused if the YB (or YC) output is used (Figure 2-19). Refer to the "PLL Macro" section on page 2-30 for more information.

Each global buffer, as well as the PLL reference clock, can be driven from one of the following:

- 3 dedicated single-ended I/Os using a hardwired connection
- 2 dedicated differential I/Os using a hardwired connection
- The FPGA core

The CCC block is fully configurable, either via flash configuration bits set in the programming bitstream or through an asynchronous interface. This asynchronous interface is dynamically accessible from inside the Fusion device to permit changes of parameters (such as divide ratios) during device operation. To increase the versatility and flexibility of the clock conditioning system, the CCC configuration is determined either by the user during the design process, with configuration data being stored in flash memory as part of the device programming procedure, or by writing data into a dedicated shift register during normal device operation. This latter mode allows the user to dynamically reconfigure the CCC without the need for core programming. The shift register is accessed through a simple serial interface. Refer to the *UJTAG Applications in Actel's Low-Power Flash Devices* handbook chapter and the "CCC and PLL Characteristics" section on page 2-31 for more information.



Notes:

1. *Visit the Actel website for future application notes concerning dynamic PLL reconfiguration. Refer to the "PLL Macro" section on page 2-30 for signal descriptions.*
2. *Many specific INBUF macros support the wide variety of single-ended and differential I/O standards for the Fusion family.*
3. *Refer to the Fusion, IGLOO/e, and ProASIC3/E Macro Library Guide for more information.*

Figure 2-19 • **Fusion CCC Options: Global Buffers with the PLL Macro**

Table 2-11 • **Available Selections of I/O Standards within CLKBUF and CLKBUF_LVDS/LVPECL Macros**

CLKBUF Macros
CLKBUF_LVCMOS5
CLKBUF_LVCMOS33[1]
CLKBUF_LVCMOS18
CLKBUF_LVCMOS15
CLKBUF_PCI
CLKBUF_LVDS[2]
CLKBUF_LVPECL

Notes:

1. *This is the default macro. For more details, refer to the Fusion, IGLOO/e and ProASIC3/E Macro Library Guide.*
2. *The BLVDS and M-LVDS standards are supported with CLKBUF_LVDS.*

Global Buffers with No Programmable Delays

The CLKBUF and CLKBUF_LVPECL/LVDS macros are composite macros that include an I/O macro driving a global buffer, hardwired together (Figure 2-20).

The CLKINT macro provides a global buffer function driven by the FPGA core.

The CLKBUF, CLKBUF_LVPECL/LVDS, and CLKINT macros are pass-through clock sources and do not use the PLL or provide any programmable delay functionality.

Many specific CLKBUF macros support the wide variety of single-ended and differential I/O standards supported by Fusion devices. The available CLKBUF macros are described in the *Fusion, IGLOO/e and ProASIC3/E Macro Library Guide*.

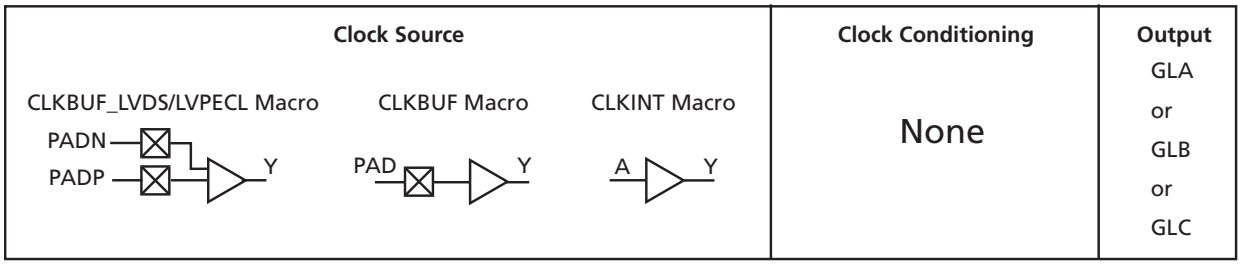


Figure 2-20 • **Global Buffers with No Programmable Delay**

Global Buffers with Programmable Delay

The CLKDLY macro is a pass-through clock source that does not use the PLL, but provides the ability to delay the clock input using a programmable delay (Figure 2-21). The CLKDLY macro takes the selected clock input and adds a user-defined delay element. This macro generates an output clock phase shift from the input clock.

The CLKDLY macro can be driven by an INBUF macro to create a composite macro, where the I/O macro drives the global buffer (with programmable delay) using a hardwired connection. In this case, the I/O must be placed in one of the dedicated global I/O locations.

Many specific INBUF macros support the wide variety of single-ended and differential I/O standards supported by the Fusion family. The available INBUF macros are described in the *Fusion, IGLOO/e and ProASIC3/E Macro Library Guide.*

The CLKDLY macro can be driven directly from the FPGA core.

The CLKDLY macro can also be driven from an I/O that is routed through the FPGA regular routing fabric. In this case, users must instantiate a special macro, PLLINT, to differentiate from the hardwired I/O connection described earlier.

The visual CLKDLY configuration in the SmartGen part of the Libero IDE and Designer tools allows the user to select the desired amount of delay and configures the delay elements appropriately. SmartGen also allows the user to select the input clock source. SmartGen will automatically instantiate the special macro, PLLINT, when needed.

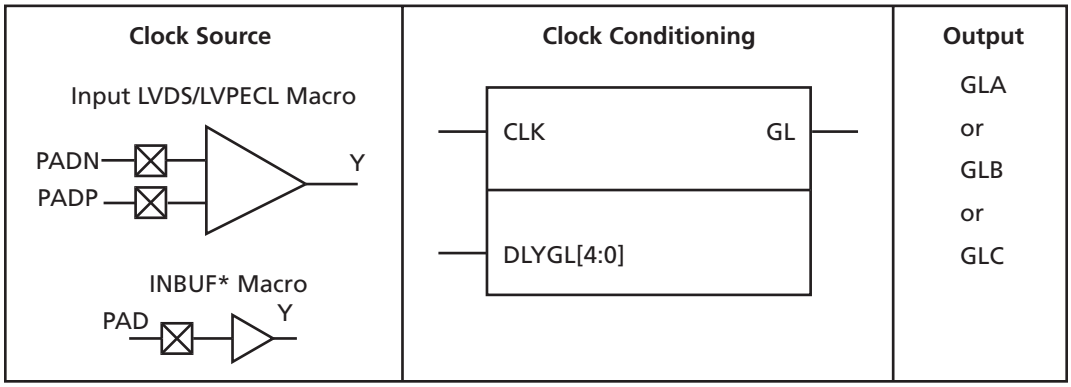
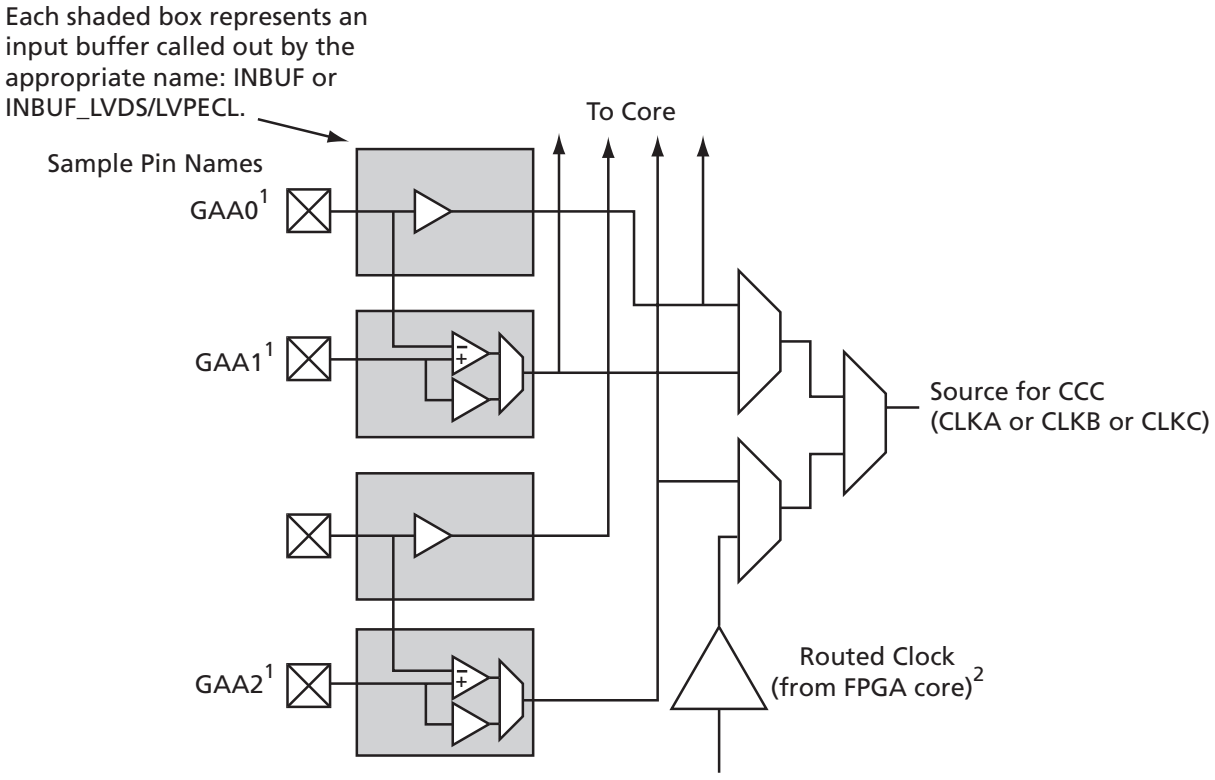


Figure 2-21 • **Fusion CCC Options: Global Buffers with Programmable Delay**

Global Input Selections

Each global buffer, as well as the PLL reference clock, can be driven from one of the following (Figure 2-22):

- 3 dedicated single-ended I/Os using a hardwired connection
- 2 dedicated differential I/Os using a hardwired connection
- The FPGA core



Notes:

3. *Represents the global input pins. Globals have direct access to the clock conditioning block and are not routed via the FPGA fabric. Refer to the "User I/O Naming Convention" section on page 2-159 for more information.*
4. *Instantiate the routed clock source input as follows:*
   a) Connect the output of a logic element to the clock input of the PLL, CLKDLY, or CLKINT macro.
   b) Do not place a clock source I/O (INBUF or INBUF_LVPECL/LVDS) in a relevant global pin location.
5. *LVDS-based clock sources are available in the east and west banks on all Fusion devices.*

Figure 2-22 • **Clock Input Sources Including CLKBUF, CLKBUF_LVDS/LVPECL, and CLKINT**

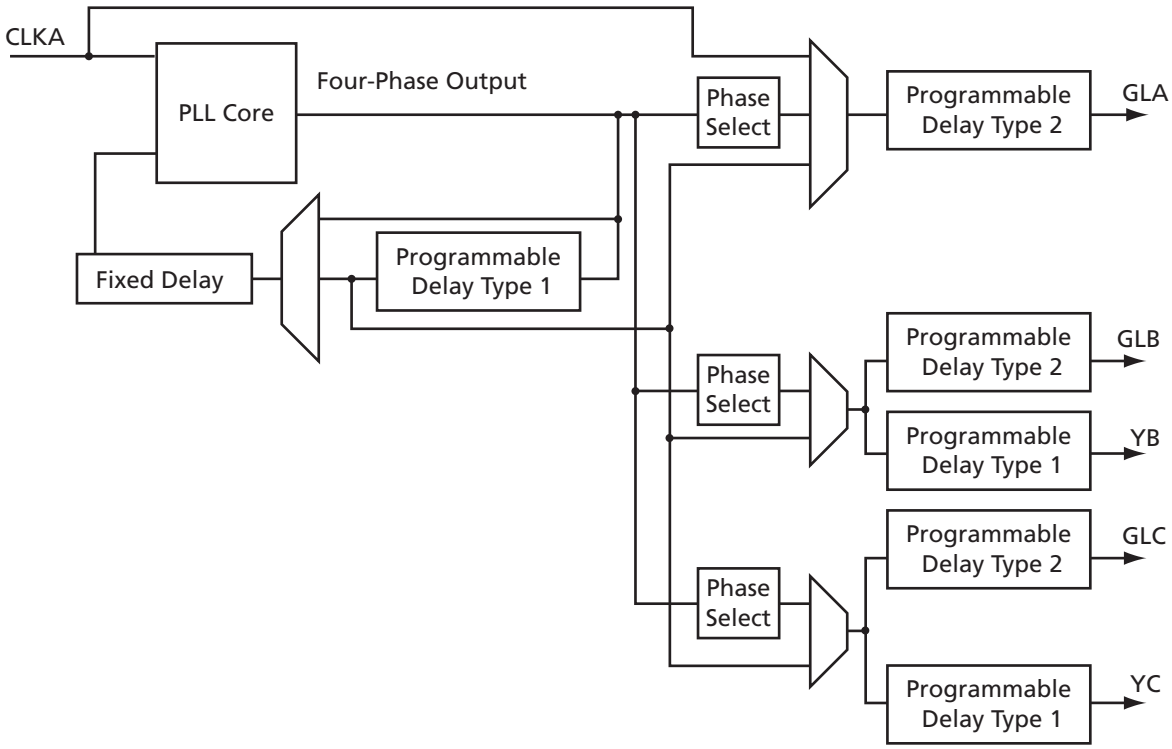
CCC Physical Implementation

The CCC circuit is composed of the following (Figure 2-23):

- PLL core
- 3 phase selectors
- 6 programmable delays and 1 fixed delay
- 5 programmable frequency dividers that provide frequency multiplication/division (not shown in Figure 2-23 because they are automatically configured based on the user's required frequencies)
- 1 dynamic shift register that provides CCC dynamic reconfiguration capability (not shown)

CCC Programming

The CCC block is fully configurable. It is configured via static flash configuration bits in the array, set by the user in the programming bitstream, or configured through an asynchronous dedicated shift register, dynamically accessible from inside the Fusion device. The dedicated shift register permits changes of parameters such as PLL divide ratios and delays during device operation. This latter mode allows the user to dynamically reconfigure the PLL without the need for core programming. The register file is accessed through a simple serial interface.



Note: Clock divider and multiplier blocks are not shown in this figure or in SmartGen. They are automatically configured based on the user's required frequencies.

Figure 2-23 • **PLL Block**

PLL Macro

The PLL functionality of the clock conditioning block is supported by the PLL macro. Note that the PLL macro reference clock uses the CLKA input of the CCC block, which is only accessible from the global A[2:0] package pins. Refer to Figure 2-22 on page 2-28 for more information.

The PLL macro provides five derived clocks (three independent) from a single reference clock. The PLL feedback loop can be driven either internally or externally. The PLL macro also provides power-down input and lock output signals. During power-up, POWERDOWN should be asserted LOW until V_{CC} is up. See Figure 2-19 on page 2-25 for more information.

Inputs:

- CLKA: selected clock input
- POWERDOWN (active low): disables PLLs. The default state is power-down on (active low).

Outputs:

- LOCK (active high): indicates that PLL output has locked on the input reference signal
- GLA, GLB, GLC: outputs to respective global networks
- YB, YC: allows output from the CCC to be routed back to the FPGA core

As previously described, the PLL allows up to five flexible and independently configurable clock outputs. Figure 2-23 on page 2-29 illustrates the various clock output options and delay elements.

As illustrated, the PLL supports three distinct output frequencies from a given input clock. Two of these (GLB and GLC) can be routed to the B and C global networks, respectively, and/or routed to the device core (YB and YC).

There are five delay elements to support phase control on all five outputs (GLA, GLB, GLC, YB, and YC).

There is also a delay element in the feedback loop that can be used to advance the clock relative to the reference clock.

The PLL macro reference clock can be driven by an INBUF macro to create a composite macro, where the I/O macro drives the global buffer (with programmable delay) using a hardwired connection. In this case, the I/O must be placed in one of the dedicated global I/O locations.

The PLL macro reference clock can be driven directly from the FPGA core.

The PLL macro reference clock can also be driven from an I/O routed through the FPGA regular routing fabric. In this case, users must instantiate a special macro, PLLINT, to differentiate it from the hardwired I/O connection described earlier.

The visual PLL configuration in SmartGen, available with the Libero IDE and Designer tools, will derive the necessary internal divider ratios based on the input frequency and desired output frequencies selected by the user. SmartGen allows the user to select the various delays and phase shift values necessary to adjust the phases between the reference clock (CLKA) and the derived clocks (GLA, GLB, GLC, YB, and YC). SmartGen also allows the user to select where the input clock is coming from. SmartGen automatically instantiates the special macro, PLLINT, when needed.

CCC and PLL Characteristics

Timing Characteristics

Table 2-12 • **Fusion CCC/PLL Specification**

Parameter		Min.	Typ.	Max.	Unit
Clock Conditioning Circuitry Input Frequency f_{IN_CCC}		1.5		350	MHz
Clock Conditioning Circuitry Output Frequency f_{OUT_CCC}		0.75		350	MHz
Delay Increments in Programmable Delay Blocks[1, 2]			160		ps
Number of Programmable Values in Each Programmable Delay Block				32	
Input Period Jitter				1.5	ns
CCC Output Peak-to-Peak Period Jitter F_{CCC_OUT}		Max Peak-to-Peak Period Jitter			
		1 Global Network Used		3 Global Networks Used	
0.75 MHz to 24 MHz		1.00%		1.00%	
24 MHz to 100 MHz		1.50%		1.50%	
100 MHz to 250 MHz		2.25%		2.25%	
250 MHz to 350 MHz		3.50%		3.50%	
Acquisition Time	LockControl = 0			300	µs
	LockControl = 1			6.0	ms
Tracking Jitter [3]	LockControl = 0			1.6	ns
	LockControl = 1			0.8	ns
Output Duty Cycle		48.5		51.5	%
Delay Range in Block: Programmable Delay 1 [1, 2]		0.6		5.56	ns
Delay Range in Block: Programmable Delay 2 [1, 2]		0.025		5.56	ns
Delay Range in Block: Fixed Delay [1, 2]			2.2		ns

Notes:

1. *This delay is a function of voltage and temperature. See Table 3-7 on page 3-9 for deratings.*
2. *T_J = 25°C, V_{CC} = 1.5 V*
3. *Tracking jitter is defined as the variation in clock edge position of PLL outputs with reference to PLL input clock edge. Tracking jitter does not measure the variation in PLL output period, which is covered by period jitter parameter.*

No-Glitch MUX (NGMUX)

Positioned downstream from the PLL/CCC blocks, the NGMUX provides a special switching sequence between two asynchronous clock domains that prevents generating any unwanted narrow clock pulses. The NGMUX is used to switch the source of a global between three different clock sources. Allowable inputs are either two PLL/CCC outputs or a PLL/CCC output and a regular net, as shown in Figure 2-24. The GLMUXCFG[1:0] configuration bits determine the source of the CLK inputs (i.e., internal signal or GLC). These are set by SmartGen during design but can also be changed by dynamically reconfiguring the PLL. The GLMUXSEL[1:0] bits control which clock source is passed through the NGMUX to the global network (GL). See Table 2-13.

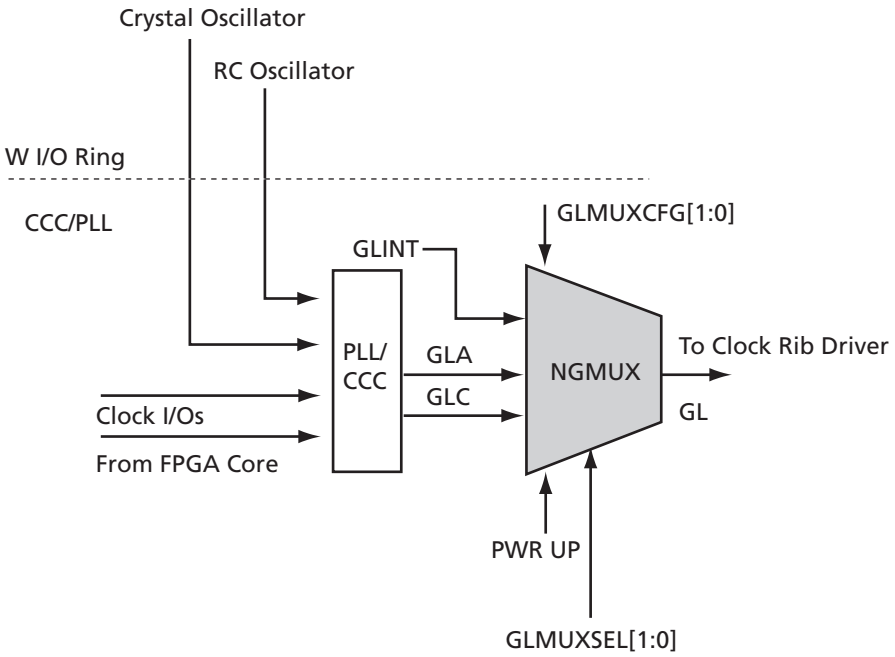


Figure 2-24 • **NGMUX**

Table 2-13 • **NGMUX Configuration and Selection Table**

GLMUXCFG[1:0]	GLMUXSEL[1:0]		Selected Input Signal	MUX Type
00	X	0	GLA	2-to-1 GLMUX
	X	1	GLC	
01	X	0	GLA	2-to-1 GLMUX
	X	1	GLINT	

The NGMUX macro is simplified to show the two clock options that have been selected by the GLMUXCFG[1:0] bits. Figure 2-25 illustrates the NGMUX macro. During design, the two clock sources are connected to CLK0 and CLK1 and are controlled by GLMUXSEL[1:0] to determine which signal is to be passed through the MUX.

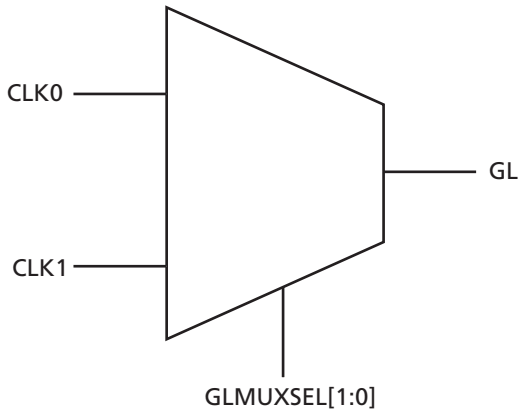


Figure 2-25 • **NGMUX Macro**

The sequence of switching between two clock sources (from CLK0 to CLK1) is as follows (Figure 2-26):

- GLMUXSEL[1:0] transitions to initiate a switch.
- GL drives one last complete CLK0 positive pulse (i.e., one rising edge followed by one falling edge).
- From that point, GL stays LOW until the second rising edge of CLK1 occurs.
- At the second CLK1 rising edge, GL will begin to continuously deliver the CLK1 signal.
- Minimum t_{sw} = 0.05 ns at 25°C (typical conditions)

For examples of NGMUX operation, refer to the *Fusion Handbook*.

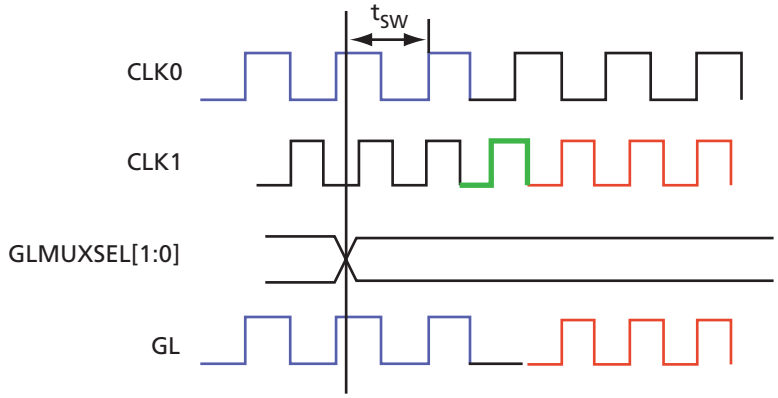


Figure 2-26 • **NGMUX Waveform**

Real-Time Counter System

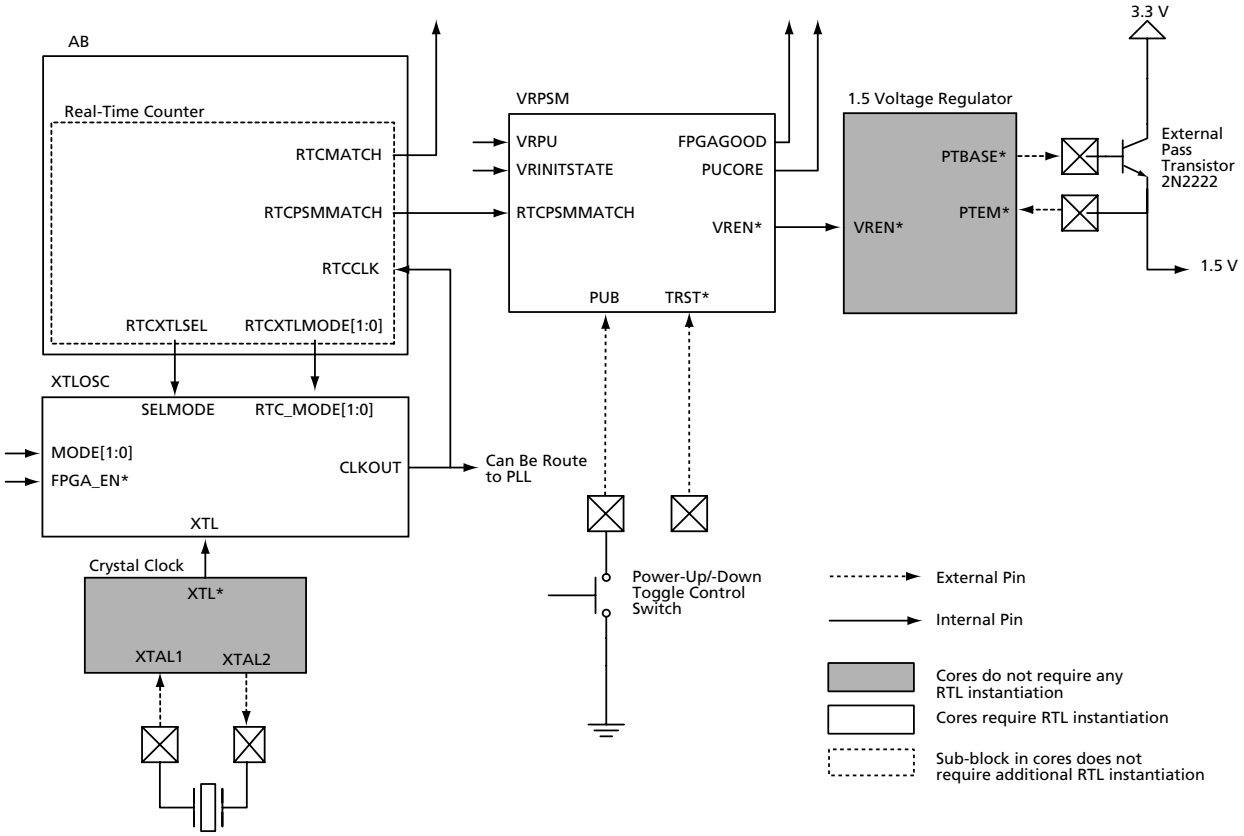
The RTC system enables Fusion devices to support standby and sleep modes of operation to reduce power consumption in many applications.

- Sleep mode, typical 10 µA
- Standby mode (RTC running), typical 3 mA with 20 MHz

The RTC system is composed of five cores:

- RTC sub-block inside Analog Block (AB)
- Voltage Regulator and Power System Monitor (VRPSM)
- Crystal oscillator (XTLOSC); refer to the "Crystal Oscillator" section in *Fusion Clock Resources* for more detail.
- Crystal clock; does not require instantiation in RTL
- 1.5 V voltage regulator; does not require instantiation in RTL

All cores are powered by 3.3 V supplies, so the RTC system is operational without a 1.5 V supply during standby mode. Figure 2-27 shows their connection.



*Note: *Signals are hardwired internally and do not exist in the macro core.*

***Figure 2-27* • Real-Time Counter System (not all the signals are shown for the AB macro)**

Modes of Operation

Standby Mode

Standby mode allows periodic power-up and power-down of the FPGA fabric. In standby mode, the real-time counter and crystal block are ON. The FPGA is not powered by disabling the 1.5 V voltage regulator. The 1.5 V voltage regulator can be enabled when the preset count is matched. Refer to the "Real-Time Counter (part of AB macro)" section for details. To enter standby mode, the RTC must be first configured and enabled. Then VRPSM is shut off by deasserting the VRPU signal. The 1.5 V voltage regulator is then disabled, and shuts off the 1.5 V output.

Sleep Mode

In sleep mode, the real-time counter and crystal blocks are OFF. The 1.5 V voltage regulator inside the VRPSM can only be enabled by the PUB or TRST pin. Refer to the "Voltage Regulator and Power System Monitor (VRPSM)" section on page 2-38 for details on power-up and power-down of the 1.5 V voltage regulator.

Standby and Sleep Mode Circuit Implementation

For extra power savings, V_{JTAG} and V_{PP} should be at the same voltage as V_{CC}, floated or ground, during standby and sleep modes. Note that when V_{JTAG} is not powered, the 1.5 V voltage regulator cannot be enabled through TRST.

V_{PP} and V_{JTAG} can control through an external switch. Actel recommends ADG839, ADG849, or ADG841 as possible switches. Figure 2-28 shows the implementation for controlling V_{PP}. The IN signal of the switch can be connected to PTBASE of the Fusion device. V_{JTAG} can be controlled in same manner.

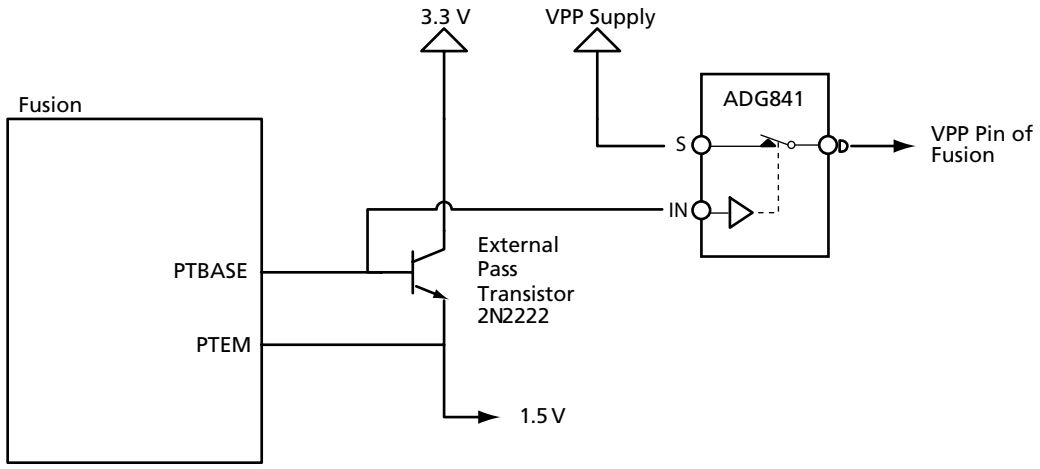


Figure 2-28 • **Implementation to Control V_{PP}**

Real-Time Counter (part of AB macro)

The RTC is a 40-bit loadable counter and used as the primary timekeeping element (Figure 2-29). The clock source, RTCCLK, must come from the CLKOUT signal of the crystal oscillator. The RTC can be configured to reset itself when a count value reaches the match value set in the Match Register.

The RTC is part of the Analog Block (AB) macro. The RTC is configured by the analog configuration MUX (ACM). Each address contains one byte of data. The circuitry in the RTC is powered by V_{CC33A}, so the RTC can be used in standby mode when the 1.5 V supply is not present.

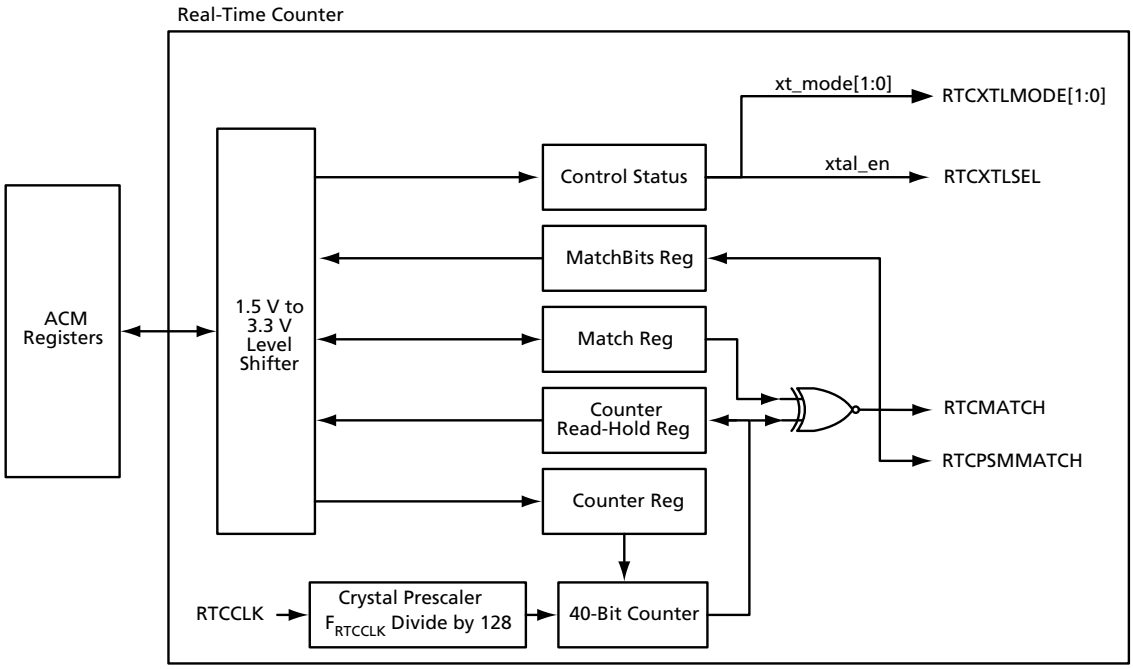


Figure 2-29 • **RTC Block Diagram**

Table 2-14 • **RTC Signal Description**

Signal Name	Width	Direction	Function
RTCCLK	1	In	Must come from CLKOUT of XTLOSC.
RTCXTLMODE[1:0]	2	Out	Controlled by xt_mode in CTRL_STAT. Signal must connect to the RTC_MODE signal in XTLOSC, as shown in Figure 2-27.
RTCXTLSEL	1	Out	Controlled by xtal_en from CTRL_STAT register. Signal must connect to RTC_MODE signal in XTLOSC in Figure 2-27.
RTCMATCH	1	Out	Match signal for FPGA 0 – Counter value does not equal the Match Register value. 1 – Counter value equals the Match Register value.
RTCPSMMATCH	1	Out	Same signal as RTCMATCH. Signal must connect to RTCPSMMATCH in VRPSM, as shown in Figure 2-27.

The 40-bit counter can be preloaded with an initial value as a starting point by the Counter Register. The count from the 40-bit counter can be read through the same set of address space. The count comes from a Read-Hold Register to avoid data changing during read.

When the counter value equals the Match Register value, all Match Bits Register values will be 0xFFFFFFFFFF. The RTCMATCH and RTCPSMMATCH signals will assert. The 40-bit counter can be configured to automatically reset to 0x0000000000 when the counter value equals the Match Register value. The automatic reset does not apply if the Match Register value is 0x0000000000.

The RTCCLK has a prescaler to divide the clock by 128 before it is used for the 40-bit counter. Below is an example of how to calculate the OFF time.

Example: Calculation for Match Count

To put the Fusion device on standby for one hour using an external crystal of 32.768 KHz:

The period of the crystal oscillator is $T_{crystal}$:

$T_{crystal}$ = 1 / 32.768 KHz = 30.518 µs

The period of the counter is $T_{counter}$:

$T_{counter}$ = 30.518 us X 128 = 3.90625 ms

The Match Count for 1 hour is Δtmatch:

Δtmatch / $T_{counter}$ = (1 hr X 60 min/hr X 60 sec/min) / 3.90625 ms = 921600 or 0xE1000

Using a 32.768 KHz crystal, the maximum standby time of the 40-bit counter is 4,294,967,296 seconds, which is 136 years.

Table 2-15 • **Memory Map for RTC in ACM Register and Description**

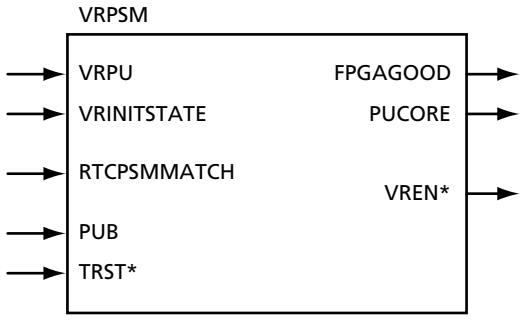
ACMADDR	Register Name	Description	Use	Default Value
0x40	COUNTER0	Counter bits 7:0	Used to preload the counter to a specified start point.	0x00
0x41	COUNTER1	Counter bits 15:8		0x00
0x42	COUNTER2	Counter bits 23:16		0x00
0x43	COUNTER3	Counter bits 31:24		0x00
0x44	COUNTER4	Counter bits 39:32		0x00
0x48	MATCHREG0	Match register bits 7:0	The RTC comparison bits	0x00
0x49	MATCHREG1	Match register bits 15:8		0x00
0x4A	MATCHREG2	Match register bits 23:16		0x00
0x4B	MATCHREG3	Match register bits 31:24		0x00
0x4C	MATCHREG4	Match register bits 39:32		0x00
0x50	MATCHBIT0	Individual match bits 7:0	The output of the XNOR gates 0 – Not matched 1 – Matched	0x00
0x51	MATCHBIT1	Individual match bits 15:8		0x00
0x52	MATCHBIT2	Individual match bits 23:16		0x00
0x53	MATCHBIT3	Individual match bits 31:24		0x00
0x54	MATCHBIT4	Individual match bits 29:32		0x00
0x58	CTRL_STAT	Control (write/read) / Status (read only) register bits	Refer to Table 2-16 on page 2-38 for details.	0x00

Table 2-16 • **RTC Control/Status Register**

Bit	Name	Description	Default Value
7	rtc_rst	RTC Reset 1 – Resets the RTC 0 – Deassert reset on after two ACM_CLK cycle.	
6	cntr_en	Counter Enable 1 – Enables the counter; rtc_rst must be deasserted as well. First counter increments after 64 RTCCLK positive edges. 0 – Disables the crystal prescaler but does not reset the counter value. Counter value can only be updated when the counter is disabled.	0
5	vr_en_mat	Voltage Regulator Enable on Match 1 – Enables RTCMATCH and RTCPSMMATCH to output 1 when the counter value equals the Match Register value. This enables the 1.5 V voltage regulator when RTCPSMMATCH connects to the RTCPSMMATCH signal in VRPSM. 0 – RTCMATCH and RTCPSMMATCH output 0 at all times.	0
4:3	xt_mode[1:0]	Crystal Mode Controls RTCXTLMODE[1:0]. Connects to RTC_MODE signal in XTLOSC. XTL_MODE uses this value when xtal_en is 1. See the "Crystal Oscillator" section on page 2-22 for mode configuration.	00
2	rst_cnt_omat	Reset Counter on Match 1 – Enables the sync clear of the counter when the counter value equals the Match Register value. The counter clears on the rising edge of the clock. If all the Match Registers are set to 0, the clear is disabled. 0 – Counter increments indefinitely	0
1	rstb_cnt	Counter Reset, active Low 0 - Resets the 40-bit counter value	0
0	xtal_en	Crystal Enable Controls RTCXTLSEL. Connects to SELMODE signal in XTLOSC. 0 – XTLOSC enables control by FPGA_EN; xt_mode is not used. Sleep mode requires this bit to equal 0. 1 – Enables XTLOSC, XTL_MODE control by xt_mode Standby mode requires this bit to be set to 1. See the "Crystal Oscillator" section on page 2-22 for further details on SELMODE configuration.	0

Voltage Regulator and Power System Monitor (VRPSM)

The VRPSM macro controls the power-up state of the FPGA. The power-up bar (PUB) pin can turn on the voltage regulator when set to 0. TRST can enable the voltage regulator when deasserted, allowing the FPGA to power-up when user want access to JTAG ports. The inputs VRINITSTATE and RTCPSMMATCH come from the flash bits and RTC, and can also power up the FPGA.



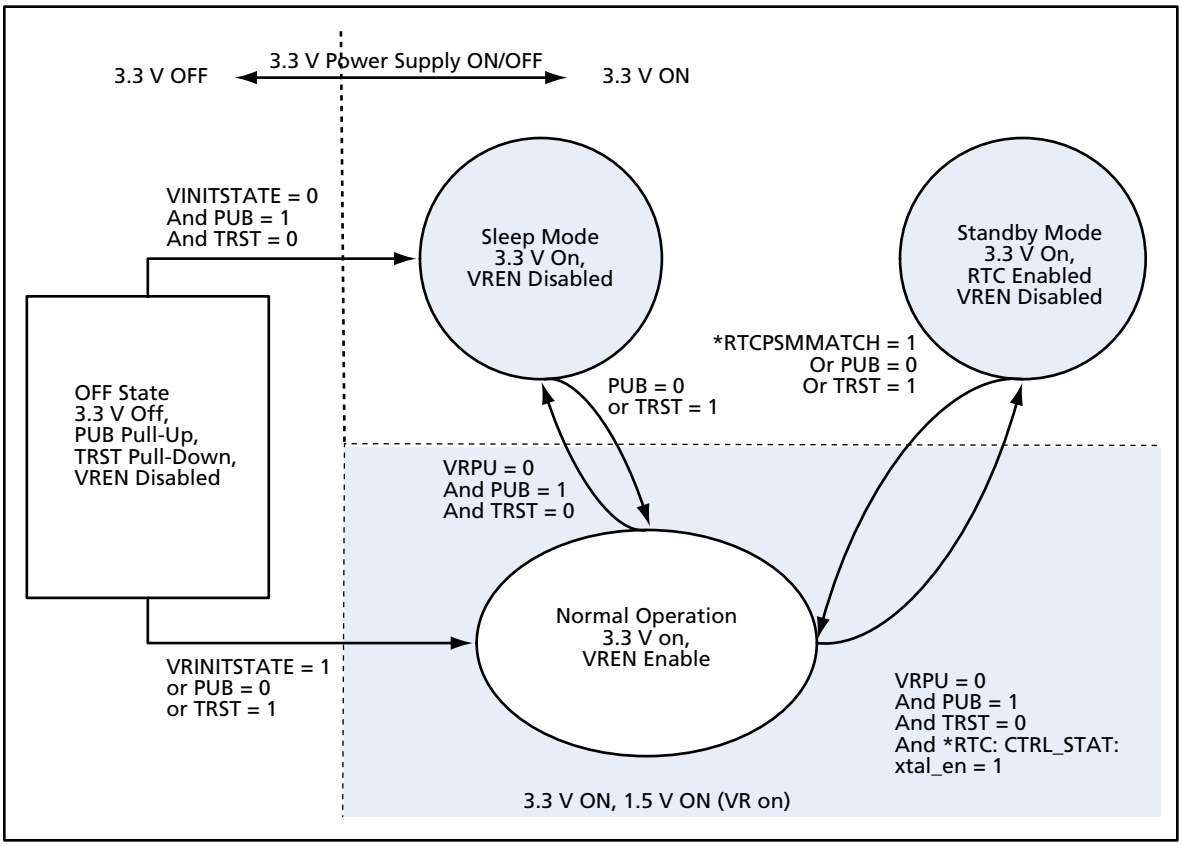
*Note: *Signals are hardwired internally and do not exist in the macro core.*

Figure 2-30 • **VRPSM Macro**

Table 2-17 • **VRPSM Signal Descriptions**

Signal Name	Width	Direction	Function
VRPU	1	In	Voltage Regulator Power-Up 0 – Voltage regulator disabled. PUB must be floated or pulled up, and the TRST pin must be grounded to disable the voltage regulator. 1 – Voltage regulator enabled
VRINITSTATE	1	In	Voltage Regulator Initial State Defines the voltage Regulator status upon power-up of the 3.3 V. The signal is configured by Actel Libero® Integrated Design Environment (IDE) when the VRPSM macro is generated. Tie off to 1 – Voltage regulator enables when 3.3 V is powered. Tie off to 0 – Voltage regulator disables when 3.3 V is powered.
RTCPSMMATCH	1	In	RTC Power System Management Match Connect from RTCPSMATCH signal from RTC in AB 0 transition to 1 turns on the voltage regulator
PUB	1	In	External pin, built-in weak pull-up Power-Up Bar 0 – Enables voltage regulator at all times
TRST*	1	In	External pin, JTAG Test Reset 1 – Enables voltage regulator at all times
FPGAGOOD	1	Out	Indicator that the FPGA is powered and functional No need to connect if it is not used. 1 – Indicates that the FPGA is powered up and functional. 0 – Not possible to read by FPGA since it has already powered off.
PUCORE	1	Out	Power-Up Core Inverted signal of PUB. No need to connect if it is not used.
VREN*	1	Out	Voltage Regulator Enable Connected to 1.5 V voltage regulator in Fusion device internally. 0 – Voltage regulator disables 1 – Voltage regulator enables

*Note: *Signals are hardwired internally and do not exist in the macro core.*



*Note: * To enter and exit standby mode without any external stimulus on PUB or TRST, the vr_en_mat in the CTRL_STAT register must also be set to 1, so that RTCPSMMATCH will assert when a match occurs; hence the device exits standby mode.*

Figure 2-31 • **State Diagram for All Different Power Modes**

When TRST is 1 or PUB is 0, the 1.5 V voltage regulator is always ON, putting the Fusion device in normal operation at all times. Therefore, when the JTAG port is not in reset, the Fusion device cannot enter sleep mode or standby mode.

To enter standby mode, the Fusion device must first power-up into normal operation. The RTC is enabled through the RTC Control/Status Register described in the "Real-Time Counter (part of AB macro)" section on page 2-35. A match value corresponding to the wake-up time is loaded into the Match Register. The 1.5 V voltage regulator is disabled by setting VRPU to 0 to allow the Fusion device to enter standby mode, when the 1.5 V supply is off but the RTC remains on.

1.5 V Voltage Regulator

The 1.5 V voltage regulator uses an external pass transistor to generate 1.5 V from a 3.3 V supply. The base of the pass transistor is tied to PTBASE, the collector is tied to 3.3 V, and an emitter is tied to PTBASE and the 1.5 V supplies of the Fusion device. Figure 2-27 on page 2-34 shows the hook-up of the 1.5 V voltage regulator to an external pass transistor.

Actel recommends using a PN2222A or 2N2222A transistor. The gain of such a transistor is approximately 25, with a maximum base current of 20 mA. The maximum current that can be supported is 0.5 A. Transistors with different gain can also be used for different current requirements.

Table 2-18 • **Electrical Characteristics**
V_{CC33A} = 3.3 V

Symbol	Parameter	Condition		Min	Typical	Max	Units
V_{OUT}	Output Voltage	Tj = 25°C		1.425	1.5	1.575	V
I_{CC33A}	Operation Current	Tj = 25°C	I_{LOAD} = 1 mA		11		mA
			I_{LOAD} = 100 mA		11		mA
			I_{LOAD} = 0.5 A		30		mA
Δ_{VOUT}	Load Regulation	Tj = 25°C	I_{LOAD} = 1 mA to 0.5 A		90		mV
Δ_{VOUT}	Line Regulation	Tj = 25°C	VCC33A = 2.97 V to 3.63 V I_{LOAD} = 1 mA		10.6		mV/V
			VCC33A = 2.97 V to 3.63 V I_{LOAD} = 100 mA		12.1		mV/V
			VCC33A = 2.97 V to 3.63 V I_{LOAD} = 500 mA		10.6		mV/V
	Dropout Voltage*	Tj = 25°C	I_{LOAD} = 1 mA		0.63		V
			I_{LOAD} = 100 mA		0.84		V
			I_{LOAD} = 0.5 A		1.35		V
I_{PTBASE}	PTBase Current	Tj = 25°C	I_{LOAD} = 1 mA		48		µA
			I_{LOAD} = 100 mA		736		µA
			I_{LOAD} = 0.5 A		12	20	mA

*Note: *Data collected with 2N2222A.*

Embedded Memories

Fusion devices include four types of embedded memory: flash block, FlashROM, SRAM, and FIFO.

Flash Memory Block

Fusion is the first FPGA that offers a flash memory block (FB). Each FB block stores 2 Mbits of data. The flash memory block macro is illustrated in Figure 2-32. The port pin name and descriptions are detailed on Table 2-19 on page 2-43. All flash memory block signals are active high, except for CLK and active low RESET. All flash memory operations are synchronous to the rising edge of CLK.

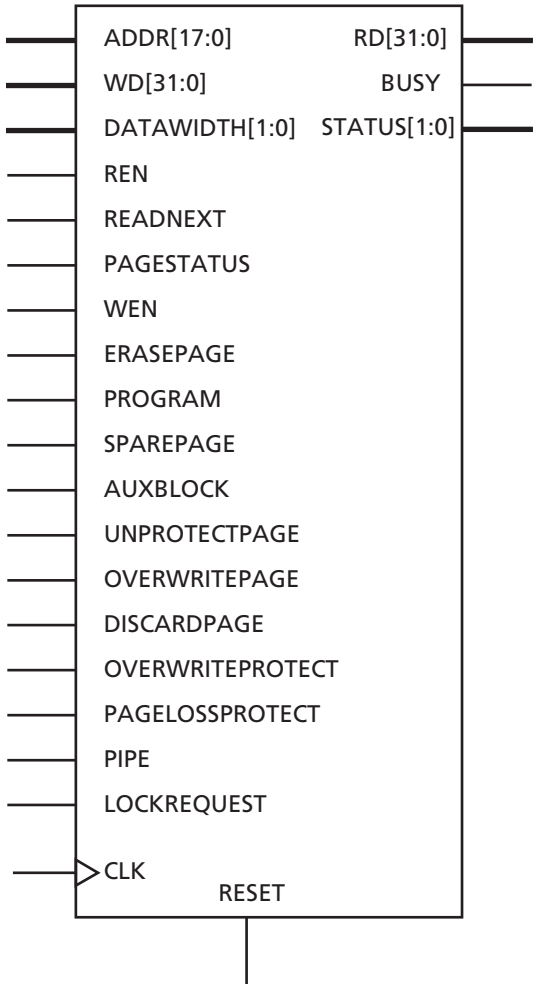


Figure 2-32 • **Flash Memory Block**

Flash Memory Block Pin Names

Table 2-19 • **Flash Memory Block Pin Names**

Interface Name	Width	Direction	Description
ADDR[17:0]	18	In	Byte offset into the FB. Byte-based address.
AUXBLOCK	1	In	When asserted, the page addressed is used to access the auxiliary block within that page.
BUSY	1	Out	When asserted, indicates that the FB is performing an operation.
CLK	1	In	User interface clock. All operations and status are synchronous to the rising edge of this clock.
DATAWIDTH[1:0]	2	In	Data width 00 = 1 byte in RD/WD[7:0] 01 = 2 bytes in RD/WD[15:0] 1x = 4 bytes in RD/WD[31:0]
DISCARDPAGE	1	In	When asserted, the contents of the Page Buffer are discarded so that a new page write can be started.
ERASEPAGE	1	In	When asserted, the address page is to be programmed with all zeros. ERASEPAGE must transition synchronously with the rising edge of CLK.
LOCKREQUEST	1	In	When asserted, indicates to the JTAG controller that the FPGA interface is accessing the FB.
OVERWRITEPAGE	1	In	When asserted, the page addressed is overwritten with the contents of the Page Buffer if the page is writable.
OVERWRITEPROTECT	1	In	When asserted, all program operations will set the overwrite protect bit of the page being programmed.
PAGESTATUS	1	In	When asserted with REN, initiates a read page status operation.
PAGELOSSPROTECT	1	In	When asserted, a modified Page Buffer must be programmed or discarded before accessing a new page.
PIPE	1	In	Adds a pipeline stage to the output for operation above 50 MHz.
PROGRAM	1	In	When asserted, writes the contents of the Page Buffer into the FB page addressed.
RD[31:0]	32	Out	Read data; data will be valid from the first non-busy cycle (BUSY = 0) after REN has been asserted.
READNEXT	1	In	When asserted with REN, initiates a read-next operation.
REN	1	In	When asserted, initiates a read operation.
RESET	1	In	When asserted, resets the state of the FB (active low).
SPAREPAGE	1	In	When asserted, the sector addressed is used to access the spare page within that sector.

Table 2-19 • **Flash Memory Block Pin Names (continued)**

Interface Name	Width	Direction	Description
STATUS[1:0]	2	Out	Status of the last operation completed: 00: Successful completion 01: Read-/Unprotect-Page: single error detected and corrected Write: operation addressed a write-protected page Erase-Page: protection violation Program: Page Buffer is unmodified Protection violation 10: Read-/Unprotect-Page: two or more errors detected 11: Write: attempt to write to another page before programming current page Erase-Page/Program: page write count has exceeded the 10-year retention threshold
UNPROTECTPAGE	1	In	When asserted, the page addressed is copied into the Page Buffer and the Page Buffer is made writable.
WD[31:0]	32	In	Write data
WEN	1	In	When asserted, stores WD in the page buffer.

All flash memory block input signals are active high, except for RESET.

Flash Memory Block Diagram

A simplified diagram of the flash memory block is shown in Figure 2-33.

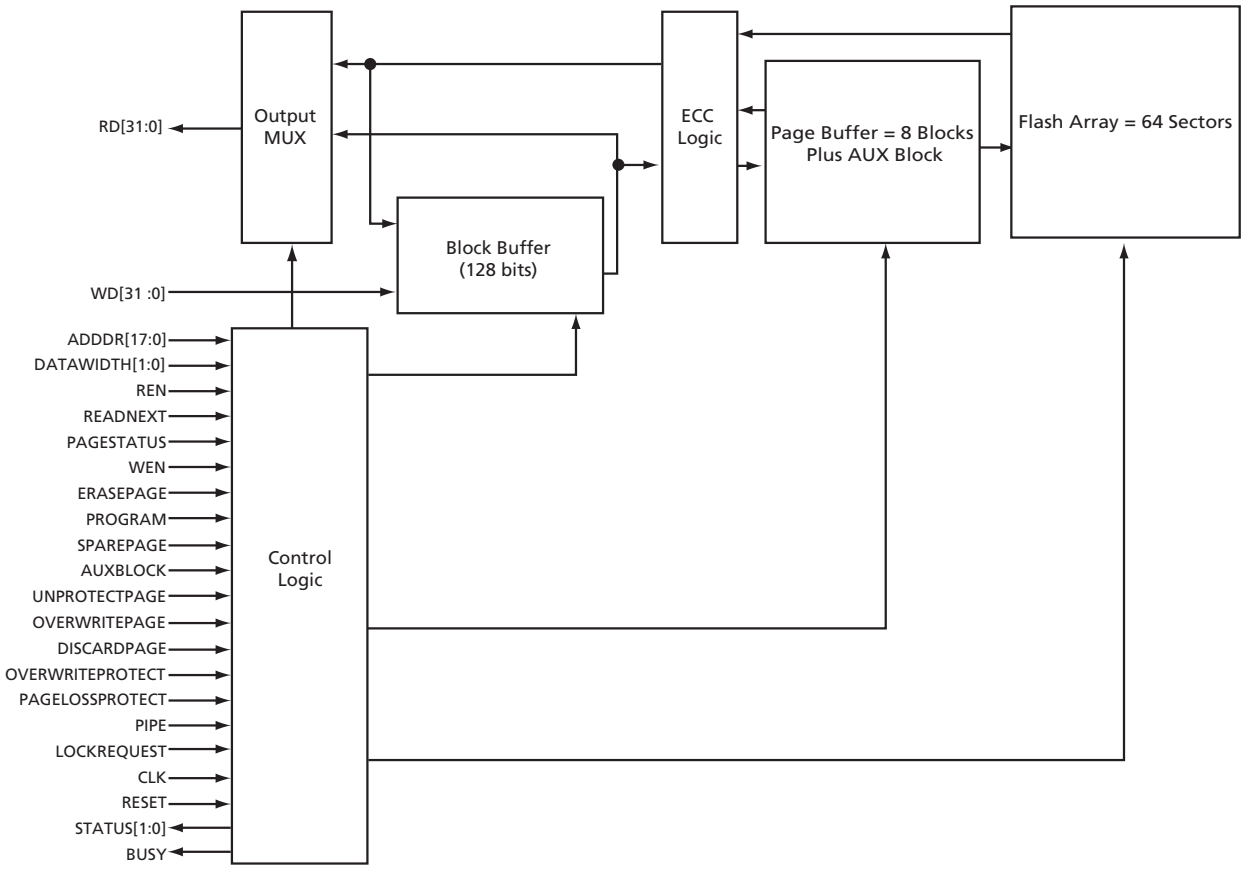


***Figure 2-33* • Flash Memory Block Diagram**

The logic consists of the following sub-blocks:

- Flash Array

  Contains all stored data. The flash array contains 64 sectors, and each sector contains 33 pages of data.
- Page Buffer

  A page-wide volatile register. A page contains 8 blocks of data and an AUX block.
- Block Buffer

  Contains the contents of the last block accessed. A block contains 128 data bits.
- ECC Logic

  The FB stores error correction information with each block to perform single-bit error correction and double-bit error detection on all data blocks.

Flash Memory Block Addressing

Figure 2-34 shows a graphical representation of the flash memory block.

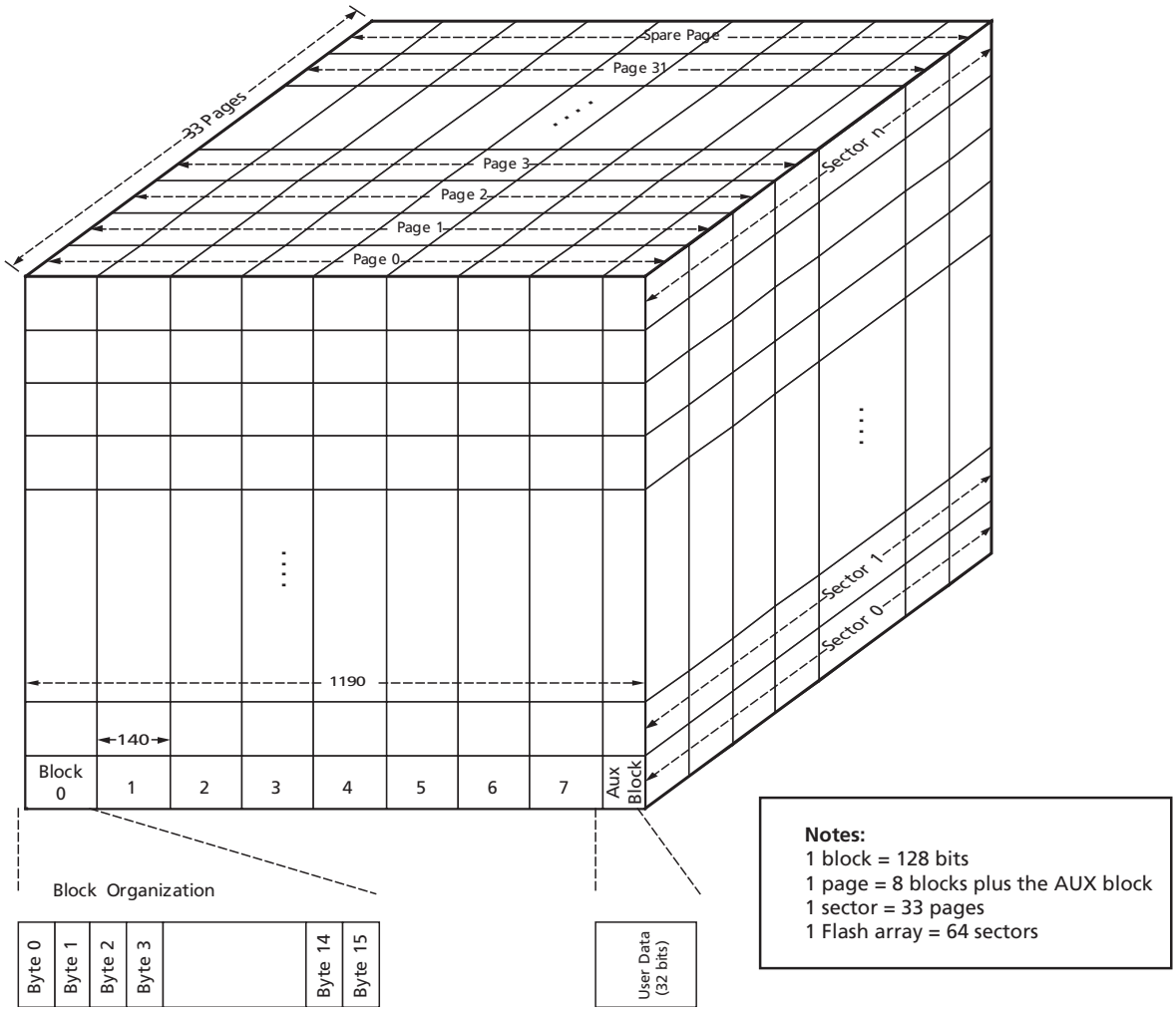


Figure 2-34 • **Flash Memory Block Organization**

Each FB is partitioned into sectors, pages, blocks, and bytes. There are 64 sectors in an FB, and each sector contains 32 pages and 1 spare page. Each page contains 8 data blocks and 1 auxiliary block. Each data block contains 16 bytes of user data, and the auxiliary block contains 4 bytes of user data.

Addressing for the FB is shown in Table 2-20.

Table 2-20 • **FB Address Bit Allocation ADDR[17:0]**

17	12	11	7	6	4	3	0
Sector		Page		Block		Byte	

When the spare page of a sector is addressed (SPAREPAGE active), ADDR[11:7] are ignored.

When the Auxiliary block is addressed (AUXBLOCK active), ADDR[6:2] are ignored.

Note: The spare page of sector 0 is unavailable for any user data. Writes to this page will return an error, and reads will return all zeroes.

Data operations are performed in widths of 1 to 4 bytes. A write to a location in a page that is not already in the Page Buffer will cause the page to be read from the FB Array and stored in the Page Buffer. The block that was addressed during the write will be put into the Block Buffer, and the data written by WD will overwrite the data in the Block Buffer. After the data is written to the Block Buffer, the Block Buffer is then written to the Page Buffer to keep both buffers in sync. Subsequent writes to the same block will overwrite the Block Buffer and the Page Buffer. A write to another block in the page will cause the addressed block to be loaded from the Page Buffer, and the write will be performed as described previously.

The data width can be selected dynamically via the DATAWIDTH input bus. The truth table for the data width settings is detailed in Table 2-21. The minimum resolvable address is one 8-bit byte. For data widths greater than 8 bits, the corresponding address bits are ignored—when DATAWIDTH = 0 (2 bytes), ADDR[0] is ignored, and when DATAWIDTH = '10' or '11' (4 bytes), ADDR[1:0] are ignored. Data pins are LSB-oriented and unused WD data pins must be grounded.

Table 2-21 • **Data Width Settings**

DATAWIDTH[1:0]	Data Width
00	1 byte [7:0]
01	2 byte [15:0]
10, 11	4 bytes [31:0]

Flash Memory Block Protection

Page Loss Protection

When the PAGELOSSPROTECT pin is set to logic 1, it prevents writes to any page other than the current page in the Page Buffer until the page is either discarded or programmed.

A write to another page while the current page is Page Loss Protected will return a STATUS of '11'.

Overwrite Protection

Any page that is Overwrite Protected will result in the STATUS being set to '01' when an attempt is made to either write, program, or erase it. To set the Overwrite Protection state for a page, set the OVERWRITEPROTECT pin when a Program operation is undertaken. To clear the Overwrite Protect state for a given page, an Unprotect Page operation must be performed on the page, and then the page must be programmed with the OVERWRITEPROTECT pin cleared to save the new page.

LOCKREQUEST

The LOCKREQUEST signal is used to give the user interface control over simultaneous access of the FB from both the User and JTAG interfaces. When LOCKREQUEST is asserted, the JTAG interface will hold off any access attempts until LOCKREQUEST is deasserted.

Flash Memory Block Operations

FB Operation Priority

The FB provides for priority of operations when multiple actions are requested simultaneously. Table 2-22 shows the priority order (priority 0 is the highest).

Table 2-22 • **FB Operation Priority**

Operation	Priority
System Initialization	0
FB Reset	1
Read	2
Write	3
Erase Page	4
Program	5
Unprotect Page	6
Discard Page	7

Access to the FB is controlled by the BUSY signal. The BUSY output is synchronous to the CLK signal. FB operations are only accepted in cycles where BUSY is logic 0.

Write Operation

Write operations are initiated with the assertion of the WEN signal. Figure 2-35 on page 2-48 illustrates the multiple Write operations.

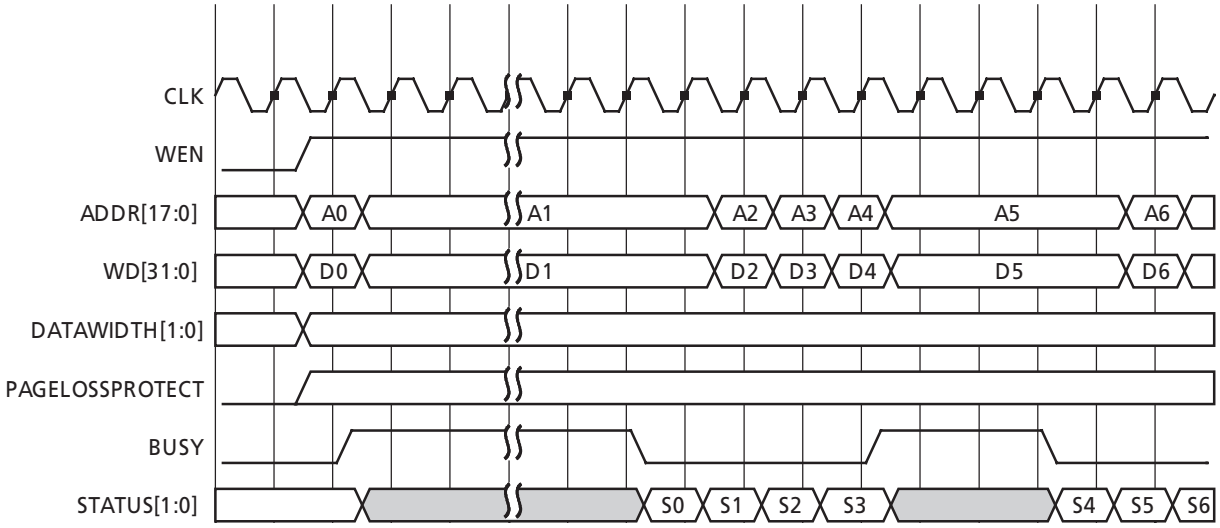


Figure 2-35 • **FB Write Waveform**

When a Write operation is initiated to a page that is currently not in the Page Buffer, the FB control logic will issue a BUSY signal to the user interface while the page is loaded from the FB Array into the Page Buffer. (Note: The number of clock cycles that the BUSY output is asserted during the load of the Page Buffer is variable.) After loading the page into the Page Buffer, the addressed data block is loaded from the Page Buffer into the Block Buffer. Subsequent writes to the same block of the page will incur no busy cycles. A write to another block in the page will assert BUSY for four cycles (five cycles when PIPE is asserted), to allow the data to be written to the Page Buffer and have the current block loaded into the Block Buffer.

Write operations are considered successful as long as the STATUS output is '00'. A non-zero STATUS indicates that an error was detected during the operation and the write was not performed. Note that the STATUS output is "sticky"; it is unchanged until another operation is started.

Only one word can be written at a time. Write word width is controlled by the DATAWIDTH bus. Users are responsible for keeping track of the contents of the Page Buffer and when to program it to the array. Just like a regular RAM, writing to random addresses is possible. Users can write into the Page Buffer in any order but will incur additional BUSY cycles. It is not necessary to modify the entire Page Buffer before saving it to nonvolatile memory.

Write errors include the following:

1. Attempting to write a page that is Overwrite Protected (STATUS = '01'). The write is not performed.
2. Attempting to write to a page that is not in the Page Buffer when Page Loss Protection is enabled (STATUS = '11'). The write is not performed.

Program Operation

A Program operation is initiated by asserting the PROGRAM signal on the interface. Program operations save the contents of the Page Buffer to the FB Array. Due to the technologies inherent in the FB, a program operation is a time consuming operation (~8 ms). While the FB is writing the data to the array, the BUSY signal will be asserted.

During a Program operation, the sector and page addresses on ADDR are compared with the stored address for the page (and sector) in the Page Buffer. If there is a mismatch between the two addresses, the Program operation will be aborted and an error will be reported on the STATUS output.

It is possible to write the Page Buffer to a different page in memory. When asserting the PROGRAM pin, if OVERWRITEPAGE is asserted as well, the FB will write the contents of the Page Buffer to the sector and page designated on the ADDR inputs if the destination page is not Overwrite Protected.

A Program operation can be utilized to either modify the contents of the page in the flash memory block or change the protections for the page. Setting the OVERWRITEPROTECT bit on the interface while asserting the PROGRAM pin will put the page addressed into Overwrite Protect Mode. Overwrite Protect Mode safeguards a page from being inadvertently overwritten during subsequent Program or Erase operations.

Program operations that result in a STATUS value of '01' do not modify the addressed page. For all other values of STATUS, the addressed page is modified.

Program errors include the following:

1. Attempting to program a page that is Overwrite Protected (STATUS = '01')
2. Attempting to program a page that is not in the Page Buffer when the Page Buffer has entered Page Loss Protection Mode (STATUS = '01')
3. Attempting to perform a program with OVERWRITEPAGE set when the page addressed has been Overwrite Protected (STATUS = '01')
4. The Write Count of the page programmed exceeding the Write Threshold defined in the part specification (STATUS = '11')
5. The ECC Logic determining that there is an uncorrectable error within the programmed page (STATUS = '10')
6. Attempting to program a page that is **not** in the Page Buffer when OVERWRITEPAGE is not set and the page in the Page Buffer is modified (STATUS = '01')
7. Attempting to program the page in the Page Buffer when the Page Buffer is **not** modified

The waveform for a Program operation is shown in Figure 2-36.

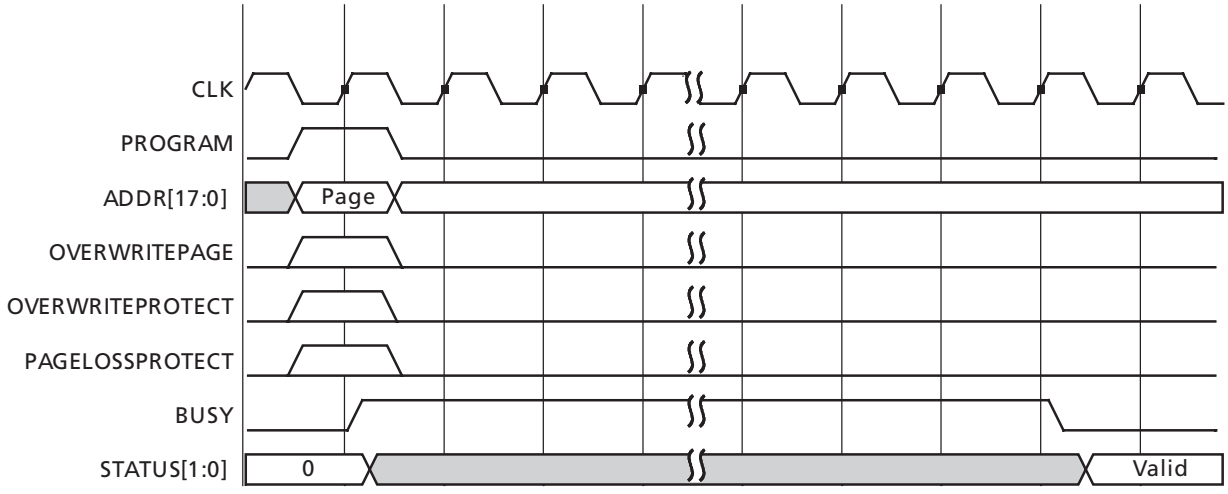


Figure 2-36 • **FB Program Waveform**

Note: OVERWRITEPAGE is only sampled when the PROGRAM or ERASEPAGE pins are asserted. OVERWRITEPAGE is ignored in all other operations.

Erase Page Operation

The Erase Page operation is initiated when the ERASEPAGE pin is asserted. The Erase Page operation allows the user to erase (set user data to zero) any page within the FB.

The use of the OVERWRITEPAGE and PAGELOSSPROTECT pins is the same for erase as for a Program Page operation.

As with the Program Page operation, a STATUS of '01' indicates that the addressed page is not erased.

A waveform for an Erase Page operation is shown in Figure 2-37.

Erase errors include the following:

1. Attempting to erase a page that is Overwrite Protected (STATUS = '01')
2. Attempting to erase a page that is not in the Page Buffer when the Page Buffer has entered Page Loss Protection mode (STATUS = '01')
3. The Write Count of the erased page exceeding the Write Threshold defined in the part specification (STATUS = '11')
4. The ECC Logic determining that there is an uncorrectable error within the erased page (STATUS = '10')

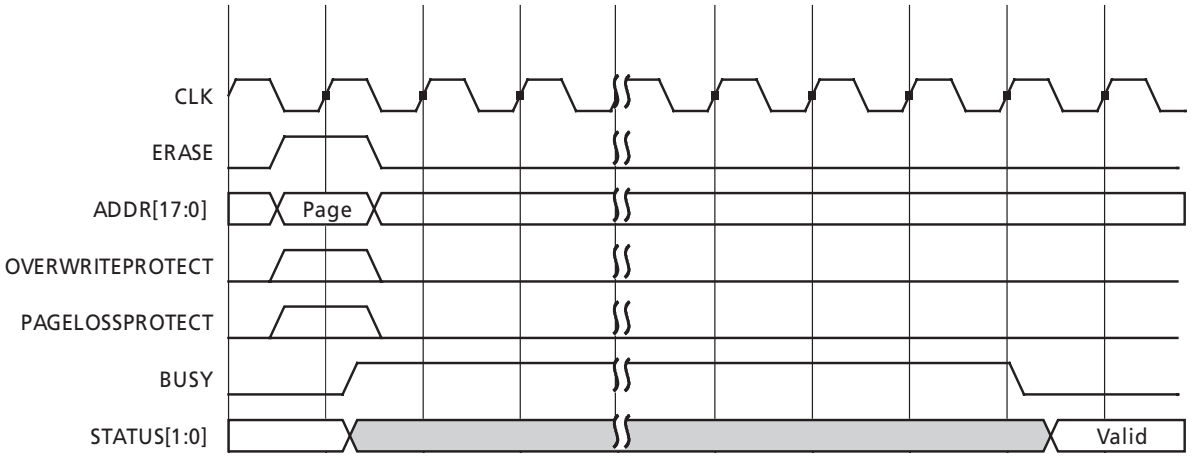


Figure 2-37 • **FB Erase Page Waveform**

Read Operation

Read operations are designed to read data from the FB Array, Page Buffer, Block Buffer, or status registers. Read operations support a normal read and a read-ahead mode (done by asserting READNEXT). Also, the timing for Read operations is dependent on the setting of PIPE.

The following diagrams illustrate representative timing for Non-Pipe Mode (Figure 2-38) and Pipe Mode (Figure 2-39) reads of the flash memory block interface.

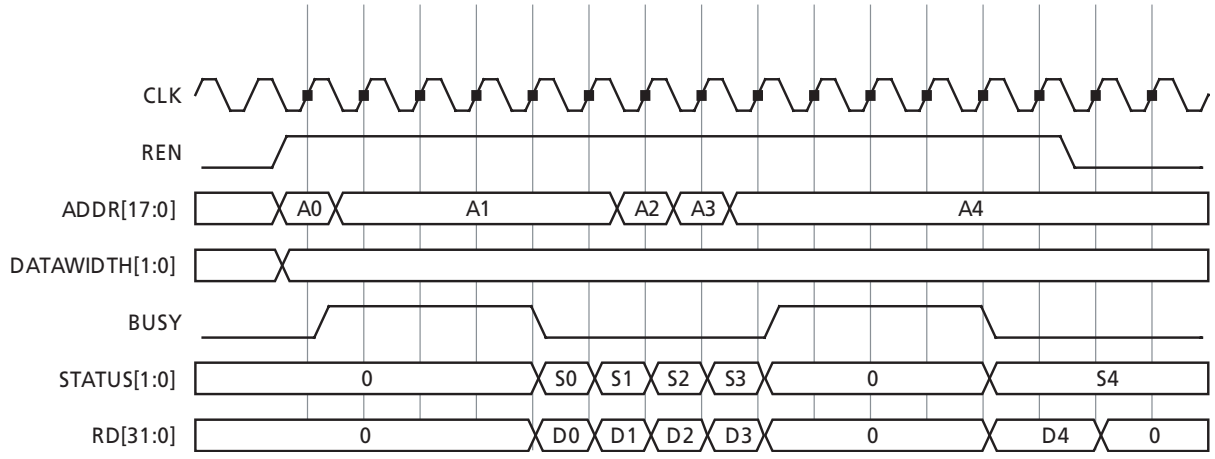


***Figure 2-38* • Read Waveform (Non-Pipe Mode, 32-bit access)**

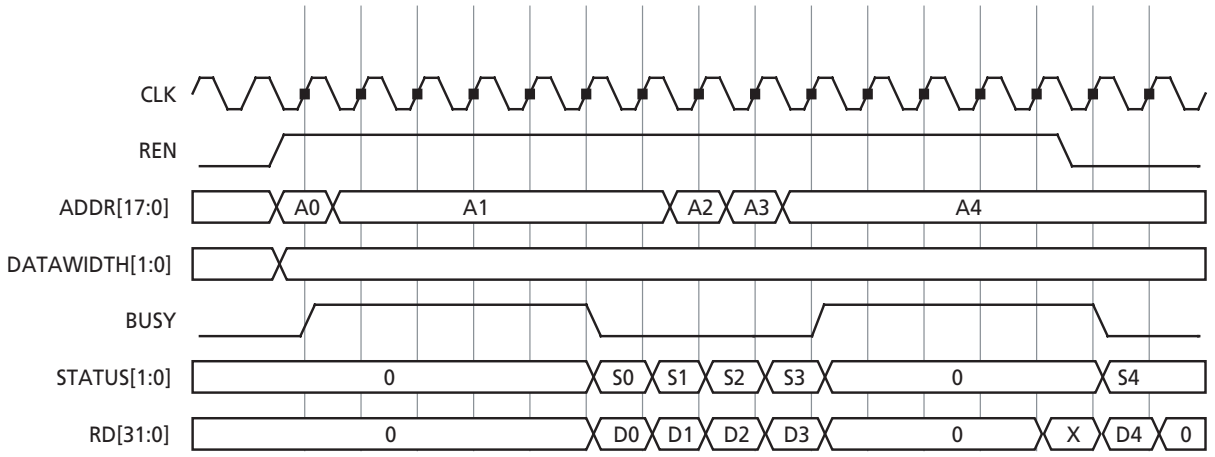


***Figure 2-39* • Read Waveform (Pipe Mode, 32-bit access)**

The following error indications are possible for Read operations:

1. STATUS = '01' when a single-bit data error was detected and corrected within the block addressed.
2. STATUS = '10' when a double-bit error was detected in the block addressed (note that the error is uncorrected).

In addition to data reads, users can read the status of any page in the FB by asserting PAGESTATUS along with REN. The format of the data returned by a page status read is shown in Table 2-23, and the definition of the page status bits is shown in Table 2-24.

Table 2-23 • **Page Status Read Data Format**

31	8	7	4	3	2	1	0
Write Count		Reserved		Over Threshold	Read Protected	Write Protected	Overwrite Protected

Table 2-24 • **Page Status Bit Definition**

Page Status Bit(s)	Definition
31–8	The number of times the page addressed has been programmed/erased
7–4	Reserved; read as 0
3	Over Threshold indicator (see the "Program Operation" section on page 2-48)
2	Read Protected; read protect bit for page, which is set via the JTAG interface and only affects JTAG operations. This bit can be overridden by using the correct user key value.
1	Write Protected; write protect bit for page, which is set via the JTAG interface and only affects JTAG operations. This bit can be overridden by using the correct user key value.
0	Overwrite Protected; designates that the user has set the OVERWRITEPROTECT bit on the interface while doing a Program operation. The page cannot be written without first performing an Unprotect Page operation.

Read Next Operation

The Read Next operation is a feature by which the next block relative to the block in the Block Buffer is read from the FB Array while performing reads from the Block Buffer. The goal is to minimize wait states during consecutive sequential Read operations.

The Read Next operation is performed in a predetermined manner because it does look-ahead reads. The general look-ahead function is as follows:

- Within a page, the next block fetched will be the next in linear address.
- When reading the last data block of a page, it will fetch the first block of the next page.
- When reading spare pages, it will read the first block of the next sector's spare page.
- Reads of the last sector will wrap around to sector 0.
- Reads of Auxiliary blocks will read the next linear page's Auxiliary block.

When an address on the ADDR input does not agree with the predetermined look-ahead address, there is a time penalty for this access. The FB will be busy finishing the current look-ahead read before it can start the next read. The worst case is a total of nine BUSY cycles before data is delivered.

The Non-Pipe Mode and Pipe Mode waveforms for Read Next operations are illustrated in Figure 2-40 and Figure 2-41.

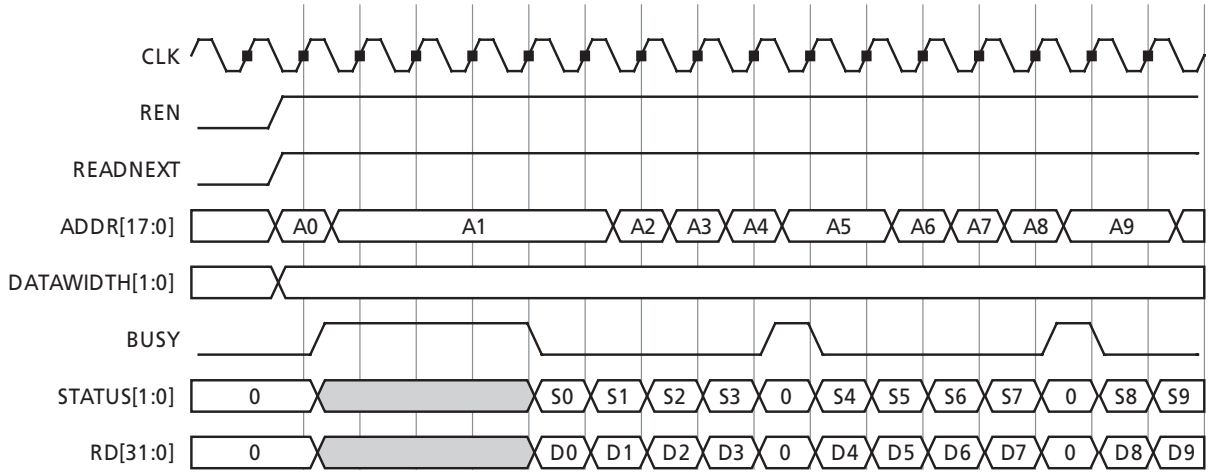


***Figure 2-40* • Read Next Waveform (Non-Pipe Mode, 32-bit access)**

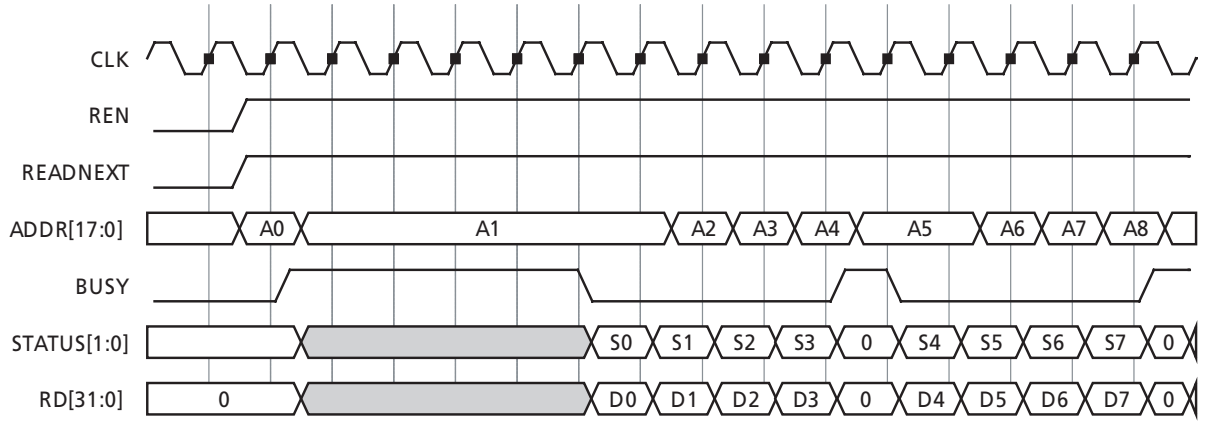


***Figure 2-41* • Read Next WaveForm (Pipe Mode, 32-bit access)**

Unprotect Page Operation

An Unprotect Page operation will clear the protection for a page addressed on the ADDR input. It is initiated by setting the UNPROTECTPAGE signal on the interface along with the page address on ADDR.

If the page is not in the Page Buffer, the Unprotect Page operation will copy the page into the Page Buffer. The Copy Page operation occurs only if the current page in the Page Buffer is not Page Loss Protected.

The waveform for an Unprotect Page operation is shown in Figure 2-42.

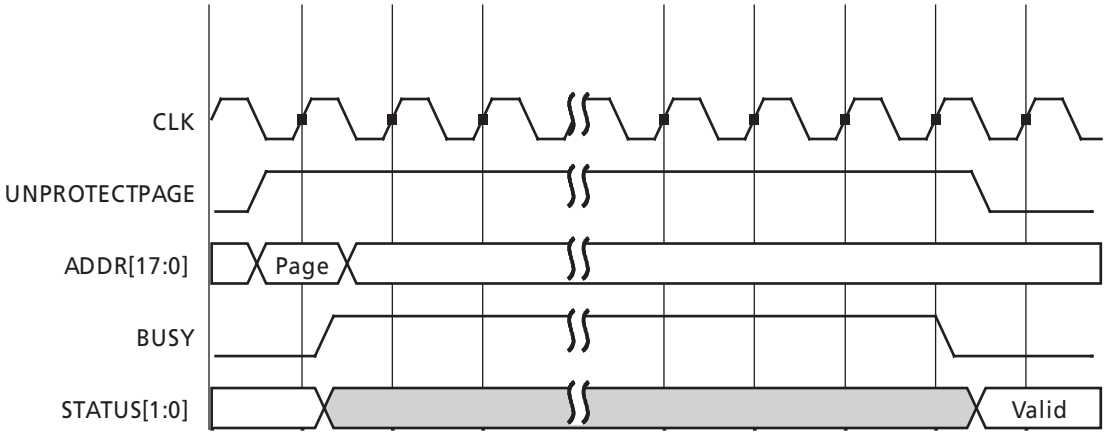


Figure 2-42 • **FB Unprotected Page Waveform**

The Unprotect Page operation can incur the following error conditions:

1. If the copy of the page to the Page Buffer determines that the page has a single-bit correctable error in the data, it will report a STATUS = '01'.
2. If the address on ADDR does not match the address of the Page Buffer, PAGELOSSPROTECT is asserted, and the Page Buffer has been modified, then STATUS = '11' and the addressed page is not loaded into the Page Buffer.
3. If the copy of the page to the Page Buffer determines that at least one block in the page has a double-bit uncorrectable error, STATUS = '10' and the Page Buffer will contain the corrupted data.

Discard Page Operation

If the contents of the modified Page Buffer have to be discarded, the DISCARDPAGE signal should be asserted. This command results in the Page Buffer being marked as unmodified.

The timing for the operation is shown in Figure 2-43. The BUSY signal will remain asserted until the operation has completed.

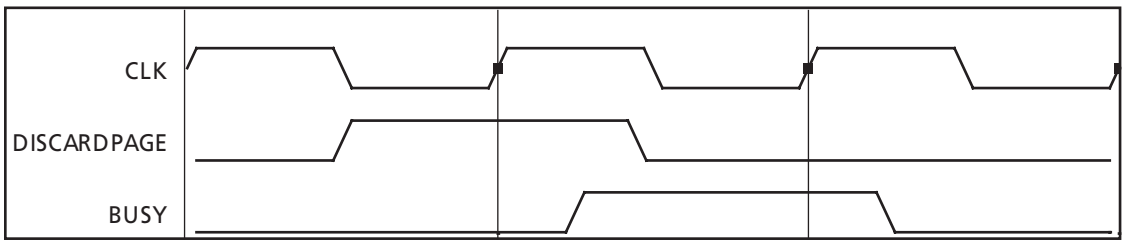


Figure 2-43 • **FB Discard Page Waveform**

Flash Memory Block Characteristics

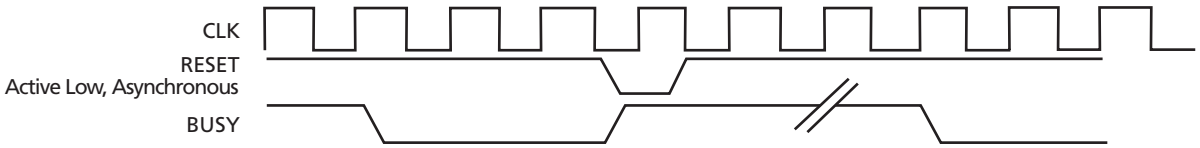


***Figure 2-44* • Reset Timing Diagram**

***Table 2-25* • Flash Memory Block Timing**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{CLK2RD}	Clock-to-Q in 5-cycle read mode of the Read Data	7.99	9.10	10.70	ns
	Clock-to-Q in 6-cycle read mode of the Read Data	5.03	5.73	6.74	ns
$t_{CLK2BUSY}$	Clock-to-Q in 5-cycle read mode of BUSY	4.95	5.63	6.62	ns
	Clock-to-Q in 6-cycle read mode of BUSY	4.45	5.07	5.96	ns
$t_{CLK2STATUS}$	Clock-to-Status in 5-cycle read mode	11.24	12.81	15.06	ns
	Clock-to-Status in 6-cycle read mode	4.48	5.10	6.00	ns
t_{DSUNVM}	Data Input Setup time for the Control Logic	1.92	2.19	2.57	ns
t_{DHNVM}	Data Input Hold time for the Control Logic	0.00	0.00	0.00	ns
t_{ASUNVM}	Address Input Setup time for the Control Logic	2.76	3.14	3.69	ns
t_{AHNVM}	Address Input Hold time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUDWNVM}$	Data Width Setup time for the Control Logic	1.85	2.11	2.48	ns
$t_{HDDWNVM}$	Data Width Hold time for the Control Logic	0.00	0.00	0.00	ns
$t_{SURENNVM}$	Read Enable Setup time for the Control Logic	3.85	4.39	5.16	ns
$t_{HDRENNVM}$	Read Enable Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUWENNVM}$	Write Enable Setup time for the Control Logic	2.37	2.69	3.17	ns
$t_{HDWENNVM}$	Write Enable Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUPROGNVM}$	Program Setup time for the Control Logic	2.16	2.46	2.89	ns
$t_{HDPROGNVM}$	Program Hold time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUSPAREPAGE}$	SparePage Setup time for the Control Logic	3.74	4.26	5.01	ns
$t_{HDSPAREPAGE}$	SparePage Hold time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUAUXBLK}$	Auxiliary Block Setup Time for the Control Logic	3.74	4.26	5.00	ns
$t_{HDAUXBLK}$	Auxiliary Block Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SURDNEXT}$	ReadNext Setup Time for the Control Logic	2.17	2.47	2.90	ns
$t_{HDRDNEXT}$	ReadNext Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUERASEPG}$	Erase Page Setup Time for the Control Logic	3.76	4.28	5.03	ns
$t_{HDERASEPG}$	Erase Page Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUUNPROTECTPG}$	Unprotect Page Setup Time for the Control Logic	2.01	2.29	2.69	ns
$t_{HDUNPROTECTPG}$	Unprotect Page Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUDISCARDPG}$	Discard Page Setup Time for the Control Logic	1.88	2.14	2.52	ns
$t_{HDDISCARDPG}$	Discard Page Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUOVERWRPRO}$	Overwrite Protect Setup Time for the Control Logic	1.64	1.86	2.19	ns

Table 2-25 • **Flash Memory Block Timing (continued)**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
$t_{HDOVERWRPRO}$	Overwrite Protect Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUPGLOSSPRO}$	Page Loss Protect Setup Time for the Control Logic	1.69	1.93	2.27	ns
$t_{HDPGLOSSPRO}$	Page Loss Protect Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUPGSTAT}$	Page Status Setup Time for the Control Logic	2.49	2.83	3.33	ns
$t_{HDPGSTAT}$	Page Status Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SUOVERWRPG}$	Over Write Page Setup Time for the Control Logic	1.88	2.14	2.52	ns
$t_{HDOVERWRPG}$	Over Write Page Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{SULOCKREQUEST}$	Lock Request Setup Time for the Control Logic	0.87	0.99	1.16	ns
$t_{HDLOCKREQUEST}$	Lock Request Hold Time for the Control Logic	0.00	0.00	0.00	ns
$t_{RECARNVM}$	Reset Recovery Time	0.94	1.07	1.25	ns
$t_{REMARNVM}$	Reset Removal Time	0.00	0.00	0.00	ns
$t_{MPWARNVM}$	Asynchronous Reset Minimum Pulse Width for the Control Logic	10.00	12.50	12.50	ns
$t_{MPWCLKNVM}$	Clock Minimum Pulse Width for the Control Logic	4.00	5.00	5.00	ns
$t_{FMAXCLKNVM}$	Maximum Frequency for Clock for the Control Logic – for AFS1500/AFS600	80.00	80.00	80.00	MHz
	Maximum Frequency for Clock for the Control Logic – for AFS250/AFS090	100.00	80.00	80.00	MHz

FlashROM

Fusion devices have 1 kbit of on-chip nonvolatile flash memory that can be read from the FPGA core fabric. The FlashROM is arranged in eight banks of 128 bits during programming. The 128 bits in each bank are addressable as 16 bytes during the read-back of the FlashROM from the FPGA core (Figure 2-45).

The FlashROM can only be programmed via the IEEE 1532 JTAG port. It cannot be programmed directly from the FPGA core. When programming, each of the eight 128-bit banks can be selectively reprogrammed. The FlashROM can only be reprogrammed on a bank boundary. Programming involves an automatic, on-chip bank erase prior to reprogramming the bank. The FlashROM supports a synchronous read and can be read on byte boundaries. The upper three bits of the FlashROM address from the FPGA core define the bank that is being accessed. The lower four bits of the FlashROM address from the FPGA core define which of the 16 bytes in the bank is being accessed.

The maximum FlashROM access clock is 20 MHz. Figure 2-46 shows the timing behavior of the FlashROM access cycle—the address has to be set up on the rising edge of the clock for DOUT to be valid on the next falling edge of the clock.

If the address is unchanged for two cycles:

- D0 becomes invalid 10 ns after the second rising edge of the clock.
- D0 becomes valid again 10 ns after the second falling edge.

If the address unchanged for three cycles:

- D0 becomes invalid 10 ns after the second rising edge of the clock.
- D0 becomes valid again 10 ns after the second falling edge.
- D0 becomes invalid 10 ns after the third rising edge of the clock.
- D0 becomes valid again 10 ns after the third falling edge.

		Byte Number in Bank 4 LSB of ADDR (READ)															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bank Number 3 MSB of ADDR (READ)	7																
	6																
	5																
	4																
	3																
	2																
	1																
	0																

Figure 2-45 • **FlashROM Architecture**

FlashROM Characteristics

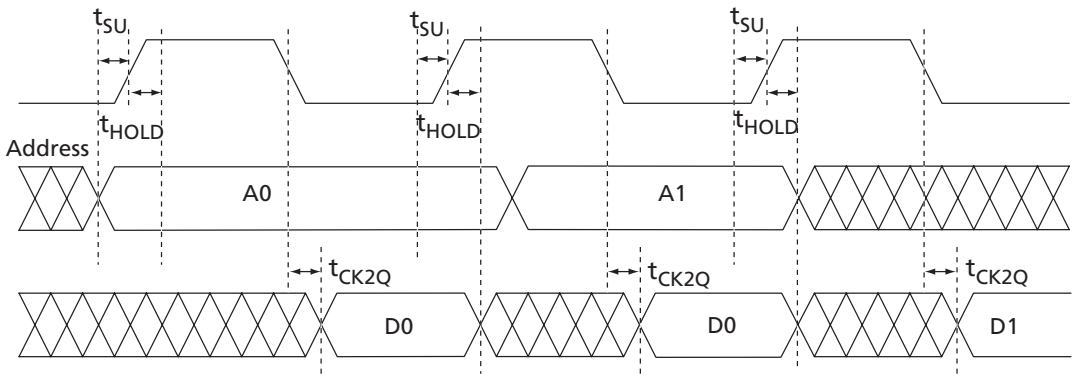


Figure 2-46 • **FlashROM Timing Diagram**

Table 2-26 • **FlashROM Access Time**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{SU}	Address Setup Time	0.53	0.61	0.71	ns
t_{HOLD}	Address Hold Time	0.00	0.00	0.00	ns
t_{CK2Q}	Clock to Out	21.42	24.40	28.68	ns
F_{MAX}	Maximum Clock frequency	15.00	15.00	15.00	MHz

SRAM and FIFO

All Fusion devices have SRAM blocks along the north side of the device. Additionally, AFS600 and AFS1500 devices have an SRAM block on the south side of the device. To meet the needs of high-performance designs, the memory blocks operate strictly in synchronous mode for both read and write operations. The read and write clocks are completely independent, and each may operate at any desired frequency less than or equal to 350 MHz. The following configurations are available:

- 4k×1, 2k×2, 1k×4, 512×9 (dual-port RAM—two read, two write or one read, one write)
- 512×9, 256×18 (two-port RAM—one read and one write)
- Sync write, sync pipelined/nonpipelined read

The Fusion SRAM memory block includes dedicated FIFO control logic to generate internal addresses and external flag logic (FULL, EMPTY, AFULL, AEMPTY).

During RAM operation, addresses are sourced by the user logic, and the FIFO controller is ignored. In FIFO mode, the internal addresses are generated by the FIFO controller and routed to the RAM array by internal MUXes. Refer to Figure 2-47 for more information about the implementation of the embedded FIFO controller.

The Fusion architecture enables the read and write sizes of RAMs to be organized independently, allowing for bus conversion. This is done with the WW (write width) and RW (read width) pins. The different D×W configurations are 256×18, 512×9, 1k×4, 2k×2, and 4k×1. For example, the write size can be set to 256×18 and the read size to 512×9.

Both the write and read widths for the RAM blocks can be specified independently with the WW (write width) and RW (read width) pins. The different D×W configurations are 256×18, 512×9, 1k×4, 2k×2, and 4k×1.

Refer to the allowable RW and WW values supported for each of the RAM macro types in Table 2-27 on page 2-61.

When a width of one, two, or four is selected, the ninth bit is unused. For example, when writing 9-bit values and reading 4-bit values, only the first four bits and the second four bits of each 9-bit value are addressable for read operations. The ninth bit is not accessible.

Conversely, when writing 4-bit values and reading 9-bit values, the ninth bit of a read operation will be undefined. The RAM blocks employ little-endian byte order for read and write operations.

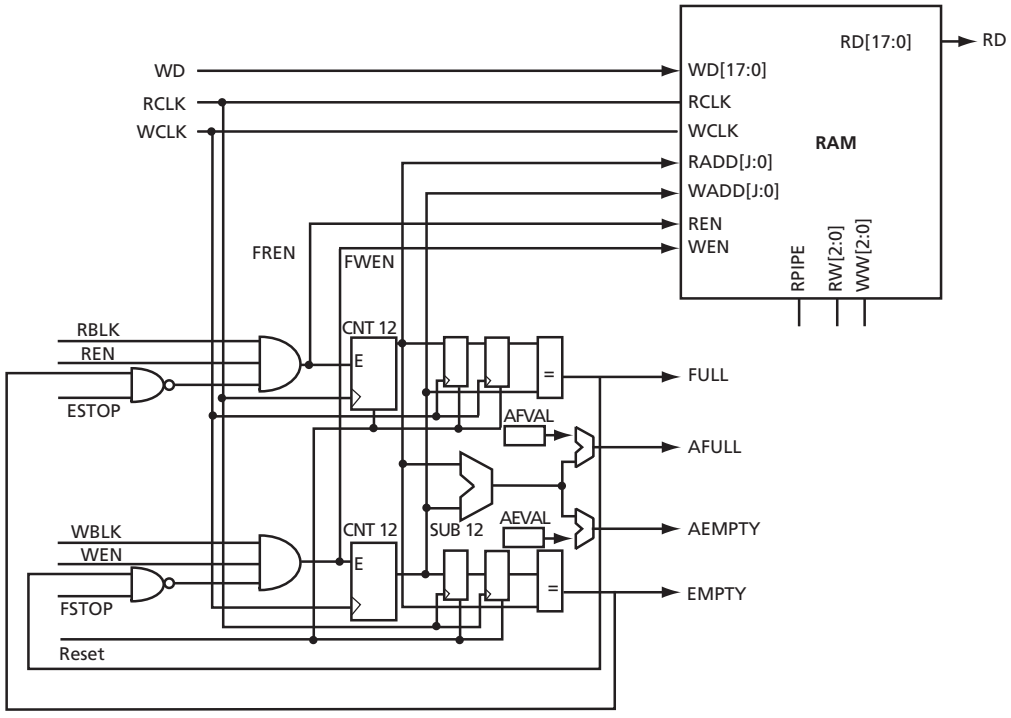


Figure 2-47 • **Fusion RAM Block with Embedded FIFO Controller**

RAM4K9 Description

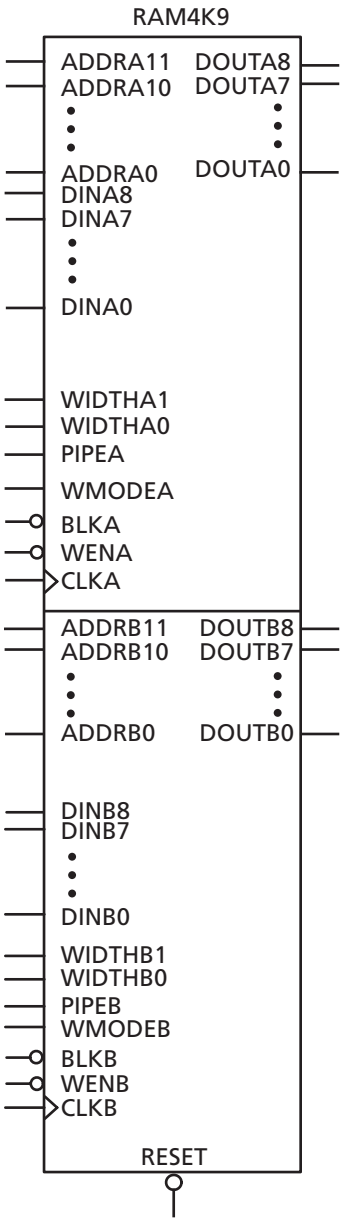


Figure 2-48 • **RAM4K9**

The following signals are used to configure the RAM4K9 memory element:

WIDTHA and WIDTHB

These signals enable the RAM to be configured in one of four allowable aspect ratios (Table 2-27).

Table 2-27 • **Allowable Aspect Ratio Settings for WIDTHA[1:0]**

WIDTHA1, WIDTHA0	WIDTHB1, WIDTHB0	D×W
00	00	4k×1
01	01	2k×2
10	10	1k×4
11	11	512×9

Note: The aspect ratio settings are constant and cannot be changed on the fly.

BLKA and BLKB

These signals are active low and will enable the respective ports when asserted. When a BLKx signal is deasserted, the corresponding port's outputs hold the previous value.

WENA and WENB

These signals switch the RAM between read and write mode for the respective ports. A LOW on these signals indicates a write operation, and a HIGH indicates a read.

CLKA and CLKB

These are the clock signals for the synchronous read and write operations. These can be driven independently or with the same driver.

PIPEA and PIPEB

These signals are used to specify pipelined read on the output. A LOW on PIPEA or PIPEB indicates a nonpipelined read, and the data appears on the corresponding output in the same clock cycle. A HIGH indicates a pipelined, read and data appears on the corresponding output in the next clock cycle.

WMODEA and WMODEB

These signals are used to configure the behavior of the output when the RAM is in write mode. A LOW on these signals makes the output retain data from the previous read. A HIGH indicates pass-through behavior, wherein the data being written will appear immediately on the output. This signal is overridden when the RAM is being read.

RESET

This active low signal resets the output to zero, disables reads and writes from the SRAM block, and clears the data hold registers when asserted. It does not reset the contents of the memory.

ADDRA and ADDRB

These are used as read or write addresses, and they are 12 bits wide. When a depth of less than 4 k is specified, the unused high-order bits must be grounded (Table 2-28).

Table 2-28 • **Address Pins Unused/Used for Various Supported Bus Widths**

	ADDRx	
D×W	Unused	Used
4k×1	None	[11:0]
2k×2	[11]	[10:0]
1k×4	[11:10]	[9:0]
512×9	[11:9]	[8:0]

Note: The "x" in ADDRx implies A or B.

DINA and DINB

These are the input data signals, and they are nine bits wide. Not all nine bits are valid in all configurations. When a data width less than nine is specified, unused high-order signals must be grounded (Table 2-29).

DOUTA and DOUTB

These are the nine-bit output data signals. Not all nine bits are valid in all configurations. As with DINA and DINB, high-order bits may not be used (Table 2-29). The output data on unused pins is undefined.

Table 2-29 • **Unused/Used Input and Output Data Pins for Various Supported Bus Widths**

	DINx/DOUTx	
D×W	**Unused**	**Used**
4k×1	[8:1]	[0]
2k×2	[8:2]	[1:0]
1k×4	[8:4]	[3:0]
512×9	None	[8:0]

Note: The "x" in DINx and DOUTx implies A or B.

RAM512X18 Description

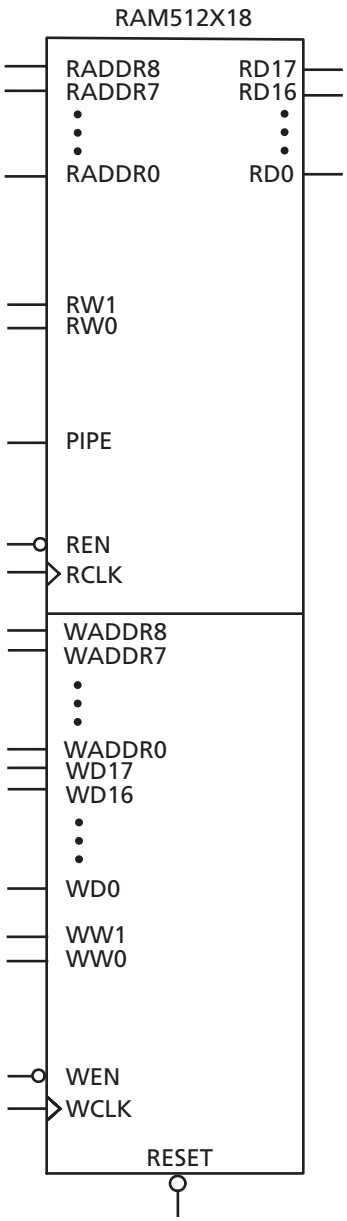


Figure 2-49 • **RAM512X18**

RAM512X18 exhibits slightly different behavior from RAM4K9, as it has dedicated read and write ports.

WW and RW

These signals enable the RAM to be configured in one of the two allowable aspect ratios (Table 2-30).

Table 2-30 • **Aspect Ratio Settings for WW[1:0]**

WW[1:0]	RW[1:0]	D×W
01	01	512×9
10	10	256×18
00, 11	00, 11	Reserved

WD and RD

These are the input and output data signals, and they are 18 bits wide. When a 512×9 aspect ratio is used for write, WD[17:9] are unused and must be grounded. If this aspect ratio is used for read, then RD[17:9] are undefined.

WADDR and RADDR

These are read and write addresses, and they are nine bits wide. When the 256×18 aspect ratio is used for write or read, WADDR[8] or RADDR[8] are unused and must be grounded.

WCLK and RCLK

These signals are the write and read clocks, respectively. They are both active high.

WEN and REN

These signals are the write and read enables, respectively. They are both active low by default. These signals can be configured as active high.

RESET

This active low signal resets the output to zero, disables reads and/or writes from the SRAM block, and clears the data hold registers when asserted. It does not reset the contents of the memory.

PIPE

This signal is used to specify pipelined read on the output. A LOW on PIPE indicates a nonpipelined read, and the data appears on the output in the same clock cycle. A HIGH indicates a pipelined read, and data appears on the output in the next clock cycle.

Clocking

The dual-port SRAM blocks are only clocked on the rising edge. SmartGen allows falling-edge-triggered clocks by adding inverters to the netlist, hence achieving dual-port SRAM blocks that are clocked on either edge (rising or falling). For dual-port SRAM, each port can be clocked on either edge or by separate clocks, by port.

Fusion devices support inversion (bubble pushing) throughout the FPGA architecture, including the clock input to the SRAM modules. Inversions added to the SRAM clock pin on the design schematic or in the HDL code will be automatically accounted for during design compile without incurring additional delay in the clock path.

The two-port SRAM can be clocked on the rising edge or falling edge of WCLK and RCLK.

If negative-edge RAM and FIFO clocking is selected for memory macros, clock edge inversion management (bubble pushing) is automatically used within the Fusion development tools, without performance penalty.

Modes of Operation

There are two read modes and one write mode:

- Read Nonpipelined (synchronous—1 clock edge): In the standard read mode, new data is driven onto the RD bus in the same clock cycle following RA and REN valid. The read address is registered on the read port clock active edge, and data appears at RD after the RAM access time. Setting PIPE to OFF enables this mode.
- Read Pipelined (synchronous—2 clock edges): The pipelined mode incurs an additional clock delay from the address to the data but enables operation at a much higher frequency. The read address is registered on the read port active clock edge, and the read data is registered and appears at RD after the second read clock edge. Setting PIPE to ON enables this mode.
- Write (synchronous—1 clock edge): On the write clock active edge, the write data is written into the SRAM at the write address when WEN is HIGH. The setup times of the write address, write enables, and write data are minimal with respect to the write clock. Write and read transfers are described with timing requirements in the "SRAM Characteristics" section on page 2-66 and the "FIFO Characteristics" section on page 2-77.

RAM Initialization

Each SRAM block can be individually initialized on power-up by means of the JTAG port using the UJTAG mechanism (refer to the "JTAG IEEE 1532" section on page 2-229 and the *Fusion SRAM/FIFO Blocks* application note). The shift register for a target block can be selected and loaded with the proper bit configuration to enable serial loading. The 4,608 bits of data can be loaded in a single operation.

SRAM Characteristics

Timing Waveforms

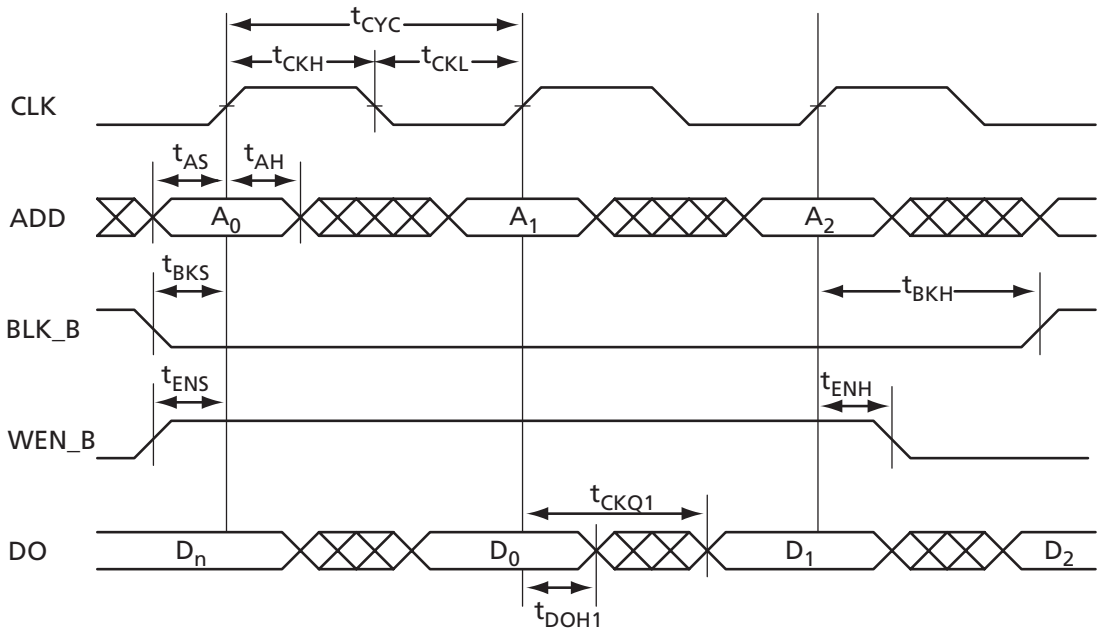


Figure 2-50 • **RAM Read for Flow-Through Output**

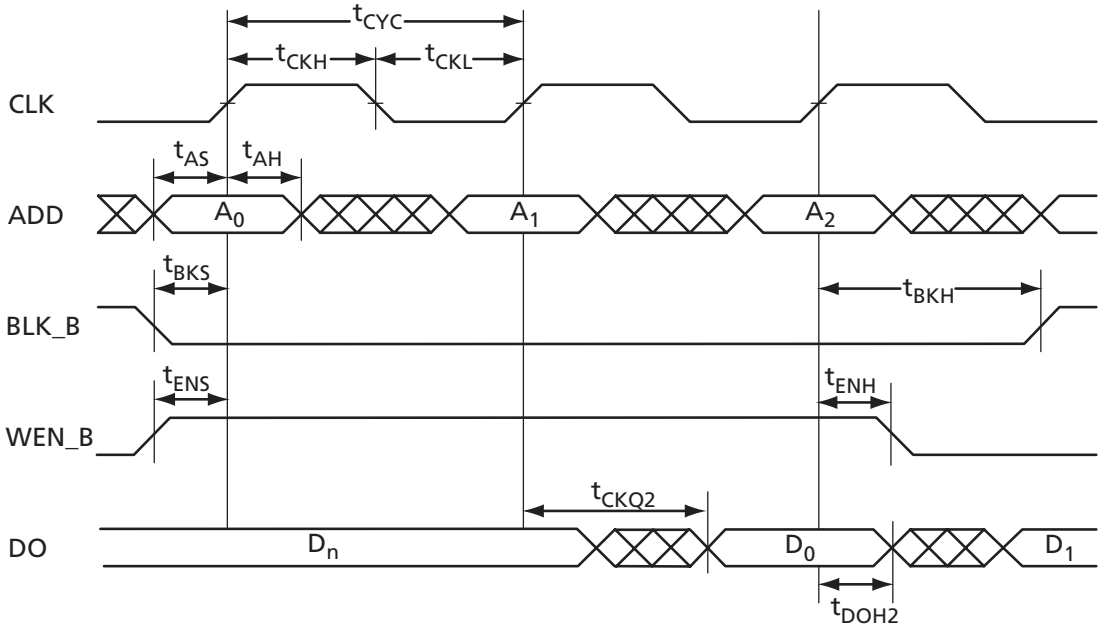


Figure 2-51 • **RAM Read for Pipelined Output**

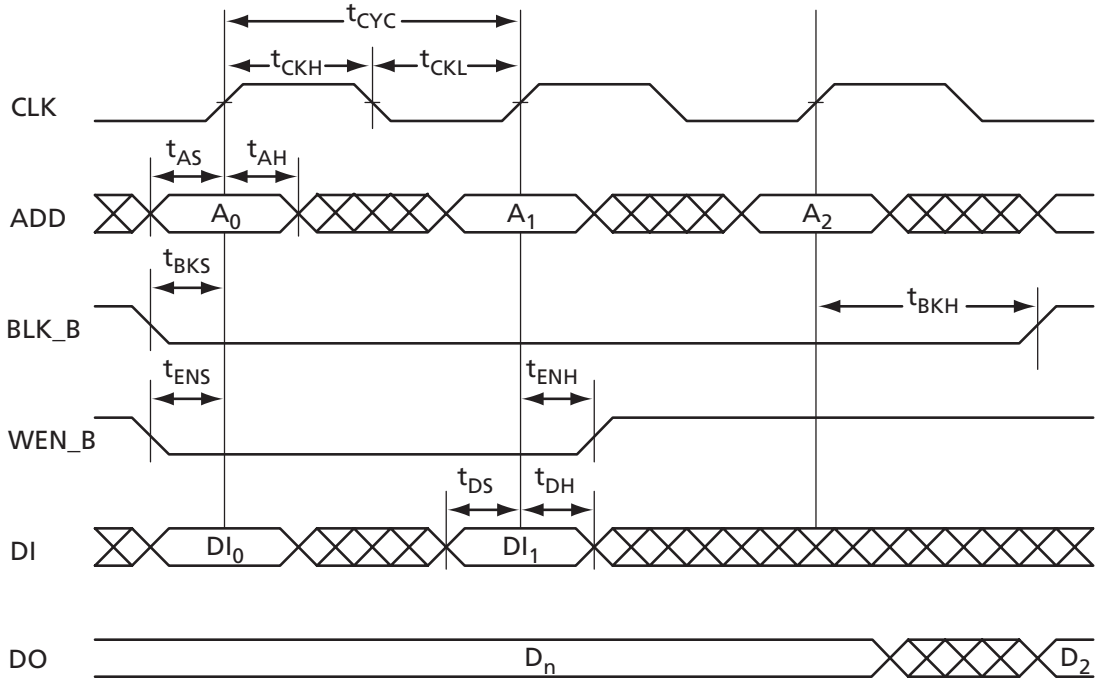


***Figure 2-52* • RAM Write, Output Retained (WMODE = 0)**

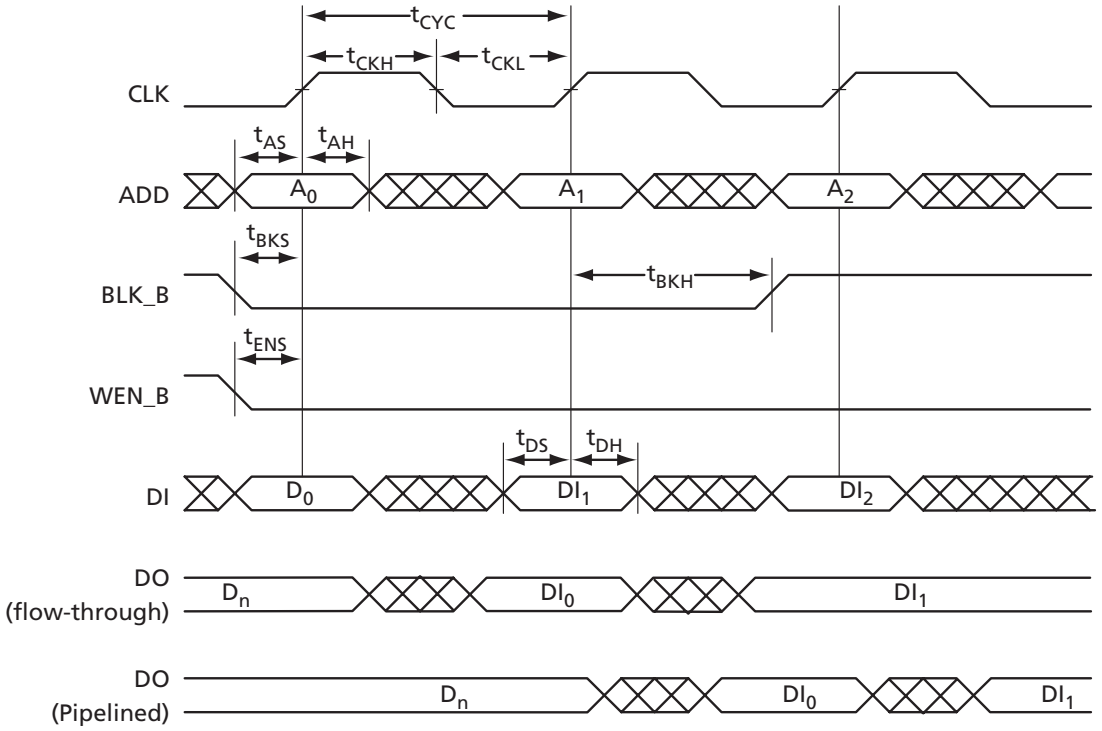


***Figure 2-53* • RAM Write, Output as Write Data (WMODE = 1)**

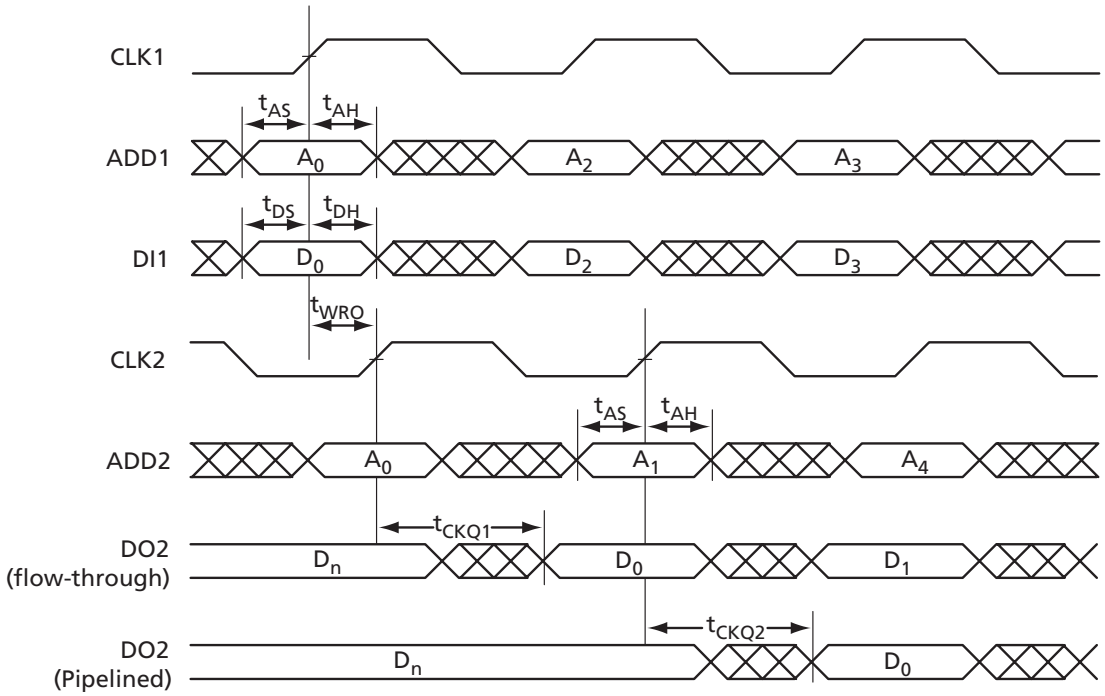


***Figure 2-54* • One Port Write / Other Port Read Same**

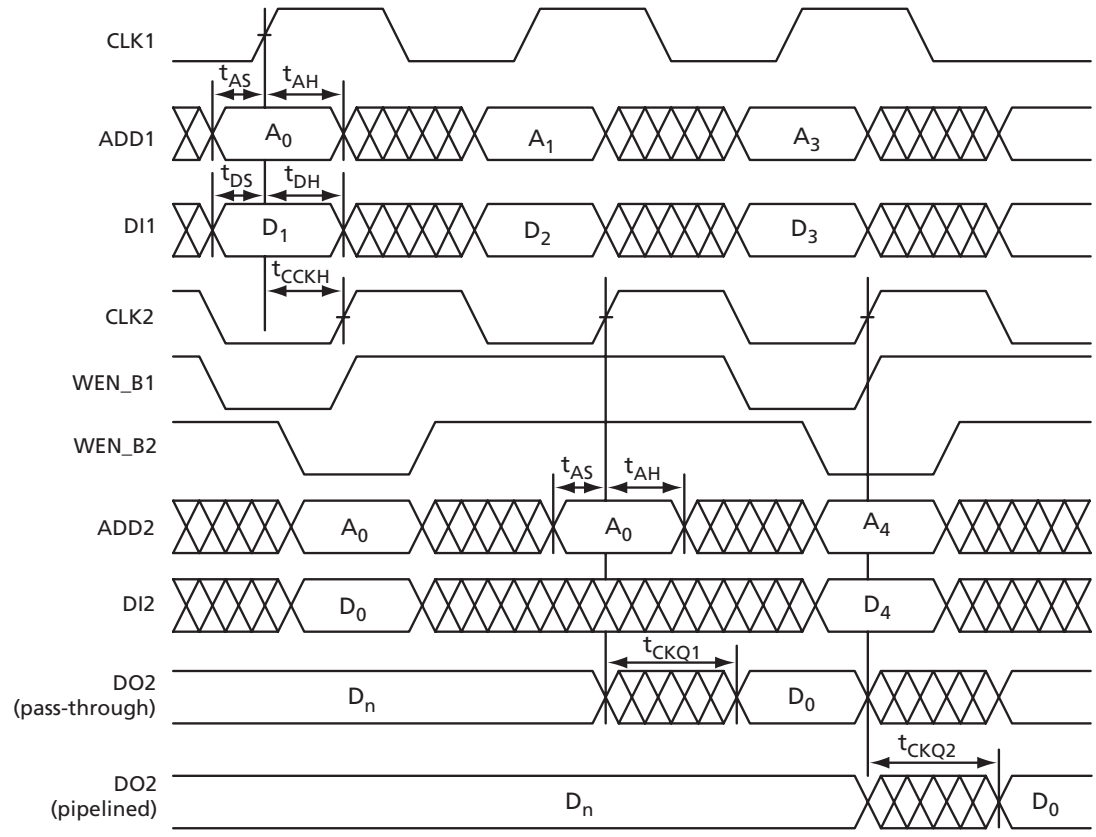


***Figure 2-55* • Write Access After Write onto Same Address**

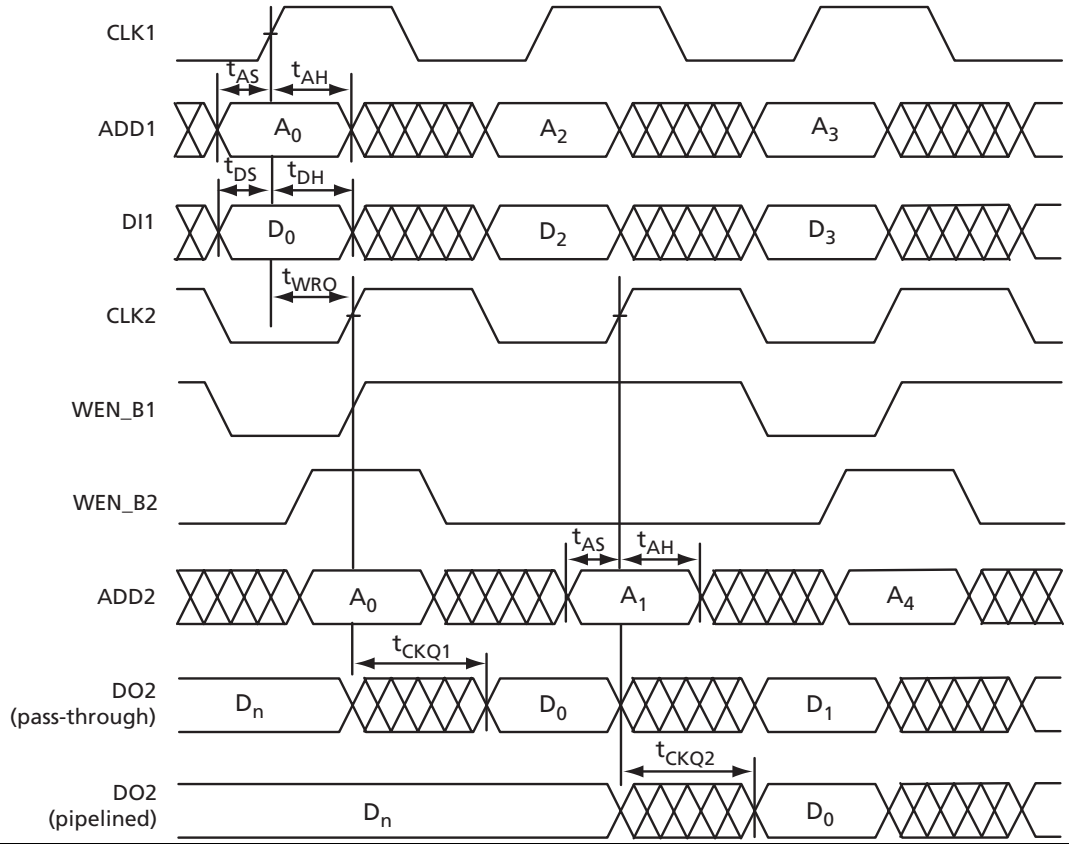


Figure 2-56 • **Read Access After Write onto Same Address**

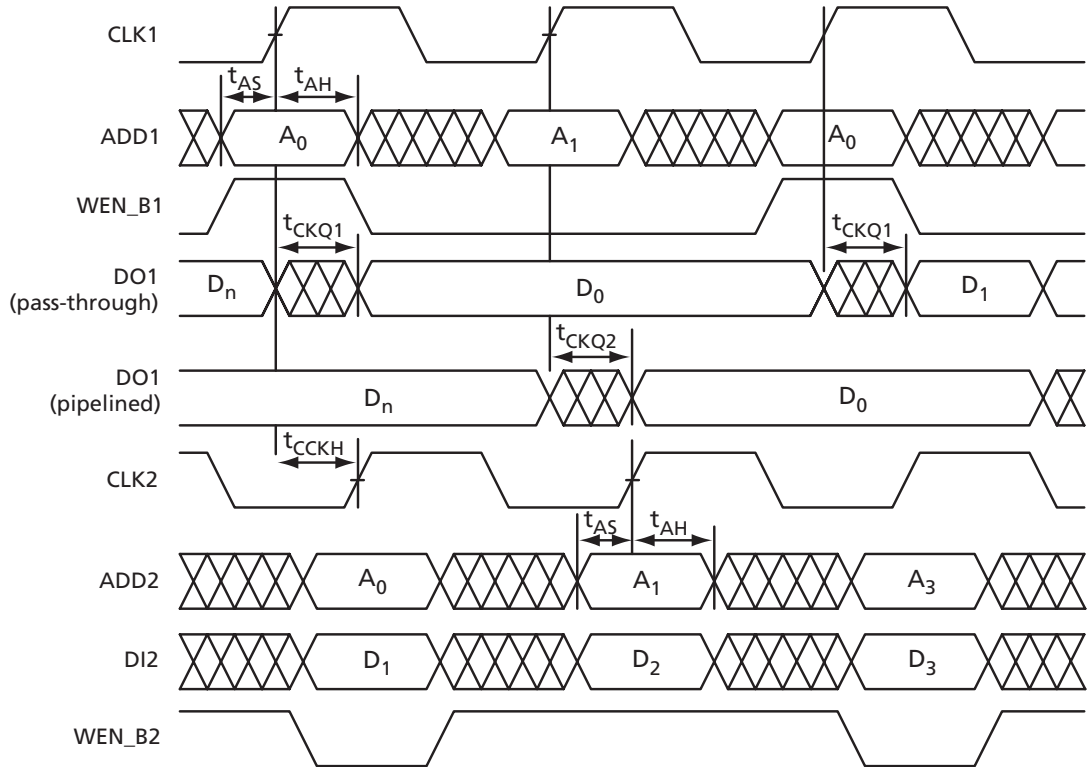


Figure 2-57 • **Write Access After Read onto Same Address**

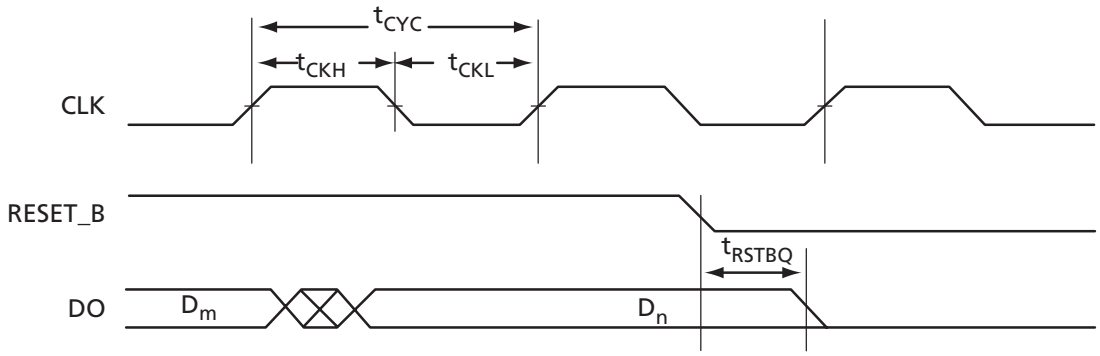


Figure 2-58 • **RAM Reset**

Timing Characteristics

Table 2-31 • **RAM4K9**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{AS}	Address setup time	0.25	0.28	0.33	ns
t_{AH}	Address hold time	0.00	0.00	0.00	ns
t_{ENS}	REN_B, WEN_B setup time	0.14	0.16	0.19	ns
t_{ENH}	REN_B, WEN_B hold time	0.10	0.11	0.13	ns
t_{BKS}	BLK_B setup time	0.23	0.27	0.31	ns
t_{BKH}	BLK_B hold time	0.02	0.02	0.02	ns
t_{DS}	Input data (DI) setup time	0.18	0.21	0.25	ns
t_{DH}	Input data (DI) hold time	0.00	0.00	0.00	ns
t_{CKQ1}	Clock HIGH to new data valid on DO (output retained, WMODE = 0)	1.79	2.03	2.39	ns
	Clock HIGH to new data valid on DO (flow-through, WMODE = 1)	2.36	2.68	3.15	ns
t_{CKQ2}	Clock HIGH to new data valid on DO (pipelined)	0.89	1.02	1.20	ns
t_{C2CWWH}	Address collision clk-to-clk delay for reliable write after write on same address—Applicable to Rising Edge	0.30	0.26	0.23	ns
t_{C2CRWH}	Address collision clk-to-clk delay for reliable read access after write on same address—Applicable to Opening Edge	0.45	0.38	0.34	ns
t_{C2CWRH}	Address collision clk-to-clk delay for reliable write access after read on same address— Applicable to Opening Edge	0.49	0.42	0.37	ns
t_{RSTBQ}	RESET_B LOW to data out LOW on DO (flow-through)	0.92	1.05	1.23	ns
	RESET_B LOW to Data Out LOW on DO (pipelined)	0.92	1.05	1.23	ns
$t_{REMRSTB}$	RESET_B removal	0.29	0.33	0.38	ns
$t_{RECRSTB}$	RESET_B recovery	1.50	1.71	2.01	ns
$t_{MPWRSTB}$	RESET_B minimum pulse width	0.21	0.24	0.29	ns
t_{CYC}	Clock cycle time	3.23	3.68	4.32	ns
F_{MAX}	Maximum frequency	310	272	231	MHz

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-32 • **RAM512X18**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{AS}	Address setup time	0.25	0.28	0.33	ns
t_{AH}	Address hold time	0.00	0.00	0.00	ns
t_{ENS}	REN_B, WEN_B setup time	0.09	0.10	0.12	ns
t_{ENH}	REN_B, WEN_B hold time	0.06	0.07	0.08	ns
t_{DS}	Input data (DI) setup time	0.18	0.21	0.25	ns
t_{DH}	Input data (DI) hold time	0.00	0.00	0.00	ns
t_{CKQ1}	Clock HIGH to new data valid on DO (output retained, WMODE = 0)	2.16	2.46	2.89	ns
t_{CKQ2}	Clock HIGH to new data valid on DO (pipelined)	0.90	1.02	1.20	ns
t_{C2CRWH}	Address collision clk-to-clk delay for reliable read access after write on same address—Applicable to Opening Edge	0.50	0.43	0.38	ns
t_{C2CWRH}	Address collision clk-to-clk delay for reliable write access after read on same address— Applicable to Opening Edge	0.59	0.50	0.44	ns
t_{RSTBQ}	RESET_B LOW to data out LOW on DO (flow-through)	0.92	1.05	1.23	ns
	RESET_B LOW to data out LOW on DO (pipelined)	0.92	1.05	1.23	ns
$t_{REMRSTB}$	RESET_B removal	0.29	0.33	0.38	ns
$t_{RECRSTB}$	RESET_B recovery	1.50	1.71	2.01	ns
$t_{MPWRSTB}$	RESET_B minimum pulse width	0.21	0.24	0.29	ns
t_{CYC}	Clock cycle time	3.23	3.68	4.32	ns
F_{MAX}	Maximum frequency	310	272	231	MHz

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

FIFO4K18 Description

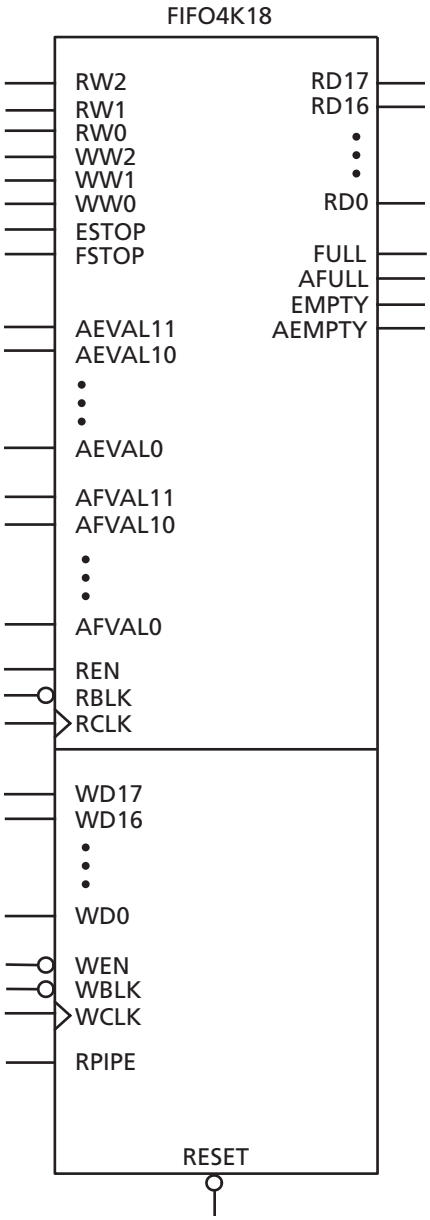


Figure 2-59 • **FIFO4KX18**

The following signals are used to configure the FIFO4K18 memory element:

WW and RW

These signals enable the FIFO to be configured in one of the five allowable aspect ratios (Table 2-33).

Table 2-33 • **Aspect Ratio Settings for WW[2:0]**

WW2, WW1, WW0	RW2, RW1, RW0	D×W
000	000	4k×1
001	001	2k×2
010	010	1k×4
011	011	512×9
100	100	256×18
101, 110, 111	101, 110, 111	Reserved

WBLK and RBLK

These signals are active low and will enable the respective ports when LOW. When the RBLK signal is HIGH, the corresponding port's outputs hold the previous value.

WEN and REN

Read and write enables. WEN is active low and REN is active high by default. These signals can be configured as active high or low.

WCLK and RCLK

These are the clock signals for the synchronous read and write operations. These can be driven independently or with the same driver.

RPIPE

This signal is used to specify pipelined read on the output. A LOW on RPIPE indicates a nonpipelined read, and the data appears on the output in the same clock cycle. A HIGH indicates a pipelined read, and data appears on the output in the next clock cycle.

RESET

This active low signal resets the output to zero when asserted. It resets the FIFO counters. It also sets all the RD pins LOW, the FULL and AFULL pins LOW, and the EMPTY and AEMPTY pins HIGH (Table 2-34).

Table 2-34 • **Input Data Signal Usage for Different Aspect Ratios**

D×W	WD/RD Unused
4k×1	WD[17:1], RD[17:1]
2k×2	WD[17:2], RD[17:2]
1k×4	WD[17:4], RD[17:4]
512×9	WD[17:9], RD[17:9]
256×18	–

WD

This is the input data bus and is 18 bits wide. Not all 18 bits are valid in all configurations. When a data width less than 18 is specified, unused higher-order signals must be grounded (Table 2-34).

RD

This is the output data bus and is 18 bits wide. Not all 18 bits are valid in all configurations. Like the WD bus, high-order bits become unusable if the data width is less than 18. The output data on unused pins is undefined (Table 2-34).

ESTOP, FSTOP

ESTOP is used to stop the FIFO read counter from further counting once the FIFO is empty (i.e., the EMPTY flag goes HIGH). A HIGH on this signal inhibits the counting.

FSTOP is used to stop the FIFO write counter from further counting once the FIFO is full (i.e., the FULL flag goes HIGH). A HIGH on this signal inhibits the counting.

For more information on these signals, refer to the "ESTOP and FSTOP Usage" section on page 2-76.

FULL, EMPTY

When the FIFO is full and no more data can be written, the FULL flag asserts HIGH. The FULL flag is synchronous to WCLK to inhibit writing immediately upon detection of a full condition and to prevent overflows. Since the write address is compared to a resynchronized (and thus time-delayed) version of the read address, the FULL flag will remain asserted until two WCLK active edges after a read operation eliminates the full condition.

When the FIFO is empty and no more data can be read, the EMPTY flag asserts HIGH. The EMPTY flag is synchronous to RCLK to inhibit reading immediately upon detection of an empty condition and to prevent underflows. Since the read address is compared to a resynchronized (and thus time-delayed) version of the write address, the EMPTY flag will remain asserted until two RCLK active edges after a write operation removes the empty condition.

For more information on these signals, refer to the "FIFO Flag Usage Considerations" section on page 2-76.

AFULL, AEMPTY

These are programmable flags and will be asserted on the threshold specified by AFVAL and AEVAL, respectively.

When the number of words stored in the FIFO reaches the amount specified by AEVAL while reading, the AEMPTY output will go HIGH. Likewise, when the number of words stored in the FIFO reaches the amount specified by AFVAL while writing, the AFULL output will go HIGH.

AFVAL, AEVAL

The AEVAL and AFVAL pins are used to specify the almost-empty and almost-full threshold values, respectively. They are 12-bit signals. For more information on these signals, refer to "FIFO Flag Usage Considerations" section.

ESTOP and FSTOP Usage

The ESTOP pin is used to stop the read counter from counting any further once the FIFO is empty (i.e., the EMPTY flag goes HIGH). Likewise, the FSTOP pin is used to stop the write counter from counting any further once the FIFO is full (i.e., the FULL flag goes HIGH).

The FIFO counters in the Fusion device start the count at 0, reach the maximum depth for the configuration (e.g., 511 for a 512×9 configuration), and then restart at 0. An example application for the ESTOP, where the read counter keeps counting, would be writing to the FIFO once and reading the same content over and over without doing another write.

FIFO Flag Usage Considerations

The AEVAL and AFVAL pins are used to specify the 12-bit AEMPTY and AFULL threshold values, respectively. The FIFO contains separate 12-bit write address (WADDR) and read address (RADDR) counters. WADDR is incremented every time a write operation is performed, and RADDR is incremented every time a read operation is performed. Whenever the difference between WADDR and RADDR is greater than or equal to AFVAL, the AFULL output is asserted. Likewise, whenever the difference between WADDR and RADDR is less than or equal to AEVAL, the AEMPTY output is asserted. To handle different read and write aspect ratios, AFVAL and AEVAL are expressed in terms of total data bits instead of total data words. When users specify AFVAL and AEVAL in terms of read or write words, the SmartGen tool translates them into bit addresses and configures these signals automatically. SmartGen configures the AFULL flag to assert when the write address exceeds the read address by at least a predefined value. In a 2k×8 FIFO, for example, a value of 1,500 for AFVAL means that the AFULL flag will be asserted after a write when the difference between the write address and the read address reaches 1,500 (there have been at least 1500 more writes than reads). It will stay asserted until the difference between the write and read addresses drops below 1,500.

The AEMPTY flag is asserted when the difference between the write address and the read address is less than a predefined value. In the example above, a value of 200 for AEVAL means that the AEMPTY flag will be asserted when a read causes the difference between the write address and the read address to drop to 200. It will stay asserted until that difference rises above 200. Note that the FIFO can be configured with different read and write widths; in this case, the AFVAL setting is based on the number of write data entries and the AEVAL setting is based on the number of read data entries. For aspect ratios of 512×9 and 256×18, only 4,096 bits can be addressed by the 12 bits of AFVAL and AEVAL. The number of words must be multiplied by 8 and 16, instead of 9 and 18. The SmartGen tool automatically uses the proper values. To avoid halfwords being written or read, which could happen if different read and write aspect ratios are specified, the FIFO will assert FULL or EMPTY as soon as at least a minimum of one word cannot be written or read. For example, if a two-bit word is written and a four-bit word is being read, the FIFO will remain in the empty state when the first word is written. This occurs even if the FIFO is not completely empty, because in this case, a complete word cannot be read. The same is applicable in the full state. If a four-bit word is written and a two-bit word is read, the FIFO is full and one word is read. The FULL flag will remain asserted because a complete word cannot be written at this point.

FIFO Characteristics

Timing Waveforms

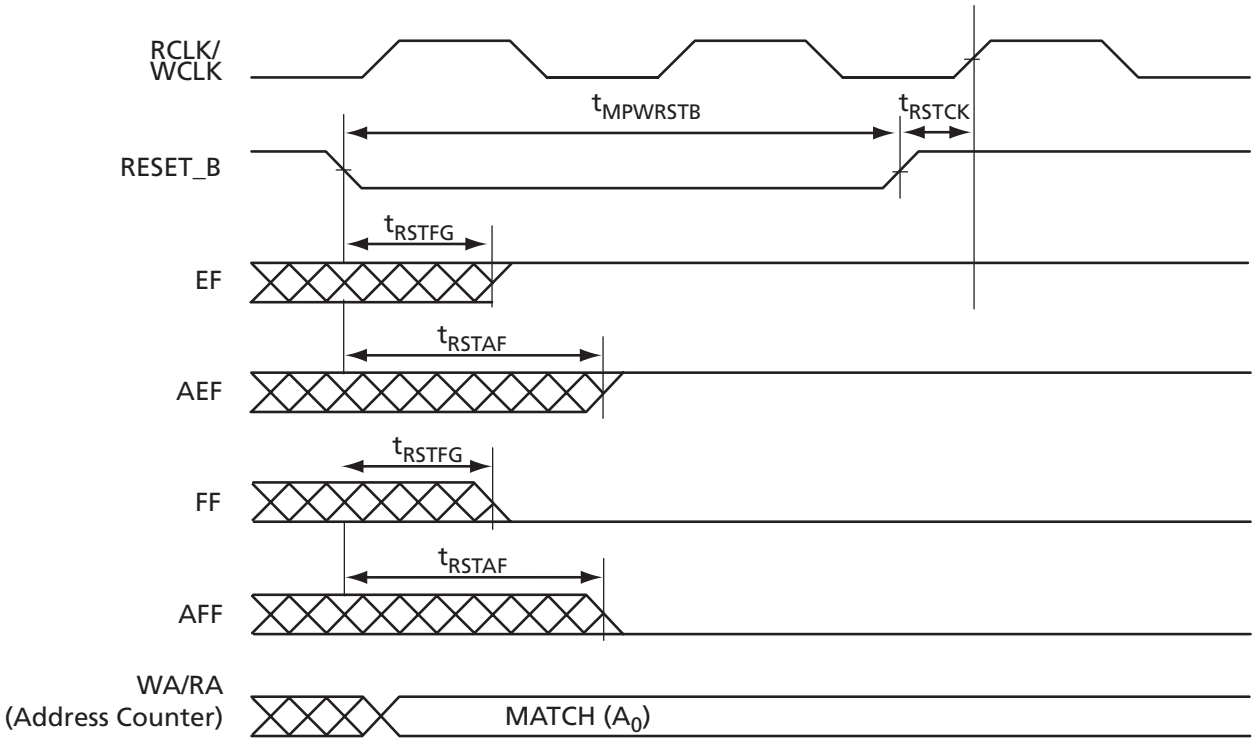


***Figure 2-60* • FIFO Reset**

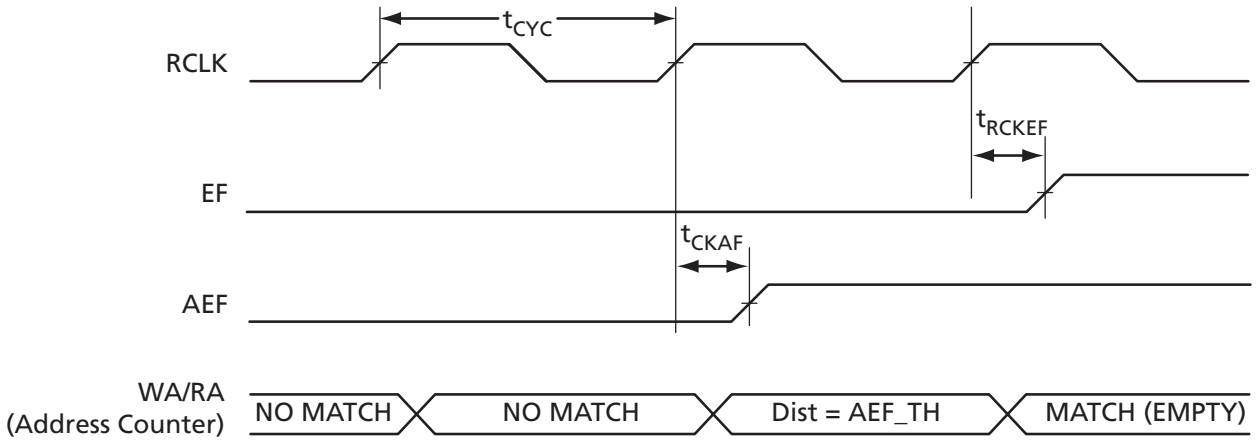


***Figure 2-61* • FIFO EMPTY Flag and AEMPTY Flag Assertion**

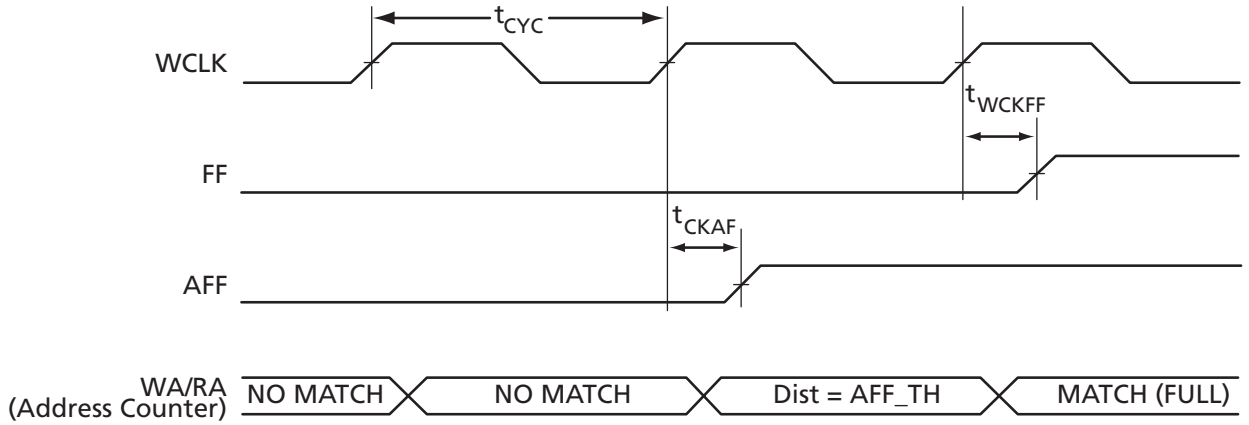


Figure 2-62 • **FIFO FULL and AFULL Flag Assertion**

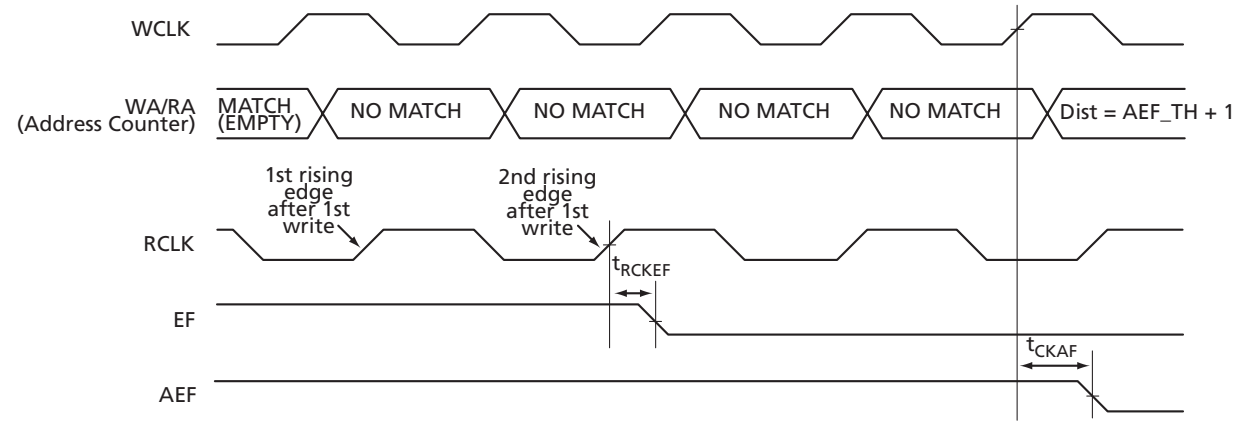


Figure 2-63 • **FIFO EMPTY Flag and AEMPTY Flag Deassertion**

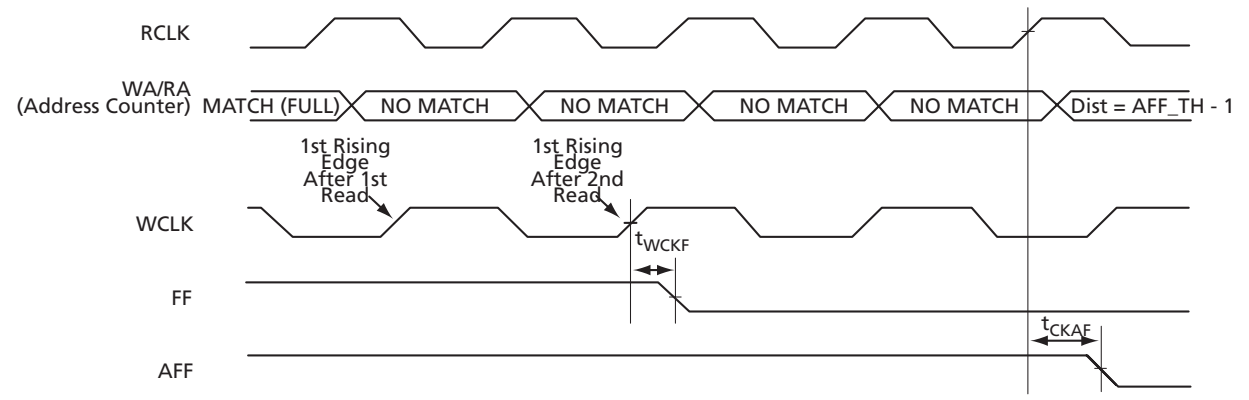


Figure 2-64 • **FIFO FULL Flag and AFULL Flag Deassertion**

Timing Characteristics

Table 2-35 • FIFO
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{ENS}	REN_B, WEN_B Setup time	1.34	1.52	1.79	ns
t_{ENH}	REN_B, WEN_B Hold time	0.00	0.00	0.00	ns
t_{BKS}	BLK_B Setup time	0.19	0.22	0.26	ns
t_{BKH}	BLK_B Hold time	0.00	0.00	0.00	ns
t_{DS}	Input data (DI) Setup time	0.18	0.21	0.25	ns
t_{DH}	Input data (DI) Hold time	0.00	0.00	0.00	ns
t_{CKQ1}	Clock High to New Data Valid on DO (flow-through)	2.17	2.47	2.90	ns
t_{CKQ2}	Clock High to New Data Valid on DO (pipelined)	0.94	1.07	1.26	ns
t_{RCKEF}	RCLK High to Empty Flag Valid	1.72	1.96	2.30	ns
t_{WCKFF}	WCLK High to Full Flag Valid	1.63	1.86	2.18	ns
t_{CKAF}	Clock High to Almost Empty/Full Flag Valid	6.19	7.05	8.29	ns
t_{RSTFG}	RESET_B Low to Empty/Full Flag Valid	1.69	1.93	2.27	ns
t_{RSTAF}	RESET_B Low to Almost-Empty/Full Flag Valid	6.13	6.98	8.20	ns
t_{RSTBQ}	RESET_B Low to Data out Low on DO (flow-through)	0.92	1.05	1.23	ns
	RESET_B Low to Data out Low on DO (pipelined)	0.92	1.05	1.23	ns
$t_{REMRSTB}$	RESET_B Removal	0.29	0.33	0.38	ns
$t_{RECRSTB}$	RESET_B Recovery	1.50	1.71	2.01	ns
$t_{MPWRSTB}$	RESET_B Minimum Pulse Width	0.21	0.24	0.29	ns
t_{CYC}	Clock Cycle time	3.23	3.68	4.32	ns
F_{MAX}	Maximum Frequency for FIFO	310	272	231	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Analog Block

With the Fusion family, Actel has introduced the world's first mixed-mode FPGA solution. Supporting a robust analog peripheral mix, Fusion devices will support a wide variety of applications. It is this Analog Block that separates Fusion from all other FPGA solutions on the market today.

By combining both flash and high-speed CMOS processes in a single chip, these devices offer the best of both worlds. The high-performance CMOS is used for building RAM resources. These high-performance structures support device operation up to 350 MHz. Additionally, the advanced Actel 0.13 µm flash process incorporates high-voltage transistors and a high-isolation, triple-well process. Both of these are suited for the flash-based programmable logic and nonvolatile memory structures.

High-voltage transistors support the integration of analog technology in several ways. They aid in noise immunity so that the analog portions of the chip can be better isolated from the digital portions, increasing analog accuracy. Because they support high voltages, Actel flash FPGAs can be connected directly to high-voltage input signals, eliminating the need for external resistor divider networks, reducing component count, and increasing accuracy. By supporting higher internal voltages, the Actel advanced flash process enables high dynamic range on analog circuitry, increasing precision and signal–noise ratio. Actel flash FPGAs also drive high-voltage outputs, eliminating the need for external level shifters and drivers.

The unique triple-well process enables the integration of high-performance analog features with increased noise immunity and better isolation. By increasing the efficiency of analog design, the triple-well process also enables a smaller overall design size, reducing die size and cost.

The Analog Block consists of the Analog Quad I/O structure, RTC (for details refer to the "Real-Time Counter System" section on page 2-34), ADC, and ACM. All of these elements are combined in the single Analog Block macro, with which the user implements this functionality (Figure 2-65).

The Analog Block needs to be reset/reinitialized after the core powers up or the device is programmed. An external reset/initialize signal, which can come from the internal voltage regulator when it powers up, must be applied.

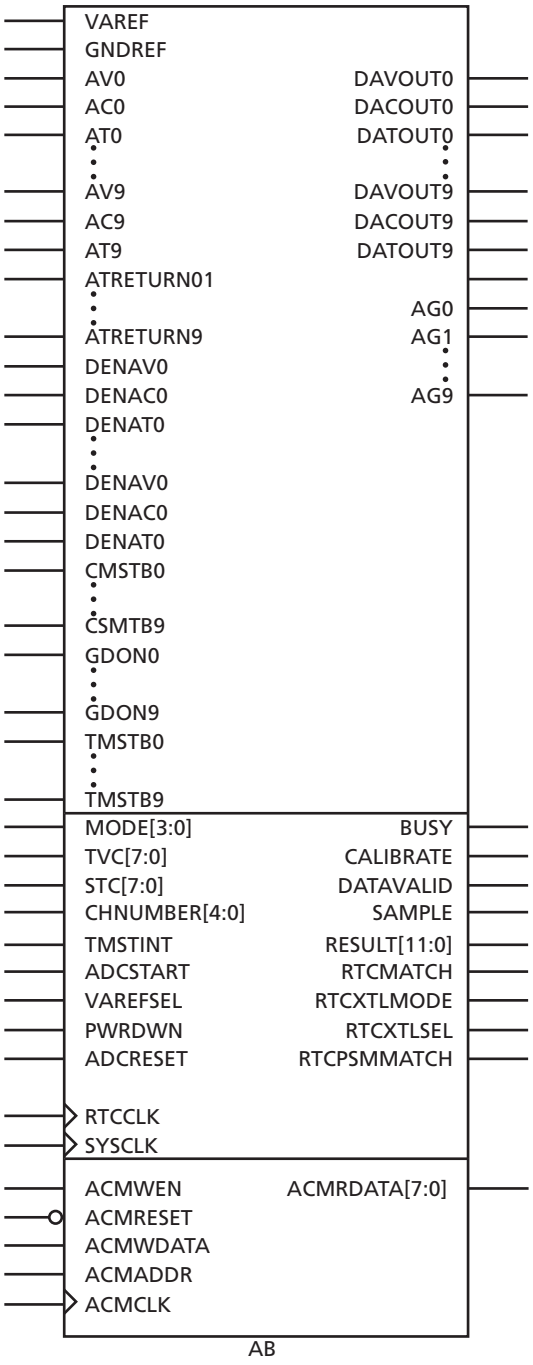


Figure 2-65 • **Analog Block Macro**

Table 2-36 describes each pin in the Analog Block. Each function within the Analog Block will be explained in detail in the following sections.

Table 2-36 • **Analog Block Pin Description**

Signal Name	Number of Bits	Direction	Function	Location of Details
VAREF	1	Input/Output	Voltage reference for ADC	ADC
GNDREF	1	Input	External ground reference	ADC
MODE[3:0]	4	Input	ADC operating mode	ADC
SYSCLK	1	Input	External system clock	
TVC[7:0]	8	Input	Clock divide control	ADC
STC[7:0]	8	Input	Sample time control	ADC
ADCSTART	1	Input	Start of conversion	ADC
PWRDWN	1	Input	ADC comparator power-down if 1. When asserted, the ADC will stop functioning, and the digital portion of the analog block will continue operating. This may result in invalid status flags from the analog block. Therefore, Actel does not recommend asserting the PWRDWN pin.	ADC
ADCRESET	1	Input	ADC resets and disables Analog Quad – active high	ADC
BUSY	1	Output	1 – Running conversion	ADC
CALIBRATE	1	Output	1 – Power-up calibration	ADC
DATAVALID	1	Output	1 – Valid conversion result	ADC
RESULT[11:0]	12	Output	Conversion result	ADC
TMSTBINT	1	Input	Internal temp. monitor strobe	ADC
SAMPLE	1	Output	1 – An analog signal is actively being sampled (stays high during signal acquisition only) 0 – No analog signal is being sampled	ADC
VAREFSEL	1	Input	0 = Output internal voltage reference (2.56 V) to VAREF 1 = Input external voltage reference from VAREF and GNDREF	ADC
CHNUMBER[4:0]	5	Input	Analog input channel select	Input multiplexer
ACMCLK	1	Input	ACM clock	ACM
ACMWEN	1	Input	ACM write enable – active high	ACM
ACMRESET	1	Input	ACM reset – active low	ACM
ACMWDATA[7:0]	8	Input	ACM write data	ACM
ACMRDATA[7:0]	8	Output	ACM read data	ACM
ACMADDR[7:0]	8	Input	ACM address	ACM

Table 2-36 • **Analog Block Pin Description (continued)**

Signal Name	Number of Bits	Direction	Function	Location of Details
CMSTB0 to CMSTB9	10	Input	Current monitor strobe – 1 per quad, active high	Analog Quad
GDON0 to GDON9	10	Input	Control to power MOS – 1 per quad	Analog Quad
TMSTB0 to TMSTB9	10	Input	Temperature monitor strobe – 1 per quad; active high	Analog Quad
DAVOUT0, DACOUT0, DATOUT0 to DAVOUT9, DACOUT9, DATOUT9	30	Output	Digital outputs – 3 per quad	Analog Quad
DENAV0, DENAC0, DENAT0 to DENAV9, DENAC9, DENAT9	30	Input	Digital input enables – 3 per quad	Analog Quad
AV0	1	Input	Analog Quad 0	Analog Quad
AC0	1	Input		Analog Quad
AG0	1	Output		Analog Quad
AT0	1	Input		Analog Quad
ATRETURN01	1	Input	Temperature monitor return shared by Analog Quads 0 and 1	Analog Quad
AV1	1	Input	Analog Quad 1	Analog Quad
AC1	1	Input		Analog Quad
AG1	1	Output		Analog Quad
AT1	1	Input		Analog Quad
AV2	1	Input	Analog Quad 2	Analog Quad
AC2	1	Input		Analog Quad
AG2	1	Output		Analog Quad
AT2	1	Input		Analog Quad
ATRETURN23	1	Input	Temperature monitor return shared by Analog Quads 2 and 3	Analog Quad
AV3	1	Input	Analog Quad 3	Analog Quad
AC3	1	Input		Analog Quad
AG3	1	Output		Analog Quad
AT3	1	Input		Analog Quad
AV4	1	Input	Analog Quad 4	Analog Quad
AC4	1	Input		Analog Quad
AG4	1	Output		Analog Quad
AT4	1	Input		Analog Quad
ATRETURN45	1	Input	Temperature monitor return shared by Analog Quads 4 and 5	Analog Quad
AV5	1	Input	Analog Quad 5	Analog Quad
AC5	1	Input		Analog Quad
AG5	1	Output		Analog Quad
AT5	1	Input		Analog Quad

Table 2-36 • **Analog Block Pin Description (continued)**

Signal Name	Number of Bits	Direction	Function	Location of Details
AV6	1	Input	Analog Quad 6	Analog Quad
AC6	1	Input		Analog Quad
AG6	1	Output		Analog Quad
AT6	1	Input		Analog Quad
ATRETURN67	1	Input	Temperature monitor return shared by Analog Quads 6 and 7	Analog Quad
AV7	1	Input	Analog Quad 7	Analog Quad
AC7	1	Input		Analog Quad
AG7	1	Output		Analog Quad
AT7	1	Input		Analog Quad
AV8	1	Input	Analog Quad 8	Analog Quad
AC8	1	Input		Analog Quad
AG8	1	Output		Analog Quad
AT8	1	Input		Analog Quad
ATRETURN89	1	Input	Temperature monitor return shared by Analog Quads 8 and 9	Analog Quad
AV9	1	Input	Analog Quad 9	Analog Quad
AC9	1	Input		Analog Quad
AG9	1	Output		Analog Quad
AT9	1	Input		Analog Quad
RTCMATCH	1	Output	MATCH	RTC
RTCPSMMATCH	1	Output	MATCH connected to VRPSM	RTC
RTCXTLMODE[1:0]	2	Output	Drives XTLOSC RTCMODE[1:0] pins	RTC
RTCXTLSEL	1	Output	Drives XTLOSC MODESEL pin	RTC
RTCCLK	1	Input	RTC clock input	RTC

Analog Quad

With the Fusion family, Actel introduces the Analog Quad, shown in Figure 2-66 on page 2-85, as the basic analog I/O structure. The Analog Quad is a four-channel system used to precondition a set of analog signals before sending it to the ADC for conversion into a digital signal. To maximize the usefulness of the Analog Quad, the analog input signals can also be configured as LVTTL digital input signals. The Analog Quad is divided into four sections.

The first section is called the Voltage Monitor Block, and its input pin is named AV. It contains a two-channel analog multiplexer that allows an incoming analog signal to be routed directly to the ADC or allows the signal to be routed to a prescaler circuit before being sent to the ADC. The prescaler can be configured to accept analog signals between –12 V and 0 or between 0 and +12 V. The prescaler circuit scales the voltage applied to the ADC input pad such that it is compatible with the ADC input voltage range. The AV pin can also be used as a digital input pin.

The second section of the Analog Quad is called the Current Monitor Block. Its input pin is named AC. The Current Monitor Block contains all the same functions as the Voltage Monitor Block with one addition, which is a current monitoring function. A small external current sensing resistor (typically less than 1 Ω) is connected between the AV and AC pins and is in series with a power

source. The Current Monitor Block contains a current monitor circuit that converts the current through the external resistor to a voltage that can then be read using the ADC.

The third part of the Analog Quad is called the Gate Driver Block, and its output pin is named AG. This section is used to drive an external FET. There are two modes available: a High Current Drive mode and a Current Source Control mode. Both negative and positive voltage polarities are available, and in the current source control mode, four different current levels are available.

The fourth section of the Analog Quad is called the Temperature Monitor Block, and its input pin name is AT. This block is similar to the Voltage Monitor Block, except that it has an additional function: it can be used to monitor the temperature of an external diode-connected transistor. It has a modified prescaler and is limited to positive voltages only.

The Analog Quad can be configured during design time by Actel Libero IDE; however, the ACM can be used to change the parameters of any of these I/Os during runtime. This type of change is referred to as a context switch. The Analog Quad is a modular structure that is replicated to generate the analog I/O resources. Each Fusion device supports between 5 and 10 Analog Quads.

The analog pads are numbered to clearly identify both the type of pad (voltage, current, gate driver, or temperature pad) and its corresponding Analog Quad (AV0, AC0, AG0, AT0, AV1, …, AC9, AG9, and AT9). There are three types of input pads (AVx, ACx, and ATx) and one type of analog output pad (AGx). Since there can be up to 10 Analog Quads on a device, there can be a maximum of 30 analog input pads and 10 analog output pads.

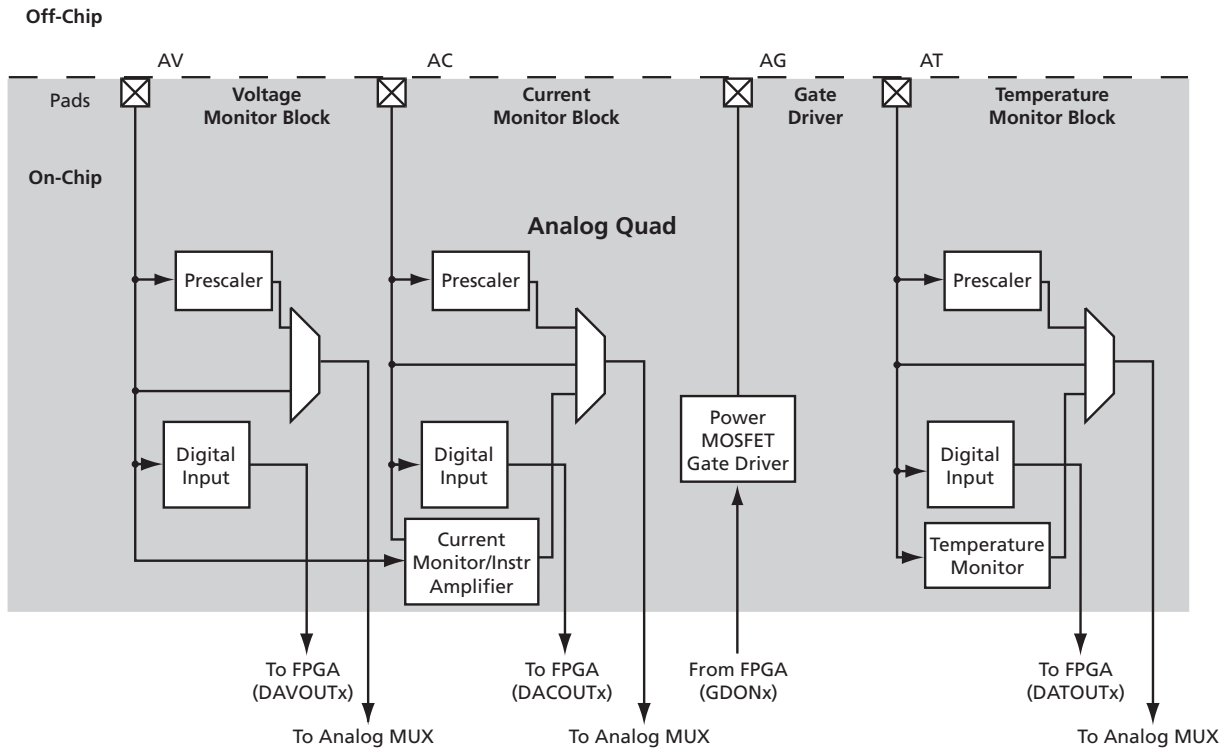


Figure 2-66 • **Analog Quad**

Voltage Monitor

The Fusion Analog Quad offers a robust set of voltage-monitoring capabilities unique in the FPGA industry. The Analog Quad comprises three analog input pads— Analog Voltage (AV), Analog Current (AC), and Analog Temperature (AT)—and a single gate driver output pad, Analog Gate (AG). There are many common characteristics among the analog input pads. Each analog input can be configured to connect directly to the input MUX of the ADC. When configured in this manner (Figure 2-67), there will be no prescaling of the input signal. Care must be taken in this mode not to drive the ADC into saturation by applying an input voltage greater than the reference voltage. The internal reference voltage of the ADC is 2.56 V. Optionally, an external reference can be supplied by the user. The external reference can be a maximum of 3.3 V DC.

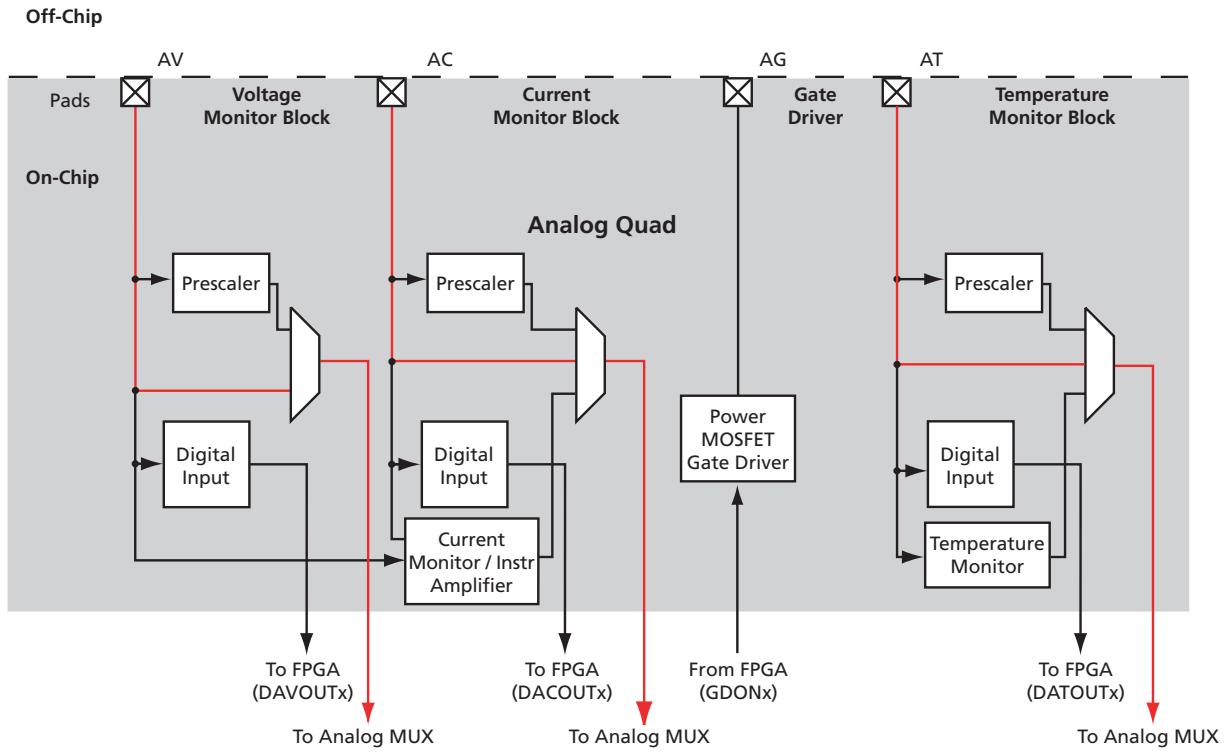


Figure 2-67 • **Analog Quad Direct Connect**

The Analog Quad offers a wide variety of prescaling options to enable the ADC to resolve the input signals. Figure 2-68 shows the path through the Analog Quad for a signal that is to be prescaled prior to conversion. The ADC internal reference voltage and the prescaler factors were selected to make both prescaling and postscaling of the signals easy binary calculations (refer to Table 2-54 on page 2-131 for details). When an analog input pad is configured with a prescaler, there will be a 1 MΩ resistor to ground. This occurs even when the device is in power-down mode. In low power standby or sleep mode (V_{CC} is OFF, V_{CC33A} is ON, V_{CCI} is ON) or when the resource is not used, analog inputs are pulled down to ground through a 1 MΩ resistor. The gate driver output is floating (or tristated), and there is no extra current on V_{CC33A}.

These scaling factors hold true whether the particular pad is configured to accept a positive or negative voltage. Note that whereas the AV and AC pads support the same prescaling factors, the AT pad supports a reduced set of prescaling factors and supports positive voltages only.

Typical scaling factors are given in Table 2-54 on page 2-131, and the gain error (which contributes to the minimum and maximum) is in Table 2-46 on page 2-118.

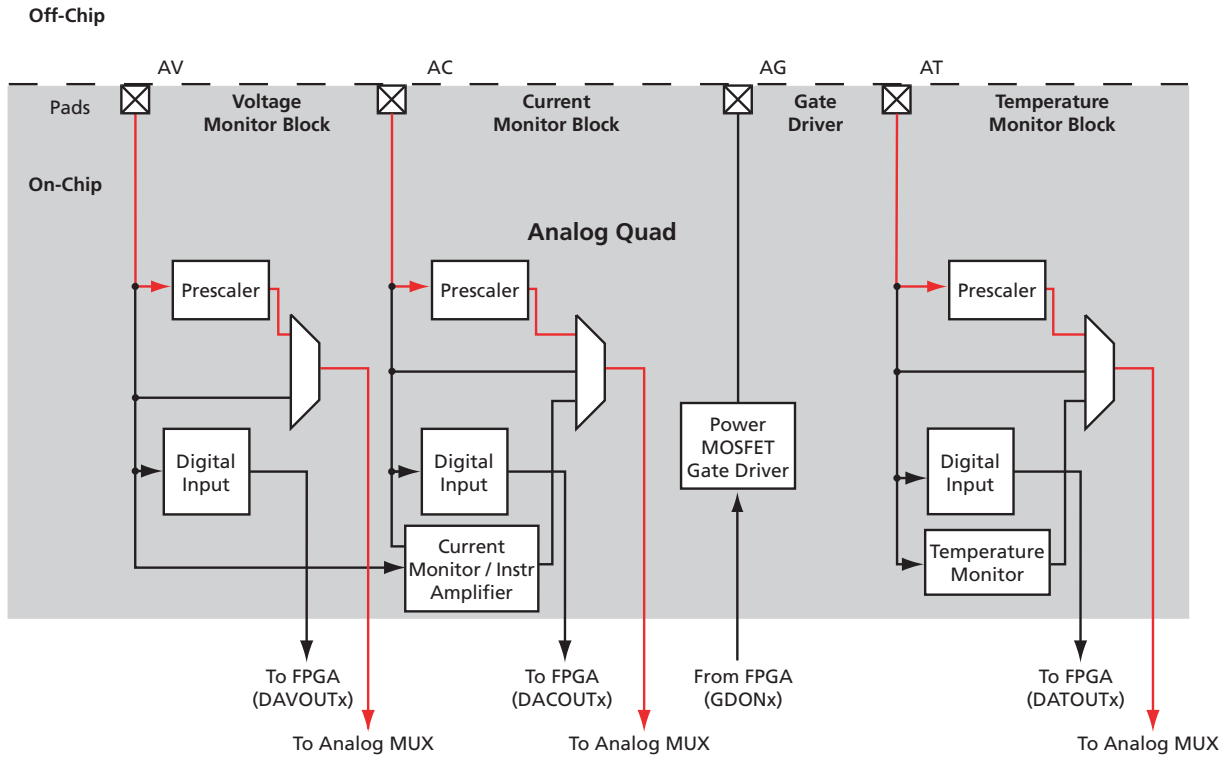


Figure 2-68 • **Analog Quad Prescaler Input Configuration**

Terminology

BW – Bandwidth

BW is a range of frequencies that a Channel can handle.

Channel

A channel is define as an analog input configured as one of the Prescaler range shown in Table 2-54 on page 2-131. The channel includes the Prescaler circuit and the ADC.

Channel Gain

Channel Gain is a measured of the deviation of the actual slope from the ideal slope. The slope is measured from the 20% and 80% point.

$$\text{Gain} = \frac{\text{Gain}_{\text{actual}}}{\text{Gain}_{\text{ideal}}}$$

EQ 2-1

Channel Gain Error

Channel Gain Error is a deviation from the ideal slope of the transfer function. The Prescaler Gain Error is expressed as the percent difference between the actual and ideal, as shown in EQ 2-2.

$$\text{Error}_{\text{Gain}} = (1\text{-Gain}) \times 100\%$$

EQ 2-2

Channel Input Offset Error

Channel Offset error is measured as the input voltage that causes the transition from zero to a count of one. An Ideal Prescaler will have offset equal to ½ of LSB voltage. Offset error is a positive or negative when the first transition point is higher or lower than ideal. Offset error is expressed in LSB or input voltage.

Total Channel Error

Total Channel Error is defined as the total error measured compared to the ideal value. Total Channel Error is the sum of gain error and offset error combined. Figure 2-69 shows how Total Channel Error is measured.

Total Channel Error is defined as the difference between the actual ADC output and ideal ADC output. In the example shown in Figure 2-69, the Total Channel Error would be a negative number.

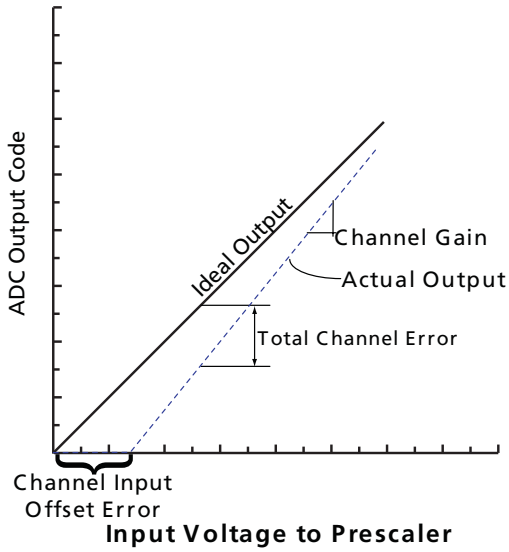


Figure 2-69 • **Total Channel Error Example**

Direct Digital Input

The AV, AC, and AT pads can also be configured as high-voltage digital inputs (Figure 2-70). As these pads are 12 V–tolerant, the digital input can also be up to 12 V. However, the frequency at which these pads can operate is limited to 10 MHz.

To enable one of these analog input pads to operate as a digital input, its corresponding Digital Input Enable (DENA*xy*) pin on the Analog Block must be pulled HIGH, where *x* is either V, C, or T (for AV, AC, or AT pads, respectively) and *y* is in the range 0 to 9, corresponding to the appropriate Analog Quad.

When the pad is configured as a digital input, the signal will come out of the Analog Block macro on the appropriate DA*x*OUT*y* pin, where *x* represents the pad type (V for AV pad, C for AC pad, or T for AT pad) and *y* represents the appropriate Analog Quad number. Example: If the AT pad in Analog Quad 5 is configured as a digital input, it will come out on the DATOUT5 pin of the Analog Block macro.

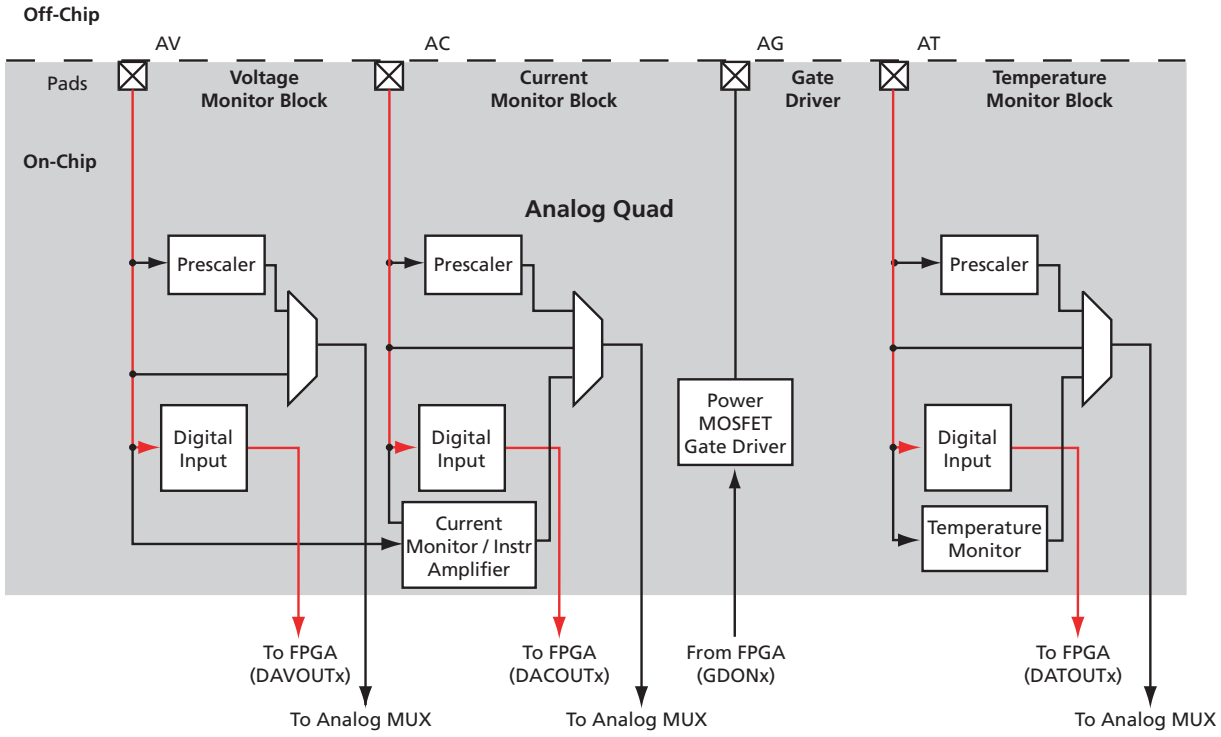


Figure 2-70 • **Analog Quad Direct Digital Input Configuration**

Current Monitor

The Fusion Analog Quad is an excellent element for voltage- and current-monitoring applications. In addition to supporting the same functionality offered by the AV pad, the AC pad can be configured to monitor current across an external sense resistor (Figure 2-71). To support this current monitor function, a differential amplifier with 10x gain passes the amplified voltage drop between the AV and AC pads to the ADC. The amplifier enables the user to use very small resistor values, thereby limiting any impact on the circuit. This function of the AC pad does not limit AV pad operation. The AV pad can still be configured for use as a direct voltage input or scaled through the AV prescaler independently of it's use as an input to the AC pad's differential amplifier.

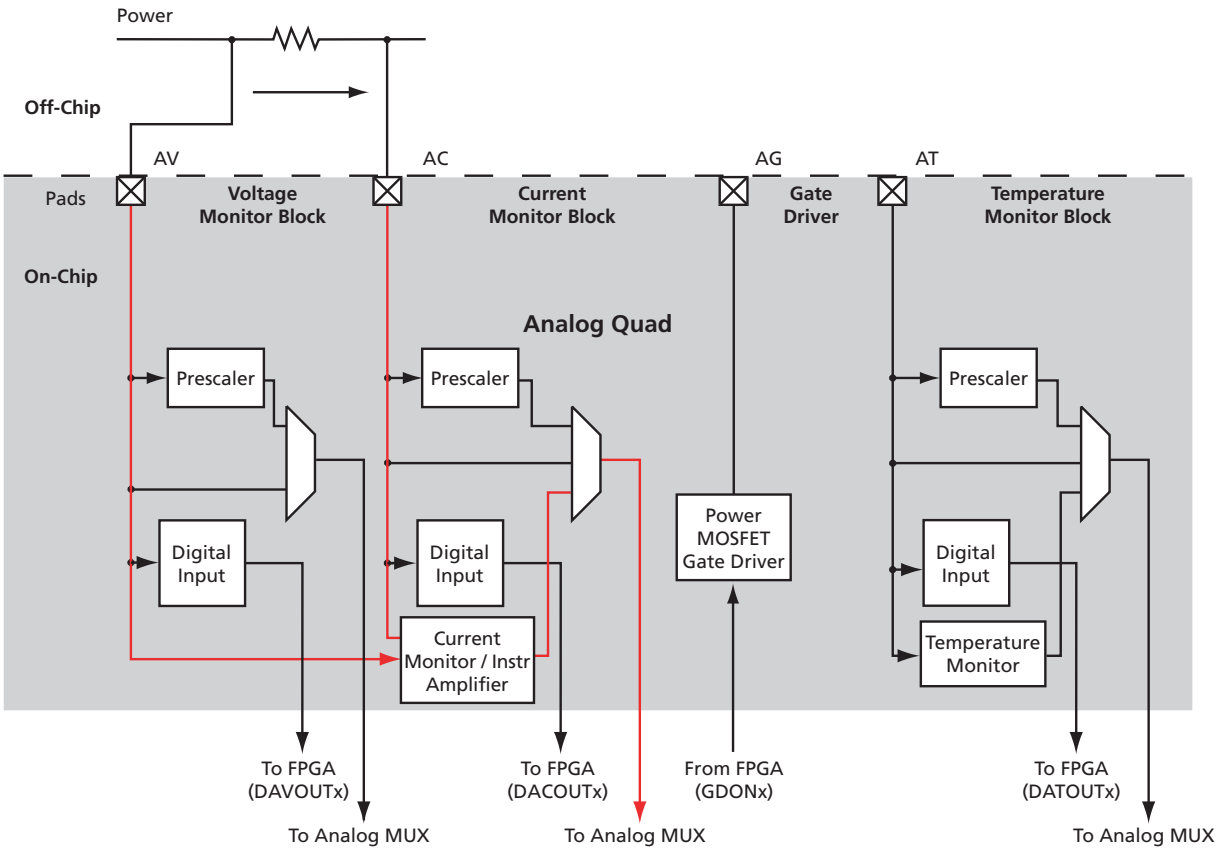


Figure 2-71 • **Analog Quad Current Monitor Configuration**

To initiate a current measurement, the appropriate Current Monitor Strobe (CMSTB) signal on the AB macro must be asserted low for at least t_{CMSLO} in order to discharge the previous measurement. Then CMSTB must be asserted high for at least t_{CMSET} prior to asserting the ADCSTART signal. The CMSTB must remain high until after the SAMPLE signal is de-asserted by the AB macro. Note that the minimum sample time cannot be less than t_{CMSHI}. Figure 2-72 shows the timing diagram of CMSTB in relationship with the ADC control signals.

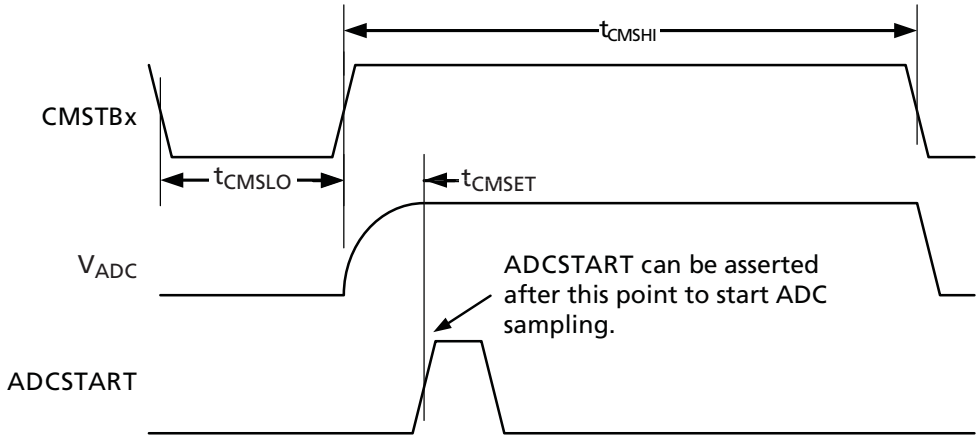


***Figure 2-72* • Timing Diagram for Current Monitor Strobe**

Figure 2-73 illustrates positive current monitor operation. The differential voltage between AV and AC goes into the 10× amplifier and is then converted by the ADC. For example, a current of 1.5 A is drawn from a 10 V supply and is measured by the voltage drop across a 0.050 Ω sense resistor, The voltage drop is amplified by ten times by the amplifier and then measured by the ADC. The 1.5 A current creates a differential voltage across the sense resistor of 75 mV. This becomes 750 mV after amplification. Thus, the ADC measures a current of 1.5 A as 750 mV. Using an ADC with 8-bit resolution and VAREF of 2.56 V, the ADC result is decimal 75. EQ 2-3 shows how to compute the current from the ADC result.

$$|I| = (ADC \times V_{AREF})/(10 \times 2^{N} \times R_{sense})$$

EQ 2-3

where

I is the current flowing through the sense resistor

ADC is the result from the ADC

VAREF is the Reference voltage

N is the number of bits

Rsense is the resistance of the sense resistor

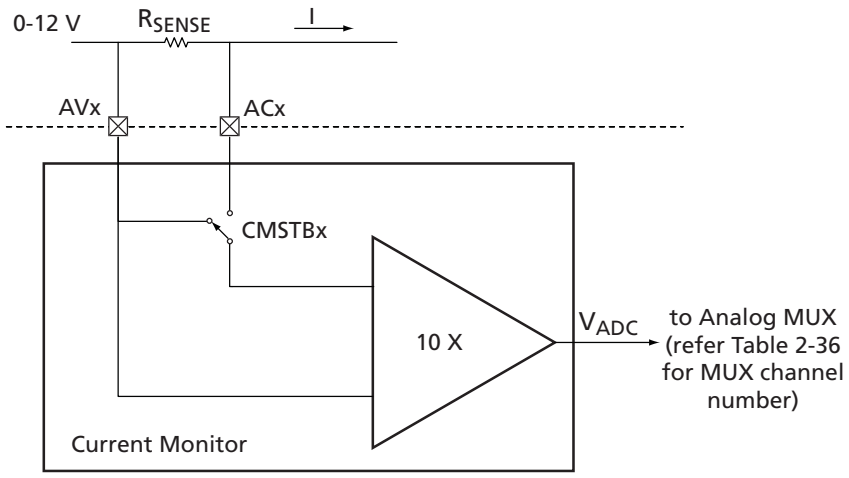


Figure 2-73 • **Positive Current Monitor**

Care must be taken when choosing the right resistor for current measurement application. Note that because of the 10× amplification, the maximum measurable difference between the AV and AC pads is V_{AREF} / 10. A larger AV-to-AC voltage drop will result in ADC saturation; that is, the digital code put out by the ADC will stay fixed at the full scale value. Therefore, the user must select the external sense resistor appropriately. Table 2-38 shows recommended resistor values for different current measurement ranges. When choosing resistor values for a system, there is a trade-off between measurement accuracy and power consumption. Choosing a large resistor will increase the voltage drop and hence increase accuracy of the measurement; however the larger voltage drop dissipates more power ($P = I^2 \times R$).

The Current Monitor is a unipolar system, meaning that the differential voltage swing must be from 0 V to $V_{AREF}/10$. Therefore, the Current Monitor only supports differential voltage where $|V_{AV}-V_{AC}|$ is greater than 0 V. This results in the requirement that the potential of the AV pad must be larger than the potential of the AC pad. This is straightforward for positive voltage systems. For a negative voltage system, it means that the AV pad must be "more negative" than the AC pad. This is shown in Figure 2-74.

In this case, both the AV pad and the AC pad are configured for negative operations and the output of the differential amplifier still falls between 0 V and V_{AREF} as required.

Table 2-37 • **Recommended Resistor for Different Current Range Measurement**

Current Range	Recommended Minimum Resistor Value (Ohms)
> 5 mA – 10 mA	10 – 20
> 10 mA – 20 mA	5 – 10
> 20 mA – 50 mA	2.5 – 5
> 50 mA – 100 mA	1 – 2
> 100 mA – 200 mA	0.5 – 1
> 200 mA – 500 mA	0.3 – 0.5
> 500 mA – 1 A	0.1 – 0.2
> 1 A – 2 A	0.05 – 0.1
> 2 A – 4 A	0.025 – 0.05
> 4 A – 8 A	0.0125 – 0.025
> 8 A – 12 A	0.00625 – 0.02

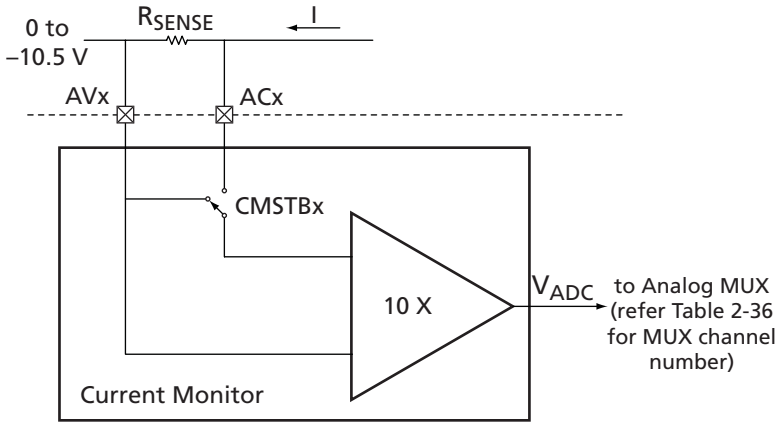


Figure 2-74 • **Negative Current Monitor**

Terminology

Accuracy

The accuracy of Fusion Current Monitor is ±2 mV minimum plus 5% of the differential voltage at the input. The input accuracy can be translated to error at the ADC output by using EQ 2-4. The 10 V/V gain is the gain of the Current Monitor Circuit, as described in the "Current Monitor" section on page 2-90. For 8-bit mode, N = 8, V_{AREF}= 2.56 V, zero differential voltage between AV and AC, the Error (E_{ADC}) is equal to 2 LSBs.

$$E_{ADC} = (2mV + 0.05|V_{AV} - V_{AC}|) \times (10V)/V \times \frac{2^N}{V_{AREF}}$$

EQ 2-4

where

N is the number of bits

V_{AREF} is the Reference voltage

V_{AV} is the voltage at AV pad

V_{AC} is the voltage at AC pad

Gate Driver

The Fusion Analog Quad includes a Gate Driver connected to the Quad's AG pin (Figure 2-75). Designed to work with external p- or n-channel MOSFETs, the Gate driver is a configurable current sink or source and requires an external pull-up or pull-down resistor. The AG supports 4 selectable gate drive levels: 1 µA, 3 µA, 10 µA, and 30 µA (Figure 2-76 on page 2-95). The AG also supports a High Current Drive mode in which it can sink 20 mA; in this mode the switching rate is approximately 1.3 MHz with 100 ns turn-on time and 600 ns turn-off time. Modeled on an open-drain-style output, it does not output a voltage level without an appropriate pull-up or pull-down resistor. If 1 V is forced on the drain, the current sinking/sourcing will exceed the ability of the transistor, and the device could be damaged.

The AG pad is turned on via the corresponding GDON*x* pin in the Analog Block macro, where *x* is the number of the corresponding Analog Quad for the AG pad to be enabled (GDON0 to GDON9).

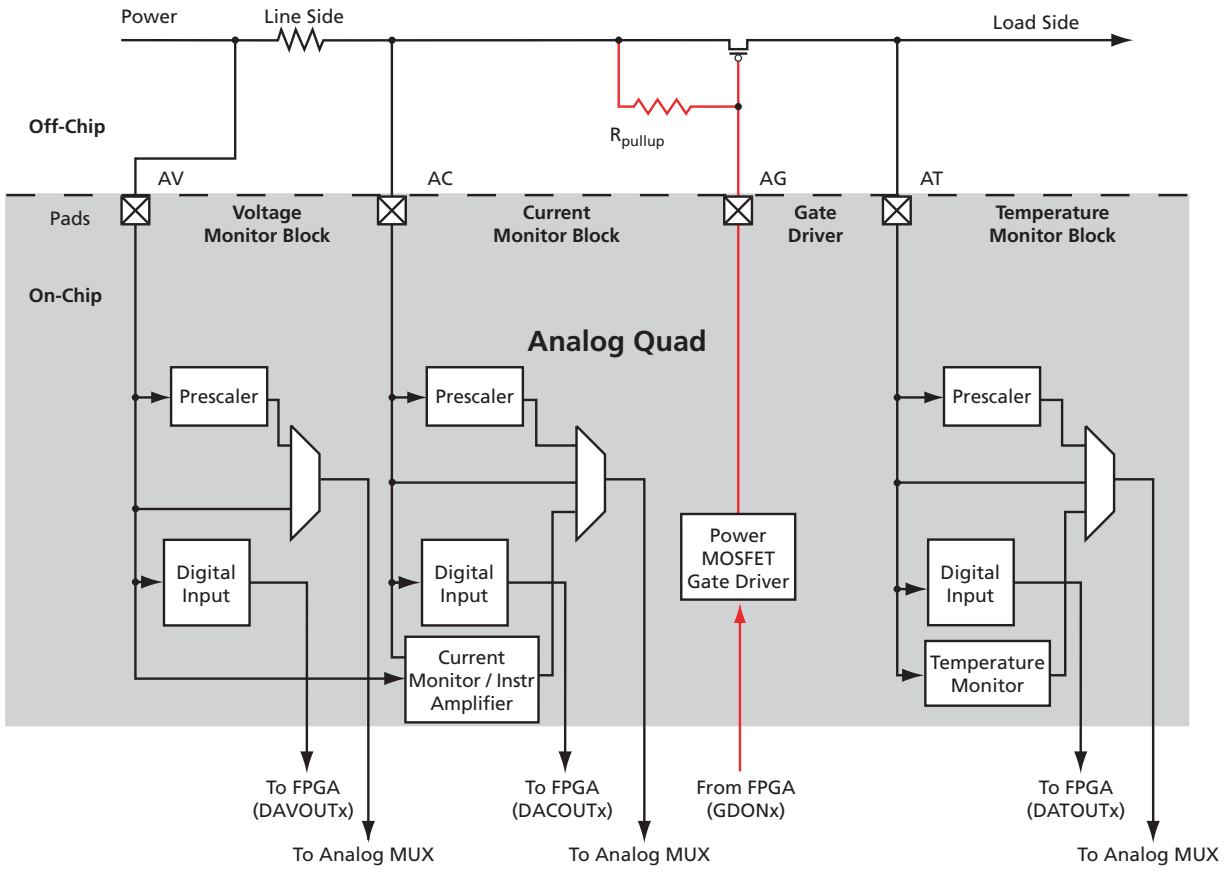


Figure 2-75 • **Gate Driver**

The gate-to-source voltage (V_{gs}) of the external MOSFET is limited to the programmable drive current times the external pull-up or pull-down resistor value (EQ 2-5).

$$V_{gs} \le I_g \times (R_{pullup} \text{ or } R_{pulldown})$$

EQ 2-5

The rate at which the gate voltage of the external MOSFET slews is determined by the current, I_g, sourced or sunk by the AG pin and the gate-to-source capacitance, C_{GS}, of the external MOSFET. As an approximation, the slew rate is given by EQ 2-6.

$$dv/dt = I_g / C_{GS}$$

EQ 2-6

C_{GS} is not a fixed capacitance but, depending on the circuitry connected to its drain terminal, can vary significantly during the course of a turn-on or turn-off transient. Thus, EQ 2-6 on page 2-94 can only be used for a first-order estimate of the switching speed of the external MOSFET.

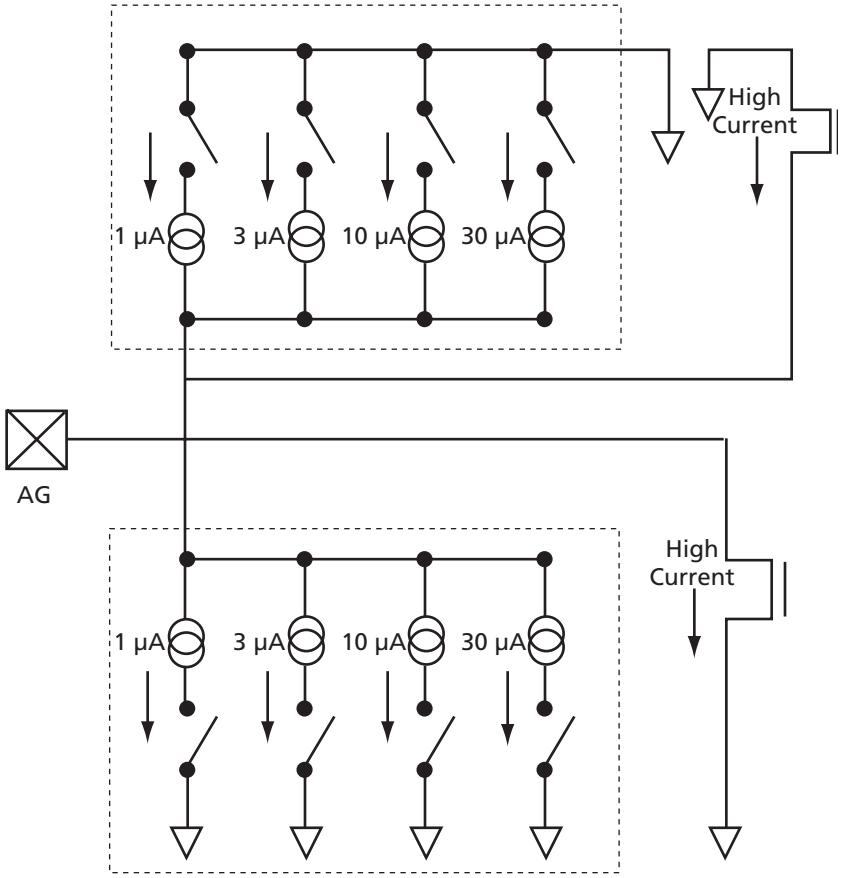


Figure 2-76 • **Gate Driver Example**

Temperature Monitor

The final pin in the Analog Quad is the Analog Temperature (AT) pin. The AT pin is used to implement an accurate temperature monitor in conjunction with an external diode-connected bipolar transistor (Figure 2-77). For improved temperature measurement accuracy, it is important to use the ATRTN pin for the return path of the current sourced by the AT pin. Each ATRTN pin is shared between two adjacent Analog Quads. Additionally, if not used for temperature monitoring, the AT pin can provide functionality similar to that of the AV pad. However, in this mode only positive voltages can be applied to the AT pin, and only two prescaler factors are available (16 V and 4 V ranges—refer to Table 2-54 on page 2-131).

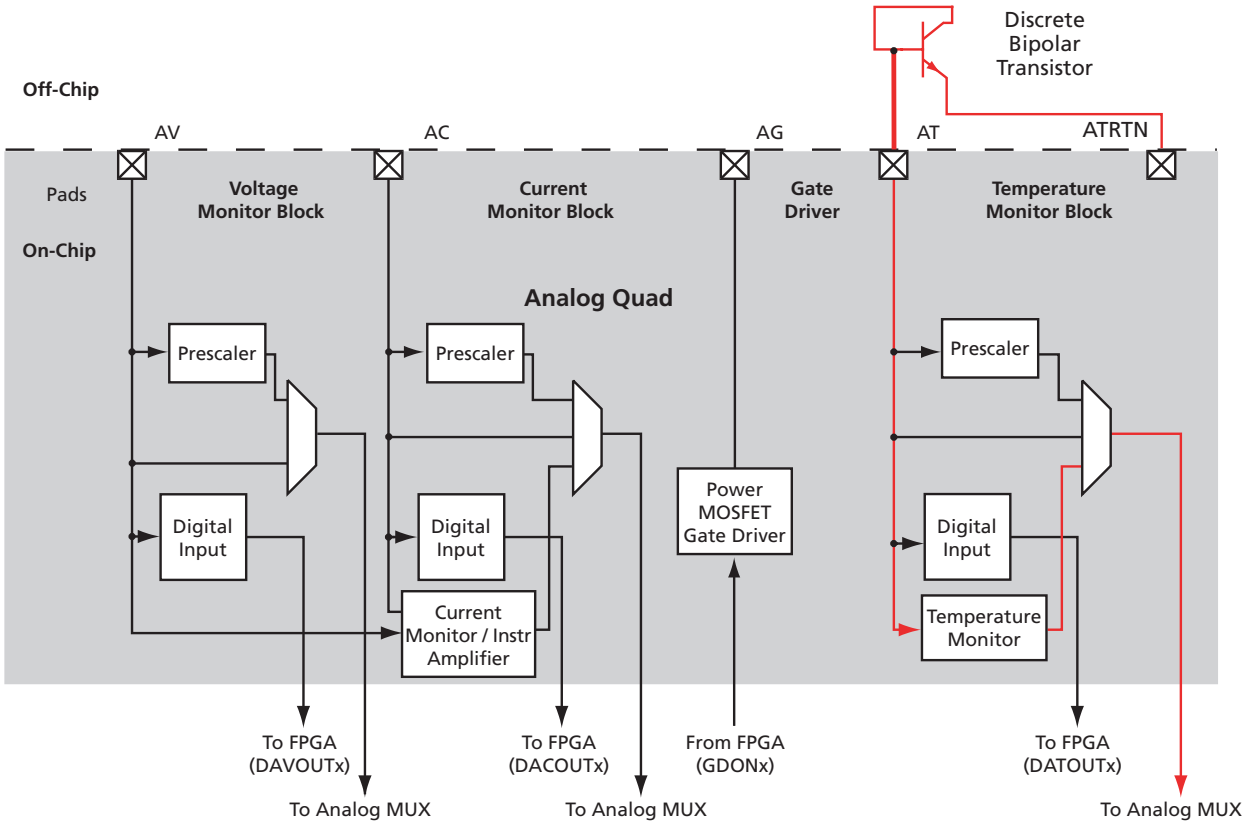


Figure 2-77 • **Temperature Monitor Quad**

Fusion uses a remote diode as a temperature sensor. The Fusion Temperature Monitor uses a differential input; the AT pin and ATRTN (AT Return) pin are the differential inputs to the Temperature Monitor. There is one Temperature Monitor in each Quad. A simplified block diagram is shown in Figure 2-78.

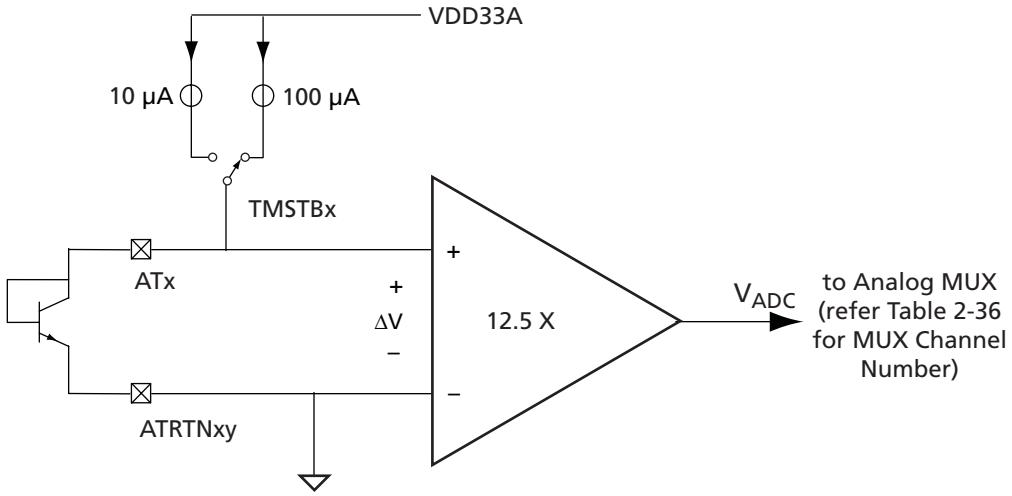


***Figure 2-78* • Block Diagram for Temperature Monitor Circuit**

The Fusion approach to measuring temperature is forcing two different currents through the diode with a ratio of 10:1. The switch that controls the different currents is controlled by the Temperature Monitor Strobe signal, TMSTB. Setting TMSTB to '1' will initiate a Temperature reading. The TMSTB should remain '1' until the ADC finishes sampling the voltage from the Temperature Monitor. The minimum sample time for the Temperature Monitor cannot be less than the minimum strobe high time minus the setup time. Figure 2-79 shows the timing diagram.

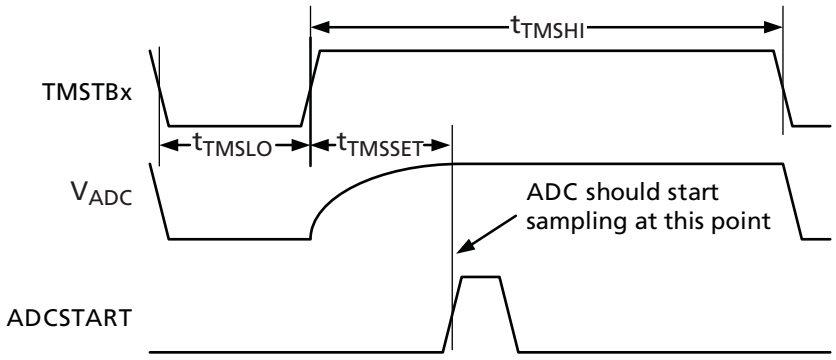


***Figure 2-79* • Timing Diagram for the Temperature Monitor Strobe Signal**

The diode's voltage is measured at each current level and the temperature is calculated based on EQ 2-7.

$$V_{TMSLO} - V_{TMSHI} = n\frac{kT}{q}\left(\ln\frac{I_{TMSLO}}{I_{TMSHI}}\right)$$

EQ 2-7

where

I_{TMSLO} is the current when the Temperature Strobe is Low, typically 100 µA

I_{TMSHI} is the current when the Temperature Strobe is High, typically 10 µA

V_{TMSLO} is diode voltage while Temperature Strobe is Low

V_{TMSHI} is diode voltage while Temperature Strobe is High

n is the non-ideality factor of the diode-connected transistor. It is typically 1.004 for the Actel-recommended transistor type 2N3904.

K = 1.3806 x 10^{-23} J/K is the Boltzman constant

Q = 1.602 x 10^{-19} C is the charge of a proton

When I_{TMSLO} / I_{TMSHI} = 10, the equation can be simplified as shown in EQ 2-8.

$$\Delta V = V_{TMSLO} - V_{TMSHI} = 1.986 \times 10^{-4} nT$$

EQ 2-8

In the Fusion TMB, the ideality factor n for 2N3904 is 1.004 and ΔV is amplified 12.5 times by an internal amplifier; hence the voltage before entering the ADC is as given in EQ 2-9.

$$V_{ADC} = \Delta V \times 12.5 = 2.5\ \text{mV}/(K \times T)$$

EQ 2-9

This means the temperature to voltage relationship is 2.5 mV per degree Kelvin. The unique design of Fusion has made the Temperature Monitor System simple for the user. When the 10-bit mode ADC is used, each LSB represents 1 degree Kelvin, as shown in EQ 2-10. That is, e. 25°C is equal to 293°K and is represented by decimal 293 counts from the ADC.

$$1K = 2.5\ \text{mV} \times \frac{2^{10}}{2.56\ \text{V}} = 1\ \text{LSB}$$

EQ 2-10

If 8-bit mode is used for the ADC resolution, each LSB represents 4 degrees Kelvin; however, the resolution remains as 1 degree Kelvin per LSB, even for 12-bit mode, due to the Temperature Monitor design. An example of the temperature data format for 10-bit mode is shown in Table 2-38.

Table 2-38 • **Temperature Data Format**

Temperature	Temperature (K)	Digital Output (ADC 10-bit mode)
–40°C	233	00 1110 1001
–20°C	253	00 1111 1101
0°C	273	01 0001 0001
1°C	274	01 0001 0010
10 °C	283	01 0001 1011
25°C	298	01 0010 1010
50 °C	323	01 0100 0011
85 °C	358	01 0110 0110

Terminology

Resolution

Resolution defines the smallest temperature change Fusion Temperature Monitor can resolve. For ADC configured as 8-bit mode, each LSB represents 4°C, and 1°C per LSB for 10-bit mode. With 12-bit mode, the Temperature Monitor can still only resolve 1°C due to Temperature Monitor design.

Offset

The Fusion Temperature Monitor has a systematic offset of +5°C, excluding error due board resistance and ideality factor of the external diode, between the operation range of –40°C to +85°C. For instance, 25°C will be read by the Temperature Monitor as 30°C plus error. The user can remove any offset error through hardware or software during the calibration routine.

Analog-to-Digital Converter Block

At the heart of the Fusion analog system is a programmable Successive Approximation Register (SAR) ADC. The ADC can support 8-, 10-, or 12-bit modes of operation. In 12-bit mode, the ADC can resolve 500 ksps. All results are MSB-justified in the ADC. The input to the ADC is a large 32:1 analog input multiplexer. A simplified block diagram of the Analog Quads, analog input multiplexer, and ADC is shown in Figure 2-80. The ADC offers multiple self-calibrating modes to ensure consistent high performance both at power-up and during runtime.

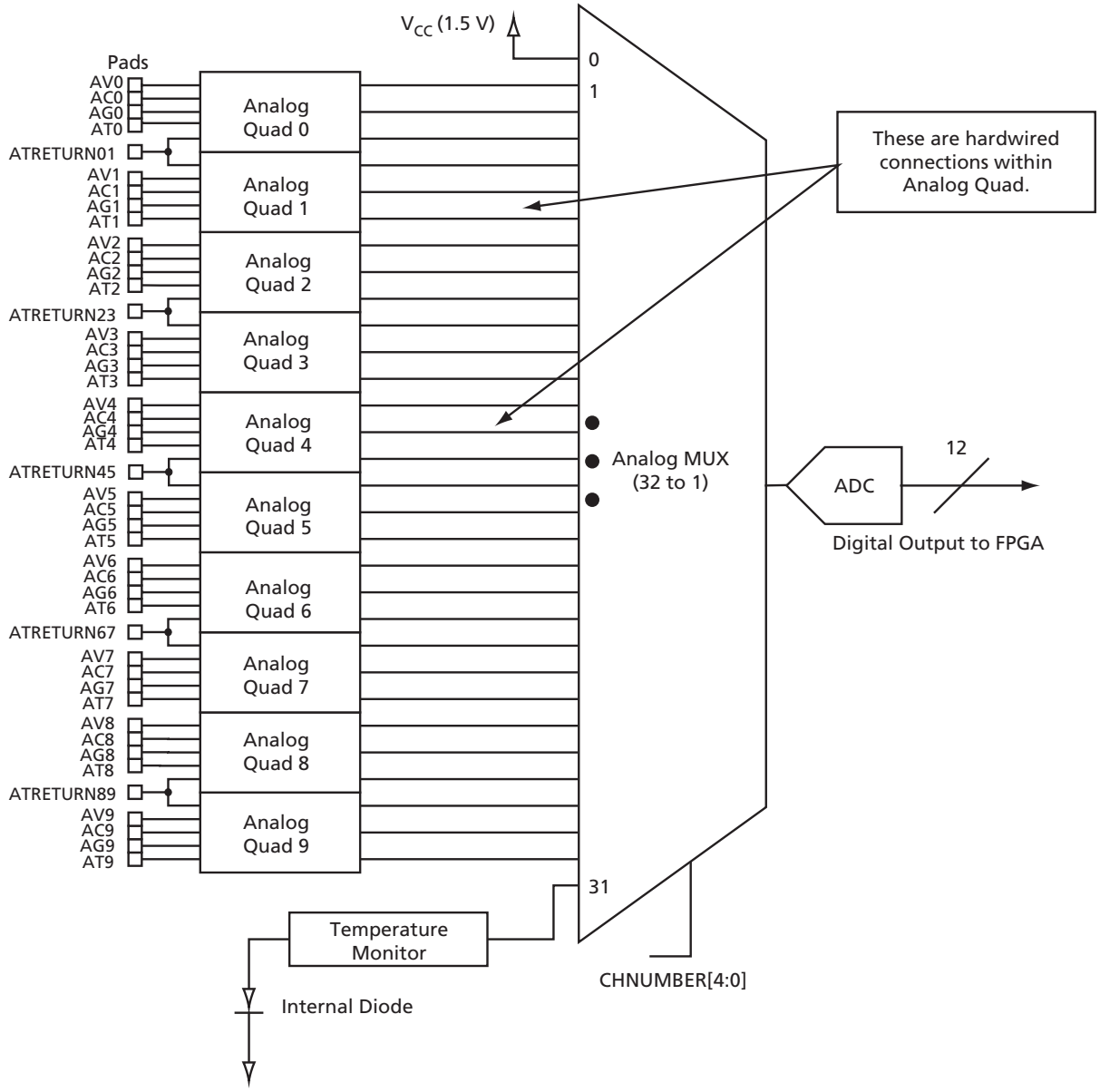


Figure 2-80 • **ADC Block Diagram**

ADC Input Multiplexer

At the input to the Fusion ADC is a 32:1 multiplexer. Of the 32 input channels, up to 30 are user definable. Two of these channels are hardwired internally. Channel 31 connects to an internal temperature diode so the temperature of the Fusion device itself can be monitored. Channel 0 is wired to the FPGA's 1.5 V V_{CC} supply, enabling the Fusion device to monitor its own power supply. Doing this internally makes it unnecessary to use an analog I/O to support these functions. The balance of the MUX inputs are connected to Analog Quads (see the "Analog Quad" section on page 2-84). Table 2-39 defines which Analog Quad inputs are associated with which specific analog MUX channels. The number of Analog Quads present is device-dependent; refer to the family list in the "Fusion Family" table on page I of this datasheet for the number of quads per device. Regardless of the number of quads populated in a device, the internal connections to both V_{CC} and the internal temperature diode remain on Channels 0 and 31, respectively. To sample the internal temperature monitor, it must be strobed (similar to the AT pads). The TMSTBINT pin on the Analog Block macro is the control for strobing the internal temperature measurement diode.

To determine which channel is selected for conversion, there is a five-pin interface on the Analog Block, CHNUMBER[4:0], defined in Table 2-40 on page 2-101. Table 2-39 shows the correlation between the analog MUX input channels and the analog input pins.

Table 2-39 • **Analog MUX Channels**

Analog MUX Channel	Signal	Analog Quad Number
0	Vcc_analog	
1	AV0	Analog Quad 0
2	AC0	
3	AT0	
4	AV1	Analog Quad 1
5	AC1	
6	AT1	
7	AV2	Analog Quad 2
8	AC2	
9	AT2	
10	AV3	Analog Quad 3
11	AC3	
12	AT3	
13	AV4	Analog Quad 4
14	AC4	
15	AT4	
16	AV5	Analog Quad 5
17	AC5	
18	AT5	
19	AV6	Analog Quad 6
20	AC6	
21	AT6	

Table 2-39 • **Analog MUX Channels (continued)**

Analog MUX Channel	Signal	Analog Quad Number
22	AV7	Analog Quad 7
23	AC7	
24	AT7	
25	AV8	Analog Quad 8
26	AC8	
27	AT8	
28	AV9	Analog Quad 9
29	AC9	
30	AT9	
31	Internal temperature monitor	

Table 2-40 • **Channel Selection**

Channel Number	CHNUMBER[4:0]
0	00000
1	00001
2	00010
3	00011
. . .	. . .
30	11110
31	11111

ADC Description

The Actel Fusion ADC is a 12-bit SAR ADC. It offers a wide variety of features for different use models. Figure 2-81 shows a block diagram of the Fusion ADC.

- Configurable resolution: 8-bit, 10-bit, and 12-bit mode
- DNL: 0.6 LSB for 10-bit mode
- INL: 0.4 LSB for 10-bit mode
- No missing code
- Internal VAREF = 2.56 V
- Maximum Sample Rate = 600 Ksps
- Power-up calibration and dynamic calibration after every sample to compensate for temperature drift over time

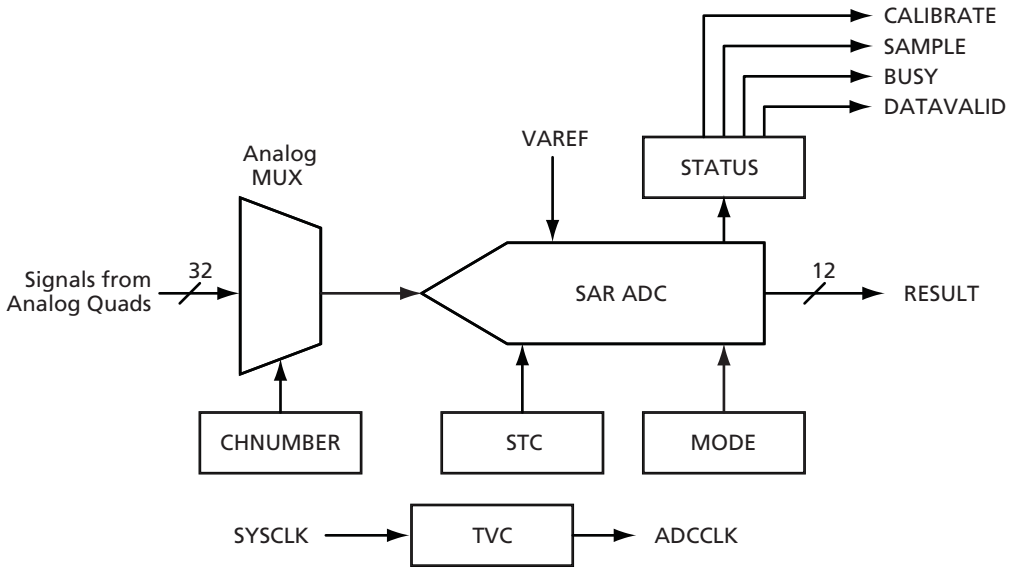


Figure 2-81 • **ADC Simplified Block Diagram**

ADC Configuration Description

The Fusion ADC can be configured to operate in 8-, 10-, or 12-bit modes, power-down after conversion, and dynamic calibration. This is controlled by MODE[3:0], as defined in Table 2-41 on page 2-103.

The output of the ADC is the RESULT[11:0] signal. In 8-bit mode, the Most Significant 8 Bits RESULT[11:4] are used as the ADC value and the Least Significant 4 Bits RESULT[3:0] are logical '0's. In 10-bit mode, RESULT[11:2] are used the ADC value and RESULT[1:0] are logical '0's.

Table 2-41 • **Mode Bits Function**

Name	Bits	Function
MODE	3	0 – Internal calibration after every conversion; two ADCCLK cycles are used after the conversion. 1 – No calibration after every conversion
MODE	2	0 – Power-down after conversion 1 – No Power-down after conversion
MODE	1:0	00 – 10-bit 01 – 12-bit 10 – 8-bit 11 – Unused

The speed of the ADC depends on its internal clock, ADCCLK, which is not accessible to users. The ADCCLK is derived from SYSCLK. Input signal TVC[7:0], Time Divider Control, determines the speed of the ADCCLK in relationship to SYSCLK, based on EQ 2-11.

$$t_{ADCCLK} = 4 \times (1 + TVC) \times t_{SYSCLK}$$

EQ 2-11

TVC: Time Divider Control (0–255)

t_{ADCCLK} is the period of ADCCLK, and must be between 0.5 MHz and 10 MHz

t_{SYSCLK} is the period of SYSCLK

Table 2-42 • **TVC Bits Function**

Name	Bits	Function
TVC	[7:0]	SYSCLK divider control

The frequency of ADCCLK, f_{ADCCLK}, must be within 0.5 Hz to 10 MHz.

The inputs to the ADC are synchronized to SYSCLK. A conversion is initiated by asserting the ADCSTART signal on a rising edge of SYSCLK. Figure 2-83 on page 2-107 and Figure 2-84 on page 2-108 show the timing diagram for the ADC.

A conversion is performed in three phases. In the first phase, the analog input voltage is sampled on the input capacitor. This phase is called sample phase. During the sample phase, the output signals BUSY and SAMPLE change from '0' to '1', indicating the ADC is busy and sampling the analog signal. The sample time can be controlled by input signals STC[7:0]. The sample time can be calculated by EQ 2-12. When controlling the sample time for the ADC along with the use of Prescaler or Current Monitor or Temperature Monitor, the minimum sample time for each must be obeyed. Refer to the corresponding section and Table 2-43 for further information.

$$t_{sample} = (2 + STC) \times t_{ADCCLK}$$

EQ 2-12

STC: Sample Time Control value (0–255)

t_{SAMPLE} is the sample time

Table 2-43 • **STC Bits Function**

Name	Bits	Function
STC	[7:0]	Sample time control

Sample time is computed based on the period of ADCCLK.

The second phase is called the distribution phase. During distribution phase, the ADC computes the equivalent digital value from the value stored in the input capacitor. In this phase, the output

signal SAMPLE goes back to '0', indicating the sample is completed; but the BUSY signal remains '1', indicating the ADC is still busy for distribution. The distribution time depends strictly on the number of bits. If the ADC is configured as a 10-bit ADC, then 10 ADCCLK cycles are needed. EQ 2-13 describes the distribution time.

$$t_{distrib} = N \times t_{ADCCLK}$$

EQ 2-13

N: Number of bits

The last phase is the post-calibration phase. This is an optional phase. The post-calibration phase takes two ADCCLK cycles. The output BUSY signal will remain '1' until the post-calibration phase is completed. If the post-calibration phase is skipped, then the BUSY signal goes to '0' after distribution phase. As soon as BUSY signal goes to '0', the DATAVALID signal goes to '1', indicating the digital result is available on the RESULT output signals. DATAVAILD will remain '1' until the next ADCSTART is asserted. Actel recommends enabling post-calibration to compensate for drift and temperature-dependent effects. This ensures that the ADC remains consistent over time and with temperature. The post-calibration phase is enabled by bit 3 of the Mode register. EQ 2-14 describes the post-calibration time.

$$t_{post\text{-}cal} = MODE[3] \times (2 \times t_{ADCCLK})$$

EQ 2-14

MODE[3]: Bit 3 of the Mode register, described in Table 2-41 on page 2-103.

The calculation for the conversion time for the ADC is summarized in EQ 2-15.

$$t_{conv} = t_{sync_read} + t_{sample} + t_{distrib} + t_{post\text{-}cal} + t_{sync_write}$$

EQ 2-15

t_{conv}: conversion time

t_{sync_read}: maximum time for a signal to synchronize with SYSCLK. For calculation purposes, the worst case is a period of SYSCLK, t_{SYSCLK}.

t_{sample}: Sample time

$t_{distrib}$: Distribution time

$t_{post\text{-}cal}$: Post-calibration time

t_{sync_write}: Maximum time for a signal to synchronize with SYSCLK. For calculation purposes, the worst case is a period of SYSCLK, t_{SYSCLK}.

Example

This example shows how to choose the correct settings to achieve the fastest sample time in 10-bit mode for a system that runs at 66 MHz.

The period of SYSCLK: t_{SYSCLK} = 1/66 MHz = 0.015 µs

Choosing TVC between 1 and 33 will meet the maximum and minimum period for the ADCCLK requirement. A higher TVC leads to a higher ADCCLK period.

The minimum TVC is chosen so that $t_{distrib}$ and $t_{post\text{-}cal}$ can be run faster. The period of ADCCLK with a TVC of 1 can be computed by EQ 2-16.

$$t_{ADCCLK} = 4 \times (1 + TVC) \times t_{SYSCLK} = 4 \times (1 + 1) \times 0.015\ \mu s = 0.12\ \mu s$$

EQ 2-16

From Table 2-47 on page 2-121, minimum conversion for 10-bit mode is 1.8 µs. To compute STC, the calculation will first compute the post-calibration time, second the distribution time, and finally the STC setting.

Since Actel recommends post-calibration for temperature drift over time, post-calibration shall be enabled and the post-calibration time, $t_{post\text{-}cal}$, can be computed by EQ 2-17. The post-calibration time is 0.24 µs.

$$t_{post\text{-}cal} = 2 \times t_{ADCCLK} = 0.24\ \mu s$$

EQ 2-17

The distribution time, $t_{distrib}$, is equal to 1.2 µs and can be computed using EQ 2-18.

$$t_{distrib} = N \times t_{ADCCLK} = 10 \times 0.12 = 1.2\ \mu s$$

EQ 2-18

The STC value can now be computed through EQ 2-19. The sample time is equal to 0.32 µs. By rearranging EQ 2-12 on page 2-103 with a t_{sample} of 0.35 µs, the STC can be computed.

$$t_{sample} = t_{conv} - t_{post\text{-}cal} - t_{distrib} - t_{sync_read} - t_{sync_write}$$
$$= 1.8\ \mu s - 0.24\ \mu s - 1.2\ \mu s - 0.15\ \mu s - 0.15\ \mu s = 0.32\ \mu s$$

$$STC = \frac{t_{sample}}{t_{ADCCLK}} - 2 = \frac{0.35\ \mu s}{0.12\ \mu s} - 2 = 2.85$$

EQ 2-19

And so, STC will be rounded up to 3 to ensure the minimum conversion time is met. The sample time, t_{sample}, with an STC of 3, is now equal to 0.36 µs.

The total sample time, using EQ 2-20, can now be summated.

$$t_{sync_read} + t_{sample} + t_{distrib} + t_{post\text{-}cal} + t_{sync_write} = 0.015\ \mu s + 0.36\ \mu s + 1.2\ \mu s + 0.24\ \mu s + 0.015\ \mu s$$
$$= 1.85\ \mu s$$

EQ 2-20

The optimal setting for the system running at 66 MHz with an ADC for 10-bit mode chosen is listed as follows:

TVC[7:0]	= 1	= 0x01
STC[7:0]	= 3	= 0x03
MODE[3:0]	= b'0100	= 0x4*

*Note that no power-down after every conversion is chosen in this case; however, if the application is power-sensitive, the MODE[2] can be set to '0', as described above, and it will not affect any performance.

Integrated Voltage Reference

The Fusion device has an integrated on-chip 2.56 V reference voltage for the ADC. The value of this reference voltage was chosen to make the prescaling and postscaling factors for the prescaler blocks change in a binary fashion. However, if desired, an external reference voltage of up to 3.3 V can be connected between the VAREF and GNDREF pins. The VAREFSEL control pin is used to select the reference voltage.

Table 2-44 • **VAREF Bit Function**

Name	Bit	Function
VAREF	0	Reference voltage selection 0 – Internal voltage reference selected. VAREF pin outputs 2.56 V. 1 – Input external voltage reference from VAREF and GNDREF

ADC Operation Description

The ADC can be powered down independently of the FPGA core, as an additional control or for power-saving considerations, via the PWRDWN pin of the Analog Block. The PWRDWN pin controls only the comparators in the ADC.

Once the ADC has powered up and been released from reset, ADCRESET, the ADC will initiate a calibration routine designed to provide optimal ADC performance. The Fusion ADC offers a robust calibration scheme to reduce integrated offset and linearity errors. The offset and linearity errors of the main capacitor array are compensated for with an 8-bit calibration capacitor array. The offset/linearity error calibration is carried out in two ways. First, a power-up calibration is carried out when the ADC comes out of reset. This is initiated by the CALIBRATE output of the Analog Block macro and is a fixed number of ADC_CLK cycles (3,840 cycles), as shown in Figure 2-82 on page 2-107. In this mode, the linearity and offset errors of the capacitors are calibrated.

To further compensate for drift and temperature-dependent effects, every conversion is followed by post-calibration of either the offset or a bit of the main capacitor array. The post-calibration ensures that, over time and with temperature, the ADC remains consistent.

After both calibration and the setting of the appropriate configurations, as explained above, the ADC is ready for operation. Setting the ADCSTART signal high for one clock period will initiate the sample and conversion of the analog signal on the channel as configured by CHNUMBER[4:0]. The status signals SAMPLE and BUSY will show when the ADC is sampling and converting (Figure 2-84 on page 2-108). Both SAMPLE and BUSY will initially go high. After the ADC has sampled and held the analog signal, SAMPLE will go low. After the entire operation has completed and the analog signal is converted, BUSY will go low and DATAVALID will go high. This indicates that the digital result is available on the RESULT[11:0] pins.

DATAVALID will remain high until a subsequent ADC_START is issued. The DATAVALID goes low on the rising edge of SYSCLK as shown in Figure 2-83 on page 2-107. The RESULT signals will be kept constant until the ADC finishes the subsequent sample. The next sampled RESULT will be available when DATAVALID goes high again. It is ideal to read the RESULT when DATAVALID is '1'. The RESULT is latched and remains unchanged until the next DATAVLAID rising edge.

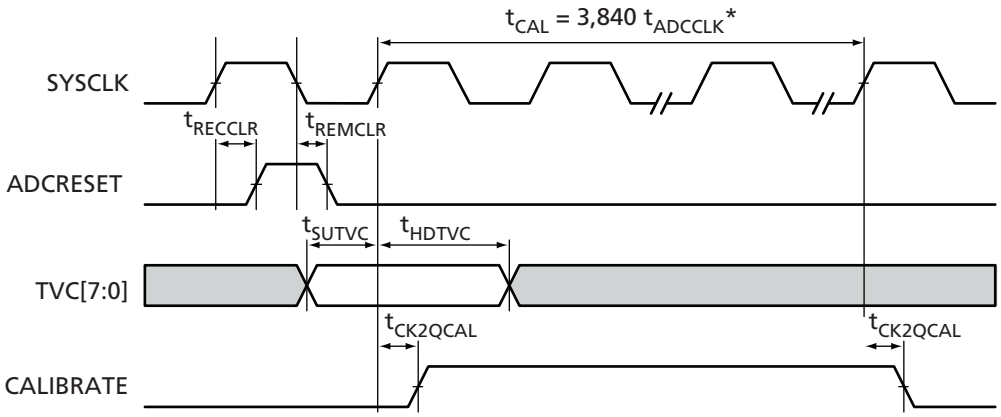
Intra-Conversion

Performing a conversion during power-up, calibration is possible but should be avoided, since the performance is not guaranteed, as shown in Table 2-46 on page 2-118. This is described as intra-conversion. Figure 2-85 on page 2-108 shows intra-conversion (conversion that starts before a conversion is finished).

Injected Conversion

A conversion can be interrupted by another conversion. Before the current conversion is finished, a second conversion can be started by issuing a pulse on signal ADCSTART. When a second conversion is issued before the current conversion is completed, the current conversion would be dropped and the ADC would start the second conversion on the rising edge of the SYSCLK. This is known as injected conversion. Since the ADC is synchronous, the minimum time to issue a second conversion is two clock cycles of SYSCLK after the previous one. Figure 2-86 on page 2-109 shows injected conversion (conversion that starts during the power-up calibration). The total time for calibration still remains 3,840 ADCCLK cycles.

Timing Diagram



*Note: *Refer to EQ 2-11 on page 2-103 for the calculation on the period of ADCCLK, t_{ADCCLK}.*

***Figure 2-82* • Power-Up Calibration Status Signal Timing Diagram**

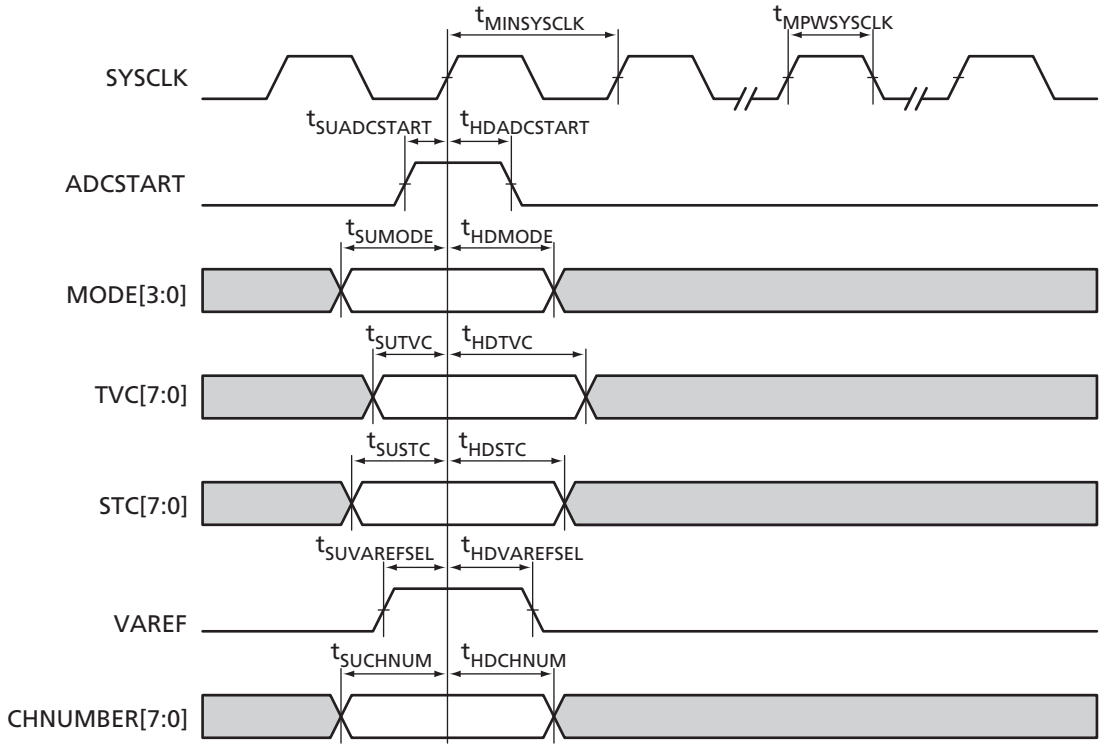
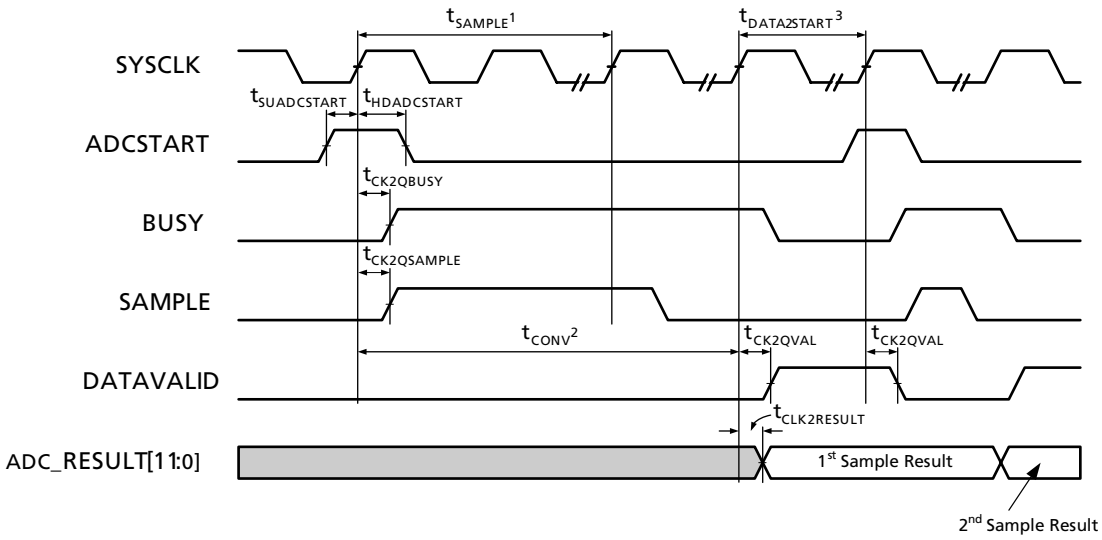


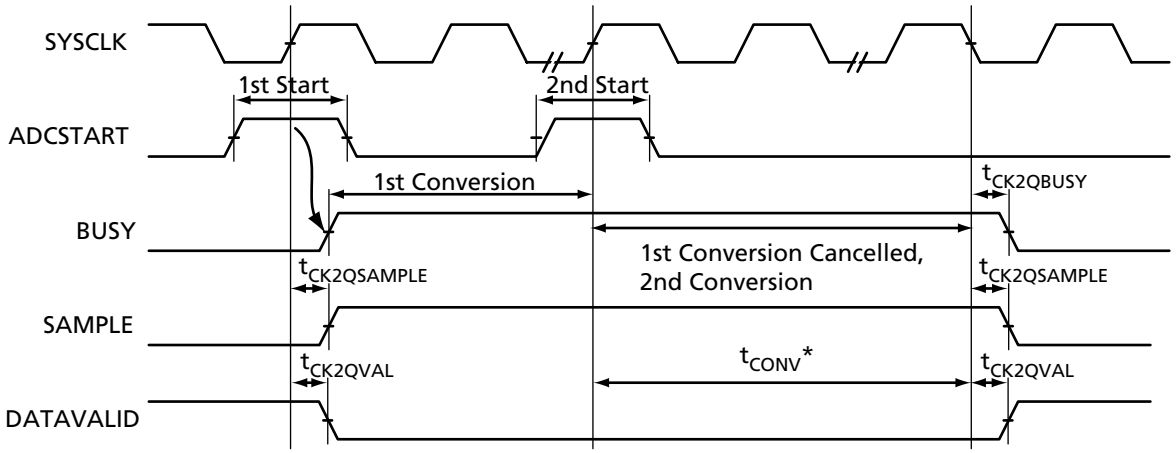
***Figure 2-83* • Input Setup Time**



Notes:

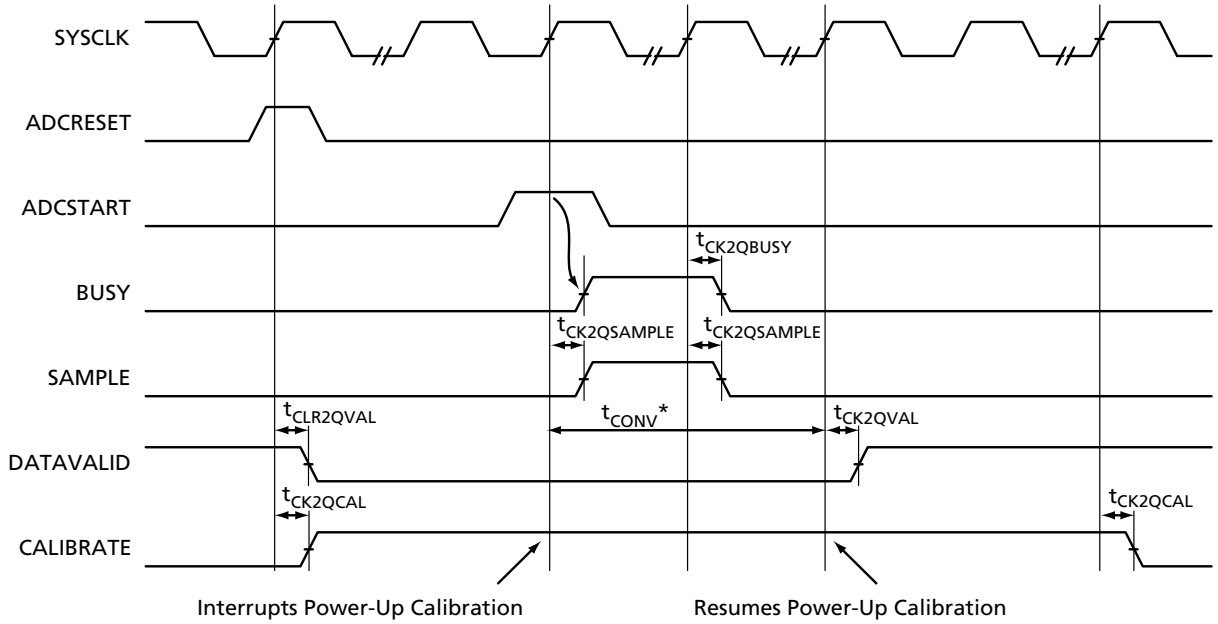
1. *Refer to EQ 2-12 on page 2-103 for the calculation on the sample time, t_{SAMPLE}.*
2. *See EQ 2-20 on page 2-105 for calculation on the conversion time, t_{CONV}.*
3. *Minimum time to issue an ADCSTART after DATAVALID is 1 SYSCLK period*

Figure 2-84 • **Standard Conversion Status Signal Timing Diagram**



*Note: *t_{CONV} represents the conversion time of the second conversion. See EQ 2-10 on page 2-98 for calculation of the conversion time, t_{CONV}.*

Figure 2-85 • **Intra-Conversion Timing Diagram**



*Note: * See EQ 2-10 on page 2-98 for calculation on the conversion time, t_{CONV}.*

Figure 2-86 • **Injected-Conversion Timing Diagram**

ADC Interface Timing

Table 2-45 • **ADC Interface Timing**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{SUMODE}	Mode Pin Setup Time	0.56	0.64	0.75	ns
t_{HDMODE}	Mode Pin Hold Time	0.26	0.29	0.34	ns
t_{SUTVC}	Clock Divide Control (TVC) Setup Time	0.68	0.77	0.90	ns
t_{HDTVC}	Clock Divide Control (TVC) Hold Time	0.32	0.36	0.43	ns
t_{SUSTC}	Sample Time Control (STC) Setup Time	1.58	1.79	2.11	ns
t_{HDSTC}	Sample Time Control (STC) Hold Time	1.27	1.45	1.71	ns
$t_{SUVAREFSEL}$	Voltage Reference Select (VAREFSEL) Setup Time	0.00	0.00	0.00	ns
$t_{HDVAREFSEL}$	Voltage Reference Select (VAREFSEL) Hold Time	0.67	0.76	0.89	ns
$t_{SUCHNUM}$	Channel Select (CHNUMBER) Setup Time	0.90	1.03	1.21	ns
$t_{HDCHNUM}$	Channel Select (CHNUMBER) Hold Time	0.00	0.00	0.00	ns
$t_{SUADCSTART}$	Start of Conversion (ADCSTART) Setup Time	0.75	0.85	1.00	ns
$t_{HDADCSTART}$	Start of Conversion (ADCSTART) Hold Time	0.43	0.49	0.57	ns
$t_{CK2QBUSY}$	Busy Clock-to-Q	1.33	1.51	1.78	ns
$t_{CK2QCAL}$	Power-Up Calibration Clock-to-Q	0.63	0.71	0.84	ns
$t_{CK2QVAL}$	Valid Conversion Result Clock-to-Q	3.12	3.55	4.17	ns
$t_{CK2QSAMPLE}$	Sample Clock-to-Q	0.22	0.25	0.30	ns
$t_{CK2QRESULT}$	Conversion Result Clock-to-Q	2.53	2.89	3.39	ns
$t_{CLR2QBUSY}$	Busy Clear-to-Q	2.06	2.35	2.76	ns
$t_{CLR2QCAL}$	Power-Up Calibration Clear-to-Q	2.15	2.45	2.88	ns
$t_{CLR2QVAL}$	Valid Conversion Result Clear-to-Q	2.41	2.74	3.22	ns
$t_{CLR2QSAMPLE}$	Sample Clear-to-Q	2.17	2.48	2.91	ns
$t_{CLR2QRESULT}$	Conversion result Clear-to-Q	2.25	2.56	3.01	ns
t_{RECCLR}	Recovery Time of Clear	0.00	0.00	0.00	ns
t_{REMCLR}	Removal Time of Clear	0.63	0.72	0.84	ns
$t_{MPWSYSCLK}$	Clock Minimum Pulse Width for the ADC	4.00	4.00	4.00	ns
$t_{FMAXSYSCLK}$	Clock Maximum Frequency for the ADC	100.00	100.00	100.00	MHz

Terminology

Conversion Time

Conversion time is the interval between the release of the hold state (imposed by the input circuitry of a track-and-hold) and the instant at which the voltage on the sampling capacitor settles to within one LSB of a new input value.

DNL – Differential Non-Linearity

For an ideal ADC, the analog-input levels that trigger any two successive output codes should differ by one LSB (DNL = 0). Any deviation from one LSB in defined as DNL (Figure 2-87).

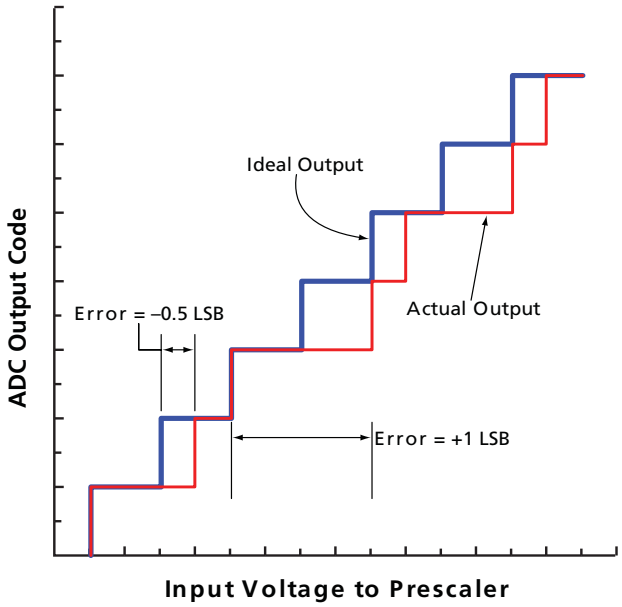


***Figure 2-87* • Differential Non-Linearity (DNL)**

ENOB – Effective Number of Bits

ENOB specifies the dynamic performance of an ADC at a specific input frequency and sampling rate. An ideal ADC's error consists only of quantization of noise. As the input frequency increases, the overall noise (particularly in the distortion components) also increases, thereby reducing the ENOB and SINAD (also see "Signal-to-Noise and Distortion Ratio (SINAD)".) ENOB for a full-scale, sinusoidal input waveform is computed using EQ 2-21.

$$ENOB = \frac{SINAD - 1.76}{6.02}$$

EQ 2-21

FS Error – Full-Scale Error

Full-scale error is the difference between the actual value that triggers that transition to full-scale and the ideal analog full-scale transition value. Full-scale error equals offset error plus gain error.

Gain Error

The gain error of an ADC indicates how well the slope of an actual transfer function matches the slope of the ideal transfer function. Gain error is usually expressed in LSB or as a percent of full-scale (%FSR). Gain error is the full-scale error minus the offset error (Figure 2-88).

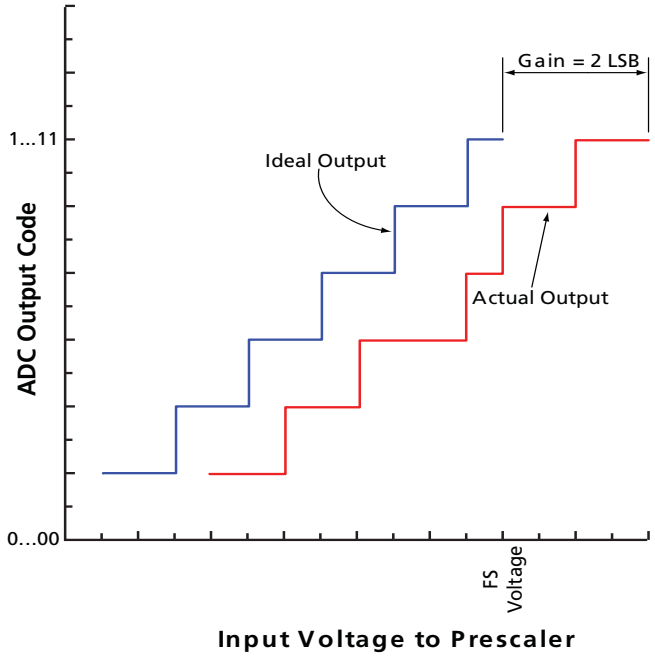


Figure 2-88 • **Gain Error**

Gain Error Drift

Gain-error drift is the variation in gain error due to a change in ambient temperature, typically expressed in ppm/°C.

INL – Integral Non-Linearity

INL is the deviation of an actual transfer function from a straight line. After nullifying offset and gain errors, the straight line is either a best-fit straight line or a line drawn between the end points of the transfer function (Figure 2-89).

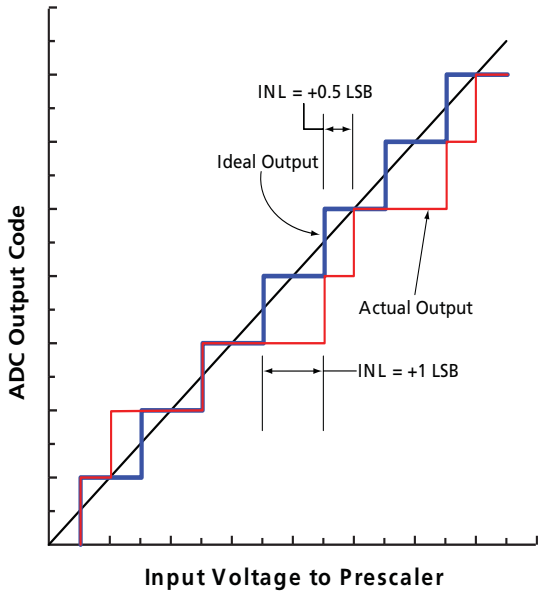


Figure 2-89 • **Integral Non-Linearity (INL)**

LSB – Least Significant Bit

In a binary number, the LSB is the least weighted bit in the group. Typically, the LSB is the furthest right bit. For an ADC, the weight of an LSB equals the full-scale voltage range of the converter divided by 2^N, where N is the converter's resolution.

EQ 2-22 shows the calculation for a 10-bit ADC with a unipolar full-scale voltage of 2.56 V:

$$1\ \text{LSB} = (2.56\ \text{V} / 2^{10}) = 2.5\ \text{mV}$$

EQ 2-22

No Missing Codes

An ADC has no missing codes if it produces all possible digital codes in response to a ramp signal applied to the analog input.

Offset Error

Offset error indicates how well the actual transfer function matches the ideal transfer function at a single point. For an ideal ADC, the first transition occurs at 0.5 LSB above zero. The offset voltage is measured by applying an analog input such that the ADC outputs all zeroes and increases until the first transition occurs (Figure 2-90).

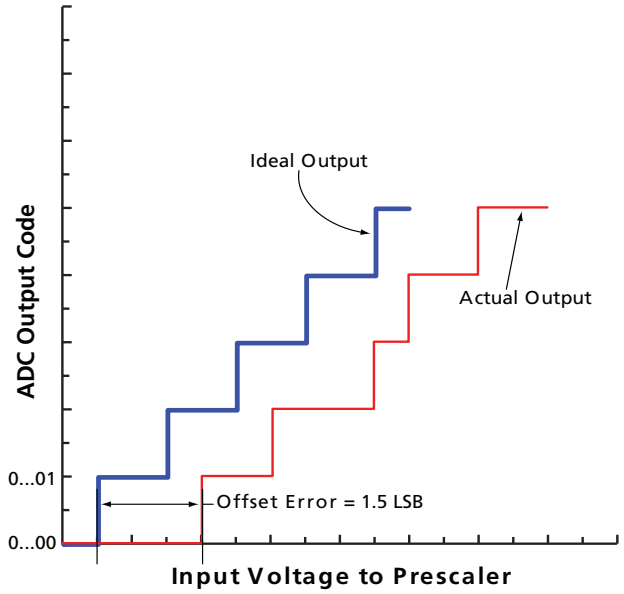


Figure 2-90 • **Offset Error**

Resolution

ADC resolution is the number of bits used to represent an analog input signal. To more accurately replicate the analog signal, resolution needs to be increased.

Sampling Rate

Sampling rate or sample frequency, specified in samples per second (sps), is the rate at which an ADC acquires (samples) the analog input.

SNR – Signal-to-Noise Ratio

SNR is the ratio of the amplitude of the desired signal to the amplitude of the noise signals at a given point in time. For a waveform perfectly reconstructed from digital samples, the theoretical maximum SNR (EQ 2-23) is the ratio of the full-scale analog input (RMS value) to the RMS quantization error (residual error). The ideal, theoretical minimum ADC noise is caused by quantization error only and results directly from the ADC's resolution (N bits):

$$SNR_{dB[MAX]} = 6.02_{dB} \times N + 1.76_{dB}$$

EQ 2-23

SINAD – Signal-to-Noise and Distortion

SINAD is the ratio of the rms amplitude to the mean value of the root-sum-square of the all other spectral components, including harmonics, but excluding DC. SINAD is a good indication of the overall dynamic performance of an ADC because it includes all components which make up noise and distortion.

Total Harmonic Distortion

THD measures the distortion content of a signal, and is specified in decibels relative to the carrier (dBc). THD is the ratio of the RMS sum of the selected harmonics of the input signal to the fundamental itself. Only harmonics within the Nyquist limit are included in the measurement.

TUE – Total Unadjusted Error

TUE is a comprehensive specification that includes linearity errors, gain error, and offset error. It is the worst-case deviation from the ideal device performance. TUE is a static specification (Figure 2-91).

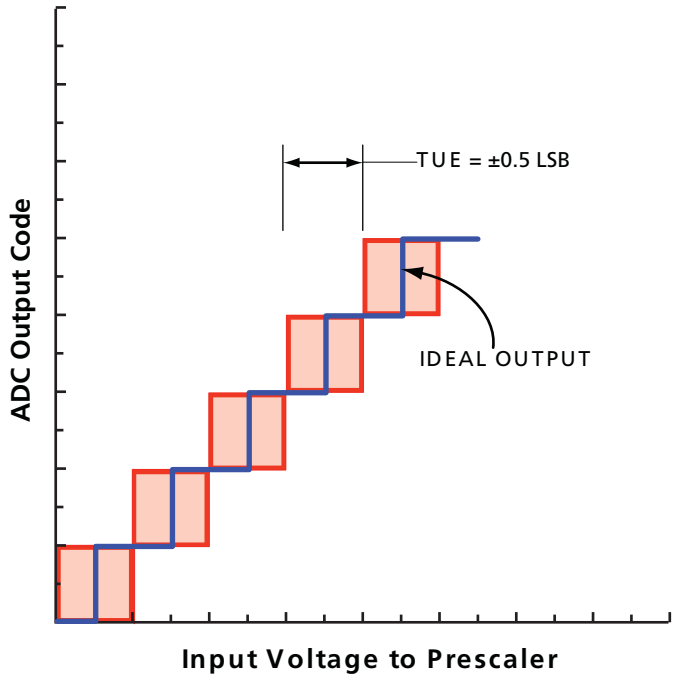


***Figure 2-91* • Total Unadjusted Error (TUE)**

Typical Performance Characteristics

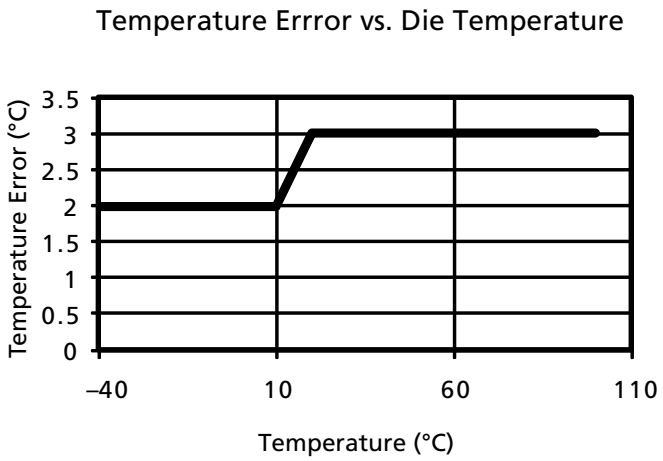


***Figure 2-92* • Temperature Error**

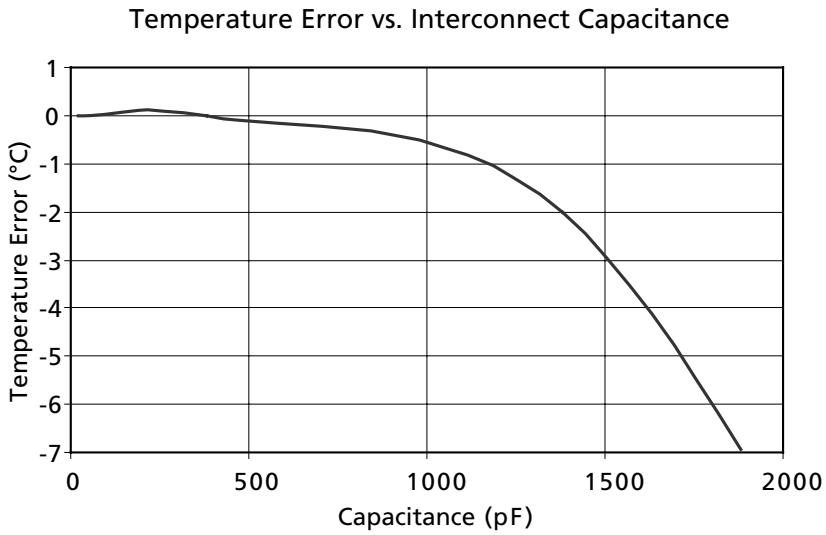


***Figure 2-93* • Effect of External Sensor Capacitance**

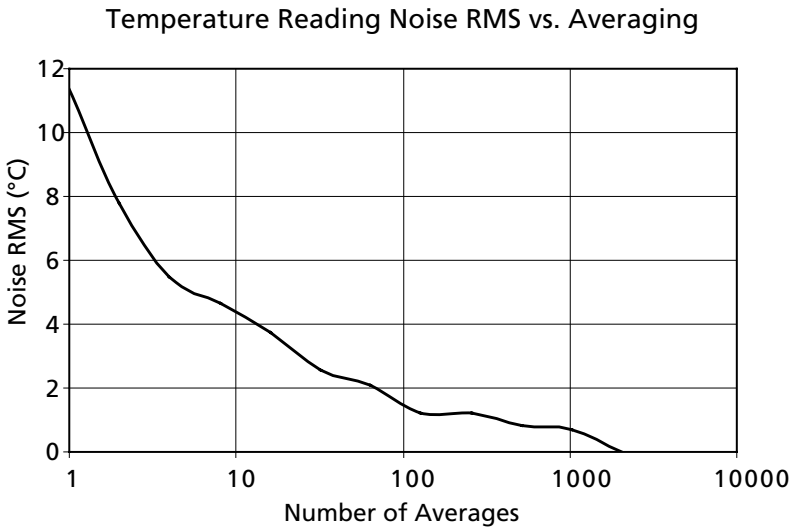


Figure 2-94 • **Temperature Reading Noise When Averaging is Used**

Analog System Characteristics

Table 2-46 • **Analog Channel Specifications**
Commercial Temperature Range Conditions, T_J = 85°C (unless noted otherwise),
Typical: V_{CC33A} = 3.3 V, V_{CC} = 1.5 V

Parameter	Description	Condition	Min.	Typ.	Max.	Units
Voltage Monitor Using Analog Pads AV, AC and AT (using prescaler)						
	Input Voltage (Prescaler)	Refer to Table 3-2 on page 3-3				
V_{INAP}	Uncalibrated Gain and Offset Errors	Refer to Table 2-48 on page 2-123				
	Calibrated Gain and Offset Errors	Refer to Table 2-49 on page 2-124				
	Bandwidth1				100	KHz
	Input Resistance	Refer to Table 3-3 on page 3-4				
	Scaling Factor	Prescaler modes (Table 2-54 on page 2-131)				
	Sample Time		10			µs
Current Monitor Using Analog Pads AV and AC						
V_{RSM}[1]	Maximum Differential Input Voltage				VAREF / 10	mV
	Resolution	Refer to "Current Monitor" section				
	Common Mode Range				– 10.5 to +12	V
CMRR	Common Mode Rejection Ratio	DC – 1 KHz		60		dB
		1 KHz - 10 KHz		50		dB
		> 10 KHz		30		dB
t_{CMSHI}	Strobe High time		ADC conv. time		200	µs
t_{CMSHI}	Strobe Low time		5			µs
t_{CMSHI}	Settling time		0.02			µs
	Accuracy	Input differential voltage > 50 mV			–2 –(0.05 x V_{RSM}) to +2 + (0.05 x V_{RSM})	mV

Notes:

1. *V_{RSM} is the maximum voltage drop across the current sense resistor.*
2. *Analog inputs used as digital inputs can tolerate the same voltage limits as the corresponding analog pad. There is no reliability concern on digital inputs as long as V_{IND} does not exceed these limits.*
3. *V_{IND} is limited to V_{CC33A} + 0.2 to allow reaching 10 MHz input frequency.*
4. *An averaging of 1,024 samples (LPF setting in Analog System Builder) is required and the maximum capacitance allowed across the AT pins is 500 pF.*
5. *The temperature offset is a fixed positive value.*
6. *The high current mode has a maximum power limit of 20 mW. Appropriate current limit resistors must be used, based on voltage on the pad.*
7. *When using SmartGen Analog System Builder, CalibIP is required to obtain 0 offset. For further details on CalibIP, refer to the Temperature, Voltage, and Current Calibration in Fusion FPGAs application note.*

Table 2-46 • Analog Channel Specifications (continued)
Commercial Temperature Range Conditions, T_J = 85°C (unless noted otherwise), Typical: V_{CC33A} = 3.3 V, V_{CC} = 1.5 V

Parameter	Description	Condition	Min.	Typ.	Max.	Units
Temperature Monitor Using Analog Pad AT						
External Temperature Monitor (external diode 2N3904, T_J = 25°C)[4]	Resolution	8-bit ADC		4		°C
		10-bit ADC		1		°C
		12-bit ADC		0.25		°C
	Offset[5]	AFS090, AFS250		5		°C
		AFS600, AFS1500 uncalibrated[7]		11		°C
		AFS600, AFS1500 calibrated[7]		0		°C
	Accuracy			±3	±5	°C
	External Sensor Source Current	High level, TMSTBx = 0		10		µA
		Low level, TMSTBx = 1		100		µA
	Max Capacitance on AT pad				1.3	nF
Internal Temperature Monitor	Resolution	8-bit ADC	4			°C
		10-bit ADC	1			°C
		12-bit ADC	0.25			°C
	Offset[5]	AFS090, AFS250	5			°C
		AFS600, AFS1500 uncalibrated[7]	11			°C
		AFS600, AFS1500 calibrated[7]	0			°C
	Accuracy			±3	±5	°C
t_{TMSHI}	Strobe High time		10		105	µs
t_{TMSLO}	Strobe Low time		5			µs
t_{TMSSET}	Settling time		5			µs

Notes:

1. *V_{RSM} is the maximum voltage drop across the current sense resistor.*
2. *Analog inputs used as digital inputs can tolerate the same voltage limits as the corresponding analog pad. There is no reliability concern on digital inputs as long as V_{IND} does not exceed these limits.*
3. *V_{IND} is limited to V_{CC33A} + 0.2 to allow reaching 10 MHz input frequency.*
4. *An averaging of 1,024 samples (LPF setting in Analog System Builder) is required and the maximum capacitance allowed across the AT pins is 500 pF.*
5. *The temperature offset is a fixed positive value.*
6. *The high current mode has a maximum power limit of 20 mW. Appropriate current limit resistors must be used, based on voltage on the pad.*
7. *When using SmartGen Analog System Builder, CalibIP is required to obtain 0 offset. For further details on CalibIP, refer to the Temperature, Voltage, and Current Calibration in Fusion FPGAs application note.*

Table 2-46 • **Analog Channel Specifications (continued)**
Commercial Temperature Range Conditions, T_J = 85°C (unless noted otherwise),
Typical: V_{CC33A} = 3.3 V, V_{CC} = 1.5 V

Parameter	Description	Condition	Min.	Typ.	Max.	Units
Digital Input using Analog Pads AV, AC and AT						
V_{IND}[2,3]	Input Voltage	Refer to Table 3-2 on page 3-3				
V_{HYSDIN}	Hysteresis			0.3		V
V_{IHDIN}	Input High			1.2		V
V_{ILDIN}	Input Low			0.9		V
V_{MPWDIN}	Minimum Pulse With		50			ns
F_{DIN}	Maximum Frequency				10	MHz
I_{STBDIN}	Input Leakage Current			2		µA
I_{DYNDIN}	Dynamic Current			20		µA
t_{INDIN}	Input Delay			10		ns
Gate Driver Output Using Analog Pad AG						
V_G	Voltage Range	Refer to Table 3-2 on page 3-3				
I_G	Output Current Drive	High Current Mode[6] at 1.0 V			±20	mA
		Low Current Mode: ±1 µA	0.8	1.0	1.3	µA
		Low Current Mode: ±3 µA	2.0	2.7	3.3	µA
		Low Current Mode: ± 10 µA	7.4	9.0	11.5	µA
		Low Current Mode: ± 30 µA	21.0	27.0	32.0	µA
I_{OFFG}	Maximum Off Current				100	nA
F_G	Maximum switching rate	High Current Mode[6] at 1.0 V, 1 kΩ resistive load		1.3		MHz
		Low Current Mode: ±1 µA, 3 MΩ resistive load		3		KHz
		Low Current Mode: ±3 µA, 1 MΩ resistive load		7		KHz
		Low Current Mode: ±10 µA, 300 kΩ resistive load		25		KHz
		Low Current Mode: ±30 µA, 105 kΩ resistive load		78		KHz

Notes:

1. *V_{RSM} is the maximum voltage drop across the current sense resistor.*
2. *Analog inputs used as digital inputs can tolerate the same voltage limits as the corresponding analog pad. There is no reliability concern on digital inputs as long as V_{IND} does not exceed these limits.*
3. *V_{IND} is limited to V_{CC33A} + 0.2 to allow reaching 10 MHz input frequency.*
4. *An averaging of 1,024 samples (LPF setting in Analog System Builder) is required and the maximum capacitance allowed across the AT pins is 500 pF.*
5. *The temperature offset is a fixed positive value.*
6. *The high current mode has a maximum power limit of 20 mW. Appropriate current limit resistors must be used, based on voltage on the pad.*
7. *When using SmartGen Analog System Builder, CalibIP is required to obtain 0 offset. For further details on CalibIP, refer to the Temperature, Voltage, and Current Calibration in Fusion FPGAs application note.*

Table 2-47 • ADC Characteristics in Direct Input Mode
Commercial Temperature Range Conditions, T_J = 85°C (unless noted otherwise),
Typical: V_{CC33A} = 3.3 V, V_{CC} = 1.5 V

Parameter	Description	Condition	Min.	Typ.	Max.	Units
Direct Input using Analog Pad AV, AC, AT						
V_{INADC}	Input Voltage (Direct Input)	Refer to Table 3-2 on page 3-3				
C_{INADC}	Input Capacitance	Channel not selected		7		pF
		Channel selected but not sampling		8		pF
		Channel selected and sampling		18		pF
Z_{INADC}	Input Impedance	8-bit mode		2		kΩ
		10-bit mode		2		kΩ
		12-bit mode		2		kΩ
Analog Reference Voltage VAREF						
VAREF	Accuracy	T_J = 25°C	2.537	2.56	2.583	V
	Temperature Drift of Internal Reference			65		ppm / °C
	External Reference		2.527		V_{CC33A} + 0.05	V
ADC Accuracy (using external reference) [1,2]						
DC Accuracy						
TUE	Total Unadjusted Error	8-bit mode		0.29		LSB
		10-bit mode		0.72		LSB
		12-bit mode		1.8		LSB
INL	Integral Non-Linearity	8-bit mode		0.20	0.25	LSB
		10-bit mode		0.32	0.43	LSB
		12-bit mode		1.71	1.80	LSB
DNL	Differential Non-Linearity (no missing code)	8-bit mode		0.20	0.24	LSB
		10-bit mode		0.60	0.65	LSB
		12-bit mode		2.40	2.48	LSB
	Offset Error	8-bit mode		0.01	0.17	LSB
		10-bit mode		0.05	0.20	LSB
		12-bit mode		0.20	0.40	LSB
	Gain Error	8-bit mode		0.0004	0.003	LSB
		10-bit mode		0.002	0.011	LSB
		12-bit mode		0.007	0.044	LSB
	Gain Error (with internal reference)	All modes		2		% FSR

Notes:

1. *Accuracy of the external reference is 2.56 V ± 4.6 mV.*
2. *Data is based on characterization.*
3. *The sample rate is time-shared among active analog inputs.*

Table 2-47 • **ADC Characteristics in Direct Input Mode (continued)**
Commercial Temperature Range Conditions, T_J = 85°C (unless noted otherwise),
Typical: V_{CC33A} = 3.3 V, V_{CC} = 1.5 V

Parameter	Description	Condition	Min.	Typ.	Max.	Units
Dynamic Performance						
SNR	Signal-to-Noise Ratio	8-bit mode	48.0	49.5		dB
		10-bit mode	58.0	60.0		dB
		12-bit mode	62.9	64.5		dB
SINAD	Signal-to-Noise Distortion	8-bit mode	47.6	49.5		dB
		10-bit mode	57.4	59.8		dB
		12-bit mode	62.0	64.2		dB
THD	Total Harmonic Distortion	8-bit mode		–74.4	–63.0	dBc
		10-bit mode		–78.3	–63.0	dBc
		12-bit mode		–77.9	–64.4	dBc
ENOB	Effective Number of Bits	8-bit mode	7.6	7.9		bits
		10-bit mode	9.5	9.6		bits
		12-bit mode	10.0	10.4		bits
Conversion Rate						
	Conversion Time	8-bit mode	1.7			µs
		10-bit mode	1.8			µs
		12-bit mode	2			µs
	Sample Rate	8-bit mode			600	Ksps
		10-bit mode			550	Ksps
		12-bit mode			500	Ksps

Notes:

1. *Accuracy of the external reference is 2.56 V ± 4.6 mV.*
2. *Data is based on characterization.*
3. *The sample rate is time-shared among active analog inputs.*

Table 2-48 • **Uncalibrated Analog Channel Accuracy*
Worst-Case Industrial Conditions, T_J = 85°C**

		Total Channel Error (LSB)			Channel Input Offset Error (LSB)			Channel Input Offset Error (mV)			Channel Gain Error (%FSR)		
Analog Pad	**Prescaler Range (V)**	**Neg. Max.**	**Med.**	**Pos. Max.**	**Neg Max**	**Med.**	**Pos. Max.**	**Neg. Max.**	**Med.**	**Pos. Max.**	**Min.**	**Typ.**	**Max.**
Positive Range		**ADC in 10-Bit Mode**											
AV, AC	16	–22	–2	12	–11	–2	14	–169	–32	224	3	0	–3
	8	–40	–5	17	–11	–5	21	–87	–40	166	2	0	–4
	4	–45	–9	24	–16	–11	36	–63	–43	144	2	0	–4
	2	–70	–19	33	–33	–20	66	–66	–39	131	2	0	–4
	1	–25	–7	5	–11	–3	26	–11	–3	26	3	–1	–3
	0.5	–41	–12	8	–12	–7	38	–6	–4	19	3	–1	–3
	0.25	–53	–14	19	–20	–14	40	–5	–3	10	5	0	–4
	0.125	–89	–29	24	–40	–28	88	–5	–4	11	7	0	–5
AT	16	–3	9	15	–4	0	4	–64	5	64	1	0	–1
	4	–10	2	15	–11	–2	11	–44	–8	44	1	0	–1
Negative Range		**ADC in 10-Bit Mode**											
AV, AC	16	–35	–10	9	–24	–6	9	–383	–96	148	5	–1	–6
	8	–65	–19	12	–34	–12	9	–268	–99	75	5	–1	–5
	4	–86	–28	21	–64	–24	19	–254	–96	76	5	–1	–6
	2	–136	–53	37	–115	–42	39	–230	–83	78	6	–2	–7
	1	–98	–35	8	–39	–8	15	–39	–8	15	10	–3	–10
	0.5	–121	–46	7	–54	–14	18	–27	–7	9	10	–4	–11
	0.25	–149	–49	19	–72	–16	40	–18	–4	10	14	–4	–12
	0.125	–188	–67	38	–112	–27	56	–14	–3	7	16	–5	–14

*Note: *Channel Accuracy includes prescaler and ADC accuracies. For 12-bit mode, multiply the LSB count by 4. For 8-bit mode, divide the LSB count by 4. Gain remains the same.*

Table 2-49 • **Calibrated Analog Channel Accuracy [1,2,3]**
Worst-Case Industrial Conditions, T_J = 85°C

		Condition	Total Channel Error (LSB)		
Analog Pad	**Prescaler Range (V)**	**Input Voltage[4] (V)**	**Negative Max.**	**Median**	**Positive Max.**
Positive Range			**ADC in 10-Bit Mode**		
AV, AC	16	0.300 to 12.0	–6	1	6
	8	0.250 to 8.00	–6	0	6
	4	0.200 to 4.00	–7	–1	7
	2	0.150 to 2.00	–7	0	7
	1	0.050 to 1.00	–6	–1	6
AT	16	0.300 to 16.0	–5	0	5
	4	0.100 to 4.00	–7	–1	7
Negative Range			**ADC in 10-Bit Mode**		
AV, AC	16	–0.400 to –10.5	–7	1	9
	8	–0.350 to –8.00	–7	–1	7
	4	–0.300 to –4.00	–7	–2	9
	2	–0.250 to –2.00	–7	–2	7
	1	–0.050 to –1.00	–16	–1	20

Notes:

1. *Channel Accuracy includes prescaler and ADC accuracies. For 12-bit mode, multiply the LSB count by 4. For 8-bit mode, divide the LSB count by 4. Overall accuracy remains the same.*
2. *Requires enabling Analog Calibration using SmartGen Analog System Builder. For further details, refer to the Temperature, Voltage, and Current Calibration in Fusion FPGAs application note.*
3. *Calibrated with two-point calibration methodology, using 20% and 80% full-scale points.*
4. *The lower limit of the input voltage is determined by the prescaler input offset.*

Table 2-50 • Analog Channel Accuracy: Monitoring Standard Positive Voltages Typical Conditions, T_A = 25°C

Input Voltage (V)	Calibrated Typical Error per Positive Prescaler Setting [1] (%FSR)							Direct ADC [2,3] (%FSR)
	16 V (AT)	16 V (12 V) (AV/AC)	8 V (AV/AC)	4 V (AT)	4 V (AV/AC)	2 V (AV/AC)	1 V (AV/AC)	VAREF = 2.56 V
15	1							
14	1							
12	1	1						
5	2	2	1					
3.3	2	2	1	1	1			
2.5	3	2	1	1	1			1
1.8	4	4	1	1	1	1		1
1.5	5	5	2	2	2	1		1
1.2	7	6	2	2	2	1		1
0.9	9	9	4	3	3	1	1	1

Notes:

1. *Requires enabling Analog Calibration using SmartGen Analog System Builder. For further details, refer to the Temperature, Voltage, and Current Calibration in Fusion FPGAs application note.*
2. *Direct ADC mode using an external VAREF of 2.56V±4.6mV, without Analog Calibration macro.*
3. *For input greater than 2.56 V, the ADC output will saturate. A higher VAREF or prescaler usage is recommended.*

Examples

Calculating Accuracy for an Uncalibrated Analog Channel

Formula

For a given prescaler range, EQ 2-24 gives the output voltage.

$$\text{Output Voltage} = (\text{Channel Output Offset in V}) + (\text{Input Voltage} \times \text{Channel Gain}) \quad \text{EQ 2-24}$$

where

Channel Output offset in V = Channel Output offset in LSBs x Equivalent voltage per LSB

Channel Gain Factor = 1+ (% Channel Gain / 100)

Example

Input Voltage = 5 V

Chosen Prescaler range = 8 V range

Refer to Table 2-48 on page 2-123.

Max. Output Voltage = (Max Positive output offset) + (Input Voltage x Max Gain Factor)

Max. Positive output offset = (8 LSB) x (8mV per LSB in 10-bit mode)

Max. Positive output offset = 64 mV

Max. Gain = 1 + (2/100)

Max. Gain = 1.02

Max. Output Voltage = (64 mV) + (5 V x 1.02)

Max. Output Voltage = **5.164 V**

Similarly,

Min. Output Voltage = (Min. Negative output offset) + (Input Voltage x Min. Gain)

= (–136 mV) + (5 V x 0.98) = **4.764 V**

Calculating Accuracy for a Calibrated Analog Channel

Formula

For a given prescaler range, EQ 2-25 gives the output voltage.

$$\textit{Output Voltage} = \textit{Channel TUE in V} + \textit{Input Voltage} \qquad \textit{EQ 2-25}$$

where

Channel TUE in V = Channel TUE in LSBs x Equivalent voltage per LSB

Example

Input Voltage = 5 V

Chosen Prescaler range = 8 V range

Refer to Table 2-49 on page 2-124.

Max. Output Voltage = Max. Channel TUE in V + Input Voltage

Max. Channel TUE in V = (6 LSB) × (8 mV per LSB in 10-bit mode) = 48 mV

Max. Output Voltage = 48 mV + 5 V = **5.048 V**

Similarly,

Min Output Voltage = Min Channel TUE in V + Input Voltage = (-48 mV) + 5 V = **4.952 V**

Calculating LSBs from a Given Error Budget

Formula

For a given prescaler range,

LSB count = ± (Input Voltage × Required % error) / (Equivalent voltage per LSB)

Example

Input Voltage = 5 V

Required error margin= 1%

Refer to Table 2-49 on page 2-124.

Equivalent voltage per LSB = 16 mV for a 16V prescaler, with ADC in 10-bit mode

LSB Count = ± (5.0 V × 1%) / (0.016)

LSB Count = **± 3.125**

Equivalent voltage per LSB = 8 mV for an 8 V prescaler, with ADC in 10-bit mode

LSB Count = ± (5.0 V × 1%) / (0.008)

LSB Count = **± 6.25**

The 8 V prescaler satisfies the calculated LSB count accuracy requirement (see Table 2-49 on page 2-124).

Analog Configuration MUX

The ACM is the interface between the FPGA, the Analog Block configurations, and the real-time counter. Actel Libero IDE will generate IP that will load and configure the Analog Block via the ACM. However, users are not limited to using the Libero IDE IP. This section provides a detailed description of the ACM's register map, truth tables for proper configuration of the Analog Block and RTC, as well as timing waveforms so users can access and control the ACM directly from their designs.

The Analog Block contains four 8-bit latches per Analog Quad that are initialized through the ACM. These latches act as configuration bits for Analog Quads. The ACM block runs from the core voltage supply (1.5 V).

Access to the ACM is achieved via 8-bit address and data busses with enables. The pin list is provided in Table 2-36 on page 2-82. The ACM clock speed is limited to a maximum of 10 MHz, more than sufficient to handle the low-bandwidth requirements of configuring the Analog Block and the RTC (sub-block of the Analog Block).

Table 2-51 decodes the ACM address space and maps it to the corresponding Analog Quad and configuration byte for that quad.

Table 2-51 • **ACM Address Decode Table for Analog Quad**

ACMADDR [7:0] in Decimal	Name	Description	Associated Peripheral
0	–	–	Analog Quad
1	AQ0	Byte 0	Analog Quad
2	AQ0	Byte 1	Analog Quad
3	AQ0	Byte 2	Analog Quad
4	AQ0	Byte 3	Analog Quad
5	AQ1	Byte 0	Analog Quad
. . .	. . .	. . .	Analog Quad
36	AQ8	Byte 3	Analog Quad
37	AQ9	Byte 0	Analog Quad
38	AQ9	Byte 1	Analog Quad
39	AQ9	Byte 2	Analog Quad
40	AQ9	Byte 3	Analog Quad
41		Undefined	Analog Quad
. . .	. . .	Undefined	Analog Quad
63		Undefined	RTC
64	COUNTER0	Counter bits 7:0	RTC
65	COUNTER1	Counter bits 15:8	RTC
66	COUNTER2	Counter bits 23:16	RTC
67	COUNTER3	Counter bits 31:24	RTC
68	COUNTER4	Counter bits 39:32	RTC
72	MATCHREG0	Match register bits 7:0	RTC

Table 2-51 • **ACM Address Decode Table for Analog Quad (continued)**

ACMADDR [7:0] in Decimal	Name	Description	Associated Peripheral
73	MATCHREG1	Match register bits 15:8	RTC
74	MATCHREG2	Match register bits 23:16	RTC
75	MATCHREG3	Match register bits 31:24	RTC
76	MATCHREG4	Match register bits 39:32	RTC
80	MATCHBITS0	Individual match bits 7:0	RTC
81	MATCHBITS1	Individual match bits 15:8	RTC
82	MATCHBITS2	Individual match bits 23:16	RTC
83	MATCHBITS3	Individual match bits 31:24	RTC
84	MATCHBITS4	Individual match bits 39:32	RTC
88	CTRL_STAT	Control (write) / Status (read) register bits 7:0	RTC

Note: ACMADDR bytes 1 to 40 pertain to the Analog Quads; bytes 64 to 89 pertain to the RTC.

ACM Characteristics[1]

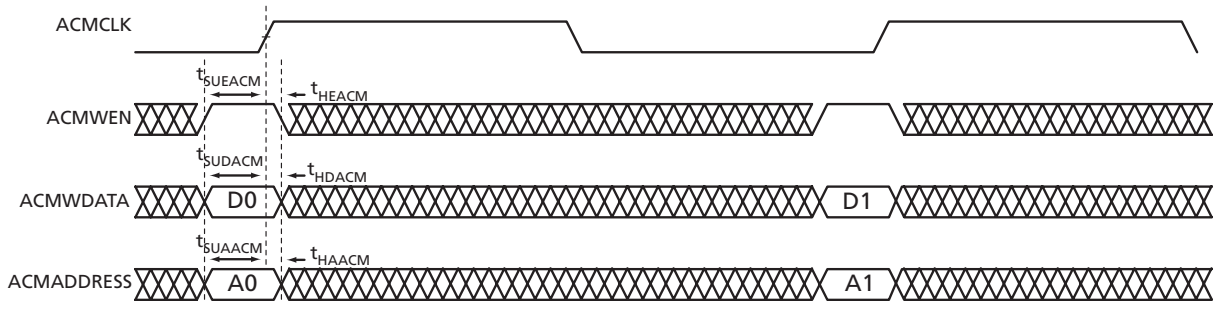


Figure 2-95 • **ACM Write Waveform**

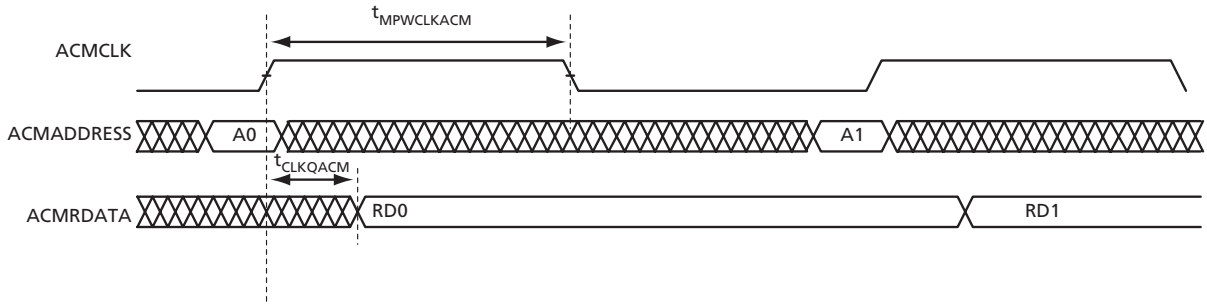


Figure 2-96 • **ACM Read Waveform**

1. *When addressing the RTC addresses (i.e., ACMADDR 64 to 89), there is no timing generator, and the rc_osc, byte_en, and aq_wen signals have no impact.*

Timing Characteristics

Table 2-52 • Analog Configuration Multiplexer (ACM) Timing
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
$t_{CLKQACM}$	Clock-to-Q of the ACM	19.73	22.48	26.42	ns
t_{SUDACM}	Data Setup time for the ACM	4.39	5.00	5.88	ns
t_{HDACM}	Data Hold time for the ACM	0.00	0.00	0.00	ns
t_{SUAACM}	Address Setup time for the ACM	4.73	5.38	6.33	ns
t_{HAACM}	Address Hold time for the ACM	0.00	0.00	0.00	ns
t_{SUEACM}	Enable Setup time for the ACM	3.93	4.48	5.27	ns
t_{HEACM}	Enable Hold time for the ACM	0.00	0.00	0.00	ns
$t_{MPWARACM}$	Asynchronous Reset Minimum Pulse Width for the ACM	10.00	10.00	10.00	ns
$t_{REMARACM}$	Asynchronous Reset Removal time for the ACM	12.98	14.79	17.38	ns
$t_{RECARACM}$	Asynchronous Reset Recovery time for the ACM	12.98	14.79	17.38	ns
$t_{MPWCLKACM}$	Clock Minimum Pulse Width for the ACM	45.00	45.00	45.00	ns
$t_{FMAXCLKACM}$	lock Maximum Frequency for the ACM	10.00	10.00	10.00	MHz

Analog Quad ACM Description

Table 2-53 maps out the ACM space associated with configuration of the Analog Quads within the Analog Block. Table 2-53 shows the byte assignment within each quad and the function of each bit within each byte. Subsequent tables will explain each bit setting and how it corresponds to a particular configuration. After 3.3 V and 1.5 V are applied to Fusion, Analog Quad configuration registers are loaded with default settings until the initialization and configuration state machine changes them to user-defined settings.

Table 2-53 • **Analog Quad ACM Byte Assignment**

Byte	Bit	Signal (Bx)	Function	Default Setting
Byte 0 (AV)	0	B0[0]	Scaling factor control – prescaler	Highest voltage range
	1	B0[1]		
	2	B0[2]		
	3	B0[3]	Analog MUX select	Prescaler
	4	B0[4]	Current monitor switch	Off
	5	B0[5]	Direct analog input switch	Off
	6	B0[6]	Selects V-pad polarity	Positive
	7	B0[7]	Prescaler op amp mode	Power-down
Byte 1 (AC)	0	B1[0]	Scaling factor control – prescaler	Highest voltage range
	1	B1[1]		
	2	B1[2]		
	3	B1[3]	Analog MUX select	Prescaler
	4	B1[4]		
	5	B1[5]	Direct analog input switch	Off
	6	B1[6]	Selects C-pad polarity	Positive
	7	B1[7]	Prescaler op amp mode	Power-down
Byte 2 (AG)	0	B2[0]	Internal chip temperature monitor	Off
	1	B2[1]	Spare	–
	2	B2[2]	Current drive control	Lowest current
	3	B2[3]		
	4	B2[4]	Spare	–
	5	B2[5]	Spare	–
	6	B2[6]	Selects G-pad polarity	Positive
	7	B2[7]	Selects low/high drive	Low drive
Byte 3 (AT)	0	B3[0]	Scaling factor control – prescaler	Highest voltage range
	1	B3[1]		
	2	B3[2]		
	3	B3[3]	Analog MUX select	Prescaler
	4	B3[4]		
	5	B3[5]	Direct analog input switch	Off
	6	B3[6]	–	–
	7	B3[7]	Prescaler op amp mode	Power-down

Table 2-54 details the settings available to control the prescaler values of the AV, AC, and AT pins. Note that the AT pin has a reduced number of available prescaler values.

Table 2-54 • **Prescaler Control Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3)**

Control Lines Bx[2:0]	Scaling Factor, Pad to ADC Input	LSB for an 8-Bit Conversion[2] (mV)	LSB for a 10-Bit Conversion[2] (mV)	LSB for a 12-Bit Conversion[2] (mV)	Full-Scale Voltage	Range Name
000 [1]	0.15625	64	16	4	16.368 V	16 V
001	0.3125	32	8	2	8.184 V	8 V
010 [1]	0.625	16	4	1	4.092 V	4 V
011	1.25	8	2	0.5	2.046 V	2 V
100	2.5	4	1	0.25	1.023 V	1 V
101	5.0	2	0.5	0.125	0.5115 V	0.5 V
110	10.0	1	0.25	0.0625	0.25575 V	0.25 V
111	20.0	0.5	0.125	0.03125	0.127875 V	0.125 V

Notes:

1. *These are the only valid ranges for the Temperature Monitor Block Prescaler.*
2. *LSB voltage equivalences assume VAREF = 2.56 V.*

Table 2-55 details the settings available to control the MUX within each of the AV, AC, and AT circuits. This MUX determines whether the signal routed to the ADC is the direct analog input, prescaled signal, or output of either the Current Monitor Block or the Temperature Monitor Block.

Table 2-55 • **Analog Multiplexer Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3)**

Control Lines Bx[4]	Control Lines Bx[3]	ADC Connected To
0	0	Prescaler
0	1	Direct input
1	0	Current amplifier temperature monitor
1	1	Not valid

Table 2-56 details the settings available to control the Direct Analog Input switch for the AV, AC, and AT pins.

Table 2-56 • **Direct Analog Input Switch Control Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3)**

Control Lines Bx[5]	Direct Input Switch
0	Off
1	On

Table 2-57 details the settings available to control the polarity of the signals coming to the AV, AC, and AT pins. Note that the only valid setting for the AT pin is logic 0 to support positive voltages.

Table 2-57 • **Voltage Polarity Control Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3)***

Control Lines Bx[6]	Input Signal Polarity
0	Positive
1	Negative

*Note: *The B3[6] signal for the AT pad should be kept at logic 0 to accept only positive voltages.*

Table 2-58 details the settings available to either power down or enable the prescaler associated with the analog inputs AV, AC, and AT.

Table 2-58 • **Prescaler Op Amp Power-Down Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3)**

Control Lines Bx[7]	Prescaler Op Amp
0	Power-down
1	Operational

Table 2-59 details the settings available to enable the Current Monitor Block associated with the AC pin.

Table 2-59 • **Current Monitor Input Switch Control Truth Table—AV (x = 0)**

Control Lines B0[4]	Current Monitor Input Switch
0	Off
1	On

Table 2-60 details the settings available to configure the drive strength of the gate drive when not in high-drive mode.

Table 2-60 • **Low-Drive Gate Driver Current Truth Table (AG)**

Control Lines B2[3]	Control Lines B2[2]	Current (µA)
0	0	1
0	1	3
1	0	10
1	1	30

Table 2-61 details the settings available to set the polarity of the gate driver (either p-channel- or n-channel-type devices).

Table 2-61 • **Gate Driver Polarity Truth Table (AG)**

Control Lines B2[6]	Gate Driver Polarity
0	Positive
1	Negative

Table 2-62 details the settings available to turn on the Gate Driver and set whether high-drive mode is on or off.

Table 2-62 • **Gate Driver Control Truth Table (AG)**

Control Lines B2[7]	GDON	Gate Driver
0	0	Off
0	1	Low drive on
1	0	Off
1	1	High drive on

Table 2-63 details the settings available to turn on and off the chip internal temperature monitor.

Table 2-63 • **Internal Temperature Monitor Control Truth Table**

Control Lines B2[0]	PDTMB	Chip Internal Temperature Monitor
0	0	Off
1	1	On

User I/Os

Introduction

Fusion devices feature a flexible I/O structure, supporting a range of mixed voltages (1.5 V, 1.8 V, 2.5 V, and 3.3 V) through a bank-selectable voltage. Table 2-65, Table 2-66, Table 2-67, and Table 2-68 on page 2-136 show the voltages and the compatible I/O standards. I/Os provide programmable slew rates, drive strengths, weak pull-up, and weak pull-down circuits. 3.3 V PCI and 3.3 V PCI-X are 5 V–tolerant. See the "5 V Input Tolerance" section on page 2-145 for possible implementations of 5 V tolerance.

All I/Os are in a known state during power-up, and any power-up sequence is allowed without current impact. Refer to the "I/O Power-Up and Supply Voltage Thresholds for Power-On Reset (Commercial and Industrial)" section on page 3-5 for more information. In low power standby or sleep mode (V_{CC} is OFF, V_{CC33A} is ON, V_{CCI} is ON) or when the resource is not used, digital inputs are tristated, digital outputs are tristated, and digital bibufs (input/output) are tristated.

I/O Tile

The Fusion I/O tile provides a flexible, programmable structure for implementing a large number of I/O standards. In addition, the registers available in the I/O tile in selected I/O banks can be used to support high-performance register inputs and outputs, with register enable if desired (Figure 2-97 on page 2-134). The registers can also be used to support the JESD-79C DDR standard within the I/O structure (see the "Double Data Rate (DDR) Support" section on page 2-140 for more information).

As depicted in Figure 2-98 on page 2-139, all I/O registers share one CLR port. The output register and output enable register share one CLK port. Refer to the "I/O Registers" section on page 2-139 for more information.

I/O Banks and I/O Standards Compatibility

The digital I/Os are grouped into I/O voltage banks. There are three digital I/O banks on the AFS090 and AFS250 devices and four digital I/O banks on the AFS600 and AFS1500 devices. Figure 2-111 on page 2-160 and Figure 2-112 on page 2-160 show the bank configuration by device. The north side of the I/O in the AFS600 and AFS1500 devices comprises two banks of Actel Pro I/Os. The Actel Pro I/Os support a wide number of voltage-referenced I/O standards in addition to the multitude of single-ended and differential I/O standards common throughout all Actel digital I/Os. Each I/O voltage bank has dedicated I/O supply and ground voltages (V_{CCI}/GNDQ for input buffers and V_{CCI}/GND for output buffers). Because of these dedicated supplies, only I/Os with compatible standards can be assigned to the same I/O voltage bank. Table 2-66 and Table 2-67 on page 2-135 show the required voltage compatibility values for each of these voltages.

For more information about I/O and global assignments to I/O banks, refer to the specific pin table of the device in the "Package Pin Assignments" on page 4-1 and the "User I/O Naming Convention" section on page 2-159.

Each Pro I/O bank is divided into minibanks. Any user I/O in a V_{REF} minibank (a minibank is the region of scope of a V_{REF} pin) can be configured as a V_{REF} pin (Figure 2-97 on page 2-134). Only one V_{REF} pin is needed to control the entire V_{REF} minibank. The location and scope of the V_{REF} minibanks can be determined by the I/O name. For details, see the "User I/O Naming Convention" section on page 2-159.

Table 2-67 on page 2-135 shows the I/O standards supported by Fusion devices and the corresponding voltage levels.

I/O standards are compatible if the following are true:

- Their V_{CCI} values are identical.
- If both of the standards need a V_{REF}, their V_{REF} values must be identical (Pro I/O only).

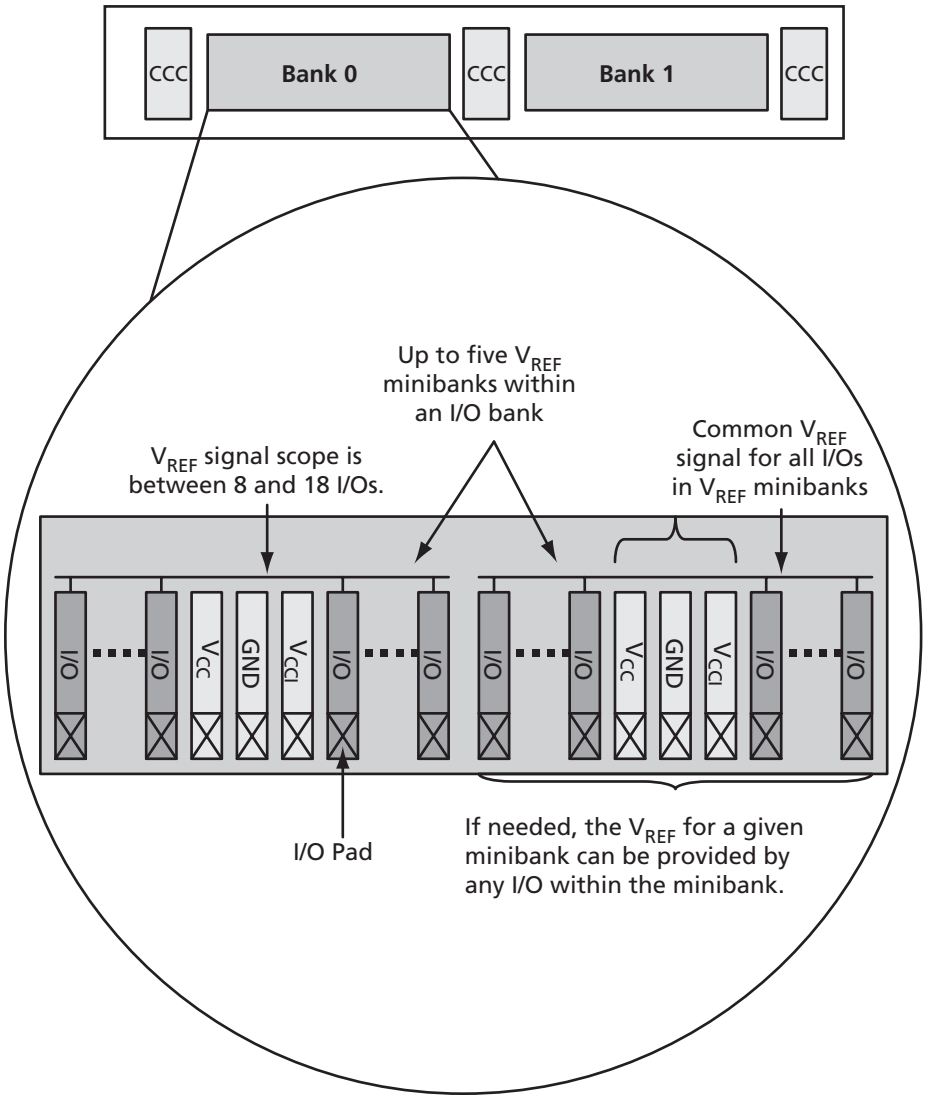


Figure 2-97 • **Fusion Pro I/O Bank Detail Showing V_{REF} Minibanks (north side ofAFS600 and AFS1500)**

Table 2-64 • **I/O Standards Supported by Bank Type**

I/O Bank	Single-Ended I/O Standards	Differential I/O Standards	Voltage-Referenced	Hot-Swap
Standard I/O	LVTTL/LVCMOS 3.3 V, LVCMOS 2.5 V / 1.8 V / 1.5 V, LVCMOS 2.5/5.0 V	–	–	Yes
Advanced I/O	LVTTL/LVCMOS 3.3 V, LVCMOS 2.5 V / 1.8 V / 1.5 V, LVCMOS 2.5/5.0 V, 3.3 V PCI / 3.3 V PCI-X	LVPECL and LVDS	–	–
Pro I/O	LVTTL/LVCMOS 3.3 V, LVCMOS 2.5 V / 1.8 V / 1.5 V, LVCMOS 2.5/5.0 V, 3.3 V PCI / 3.3 V PCI-X	LVPECL and LVDS	GTL+ 2.5 V / 3.3 V, GTL 2.5 V / 3.3 V, HSTL Class I and II, SSTL2 Class I and II, SSTL3 Class I and II	Yes

Table 2-65 • **I/O Bank Support by Device**

I/O Bank	AFS090	AFS250	AFS600	AFS1500
Standard I/O	N	N	–	–
Advanced I/O	E, W	E, W	E, W	E, W
Pro I/O	–	–	N	N
Analog Quad	S	S	S	S

Note: E = East side of the device
W = West side of the device
N = North side of the device
S = South side of the device

Table 2-66 • **Fusion V_{CCI} Voltages and Compatible Standards**

V_{CCI} (typical)	Compatible Standards
3.3 V	LVTTL/LVCMOS 3.3, PCI 3.3, SSTL3 (Class I and II),* GTL+ 3.3, GTL 3.3,* LVPECL
2.5 V	LVCMOS 2.5, LVCMOS 2.5/5.0, SSTL2 (Class I and II),* GTL+ 2.5,* GTL 2.5,* LVDS, BLVDS, M-LVDS
1.8 V	LVCMOS 1.8
1.5 V	LVCMOS 1.5, HSTL (Class I),* HSTL (Class II)*

*Note: *I/O standard supported by Pro I/O banks.*

Table 2-67 • **Fusion V_{REF} Voltages and Compatible Standards***

V_{REF} (typical)	Compatible Standards
1.5 V	SSTL3 (Class I and II)
1.25 V	SSTL2 (Class I and II)
1.0 V	GTL+ 2.5, GTL+ 3.3
0.8 V	GTL 2.5, GTL 3.3
0.75 V	HSTL (Class I), HSTL (Class II)

*Note: *I/O standards supported by Pro I/O banks.*

Table 2-68 • **Fusion Standard and Advanced I/O Features**

I/O Bank Voltage (typical)	Minibank Voltage (typical)	LVTTL/LVCMOS 3.3 V	LVCMOS 2.5 V	LVCMOS 1.8 V	LVCMOS 1.5 V	3.3 V PCI / PCI-X	GTL + (3.3 V)	GTL + (2.5 V)	GTL (3.3 V)	GTL (2.5 V)	HSTL Class I and II (1.5 V)	SSTL2 Class I and II (2.5 V)	SSTL3 Class I and II (3.3 V)	LVDS (2.5 V ± 5%)	LVPECL (3.3 V)
3.3 V	–														
	0.80 V														
	1.00 V														
	1.50 V														
2.5 V	–														
	0.80 V														
	1.00 V														
	1.25 V														
1.8 V	–														
1.5 V	–														
	0.75 V														

Note: White box: Allowable I/O standard combinations
Gray box: Illegal I/O standard combinations

Features Supported on Pro I/Os

Table 2-69 lists all features supported by transmitter/receiver for single-ended and differential I/Os.

Table 2-69 • **Fusion Pro I/O Features**

Feature	Description
Single-ended and voltage-referenced transmitter features	• Hot insertion in every mode except PCI or 5 V input tolerant (these modes use clamp diodes and do not allow hot insertion) • Activation of hot insertion (disabling the clamp diode) is selectable by I/Os. • Weak pull-up and pull-down • Two slew rates • Skew between output buffer enable/disable time: 2 ns delay (rising edge) and 0 ns delay (falling edge); see "Selectable Skew between Output Buffer Enable/Disable Time" on page 2-150 for more information • Five drive strengths • 5 V–tolerant receiver ("5 V Input Tolerance" section on page 2-145) • LVTTL/LVCMOS 3.3 V outputs compatible with 5 V TTL inputs ("5 V Output Tolerance" section on page 2-149) • High performance (Table 2-73 on page 2-144)
Single-ended receiver features	• Schmitt trigger option • ESD protection • Programmable delay: 0 ns if bypassed, 0.625 ns with '000' setting, 6.575 ns with '111' setting, 0.85-ns intermediate delay increments (at 25°C, 1.5 V) • High performance (Table 2-73 on page 2-144) • Separate ground planes, GND/GNDQ, for input buffers only to avoid output-induced noise in the input circuitry
Voltage-referenced differential receiver features	• Programmable Delay: 0 ns if bypassed, 0.625 ns with '000' setting, 6.575 ns with '111' setting, 0.85-ns intermediate delay increments (at 25°C, 1.5 V) • High performance (Table 2-73 on page 2-144) • Separate ground planes, GND/GNDQ, for input buffers only to avoid output-induced noise in the input circuitry
CMOS-style LVDS, BLVDS, M-LVDS, or LVPECL transmitter	• Two I/Os and external resistors are used to provide a CMOS-style LVDS, BLVDS, M-LVDS, or LVPECL transmitter solution. • Activation of hot insertion (disabling the clamp diode) is selectable by I/Os. • Weak pull-up and pull-down • Fast slew rate
LVDS/LVPECL differential receiver features	• ESD protection • High performance (Table 2-73 on page 2-144) • Programmable delay: 0.625 ns with '000' setting, 6.575 ns with '111' setting, 0.85-ns intermediate delay increments (at 25°C, 1.5 V) • Separate input buffer ground and power planes to avoid output-induced noise in the input circuitry

Table 2-70 • **Maximum I/O Frequency for Single-Ended, Voltage-Referenced, and Differential I/Os; All I/O Bank Types (maximum drive strength and high slew selected)**

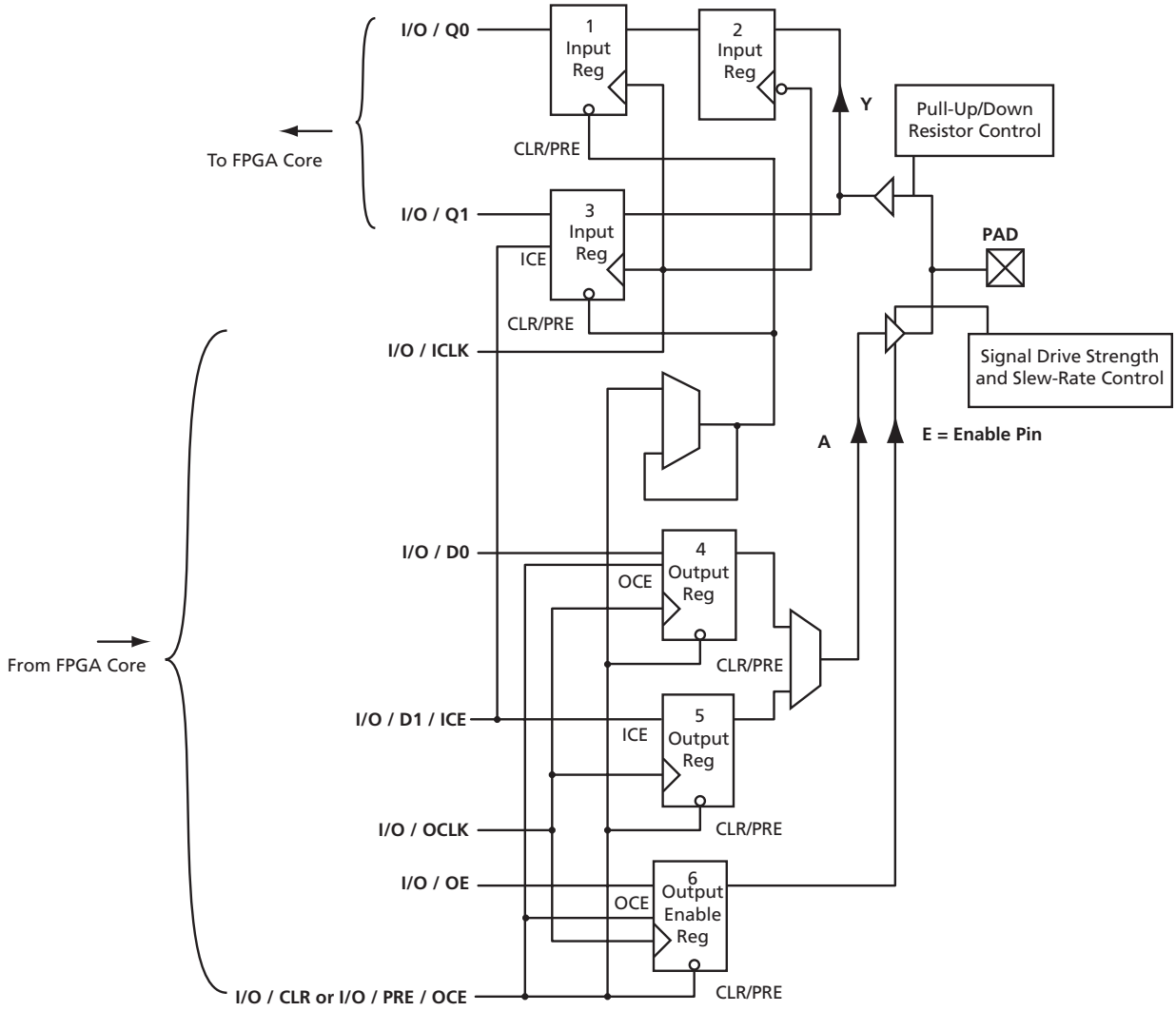
Specification	Performance Up To
LVTTL/LVCMOS 3.3 V	200 MHz
LVCMOS 2.5 V	250 MHz
LVCMOS 1.8 V	200 MHz
LVCMOS 1.5 V	130 MHz
PCI	200 MHz
PCI-X	200 MHz
HSTL-I	300 MHz
HSTL-II	300 MHz
SSTL2-I	300 MHz
SSTL2-II	300 MHz
SSTL3-I	300 MHz
SSTL3-II	300 MHz
GTL+ 3.3 V	300 MHz
GTL+ 2.5 V	300 MHz
GTL 3.3 V	300 MHz
GTL 2.5 V	300 MHz
LVDS	350 MHz
LVPECL	300 MHz

I/O Registers

Each I/O module contains several input, output, and enable registers. Refer to Figure 2-98 for a simplified representation of the I/O block.

The number of input registers is selected by a set of switches (not shown in Figure 2-98) between registers to implement single or differential data transmission to and from the FPGA core. The Designer software sets these switches for the user.

A common CLR/PRE signal is employed by all I/O registers when I/O register combining is used. Input register 2 does not have a CLR/PRE pin, as this register is used for DDR implementation. The I/O register combining must satisfy some rules.



Note: Fusion I/Os have registers to support DDR functionality (see the "Double Data Rate (DDR) Support" section on page 2-140 for more information).

Figure 2-98 • **I/O Block Logical Representation**

Double Data Rate (DDR) Support

Fusion Pro I/Os support 350 MHz DDR inputs and outputs. In DDR mode, new data is present on every transition of the clock signal. Clock and data lines have identical bandwidths and signal integrity requirements, making it very efficient for implementing very high-speed systems.

DDR interfaces can be implemented using HSTL, SSTL, LVDS, and LVPECL I/O standards. In addition, high-speed DDR interfaces can be implemented using LVDS I/O.

Input Support for DDR

The basic structure to support a DDR input is shown in Figure 2-99. Three input registers are used to capture incoming data, which is presented to the core on each rising edge of the I/O register clock.

Each I/O tile on Fusion devices supports DDR inputs.

Output Support for DDR

The basic DDR output structure is shown in Figure 2-100 on page 2-141. New data is presented to the output every half clock cycle. Note: DDR macros and I/O registers do not require additional routing. The combiner automatically recognizes the DDR macro and pushes its registers to the I/O register area at the edge of the chip. The routing delay from the I/O registers to the I/O buffers is already taken into account in the DDR macro.

Refer to the Actel application note *Using DDR for Fusion Devices* for more information.

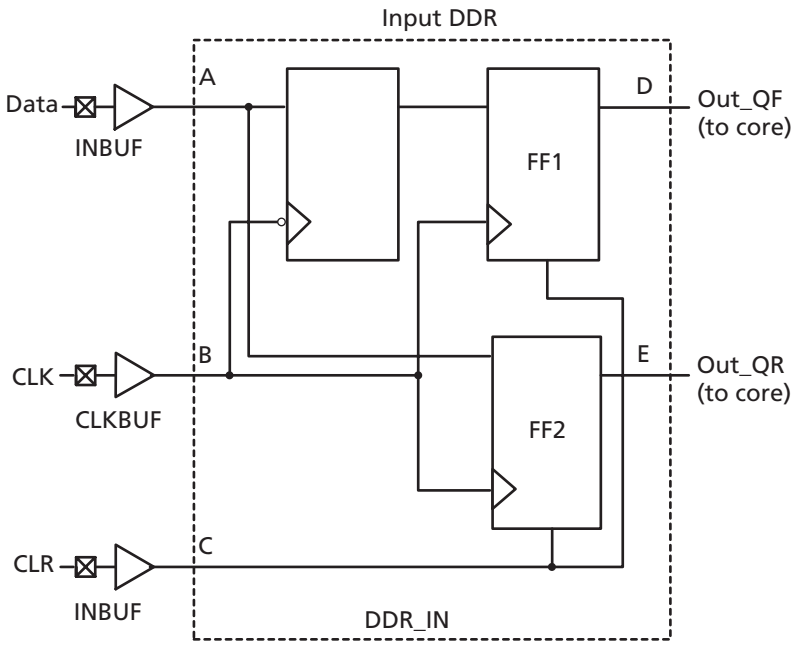


***Figure 2-99* • DDR Input Register Support in Fusion Devices**

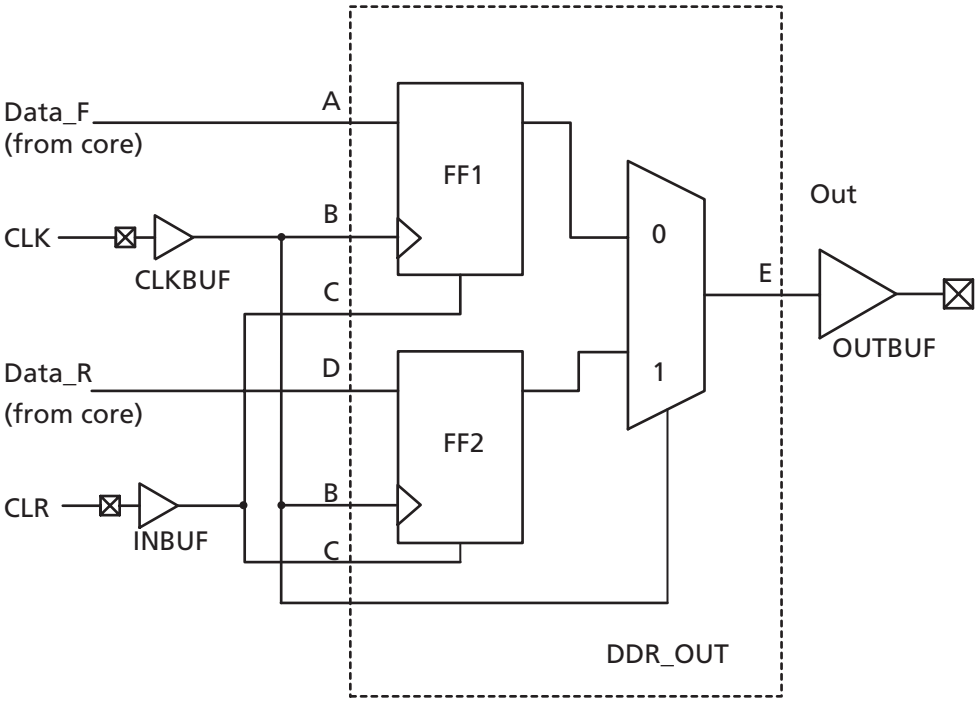


***Figure 2-100* • DDR Output Support in Fusion Devices**

Hot-Swap Support

Hot-swapping (also called hot plugging) is the operation of hot insertion or hot removal of a card in (or from) a powered-up system. The levels of hot-swap support and examples of related applications are described in Table 2-71. The I/Os also need to be configured in hot insertion mode if hot plugging compliance is required.

Table 2-71 • **Levels of Hot-Swap Support**

Hot Swapping Level	Description	Power Applied to Device	Bus State	Card Ground Connection	Device Circuitry Connected to Bus Pins	Example of Application with Cards that Contain Fusion Devices	Compliance of Fusion Devices
1	Cold-swap	No	–	–	–	System and card with Actel FPGA chip are powered down, then card gets plugged into system, then power supplies are turned on for system but not for FPGA on card.	Compliant I/Os can but do not have to be set to hot insertion mode.
2	Hot-swap while reset	Yes	Held in reset state	Must be made and maintained for 1 ms before, during, and after insertion/ removal	–	In PCI hot plug specification, reset control circuitry isolates the card busses until the card supplies are at their nominal operating levels and stable.	Compliant I/Os can but do not have to be set to hot insertion mode.
3	Hot-swap while bus idle	Yes	Held idle (no ongoing I/O processes during insertion/removal)	Same as Level 2	Must remain glitch-free during power-up or power-down	Board bus shared with card bus is "frozen," and there is no toggling activity on bus. It is critical that the logic states set on the bus signal do not get disturbed during card insertion/removal.	Compliant with cards with two levels of staging. I/Os have to be set to hot insertion mode.
4	Hot-swap on an active bus	Yes	Bus may have active I/O processes ongoing, but device being inserted or removed must be idle.	Same as Level 2	Same as Level 3	There is activity on the system bus, and it is critical that the logic states set on the bus signal do not get disturbed during card insertion/removal.	Compliant with cards with two levels of staging. I/Os have to be set to hot insertion mode.

For Fusion devices requiring Level 3 and/or Level 4 compliance, the board drivers connected to Fusion I/Os need to have 10 kΩ (or lower) output drive resistance at hot insertion, and 1 kΩ (or lower) output drive resistance at hot removal. This is the resistance of the transmitter sending a signal to the Fusion I/O, and no additional resistance is needed on the board. If that cannot be assured, three levels of staging can be used to meet Level 3 and/or Level 4 compliance. Cards with two levels of staging should have the following sequence:

1. Grounds
2. Powers, I/Os, other pins

Cold-Sparing Support

Cold-sparing means that a subsystem with no power applied (usually a circuit board) is electrically connected to the system that is in operation. This means that all input buffers of the subsystem must present very high input impedance with no power applied so as not to disturb the operating portion of the system.

Pro I/O banks and standard I/O banks fully support cold-sparing.

For Pro I/O banks, standards such as PCI that require I/O clamp diodes, can also achieve cold-sparing compliance, since clamp diodes get disconnected internally when the supplies are at 0 V.

For Advanced I/O banks, since the I/O clamp diode is always active, cold-sparing can be accomplished either by employing a bus switch to isolate the device I/Os from the rest of the system or by driving each advanced I/O pin to 0 V.

If Standard I/O banks are used in applications requiring cold-sparing, a discharge path from the power supply to ground should be provided. This can be done with a discharge resistor or a switched resistor. This is necessary because the standard I/O buffers do not have built-in I/O clamp diodes.

If a resistor is chosen, the resistor value must be calculated based on decoupling capacitance on a given power supply on the board (this decoupling capacitor is in parallel with the resistor). The RC time constant should ensure full discharge of supplies before cold-sparing functionality is required. The resistor is necessary to ensure that the power pins are discharged to ground every time there is an interruption of power to the device.

I/O cold-sparing may add additional current if the pin is configured with either a pull-up or pull down resistor and driven in the opposite direction. A small static current is induced on each IO pin when the pin is driven to a voltage opposite to the weak pull resistor. The current is equal to the voltage drop across the input pin divided by the pull resistor. Please refer to Table 2-92 on page 2-171, Table 2-93 on page 2-171, and Table 2-94 on page 2-173 for the specific pull resistor value for the corresponding I/O standard.

For example, assuming an LVTTL 3.3 V input pin is configured with a weak Pull-up resistor, a current will flow through the pull-up resistor if the input pin is driven low. For an LVTTL 3.3 V, pull-up resistor is ~45 kΩ and the resulting current is equal to 3.3 V / 45 kΩ = 73 µA for the I/O pin. This is true also when a weak pull-down is chosen and the input pin is driven high. Avoiding this current can be done by driving the input low when a weak pull-down resistor is used, and driving it high when a weak pull-up resistor is used.

In Active and Static modes, this current draw can occur in the following cases:

- Input buffers with pull-up, driven low
- Input buffers with pull-down, driven high
- Bidirectional buffers with pull-up, driven low
- Bidirectional buffers with pull-down, driven high
- Output buffers with pull-up, driven low
- Output buffers with pull-down, driven high
- Tristate buffers with pull-up, driven low
- Tristate buffers with pull-down, driven high

Electrostatic Discharge (ESD) Protection

Fusion devices are tested per JEDEC Standard JESD22-A114-B.

Fusion devices contain clamp diodes at every I/O, global, and power pad. Clamp diodes protect all device pads against damage from ESD as well as from excessive voltage transients.

Each I/O has two clamp diodes. One diode has its positive (P) side connected to the pad and its negative (N) side connected to V_{CCI}. The second diode has its P side connected to GND and its N side connected to the pad. During operation, these diodes are normally biased in the Off state, except when transient voltage is significantly above V_{CCI} or below GND levels.

By selecting the appropriate I/O configuration, the diode is turned on or off. Refer to Table 2-72 and Table 2-73 on page 2-144 for more information about I/O standards and the clamp diode.

The second diode is always connected to the pad, regardless of the I/O configuration selected.

Table 2-72 • **Fusion Standard and Advanced I/O – Hot-Swap and 5 V Input Tolerance Capabilities**

	Clamp Diode		Hot Insertion		5 V Input Tolerance [1]			
I/O Assignment	Standard I/O	Advanced I/O	Standard I/O	Advanced I/O	Standard I/O	Advanced I/O	Input Buffer	Output Buffer
3.3 V LVTTL/LVCMOS	No	Yes	Yes	No	Yes[1]	Yes[1]	Enabled/Disabled	
3.3 V PCI, 3.3 V PCI-X	N/A	Yes	N/A	No	N/A	Yes[1]	Enabled/Disabled	
LVCMOS 2.5 V	No	Yes	Yes	No	No	No	Enabled/Disabled	
LVCMOS 2.5 V / 5.0 V	N/A	Yes	N/A	No	N/A	Yes[2]	Enabled/Disabled	
LVCMOS 1.8 V	No	Yes	Yes	No	No	No	Enabled/Disabled	
LVCMOS 1.5 V	No	Yes	Yes	No	No	No	Enabled/Disabled	
Differential, LVDS/BLVDS/M-LVDS/ LVPECL [3]	N/A	Yes	N/A	No	N/A	No	Enabled/Disabled	

Notes:

1. *Can be implemented with an external IDT bus switch, resistor divider, or Zener with resistor.*
2. *Can be implemented with an external resistor and an internal clamp diode.*
3. *Bidirectional LVPECL buffers are not supported. I/Os can be configured as either input buffers or output buffers.*

Table 2-73 • **Fusion Pro I/O – Hot-Swap and 5 V Input Tolerance Capabilities**

I/O Assignment	Clamp Diode	Hot Insertion	5 V Input Tolerance	Input Buffer	Output Buffer
3.3 V LVTTL/LVCMOS	No	Yes	Yes[1]	Enabled/Disabled	
3.3 V PCI, 3.3 V PCI-X	Yes	No	Yes[1]	Enabled/Disabled	
LVCMOS 2.5 V [3]	No	Yes	No	Enabled/Disabled	
LVCMOS 2.5 V / 5.0 V [3]	Yes	No	Yes[2]	Enabled/Disabled	
LVCMOS 1.8 V	No	Yes	No	Enabled/Disabled	
LVCMOS 1.5 V	No	Yes	No	Enabled/Disabled	
Voltage-Referenced Input Buffer	No	Yes	No	Enabled/Disabled	
Differential, LVDS/BLVDS/M-LVDS/LVPECL[4]	No	Yes	No	Enabled/Disabled	

Notes:

1. *Can be implemented with an external IDT bus switch, resistor divider, or Zener with resistor.*
2. *Can be implemented with an external resistor and an internal clamp diode.*
3. *In the SmartGen, FlashROM, Flash Memory System Builder, and Analog System Builder User's Guide, select the LVCMOS5 macro for the LVCMOS 2.5 V / 5.0 V I/O standard or the LVCMOS25 macro for the LVCMOS 2.5 V I/O standard.*
4. *Bidirectional LVPECL buffers are not supported. I/Os can be configured as either input buffers or output buffers.*

5 V Input Tolerance

I/Os can support 5 V input tolerance when LVTTL 3.3 V, LVCMOS 3.3 V, LVCMOS 2.5 V / 5 V, and LVCMOS 2.5 V configurations are used (see Table 2-74 on page 2-148 for more details). There are four recommended solutions (see Figure 2-101 to Figure 2-104 on page 2-147 for details of board and macro setups) to achieve 5 V receiver tolerance. All the solutions meet a common requirement of limiting the voltage at the input to 3.6 V or less. In fact, the I/O absolute maximum voltage rating is 3.6 V, and any voltage above 3.6 V may cause long-term gate oxide failures.

Solution 1

The board-level design needs to ensure that the reflected waveform at the pad does not exceed the limits provided in Table 3-4 on page 3-4. This is a long-term reliability requirement.

This scheme will also work for a 3.3 V PCI / PCI-X configuration, but the internal diode should not be used for clamping, and the voltage must be limited by the two external resistors, as explained below. Relying on the diode clamping would create an excessive pad DC voltage of 3.3 V + 0.7 V = 4 V.

The following are some examples of possible resistor values (based on a simplified simulation model with no line effects and 10 Ω transmitter output resistance, where Rtx_out_high = (V_{CCI} – V_{OH}) / I_{OH}, Rtx_out_low = V_{OL} / I_{OL}).

Example 1 (high speed, high current):

Rtx_out_high = Rtx_out_low = 10 Ω

R1 = 36 Ω (±5%), P(r1)min = 0.069 Ω

R2 = 82 Ω (±5%), P(r2)min = 0.158 Ω

Imax_tx = 5.5 V / (82 * 0.95 + 36 * 0.95 + 10) = 45.04 mA

t_{RISE} = t_{FALL} = 0.85 ns at C_pad_load = 10 pF (includes up to 25% safety margin)

t_{RISE} = t_{FALL} = 4 ns at C_pad_load = 50 pF (includes up to 25% safety margin)

Example 2 (low–medium speed, medium current):

Rtx_out_high = Rtx_out_low = 10 Ω

R1 = 220 Ω (±5%), P(r1)min = 0.018 Ω

R2 = 390 Ω (±5%), P(r2)min = 0.032 Ω

Imax_tx = 5.5 V / (220 * 0.95 + 390 * 0.95 + 10) = 9.17 mA

t_{RISE} = t_{FALL} = 4 ns at C_pad_load = 10 pF (includes up to 25% safety margin)

t_{RISE} = t_{FALL} = 20 ns at C_pad_load = 50 pF (includes up to 25% safety margin)

Other values of resistors are also allowed as long as the resistors are sized appropriately to limit the voltage at the receiving end to 2.5 V < Vin(rx) < 3.6 V when the transmitter sends a logic 1. This range of Vin_dc(rx) must be assured for any combination of transmitter supply (5 V ± 0.5 V), transmitter output resistance, and board resistor tolerances.

Temporary overshoots are allowed according to Table 3-4 on page 3-4.

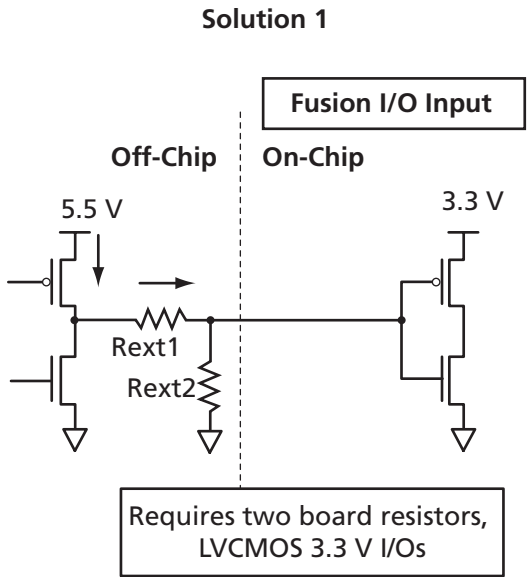


Figure 2-101 • **Solution 1**

Solution 2

The board-level design must ensure that the reflected waveform at the pad does not exceed limits provided in Table 3-4 on page 3-4. This is a long-term reliability requirement.

This scheme will also work for a 3.3 V PCI/PCI-X configuration, but the internal diode should not be used for clamping, and the voltage must be limited by the external resistors and Zener, as shown in Figure 2-102. Relying on the diode clamping would create an excessive pad DC voltage of 3.3 V + 0.7 V = 4 V.

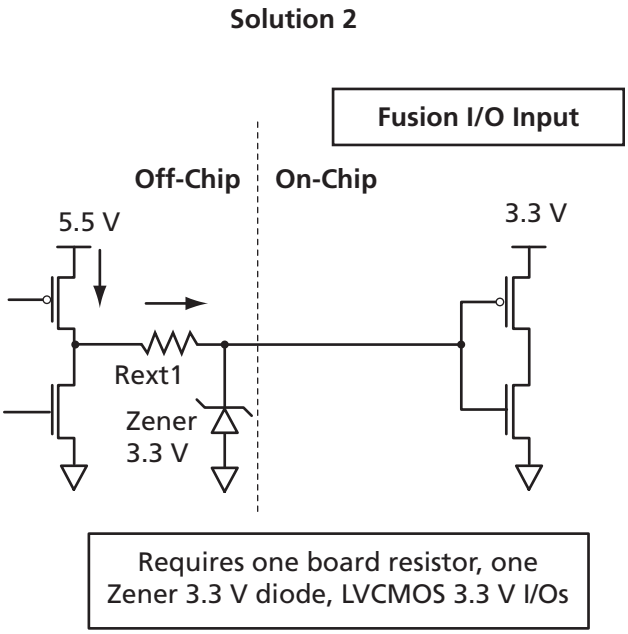


Figure 2-102 • **Solution 2**

Solution 3

The board-level design must ensure that the reflected waveform at the pad does not exceed limits provided in Table 3-4 on page 3-4. This is a long-term reliability requirement.

This scheme will also work for a 3.3 V PCI/PCIX configuration, but the internal diode should not be used for clamping, and the voltage must be limited by the bus switch, as shown in Figure 2-103. Relying on the diode clamping would create an excessive pad DC voltage of 3.3 V + 0.7 V = 4 V.

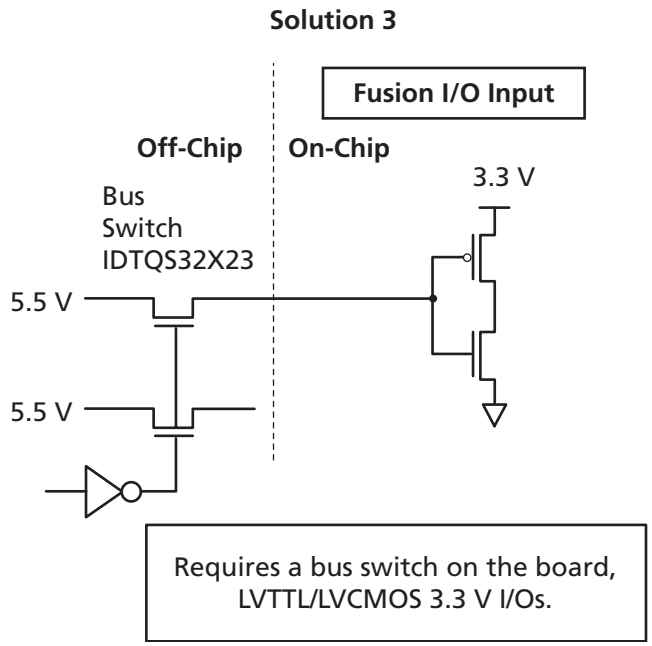


***Figure 2-103* • Solution 3**

Solution 4

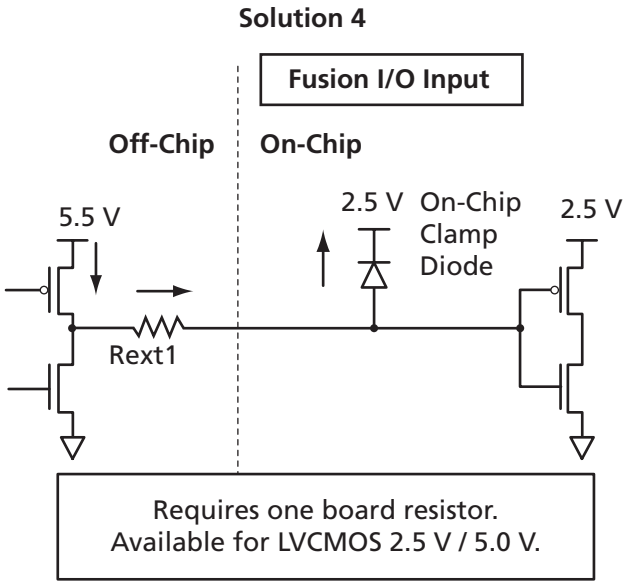


***Figure 2-104* • Solution 4**

Table 2-74 • **Comparison Table for 5 V–Compliant Receiver Scheme**

Scheme	Board Components	Speed	Current Limitations
1	Two resistors	Low to high[1]	Limited by transmitter's drive strength
2	Resistor and Zener 3.3 V	Medium	Limited by transmitter's drive strength
3	Bus switch	High	N/A
4	Minimum resistor value[2] R = 47 Ω at T_J = 70°C R = 150 Ω at T_J = 85°C R = 420 Ω at T_J = 100°C	Medium	Maximum diode current at 100% duty cycle, signal constantly at '1' 52.7 mA at T_J =70°C / 10-year lifetime 16.5 mA at T_J = 85°C / 10-year lifetime 5.9 mA at T_J = 100°C / 10-year lifetime For duty cycles other than 100%, the currents can be increased by a factor = 1 / (duty cycle). Example: 20% duty cycle at 70°C Maximum current = (1 / 0.2) * 52.7 mA = 5 * 52.7 mA = 263.5 mA

Notes:

1. *Speed and current consumption increase as the board resistance values decrease.*
2. *Resistor values ensure I/O diode long-term reliability.*

5 V Output Tolerance

Fusion I/Os must be set to 3.3 V LVTTL or 3.3 V LVCMOS mode to reliably drive 5 V TTL receivers. It is also critical that there be NO external I/O pull-up resistor to 5 V, since this resistor would pull the I/O pad voltage beyond the 3.6 V absolute maximum value and consequently cause damage to the I/O.

When set to 3.3 V LVTTL or 3.3 V LVCMOS mode, Fusion I/Os can directly drive signals into 5 V TTL receivers. In fact, V_{OL} = 0.4 V and V_{OH} = 2.4 V in both 3.3 V LVTTL and 3.3 V LVCMOS modes exceed the V_{IL} = 0.8 V and V_{IH} = 2 V level requirements of 5 V TTL receivers. Therefore, level '1' and level '0' will be recognized correctly by 5 V TTL receivers.

Simultaneously Switching Outputs and PCB Layout

- Simultaneously switching outputs (SSOs) can produce signal integrity problems on adjacent signals that are not part of the SSO bus. Both inductive and capacitive coupling parasitics of bond wires inside packages and of traces on PCBs will transfer noise from SSO busses onto signals adjacent to those busses. Additionally, SSOs can produce ground bounce noise and V_{CCI} dip noise. These two noise types are caused by rapidly changing currents through GND and V_{CCI} package pin inductances during switching activities:
- Ground bounce noise voltage = L(GND) * di/dt
- V_{CCI} dip noise voltage = L(V_{CCI}) * di/dt

Any group of four or more input pins switching on the same clock edge is considered an SSO bus. The shielding should be done both on the board and inside the package unless otherwise described.

In-package shielding can be achieved in several ways; the required shielding will vary depending on whether pins next to SSO bus are LVTTL/LVCMOS inputs, LVTTL/LVCMOS outputs, or GTL/SSTL/HSTL/LVDS/LVPECL inputs and outputs. Board traces in the vicinity of the SSO bus have to be adequately shielded from mutual coupling and inductive noise that can be generated by the SSO bus. Also, noise generated by the SSO bus needs to be reduced inside the package.

PCBs perform an important function in feeding stable supply voltages to the IC and, at the same time, maintaining signal integrity between devices.

Key issues that need to considered are as follows:

- Power and ground plane design and decoupling network design
- Transmission line reflections and terminations

Selectable Skew between Output Buffer Enable/Disable Time

The configurable skew block is used to delay the output buffer assertion (enable) without affecting deassertion (disable) time.

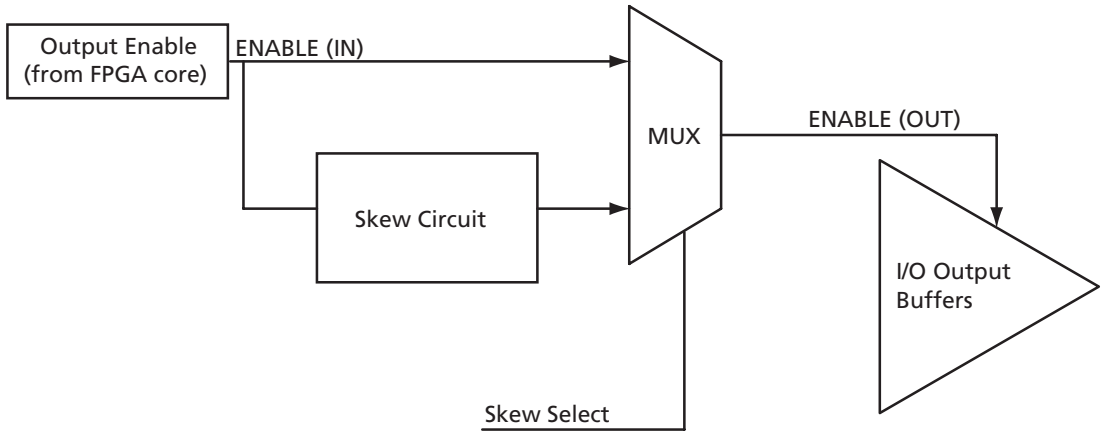


***Figure 2-105* • Block Diagram of Output Enable Path**

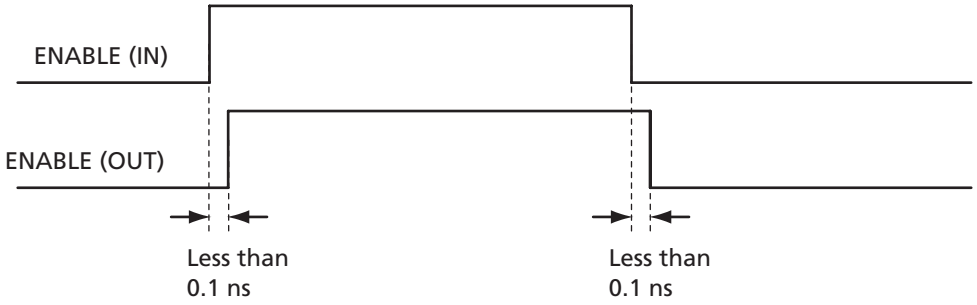


***Figure 2-106* • Timing Diagram (option1: bypasses skew circuit)**

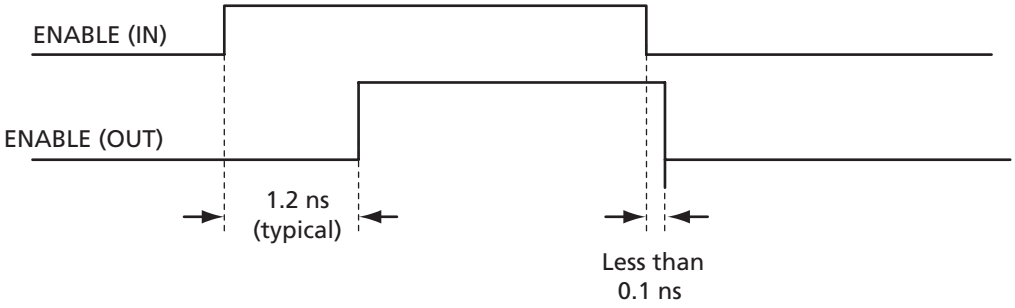


***Figure 2-107* • Timing Diagram (option 2: enables skew circuit)**

At the system level, the skew circuit can be used in applications where transmission activities on bidirectional data lines need to be coordinated. This circuit, when selected, provides a timing margin that can prevent bus contention and subsequent data loss or transmitter overstress due to transmitter-to-transmitter current shorts. Figure 2-108 presents an example of the skew circuit implementation in a bidirectional communication system. Figure 2-109 shows how bus contention is created, and Figure 2-110 on page 2-152 shows how it can be avoided with the skew circuit.

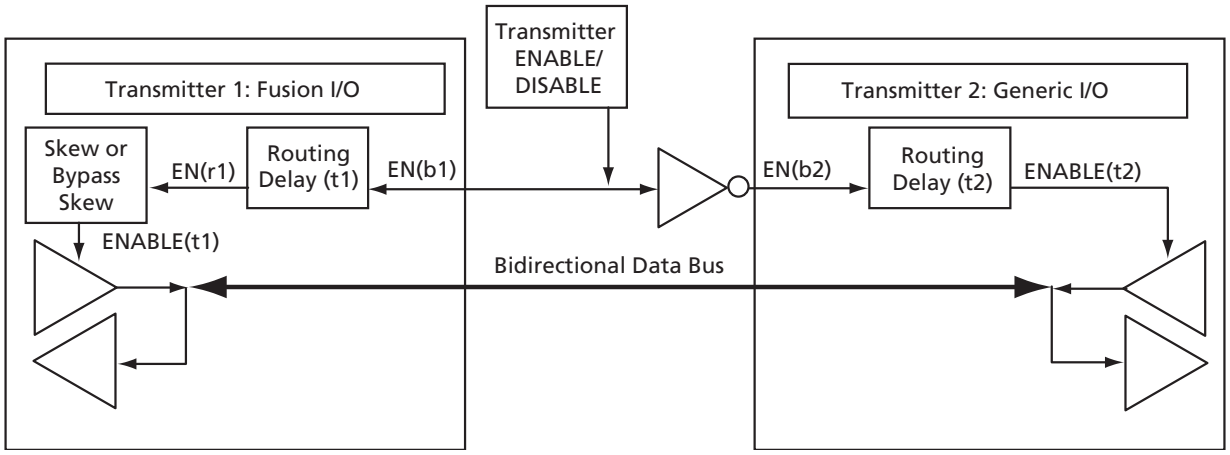


***Figure 2-108* • Example of Implementation of Skew Circuits in Bidirectional Transmission Systems Using Fusion Devices**

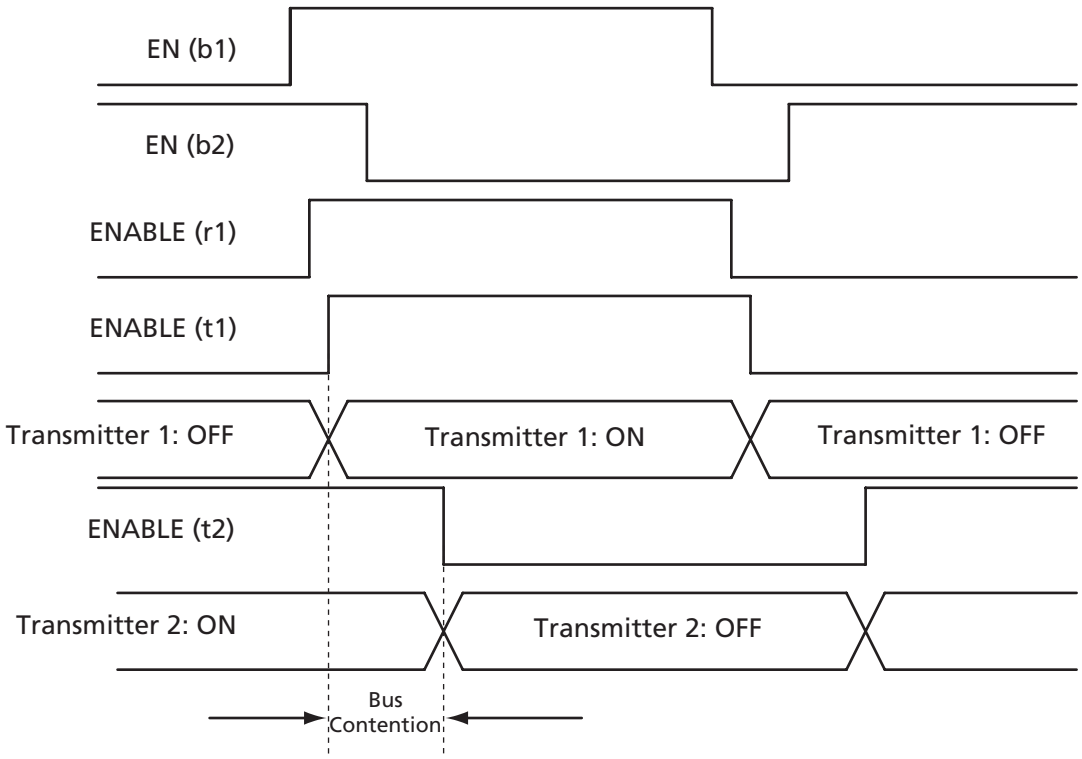


***Figure 2-109* • Timing Diagram (bypasses skew circuit)**

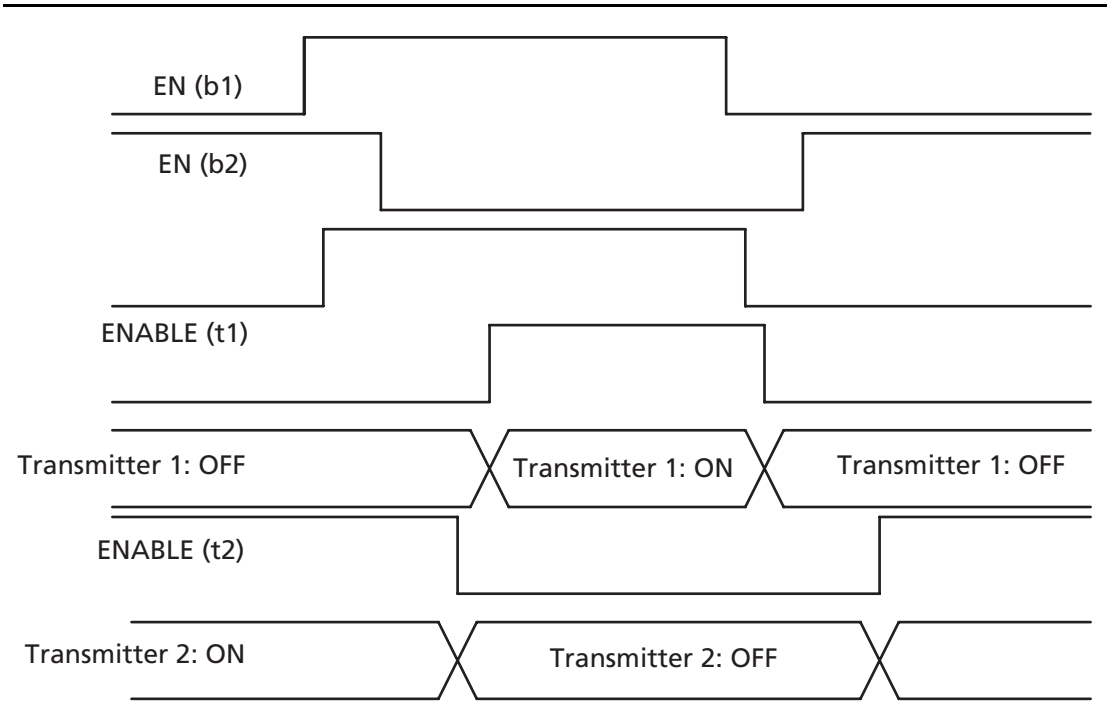


***Figure 2-110* • Timing Diagram (with skew circuit selected)**

Weak Pull-Up and Weak Pull-Down Resistors

Fusion devices support optional weak pull-up and pull-down resistors for each I/O pin. When the I/O is pulled up, it is connected to the V_{CCI} of its corresponding I/O bank. When it is pulled down, it is connected to GND. Refer to Table 2-94 on page 2-173 for more information.

Slew Rate Control and Drive Strength

Fusion devices support output slew rate control: high and low. The high slew rate option is recommended to minimize the propagation delay. This high-speed option may introduce noise into the system if appropriate signal integrity measures are not adopted. Selecting a low slew rate reduces this kind of noise but adds some delays in the system. Low slew rate is recommended when bus transients are expected. Drive strength should also be selected according to the design requirements and noise immunity of the system.

The output slew rate and multiple drive strength controls are available in LVTTL/LVCMOS 3.3 V, LVCMOS 2.5 V, LVCMOS 2.5 V / 5.0 V input, LVCMOS 1.8 V, and LVCMOS 1.5 V. All other I/O standards have a high output slew rate by default.

For Fusion slew rate and drive strength specifications, refer to the appropriate I/O bank table:

- Fusion Standard I/O (Table 2-75 on page 2-153)
- Fusion Advanced I/O (Table 2-76 on page 2-153)
- Fusion Pro I/O (Table 2-77 on page 2-153)

Table 2-80 on page 2-156 lists the default values for the above selectable I/O attributes as well as those that are preset for each I/O standard.

Refer to Table 2-75, Table 2-76, and Table 2-77 on page 2-153 for SLEW and OUT_DRIVE settings. Table 2-78 on page 2-154 and Table 2-79 on page 2-155 list the I/O default attributes. Table 2-80 on page 2-156 lists the voltages for the supported I/O standards.

Table 2-75 • **Fusion Standard I/O Standards—OUT_DRIVE Settings**

I/O Standards	OUT_DRIVE (mA)					
	2	4	6	8	Slew	
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	High	Low
LVCMOS 2.5 V	✓	✓	✓	✓	High	Low
LVCMOS 1.8 V	✓	✓	–	–	High	Low
LVCMOS 1.5 V	✓	–	–	–	High	Low

Table 2-76 • **Fusion Advanced I/O Standards—SLEW and OUT_DRIVE Settings**

I/O Standards	OUT_DRIVE (mA)							
	2	4	6	8	12	16	Slew	
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 2.5 V	✓	✓	✓	✓	✓	–	High	Low
LVCMOS 1.8 V	✓	✓	✓	✓	–	–	High	Low
LVCMOS 1.5 V	✓	✓	–	–	–	–	High	Low

Table 2-77 • **Fusion Pro I/O Standards—SLEW and OUT_DRIVE Settings**

I/O Standards	OUT_DRIVE (mA)							Slew	
	2	4	6	8	12	16	24		
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 2.5 V	✓	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 2.5 V/5.0 V	✓	✓	✓	✓	✓	✓	✓	High	Low
LVCMOS 1.8 V	✓	✓	✓	✓	✓	✓	–	High	Low
LVCMOS 1.5 V	✓	✓	✓	✓	✓	–	–	High	Low

Table 2-78 • **Fusion Pro I/O Default Attributes**

I/O Standards	SLEW (output only)	OUT_DRIVE (output only)	SKEW (tribuf and bibuf only)	RES_PULL	OUT_LOAD (output only)	COMBINE_REGISTER	IN_DELAY (input only)	IN_DELAY_VAL (input only)	SCHMITT_TRIGGER (input only)
LVTTL/LVCMOS 3.3 V	Refer to the following tables for more information: Table 2-75 on page 2-153 Table 2-76 on page 2-153 Table 2-77 on page 2-153	Refer to the following tables for more information: Table 2-75 on page 2-153 Table 2-76 on page 2-153 Table 2-77 on page 2-153	Off	None	35 pF	–	Off	0	Off
LVCMOS 2.5 V			Off	None	35 pF	–	Off	0	Off
LVCMOS 2.5/5.0 V			Off	None	35 pF	–	Off	0	Off
LVCMOS 1.8 V			Off	None	35 pF	–	Off	0	Off
LVCMOS 1.5 V			Off	None	35 pF	–	Off	0	Off
PCI (3.3 V)			Off	None	10 pF	–	Off	0	Off
PCI-X (3.3 V)			Off	None	10 pF	–	Off	0	Off
GTL+ (3.3 V)			Off	None	10 pF	–	Off	0	Off
GTL+ (2.5 V)			Off	None	10 pF	–	Off	0	Off
GTL (3.3 V)			Off	None	10 pF	–	Off	0	Off
GTL (2.5 V)			Off	None	10 pF	–	Off	0	Off
HSTL Class I			Off	None	20 pF	–	Off	0	Off
HSTL Class II			Off	None	20 pF	–	Off	0	Off
SSTL2 Class I and II			Off	None	30 pF	–	Off	0	Off
SSTL3 Class I and II			Off	None	30 pF	–	Off	0	Off
LVDS, BLVDS, M-LVDS			Off	None	0 pF	–	Off	0	Off
LVPECL			Off	None	0 pF	–	Off	0	Off

Table 2-79 • **Advanced I/O Default Attributes**

I/O Standards	SLEW (output only)	OUT_DRIVE (output only)	SKEW (tribuf and bibuf only)	RES_PULL	OUT_LOAD (output only)	COMBINE_REGISTER
LVTTL/LVCMOS 3.3 V	Refer to the following tables for more information: Table 2-75 on page 2-153 Table 2-76 on page 2-153 Table 2-77 on page 2-153	Refer to the following tables for more information: Table 2-75 on page 2-153 Table 2-76 on page 2-153 Table 2-77 on page 2-153	Off	None	35 pF	–
LVCMOS 2.5 V			Off	None	35 pF	–
LVCMOS 2.5/5.0 V			Off	None	35 pF	–
LVCMOS 1.8 V			Off	None	35 pF	–
LVCMOS 1.5 V			Off	None	35 pF	–
PCI (3.3 V)			Off	None	10 pF	–
PCI-X (3.3 V)			Off	None	10 pF	–
LVDS, BLVDS, M-LVDS			Off	None	–	–
LVPECL			Off	None	–	–

Table 2-80 • **Fusion Pro I/O Supported Standards and Corresponding V_{REF} and V_{TT} Voltages**

I/O Standard	Input/Output Supply Voltage (V_{CCI_TYP})	Input Reference Voltage (V_{REF_TYP})	Board Termination Voltage (V_{TT_TYP})
LVTTL/LVCMOS 3.3 V	3.30 V	–	–
LVCMOS 2.5 V	2.50 V	–	–
LVCMOS 2.5 V / 5.0 V Input	2.50 V	–	–
LVCMOS 1.8 V	1.80 V	–	–
LVCMOS 1.5 V	1.50 V	–	–
PCI 3.3 V	3.30 V	–	–
PCI-X 3.3 V	3.30 V	–	–
GTL+ 3.3 V	3.30 V	1.00 V	1.50 V
GTL+ 2.5 V	2.50 V	1.00 V	1.50 V
GTL 3.3 V	3.30 V	0.80 V	1.20 V
GTL 2.5 V	2.50 V	0.80 V	1.20 V
HSTL Class I	1.50 V	0.75 V	0.75 V
HSTL Class II	1.50 V	0.75 V	0.75 V
SSTL3 Class I	3.30 V	1.50 V	1.50 V
SSTL3 Class II	3.30 V	1.50 V	1.50 V
SSTL2 Class I	2.50 V	1.25 V	1.25 V
SSTL2 Class II	2.50 V	1.25 V	1.25 V
LVDS, BLVDS, M-LVDS	2.50 V	–	–
LVPECL	3.30 V	–	–

I/O Software Support

In the Fusion development software, default settings have been defined for the various I/O standards supported. Changes can be made to the default settings via the use of attributes; however, not all I/O attributes are applicable for all I/O standards. Table 2-81 and Table 2-82 list the valid I/O attributes that can be manipulated by the user for each I/O standard.

Single-ended I/O standards in Fusion support up to five different drive strengths.

Table 2-81 • **Fusion Standard and Advanced I/O Attributes vs. I/O Standard Applications**

I/O Standards	SLEW (output only)	OUT_DRIVE (output only)	SKEW (all macros with OE)*	RES_PULL	OUT_LOAD (output only)	COMBINE_REGISTER
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓
LVCMOS 2.5 V	✓	✓	✓	✓	✓	✓
LVCMOS 2.5/5.0 V	✓	✓	✓	✓	✓	✓
LVCMOS 1.8 V	✓	✓	✓	✓	✓	✓
LVCMOS 1.5 V	✓	✓	✓	✓	✓	✓
PCI (3.3 V)			✓		✓	✓
PCI-X (3.3 V)	✓		✓		✓	✓
LVDS, BLVDS, M-LVDS			✓			✓
LVPECL						✓

*Note: *This feature does not apply to the standard I/O banks, which are the north I/O banks of AFS090 and AFS250 devices*

Table 2-82 • **Fusion Pro I/O Attributes vs. I/O Standard Applications**

I/O Standards	SLEW (output only)	OUT_DRIVE (output only)	SKEW (all macros with OE)	RES_PULL	OUT_LOAD (output only)	COMBINE_REGISTER	IN_DELAY (input only)	IN_DELAY_VAL (input only)	SCHMITT_TRIGGER (input only)	HOT_SWAPPABLE
LVTTL/LVCMOS 3.3 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 2.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 2.5/5.0 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 1.8 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS 1.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
PCI (3.3 V)			✓		✓	✓	✓	✓		
PCI-X (3.3 V)	✓		✓		✓	✓	✓	✓		
GTL+ (3.3 V)			✓		✓	✓	✓	✓		✓
GTL+ (2.5 V)			✓		✓	✓	✓	✓		✓
GTL (3.3 V)			✓		✓	✓	✓	✓		✓
GTL (2.5 V)			✓		✓	✓	✓	✓		✓
HSTL Class I			✓		✓	✓	✓	✓		✓
HSTL Class II			✓		✓	✓	✓	✓		✓
SSTL2 Class I and II			✓		✓	✓	✓	✓		✓
SSTL3 Class I and II			✓		✓	✓	✓	✓		✓
LVDS, BLVDS, M-LVDS			✓			✓	✓	✓		✓
LVPECL						✓	✓	✓		✓

User I/O Naming Convention

Due to the comprehensive and flexible nature of Fusion device user I/Os, a naming scheme is used to show the details of the I/O (Figure 2-111 on page 2-160 and Figure 2-112 on page 2-160). The name identifies to which I/O bank it belongs, as well as the pairing and pin polarity for differential I/Os.

I/O Nomenclature = Gmn/IOuxwByVz

Gmn is only used for I/Os that also have CCC access—i.e., global pins.

G = Global

m = Global pin location associated with each CCC on the device: A (northwest corner), B (northeast corner), C (east middle), D (southeast corner), E (southwest corner), and F (west middle).

n = Global input MUX and pin number of the associated Global location m, either A0, A1, A2, B0, B1, B2, C0, C1, or C2. Figure 2-22 on page 2-28 shows the three input pins per clock source MUX at CCC location m.

u = I/O pair number in the bank, starting at 00 from the northwest I/O bank and proceeding in a clockwise direction.

x = P (Positive) or N (Negative) for differential pairs, or R (Regular – single-ended) for the I/Os that support single-ended and voltage-referenced I/O standards only. U (Positive-LVDS only) or V (Negative-LVDS only) restrict the I/O differential pair from being selected as an LVPECL pair.

w = D (Differential Pair), P (Pair), or S (Single-Ended). D (Differential Pair) if both members of the pair are bonded out to adjacent pins or are separated only by one GND or NC pin; P (Pair) if both members of the pair are bonded out but do not meet the adjacency requirement; or S (Single-Ended) if the I/O pair is not bonded out. For Differential (D) pairs, adjacency for ball grid packages means only vertical or horizontal. Diagonal adjacency does not meet the requirements for a true differential pair.

B = Bank

y = Bank number (0–3). The Bank number starts at 0 from the northwest I/O bank and proceeds in a clockwise direction.

V = Reference voltage

z = Minibank number

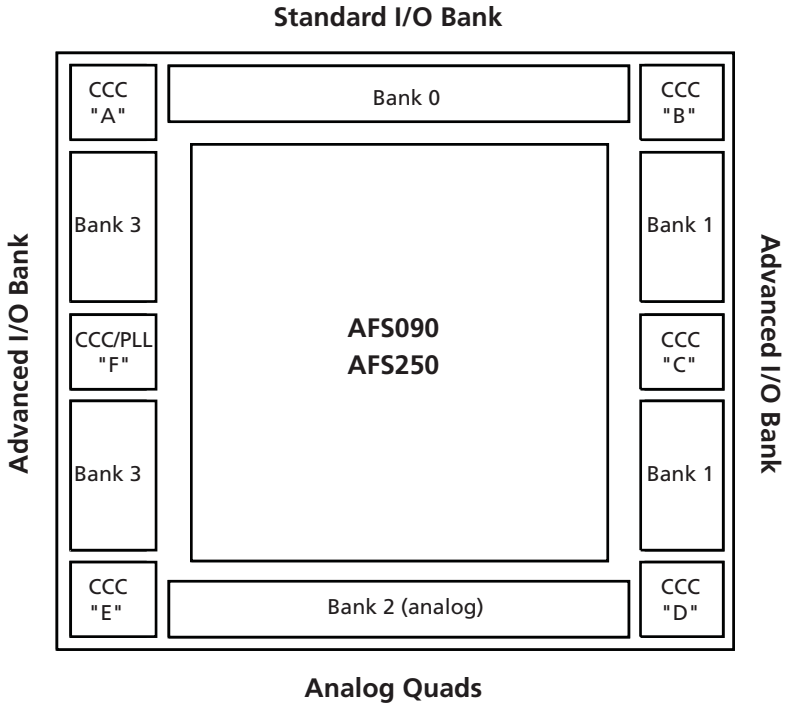


***Figure 2-111* • Naming Conventions of Fusion Devices with Three Digital I/O Banks**

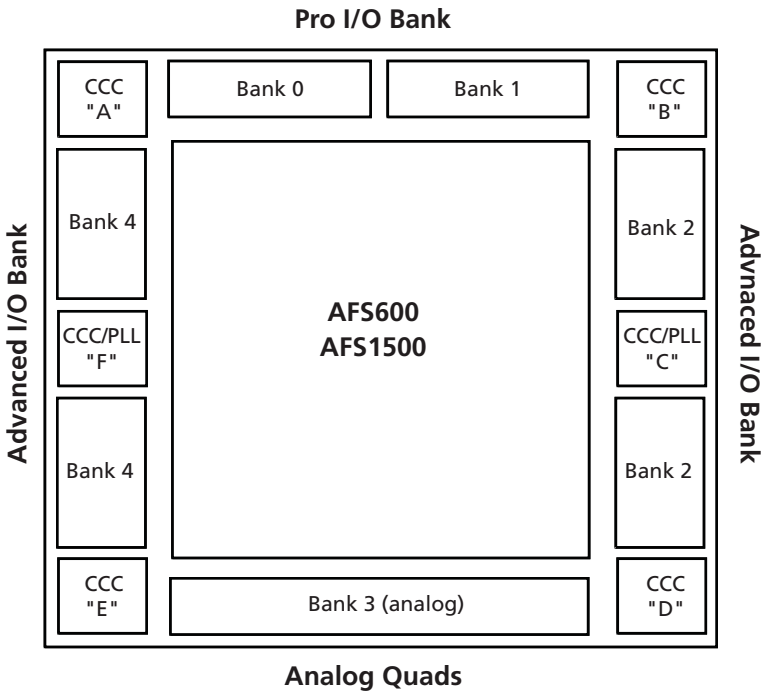


***Figure 2-112* • Naming Conventions of Fusion Devices with Four I/O Banks**

User I/O Characteristics

Timing Model

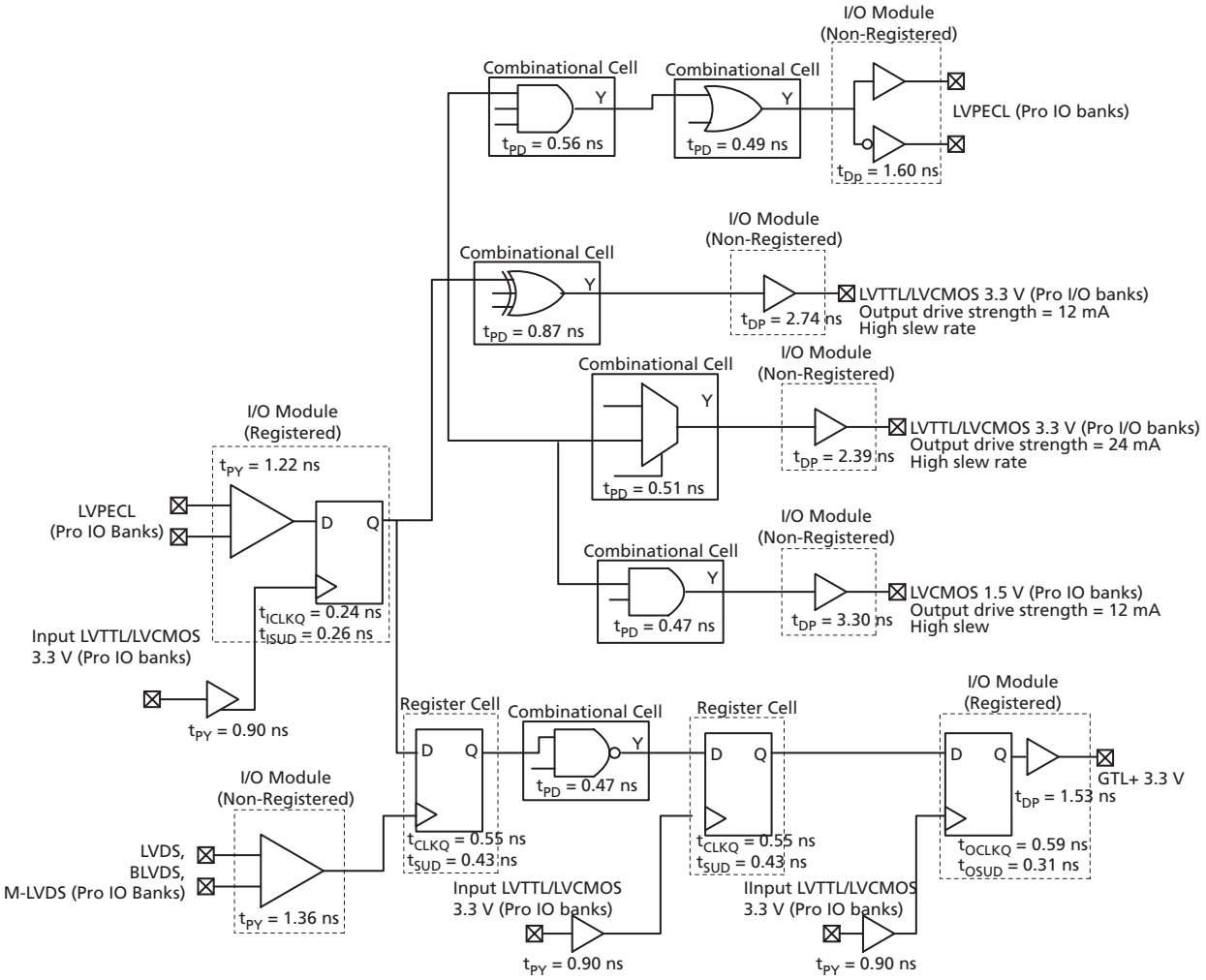


***Figure 2-113* • Timing Model**
Operating Conditions: –2 Speed, Commercial Temperature Range (T_J = 70°C), Worst-Case V_{CC} = 1.425 V

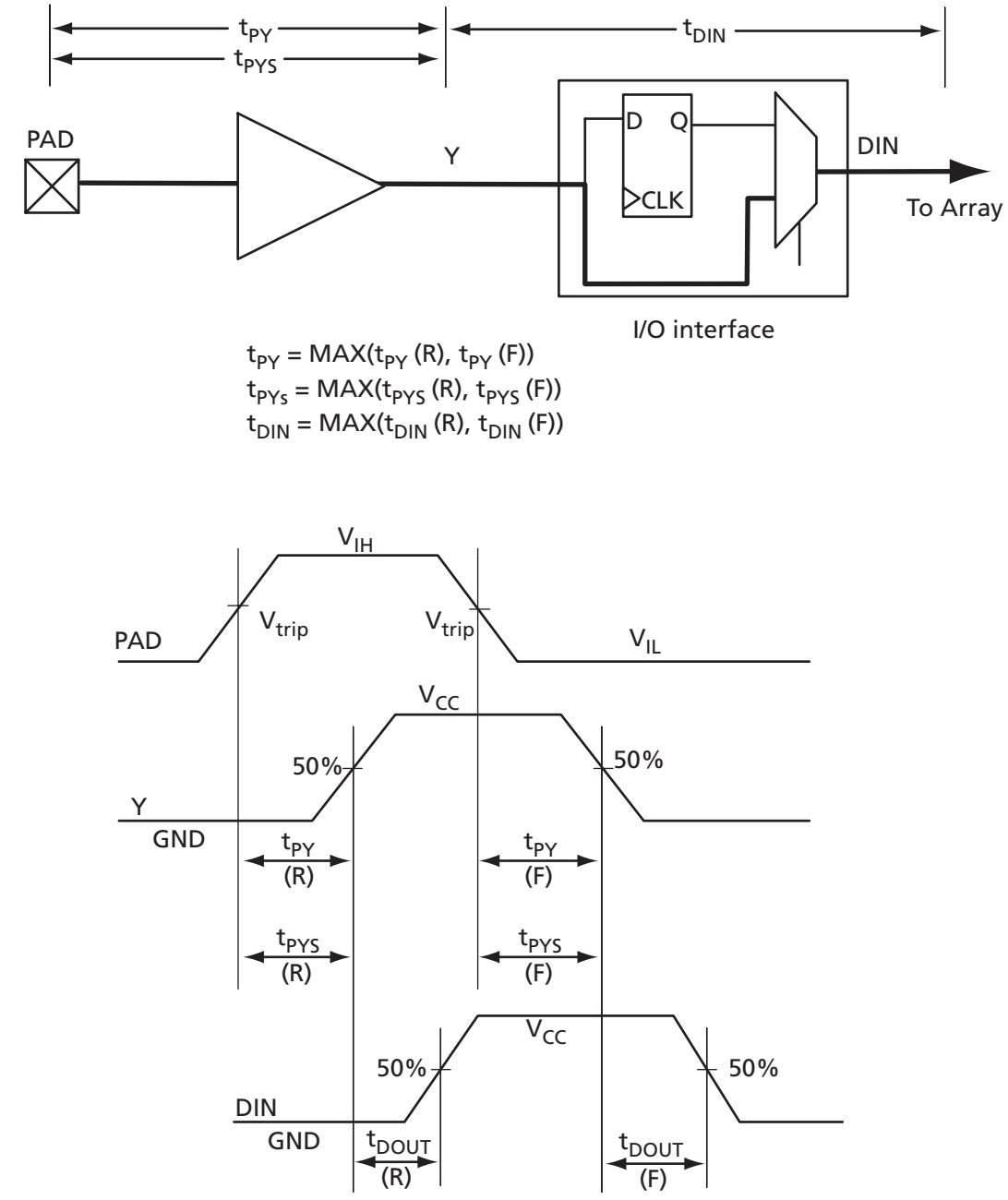


***Figure 2-114* • Input Buffer Timing Model and Delays (example)**

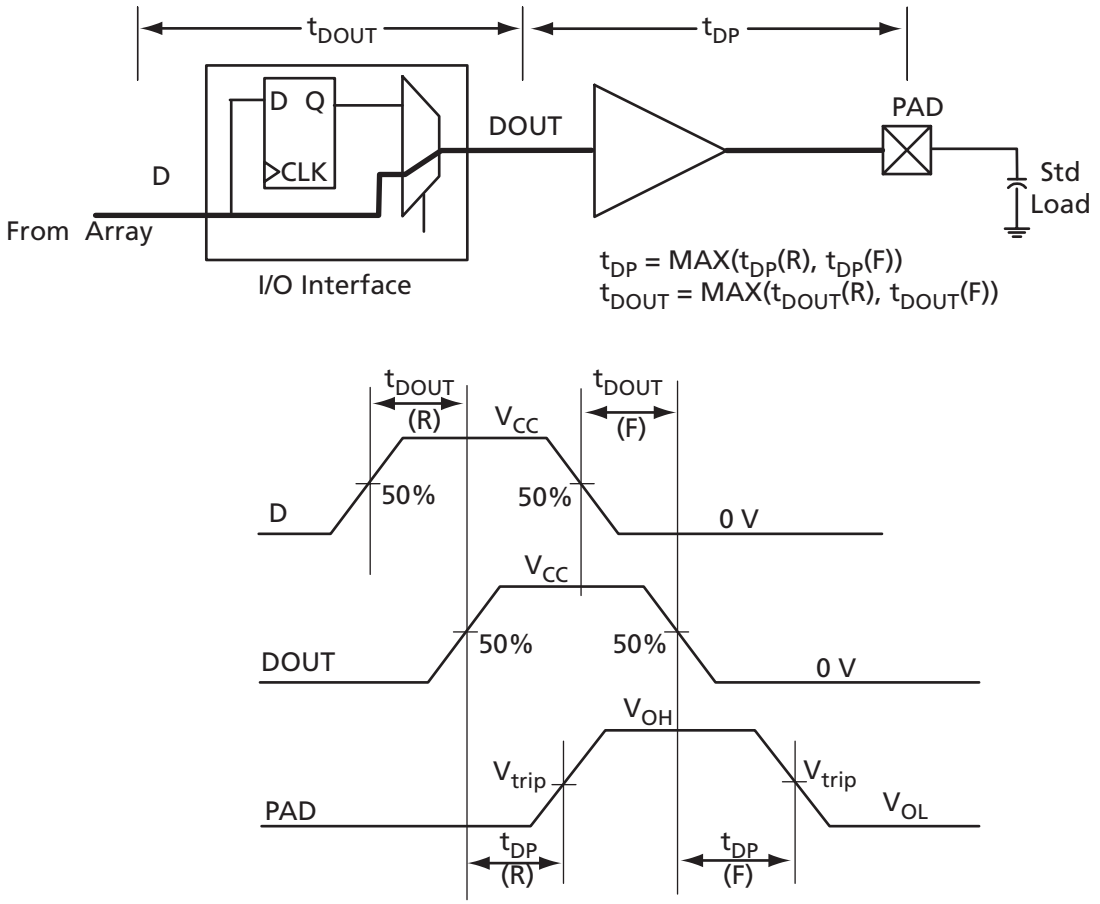


Figure 2-115 • **Output Buffer Model and Delays (example)**

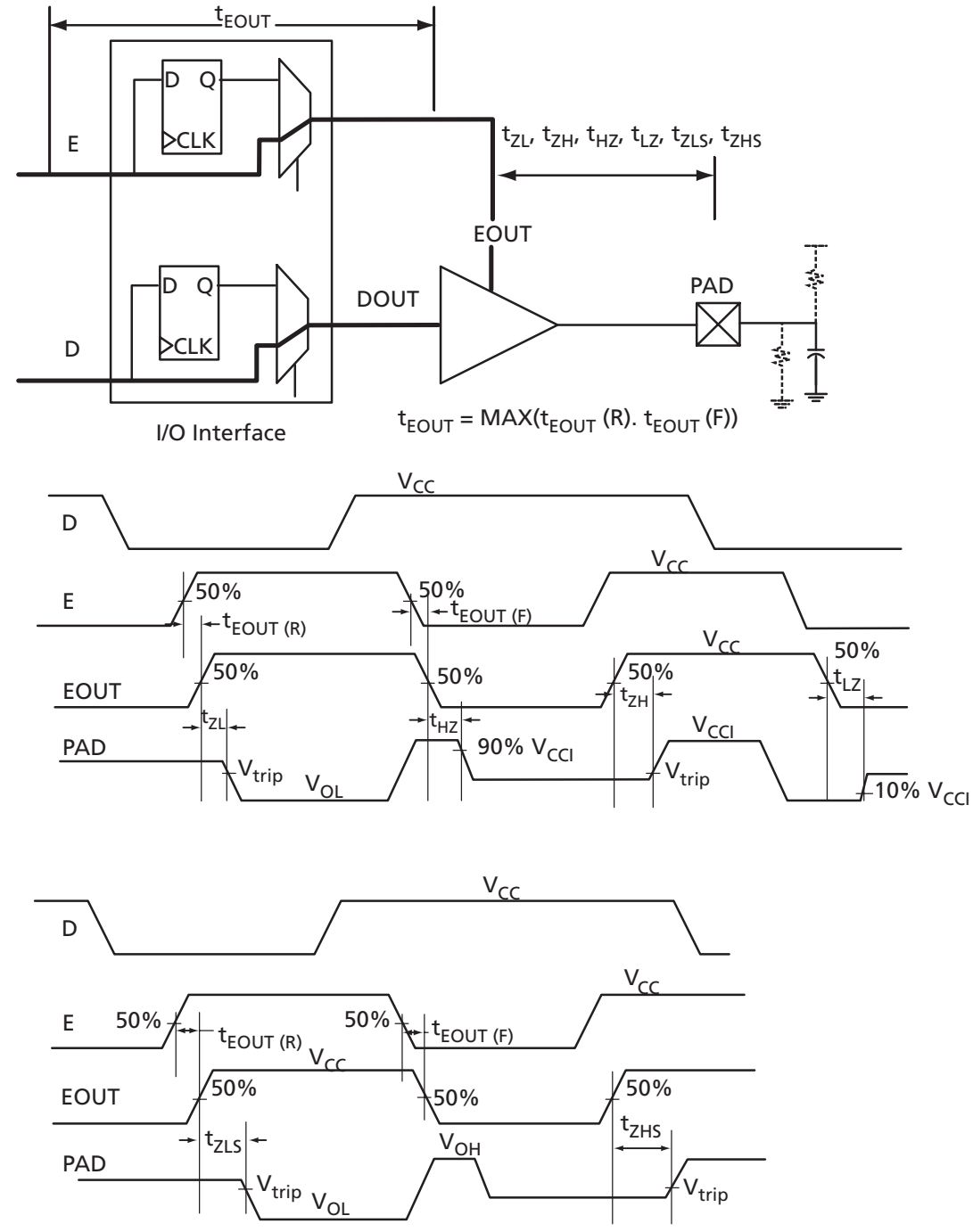


Figure 2-116 • **Tristate Output Buffer Timing Model and Delays (example)**

Overview of I/O Performance

Summary of I/O DC Input and Output Levels – Default I/O Software Settings

Table 2-83 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions Applicable to Pro I/Os

I/O Standard	Drive Strength	Slew Rate	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}
			Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA
3.3 V LVTTL / 3.3 V LVCMOS	12 mA	High	–0.3	0.8	2	3.6	0.4	2.4	12	12
2.5 V LVCMOS	12 mA	High	–0.3	0.7	1.7	3.6	0.7	1.7	12	12
1.8 V LVCMOS	12 mA	High	–0.3	0.35 * V_{CCI}	0.65*V_{CCI}	3.6	0.45	V_{CCI} – 0.45	12	12
1.5 V LVCMOS	12 mA	High	–0.3	0.35 * V_{CCI}	0.65*V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	12	12
3.3 V PCI	Per PCI Specification									
3.3 V PCI-X	Per PCI-X Specification									
3.3 V GTL	25 mA[2]	High	–0.3	V_{REF} – 0.05	V_{REF} + 0.05	3.6	0.4	–	25	25
2.5 V GTL	25 mA[2]	High	–0.3	V_{REF} – 0.05	V_{REF} + 0.05	3.6	0.4	–	25	25
3.3 V GTL+	35 mA	High	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.6	–	51	51
2.5 V GTL+	33 mA	High	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.6	–	40	40
HSTL (I)	8 mA	High	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.4	V_{CCI} – 0.4	8	8
HSTL (II)	15 mA[2]	High	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.4	V_{CCI} – 0.4	15	15
SSTL2 (I)	15 mA	High	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.54	V_{CCI} – 0.62	15	15
SSTL2 (II)	18 mA	High	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.35	V_{CCI} – 0.43	18	18
SSTL3 (I)	14 mA	High	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.7	V_{CCI} – 1.1	14	14
SSTL3 (II)	21 mA	High	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.5	V_{CCI} – 0.9	21	21

Notes:

1. *Currents are measured at 85°C junction temperature.*
2. *Output drive strength is below JEDEC specification.*
3. *Output slew rate can be extracted by the IBIS models.*

Table 2-84 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions Applicable to Advanced I/Os

I/O Standard	Drive Strength	Slew Rate	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}
			Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA
3.3 V LVTTL / 3.3 V LVCMOS	12 mA	High	–0.3	0.8	2	3.6	0.4	2.4	12	12
2.5 V LVCMOS	12 mA	High	–0.3	0.7	1.7	2.7	0.7	1.7	12	12
1.8 V LVCMOS	12 mA	High	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	12	12
1.5 V LVCMOS	12 mA	High	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.575	0.25 * V_{CCI}	0.75 * V_{CCI}	12	12
3.3 V PCI	Per PCI specifications									
3.3 V PCI-X	Per PCI-X specifications									

Note: Currents are measured at 85°C junction temperature.

Table 2-85 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions
Applicable to Standard I/Os

I/O Standard	Drive Strength	Slew Rate	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}
			Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA
3.3 V LVTTL / 3.3 V LVCMOS	8 mA	High	–0.3	0.8	2	3.6	0.4	2.4	8	8
2.5 V LVCMOS	8 mA	High	–0.3	0.7	1.7	3.6	0.7	1.7	8	8
1.8 V LVCMOS	4 mA	High	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	4	4
1.5 V LVCMOS	2 mA	High	–0.3	0.35 * V_{CCI}	0.65*V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	2	2

Note: Currents are measured at 85°C junction temperature.

Table 2-86 • Summary of Maximum and Minimum DC Input Levels Applicable to Commercial and Industrial Conditions
Applicable to All I/O Bank Types

DC I/O Standards	Commercial[1]		Industrial[2]	
	I_{IL}[3]	I_{IH}[4]	I_{IL}[3]	I_{IH}[4]
	µA	µA	µA	µA
3.3 V LVTTL / 3.3 V LVCMOS	10	10	15	15
2.5 V LVCMOS	10	10	15	15
1.8 V LVCMOS	10	10	15	15
1.5 V LVCMOS	10	10	15	15
3.3 V PCI	10	10	15	15
3.3 V PCI-X	10	10	15	15
3.3 V GTL	10	10	15	15
2.5 V GTL	10	10	15	15
3.3 V GTL+	10	10	15	15
2.5 V GTL+	10	10	15	15
HSTL (I)	10	10	15	15
HSTL (II)	10	10	15	15
SSTL2 (I)	10	10	15	15
SSTL2 (II)	10	10	15	15
SSTL3 (I)	10	10	15	15
SSTL3 (II)	10	10	15	15

Notes:

1. *Commercial range (0°C < T_J < 85°C)*
2. *Industrial range (–40°C < T_J < 100°C)*
3. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
4. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*

Summary of I/O Timing Characteristics – Default I/O Software Settings

Table 2-87 • **Summary of AC Measuring Points Applicable to All I/O Bank Types**

Standard	Input Reference Voltage (V_{REF_TYP})	Board Termination Voltage (V_{TT_REF})	Measuring Trip Point (V_{trip})
3.3 V LVTTL / 3.3 V LVCMOS	–	–	1.4 V
2.5 V LVCMOS	–	–	1.2 V
1.8 V LVCMOS	–	–	0.90 V
1.5 V LVCMOS	–	–	0.75 V
3.3 V PCI	–	–	0.285 * V_{CCI} (RR) 0.615 * V_{CCI} (FF))
3.3 V PCI-X	–	–	0.285 * V_{CCI} (RR) 0.615 * V_{CCI} (FF)
3.3 V GTL	0.8 V	1.2 V	V_{REF}
2.5 V GTL	0.8 V	1.2 V	V_{REF}
3.3 V GTL+	1.0 V	1.5 V	V_{REF}
2.5 V GTL+	1.0 V	1.5 V	V_{REF}
HSTL (I)	0.75 V	0.75 V	V_{REF}
HSTL (II)	0.75 V	0.75 V	V_{REF}
SSTL2 (I)	1.25 V	1.25 V	V_{REF}
SSTL2 (II)	1.25 V	1.25 V	V_{REF}
SSTL3 (I)	1.5 V	1.485 V	V_{REF}
SSTL3 (II)	1.5 V	1.485 V	V_{REF}
LVDS	–	–	Cross point
LVPECL	–	–	Cross point

Table 2-88 • **I/O AC Parameter Definitions**

Parameter	Definition
t_{DP}	Data to Pad delay through the Output Buffer
t_{PY}	Pad to Data delay through the Input Buffer with Schmitt trigger disabled
t_{DOUT}	Data to Output Buffer delay through the I/O interface
t_{EOUT}	Enable to Output Buffer Tristate Control delay through the I/O interface
t_{DIN}	Input Buffer to Data delay through the I/O interface
t_{PYS}	Pad to Data delay through the Input Buffer with Schmitt trigger enabled
t_{HZ}	Enable to Pad delay through the Output Buffer—HIGH to Z
t_{ZH}	Enable to Pad delay through the Output Buffer—Z to HIGH
t_{LZ}	Enable to Pad delay through the Output Buffer—LOW to Z
t_{ZL}	Enable to Pad delay through the Output Buffer—Z to LOW
t_{ZHS}	Enable to Pad delay through the Output Buffer with delayed enable—Z to HIGH
t_{ZLS}	Enable to Pad delay through the Output Buffer with delayed enable—Z to LOW

Table 2-89 • **Summary of I/O Timing Characteristics – Software Default Settings**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = I/O Standard Dependent
Applicable to Pro I/Os

I/O Standard	Drive Strength (mA)	Slew Rate	Capacitive Load (pF)	External Resistor (Ohm)	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
3.3 V LVTTL/ 3.3 V LVCMOS	12 mA	High	35	–	0.49	2.74	0.03	0.90	1.17	0.32	2.79	2.14	2.45	2.70	4.46	3.81	ns
2.5 V LVCMOS	12 mA	High	35	–	0.49	2.80	0.03	1.13	1.24	0.32	2.85	2.61	2.51	2.61	4.52	4.28	ns
1.8 V LVCMOS	12 mA	High	35	–	0.49	2.83	0.03	1.08	1.42	0.32	2.89	2.31	2.79	3.16	4.56	3.98	ns
1.5 V LVCMOS	12 mA	High	35	–	0.49	3.30	0.03	1.27	1.60	0.32	3.36	2.70	2.96	3.27	5.03	4.37	ns
3.3 V PCI	Per PCI spec	High	10	25 [2]	0.49	2.09	0.03	0.78	1.25	0.32	2.13	1.49	2.45	2.70	3.80	3.16	ns
3.3 V PCI-X	Per PCI-X spec	High	10	25 [2]	0.49	2.09	0.03	0.77	1.17	0.32	2.13	1.49	2.45	2.70	3.80	3.16	ns
3.3 V GTL	25 mA	High	10	25	0.49	1.55	0.03	2.19	–	0.32	1.52	1.55	0.00	0.00	3.19	3.22	ns
2.5 V GTL	25 mA	High	10	25	0.49	1.59	0.03	1.83	–	0.32	1.61	1.59	0.00	0.00	3.28	3.26	ns
3.3 V GTL+	35 mA	High	10	25	0.49	1.53	0.03	1.19	–	0.32	1.56	1.53	0.00	0.00	3.23	3.20	ns
2.5 V GTL+	33 mA	High	10	25	0.49	1.65	0.03	1.13	–	0.32	1.68	1.57	0.00	0.00	3.35	3.24	ns
HSTL (I)	8 mA	High	20	50	0.49	2.37	0.03	1.59	–	0.32	2.42	2.35	0.00	0.00	4.09	4.02	ns
HSTL (II)	15 mA	High	20	25	0.49	2.26	0.03	1.59	–	0.32	2.30	2.03	0.00	0.00	3.97	3.70	ns
SSTL2 (I)	17 mA	High	30	50	0.49	1.59	0.03	1.00	–	0.32	1.62	1.38	0.00	0.00	3.29	3.05	ns
SSTL2 (II)	21 mA	High	30	25	0.49	1.62	0.03	1.00	–	0.32	1.65	1.32	0.00	0.00	3.32	2.99	ns
SSTL3 (I)	16 mA	High	30	50	0.49	1.72	0.03	0.93	–	0.32	1.75	1.37	0.00	0.00	3.42	3.04	ns
SSTL3 (II)	24 mA	High	30	25	0.49	1.54	0.03	0.93	–	0.32	1.57	1.25	0.00	0.00	3.24	2.92	ns
LVDS	24 mA	High	–	–	0.49	1.57	0.03	1.36	–	–	–	–	–	–	–	–	ns
LVPECL	24 mA	High	–	–	0.49	1.60	0.03	1.22	–	–	–	–	–	–	–	–	ns

Notes:

1. *For specific junction temperature and voltage-supply levels, refer to Table 3-6 on page 3-7 for derating values.*
2. *Resistance is used to measure I/O propagation delays as defined in PCI specifications. See Figure 2-121 on page 2-198 for connectivity. This resistor is not required during normal operation.*

Table 2-90 • Summary of I/O Timing Characteristics – Software Default Settings
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = I/O Standard Dependent
Applicable to Advanced I/Os

I/O Standard	Drive Strength (mA)	Slew Rate	Capacitive Load (pF)	External Resistor (Ohm)	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
3.3 V LVTTL/ 3.3 V LVCMOS	12 mA	High	35 pF	–	0.49	2.64	0.03	0.90	0.32	2.69	2.11	2.40	2.68	4.36	3.78	ns
2.5 V LVCMOS	12 mA	High	35 pF	–	0.49	2.66	0.03	0.98	0.32	2.71	2.56	2.47	2.57	4.38	4.23	ns
1.8 V LVCMOS	12 mA	High	3 5pF	–	0.49	2.64	0.03	0.91	0.32	2.69	2.27	2.76	3.05	4.36	3.94	ns
1.5 V LVCMOS	12 mA	High	35 pF	–	0.49	3.05	0.03	1.07	0.32	3.10	2.67	2.95	3.14	4.77	4.34	ns
3.3 V PCI	Per PCI spec	High	10 pF	25 [2]	0.49	2.00	0.03	0.65	0.32	2.04	1.46	2.40	2.68	3.71	3.13	ns
3.3 V PCI-X	Per PCI-X spec	High	10 pF	25 [2]	0.49	2.00	0.03	0.62	0.32	2.04	1.46	2.40	2.68	3.71	3.13	ns
LVDS	24 mA	High	–	–	0.49	1.37	0.03	1.20	N/A	N/A	N/A	N/A	N/A	N/A	N/A	ns
LVPECL	24 mA	High	–	–	0.49	1.34	0.03	1.05	N/A	N/A	N/A	N/A	N/A	N/A	N/A	ns

Notes:

1. *For specific junction temperature and voltage-supply levels, refer to Table 3-6 on page 3-7 for derating values.*
2. *Resistance is used to measure I/O propagation delays as defined in PCI specifications. See Figure 2-121 on page 2-198 for connectivity. This resistor is not required during normal operation.*

Table 2-91 • Summary of I/O Timing Characteristics – Software Default Settings
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = I/O Standard Dependent
Applicable to Standard I/Os

I/O Standard	Drive Strength (mA)	Slew Rate	Capacitive Load (pF)	External Resistor (Ohm)	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
3.3 V LVTTL/ 3.3 V LVCMOS	8 mA	High	35 pF	–	0.49	3.29	0.03	0.75	0.32	3.36	2.80	1.79	2.01	ns
2.5 V LVCMOS	8 mA	High	35pF	–	0.49	3.56	0.03	0.96	0.32	3.40	3.56	1.78	1.91	ns
1.8 V LVCMOS	4 mA	High	35pF	–	0.49	4.74	0.03	0.90	0.32	4.02	4.74	1.80	1.85	ns
1.5 V LVCMOS	2 mA	High	35pF	–	0.49	5.71	0.03	1.06	0.32	4.71	5.71	1.83	1.83	ns

Note: For specific junction temperature and voltage-supply levels, refer to Table 3-6 on page 3-7 for derating values.

Detailed I/O DC Characteristics

Table 2-92 • Input Capacitance

Symbol	Definition	Conditions	Min.	Max.	Units
C_{IN}	Input capacitance	V_{IN} = 0, f = 1.0 MHz		8	pF
C_{INCLK}	Input capacitance on the clock pin	V_{IN} = 0, f = 1.0 MHz		8	pF

Table 2-93 • I/O Output Buffer Maximum Resistances [1]

Standard	Drive Strength	$R_{PULL\text{-}DOWN}$ (ohms) [2]	$R_{PULL\text{-}UP}$ (ohms) [3]
Applicable to Pro I/O Banks			
3.3 V LVTTL / 3.3 V LVCMOS	4 mA	100	300
	8 mA	50	150
	12 mA	25	75
	16 mA	17	50
	24 mA	11	33
2.5 V LVCMOS	4 mA	100	200
	8 mA	50	100
	12 mA	25	50
	16 mA	20	40
	24 mA	11	22
1.8 V LVCMOS	2 mA	200	225
	4 mA	100	112
	6 mA	50	56
	8 mA	50	56
	12 mA	20	22
	16 mA	20	22
1.5 V LVCMOS	2 mA	200	224
	4 mA	100	112
	6 mA	67	75
	8 mA	33	37
	12 mA	33	37
3.3 V PCI/PCI-X	Per PCI/PCI-X specification	25	75
3.3 V GTL	25 mA	11	–
2.5 V GTL	25 mA	14	–
3.3 V GTL+	35 mA	12	–

Notes:

1. *These maximum values are provided for informational reasons only. Minimum output buffer resistance values depend on V_{CC}, drive strength selection, temperature, and process. For board design considerations and detailed output buffer resistances, use the corresponding IBIS models located on the Actel website at http://www.actel.com/techdocs/models/ibis.html.*
2. $R_{(PULL\text{-}DOWN\text{-}MAX)} = V_{OLspec} / I_{OLspec}$
3. $R_{(PULL\text{-}UP\text{-}MAX)} = (V_{CCImax} - V_{OHspec}) / I_{OHspec}$

Table 2-93 • **I/O Output Buffer Maximum Resistances [1] (continued)**

Standard	Drive Strength	$R_{PULL\text{-}DOWN}$ (ohms) [2]	$R_{PULL\text{-}UP}$ (ohms) [3]
2.5 V GTL+	33 mA	15	–
HSTL (I)	8 mA	50	50
HSTL (II)	15 mA	25	25
SSTL2 (I)	17 mA	27	31
SSTL2 (II)	21 mA	13	15
SSTL3 (I)	16 mA	44	69
SSTL3 (II)	24 mA	18	32
Applicable to Advanced I/O Banks			
3.3 V LVTTL / 3.3 V LVCMOS	2 mA	100	300
	4 mA	100	300
	6 mA	50	150
	8 mA	50	150
	12 mA	25	75
	16 mA	17	50
	24 mA	11	33
2.5 V LVCMOS	2 mA	100	200
	4 mA	100	200
	6 mA	50	100
	8 mA	50	100
	12 mA	25	50
	16 mA	20	40
	24 mA	11	22
1.8 V LVCMOS	2 mA	200	225
	4 mA	100	112
	6 mA	50	56
	8 mA	50	56
	12 mA	20	22
	16 mA	20	22

Notes:

1. *These maximum values are provided for informational reasons only. Minimum output buffer resistance values depend on V_{CC}, drive strength selection, temperature, and process. For board design considerations and detailed output buffer resistances, use the corresponding IBIS models located on the Actel website at http://www.actel.com/techdocs/models/ibis.html.*
2. $R_{(PULL\text{-}DOWN\text{-}MAX)} = V_{OLspec} / I_{OLspec}$
3. $R_{(PULL\text{-}UP\text{-}MAX)} = (V_{CCImax} – V_{OHspec}) / I_{OHspec}$

Table 2-93 • I/O Output Buffer Maximum Resistances [1] (continued)

Standard	Drive Strength	$R_{PULL\text{-}DOWN}$ (ohms) [2]	$R_{PULL\text{-}UP}$ (ohms) [3]
1.5 V LVCMOS	2 mA	200	224
	4 mA	100	112
	6 mA	67	75
	8 mA	33	37
	12 mA	33	37
3.3 V PCI/PCI-X	Per PCI/PCI-X specification	25	75
Applicable to Standard I/O Banks			
3.3 V LVTTL / 3.3 V LVCMOS	2 mA	100	300
	4 mA	100	300
	6 mA	50	150
	8 mA	50	150
2.5 V LVCMOS	2 mA	100	200
	4 mA	100	200
	6 mA	50	100
	8 mA	50	100
1.8 V LVCMOS	2 mA	200	225
	4 mA	100	112
1.5 V LVCMOS	2 mA	200	224

Notes:

1. *These maximum values are provided for informational reasons only. Minimum output buffer resistance values depend on V_{CC}, drive strength selection, temperature, and process. For board design considerations and detailed output buffer resistances, use the corresponding IBIS models located on the Actel website at http://www.actel.com/techdocs/models/ibis.html.*
2. $R_{(PULL\text{-}DOWN\text{-}MAX)} = V_{OLspec} / I_{OLspec}$
3. $R_{(PULL\text{-}UP\text{-}MAX)} = (V_{CCImax} - V_{OHspec}) / I_{OHspec}$

Table 2-94 • I/O Weak Pull-Up/Pull-Down Resistances
Minimum and Maximum Weak Pull-Up/Pull-Down Resistance Values

	$R_{(WEAK\ PULL\text{-}UP)}$ [1] (ohms)		$R_{(WEAK\ PULL\text{-}DOWN)}$ [2] (ohms)	
V_{CCI}	Min.	Max.	Min.	Max.
3.3 V	10 k	45 k	10 k	45 k
2.5 V	11 k	55 k	12 k	74 k
1.8 V	18 k	70 k	17 k	110 k
1.5 V	19 k	90 k	19 k	140 k

Notes:

1. $R_{(WEAK\ PULL\text{-}DOWN\text{-}MAX)} = V_{OLspec} / I_{WEAK\ PULL\text{-}DOWN\text{-}MIN}$
2. $R_{(WEAK\ PULL\text{-}UP\text{-}MAX)} = (V_{CCImax} - V_{OHspec}) / I_{WEAK\ PULL\text{-}UP\text{-}MIN}$

Table 2-95 • **I/O Short Currents I_{OSH}/I_{OSL}**

	Drive Strength	I_{OSH} (mA)*	I_{OSL} (mA)*
Applicable to Pro I/O Banks			
3.3 V LVTTL / 3.3 V LVCMOS	4 mA	25	27
	8 mA	51	54
	12 mA	103	109
	16 mA	132	127
	24 mA	268	181
2.5 V LVCMOS	4 mA	16	18
	8 mA	32	37
	12 mA	65	74
	16 mA	83	87
	24 mA	169	124
1.8 V LVCMOS	2 mA	9	11
	4 mA	17	22
	6 mA	35	44
	8 mA	45	51
	12 mA	91	74
	16 mA	91	74
1.5 V LVCMOS	2 mA	13	16
	4 mA	25	33
	6 mA	32	39
	8 mA	66	55
	12 mA	66	55
Applicable to Advanced I/O Banks			
3.3 V LVTTL / 3.3 V LVCMOS	2 mA	25	27
	4 mA	25	27
	6 mA	51	54
	8 mA	51	54
	12 mA	103	109
	16 mA	132	127
	24 mA	268	181
3.3 V LVCMOS	2 mA	25	27
	4 mA	25	27
	6 mA	51	54
	8 mA	51	54
	12 mA	103	109
	16 mA	132	127
	24 mA	268	181

*Note: *T_J = 100°C*

Table 2-95 • **I/O Short Currents I_{OSH}/I_{OSL} (continued)**

	Drive Strength	I_{OSH} (mA)*	I_{OSL} (mA)*
2.5 V LVCMOS	2 mA	16	18
	4 mA	16	18
	6 mA	32	37
	8 mA	32	37
	12 mA	65	74
	16 mA	83	87
	24 mA	169	124
1.8 V LVCMOS	2 mA	9	11
	4 mA	17	22
	6 mA	35	44
	8 mA	45	51
	12 mA	91	74
	16 mA	91	74
1.5 V LVCMOS	2 mA	13	16
	4 mA	25	33
	6 mA	32	39
	8 mA	66	55
	12 mA	66	55
3.3 V PCI/PCI-X	Per PCI/PCI-X specification	103	109
Applicable to Standard I/O Banks			
3.3 V LVTTL / 3.3 V LVCMOS	2 mA	25	27
	4 mA	25	27
	6 mA	51	54
	8 mA	51	54
2.5 V LVCMOS	2 mA	16	18
	4 mA	16	18
	6 mA	32	37
	8 mA	32	37
1.8 V LVCMOS	2 mA	9	11
	4 mA	17	22
1.5 V LVCMOS	2 mA	13	16

*Note: *T_J = 100°C*

The length of time an I/O can withstand I_{OSH}/I_{OSL} events depends on the junction temperature. The reliability data below is based on a 3.3 V, 36 mA I/O setting, which is the worst case for this type of analysis.

For example, at 100°C, the short current condition would have to be sustained for more than six months to cause a reliability concern. The I/O design does not contain any short circuit protection, but such protection would only be needed in extremely prolonged stress conditions.

Table 2-96 • **Short Current Event Duration before Failure**

Temperature	Time before Failure
–40°C	>20 years
0°C	>20 years
25°C	>20 years
70°C	5 years
85°C	2 years
100°C	6 months

Table 2-97 • **Schmitt Trigger Input Hysteresis**
Hysteresis Voltage Value (typ.) for Schmitt Mode Input Buffers

Input Buffer Configuration	Hysteresis Value (typ.)
3.3 V LVTTL/LVCMOS/PCI/PCI-X (Schmitt trigger mode)	240 mV
2.5 V LVCMOS (Schmitt trigger mode)	140 mV
1.8 V LVCMOS (Schmitt trigger mode)	80 mV
1.5 V LVCMOS (Schmitt trigger mode)	60 mV

Table 2-98 • **I/O Input Rise Time, Fall Time, and Related I/O Reliability**

Input Buffer	Input Rise/Fall Time (min.)	Input Rise/Fall Time (max.)	Reliability
LVTTL/LVCMOS (Schmitt trigger disabled)	No requirement	10 ns*	20 years (100°C)
LVTTL/LVCMOS (Schmitt trigger enabled)	No requirement	No requirement, but input noise voltage cannot exceed Schmitt hysteresis	20 years (100°C)
HSTL/SSTL/GTL	No requirement	10 ns*	10 years (100°C)
LVDS/BLVDS/M-LVDS/LVPECL	No requirement	10 ns*	10 years (100°C)

*Note: *The maximum input rise/fall time is related only to the noise induced into the input buffer trace. If the noise is low, the rise time and fall time of input buffers, when Schmitt trigger is disabled, can be increased beyond the maximum value. The longer the rise/fall times, the more susceptible the input signal is to the board noise. Actel recommends signal integrity evaluation/characterization of the system to ensure there is no excessive noise coupling into input signals.*

Single-Ended I/O Characteristics

3.3 V LVTTL / 3.3 V LVCMOS

Low-Voltage Transistor–Transistor Logic is a general-purpose standard (EIA/JESD) for 3.3 V applications. It uses an LVTTL input buffer and push-pull output buffer. The 3.3 V LVCMOS standard is supported as part of the 3.3 V LVTTL support.

Table 2-99 • Minimum and Maximum DC Input and Output Levels

3.3 V LVTTL / 3.3 V LVCMOS	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
Applicable to Pro I/O Banks												
4 mA	–0.3	0.8	2	3.6	0.4	2.4	4	4	27	25	10	10
8 mA	–0.3	0.8	2	3.6	0.4	2.4	8	8	54	51	10	10
12 mA	–0.3	0.8	2	3.6	0.4	2.4	12	12	109	103	10	10
16 mA	–0.3	0.8	2	3.6	0.4	2.4	16	16	127	132	10	10
24 mA	–0.3	0.8	2	3.6	0.4	2.4	24	24	181	268	10	10
Applicable to Advanced I/O Banks												
2 mA	–0.3	0.8	2	3.6	0.4	2.4	2	2	27	25	10	10
4 mA	–0.3	0.8	2	3.6	0.4	2.4	4	4	27	25	10	10
6 mA	–0.3	0.8	2	3.6	0.4	2.4	6	6	54	51	10	10
8 mA	–0.3	0.8	2	3.6	0.4	2.4	8	8	54	51	10	10
12 mA	–0.3	0.8	2	3.6	0.4	2.4	12	12	109	103	10	10
16 mA	–0.3	0.8	2	3.6	0.4	2.4	16	16	127	132	10	10
24 mA	–0.3	0.8	2	3.6	0.4	2.4	24	24	181	268	10	10
Applicable to Standard I/O Banks												
2 mA	–0.3	0.8	2	3.6	0.4	2.4	2	2	27	25	10	10
4 mA	–0.3	0.8	2	3.6	0.4	2.4	4	4	27	25	10	10
6 mA	–0.3	0.8	2	3.6	0.4	2.4	6	6	54	51	10	10
8 mA	–0.3	0.8	2	3.6	0.4	2.4	8	8	54	51	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*
5. *Software default selection highlighted in gray.*

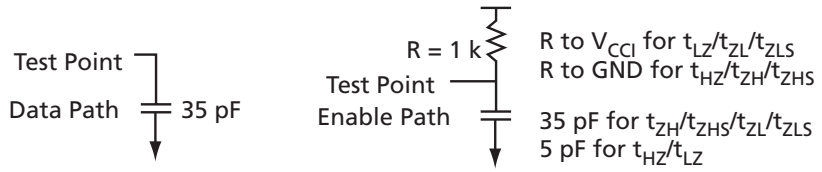


Figure 2-117 • AC Loading

Table 2-100 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	C_{LOAD} (pF)
0	3.3	1.4	–	35

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-101 • 3.3 V LVTTL / 3.3 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = 3.0 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.66	11.01	0.04	1.20	1.57	0.43	11.21	9.05	2.69	2.44	13.45	11.29	ns
	–1	0.56	9.36	0.04	1.02	1.33	0.36	9.54	7.70	2.29	2.08	11.44	9.60	ns
	–2	0.49	8.22	0.03	0.90	1.17	0.32	8.37	6.76	2.01	1.82	10.04	8.43	ns
8 mA	Std.	0.66	7.86	0.04	1.20	1.57	0.43	8.01	6.44	3.04	3.06	10.24	8.68	ns
	–1	0.56	6.69	0.04	1.02	1.33	0.36	6.81	5.48	2.58	2.61	8.71	7.38	ns
	–2	0.49	5.87	0.03	0.90	1.17	0.32	5.98	4.81	2.27	2.29	7.65	6.48	ns
12 mA	Std.	0.66	6.03	0.04	1.20	1.57	0.43	6.14	5.02	3.28	3.47	8.37	7.26	ns
	–1	0.56	5.13	0.04	1.02	1.33	0.36	5.22	4.27	2.79	2.95	7.12	6.17	ns
	–2	0.49	4.50	0.03	0.90	1.17	0.32	4.58	3.75	2.45	2.59	6.25	5.42	ns
16 mA	Std.	0.66	5.62	0.04	1.20	1.57	0.43	5.72	4.72	3.32	3.58	7.96	6.96	ns
	–1	0.56	4.78	0.04	1.02	1.33	0.36	4.87	4.02	2.83	3.04	6.77	5.92	ns
	–2	0.49	4.20	0.03	0.90	1.17	0.32	4.27	3.53	2.48	2.67	5.94	5.20	ns
24 mA	Std.	0.66	5.24	0.04	1.20	1.57	0.43	5.34	4.69	3.39	3.96	7.58	6.93	ns
	–1	0.56	4.46	0.04	1.02	1.33	0.36	4.54	3.99	2.88	3.37	6.44	5.89	ns
	–2	0.49	3.92	0.03	0.90	1.17	0.32	3.99	3.50	2.53	2.96	5.66	5.17	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-102 • 3.3 V LVTTL / 3.3 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.66	7.88	0.04	1.20	1.57	0.43	8.03	6.70	2.69	2.59	10.26	8.94	ns
	–1	0.56	6.71	0.04	1.02	1.33	0.36	6.83	5.70	2.29	2.20	8.73	7.60	ns
	–2	0.49	5.89	0.03	0.90	1.17	0.32	6.00	5.01	2.01	1.93	7.67	6.67	ns
8 mA	Std.	0.66	5.08	0.04	1.20	1.57	0.43	5.17	4.14	3.05	3.21	7.41	6.38	ns
	–1	0.56	4.32	0.04	1.02	1.33	0.36	4.40	3.52	2.59	2.73	6.30	5.43	ns
	–2	0.49	3.79	0.03	0.90	1.17	0.32	3.86	3.09	2.28	2.40	5.53	4.76	ns
12 mA	Std.	0.66	3.67	0.04	1.20	1.57	0.43	3.74	2.87	3.28	3.61	5.97	5.11	ns
	–1	0.56	3.12	0.04	1.02	1.33	0.36	3.18	2.44	2.79	3.07	5.08	4.34	ns
	–2	0.49	2.74	0.03	0.90	1.17	0.32	2.79	2.14	2.45	2.70	4.46	3.81	ns
16 mA	Std.	0.66	3.46	0.04	1.20	1.57	0.43	3.53	2.61	3.33	3.72	5.76	4.84	ns
	–1	0.56	2.95	0.04	1.02	1.33	0.36	3.00	2.22	2.83	3.17	4.90	4.12	ns
	–2	0.49	2.59	0.03	0.90	1.17	0.32	2.63	1.95	2.49	2.78	4.30	3.62	ns
24 mA	Std.	0.66	3.21	0.04	1.20	1.57	0.43	3.27	2.16	3.39	4.13	5.50	4.39	ns
	–1	0.56	2.73	0.04	1.02	1.33	0.36	2.78	1.83	2.88	3.51	4.68	3.74	ns
	–2	0.49	2.39	0.03	0.90	1.17	0.32	2.44	1.61	2.53	3.08	4.11	3.28	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

***Table 2-103* • 3.3 V LVTTL / 3.3 V LVCMOS Low Slew**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = 3.0 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.66	10.26	0.04	1.20	0.43	10.45	8.90	2.64	2.46	12.68	11.13	ns
	–1	0.56	8.72	0.04	1.02	0.36	8.89	7.57	2.25	2.09	10.79	9.47	ns
	–2	0.49	7.66	0.03	0.90	0.32	7.80	6.64	1.98	1.83	9.47	8.31	ns
8 mA	Std.	0.66	7.27	0.04	1.20	0.43	7.41	6.28	2.98	3.04	9.65	8.52	ns
	–1	0.56	6.19	0.04	1.02	0.36	6.30	5.35	2.54	2.59	8.20	7.25	ns
	–2	0.49	5.43	0.03	0.90	0.32	5.53	4.69	2.23	2.27	7.20	6.36	ns
12 mA	Std.	0.66	5.58	0.04	1.20	0.43	5.68	4.87	3.21	3.42	7.92	7.11	ns
	–1	0.56	4.75	0.04	1.02	0.36	4.84	4.14	2.73	2.91	6.74	6.05	ns
	–2	0.49	4.17	0.03	0.90	0.32	4.24	3.64	2.39	2.55	5.91	5.31	ns
16 mA	Std.	0.66	5.21	0.04	1.20	0.43	5.30	4.56	3.26	3.51	7.54	6.80	ns
	–1	0.56	4.43	0.04	1.02	0.36	4.51	3.88	2.77	2.99	6.41	5.79	ns
	–2	0.49	3.89	0.03	0.90	0.32	3.96	3.41	2.43	2.62	5.63	5.08	ns
24 mA	Std.	0.66	4.85	0.04	1.20	0.43	4.94	4.54	3.32	3.88	7.18	6.78	ns
	–1	0.56	4.13	0.04	1.02	0.36	4.20	3.87	2.82	3.30	6.10	5.77	ns
	–2	0.49	3.62	0.03	0.90	0.32	3.69	3.39	2.48	2.90	5.36	5.06	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-104 • 3.3 V LVTTL / 3.3 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.66	7.66	0.04	1.20	0.43	7.80	6.59	2.65	2.61	10.03	8.82	ns
	–1	0.56	6.51	0.04	1.02	0.36	6.63	5.60	2.25	2.22	8.54	7.51	ns
	–2	0.49	5.72	0.03	0.90	0.32	5.82	4.92	1.98	1.95	7.49	6.59	ns
8 mA	Std.	0.66	4.91	0.04	1.20	0.43	5.00	4.07	2.99	3.20	7.23	6.31	ns
	–1	0.56	4.17	0.04	1.02	0.36	4.25	3.46	2.54	2.73	6.15	5.36	ns
	–2	0.49	3.66	0.03	0.90	0.32	3.73	3.04	2.23	2.39	5.40	4.71	ns
12 mA	Std.	0.66	3.53	0.04	1.20	0.43	3.60	2.82	3.21	3.58	5.83	5.06	ns
	–1	0.56	3.00	0.04	1.02	0.36	3.06	2.40	2.73	3.05	4.96	4.30	ns
	–2	0.49	2.64	0.03	0.90	0.32	2.69	2.11	2.40	2.68	4.36	3.78	ns
16 mA	Std.	0.66	3.33	0.04	1.20	0.43	3.39	2.56	3.26	3.68	5.63	4.80	ns
	–1	0.56	2.83	0.04	1.02	0.36	2.89	2.18	2.77	3.13	4.79	4.08	ns
	–2	0.49	2.49	0.03	0.90	0.32	2.53	1.91	2.44	2.75	4.20	3.58	ns
24 mA	Std.	0.66	3.08	0.04	1.20	0.43	3.13	2.12	3.32	4.06	5.37	4.35	ns
	–1	0.56	2.62	0.04	1.02	0.36	2.66	1.80	2.83	3.45	4.57	3.70	ns
	–2	0.49	2.30	0.03	0.90	0.32	2.34	1.58	2.48	3.03	4.01	3.25	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-105 • 3.3 V LVTTL / 3.3 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	9.46	0.04	1.00	0.43	9.64	8.54	2.07	2.04	ns
	–1	0.56	8.05	0.04	0.85	0.36	8.20	7.27	1.76	1.73	ns
	–2	0.49	7.07	0.03	0.75	0.32	7.20	6.38	1.55	1.52	ns
4 mA	Std.	0.66	9.46	0.04	1.00	0.43	9.64	8.54	2.07	2.04	ns
	–1	0.56	8.05	0.04	0.85	0.36	8.20	7.27	1.76	1.73	ns
	–2	0.49	7.07	0.03	0.75	0.32	7.20	6.38	1.55	1.52	ns
6 mA	Std.	0.66	6.57	0.04	1.00	0.43	6.69	5.98	2.40	2.57	ns
	–1	0.56	5.59	0.04	0.85	0.36	5.69	5.09	2.04	2.19	ns
	–2	0.49	4.91	0.03	0.75	0.32	5.00	4.47	1.79	1.92	ns
8 mA	Std.	0.66	6.57	0.04	1.00	0.43	6.69	5.98	2.40	2.57	ns
	–1	0.56	5.59	0.04	0.85	0.36	5.69	5.09	2.04	2.19	ns
	–2	0.49	4.91	0.03	0.75	0.32	5.00	4.47	1.79	1.92	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-106 • 3.3 V LVTTL / 3.3 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	7.07	0.04	1.00	0.43	7.20	6.23	2.07	2.15	ns
	–1	0.56	6.01	0.04	0.85	0.36	6.12	5.30	1.76	1.83	ns
	–2 [2]	0.49	5.28	0.03	0.75	0.32	5.37	4.65	1.55	1.60	ns
4 mA	Std.	0.66	7.07	0.04	1.00	0.43	7.20	6.23	2.07	2.15	ns
	–1	0.56	6.01	0.04	0.85	0.36	6.12	5.30	1.76	1.83	ns
	–2	0.49	5.28	0.03	0.75	0.32	5.37	4.65	1.55	1.60	ns
6 mA	Std.	0.66	4.41	0.04	1.00	0.43	4.49	3.75	2.39	2.69	ns
	–1	0.56	3.75	0.04	0.85	0.36	3.82	3.19	2.04	2.29	ns
	–2	0.49	3.29	0.03	0.75	0.32	3.36	2.80	1.79	2.01	ns
8 mA	Std.	0.66	4.41	0.04	1.00	0.43	4.49	3.75	2.39	2.69	ns
	–1	0.56	3.75	0.04	0.85	0.36	3.82	3.19	2.04	2.29	ns
	–2	0.49	3.29	0.03	0.75	0.32	3.36	2.80	1.79	2.01	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

2.5 V LVCMOS

Low-Voltage CMOS for 2.5 V is an extension of the LVCMOS standard (JESD8-5) used for general-purpose 2.5 V applications. It uses a 5 V–tolerant input buffer and push-pull output buffer.

***Table 2-107* • Minimum and Maximum DC Input and Output Levels**

2.5 V LVCMOS	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
Applicable to Pro I/O Banks												
4 mA	–0.3	0.7	1.7	3.6	0.7	1.7	4	4	18	16	10	10
8 mA	–0.3	0.7	1.7	3.6	0.7	1.7	8	8	37	32	10	10
12 mA	–0.3	0.7	1.7	3.6	0.7	1.7	12	12	74	65	10	10
16 mA	–0.3	0.7	1.7	3.6	0.7	1.7	16	16	87	83	10	10
24 mA	–0.3	0.7	1.7	3.6	0.7	1.7	24	24	124	169	10	10
Applicable to Advanced I/O Banks												
2 mA	–0.3	0.7	1.7	2.7	0.7	1.7	2	2	18	16	10	10
4 mA	–0.3	0.7	1.7	2.7	0.7	1.7	4	4	18	16	10	10
6 mA	–0.3	0.7	1.7	2.7	0.7	1.7	6	6	37	32	10	10
8 mA	–0.3	0.7	1.7	2.7	0.7	1.7	8	8	37	32	10	10
12 mA	–0.3	0.7	1.7	2.7	0.7	1.7	12	12	74	65	10	10
16 mA	–0.3	0.7	1.7	2.7	0.7	1.7	16	16	87	83	10	10
24 mA	–0.3	0.7	1.7	2.7	0.7	1.7	24	24	124	169	10	10
Applicable to Standard I/O Banks												
2 mA	–0.3	0.7	1.7	3.6	0.7	1.7	2	2	18	16	10	10
4 mA	–0.3	0.7	1.7	3.6	0.7	1.7	4	4	18	16	10	10
6 mA	–0.3	0.7	1.7	3.6	0.7	1.7	6	6	37	32	10	10
8 mA	–0.3	0.7	1.7	3.6	0.7	1.7	8	8	37	32	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*
5. *Software default selection highlighted in gray.*

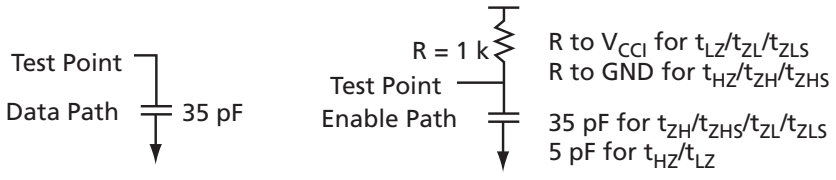


***Figure 2-118* • AC Loading**

***Table 2-108* • AC Waveforms, Measuring Points, and Capacitive Loads**

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	C_{LOAD} (pF)
0	2.5	1.2	–	35

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

***Table 2-109* • 2.5 V LVCMOS Low Slew**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = 2.3 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.60	12.00	0.04	1.51	1.66	0.43	12.23	11.61	2.72	2.20	14.46	13.85	ns
	–1	0.51	10.21	0.04	1.29	1.41	0.36	10.40	9.88	2.31	1.87	12.30	11.78	ns
	–2	0.45	8.96	0.03	1.13	1.24	0.32	9.13	8.67	2.03	1.64	10.80	10.34	ns
8 mA	Std.	0.60	8.73	0.04	1.51	1.66	0.43	8.89	8.01	3.10	2.93	11.13	10.25	ns
	–1	0.51	7.43	0.04	1.29	1.41	0.36	7.57	6.82	2.64	2.49	9.47	8.72	ns
	–2	0.45	6.52	0.03	1.13	1.24	0.32	6.64	5.98	2.32	2.19	8.31	7.65	ns
12 mA	Std.	0.66	6.77	0.04	1.51	1.66	0.43	6.90	6.11	3.37	3.39	9.14	8.34	ns
	–1	0.56	5.76	0.04	1.29	1.41	0.36	5.87	5.20	2.86	2.89	7.77	7.10	ns
	–2	0.49	5.06	0.03	1.13	1.24	0.32	5.15	4.56	2.51	2.53	6.82	6.23	ns
16 mA	Std.	0.66	6.31	0.04	1.51	1.66	0.43	6.42	5.73	3.42	3.52	8.66	7.96	ns
	–1	0.56	5.37	0.04	1.29	1.41	0.36	5.46	4.87	2.91	3.00	7.37	6.77	ns
	–2	0.49	4.71	0.03	1.13	1.24	0.32	4.80	4.28	2.56	2.63	6.47	5.95	ns
24 mA	Std.	0.66	5.93	0.04	1.51	1.66	0.43	6.04	5.70	3.49	4.00	8.28	7.94	ns
	–1	0.56	5.05	0.04	1.29	1.41	0.36	5.14	4.85	2.97	3.40	7.04	6.75	ns
	–2	0.49	4.43	0.03	1.13	1.24	0.32	4.51	4.26	2.61	2.99	6.18	5.93	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-110 • 2.5 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.60	8.82	0.04	1.51	1.66	0.43	8.13	8.82	2.72	2.29	10.37	11.05	ns
	–1	0.51	7.50	0.04	1.29	1.41	0.36	6.92	7.50	2.31	1.95	8.82	9.40	ns
	–2	0.45	6.58	0.03	1.13	1.24	0.32	6.07	6.58	2.03	1.71	7.74	8.25	ns
8 mA	Std.	0.60	5.27	0.04	1.51	1.66	0.43	5.27	5.27	3.10	3.03	7.50	7.51	ns
	–1	0.51	4.48	0.04	1.29	1.41	0.36	4.48	4.48	2.64	2.58	6.38	6.38	ns
	–2	0.45	3.94	0.03	1.13	1.24	0.32	3.93	3.94	2.32	2.26	5.60	5.61	ns
12 mA	Std.	0.66	3.74	0.04	1.51	1.66	0.43	3.81	3.49	3.37	3.49	6.05	5.73	ns
	–1	0.56	3.18	0.04	1.29	1.41	0.36	3.24	2.97	2.86	2.97	5.15	4.87	ns
	–2	0.49	2.80	0.03	1.13	1.24	0.32	2.85	2.61	2.51	2.61	4.52	4.28	ns
16 mA	Std.	0.66	3.53	0.04	1.51	1.66	0.43	3.59	3.12	3.42	3.62	5.83	5.35	ns
	–1	0.56	3.00	0.04	1.29	1.41	0.36	3.06	2.65	2.91	3.08	4.96	4.55	ns
	–2	0.49	2.63	0.03	1.13	1.24	0.32	2.68	2.33	2.56	2.71	4.35	4.00	ns
24 mA	Std.	0.66	3.26	0.04	1.51	1.66	0.43	3.32	2.48	3.49	4.11	5.56	4.72	ns
	–1	0.56	2.77	0.04	1.29	1.41	0.36	2.83	2.11	2.97	3.49	4.73	4.01	ns
	–2	0.49	2.44	0.03	1.13	1.24	0.32	2.48	1.85	2.61	3.07	4.15	3.52	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-111 • 2.5 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.66	11.40	0.04	1.31	0.43	11.22	11.40	2.68	2.20	13.45	13.63	ns
	–1	0.56	9.69	0.04	1.11	0.36	9.54	9.69	2.28	1.88	11.44	11.60	ns
	–2	0.49	8.51	0.03	0.98	0.32	8.38	8.51	2.00	1.65	10.05	10.18	ns
8 mA	Std.	0.66	7.96	0.04	1.31	0.43	8.11	7.81	3.05	2.89	10.34	10.05	ns
	–1	0.56	6.77	0.04	1.11	0.36	6.90	6.65	2.59	2.46	8.80	8.55	ns
	–2	0.49	5.94	0.03	0.98	0.32	6.05	5.84	2.28	2.16	7.72	7.50	ns
12 mA	Std.	0.66	6.18	0.04	1.31	0.43	6.29	5.92	3.30	3.32	8.53	8.15	ns
	–1	0.56	5.26	0.04	1.11	0.36	5.35	5.03	2.81	2.83	7.26	6.94	ns
	–2	0.49	4.61	0.03	0.98	0.32	4.70	4.42	2.47	2.48	6.37	6.09	ns
16 mA	Std.	0.66	6.18	0.04	1.31	0.43	6.29	5.92	3.30	3.32	8.53	8.15	ns
	–1	0.56	5.26	0.04	1.11	0.36	5.35	5.03	2.81	2.83	7.26	6.94	ns
	–2	0.49	4.61	0.03	0.98	0.32	4.70	4.42	2.47	2.48	6.37	6.09	ns
24 mA	Std.	0.66	6.18	0.04	1.31	0.43	6.29	5.92	3.30	3.32	8.53	8.15	ns
	–1	0.56	5.26	0.04	1.11	0.36	5.35	5.03	2.81	2.83	7.26	6.94	ns
	–2	0.49	4.61	0.03	0.98	0.32	4.70	4.42	2.47	2.48	6.37	6.09	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-112 • 2.5 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
4 mA	Std.	0.66	8.66	0.04	1.31	0.43	7.83	8.66	2.68	2.30	10.07	10.90	ns
	–1	0.56	7.37	0.04	1.11	0.36	6.66	7.37	2.28	1.96	8.56	9.27	ns
	–2	0.49	6.47	0.03	0.98	0.32	5.85	6.47	2.00	1.72	7.52	8.14	ns
8 mA	Std.	0.66	5.17	0.04	1.31	0.43	5.04	5.17	3.05	3.00	7.27	7.40	ns
	–1	0.56	4.39	0.04	1.11	0.36	4.28	4.39	2.59	2.55	6.19	6.30	ns
	–2	0.49	3.86	0.03	0.98	0.32	3.76	3.86	2.28	2.24	5.43	5.53	ns
12 mA	Std.	0.66	3.56	0.04	1.31	0.43	3.63	3.43	3.30	3.44	5.86	5.67	ns
	–1	0.56	3.03	0.04	1.11	0.36	3.08	2.92	2.81	2.92	4.99	4.82	ns
	–2	0.49	2.66	0.03	0.98	0.32	2.71	2.56	2.47	2.57	4.38	4.23	ns
16 mA	Std.	0.66	3.35	0.04	1.31	0.43	3.41	3.06	3.36	3.55	5.65	5.30	ns
	–1	0.56	2.85	0.04	1.11	0.36	2.90	2.60	2.86	3.02	4.81	4.51	ns
	–2	0.49	2.50	0.03	0.98	0.32	2.55	2.29	2.51	2.65	4.22	3.96	ns
24 mA	Std.	0.66	3.56	0.04	1.31	0.43	3.63	3.43	3.30	3.44	5.86	5.67	ns
	–1	0.56	3.03	0.04	1.11	0.36	3.08	2.92	2.81	2.92	4.99	4.82	ns
	–2	0.49	2.66	0.03	0.98	0.32	2.71	2.56	2.47	2.57	4.38	4.23	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-113 • 2.5 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	11.00	0.04	1.29	0.43	10.37	11.00	2.03	1.83	ns
	–1	0.56	9.35	0.04	1.10	0.36	8.83	9.35	1.73	1.56	ns
	–2	0.49	8.21	0.03	0.96	0.32	7.75	8.21	1.52	1.37	ns
4 mA	Std.	0.66	11.00	0.04	1.29	0.43	10.37	11.00	2.03	1.83	ns
	–1	0.56	9.35	0.04	1.10	0.36	8.83	9.35	1.73	1.56	ns
	–2	0.49	8.21	0.03	0.96	0.32	7.75	8.21	1.52	1.37	ns
6 mA	Std.	0.66	7.50	0.04	1.29	0.43	7.36	7.50	2.39	2.46	ns
	–1	0.56	6.38	0.04	1.10	0.36	6.26	6.38	2.03	2.10	ns
	–2	0.49	5.60	0.03	0.96	0.32	5.49	5.60	1.78	1.84	ns
8 mA	Std.	0.66	7.50	0.04	1.29	0.43	7.36	7.50	2.39	2.46	ns
	–1	0.56	6.38	0.04	1.10	0.36	6.26	6.38	2.03	2.10	ns
	–2	0.49	5.60	0.03	0.96	0.32	5.49	5.60	1.78	1.84	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-114 • 2.5 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	8.20	0.04	1.29	0.43	7.24	8.20	2.03	1.91	ns
	–1	0.56	6.98	0.04	1.10	0.36	6.16	6.98	1.73	1.62	ns
	–2	0.49	6.13	0.03	0.96	0.32	5.41	6.13	1.52	1.43	ns
4 mA	Std.	0.66	8.20	0.04	1.29	0.43	7.24	8.20	2.03	1.91	ns
	–1	0.56	6.98	0.04	1.10	0.36	6.16	6.98	1.73	1.62	ns
	–2	0.49	6.13	0.03	0.96	0.32	5.41	6.13	1.52	1.43	ns
6 mA	Std.	0.66	4.77	0.04	1.29	0.43	4.55	4.77	2.38	2.55	ns
	–1	0.56	4.05	0.04	1.10	0.36	3.87	4.05	2.03	2.17	ns
	–2	0.49	3.56	0.03	0.96	0.32	3.40	3.56	1.78	1.91	ns
8 mA	Std.	0.66	4.77	0.04	1.29	0.43	4.55	4.77	2.38	2.55	ns
	–1	0.56	4.05	0.04	1.10	0.36	3.87	4.05	2.03	2.17	ns
	–2	0.49	3.56	0.03	0.96	0.32	3.40	3.56	1.78	1.91	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

1.8 V LVCMOS

Low-Voltage CMOS for 1.8 V is an extension of the LVCMOS standard (JESD8-5) used for general-purpose 1.8 V applications. It uses a 1.8 V input buffer and push-pull output buffer.

***Table 2-115* • Minimum and Maximum DC Input and Output Levels**

1.8 V LVCMOS	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
Applicable to Pro I/O Banks												
2 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	2	2	11	9	10	10
4 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	4	4	22	17	10	10
6 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	6	6	44	35	10	10
8 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	8	8	51	45	10	10
12 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	12	12	74	91	10	10
16 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	16	16	74	91	10	10
Applicable to Advanced I/O Banks												
2 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	2	2	11	9	10	10
4 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	4	4	22	17	10	10
6 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	6	6	44	35	10	10
8 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	8	8	51	45	10	10
12 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	12	12	74	91	10	10
16 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.9	0.45	V_{CCI} – 0.45	16	16	74	91	10	10
Applicable to Standard I/O Banks												
2 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	2	2	11	9	10	10
4 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.45	V_{CCI} – 0.45	4	4	22	17	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*
5. *Software default selection highlighted in gray.*

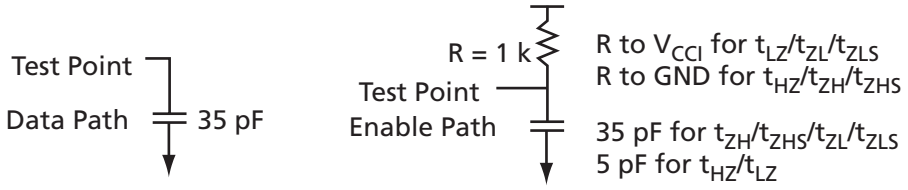


***Figure 2-119* • AC Loading**

Table 2-116 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input LOW (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	C_{LOAD} (pF)
0	1.8	0.9	–	35

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-117 • 1.8 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.7 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	15.84	0.04	1.45	1.91	0.43	15.65	15.84	2.78	1.58	17.89	18.07	ns
	–1	0.56	13.47	0.04	1.23	1.62	0.36	13.31	13.47	2.37	1.35	15.22	15.37	ns
	–2	0.49	11.83	0.03	1.08	1.42	0.32	11.69	11.83	2.08	1.18	13.36	13.50	ns
4 mA	Std.	0.66	11.39	0.04	1.45	1.91	0.43	11.60	10.76	3.26	2.77	13.84	12.99	ns
	–1	0.56	9.69	0.04	1.23	1.62	0.36	9.87	9.15	2.77	2.36	11.77	11.05	ns
	–2	0.49	8.51	0.03	1.08	1.42	0.32	8.66	8.03	2.43	2.07	10.33	9.70	ns
8 mA	Std.	0.66	8.97	0.04	1.45	1.91	0.43	9.14	8.10	3.57	3.36	11.37	10.33	ns
	–1	0.56	7.63	0.04	1.23	1.62	0.36	7.77	6.89	3.04	2.86	9.67	8.79	ns
	–2	0.49	6.70	0.03	1.08	1.42	0.32	6.82	6.05	2.66	2.51	8.49	7.72	ns
12 mA	Std.	0.66	8.35	0.04	1.45	1.91	0.43	8.50	7.59	3.64	3.52	10.74	9.82	ns
	–1	0.56	7.10	0.04	1.23	1.62	0.36	7.23	6.45	3.10	3.00	9.14	8.35	ns
	–2	0.49	6.24	0.03	1.08	1.42	0.32	6.35	5.66	2.72	2.63	8.02	7.33	ns
16 mA	Std.	0.66	7.94	0.04	1.45	1.91	0.43	8.09	7.56	3.74	4.11	10.32	9.80	ns
	–1	0.56	6.75	0.04	1.23	1.62	0.36	6.88	6.43	3.18	3.49	8.78	8.33	ns
	–2	0.49	5.93	0.03	1.08	1.42	0.32	6.04	5.65	2.79	3.07	7.71	7.32	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-118 • 1.8 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.7 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	12.10	0.04	1.45	1.91	0.43	9.59	12.10	2.78	1.64	11.83	14.34	ns
	–1	0.56	10.30	0.04	1.23	1.62	0.36	8.16	10.30	2.37	1.39	10.06	12.20	ns
	–2	0.49	9.04	0.03	1.08	1.42	0.32	7.16	9.04	2.08	1.22	8.83	10.71	ns
4 mA	Std.	0.66	7.05	0.04	1.45	1.91	0.43	6.20	7.05	3.25	2.86	8.44	9.29	ns
	–1	0.56	6.00	0.04	1.23	1.62	0.36	5.28	6.00	2.76	2.44	7.18	7.90	ns
	–2	0.49	5.27	0.03	1.08	1.42	0.32	4.63	5.27	2.43	2.14	6.30	6.94	ns
8 mA	Std.	0.66	4.52	0.04	1.45	1.91	0.43	4.47	4.52	3.57	3.47	6.70	6.76	ns
	–1	0.56	3.85	0.04	1.23	1.62	0.36	3.80	3.85	3.04	2.95	5.70	5.75	ns
	–2	0.49	3.38	0.03	1.08	1.42	0.32	3.33	3.38	2.66	2.59	5.00	5.05	ns
12 mA	Std.	0.66	4.12	0.04	1.45	1.91	0.43	4.20	3.99	3.63	3.62	6.43	6.23	ns
	–1	0.56	3.51	0.04	1.23	1.62	0.36	3.57	3.40	3.09	3.08	5.47	5.30	ns
	–2	0.49	3.08	0.03	1.08	1.42	0.32	3.14	2.98	2.71	2.71	4.81	4.65	ns
16 mA	Std.	0.66	3.80	0.04	1.45	1.91	0.43	3.87	3.09	3.73	4.24	6.10	5.32	ns
	–1	0.56	3.23	0.04	1.23	1.62	0.36	3.29	2.63	3.18	3.60	5.19	4.53	ns
	–2	0.49	2.83	0.03	1.08	1.42	0.32	2.89	2.31	2.79	3.16	4.56	3.98	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-119 • **1.8 V LVCMOS Low Slew**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.7 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	15.53	0.04	1.31	0.43	14.11	15.53	2.78	1.60	16.35	17.77	ns
	–1	0.56	13.21	0.04	1.11	0.36	12.01	13.21	2.36	1.36	13.91	15.11	ns
	–2[2]	0.49	11.60	0.03	0.98	0.32	10.54	11.60	2.07	1.19	12.21	13.27	ns
4 mA	Std.	0.66	10.48	0.04	1.31	0.43	10.41	10.48	3.23	2.73	12.65	12.71	ns
	–1	0.56	8.91	0.04	1.11	0.36	8.86	8.91	2.75	2.33	10.76	10.81	ns
	–2	0.49	7.82	0.03	0.98	0.32	7.77	7.82	2.41	2.04	9.44	9.49	ns
8 mA	Std.	0.66	8.05	0.04	1.31	0.43	8.20	7.84	3.54	3.27	10.43	10.08	ns
	–1	0.56	6.85	0.04	1.11	0.36	6.97	6.67	3.01	2.78	8.88	8.57	ns
	–2	0.49	6.01	0.03	0.98	0.32	6.12	5.86	2.64	2.44	7.79	7.53	ns
12 mA	Std.	0.66	7.50	0.04	1.31	0.43	7.64	7.30	3.61	3.41	9.88	9.53	ns
	–1	0.56	6.38	0.04	1.11	0.36	6.50	6.21	3.07	2.90	8.40	8.11	ns
	–2	0.49	5.60	0.03	0.98	0.32	5.71	5.45	2.69	2.55	7.38	7.12	ns
16 mA	Std.	0.66	7.29	0.04	1.31	0.43	7.23	7.29	3.71	3.95	9.47	9.53	ns
	–1	0.56	6.20	0.04	1.11	0.36	6.15	6.20	3.15	3.36	8.06	8.11	ns
	–2	0.49	5.45	0.03	0.98	0.32	5.40	5.45	2.77	2.95	7.07	7.12	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-120 • 1.8 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.7 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	11.86	0.04	1.22	0.43	9.14	11.86	2.77	1.66	11.37	14.10	ns
	–1	0.56	10.09	0.04	1.04	0.36	7.77	10.09	2.36	1.41	9.67	11.99	ns
	–2	0.49	8.86	0.03	0.91	0.32	6.82	8.86	2.07	1.24	8.49	10.53	ns
4 mA	Std.	0.66	6.91	0.04	1.22	0.43	5.86	6.91	3.22	2.84	8.10	9.15	ns
	–1	0.56	5.88	0.04	1.04	0.36	4.99	5.88	2.74	2.41	6.89	7.78	ns
	–2	0.49	5.16	0.03	0.91	0.32	4.38	5.16	2.41	2.12	6.05	6.83	ns
8 mA	Std.	0.66	4.45	0.04	1.22	0.43	4.18	4.45	3.53	3.38	6.42	6.68	ns
	–1	0.56	3.78	0.04	1.04	0.36	3.56	3.78	3.00	2.88	5.46	5.69	ns
	–2	0.49	3.32	0.03	0.91	0.32	3.12	3.32	2.64	2.53	4.79	4.99	ns
12 mA	Std.	0.66	3.92	0.04	1.22	0.43	3.93	3.92	3.60	3.52	6.16	6.16	ns
	–1	0.56	3.34	0.04	1.04	0.36	3.34	3.34	3.06	3.00	5.24	5.24	ns
	–2	0.49	2.93	0.03	0.91	0.32	2.93	2.93	2.69	2.63	4.60	4.60	ns
16 mA	Std.	0.66	3.53	0.04	1.22	0.43	3.60	3.04	3.70	4.08	5.84	5.28	ns
	–1	0.56	3.01	0.04	1.04	0.36	3.06	2.59	3.15	3.47	4.96	4.49	ns
	–2	0.49	2.64	0.03	0.91	0.32	2.69	2.27	2.76	3.05	4.36	3.94	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-121 • 1.8 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.7 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	15.01	0.04	1.20	0.43	13.15	15.01	1.99	1.99	ns
	–1	0.56	12.77	0.04	1.02	0.36	11.19	12.77	1.70	1.70	ns
	–2	0.49	11.21	0.03	0.90	0.32	9.82	11.21	1.49	1.49	ns
4 mA	Std.	0.66	10.10	0.04	1.20	0.43	9.55	10.10	2.41	2.37	ns
	–1	0.56	8.59	0.04	1.02	0.36	8.13	8.59	2.05	2.02	ns
	–2	0.49	7.54	0.03	0.90	0.32	7.13	7.54	1.80	1.77	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-122 • **1.8 V LVCMOS High Slew**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.7 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	11.21	0.04	1.20	0.43	8.53	11.21	1.99	1.21	ns
	–1	0.56	9.54	0.04	1.02	0.36	7.26	9.54	1.69	1.03	ns
	–2	0.49	8.37	0.03	0.90	0.32	6.37	8.37	1.49	0.90	ns
4 mA	Std.	0.66	6.34	0.04	1.20	0.43	5.38	6.34	2.41	2.48	ns
	–1	0.56	5.40	0.04	1.02	0.36	4.58	5.40	2.05	2.11	ns
	–2	0.49	4.74	0.03	0.90	0.32	4.02	4.74	1.80	1.85	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

1.5 V LVCMOS (JESD8-11)

Low-Voltage CMOS for 1.5 V is an extension of the LVCMOS standard (JESD8-5) used for general-purpose 1.5 V applications. It uses a 1.5 V input buffer and push-pull output buffer.

Table 2-123 • Minimum and Maximum DC Input and Output Levels

1.5 V LVCMOS	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
Applicable to Pro I/O Banks												
2 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	2	2	16	13	10	10
4 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	4	4	33	25	10	10
6 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	6	6	39	32	10	10
8 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	8	8	55	66	10	10
12 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	12	12	55	66	10	10
Applicable to Advanced I/O Banks												
2 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.575	0.25 * V_{CCI}	0.75 * V_{CCI}	2	2	16	13	10	10
4 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.575	0.25 * V_{CCI}	0.75 * V_{CCI}	4	4	33	25	10	10
6 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.575	0.25 * V_{CCI}	0.75 * V_{CCI}	6	6	39	32	10	10
8 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.575	0.25 * V_{CCI}	0.75 * V_{CCI}	8	8	55	66	10	10
12 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	1.575	0.25 * V_{CCI}	0.75 * V_{CCI}	12	12	55	66	10	10
Applicable to Pro I/O Banks												
2 mA	–0.3	0.35 * V_{CCI}	0.65 * V_{CCI}	3.6	0.25 * V_{CCI}	0.75 * V_{CCI}	2	2	16	13	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*
5. *Software default selection highlighted in gray.*

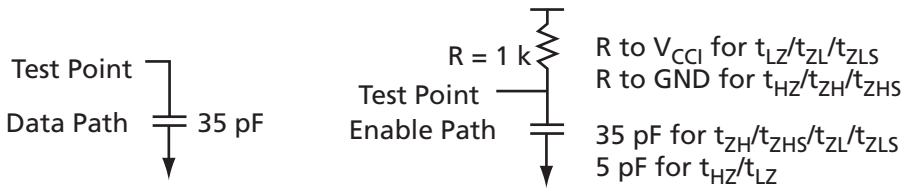


Figure 2-120 • AC Loading

Table 2-124 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	C_{LOAD} (pF)
0	1.5	0.75	–	35

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-125 • 1.5 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	14.11	0.04	1.70	2.14	0.43	14.37	13.14	3.40	2.68	16.61	15.37	ns
	–1	0.56	12.00	0.04	1.44	1.82	0.36	12.22	11.17	2.90	2.28	14.13	13.08	ns
	–2	0.49	10.54	0.03	1.27	1.60	0.32	10.73	9.81	2.54	2.00	12.40	11.48	ns
4 mA	Std.	0.66	11.23	0.04	1.70	2.14	0.43	11.44	9.87	3.77	3.36	13.68	12.10	ns
	–1	0.56	9.55	0.04	1.44	1.82	0.36	9.73	8.39	3.21	2.86	11.63	10.29	ns
	–2	0.49	8.39	0.03	1.27	1.60	0.32	8.54	7.37	2.81	2.51	10.21	9.04	ns
8 mA	Std.	0.66	10.45	0.04	1.70	2.14	0.43	10.65	9.24	3.84	3.55	12.88	11.48	ns
	–1	0.56	8.89	0.04	1.44	1.82	0.36	9.06	7.86	3.27	3.02	10.96	9.76	ns
	–2	0.49	7.81	0.03	1.27	1.60	0.32	7.95	6.90	2.87	2.65	9.62	8.57	ns
12 mA	Std.	0.66	10.02	0.04	1.70	2.14	0.43	10.20	9.23	3.97	4.22	12.44	11.47	ns
	–1	0.56	8.52	0.04	1.44	1.82	0.36	8.68	7.85	3.38	3.59	10.58	9.75	ns
	–2	0.49	7.48	0.03	1.27	1.60	0.32	7.62	6.89	2.97	3.15	9.29	8.56	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-126 • 1.5 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V
Applicable to Pro I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	8.53	0.04	1.70	2.14	0.43	7.26	8.53	3.39	2.79	9.50	10.77	ns
	–1	0.56	7.26	0.04	1.44	1.82	0.36	6.18	7.26	2.89	2.37	8.08	9.16	ns
	–2	0.49	6.37	0.03	1.27	1.60	0.32	5.42	6.37	2.53	2.08	7.09	8.04	ns
4 mA	Std.	0.66	5.41	0.04	1.70	2.14	0.43	5.22	5.41	3.75	3.48	7.45	7.65	ns
	–1	0.56	4.60	0.04	1.44	1.82	0.36	4.44	4.60	3.19	2.96	6.34	6.50	ns
	–2	0.49	4.04	0.03	1.27	1.60	0.32	3.89	4.04	2.80	2.60	5.56	5.71	ns
8 mA	Std.	0.66	4.80	0.04	1.70	2.14	0.43	4.89	4.75	3.83	3.67	7.13	6.98	ns
	–1	0.56	4.09	0.04	1.44	1.82	0.36	4.16	4.04	3.26	3.12	6.06	5.94	ns
	–2	0.49	3.59	0.03	1.27	1.60	0.32	3.65	3.54	2.86	2.74	5.32	5.21	ns
12 mA	Std.	0.66	4.42	0.04	1.70	2.14	0.43	4.50	3.62	3.96	4.37	6.74	5.86	ns
	–1	0.56	3.76	0.04	1.44	1.82	0.36	3.83	3.08	3.37	3.72	5.73	4.98	ns
	–2	0.49	3.30	0.03	1.27	1.60	0.32	3.36	2.70	2.96	3.27	5.03	4.37	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-127 • 1.5 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	12.78	0.04	1.31	0.43	12.81	12.78	3.40	2.64	15.05	15.02	ns
	–1	0.56	10.87	0.04	1.11	0.36	10.90	10.87	2.89	2.25	12.80	12.78	ns
	–2	0.49	9.55	0.03	0.98	0.32	9.57	9.55	2.54	1.97	11.24	11.22	ns
4 mA	Std.	0.66	10.01	0.04	1.31	0.43	10.19	9.55	3.75	3.27	12.43	11.78	ns
	–1	0.56	8.51	0.04	1.11	0.36	8.67	8.12	3.19	2.78	10.57	10.02	ns
	–2	0.49	7.47	0.03	0.98	0.32	7.61	7.13	2.80	2.44	9.28	8.80	ns
8 mA	Std.	0.66	9.33	0.04	1.31	0.43	9.51	8.89	3.83	3.43	11.74	11.13	ns
	–1	0.56	7.94	0.04	1.11	0.36	8.09	7.56	3.26	2.92	9.99	9.47	ns
	–2	0.49	6.97	0.03	0.98	0.32	7.10	6.64	2.86	2.56	8.77	8.31	ns
12 mA	Std.	0.66	8.91	0.04	1.31	0.43	9.07	8.89	3.95	4.05	11.31	11.13	ns
	–1	0.56	7.58	0.04	1.11	0.36	7.72	7.57	3.36	3.44	9.62	9.47	ns
	–2	0.49	6.65	0.03	0.98	0.32	6.78	6.64	2.95	3.02	8.45	8.31	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-128 • 1.5 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V
Applicable to Advanced I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
2 mA	Std.	0.66	8.36	0.04	1.44	0.43	6.82	8.36	3.39	2.77	9.06	10.60	ns
	–1	0.56	7.11	0.04	1.22	0.36	5.80	7.11	2.88	2.35	7.71	9.02	ns
	–2	0.49	6.24	0.03	1.07	0.32	5.10	6.24	2.53	2.06	6.76	7.91	ns
4 mA	Std.	0.66	5.31	0.04	1.44	0.43	4.85	5.31	3.74	3.40	7.09	7.55	ns
	–1	0.56	4.52	0.04	1.22	0.36	4.13	4.52	3.18	2.89	6.03	6.42	ns
	–2	0.49	3.97	0.03	1.07	0.32	3.62	3.97	2.79	2.54	5.29	5.64	ns
8 mA	Std.	0.66	4.67	0.04	1.44	0.43	4.55	4.67	3.82	3.56	6.78	6.90	ns
	–1	0.56	3.97	0.04	1.22	0.36	3.87	3.97	3.25	3.03	5.77	5.87	ns
	–2	0.49	3.49	0.03	1.07	0.32	3.40	3.49	2.85	2.66	5.07	5.16	ns
12 mA	Std.	0.66	4.08	0.04	1.44	0.43	4.15	3.58	3.94	4.20	6.39	5.81	ns
	–1	0.56	3.47	0.04	1.22	0.36	3.53	3.04	3.36	3.58	5.44	4.95	ns
	–2	0.49	3.05	0.03	1.07	0.32	3.10	2.67	2.95	3.14	4.77	4.34	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-129 • 1.5 V LVCMOS Low Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	12.33	0.04	1.42	0.43	11.79	12.33	2.45	2.32	ns
	–1	0.56	10.49	0.04	1.21	0.36	10.03	10.49	2.08	1.98	ns
	–2	0.49	9.21	0.03	1.06	0.32	8.81	9.21	1.83	1.73	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-130 • 1.5 V LVCMOS High Slew
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V
Applicable to Standard I/Os

Drive Strength	Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	Units
2 mA	Std.	0.66	7.65	0.04	1.42	0.43	6.31	7.65	2.45	2.45	ns
	–1	0.56	6.50	0.04	1.21	0.36	5.37	6.50	2.08	2.08	ns
	–2	0.49	5.71	0.03	1.06	0.32	4.71	5.71	1.83	1.83	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

3.3 V PCI, 3.3 V PCI-X

The Peripheral Component Interface for 3.3 V standard specifies support for 33 MHz and 66 MHz PCI Bus applications.

Table 2-131 • Minimum and Maximum DC Input and Output Levels

3.3 V PCI/PCI-X	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
Per PCI specification	Per PCI curves										10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

AC loadings are defined per the PCI/PCI-X specifications for the datapath; Actel loadings for enable path characterization are described in Figure 2-121.

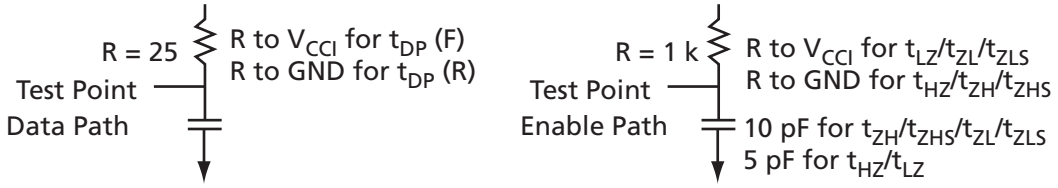


Figure 2-121 • AC Loading

AC loadings are defined per PCI/PCI-X specifications for the data path; Actel loading for tristate is described in Table 2-132.

Table 2-132 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	C_{LOAD} (pF)
0	3.3	0.285 * V_{CCI} for $t_{DP(R)}$ 0.615 * V_{CCI} for $t_{DP(F)}$	–	10

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-133 • 3.3 V PCI/PCI-X
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V
Applicable to Pro I/Os

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.81	0.04	1.05	1.67	0.43	2.86	2.00	3.28	3.61	5.09	4.23	ns
–1	0.56	2.39	0.04	0.89	1.42	0.36	2.43	1.70	2.79	3.07	4.33	3.60	ns
–2	0.49	2.09	0.03	0.78	1.25	0.32	2.13	1.49	2.45	2.70	3.80	3.16	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Table 2-134 • 3.3 V PCI/PCI-X
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V
Applicable to Advanced I/Os

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{PYS}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.68	0.04	0.86	0.43	2.73	1.95	3.21	3.58	4.97	4.19	0.66	ns
–1	0.56	2.28	0.04	0.73	0.36	2.32	1.66	2.73	3.05	4.22	3.56	0.56	ns
–2	0.49	2.00	0.03	0.65	0.32	2.04	1.46	2.40	2.68	3.71	3.13	0.49	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Voltage Referenced I/O Characteristics

3.3 V GTL

Gunning Transceiver Logic is a high-speed bus standard (JESD8-3). It provides a differential amplifier input buffer and an open-drain output buffer. The V_{CCI} pin should be connected to 3.3 V.

Table 2-135 • Minimum and Maximum DC Input and Output Levels

3.3 V GTL	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
25 mA[3]	–0.3	V_{REF} – 0.05	V_{REF} + 0.05	3.6	0.4	–	25	25	181	268	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

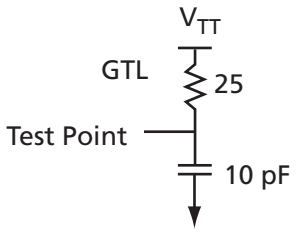


Figure 2-122 • AC Loading

Table 2-136 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.05	V_{REF} + 0.05	0.8	0.8	1.2	10

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-137 • 3.3 V GTL
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V, V_{REF} = 0.8 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.08	0.04	2.93	0.43	2.04	2.08			4.27	4.31	ns
–1	0.56	1.77	0.04	2.50	0.36	1.73	1.77			3.63	3.67	ns
–2	0.49	1.55	0.03	2.19	0.32	1.52	1.55			3.19	3.22	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

2.5 V GTL

Gunning Transceiver Logic is a high-speed bus standard (JESD8-3). It provides a differential amplifier input buffer and an open-drain output buffer. The V_{CCI} pin should be connected to 2.5 V.

Table 2-138 • Minimum and Maximum DC Input and Output Levels

2.5 GTL	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
25 mA[3]	–0.3	V_{REF} – 0.05	V_{REF} + 0.05	3.6	0.4	–	25	25	124	169	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

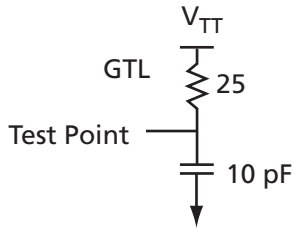


Figure 2-123 • AC Loading

Table 2-139 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.05	V_{REF} + 0.05	0.8	0.8	1.2	10

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-140 • 2.5 V GTL
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V, V_{REF} = 0.8 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.13	0.04	2.46	0.43	2.16	2.13			4.40	4.36	ns
–1	0.56	1.81	0.04	2.09	0.36	1.84	1.81			3.74	3.71	ns
–2	0.49	1.59	0.03	1.83	0.32	1.61	1.59			3.28	3.26	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

3.3 V GTL+

Gunning Transceiver Logic Plus is a high-speed bus standard (JESD8-3). It provides a differential amplifier input buffer and an open-drain output buffer. The V_{CCI} pin should be connected to 3.3 V.

***Table 2-141* • Minimum and Maximum DC Input and Output Levels**

3.3 V GTL+	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
35 mA	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.6	–	35	35	181	268	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

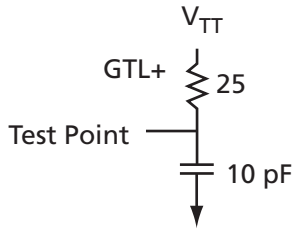


***Figure 2-124* • AC Loading**

***Table 2-142* • AC Waveforms, Measuring Points, and Capacitive Loads**

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.1	V_{REF} + 0.1	1.0	1.0	1.5	10

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

***Table 2-143* • 3.3 V GTL+**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V, V_{REF} = 1.0 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.06	0.04	1.59	0.43	2.09	2.06			4.33	4.29	ns
–1	0.56	1.75	0.04	1.35	0.36	1.78	1.75			3.68	3.65	ns
–2	0.49	1.53	0.03	1.19	0.32	1.56	1.53			3.23	3.20	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

2.5 V GTL+

Gunning Transceiver Logic Plus is a high-speed bus standard (JESD8-3). It provides a differential amplifier input buffer and an open-drain output buffer. The V_{CCI} pin should be connected to 2.5 V.

Table 2-144 • Minimum and Maximum DC Input and Output Levels

2.5 V GTL+	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
33 mA	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.6	–	33	33	124	169	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

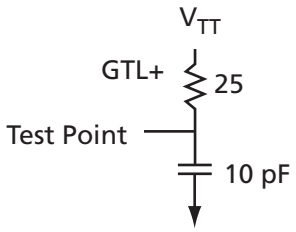


Figure 2-125 • AC Loading

Table 2-145 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.1	V_{REF} + 0.1	1.0	1.0	1.5	10

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-146 • 2.5 V GTL+
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V, V_{REF} = 1.0 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.21	0.04	1.51	0.43	2.25	2.10			4.48	4.34	ns
–1	0.56	1.88	0.04	1.29	0.36	1.91	1.79			3.81	3.69	ns
–2	0.49	1.65	0.03	1.13	0.32	1.68	1.57			3.35	4.34	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

HSTL Class I

High-Speed Transceiver Logic is a general-purpose high-speed 1.5 V bus standard (EIA/JESD8-6). Fusion devices support Class I. This provides a differential amplifier input buffer and a push-pull output buffer.

Table 2-147 • Minimum and Maximum DC Input and Output Levels

HSTL Class I	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL} [1]	I_{IH} [2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA [3]	Max., mA [3]	µA [4]	µA [4]
8 mA	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.4	V_{CCI} – 0.4	8	8	39	32	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

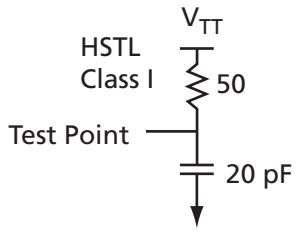


Figure 2-126 • AC Loading

Table 2-148 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.1	V_{REF} + 0.1	0.75	0.75	0.75	20

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-149 • HSTL Class I
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V, V_{REF} = 0.75 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	3.18	0.04	2.12	0.43	3.24	3.14			5.47	5.38	ns
–1	0.56	2.70	0.04	1.81	0.36	2.75	2.67			4.66	4.58	ns
–2	0.49	2.37	0.03	1.59	0.32	2.42	2.35			4.09	4.02	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

HSTL Class II

High-Speed Transceiver Logic is a general-purpose high-speed 1.5 V bus standard (EIA/JESD8-6). Fusion devices support Class II. This provides a differential amplifier input buffer and a push-pull output buffer.

Table 2-150 • Minimum and Maximum DC Input and Output Levels

HSTL Class II	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
15 mA[3]	–0.3	V_{REF} – 0.1	V_{REF} + 0.1	3.6	0.4	V_{CCI} – 0.4	15	15	55	66	10	10

Note:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*
5. *Output drive strength is below JEDEC specification.*

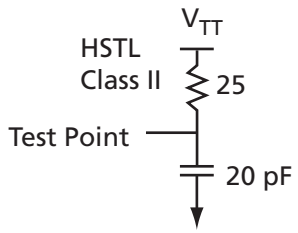


Figure 2-127 • AC Loading

Table 2-151 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.1	V_{REF} + 0.1	0.75	0.75	0.75	20

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-152 • HSTL Class II
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 1.4 V, V_{REF} = 0.75 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	3.02	0.04	2.12	0.43	3.08	2.71			5.32	4.95	ns
–1	0.56	2.57	0.04	1.81	0.36	2.62	2.31			4.52	4.21	ns
–2	0.49	2.26	0.03	1.59	0.32	2.30	2.03			3.97	3.70	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

SSTL2 Class I

Stub-Speed Terminated Logic for 2.5 V memory bus standard (JESD8-9). Fusion devices support Class I. This provides a differential amplifier input buffer and a push-pull output buffer.

Table 2-153 • Minimum and Maximum DC Input and Output Levels

SSTL2 Class I	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
15 mA	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.54	V_{CCI} – 0.62	15	15	87	83	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

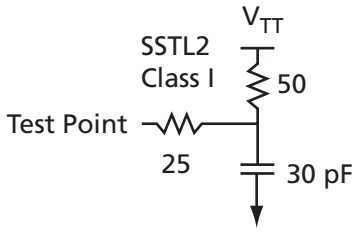


Figure 2-128 • AC Loading

Table 2-154 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.2	V_{REF} + 0.2	1.25	1.25	1.25	30

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-155 • SSTL 2 Class I
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V, V_{REF} = 1.25 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.13	0.04	1.33	0.43	2.17	1.85			4.40	4.08	ns
–1	0.56	1.81	0.04	1.14	0.36	1.84	1.57			3.74	3.47	ns
–2	0.49	1.59	0.03	1.00	0.32	1.62	1.38			3.29	3.05	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

SSTL2 Class II

Stub-Speed Terminated Logic for 2.5 V memory bus standard (JESD8-9). Fusion devices support Class II. This provides a differential amplifier input buffer and a push-pull output buffer.

Table 2-156 • Minimum and Maximum DC Input and Output Levels

SSTL2 Class II	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
18 mA	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.35	V_{CCI} – 0.43	18	18	124	169	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

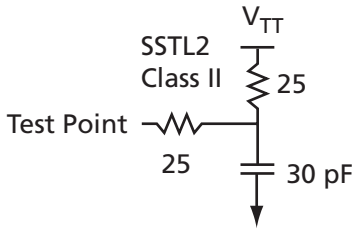


Figure 2-129 • AC Loading

Table 2-157 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.2	V_{REF} + 0.2	1.25	1.25	1.25	30

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-158 • SSTL 2 Class II
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V, V_{REF} = 1.25 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.17	0.04	1.33	0.43	2.21	1.77			4.44	4.01	ns
–1	0.56	1.84	0.04	1.14	0.36	1.88	1.51			3.78	3.41	ns
–2	0.49	1.62	0.03	1.00	0.32	1.65	1.32			3.32	2.99	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

SSTL3 Class I

Stub-Speed Terminated Logic for 3.3 V memory bus standard (JESD8-8). Fusion devices support Class I. This provides a differential amplifier input buffer and a push-pull output buffer.

Table 2-159 • Minimum and Maximum DC Input and Output Levels

SSTL3 Class I	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL} [1]	I_{IH} [2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA [3]	Max., mA [3]	µA [4]	µA [4]
14 mA	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.7	V_{CCI} – 1.1	14	14	54	51	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

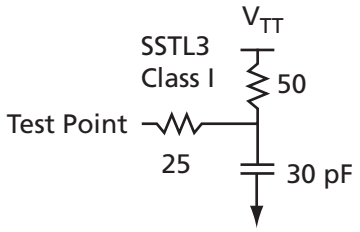


Figure 2-130 • AC Loading

Table 2-160 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.2	V_{REF} + 0.2	1.5	1.5	1.485	30

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-161 • SSTL3 Class I
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V, V_{REF} = 1.5 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.31	0.04	1.25	0.43	2.35	1.84			4.59	4.07	ns
–1	0.56	1.96	0.04	1.06	0.36	2.00	1.56			3.90	3.46	ns
–2	0.49	1.72	0.03	0.93	0.32	1.75	1.37			3.42	3.04	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

SSTL3 Class II

Stub-Speed Terminated Logic for 3.3 V memory bus standard (JESD8-8). Fusion devices support Class II. This provides a differential amplifier input buffer and a push-pull output buffer.

Table 2-162 • Minimum and Maximum DC Input and Output Levels

SSTL3 Class II	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}	I_{OSL}	I_{OSH}	I_{IL}[1]	I_{IH}[2]
Drive Strength	Min., V	Max., V	Min., V	Max., V	Max., V	Min., V	mA	mA	Max., mA[3]	Max., mA[3]	µA[4]	µA[4]
21 mA	–0.3	V_{REF} – 0.2	V_{REF} + 0.2	3.6	0.5	V_{CCI} – 0.9	21	21	109	103	10	10

Notes:

1. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
2. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*
3. *Currents are measured at high temperature (100°C junction temperature) and maximum voltage.*
4. *Currents are measured at 85°C junction temperature.*

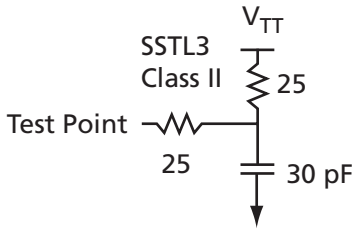


Figure 2-131 • AC Loading

Table 2-163 • AC Waveforms, Measuring Points, and Capacitive Loads

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)	V_{TT} (typ.) (V)	C_{LOAD} (pF)
V_{REF} – 0.2	V_{REF} + 0.2	1.5	1.5	1.485	30

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-164 • SSTL3- Class II
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 3.0 V, V_{REF} = 1.5 V

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	t_{EOUT}	t_{ZL}	t_{ZH}	t_{LZ}	t_{HZ}	t_{ZLS}	t_{ZHS}	Units
Std.	0.66	2.07	0.04	1.25	0.43	2.10	1.67			4.34	3.91	ns
–1	0.56	1.76	0.04	1.06	0.36	1.79	1.42			3.69	3.32	ns
–2	0.49	1.54	0.03	0.93	0.32	1.57	1.25			3.24	2.92	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Differential I/O Characteristics

Configuration of the I/O modules as a differential pair is handled by the Actel Designer software when the user instantiates a differential I/O macro in the design.

Differential I/Os can also be used in conjunction with the embedded Input Register (InReg), Output Register (OutReg), Enable Register (EnReg), and Double Data Rate (DDR). However, there is no support for bidirectional I/Os or tristates with these standards.

LVDS

Low-Voltage Differential Signal (ANSI/TIA/EIA-644) is a high-speed differential I/O standard. It requires that one data bit be carried through two signal lines, so two pins are needed. It also requires external resistor termination.

The full implementation of the LVDS transmitter and receiver is shown in an example in Figure 2-132. The building blocks of the LVDS transmitter–receiver are one transmitter macro, one receiver macro, three board resistors at the transmitter end, and one resistor at the receiver end. The values for the three driver resistors are different from those used in the LVPECL implementation because the output standard specifications are different.

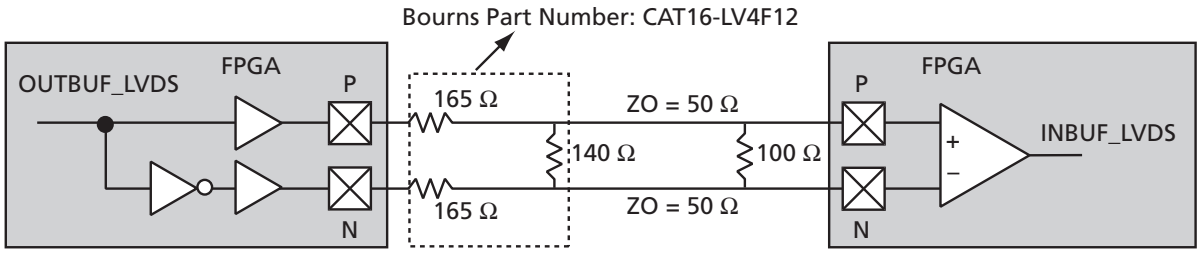


***Figure 2-132* • LVDS Circuit Diagram and Board-Level Implementation**

***Table 2-165* • Minimum and Maximum DC Input and Output Levels**

DC Parameter	Description	Min.	Typ.	Max.	Units
V_{CCI}	Supply Voltage	2.375	2.5	2.625	V
V_{OL}	Output LOW Voltage	0.9	1.075	1.25	V
V_{OH}	Input HIGH Voltage	1.25	1.425	1.6	V
I_{OL} [3]	Output LOW Voltage	0.65	0.91	1.16	mA
I_{OH} [3]	Output HIGH Voltage	0.65	0.91	1.16	mA
V_I	Input Voltage	0		2.925	V
I_{IL} [4,5]	Input LOW Voltage			10	µA
I_{IH} [4,6]	Input HIGH Voltage			10	µA
V_{ODIFF}	Differential Output Voltage	250	350	450	mV
V_{OCM}	Output Common Mode Voltage	1.125	1.25	1.375	V
V_{ICM}	Input Common Mode Voltage	0.05	1.25	2.35	V
V_{IDIFF}	Input Differential Voltage	100	350		mV

Notes:

1. *±5%*
2. *Differential input voltage = ±350 mV*
3. *I_{OL}/I_{OH} defined by V_{ODIFF}/(Resistor Network)*
4. *Currents are measured at 85°C junction temperature.*
5. *I_{IL} is the input leakage current per I/O pin over recommended operation conditions where –0.3 V < V_{IN} < V_{IL}.*
6. *I_{IH} is the input leakage current per I/O pin over recommended operating conditions V_{IH} < V_{IN} < V_{CCI}. Input current is larger when operating outside recommended ranges.*

Table 2-166 • **AC Waveforms, Measuring Points, and Capacitive Loads**

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)
1.075	1.325	Cross point	–

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

Table 2-167 • **LVDS**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V, Worst-Case V_{CCI} = 2.3 V
Applicable to Pro I/Os

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	Units
Std.	0.66	2.10	0.04	1.82	ns
–1	0.56	1.79	0.04	1.55	ns
–2	0.49	1.57	0.03	1.36	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

BLVDS/M-LVDS

Bus LVDS (BLVDS) and Multipoint LVDS (M-LVDS) specifications extend the existing LVDS standard to high-performance multipoint bus applications. Multidrop and multipoint bus configurations can contain any combination of drivers, receivers, and transceivers. Actel LVDS drivers provide the higher drive current required by BLVDS and M-LVDS to accommodate the loading. The driver requires series terminations for better signal quality and to control voltage swing. Termination is also required at both ends of the bus, since the driver can be located anywhere on the bus. These configurations can be implemented using TRIBUF_LVDS and BIBUF_LVDS macros along with appropriate terminations. Multipoint designs using Actel LVDS macros can achieve up to 200 MHz with a maximum of 20 loads. A sample application is given in Figure 2-133. The input and output buffer delays are available in the LVDS section in Table 2-168.

Example: For a bus consisting of 20 equidistant loads, the following terminations provide the required differential voltage, in worst-case industrial operating conditions at the farthest receiver: R_S = 60 Ω and R_T = 70 Ω, given Z_0 = 50 Ω (2") and Z_{stub} = 50 Ω (~1.5").

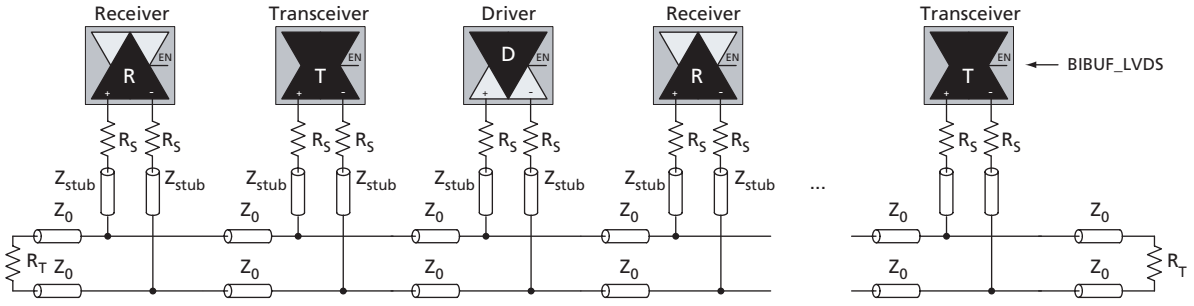


Figure 2-133 • **BLVDS/M-LVDS Multipoint Application Using LVDS I/O Buffers**

LVPECL

Low-Voltage Positive Emitter-Coupled Logic (LVPECL) is another differential I/O standard. It requires that one data bit be carried through two signal lines. Like LVDS, two pins are needed. It also requires external resistor termination.

The full implementation of the LVDS transmitter and receiver is shown in an example in Figure 2-134. The building blocks of the LVPECL transmitter–receiver are one transmitter macro, one receiver macro, three board resistors at the transmitter end, and one resistor at the receiver end. The values for the three driver resistors are different from those used in the LVDS implementation because the output standard specifications are different.

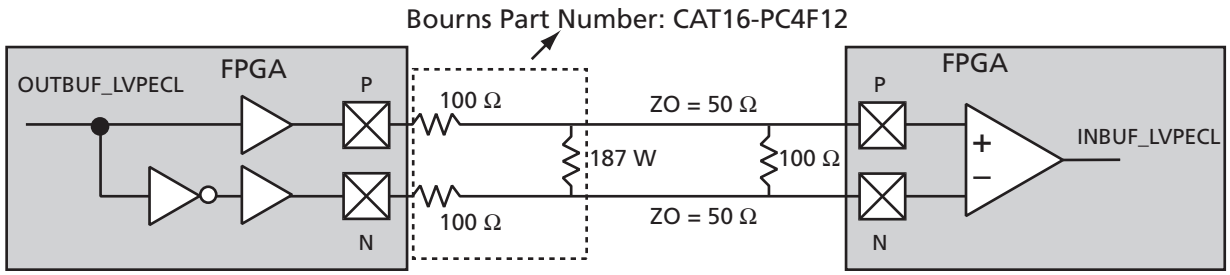


***Figure 2-134* • LVPECL Circuit Diagram and Board-Level Implementation**

***Table 2-168* • Minimum and Maximum DC Input and Output Levels**

DC Parameter	Description	Min.	Max.	Min.	Max.	Min.	Max.	Units
V_{CCI}	Supply Voltage	3.0		3.3		3.6		V
V_{OL}	Output LOW Voltage	0.96	1.27	1.06	1.43	1.30	1.57	V
V_{OH}	Output HIGH Voltage	1.8	2.11	1.92	2.28	2.13	2.41	V
V_{IL}, V_{IH}	Input LOW, Input HIGH Voltages	0	3.3	0	3.6	0	3.9	V
V_{ODIFF}	Differential Output Voltage	0.625	0.97	0.625	0.97	0.625	0.97	V
V_{OCM}	Output Common Mode Voltage	1.762	1.98	1.762	1.98	1.762	1.98	V
V_{ICM}	Input Common Mode Voltage	1.01	2.57	1.01	2.57	1.01	2.57	V
V_{IDIFF}	Input Differential Voltage	300		300		300		mV

***Table 2-169* • AC Waveforms, Measuring Points, and Capacitive Loads**

Input LOW (V)	Input HIGH (V)	Measuring Point* (V)	V_{REF} (typ.) (V)
1.64	1.94	Cross point	–

*Note: *Measuring point = V_{trip}. See Table 2-87 on page 2-167 for a complete table of trip points.*

Timing Characteristics

***Table 2-170* • LVPECL**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V,
Worst-Case V_{CCI} = 3.0 V
Applicable to Pro I/Os

Speed Grade	t_{DOUT}	t_{DP}	t_{DIN}	t_{PY}	Units
Std.	0.66	2.14	0.04	1.63	ns
–1	0.56	1.82	0.04	1.39	ns
–2	0.49	1.60	0.03	1.22	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

I/O Register Specifications

Fully Registered I/O Buffers with Synchronous Enable and Asynchronous Preset

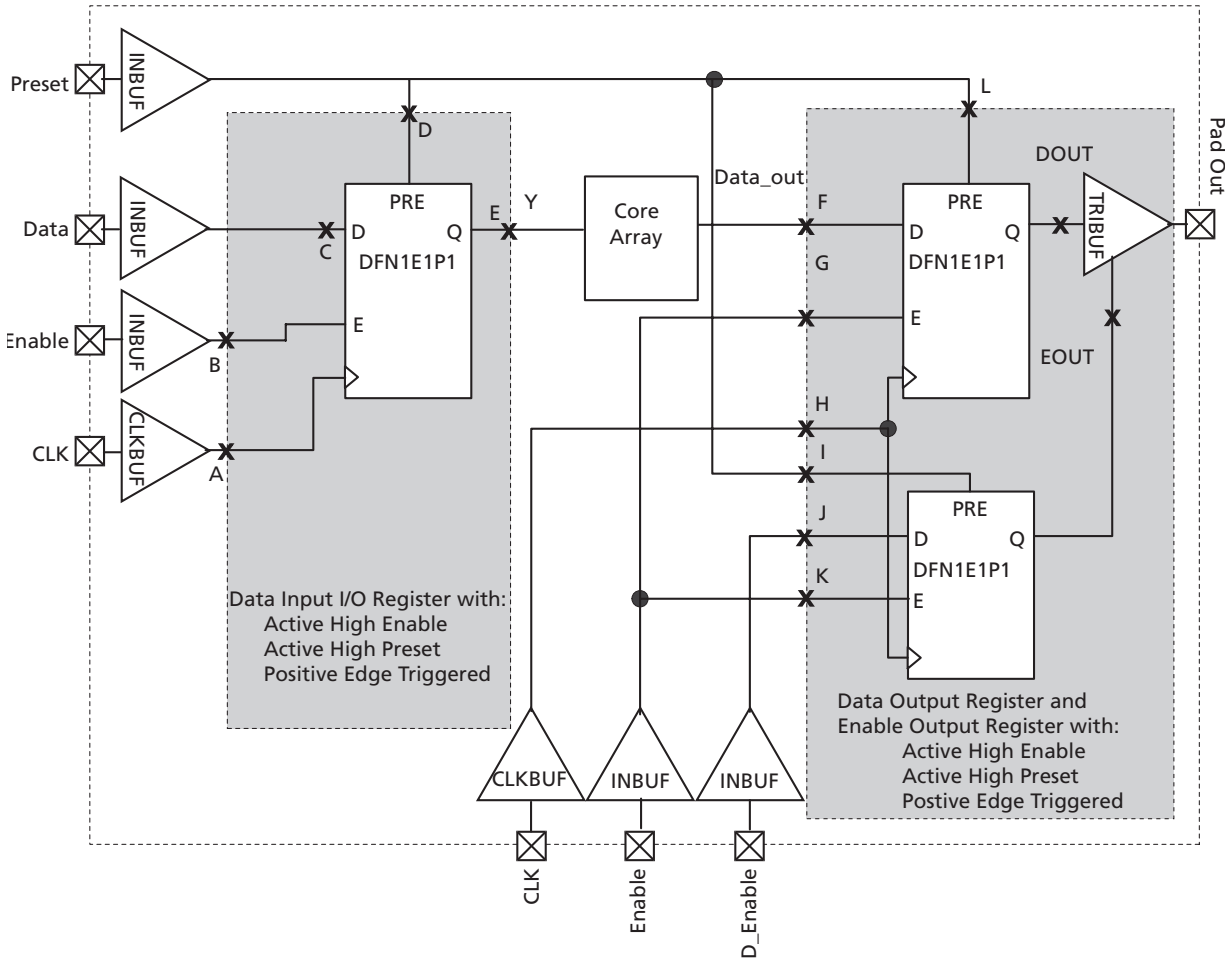


Figure 2-135 • **Timing Model of Registered I/O Buffers with Synchronous Enable and Asynchronous Preset**

***Table 2-171* • Parameter Definitions and Measuring Nodes**

Parameter Name	Parameter Definition	Measuring Nodes (from, to)*
t_{OCLKQ}	Clock-to-Q of the Output Data Register	H, DOUT
t_{OSUD}	Data Setup Time for the Output Data Register	F, H
t_{OHD}	Data Hold Time for the Output Data Register	F, H
t_{OSUE}	Enable Setup Time for the Output Data Register	G, H
t_{OHE}	Enable Hold Time for the Output Data Register	G, H
t_{OPRE2Q}	Asynchronous Preset-to-Q of the Output Data Register	L,DOUT
$t_{OREMPRE}$	Asynchronous Preset Removal Time for the Output Data Register	L, H
$t_{ORECPRE}$	Asynchronous Preset Recovery Time for the Output Data Register	L, H
t_{OECLKQ}	Clock-to-Q of the Output Enable Register	H, EOUT
t_{OESUD}	Data Setup Time for the Output Enable Register	J, H
t_{OEHD}	Data Hold Time for the Output Enable Register	J, H
t_{OESUE}	Enable Setup Time for the Output Enable Register	K, H
t_{OEHE}	Enable Hold Time for the Output Enable Register	K, H
$t_{OEPRE2Q}$	Asynchronous Preset-to-Q of the Output Enable Register	I, EOUT
$t_{OEREMPRE}$	Asynchronous Preset Removal Time for the Output Enable Register	I, H
$t_{OERECPRE}$	Asynchronous Preset Recovery Time for the Output Enable Register	I, H
t_{ICLKQ}	Clock-to-Q of the Input Data Register	A, E
t_{ISUD}	Data Setup Time for the Input Data Register	C, A
t_{IHD}	Data Hold Time for the Input Data Register	C, A
t_{ISUE}	Enable Setup Time for the Input Data Register	B, A
t_{IHE}	Enable Hold Time for the Input Data Register	B, A
t_{IPRE2Q}	Asynchronous Preset-to-Q of the Input Data Register	D, E
$t_{IREMPRE}$	Asynchronous Preset Removal Time for the Input Data Register	D, A
$t_{IRECPRE}$	Asynchronous Preset Recovery Time for the Input Data Register	D, A

*Note: *See Figure 2-135 on page 2-213 for more information.*

Fully Registered I/O Buffers with Synchronous Enable and Asynchronous Clear

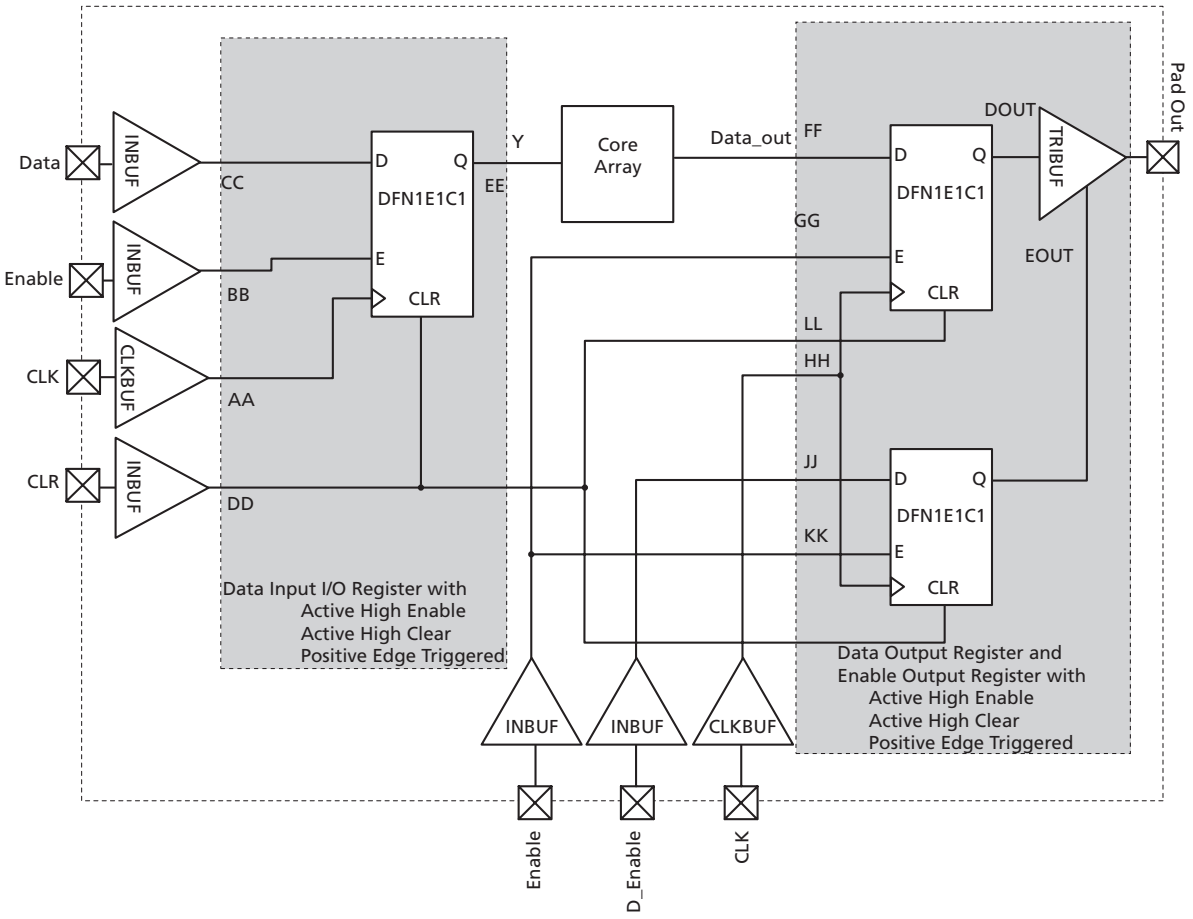


***Figure 2-136* • Timing Model of the Registered I/O Buffers with Synchronous Enable and Asynchronous Clear**

***Table 2-172* • Parameter Definitions and Measuring Nodes**

Parameter Name	Parameter Definition	Measuring Nodes (from, to)*
t_{OCLKQ}	Clock-to-Q of the Output Data Register	HH, DOUT
t_{OSUD}	Data Setup Time for the Output Data Register	FF, HH
t_{OHD}	Data Hold Time for the Output Data Register	FF, HH
t_{OSUE}	Enable Setup Time for the Output Data Register	GG, HH
t_{OHE}	Enable Hold Time for the Output Data Register	GG, HH
t_{OCLR2Q}	Asynchronous Clear-to-Q of the Output Data Register	LL, DOUT
$t_{OREMCLR}$	Asynchronous Clear Removal Time for the Output Data Register	LL, HH
$t_{ORECCLR}$	Asynchronous Clear Recovery Time for the Output Data Register	LL, HH
t_{OECLKQ}	Clock-to-Q of the Output Enable Register	HH, EOUT
t_{OESUD}	Data Setup Time for the Output Enable Register	JJ, HH
t_{OEHD}	Data Hold Time for the Output Enable Register	JJ, HH
t_{OESUE}	Enable Setup Time for the Output Enable Register	KK, HH
t_{OEHE}	Enable Hold Time for the Output Enable Register	KK, HH
$t_{OECLR2Q}$	Asynchronous Clear-to-Q of the Output Enable Register	II, EOUT
$t_{OEREMCLR}$	Asynchronous Clear Removal Time for the Output Enable Register	II, HH
$t_{ORECCLR}$	Asynchronous Clear Recovery Time for the Output Enable Register	II, HH
t_{ICLKQ}	Clock-to-Q of the Input Data Register	AA, EE
t_{ISUD}	Data Setup Time for the Input Data Register	CC, AA
t_{IHD}	Data Hold Time for the Input Data Register	CC, AA
t_{ISUE}	Enable Setup Time for the Input Data Register	BB, AA
t_{IHE}	Enable Hold Time for the Input Data Register	BB, AA
t_{ICLR2Q}	Asynchronous Clear-to-Q of the Input Data Register	DD, EE
$t_{IREMCLR}$	Asynchronous Clear Removal Time for the Input Data Register	DD, AA
$t_{IRECCLR}$	Asynchronous Clear Recovery Time for the Input Data Register	DD, AA

*Note: *See Figure 2-136 on page 2-215 for more information.*

Input Register

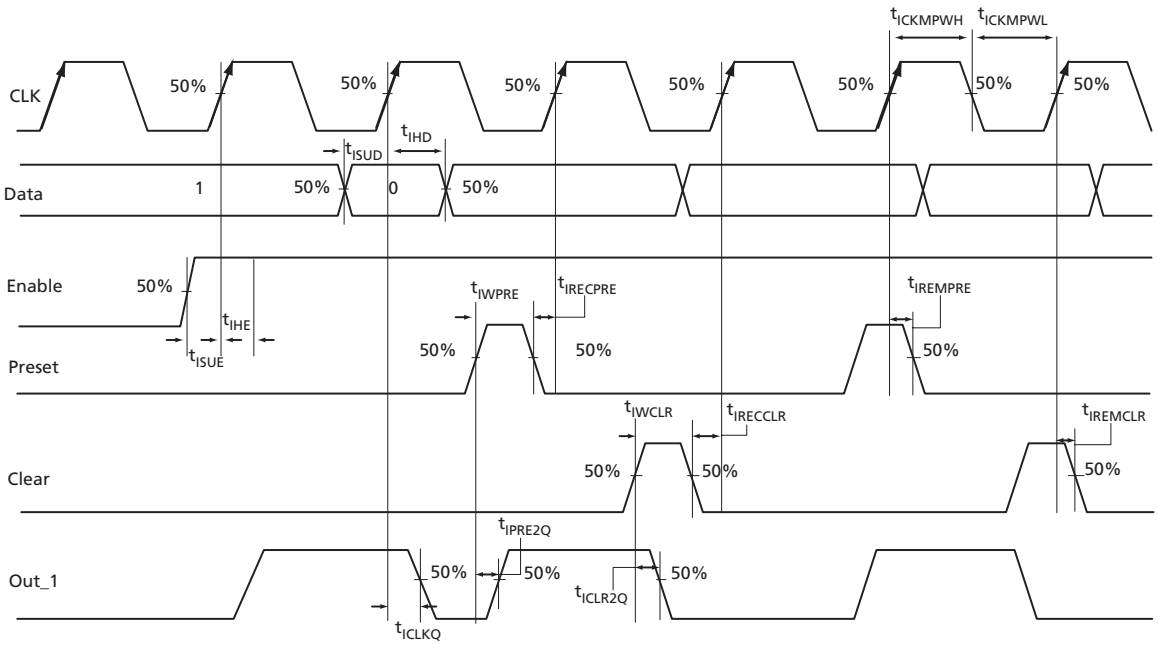


***Figure 2-137* • Input Register Timing Diagram**

Timing Characteristics

***Table 2-173* • Input Data Register Propagation Delays**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{ICLKQ}	Clock-to-Q of the Input Data Register	0.24	0.27	0.32	ns
t_{ISUD}	Data Setup Time for the Input Data Register	0.26	0.30	0.35	ns
t_{IHD}	Data Hold Time for the Input Data Register	0.00	0.00	0.00	ns
t_{ISUE}	Enable Setup Time for the Input Data Register	0.37	0.42	0.50	ns
t_{IHE}	Enable Hold Time for the Input Data Register	0.00	0.00	0.00	ns
t_{ICLR2Q}	Asynchronous Clear-to-Q of the Input Data Register	0.45	0.52	0.61	ns
t_{IPRE2Q}	Asynchronous Preset-to-Q of the Input Data Register	0.45	0.52	0.61	ns
$t_{IREMCLR}$	Asynchronous Clear Removal Time for the Input Data Register	0.00	0.00	0.00	ns
$t_{IRECCLR}$	Asynchronous Clear Recovery Time for the Input Data Register	0.22	0.25	0.30	ns
$t_{IREMPRE}$	Asynchronous Preset Removal Time for the Input Data Register	0.00	0.00	0.00	ns
$t_{IRECPRE}$	Asynchronous Preset Recovery Time for the Input Data Register	0.22	0.25	0.30	ns
t_{IWCLR}	Asynchronous Clear Minimum Pulse Width for the Input Data Register	0.22	0.25	0.30	ns
t_{IWPRE}	Asynchronous Preset Minimum Pulse Width for the Input Data Register	0.22	0.25	0.30	ns
$t_{ICKMPWH}$	Clock Minimum Pulse Width HIGH for the Input Data Register	0.36	0.41	0.48	ns
$t_{ICKMPWL}$	Clock Minimum Pulse Width LOW for the Input Data Register	0.32	0.37	0.43	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Output Register

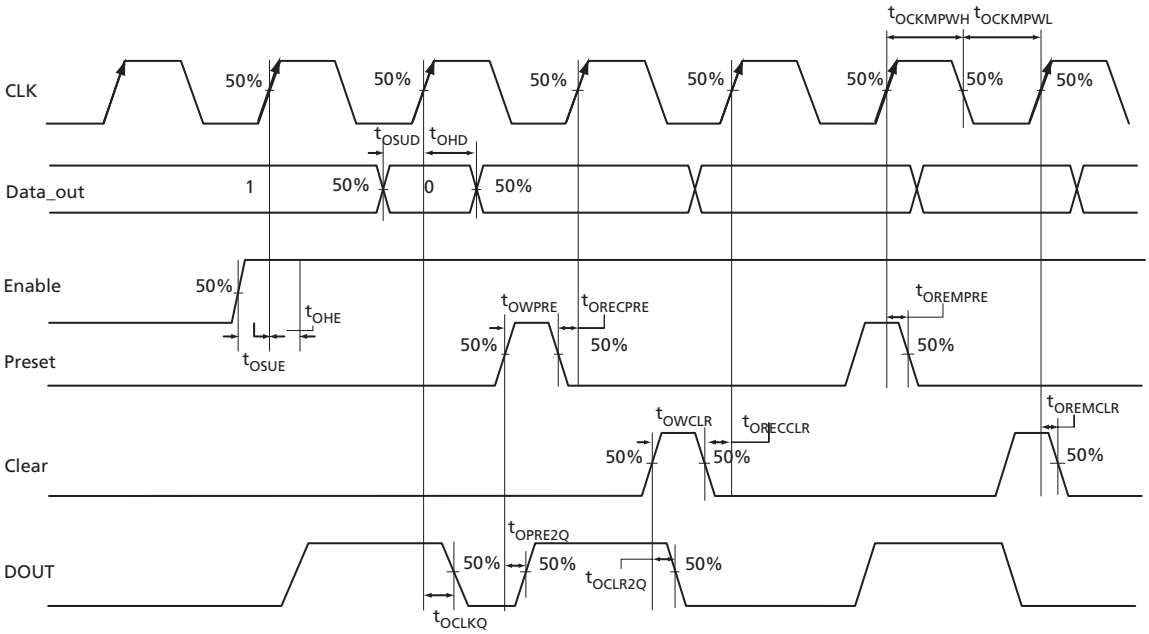


Figure 2-138 • **Output Register Timing Diagram**

Timing Characteristics

Table 2-174 • **Output Data Register Propagation Delays**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{OCLKQ}	Clock-to-Q of the Output Data Register	0.59	0.67	0.79	ns
t_{OSUD}	Data Setup Time for the Output Data Register	0.31	0.36	0.42	ns
t_{OHD}	Data Hold Time for the Output Data Register	0.00	0.00	0.00	ns
t_{OSUE}	Enable Setup Time for the Output Data Register	0.44	0.50	0.59	ns
t_{OHE}	Enable Hold Time for the Output Data Register	0.00	0.00	0.00	ns
t_{OCLR2Q}	Asynchronous Clear-to-Q of the Output Data Register	0.80	0.91	1.07	ns
t_{OPRE2Q}	Asynchronous Preset-to-Q of the Output Data Register	0.80	0.91	1.07	ns
$t_{OREMCLR}$	Asynchronous Clear Removal Time for the Output Data Register	0.00	0.00	0.00	ns
$t_{ORECCLR}$	Asynchronous Clear Recovery Time for the Output Data Register	0.22	0.25	0.30	ns
$t_{OREMPRE}$	Asynchronous Preset Removal Time for the Output Data Register	0.00	0.00	0.00	ns
$t_{ORECPRE}$	Asynchronous Preset Recovery Time for the Output Data Register	0.22	0.25	0.30	ns
t_{OWCLR}	Asynchronous Clear Minimum Pulse Width for the Output Data Register	0.22	0.25	0.30	ns
t_{OWPRE}	Asynchronous Preset Minimum Pulse Width for the Output Data Register	0.22	0.25	0.30	ns
$t_{OCKMPWH}$	Clock Minimum Pulse Width HIGH for the Output Data Register	0.36	0.41	0.48	ns
$t_{OCKMPWL}$	Clock Minimum Pulse Width LOW for the Output Data Register	0.32	0.37	0.43	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Output Enable Register

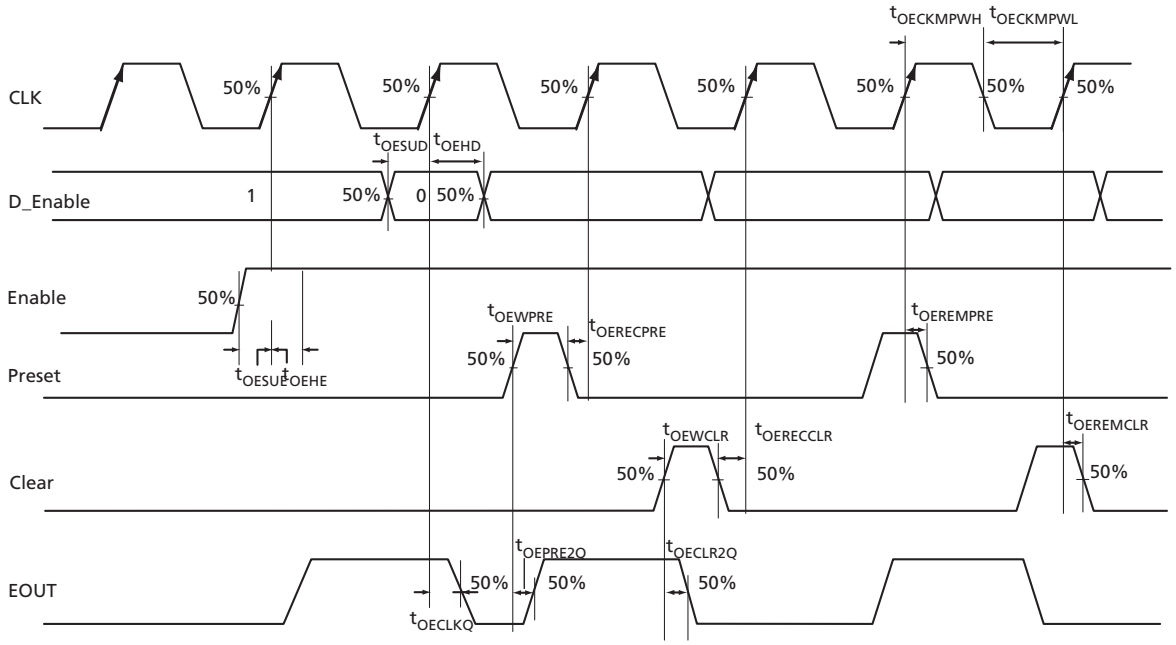


Figure 2-139 • **Output Enable Register Timing Diagram**

Timing Characteristics

Table 2-175 • **Output Enable Register Propagation Delays**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{OECLKQ}	Clock-to-Q of the Output Enable Register	0.44	0.51	0.59	ns
t_{OESUD}	Data Setup Time for the Output Enable Register	0.31	0.36	0.42	ns
t_{OEHD}	Data Hold Time for the Output Enable Register	0.00	0.00	0.00	ns
t_{OESUE}	Enable Setup Time for the Output Enable Register	0.44	0.50	0.58	ns
t_{OEHE}	Enable Hold Time for the Output Enable Register	0.00	0.00	0.00	ns
$t_{OECLR2Q}$	Asynchronous Clear-to-Q of the Output Enable Register	0.67	0.76	0.89	ns
$t_{OEPRE2Q}$	Asynchronous Preset-to-Q of the Output Enable Register	0.67	0.76	0.89	ns
$t_{OEREMCLR}$	Asynchronous Clear Removal Time for the Output Enable Register	0.00	0.00	0.00	ns
$t_{OERECCLR}$	Asynchronous Clear Recovery Time for the Output Enable Register	0.22	0.25	0.30	ns
$t_{OEREMPRE}$	Asynchronous Preset Removal Time for the Output Enable Register	0.00	0.00	0.00	ns
$t_{OERECPRE}$	Asynchronous Preset Recovery Time for the Output Enable Register	0.22	0.25	0.30	ns
t_{OEWCLR}	Asynchronous Clear Minimum Pulse Width for the Output Enable Register	0.22	0.25	0.30	ns
t_{OEWPRE}	Asynchronous Preset Minimum Pulse Width for the Output Enable Register	0.22	0.25	0.30	ns
$t_{OECKMPWH}$	Clock Minimum Pulse Width HIGH for the Output Enable Register	0.36	0.41	0.48	ns
$t_{OECKMPWL}$	Clock Minimum Pulse Width LOW for the Output Enable Register	0.32	0.37	0.43	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

DDR Module Specifications

Input DDR Module

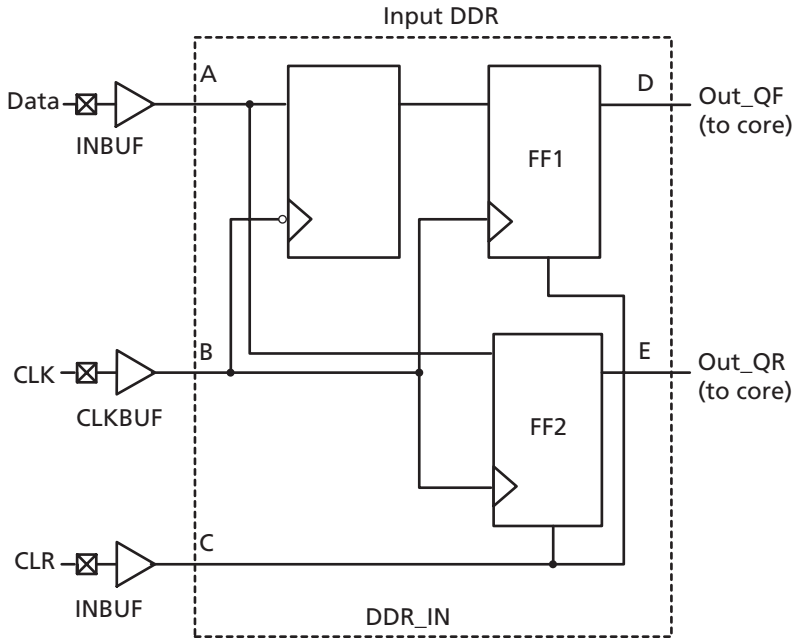


Figure 2-140 • **Input DDR Timing Model**

Table 2-176 • **Parameter Definitions**

Parameter Name	Parameter Definition	Measuring Nodes (from, to)
$t_{DDRICLKQ1}$	Clock-to-Out Out_QR	B, D
$t_{DDRICLKQ2}$	Clock-to-Out Out_QF	B, E
$t_{DDRISUD}$	Data Setup Time of DDR Input	A, B
t_{DDRIHD}	Data Hold Time of DDR Input	A, B
$t_{DDRICLR2Q1}$	Clear-to-Out Out_QR	C, D
$t_{DDRICLR2Q2}$	Clear-to-Out Out_QF	C, E
$t_{DDRIREMCLR}$	Clear Removal	C, B
$t_{DDRIRECCLR}$	Clear Recovery	C, B

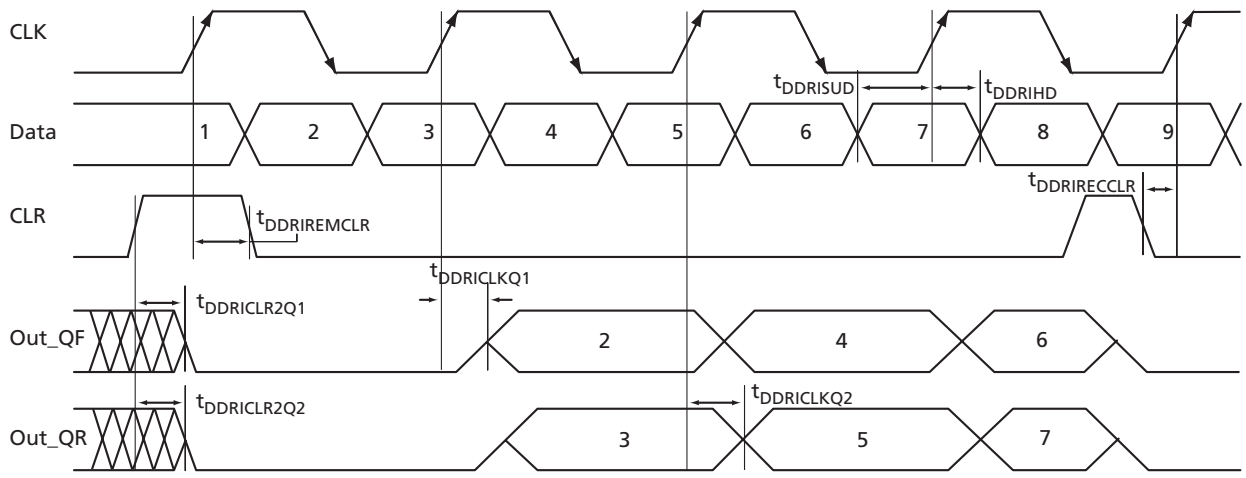


***Figure 2-141* • Input DDR Timing Diagram**

Timing Characteristics

***Table 2-177* • Input DDR Propagation Delays**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
$t_{DDRICLKQ1}$	Clock-to-Out Out_QR for Input DDR	0.39	0.44	0.52	ns
$t_{DDRICLKQ2}$	Clock-to-Out Out_QF for Input DDR	0.27	0.31	0.37	ns
$t_{DDRISUD}$	Data Setup for Input DDR	0.28	0.32	0.38	ns
t_{DDRIHD}	Data Hold for Input DDR	0.00	0.00	0.00	ns
$t_{DDRICLR2Q1}$	Asynchronous Clear-to-Out Out_QR for Input DDR	0.57	0.65	0.76	ns
$t_{DDRICLR2Q2}$	Asynchronous Clear-to-Out Out_QF for Input DDR	0.46	0.53	0.62	ns
$t_{DDRIREMCLR}$	Asynchronous Clear Removal Time for Input DDR	0.00	0.00	0.00	ns
$t_{DDRIRECCLR}$	Asynchronous Clear Recovery Time for Input DDR	0.22	0.25	0.30	ns
$t_{DDRIWCLR}$	Asynchronous Clear Minimum Pulse Width for Input DDR	0.22	0.25	0.30	ns
$t_{DDRICKMPWH}$	Clock Minimum Pulse Width HIGH for Input DDR	0.36	0.41	0.48	ns
$t_{DDRICKMPWL}$	Clock Minimum Pulse Width LOW for Input DDR	0.32	0.37	0.43	ns
$F_{DDRIMAX}$	Maximum Frequency for Input DDR	1,404	1,048	1,232	MHz

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Output DDR

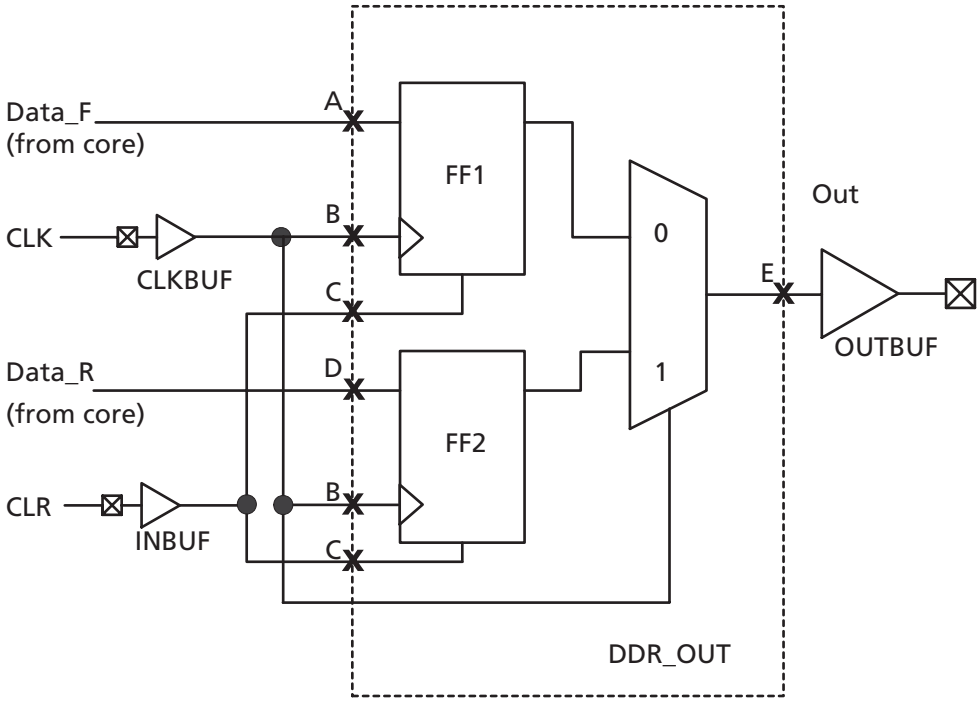


***Figure 2-142* • Output DDR Timing Model**

***Table 2-178* • Parameter Definitions**

Parameter Name	Parameter Definition	Measuring Nodes (From, To)
$t_{DDROCLKQ}$	Clock-to-Out	B, E
$t_{DDROCLR2Q}$	Asynchronous Clear-to-Out	C, E
$t_{DDROREMCLR}$	Clear Removal	C, B
$t_{DDRORECCLR}$	Clear Recovery	C, B
$t_{DDROSUD1}$	Data Setup Data_F	A, B
$t_{DDROSUD2}$	Data Setup Data_R	D, B
$t_{DDROHD1}$	Data Hold Data_F	A, B
$t_{DDROHD2}$	Data Hold Data_R	D, B

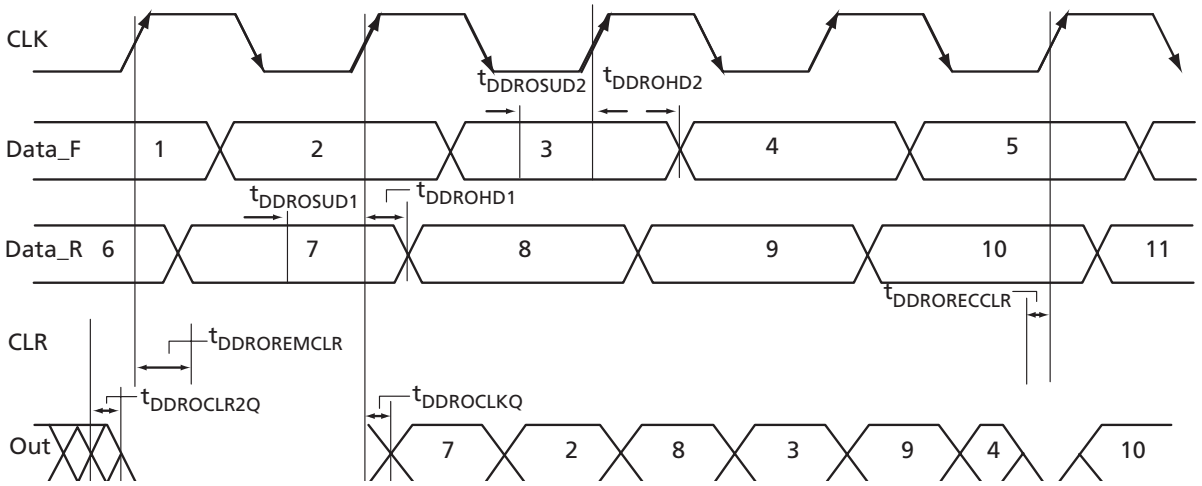


Figure 2-143 • **Output DDR Timing Diagram**

Timing Characteristics

Table 2-179 • **Output DDR Propagation Delays**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
$t_{DDROCLKQ}$	Clock-to-Out of DDR for Output DDR	0.70	0.80	0.94	ns
$t_{DDROSUD1}$	Data_F Data Setup for Output DDR	0.38	0.43	0.51	ns
$t_{DDROSUD2}$	Data_R Data Setup for Output DDR	0.38	0.43	0.51	ns
$t_{DDROHD1}$	Data_F Data Hold for Output DDR	0.00	0.00	0.00	ns
$t_{DDROHD2}$	Data_R Data Hold for Output DDR	0.00	0.00	0.00	ns
$t_{DDROCLR2Q}$	Asynchronous Clear-to-Out for Output DDR	0.80	0.91	1.07	ns
$t_{DDROREMCLR}$	Asynchronous Clear Removal Time for Output DDR	0.00	0.00	0.00	ns
$t_{DDRORECCLR}$	Asynchronous Clear Recovery Time for Output DDR	0.22	0.25	0.30	ns
$t_{DDROWCLR1}$	Asynchronous Clear Minimum Pulse Width for Output DDR	0.22	0.25	0.30	ns
$t_{DDROCKMPWH}$	Clock Minimum Pulse Width HIGH for the Output DDR	0.36	0.41	0.48	ns
$t_{DDROCKMPWL}$	Clock Minimum Pulse Width LOW for the Output DDR	0.32	0.37	0.43	ns
F_{DDOMAX}	Maximum Frequency for the Output DDR	1,048	1,232	1,404	MHz

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Pin Descriptions

Supply Pins

GND **Ground**

Ground supply voltage to the core, I/O outputs, and I/O logic.

GNDQ **Ground (quiet)**

Quiet ground supply voltage to input buffers of I/O banks. Within the package, the GNDQ plane is decoupled from the simultaneous switching noise originated from the output buffer ground domain. This minimizes the noise transfer within the package and improves input signal integrity. GNDQ needs to always be connected on the board to GND. Note: In FG256, FG484, and FG676 packages, GNDQ and GND pins are connected within the package and are labeled as GND pins in the respective package pin assignment tables.

ADCGNDREF **Analog Reference Ground**

Analog ground reference used by the ADC. This pad should be connected to a quiet analog ground.

GNDA **Ground (analog)**

Quiet ground supply voltage to the Analog Block of Fusion devices. The use of a separate analog ground helps isolate the analog functionality of the Fusion device from any digital switching noise. A 0.2 V maximum differential voltage between GND and GNDA/GNDQ should apply to system implementation.

GNDAQ **Ground (analog quiet)**

Quiet ground supply voltage to the analog I/O of Fusion devices. The use of a separate analog ground helps isolate the analog functionality of the Fusion device from any digital switching noise. A 0.2 V maximum differential voltage between GND and GNDA/GNDQ should apply to system implementation. Note: In FG256, FG484, and FG676 packages, GNDAQ and GNDA pins are connected within the package and are labeled as GNDA pins in the respective package pin assignment tables. In FG256 and

GNDNVM **Flash Memory Ground**

Ground supply used by the Fusion device's flash memory block module(s).

GNDOSC **Oscillator Ground**

Ground supply for both integrated RC oscillator and crystal oscillator circuit.

V_{CC15A} **Analog Power Supply (1.5 V)**

1.5 V clean analog power supply input for use by the 1.5 V portion of the analog circuitry.

V_{CC33A} **Analog Power Supply (3.3 V)**

3.3 V clean analog power supply input for use by the 3.3 V portion of the analog circuitry.

V_{CC33N} **Negative 3.3 V Output**

This is the –3.3 V output from the voltage converter. A 2.2 µF capacitor must be connected from this pin to ground.

$V_{CC33PMP}$ **Analog Power Supply (3.3 V)**

3.3 V clean analog power supply input for use by the analog charge pump. To avoid high current draw, $V_{CC33PMP}$ should be powered up simultaneously with or after V_{CC33A}.

V_{CCNVM} **Flash Memory Block Power Supply (1.5 V)**

1.5 V power supply input used by the Fusion device's flash memory block module(s). To avoid high current draw, V_{CC} should be powered up before or simultaneously with V_{CCNVM}.

V_{CCOSC} **Oscillator Power Supply (3.3 V)**

Power supply for both integrated RC oscillator and crystal oscillator circuit. The internal 100 MHz oscillator, powered by the V_{CCOSC} pin, is needed for device programming, operation of the V_{DDN33}

pump, and eNVM operation. V_{CCOSC} is off only when V_{CCA} is off. V_{CCOSC} must be powered whenever the Fusion device needs to function.

V_{CC} Core Supply Voltage

Supply voltage to the FPGA core, nominally 1.5 V. V_{CC} is also required for powering the JTAG state machine, in addition to V_{JTAG}. Even when a Fusion device is in bypass mode in a JTAG chain of interconnected devices, both V_{CC} and V_{JTAG} must remain powered to allow JTAG signals to pass through the Fusion device.

V_{CCI}B*x* I/O Supply Voltage

Supply voltage to the bank's I/O output buffers and I/O logic. B*x* is the I/O bank number. There are either four (AFS090 and AFS250) or five (AFS600 and AFS1500) I/O banks on the Fusion devices plus a dedicated V_{JTAG} bank.

Each bank can have a separate V_{CCI} connection. All I/Os in a bank will run off the same V_{CCI}Bx supply. V_{CCI} can be 1.5 V, 1.8 V, 2.5 V, or 3.3 V, nominal voltage. Unused I/O banks should have their corresponding V_{CCI} pins tied to GND.

$V_{CCPLA/B}$ PLL Supply Voltage

Supply voltage to analog PLL, nominally 1.5 V, where A and B refer to the PLL. AFS090 and AFS250 each have a single PLL. The AFS600 and AFS1500 devices each have two PLLs. Actel recommends tying V_{CCPLX} to V_{CC} and using proper filtering circuits to decouple V_{CC} noise from PLL.

If unused, $V_{CCPLA/B}$ should be tied to GND.

$V_{COMPLA/B}$ Ground for West and East PLL

V_{COMPLA} is the ground of the west PLL (CCC location F) and V_{COMPLB} is the ground of the east PLL (CCC location C).

V_{JTAG} JTAG Supply Voltage

Fusion devices have a separate bank for the dedicated JTAG pins. The JTAG pins can be run at any voltage from 1.5 V to 3.3 V (nominal). Isolating the JTAG power supply in a separate I/O bank gives greater flexibility in supply selection and simplifies power supply and PCB design. If the JTAG interface is neither used nor planned to be used, the V_{JTAG} pin together with the TRST pin could be tied to GND. It should be noted that V_{CC} is required to be powered for JTAG operation; V_{JTAG} alone is insufficient. If a Fusion device is in a JTAG chain of interconnected boards and it is desired to power down the board containing the Fusion device, this may be done provided both V_{JTAG} and V_{CC} to the Fusion part remain powered; otherwise, JTAG signals will not be able to transition the Fusion device, even in bypass mode.

V_{PUMP} Programming Supply Voltage

Fusion devices support single-voltage ISP programming of the configuration flash and FlashROM. For programming, V_{PUMP} should be in the 3.3 V +/-5% range. During normal device operation, V_{PUMP} can be left floating or can be tied to any voltage between 0 V and 3.6 V.

When the V_{PUMP} pin is tied to ground, it shuts off the charge pump circuitry, resulting in no sources of oscillation from the charge pump circuitry.

For proper programming, 0.01 µF and 0.33 µF capacitors (both rated at 16 V) are to be connected in parallel across V_{PUMP} and GND, and positioned as close to the FPGA pins as possible.

User-Defined Supply Pins

V_{REF} I/O Voltage Reference

Reference voltage for I/O minibanks. Both AFS600 and AFS1500 (north bank only) support Actel Pro I/O. These I/O banks support voltage reference standard I/O. The V_{REF} pins are configured by the user from regular I/Os, and any I/O in a bank, except JTAG I/Os, can be designated as the voltage reference I/O. Only certain I/O standards require a voltage reference—HSTL (I) and (II), SSTL2 (I) and (II), SSTL3 (I) and (II), and GTL/GTL+. One V_{REF} pin can support the number of I/Os available in its minibank.

VAREF **Analog Reference Voltage**

The Fusion device can be configured to generate a 2.56 V internal reference voltage that can be used by the ADC. While using the internal reference, the reference voltage is output on the VAREF pin for use as a system reference. If a different reference voltage is required, it can be supplied by an external source and applied to this pin. The valid range of values that can be supplied to the ADC is 1.0 V to 3.3 V. When VAREF is internally generated by the Fusion device, a bypass capacitor must be connected from this pin to ground. The value of the bypass capacitor should be between 3.3 µF and 22 µF, which is based on the needs of the individual designs. The choice of the capacitor value has an impact on the settling time it takes the VAREF signal to reach the required specification of 2.56 V to initiate valid conversions by the ADC. If the lower capacitor value is chosen, the settling time required for VAREF to achieve 2.56 V will be shorter than when selecting the larger capacitor value. The above range of capacitor values supports the accuracy specification of the ADC, which is detailed in the datasheet. Designers choosing the smaller capacitor value will not obtain as much margin in the accuracy as that achieved with a larger capacitor value. Depending on the capacitor value selected in the Analog System Builder, a tool in Libero IDE, an automatic delay circuit will be generated using logic tiles available within the FPGA to ensure that VAREF has achieved the 2.56 V value. Actel recommends customers use 10 µF as the value of the bypass capacitor. Designers choosing to use an external VAREF need to ensure that a stable and clean VAREF source is supplied to the VAREF pin before initiating conversions by the ADC. Designers should also make sure that the ADCRESET signal is deasserted before initiating valid conversions.[2]

User Pins

I/O **User Input/Output**

The I/O pin functions as an input, output, tristate, or bidirectional buffer. Input and output signal levels are compatible with the I/O standard selected. Unused I/O pins are configured as inputs with pull-up resistors.

During programming, I/Os become tristated and weakly pulled up to V_{CCI}. With the V_{CCI} and V_{CC} supplies continuously powered up, when the device transitions from programming to operating mode, the I/Os get instantly configured to the desired user configuration.

A*xy* **Analog Input/Output**

Analog I/O pin, where *x* is the analog pad type (C = current pad, G = Gate driver pad, T = Temperature pad, V = Voltage pad) and *y* is the Analog Quad number (0 to 9). There is a minimum 1 MΩ to ground on AV, AC, and AT. This pin can be left floating when it is unused.

ATRTN*x* **Temperature Monitor Return**

AT returns are the returns for the temperature sensors. The cathode terminal of the external diodes should be connected to these pins. There is one analog return pin for every two Analog Quads. The *x* in the ATRTN*x* designator indicates the quad pairing (*x* = 0 for AQ1 and AQ2, *x* = 1 for AQ2 and AQ3, ..., *x* = 4 for AQ8 and AQ9). The signals that drive these pins are called out as ATRETURN*xy* in the software (where *x* and *y* refer to the quads that share the return signal). ATRTN is internally connected to ground. It can be left floating when it is unused. The maximum capacitance allowed across the AT pins is 500 pF.

GL **Globals**

GL I/Os have access to certain clock conditioning circuitry (and the PLL) and/or have direct access to the global network (spines). Additionally, the global I/Os can be used as Pro I/Os since they have identical capabilities. Unused GL pins are configured as inputs with pull-up resistors. See more detailed descriptions of global I/O connectivity in the "Clock Conditioning Circuits" section on page 2-24.

2. The ADC is functional with an external reference down to 1V, however to meet the performance parameters highlighted in the datasheet refer to the VAREF specification in Table 3-2 on page 3-3.

Refer to the "User I/O Naming Convention" section on page 2-159 for a description of naming of global pins.

JTAG Pins

Fusion devices have a separate bank for the dedicated JTAG pins. The JTAG pins can be run at any voltage from 1.5 V to 3.3 V (nominal). V_{CC} must also be powered for the JTAG state machine to operate, even if the device is in bypass mode; V_{JTAG} alone is insufficient. Both V_{JTAG} and V_{CC} to the Fusion part must be supplied to allow JTAG signals to transition the Fusion device.

Isolating the JTAG power supply in a separate I/O bank gives greater flexibility with supply selection and simplifies power supply and PCB design. If the JTAG interface is neither used nor planned to be used, the V_{JTAG} pin together with the TRST pin could be tied to GND.

TCK Test Clock

Test clock input for JTAG boundary scan, ISP, and UJTAG. The TCK pin does not have an internal pull-up/-down resistor. If JTAG is not used, Actel recommends tying off TCK to GND or V_{JTAG} through a resistor placed close to the FPGA pin. This prevents JTAG operation in case TMS enters an undesired state.

Note that to operate at all V_{JTAG} voltages, 500 Ω to 1 kΩ will satisfy the requirements. Refer to Table 2-180 for more information.

***Table 2-180* • Recommended Tie-Off Values for the TCK and TRST Pins**

V_{JTAG}	Tie-Off Resistance[2, 3]
V_{JTAG} at 3.3 V	200 Ω to 1 kΩ
V_{JTAG} at 2.5 V	200 Ω to 1 kΩ
V_{JTAG} at 1.8 V	500 Ω to 1 kΩ
V_{JTAG} at 1.5 V	500 Ω to 1 kΩ

Notes:

1. *Equivalent parallel resistance if more than one device is on JTAG chain.*
2. *The TCK pin can be pulled up/down.*
3. *The TRST pin can only be pulled down.*

TDI Test Data Input

Serial input for JTAG boundary scan, ISP, and UJTAG usage. There is an internal weak pull-up resistor on the TDI pin.

TDO Test Data Output

Serial output for JTAG boundary scan, ISP, and UJTAG usage.

TMS Test Mode Select

The TMS pin controls the use of the IEEE1532 boundary scan pins (TCK, TDI, TDO, TRST). There is an internal weak pull-up resistor on the TMS pin.

TRST Boundary Scan Reset Pin

The TRST pin functions as an active low input to asynchronously initialize (or reset) the boundary scan circuitry. There is an internal weak pull-up resistor on the TRST pin. If JTAG is not used, an external pull-down resistor could be included to ensure the TAP is held in reset mode. The resistor values must be chosen from Table 2-180 and must satisfy the parallel resistance value requirement. The values in Table 2-180 correspond to the resistor recommended when a single device is used and to the equivalent parallel resistor when multiple devices are connected via a JTAG chain.

In critical applications, an upset in the JTAG circuit could allow entering an undesired JTAG state. In such cases, Actel recommends tying off TRST to GND through a resistor placed close to the FPGA pin.

Note that to operate at all V_{JTAG} voltages, 500 Ω to 1 kΩ will satisfy the requirements.

Special Function Pins

NC **No Connect**

This pin is not connected to circuitry within the device. These pins can be driven to any voltage or can be left floating with no effect on the operation of the device.

DC **Don't Connect**

This pin should not be connected to any signals on the PCB. These pins should be left unconnected.

NCAP **Negative Capacitor**

Negative Capacitor is where the negative terminal of the charge pump capacitor is connected. A capacitor, with a 2.2 µF recommended value, is required to connect between PCAP and NCAP.

PCAP **Positive Capacitor**

Positive Capacitor is where the positive terminal of the charge pump capacitor is connected. A capacitor, with a 2.2 µF recommended value, is required to connect between PCAP and NCAP.

PUB **Push Button**

Push button is the connection for the external momentary switch used to turn on the 1.5 V voltage regulator and can be floating if not used.

PTBASE **Pass Transistor Base**

Pass Transistor Base is the control signal of the voltage regulator. This pin should be connected to the base of the external pass transistor used with the 1.5 V internal voltage regulator and can be floating if not used.

PTEM **Pass Transistor Emitter**

Pass Transistor Emitter is the feedback input of the voltage regulator.

This pin should be connected to the emitter of the external pass transistor used with the 1.5 V internal voltage regulator and can be floating if not used.

XTAL1 **Crystal Oscillator Circuit Input**

Input to crystal oscillator circuit. Pin for connecting external crystal, ceramic resonator, RC network, or external clock input. When using an external crystal or ceramic oscillator, external capacitors are also recommended (Please refer to the crystal oscillator manufacturer for proper capacitor value).

If using external RC network or clock input, XTAL1 should be used and XTAL2 left unconnected.

XTAL2 **Crystal Oscillator Circuit Input**

Input to crystal oscillator circuit. Pin for connecting external crystal, ceramic resonator, RC network, or external clock input. When using an external crystal or ceramic oscillator, external capacitors are also recommended (Please refer to the crystal oscillator manufacturer for proper capacitor value).

If using external RC network or clock input, XTAL1 should be used and XTAL2 left unconnected.

Security

Fusion devices have a built-in 128-bit AES decryption core. The decryption core facilitates secure, in-system programming of the FPGA core array fabric and the FlashROM. The FlashROM and the FPGA core fabric can be programmed independently from each other, allowing the FlashROM to be updated without the need for change to the FPGA core fabric. The AES master key is stored in on-chip nonvolatile memory (flash). The AES master key can be preloaded into parts in a secure programming environment (such as the Actel in-house programming center), and then "blank" parts can be shipped to an untrusted programming or manufacturing center for final personalization with an AES-encrypted bitstream. Late stage product changes or personalization can be implemented easily and securely by simply sending a STAPL file with AES-encrypted data.

Secure remote field updates over public networks (such as the Internet) are possible by sending and programming a STAPL file with AES-encrypted data. For more information, refer to the *Fusion Security* application note.

128-Bit AES Decryption

The 128-bit AES standard (FIPS-197) block cipher is the National Institute of Standards and Technology (NIST) replacement for DES (Data Encryption Standard FIPS46-2). AES has been designed to protect sensitive government information well into the 21st century. It replaces the aging DES, which NIST adopted in 1977 as a Federal Information Processing Standard used by federal agencies to protect sensitive, unclassified information. The 128-bit AES standard has 3.4×10^{38} possible 128-bit key variants, and it has been estimated that it would take 1,000 trillion years to crack 128-bit AES cipher text using exhaustive techniques. Keys are stored (securely) in Fusion devices in nonvolatile flash memory. All programming files sent to the device can be authenticated by the part prior to programming to ensure that bad programming data is not loaded into the part that may possibly damage it. All programming verification is performed on-chip, ensuring that the contents of Fusion devices remain secure.

AES decryption can also be used on the 1,024-bit FlashROM to allow for secure remote updates of the FlashROM contents. This allows for easy, secure support for subscription model products. See the application note *Fusion Security* for more details.

AES for Flash Memory

AES decryption can also be used on the flash memory blocks. This allows for the secure update of the flash memory blocks. During runtime, the encrypted data can be clocked in via the JTAG interface. The data can be passed through the internal AES decryption engine, and the decrypted data can then be stored in the flash memory block.

Programming

Programming can be performed using various programming tools, such as Silicon Sculptor II (BP Micro Systems) or FlashPro3 (Actel).

The user can generate STP programming files from the Designer software and can use these files to program a device.

Fusion devices can be programmed in-system. During programming, V_{CCOSC} is needed in order to power the internal 100 MHz oscillator. This oscillator is used as a source for the 20 MHz oscillator that is used to drive the charge pump for programming.

ISP

Fusion devices support IEEE 1532 ISP via JTAG and require a single V_{PUMP} voltage of 3.3 V during programming. In addition, programming via a microcontroller in a target system can be achieved. Refer to the standard or the *In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3* document for more details.

JTAG IEEE 1532

Programming with IEEE 1532

Fusion devices support the JTAG-based IEEE1532 standard for ISP. As part of this support, when a Fusion device is in an unprogrammed state, all user I/O pins are disabled. This is achieved by keeping the global IO_EN signal deactivated, which also has the effect of disabling the input buffers. Consequently, the SAMPLE instruction will have no effect while the Fusion device is in this unprogrammed state—different behavior from that of the ProASIC$^{PLUS®}$ device family. This is done because SAMPLE is defined in the IEEE1532 specification as a noninvasive instruction. If the input buffers were to be enabled by SAMPLE temporarily turning on the I/Os, then it would not truly be a noninvasive instruction. Refer to the standard or the *In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3* document for more details.

Boundary Scan

Fusion devices are compatible with IEEE Standard 1149.1, which defines a hardware architecture and the set of mechanisms for boundary scan testing. The basic Fusion boundary scan logic circuit is composed of the test access port (TAP) controller, test data registers, and instruction register

(Figure 2-144 on page 2-231). This circuit supports all mandatory IEEE 1149.1 instructions (EXTEST, SAMPLE/PRELOAD, and BYPASS) and the optional IDCODE instruction (Table 2-182 on page 2-231).

Each test section is accessed through the TAP, which has five associated pins: TCK (test clock input), TDI, TDO (test data input and output), TMS (test mode selector), and TRST (test reset input). TMS, TDI, and TRST are equipped with pull-up resistors to ensure proper operation when no input data is supplied to them. These pins are dedicated for boundary scan test usage. Refer to the "JTAG Pins" section on page 2-227 for pull-up/-down recommendations for TDO and TCK pins. The TAP controller is a 4-bit state machine (16 states) that operates as shown in Figure 2-144 on page 2-231. The 1s and 0s represent the values that must be present on TMS at a rising edge of TCK for the given state transition to occur. IR and DR indicate that the instruction register or the data register is operating in that state.

Table 2-181 • **TRST and TCK Pull-Down Recommendations**

V_{JTAG}	Tie-Off Resistance*
V_{JTAG} at 3.3 V	200 Ω to 1 kΩ
V_{JTAG} at 2.5 V	200 Ω to 1 kΩ
V_{JTAG} at 1.8 V	500 Ω to 1 kΩ
V_{JTAG} at 1.5 V	500 Ω to 1 kΩ

*Note: *Equivalent parallel resistance if more than one device is on JTAG chain.*

The TAP controller receives two control inputs (TMS and TCK) and generates control and clock signals for the rest of the test logic architecture. On power-up, the TAP controller enters the Test-Logic-Reset state. To guarantee a reset of the controller from any of the possible states, TMS must remain HIGH for five TCK cycles. The TRST pin can also be used to asynchronously place the TAP controller in the Test-Logic-Reset state.

Fusion devices support three types of test data registers: bypass, device identification, and boundary scan. The bypass register is selected when no other register needs to be accessed in a device. This speeds up test data transfer to other devices in a test data path. The 32-bit device identification register is a shift register with four fields (LSB, ID number, part number, and version). The boundary scan register observes and controls the state of each I/O pin. Each I/O cell has three boundary scan register cells, each with a serial-in, serial-out, parallel-in, and parallel-out pin.

The serial pins are used to serially connect all the boundary scan register cells in a device into a boundary scan register chain, which starts at the TDI pin and ends at the TDO pin. The parallel ports are connected to the internal core logic I/O tile and the input, output, and control ports of an I/O buffer to capture and load data into the register to control or observe the logic state of each I/O.

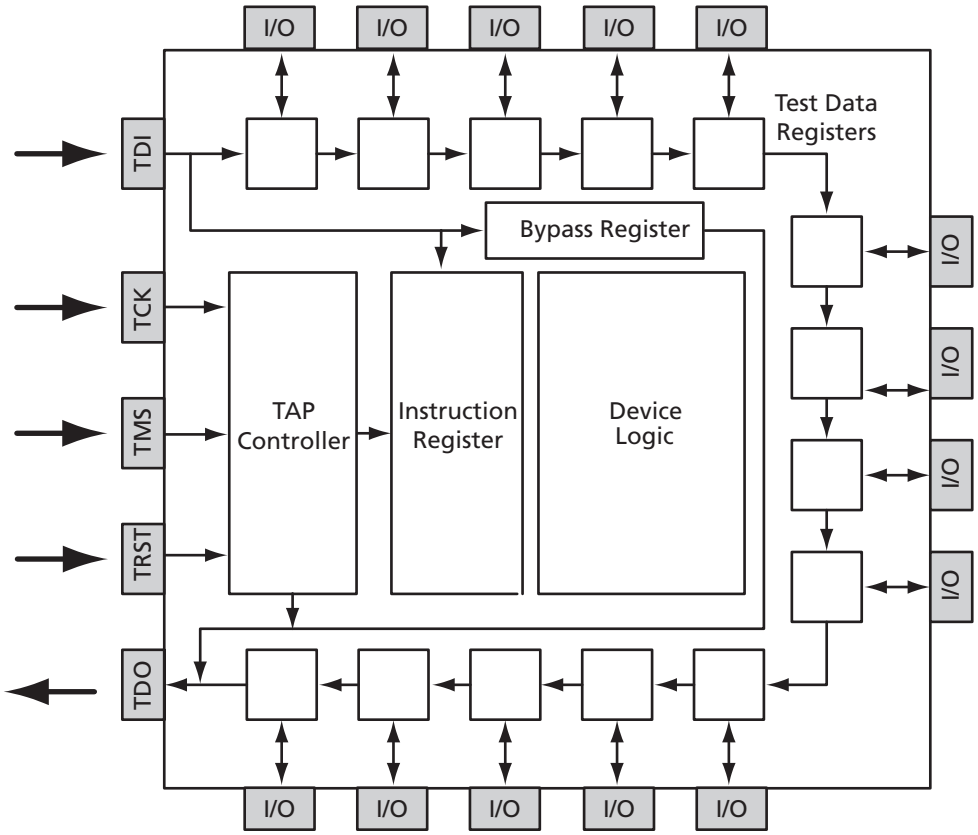


***Figure 2-144* • Boundary Scan Chain in Fusion**

***Table 2-182* • Boundary Scan Opcodes**

	Hex Opcode
EXTEST	00
HIGHZ	07
USERCODE	0E
SAMPLE/PRELOAD	01
IDCODE	0F
CLAMP	05
BYPASS	FF

IEEE 1532 Characteristics

JTAG timing delays do not include JTAG I/Os. To obtain complete JTAG timing, add I/O buffer delays to the corresponding standard selected; refer to the I/O timing characteristics in the "User I/Os" section on page 2-133 for more details.

Timing Characteristics

Table 2-183 • **JTAG 1532**
Commercial Temperature Range Conditions: T_J = 70°C, Worst-Case V_{CC} = 1.425 V

Parameter	Description	–2	–1	Std.	Units
t_{DISU}	Test Data Input Setup Time	0.50	0.57	0.67	ns
t_{DIHD}	Test Data Input Hold Time	1.00	1.13	1.33	ns
t_{TMSSU}	Test Mode Select Setup Time	0.50	0.57	0.67	ns
t_{TMDHD}	Test Mode Select Hold Time	1.00	1.13	1.33	ns
t_{TCK2Q}	Clock to Q (data out)	6.00	6.80	8.00	ns
t_{RSTB2Q}	Reset to Q (data out)	20.00	22.67	26.67	ns
F_{TCKMAX}	TCK Maximum Frequency	25.00	22.00	19.00	MHz
$t_{TRSTREM}$	ResetB Removal Time	0.00	0.00	0.00	ns
$t_{TRSTREC}$	ResetB Recovery Time	0.20	0.23	0.27	ns

Note: For the derating values at specific junction temperature and voltage supply levels, refer to Table 3-7 on page 3-9.

Part Number and Revision Date

Part Number 51700092-014-1
Revised July 2009

List of Changes

The following table lists critical changes that were made in the current version of the document.

Previous Version	Changes in Current Version (v2.0)	Page
Preliminary v1.7 (October 2008)	The MicroBlade and Fusion datasheets have been combined. Pigeon Point information is new. CoreMP7 support was removed since it is no longer offered. –F was removed from the datasheet since it is no longer offered. The operating temperature was changed from ambient to junction to better reflect actual conditions of operations. Commercial: 0°C to 85°C Industrial: –40°C to 100°C The version number category was changed from Preliminary to Production, which means the datasheet contains information based on final characterization. The version number changed from Preliminary v1.7 to v2.0.	N/A
	The phrase "Commercial-Case Conditions" in timing table titles was changed to "Commercial Temperature Range Conditions."	N/A
	The "Crystal Oscillator" section was updated significantly. Please review carefully.	2-22
	The "Real-Time Counter (part of AB macro)" section was updated significantly. Please review carefully.	2-35
	There was a typo in Table 2-19 • Flash Memory Block Pin Names for the ERASEPAGE description; it was the same as DISCARDPAGE. As as a result, the ERASEPAGE description was updated.	2-43
	The $t_{FMAXCLKNVM}$ parameter was updated in Table 2-25 • Flash Memory Block Timing.	2-55
	Table 2-31 • RAM4K9 and Table 2-32 • RAM512X18 were updated.	2-71 to 2-71
	In Table 2-36 • Analog Block Pin Description, the Function description for PWRDWN was changed from "Comparator power-down if 1" to "ADC comparator power-down if 1. When asserted, the ADC will stop functioning, and the digital portion of the analog block will continue operating. This may result in invalid status flags from the analog block. Therefore, Actel does not recommend asserting the PWRDWN pin."	2-82
	Figure 2-76 • Gate Driver Example was updated.	2-95
	The "ADC Configuration Description" section was updated. Please review carefully.	2-102
	Figure 2-85 • Intra-Conversion Timing Diagram and Figure 2-86 • Injected-Conversion Timing Diagram are new.	2-108
	The "Typical Performance Characteristics" section is new.	2-116
	Table 2-46 • Analog Channel Specifications was significantly updated.	2-118

Previous Version	Changes in Current Version (v2.0)	Page
Preliminary v1.7 (continued)	Table 2-47 • ADC Characteristics in Direct Input Mode was significantly updated.	2-124
	In Table 2-50 • Analog Channel Accuracy: Monitoring Standard Positive Voltages, note 1 was updated.	2-125
	In Table 2-49 • Calibrated Analog Channel Accuracy [1,2,3], note 2 was updated.	
	In Table 2-51 • ACM Address Decode Table for Analog Quad, bit 89 was removed.	2-127
	The data in the 2.5 V LCMOS and LVCMOS 2.5 V / 5.0 V rows were updated in Table 2-72 • Fusion Standard and Advanced I/O – Hot-Swap and 5 V Input Tolerance Capabilities.	2-144
	In Table 2-75 • Fusion Standard I/O Standards—OUT_DRIVE Settings, LVCMOS 1.5 V, for OUT_DRIVE 2, was changed from a dash to a checkmark.	2-153
	The "V_{CC15A} Analog Power Supply (1.5 V)" definition was changed from "A 1.5 V analog power supply input should be used to provide this input" to "1.5 V clean analog power supply input for use by the 1.5 V portion of the analog circuitry."	2-224
	In the "$V_{CC33PMP}$ Analog Power Supply (3.3 V)" pin description, the following text was changed from "$V_{CC33PMP}$ should be powered up before or simultaneously with V_{CC33A}" to "$V_{CC33PMP}$ should be powered up simultaneously with or after V_{CC33A}."	2-224
	The "V_{CCOSC} Oscillator Power Supply (3.3 V)" section was updated to include information about when to power the pin.	2-224
	In the "128-Bit AES Decryption" section, FIPS-192 was incorrect and changed to FIPS-197.	2-229
	The note in Table 2-81 • Fusion Standard and Advanced I/O Attributes vs. I/O Standard Applications was updated.	2-157
	For 1.5 V LVCMOS, the V_{IL} and V_{IH} parameters, 0.30 * V_{CCI} was changed to 0.35 * V_{CCI} and 0.70 * V_{CCI} was changed to 0.65 * V_{CCI} in Table 2-83 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions, Table 2-84 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions, and Table 2-85 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions. In Table 2-84 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions, the V_{IH} max column was updated.	2-165 to 2-166
	Table 2-86 • Summary of Maximum and Minimum DC Input Levels Applicable to Commercial and Industrial Conditions was updated to include notes 3 and 4. The temperature ranges were also updated in notes 1 and 2.	2-166
	The titles in Table 2-89 • Summary of I/O Timing Characteristics – Software Default Settings to Table 2-91 • Summary of I/O Timing Characteristics – Software Default Settings were updated to "V_{CCI} = I/O Standard Dependent."	2-168 to 2-170
	Below Table 2-95 • I/O Short Currents IOSH/IOSL, the paragraph was updated to change 110°C to 100°C and three months was changed to six months.	2-174

Previous Version	Changes in Current Version (v2.0)	Page
Preliminary v1.7 (continued)	Table 2-96 • Short Current Event Duration before Failure was updated to remove 110°C data.	2-176
	In Table 2-98 • I/O Input Rise Time, Fall Time, and Related I/O Reliability, LVTTL/LVCMOS rows were changed from 110°C to 100°C.	2-176
Advance v1.6 (August 2008)	For the V_{IL} and V_{IH} parameters, 0.30 * V_{CCI} was changed to 0.35 * V_{CCI} and 0.70 * V_{CCI} was changed to 0.65 * V_{CCI} in Table 2-123 • Minimum and Maximum DC Input and Output Levels.	2-194
	The version number category was changed from Advance to Preliminary, which means the datasheet contains information based on simulation and/or initial characterization. The information is believed to be correct, but changes are possible.	N/A
	The following updates were made to Table 2-38 • Temperature Data Format: Temperature — Digital Output 213 — 00 1111 1101 283 — 01 0001 1011 358 — 01 0110 0110 – only the digital output was updated. Temperature 358 remains in the temperature column.	2-98
	In Advance v1.2, the "V_{AREF} Analog Reference Voltage" pin description was significantly updated but the change was not noted in the change table.	2-226
Advance v1.5 (July 2008)	The references to the *Peripherals User's Guide* in the "No-Glitch MUX (NGMUX)" section and "Voltage Regulator Power Supply Monitor (VRPSM)" section were changed to *Fusion Handbook*.	2-32, 2-42
Advance v1.4 (July 2008)	The title of the datasheet changed from Actel Programmable System Chips to Actel Fusion Mixed-Signal FPGAs. In addition, all instances of programmable system chip were changed to mixed-signal FPGA.	N/A
Advance v1.2 (June 2008)	The "ADC Description" section was significantly updated. Please review carefully.	2-102
Advance v1.1 (May 2008)	Table 2-25 • Flash Memory Block Timing was significantly updated.	2-55
	The "V_{AREF} Analog Reference Voltage" pin description section was significantly update. Please review it carefully.	2-226
	Table 2-45 • ADC Interface Timingwas significantly updated.	2-110
	Table 2-56 • Direct Analog Input Switch Control Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3) was significantly updated.	2-131
	The following sentence was deleted from the "Voltage Monitor" section: The Analog Quad inputs are tolerant up to 12 V + 10%.	2-86
Advance v1.0 (January 2008)	The following text was incorrect and therefore deleted: **VCC33A** **Analog Power Filter** Analog power pin for the analog power supply low-pass filter. An external 100 pF capacitor should be connected between this pin and ground. There is still a description of V_{CC33A} on page 2-224.	2-204
Advance v0.9 (October 2007)	All Timing Characteristics tables were updated. For the Differential I/O Standards, the Standard I/O support tables are new.	N/A
	Table 2-3 • Array Coordinates was updated to change the max x and y values	2-9
	Table 2-12 • Fusion CCC/PLL Specification was updated.	2-31
	A note was added to Table 2-16 · RTC ACM Memory Map.	2-37

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.9 (continued)	A reference to the Peripheral's User's Guide was added to the "Voltage Regulator Power Supply Monitor (VRPSM)" section.	2-42
	In Table 2-25 • Flash Memory Block Timing, the commercial conditions were updated.	2-55
	In Table 2-26 • FlashROM Access Time, the commercial conditions were missing and have been added below the title of the table.	2-58
	In Table 2-36 • Analog Block Pin Description, the function description was updated for the ADCRESET.	2-82
	In the "Voltage Monitor" section, the following sentence originally had ± 10% and it was changed to +10%. The Analog Quad inputs are tolerant up to 12 V + 10%. In addition, this statement was deleted from the datasheet: Each I/O will draw power when connected to power (3 mA at 3 V).	2-86
	The "Terminology" section is new.	2-88
	The "Current Monitor" section was significantly updated. Figure 2-72 • Timing Diagram for Current Monitor Strobe to Figure 2-74 • Negative Current Monitor and Table 2-37 • Recommended Resistor for Different Current Range Measurement are new.	2-90
	The "ADC Description" section was updated to add the "Terminology" section.	2-93
	In the "Gate Driver" section, 25 mA was changed to 20 mA and 1.5 MHz was changed to 1.3 MHz. In addition, the following sentence was deleted: The maximum AG pad switching frequency is 1.25 MHz.	2-94
	The "Temperature Monitor" section was updated to rewrite most of the text and add Figure 2-78, Figure 2-79, and Table 2-38 • Temperature Data Format.	2-96
	In Table 2-38 • Temperature Data Format, the temperature K column was changed for 85°C from 538 to 358.	2-98
	In Table 2-45 • ADC Interface Timing, "Typical-Case" was changed to "Worst-Case."	2-110
	The "ADC Interface Timing" section is new.	2-110
	Table 2-46 • Analog Channel Specifications was updated.	2-118
	The "V_{CC15A} Analog Power Supply (1.5 V)" section was updated.	2-224
	The "$V_{CCPLA/B}$ PLL Supply Voltage" section is new.	2-225
	In "V_{CCNVM} Flash Memory Block Power Supply (1.5 V)" section, supply was changed to supply input.	2-224
	The "$V_{CCPLA/B}$ PLL Supply Voltage" pin description was updated to include the following statement: Actel recommends tying V_{CCPLX} to V_{CC} and using proper filtering circuits to decouple V_{CC} noise from PLL.	2-225
	The "$V_{COMPLA/B}$ Ground for West and East PLL" section was updated.	2-225
	In Table 2-47 • ADC Characteristics in Direct Input Mode, the commercial conditions were updated and note 2 is new.	2-121
	The $V_{CC33ACAP}$ signal name was changed to "XTAL1 Crystal Oscillator Circuit Input".	2-228
	Table 2-48 • Uncalibrated Analog Channel Accuracy*is new.	2-123
	Table 2-49 • Calibrated Analog Channel Accuracy [1,2,3], is new.	2-124

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.9 (continued)	Table 2-50 • Analog Channel Accuracy: Monitoring Standard Positive Voltages is new.	2-125
	In Table 2-57 • Voltage Polarity Control Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3)*, the following I/O Bank names were changed: Hot-Swap changed to Standard LVDS changed to Advanced	2-131
	In Table 2-58 • Prescaler Op Amp Power-Down Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3), the following I/O Bank names were changed: Hot-Swap changed to Standard LVDS changed to Advanced	2-132
	In the title of Table 2-64 • I/O Standards Supported by Bank Type, LVDS I/O was changed to Advanced I/O.	2-134
	The title was changed from "Fusion Standard, LVDS, and Standard plus Hot-Swap I/O" to Table 2-68 • Fusion Standard and Advanced I/O Features. In addition, the table headings were all updated. The heading used to be Standard and LVDS I/O and was changed to Advanced I/O. Standard Hot-Swap was changed to just Standard.	2-136
	This sentence was deleted from the "Slew Rate Control and Drive Strength" section: The Standard hot-swap I/Os do not support slew rate control. In addition, these references were changed: • From: Fusion hot-swap I/O (Table 2-69 on page 2-122) To: Fusion Standard I/O • From: Fusion LVDS I/O (Table 2-70 on page 2-122) To: Fusion Advanced I/O	2-152
	The "Cold-Sparing Support" section was significantly updated.	2-143
	In the title of Table 2-75 • Fusion Standard I/O Standards—OUT_DRIVE Settings, Hot-Swap was changed to Standard.	2-153
	In the title of Table 2-76 • Fusion Advanced I/O Standards—SLEW and OUT_DRIVE Settings, LVDS was changed to Advanced.	2-153
	In the title of Table 2-81 • Fusion Standard and Advanced I/O Attributes vs. I/O Standard Applications, LVDS was changed to Advanced.	2-157
	In Figure 2-111 • Naming Conventions of Fusion Devices with Three Digital I/O Banks and Figure 2-112 • Naming Conventions of Fusion Devices with Four I/O Banks the following names were changed: Hot-Swap changed to Standard LVDS changed to Advanced	2-160
	The Figure 2-113 • Timing Model was updated.	2-161
	In the notes for Table 2-86 • Summary of Maximum and Minimum DC Input Levels Applicable to Commercial and Industrial Conditions, T_J was changed to T_A.	2-166
Advance v0.7 (January 2007)	Figure 2-16 • Fusion Clocking Options and the "RC Oscillator" section were updated to change GND_OSC and VCC_OSC to GNDOSC and VCCOSC.	2-20, 2-21
	Figure 2-19 • Fusion CCC Options: Global Buffers with the PLL Macro was updated to change the positions of OADIVRST and OADIVHALF, and a note was added.	2-25
	The "Crystal Oscillator" section was updated to include information about controlling and enabling/disabling the crystal oscillator.	2-22

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.7 (continued)	Table 2-11 · Electrical Characteristics of the Crystal Oscillator was updated to change the typical value of $I_{DYNXTAL}$ for 0.032–0.2 MHz to 0.19.	2-24
	The "1.5 V Voltage Regulator" section was updated to add "or floating" in the paragraph stating that an external pull-down is required on TRST to power down the VR.	2-41
	The "1.5 V Voltage Regulator" section was updated to include information on powering down with the VR.	2-41
	This sentence was updated in the "No-Glitch MUX (NGMUX)" section to delete GLA: The GLMUXCFG[1:0] configuration bits determine the source of the CLK inputs (i.e., internal signal or GLC).	2-32
	In Table 2-13 • NGMUX Configuration and Selection Table, 10 and 11 were deleted.	2-32
	The method to enable sleep mode was updated for bit 0 in Table 2-16 • RTC Control/Status Register.	2-38
	S2 was changed to D2 in Figure 2-39 • Read Waveform (Pipe Mode, 32-bit access) for RD[31:0] was updated.	2-51
	The definitions for bits 2 and 3 were updated in Table 2-24 • Page Status Bit Definition.	2-52
	Figure 2-46 • FlashROM Timing Diagram was updated.	2-58
	Table 2-26 • FlashROM Access Time is new.	2-58
	Figure 2-55 • Write Access After Write onto Same Address, Figure 2-56 • Read Access After Write onto Same Address, and Figure 2-57 • Write Access After Read onto Same Address are new.	2-68–2-70
	Table 2-31 • RAM4K9 and Table 2-32 • RAM512X18 were updated.	2-71, 2-72
	The VAREF and SAMPLE functions were updated in Table 2-36 • Analog Block Pin Description.	2-82
	The title of Figure 2-72 • Timing Diagram for Current Monitor Strobe was updated to add the word "positive."	2-91
	The "Gate Driver" section was updated to give information about the switching rate in High Current Drive mode.	2-94
	The "ADC Description" section was updated to include information about the SAMPLE and BUSY signals and the maximum frequencies for SYSCLK and ADCCLK. EQ 2-12 was updated to add parentheses around the entire expression in the denominator.	2-102
	Table 2-46 · Analog Channel Specifications and Table 2-47 · ADC Characteristics in Direct Input Mode were updated.	2-118, 2-121
	The note was removed from Table 2-55 • Analog Multiplexer Truth Table—AV (x = 0), AC (x = 1), and AT (x = 3).	2-131
	Table 2-63 • Internal Temperature Monitor Control Truth Table is new.	2-132
	The "Cold-Sparing Support" section was updated to add information about cases where current draw can occur.	2-143
	Figure 2-104 • Solution 4 was updated.	2-147
	Table 2-75 • Fusion Standard I/O Standards—OUT_DRIVE Settings was updated.	2-153

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.7 (continued)	The "GNDA Ground (analog)" section and "GNDAQ Ground (analog quiet)" section were updated to add information about maximum differential voltage.	2-224
	The "V_{AREF} Analog Reference Voltage" section and "VPUMP Programming Supply Voltage" section were updated.	2-226
	The "$V_{CCPLA/B}$ PLL Supply Voltage" section was updated to include information about the east and west PLLs.	2-225
	The V_{COMPLF} pin description was deleted.	N/A
	The "Axy Analog Input/Output" section was updated with information about grounding and floating the pin.	2-226
	The voltage range in the "VPUMP Programming Supply Voltage" section was updated. The parenthetical reference to "pulled up" was removed from the statement, "V_{PUMP} can be left floating or can be tied (pulled up) to any voltage between 0 V and 3.6 V."	2-225
	The "ATRTNx Temperature Monitor Return" section was updated with information about grounding and floating the pin.	2-226
	The following text was deleted from the "V_{REF} I/O Voltage Reference" section: (all digital I/O).	2-225
	The "NCAP Negative Capacitor" section and "PCAP Positive Capacitor" section were updated to include information about the type of capacitor that is required to connect the two.	2-228
	1 µF was changed to 100 pF in the "XTAL1 Crystal Oscillator Circuit Input".	2-228
	The "Programming" section was updated to include information about V_{CCOSC}.	2-229
Advance v0.5 (June 2006)	The second paragraph of the "PLL Macro" section was updated to include information about POWERDOWN.	2-30
	The description for bit 0 was updated in Table 2-17 · RTC Control/Status Register.	2-38
	3.9 was changed to 7.8 in the "Crystal Oscillator (Xtal Osc)" section.	2-40.
	All function descriptions in Table 2-18 · Signals for VRPSM Macro.	2-42
	In Table 2-19 • Flash Memory Block Pin Names, the RD[31:0] description was updated.	2-43
	The "RESET" section was updated.	2-61
	The "RESET" section was updated.	2-64
	Table 2-35 • FIFO was updated.	2-79
	The VAREF function description was updated in Table 2-36 • Analog Block Pin Description.	2-82
	The "Voltage Monitor" section was updated to include information about low power mode and sleep mode.	2-86
	The text in the "Current Monitor" section was changed from 2 mV to 1 mV.	2-90
	The "Gate Driver" section was updated to include information about forcing 1 V on the drain.	2-94
	The "Analog-to-Digital Converter Block" section was updated with the following statement: "All results are MSB justified in the ADC."	2-99
	The information about the ADCSTART signal was updated in the "ADC Description" section.	2-102

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.5 (continued)	Table 2-46 · Analog Channel Specifications was updated.	2-118
	Table 2-47 · ADC Characteristics in Direct Input Mode was updated.	2-121
	Table 2-51 • ACM Address Decode Table for Analog Quad was updated.	2-127
	In Table 2-53 • Analog Quad ACM Byte Assignment, the Function and Default Setting for Bit 6 in Byte 3 was updated.	2-130
	The "Introduction" section was updated to include information about digital inputs, outputs, and bibufs.	2-133
	In Table 2-69 • Fusion Pro I/O Features, the programmable delay descriptions were updated for the following features: Single-ended receiver Voltage-referenced differential receiver LVDS/LVPECL differential receiver features	2-137
	The "User I/O Naming Convention" section was updated to include "V" and "z" descriptions	2-159
	The "$V_{CC33PMP}$ Analog Power Supply (3.3 V)" section was updated to include information about avoiding high current draw.	2-224
	The "V_{CCNVM} Flash Memory Block Power Supply (1.5 V)" section was updated to include information about avoiding high current draw.	2-224
	The "VMVx I/O Supply Voltage (quiet)" section was updated to include this statement: VMV and V_{CCI} must be connected to the same power supply and V_{CCI} pins within a given I/O bank.	2-185
	The "PUB Push Button" section was updated to include information about leaving the pin floating if it is not used.	2-228
	The "PTBASE Pass Transistor Base" section was updated to include information about leaving the pin floating if it is not used.	2-228
	The "PTEM Pass Transistor Emitter" section was updated to include information about leaving the pin floating if it is not used.	2-228
Advance v0.4 (April 2006)	The "Voltage Regulator Power Supply Monitor (VRPSM)" section was updated.	2-42
Advance v0.2 (April 2006)	Figure 2-46 • FlashROM Timing Diagram was updated.	2-58
	The "FlashROM" section was updated.	2-57
	The "RESET" section was updated.	2-61
	The "RESET" section was updated.	2-64
	Figure 2-27 · Real-Time Counter System was updated.	2-35
	Table 2-19 • Flash Memory Block Pin Names was updated.	2-43
	Figure 2-33 • Flash Memory Block Diagram was updated to include AUX block information.	2-45
	Figure 2-34 • Flash Memory Block Organization was updated to include AUX block information.	2-46
	The note in the "Program Operation" section was updated.	2-48
	Figure 2-76 • Gate Driver Example was updated.	2-95
	The "Analog Quad ACM Description" section was updated.	2-130

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.2 (continued)	Information about the maximum pad input frequency was added to the "Gate Driver" section.	2-94
	Figure 2-65 • Analog Block Macro was updated.	2-81
	Figure 2-65 • Analog Block Macro was updated.	2-81
	The "Analog Quad" section was updated.	2-84
	The "Voltage Monitor" section was updated.	2-86
	The "Direct Digital Input" section was updated.	2-89
	The "Current Monitor" section was updated.	2-90
	Information about the maximum pad input frequency was added to the "Gate Driver" section.	2-94
	The "Temperature Monitor" section was updated.	2-96
	EQ 2-12 is new.	2-103
	The "ADC Description" section was updated.	2-102
	Figure 2-16 • Fusion Clocking Options was updated.	2-20
	Table 2-46 · Analog Channel Specifications was updated.	2-118
	The notes in Table 2-72 • Fusion Standard and Advanced I/O – Hot-Swap and 5 V Input Tolerance Capabilities were updated.	2-144
	The "Simultaneously Switching Outputs and PCB Layout" section is new.	2-149
	LVPECL and LVDS were updated in Table 2-81 • Fusion Standard and Advanced I/O Attributes vs. I/O Standard Applications.	2-157
	LVPECL and LVDS were updated in Table 2-82 • Fusion Pro I/O Attributes vs. I/O Standard Applications.	2-158
	The "Timing Model" was updated.	2-161
	All voltage-referenced Minimum and Maximum DC Input and Output Level tables were updated.	N/A
	All Timing Characteristic tables were updated	N/A
	Table 2-83 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions was updated.	2-165
	Table 2-79 • Summary of I/O Timing Characteristics – Software Default Settings was updated.	2-134
	Table 2-93 • I/O Output Buffer Maximum Resistances [1] was updated.	2-171
	The "BLVDS/M-LVDS" section is new. BLVDS and M-LVDS are two new I/O standards included in the datasheet.	2-211
	The "CoreMP7 and Cortex-M1 Software Tools" section is new.	2-257
	Table 2-83 • Summary of Maximum and Minimum DC Input and Output Levels Applicable to Commercial and Industrial Conditions was updated.	2-165
	Table 2-79 • Summary of I/O Timing Characteristics – Software Default Settings was updated.	2-134
	Table 2-93 • I/O Output Buffer Maximum Resistances [1] was updated.	2-171
	The "BLVDS/M-LVDS" section is new. BLVDS and M-LVDS are two new I/O standards included in the datasheet.	2-211

Rate This Document

3 – DC and Power Characteristics

General Specifications

Operating Conditions

Stresses beyond those listed in Table 3-1 may cause permanent damage to the device.

Exposure to absolute maximum rated conditions for extended periods may affect device reliability. Devices should not be operated outside the recommended operating ranges specified in Table 3-2 on page 3-3.

Table 3-1 • **Absolute Maximum Ratings**

Symbol	Parameter	Commercial	Industrial	Units
V_{CC}	DC core supply voltage	–0.3 to 1.65	–0.3 to 1.65	V
V_{JTAG}	JTAG DC voltage	–0.3 to 3.75	–0.3 to 3.75	V
V_{PUMP}	Programming voltage	–0.3 to 3.75	–0.3 to 3.75	V
V_{CCPLL}	Analog power supply (PLL)	–0.3 to 1.65	–0.3 to 1.65	V
V_{CCI}	DC I/O output buffer supply voltage	–0.3 to 3.75	–0.3 to 3.75	V
VI	I/O input voltage [1]	–0.3 V to 3.6 V (when I/O hot insertion mode is enabled) –0.3 V to (V_{CCI} + 1 V) or 3.6 V, whichever voltage is lower (when I/O hot-insertion mode is disabled)		V
V_{CC33A}	+3.3 V power supply	–0.3 to 3.75 [2]	–0.3 to 3.75 [2]	V
$V_{CC33PMP}$	+3.3 V power supply	–0.3 to 3.75 [2]	–0.3 to 3.75 [2]	V
VAREF	Voltage reference for ADC	–0.3 to 3.75	–0.3 to 3.75	V
V_{CC15A}	Digital power supply for the analog system	–0.3 to 1.65	–0.3 to 1.65	V
V_{CCNVM}	Embedded flash power supply	–0.3 to 1.65	–0.3 to 1.65	V
V_{CCOSC}	Oscillator power supply	–0.3 to 3.75	–0.3 to 3.75	V

Notes:

1. *The device should be operated within the limits specified by the datasheet. During transitions, the input signal may undershoot or overshoot according to the limits shown in Table 3-4 on page 3-4.*
2. *Analog data not valid beyond 3.65 V.*
3. *The high current mode has a maximum power limit of 20 mW. Appropriate current limit resistors must be used, based on voltage on the pad.*
4. *For flash programming and retention maximum limits, refer to Table 3-5 on page 3-5. For recommended operating limits refer to Table 3-2 on page 3-3.*

Table 3-1 • **Absolute Maximum Ratings (continued)**

Symbol	Parameter	Commercial	Industrial	Units
AV, AC	Unpowered, ADC reset asserted or unconfigured	–11.0 to 12.6	–11.0 to 12.0	V
	Analog input (+16 V to +2 V prescaler range)	–0.4 to 12.6	–0.4 to 12.0	V
	Analog input (+1 V to +0.125 V prescaler range)	–0.4 to 3.75	–0.4 to 3.75	V
	Analog input (–16 V to –2 V prescaler range)	–11.0 to 0.4	–11.0 to 0.4	V
	Analog input (–1 V to –0.125 V prescaler range)	–3.75 to 0.4	–3.75 to 0.4	V
	Analog input (direct input to ADC)	–0.4 to 3.75	–0.4 to 3.75	V
	Digital input	–0.4 to 12.6	–0.4 to 12.0	V
AG	Unpowered, ADC reset asserted or unconfigured	–11.0 to 12.6	–11.0 to 12.0	V
	Low Current Mode (1 µA, 3 µA, 10 µA, 30 µA)	–0.4 to 12.6	–0.4 to 12.0	V
	Low Current Mode (–1 µA, –3 µA, –10 µA, –30 µA)	–11.0 to 0.4	–11.0 to 0.4	V
	High Current Mode [3]	–11.0 to 12.6	–11.0 to 12.0	V
AT	Unpowered, ADC reset asserted or unconfigured	–0.4 to 16.0	–0.4 to 15.0	V
	Analog input (+16 V, 4 V prescaler range)	–0.4 to 16.0	–0.4 to 15.0	V
	Analog input (direct input to ADC)	–0.4 to 3.75	–0.4 to 3.75	V
	Digital input	–0.4 to 16.0	–0.4 to 15.0	V
T_{STG} [4]	Storage temperature	–65 to +150		°C
T_J [4]	Junction temperature	+125		°C

Notes:

1. *The device should be operated within the limits specified by the datasheet. During transitions, the input signal may undershoot or overshoot according to the limits shown in Table 3-4 on page 3-4.*
2. *Analog data not valid beyond 3.65 V.*
3. *The high current mode has a maximum power limit of 20 mW. Appropriate current limit resistors must be used, based on voltage on the pad.*
4. *For flash programming and retention maximum limits, refer to Table 3-5 on page 3-5. For recommended operating limits refer to Table 3-2 on page 3-3.*

Table 3-2 • Recommended Operating Conditions

Symbol	Parameter		Commercial	Industrial	Units
T_J	Junction temperature		0 to +85	–40 to +100	°C
V_{CC}	1.5 V DC core supply voltage		1.425 to 1.575	1.425 to 1.575	V
V_{JTAG}	JTAG DC voltage		1.4 to 3.6	1.4 to 3.6	V
V_{PUMP}	Programming voltage	Programming mode	3.15 to 3.45	3.15 to 3.45	V
		Operation[3]	0 to 3.6	0 to 3.6	V
V_{CCPLL}	Analog power supply (PLL)		1.425 to 1.575	1.425 to 1.575	V
V_{CCI}	1.5 V DC supply voltage		1.425 to 1.575	1.425 to 1.575	V
	1.8 V DC supply voltage		1.7 to 1.9	1.7 to 1.9	V
	2.5 V DC supply voltage		2.3 to 2.7	2.3 to 2.7	V
	3.3 V DC supply voltage		3.0 to 3.6	3.0 to 3.6	V
	LVDS differential I/O		2.375 to 2.625	2.375 to 2.625	V
	LVPECL differential I/O		3.0 to 3.6	3.0 to 3.6	V
V_{CC33A}	+3.3 V power supply		2.97 to 3.63	2.97 to 3.63	V
$V_{CC33PMP}$	+3.3 V power supply		2.97 to 3.63	2.97 to 3.63	V
VAREF	Voltage reference for ADC		2.527 to 2.593	2.527 to 2.593	V
V_{CC15A} [6]	Digital power supply for the analog system		1.425 to 1.575	1.425 to 1.575	V
V_{CCNVM}	Embedded flash power supply		1.425 to 1.575	1.425 to 1.575	V
V_{CCOSC}	Oscillator power supply		2.97 to 3.63	2.97 to 3.63	V
AV, AC [4]	Unpowered, ADC reset asserted or unconfigured		–10.5 to 12.0	–10.5 to 11.6	V
	Analog input (+16 V to +2 V prescaler range)		–0.3 to 12.0	–0.3 to 11.6	V
	Analog input (+1 V to + 0.125 V prescaler range)		–0.3 to 3.6	–0.3 to 3.6	V
	Analog input (–16 V to –2 V prescaler range)		–10.5 to 0.3	–10.5 to 0.3	V
	Analog input (–1 V to –0.125 V prescaler range)		–3.6 to 0.3	–3.6 to 0.3	V
	Analog input (direct input to ADC)		–0.3 to 3.6	–0.3 to 3.6	V
	Digital input		–0.3 to 12.0	–0.3 to 11.6	V
AG [4]	Unpowered, ADC reset asserted or unconfigured		–10.5 to 12.0	–10.5 to 11.6	V
	Low Current Mode (1 µA, 3 µA, 10 µA, 30 µA)		–0.3 to 12.0	–0.3 to 11.6	V
	Low Current Mode (–1 µA, –3 µA, –10 µA, –30 µA)		–10.5 to 0.3	–10.5 to 0.3	V
	High Current Mode [5]		–10.5 to 12.0	–10.5 to 11.6	V
AT [4]	Unpowered, ADC reset asserted or unconfigured		–0.3 to 15.5	–0.3 to 14.5	V
	Analog input (+16 V, +4 V prescaler range)		–0.3 to 15.5	–0.3 to 14.5	V
	Analog input (direct input to ADC)		–0.3 to 3.6	–0.3 to 3.6	V
	Digital input		–0.3 to 15.5	–0.3 to 14.5	V

Notes:

1. *The ranges given here are for power supplies only. The recommended input voltage ranges specific to each I/O standard are given in Table 2-82 on page 2-158.*
2. *All parameters representing voltages are measured with respect to GND unless otherwise specified.*
3. *V_{PUMP} can be left floating during normal operation (not programming mode).*
4. *The input voltage may overshoot by up to 500 mV above the Recommended Maximum (150 mV in Direct mode), provided the duration of the overshoot is less than 50% of the operating lifetime of the device.*
5. *The AG pad should also conform to the limits as specified in Table 2-45 on page 2-110.*
6. *Violating the V_{CC15A} recommended voltage supply during an embedded flash program cycle can corrupt the page being programmed.*

Table 3-3 • **Input Resistance of Analog Pads**

Pads	Pad Configuration	Prescaler Range	Input Resistance to Ground
AV, AC	Analog Input (direct input to ADC)	+16 V to +2 V	1 MΩ (typical)
		+1 V to +0.125 V	> 10 MΩ
	Analog Input (positive prescaler)	+16 V to +2 V	1 MΩ (typical)
		+1 V to +0.125 V	> 10 MΩ
	Analog Input (negative prescaler)	–16 V to –2 V	1 MΩ (typical)
		–1 V to –0.125 V	> 10 MΩ
	Digital input	+16 V to +2 V	1 MΩ (typical)
	Current monitor	+16 V to +2 V	1 MΩ (typical)
		–16 V to –2 V	1 MΩ (typical)
AT	Analog Input (direct input to ADC)	+16 V, +4 V	1 MΩ (typical)
	Analog Input (positive prescaler)	+16 V, +4 V	1 MΩ (typical)
	Digital input	+16 V, +4 V	1 MΩ (typical)
	Temperature monitor	+16 V, +4 V	> 10 MΩ

Table 3-4 • **Overshoot and Undershoot Limits [1]**

V_{CCI}	Average V_{CCI}–GND Overshoot or Undershoot Duration as a Percentage of Clock Cycle[2]	Maximum Overshoot/ Undershoot[2]
2.7 V or less	10%	1.4 V
	5%	1.49 V
3.0 V	10%	1.1 V
	5%	1.19 V
3.3 V	10%	0.79 V
	5%	0.88 V
3.6 V	10%	0.45 V
	5%	0.54 V

Notes:

1. *Based on reliability requirements at a junction temperature of 85°C.*
2. *The duration is allowed at one cycle out of six clock cycle. If the overshoot/undershoot occurs at one out of two cycles, the maximum overshoot/undershoot has to be reduced by 0.15 V.*

Table 3-5 • **FPGA Programming, Storage, and Operating Limits**

Product Grade	Storage Temperature	Element	Grade Programming Cycles	Retention
Commercial	Min. T_J = 0°C Min. T_J = 85°C	FPGA/FlashROM	500	20 years
		Embedded Flash	< 1,000	20 years
			< 10,000	10 years
			< 15,000	5 years
Industrial	Min. T_J = –40°C Min. T_J = 100°C	FPGA/FlashROM	500	20 years
		Embedded Flash	< 1,000	20 years
			< 10,000	10 years
			< 15,000	5 years

I/O Power-Up and Supply Voltage Thresholds for Power-On Reset (Commercial and Industrial)

Sophisticated power-up management circuitry is designed into every Fusion device. These circuits ensure easy transition from the powered off state to the powered up state of the device. The many different supplies can power up in any sequence with minimized current spikes or surges. In addition, the I/O will be in a known state through the power-up sequence. The basic principle is shown in Figure 3-1 on page 3-6.

There are five regions to consider during power-up.

Fusion I/Os are activated only if ALL of the following three conditions are met:

1. V_{CC} and V_{CCI} are above the minimum specified trip points (Figure 3-1).
2. $V_{CCI} > V_{CC} – 0.75$ V (typical).
3. Chip is in the operating mode.

V_{CCI} Trip Point:

Ramping up: 0.6 V < trip_point_up < 1.2 V

Ramping down: 0.5 V < trip_point_down < 1.1 V

V_{CC} Trip Point:

Ramping up: 0.6 V < trip_point_up < 1.1 V

Ramping down: 0.5 V < trip_point_down < 1 V

V_{CC} and V_{CCI} ramp-up trip points are about 100 mV higher than ramp-down trip points. This specifically built-in hysteresis prevents undesirable power-up oscillations and current surges. Note the following:

- During programming, I/Os become tristated and weakly pulled up to V_{CCI}.
- JTAG supply, PLL power supplies, and charge pump V_{PUMP} supply have no influence on I/O behavior.

Internal Power-Up Activation Sequence

1. Core
2. Input buffers
3. Output buffers, after 200 ns delay from input buffer activation

PLL Behavior at Brownout Condition

Actel recommends using monotonic power supplies or voltage regulators to ensure proper power-up behavior. Power ramp-up should be monotonic at least until V_{CC} and V_{CCPLX} exceed brownout activation levels. The V_{CC} activation level is specified as 1.1 V worst-case (see Figure 3-1 on page 3-6 for more details).

When PLL power supply voltage and/or V_{CC} levels drop below the VCC brownout levels (0.75 V ± 0.25 V), the PLL output lock signal goes low and/or the output clock is lost.

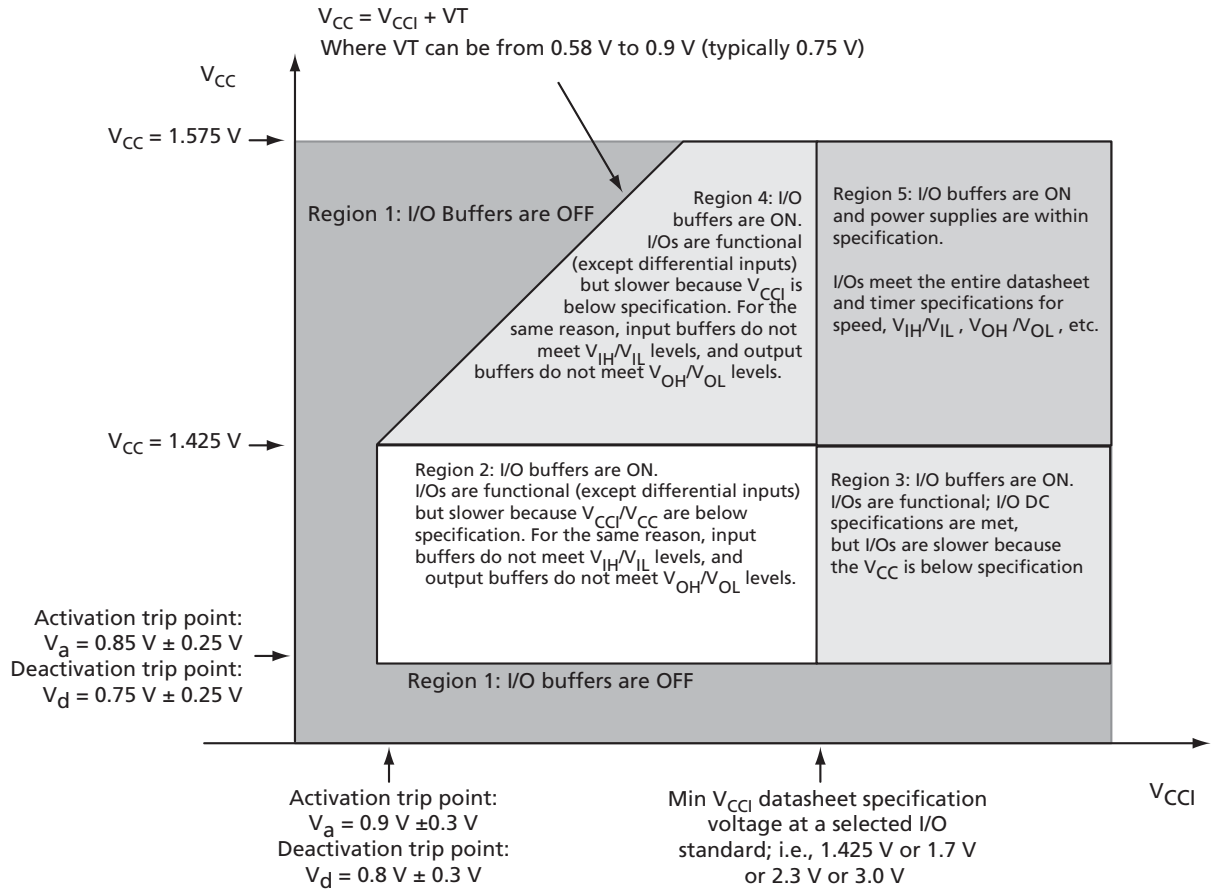


Figure 3-1 • **I/O State as a Function of V_{CCI} and V_{CC} Voltage Levels**

Thermal Characteristics

Introduction

The temperature variable in the Actel Designer software refers to the junction temperature, not the ambient, case, or board temperatures. This is an important distinction because dynamic and static power consumption will cause the chip's junction temperature to be higher than the ambient, case, or board temperatures. EQ 3-1 through EQ 3-3 give the relationship between thermal resistance, temperature gradient, and power.

$$\theta_{JA} = \frac{T_J - \theta_A}{P}$$

EQ 3-1

$$\theta_{JB} = \frac{T_J - T_B}{P}$$

EQ 3-2

$$\theta_{JC} = \frac{T_J - T_C}{P}$$

EQ 3-3

where

θ_{JA} = Junction-to-air thermal resistance
θ_{JB} = Junction-to-board thermal resistance
θ_{JC} = Junction-to-case thermal resistance
T_J = Junction temperature
T_A = Ambient temperature
T_B = Board temperature (measured 1.0 mm away from the package edge)
T_C = Case temperature
P = Total power dissipated by the device

Table 3-6 • **Package Thermal Resistance**

Product	Die Size (mm)	θ_{JA} Still Air	θ_{JA} 1.0 m/s	θ_{JA} 2.5 m/s	θ_{JC}	θ_{JB}	Units
AFS090-QN108	X = 3.4; Y = 4.8	34.5	30.0	27.7	8.1	16.7	°C/W
AFS090-QN180	X = 3.4; Y = 4.8	33.3	27.6	25.7	9.2	21.2	°C/W
AFS250-QN180	X = 4.0; Y = 5.6	32.2	26.5	24.7	5.7	15.0	°C/W
AFS090-FG256	X = 3.4; Y = 4.8	37.7	33.9	32.2	11.5	29.7	°C/W
AFS250-FG256	X = 4.0; Y = 5.6	33.7	30.0	28.3	9.3	24.8	°C/W
AFS600-FG256	X = 5.10; Y = 7.3	28.9	25.2	23.5	6.8	19.9	°C/W
AFS1500-FG256	X = 7.62; Y = 9.98	23.3	19.6	18.0	4.3	14.2	°C/W
AFS600-FG484	X = 5.10; Y = 7.3	21.8	18.2	16.7	7.7	16.8	°C/W
AFS1500-FG484	X = 7.62; Y = 9.98	21.6	16.8	15.2	5.6	14.9	°C/W

Theta-JA

Junction-to-ambient thermal resistance (θ_{JA}) is determined under standard conditions specified by JEDEC (JESD-51), but it has little relevance in actual performance of the product. It should be used with caution but is useful for comparing the thermal performance of one package to another.

A sample calculation showing the maximum power dissipation allowed for the AFS600-FG484 package under forced convection of 1.0 m/s and 75°C ambient temperature is as follows:

$$\text{Maximum Power Allowed} = \frac{T_{J(MAX)} - T_{A(MAX)}}{\theta_{JA}}$$

EQ 3-4

where

θ_{JA} = 19.00°C/W (taken from Table 3-6 on page 3-7).

T_A = 75.00°C

$$\text{Maximum Power Allowed} = \frac{100.00°C - 75.00°C}{19.00°C/W} = 1.3\ W$$

EQ 3-5

The power consumption of a device can be calculated using the Actel power calculator. The device's power consumption must be lower than the calculated maximum power dissipation by the package. If the power consumption is higher than the device's maximum allowable power dissipation, a heat sink can be attached on top of the case, or the airflow inside the system must be increased.

Theta-JB

Junction-to-board thermal resistance (θ_{JB}) measures the ability of the package to dissipate heat from the surface of the chip to the PCB. As defined by the JEDEC (JESD-51) standard, the thermal resistance from junction to board uses an isothermal ring cold plate zone concept. The ring cold plate is simply a means to generate an isothermal boundary condition at the perimeter. The cold plate is mounted on a JEDEC standard board with a minimum distance of 5.0 mm away from the package edge.

Theta-JC

Junction-to-case thermal resistance (θ_{JC}) measures the ability of a device to dissipate heat from the surface of the chip to the top or bottom surface of the package. It is applicable for packages used with external heat sinks. Constant temperature is applied to the surface in consideration and acts as a boundary condition. This only applies to situations where all or nearly all of the heat is dissipated through the surface in consideration.

Calculation for Heat Sink

For example, in a design implemented in an AFS600-FG484 package with 2.5 m/s airflow, the power consumption value using the power calculator is 3.00 W. The user-dependent T_a and T_j are given as follows:

T_J = 100.00°C

T_A = 70.00°C

From the datasheet:

θ_{JA} = 17.00°C/W

θ_{JC} = 8.28°C/W

$$P = \frac{T_J - T_A}{\theta_{JA}} = \frac{100°C - 70°C}{17.00\ W} = 1.76\ W$$

EQ 3-6

The 1.76 W power is less than the required 3.00 W. The design therefore requires a heat sink, or the airflow where the device is mounted should be increased. The design's total junction-to-air thermal resistance requirement can be estimated by EQ 3-7:

$$\theta_{ja(total)} = \frac{T_J - T_A}{P} = \frac{100°C - 70°C}{3.00\ W} = 10.00°C/W$$

EQ 3-7

Determining the heat sink's thermal performance proceeds as follows:

$$\theta_{JA(TOTAL)} = \theta_{JC} + \theta_{CS} + \theta_{SA}$$

EQ 3-8

where

θ_{JA} = 0.37°C/W

= Thermal resistance of the interface material between the case and the heat sink, usually provided by the thermal interface manufacturer

θ_{SA} = Thermal resistance of the heat sink in °C/W

$$\theta_{SA} = \theta_{JA(TOTAL)} - \theta_{JC} - \theta_{CS}$$

EQ 3-9

$$\theta_{SA} = 13.33°C/W - 8.28°C/W - 0.37°C/W = 5.01°C/W$$

A heat sink with a thermal resistance of 5.01°C/W or better should be used. Thermal resistance of heat sinks is a function of airflow. The heat sink performance can be significantly improved with increased airflow.

Carefully estimating thermal resistance is important in the long-term reliability of an Actel FPGA. Design engineers should always correlate the power consumption of the device with the maximum allowable power dissipation of the package selected for that device.

Note: The junction-to-air and junction-to-board thermal resistances are based on JEDEC standard (JESD-51) and assumptions made in building the model. It may not be realized in actual application and therefore should be used with a degree of caution. Junction-to-case thermal resistance assumes that all power is dissipated through the case.

Temperature and Voltage Derating Factors

Table 3-7 • **Temperature and Voltage Derating Factors for Timing Delays (normalized to T_J = 70°C, Worst-Case V_{CC} = 1.425 V)**

Array Voltage V_{CC} (V)	Junction Temperature (°C)					
	–40°C	0°C	25°C	70°C	85°C	100°C
1.425	0.88	0.93	0.95	1.00	1.02	1.05
1.500	0.83	0.88	0.90	0.95	0.96	0.99
1.575	0.80	0.85	0.87	0.91	0.93	0.96

Calculating Power Dissipation

Quiescent Supply Current

Table 3-8 • **AFS1500 Quiescent Supply Current Characteristics**

Parameter	Description	Conditions	Temp.	Min.	Typ.	Max.	Unit
I_{CC}[1]	1.5 V quiescent current	Operational standby[4], V_{CC} = 1.575 V	T_J = 25°C		20	40	mA
			T_J = 85°C		32	65	mA
			T_J = 100°C		59	120	mA
		Standby mode[5] or Sleep mode[6], V_{CC} = 0 V			0	0	µA
I_{CC33}[2]	3.3 V analog supplies current	Operational standby[4], V_{CC33} = 3.63 V	T_J = 25°C		9.8	13	mA
			T_J = 85°C		10.7	14	mA
			T_J = 100°C		10.8	15	mA
		Operational standby, only Analog Quad and –3.3 V output ON, V_{CC33} = 3.63 V	T_J = 25°C		0.31	2	mA
			T_J = 85°C		0.35	2	mA
			T_J = 100°C		0.45	2	mA
		Standby mode[5], V_{CC33} = 3.63 V	T_J = 25°C		2.9	3.6	mA
			T_J = 85°C		2.9	4	mA
			T_J = 100°C		3.3	6	mA
		Sleep mode[6], V_{CC33} = 3.63 V	T_J = 25°C		17	19	µA
			T_J = 85°C		18	20	µA
			T_J = 100°C		24	25	µA
I_{CCI}[3]	I/O quiescent current	Operational standby[4], Standby mode, and Sleep Mode[6], V_{CCI}x = 3.63 V	T_J = 25°C		417	649	µA
			T_J = 85°C		417	649	µA
			T_J = 100°C		417	649	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC} and I_{CC15A}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, I_{CCI2}, and I_{CCI4}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*

Table 3-8 • AFS1500 Quiescent Supply Current Characteristics (continued)

Parameter	Description	Conditions	Temp.	Min.	Typ.	Max.	Unit
I_{JTAG}	JTAG I/O quiescent current	Operational standby[4], V_{JTAG} = 3.63 V	T_J = 25°C		80	100	µA
			T_J = 85°C		80	100	µA
			T_J = 100°C		80	100	µA
		Standby mode[5] or Sleep mode[6], V_{JTAG} = 0 V			0	0	µA
I_{PP}	Programming supply current	Non-programming mode, V_{PP} = 3.63 V	T_J = 25°C		39	80	µA
			T_J = 85°C		40	80	µA
			T_J = 100°C		40	80	µA
		Standby mode[5] or Sleep mode[6], V_{PP} = 0 V			0	0	µA
I_{CCNVM}	Embedded NVM current	Reset asserted, V_{CCNVM} = 1.575 V	T_J = 25°C		50	150	µA
			T_J =85°C		50	150	µA
			T_J = 100°C		50	150	µA
I_{CCPLL}	1.5 V PLL quiescent current	Operational standby , V_{CCPLL} = 1.575 V	T_J = 25°C		130	200	µA
			T_J = 85°C		130	200	µA
			T_J = 100°C		130	200	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC} and I_{CC15A}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, I_{CCI2}, and I_{CCI4}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*

Table 3-9 • **AFS600 Quiescent Supply Current Characteristics**

Parameter	Description	Conditions	Temp.	Min	Typ	Max	Unit
I_{CC}[1]	1.5 V quiescent current	Operational standby[4], V_{CC} = 1.575 V	T_J = 25°C		13	25	mA
			T_J = 85°C		20	45	mA
			T_J=100°C		25	75	mA
		Standby mode[5] or Sleep mode[6], V_{CC} = 0 V			0	0	µA
I_{CC33}[2]	3.3 V analog supplies current	Operational standby[4], V_{CC33} = 3.63 V	T_J = 25°C		9.8	13	mA
			T_J = 85°C		10.7	14	mA
			T_J = 100°C		10.8	15	mA
		Operational standby, only Analog Quad and –3.3 V output ON, V_{CC33} = 3.63 V	T_J = 25°C		0.31	2	mA
			T_J = 85°C		0.35	2	mA
			T_J = 100°C		0.45	2	mA
		Standby mode[5], V_{CC33} = 3.63 V	T_J = 25°C		2.8	3.6	mA
			T_J = 85°C		2.9	4	mA
			T_J = 100°C		3.5	6	mA
		Sleep mode[6], V_{CC33} = 3.63 V	T_J = 25°C		17	19	µA
			T_J = 85°C		18	20	µA
			T_J = 100°C		24	25	µA
I_{CCI}[3]	I/O quiescent current	Operational standby[4], V_{CCI}x = 3.63 V	T_J = 25°C		417	648	µA
			T_J = 85°C		417	648	µA
			T_J = 100°C		417	649	µA
I_{JTAG}	JTAG I/O quiescent current	Operational standby[4], V_{JTAG} = 3.63 V	T_J = 25°C		80	100	µA
			T_J = 85°C		80	100	µA
			T_J = 100°C		80	100	µA
		Standby mode[5] or Sleep mode[6], V_{JTAG} = 0 V			0	0	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC} and I_{CC15A}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, I_{CCI2}, and I_{CCI4}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*

Table 3-9 • **AFS600 Quiescent Supply Current Characteristics (continued)**

Parameter	Description	Conditions	Temp.	Min	Typ	Max	Unit
I_{PP}	Programming supply current	Non-programming mode, V_{PP} = 3.63 V	T_J = 25°C		36	80	µA
			T_J = 85°C		36	80	µA
			T_J = 100°C		36	80	µA
		Standby mode[5] or Sleep mode[6], V_{PP} = 0 V			0	0	µA
I_{CCNVM}	Embedded NVM current	Reset asserted, V_{CCNVM} = 1.575 V	T_J = 25°C		22	80	µA
			T_J = 85°C		24	80	µA
			T_J = 100°C		25	80	µA
I_{CCPLL}	1.5 V PLL quiescent current	Operational standby, V_{CCPLL} = 1.575 V	T_J = 25°C		130	200	µA
			T_J = 85°C		130	200	µA
			T_J = 100°C		130	200	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC} and I_{CC15A}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, I_{CCI2}, and I_{CCI4}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*

Table 3-10 • **AFS250 Quiescent Supply Current Characteristics**

Parameter	Description	Conditions	Temp.	Min	Typ	Max	Unit
I_{CC}[1]	1.5 V quiescent current	Operational standby[4], V_{CC} = 1.575 V	T_J = 25°C		4.8	10	mA
			T_J = 85°C		8.2	30	mA
			T_J = 100°C		15	50	mA
		Standby mode[5] or Sleep mode[6], V_{CC} = 0 V			0	0	µA
I_{CC33}[2]	3.3 V analog supplies current	Operational standby[4], V_{CC33} = 3.63 V	T_J = 25°C		9.8	13	mA
			T_J = 85°C		9.8	14	mA
			T_J = 100°C		10.8	15	mA
		Operational standby, only Analog Quad and –3.3 V output ON, V_{CC33} = 3.63 V	T_J = 25°C		0.29	2	mA
			T_J = 85°C		0.31	2	mA
			T_J = 100°C		0.45	2	mA
		Standby mode[5], V_{CC33} = 3.63V	T_J = 25°C		2.9	3.0	mA
			T_J = 85°C		2.9	3.1	mA
			T_J = 100°C		3.5	6	mA
		Sleep mode[6], V_{CC33} = 3.63 V	T_J = 25°C		19	18	µA
			T_J = 85°C		19	20	µA
			T_J = 100°C		24	25	µA
I_{CCI}[3]	I/O quiescent current	Operational standby[6], V_{CCI}x = 3.63 V	T_J = 25°C		266	437	µA
			T_J = 85°C		266	437	µA
			T_J = 100°C		266	437	µA
I_{JTAG}	JTAG I/O quiescent current	Operational standby[4], V_{JTAG} = 3.63 V	T_J = 25°C		80	100	µA
			T_J = 85°C		80	100	µA
			T_J = 100°C		80	100	µA
		Standby mode[5] or Sleep mode[6], V_{JTAG} = 0 V			0	0	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC}, I_{CCPLL}, I_{CC15A}, I_{CCNVM}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, and I_{CCI2}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = $V_{JTA\ G}$ = V_{PP} = 0 V.*

Table 3-10 • AFS250 Quiescent Supply Current Characteristics (continued)

Parameter	Description	Conditions	Temp.	Min	Typ	Max	Unit
I_{PP}	Programming supply current	Non-programming mode, V_{PP} = 3.63 V	T_J = 25°C		37	80	µA
			T_J = 85°C		37	80	µA
			T_J = 100°C		80	100	µA
		Standby mode[5] or Sleep mode[6], V_{PP} = 0 V			0	0	µA
I_{CCNVM}	Embedded NVM current	Reset asserted, V_{CCNVM} = 1.575 V	T_J = 25°C		10	40	µA
			T_J = 85°C		14	40	µA
			T_J = 100°C		14	40	µA
I_{CCPLL}	1.5 V PLL quiescent current	Operational standby, V_{CCPLL} = 1.575 V	T_J = 25°C		65	100	µA
			T_J = 85°C		65	100	µA
			T_J = 100°C		65	100	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC}, I_{CCPLL}, I_{CC15A}, I_{CCNVM}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, and I_{CCI2}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = $V_{JTA\ G}$ = V_{PP} = 0 V.*

Table 3-11 • **AFS090 Quiescent Supply Current Characteristics**

Parameter	Description	Conditions	Temp.	Min	Typ	Max	Unit
I_{CC}[1]	1.5 V quiescent current	Operational standby[4], V_{CC} = 1.575 V	T_J = 25°C		5	7.5	mA
			T_J = 85°C		6.5	20	mA
			T_J = 100°C		14	48	mA
		Standby mode[5] or Sleep mode[6], V_{CC} = 0 V			0	0	µA
I_{CC33}[2]	3.3 V analog supplies current	Operational standby[4], V_{CC33} = 3.63 V	T_J = 25°C		9.8	12	mA
			T_J = 85°C		9.8	12	mA
			T_J = 100°C		10.7	15	mA
		Operational standby, only Analog Quad and –3.3 V output ON, V_{CC33} = 3.63 V	T_J = 25°C		0.30	2	mA
			T_J = 85°C		0.30	2	mA
			T_J = 100°C		0.45	2	mA
		Standby mode[5], V_{CC33} = 3.63 V	T_J = 25°C		2.9	2.9	mA
			T_J = 85°C		2.9	3.0	mA
			T_J = 100°C		3.5	6	mA
		Sleep mode[6], V_{CC33} = 3.63 V	T_J = 25°C		17	18	µA
			T_J = 85°C		18	20	µA
			T_J = 100°C		24	25	µA
I_{CCI}[3]	I/O quiescent current	Operational standby[6], V_{CCI}x = 3.63 V	T_J = 25°C		260	437	µA
			T_J = 85°C		260	437	µA
			T_J = 100°C		260	437	µA
I_{JTAG}	JTAG I/O quiescent current	Operational standby[4], V_{JTAG} = 3.63 V	T_J = 25°C		80	100	µA
			T_J = 85°C		80	100	µA
			T_J = 100°C		80	100	µA
		Standby mode[5] or Sleep mode[6], V_{JTAG} = 0 V			0	0	µA
I_{PP}	Programming supply current	Non-programming mode, V_{PP} = 3.63 V	T_J = 25°C		37	80	µA
			T_J = 85°C		37	80	µA
			T_J = 100°C		80	100	µA
		Standby mode[5] or Sleep mode[6], V_{PP} = 0 V			0	0	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC}, I_{CCPLL}, I_{CC15A}, I_{CCNVM}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, and I_{CCI2}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*
6. *Sleep Mode, V_{CC} = V_{JTAG} = V_{PP} = 0 V.*

Table 3-11 • AFS090 Quiescent Supply Current Characteristics (continued)

Parameter	Description	Conditions	Temp.	Min	Typ	Max	Unit
I_{CCNVM}	Embedded NVM current	Reset asserted, V_{CCNVM} = 1.575 V	T_J = 25°C		10	40	µA
			T_J = 85°C		14	40	µA
			T_J = 100°C		14	40	µA
I_{CCPLL}	1.5 V PLL quiescent current	Operational standby, V_{CCPLL} = 1.575 V	T_J = 25°C		65	100	µA
			T_J = 85°C		65	100	µA
			T_J = 100°C		65	100	µA

Notes:

1. *I_{CC} is the 1.5 V power supplies, I_{CC}, I_{CCPLL}, I_{CC15A}, I_{CCNVM}.*
2. *I_{CC33A} includes I_{CC33A}, $I_{CC33PMP}$, and I_{CCOSC}.*
3. *I_{CCI} includes all I_{CCI0}, I_{CCI1}, and I_{CCI2}.*
4. *Operational standby is when the Fusion device is powered up, all blocks are used, no I/O is toggling, Voltage Regulator is loaded with 200 mA, $V_{CC33PMP}$ is ON, XTAL is ON, and ADC is ON.*
5. *XTAL is configured as high gain, $V_{CC} = V_{JTAG} = V_{PP} = 0$ V.*
6. *Sleep Mode, $V_{CC} = V_{JTAG} = V_{PP} = 0$ V.*

Power per I/O Pin

Table 3-12 • **Summary of I/O Input Buffer Power (per pin)—Default I/O Software Settings**

	V_{CCI} (V)	Static Power P_{DC7} (mW)[1]	Dynamic Power P_{AC9} (µW/MHz)[2]
Applicable to Pro I/O Banks			
Single-Ended			
3.3 V LVTTL/LVCMOS	3.3	–	17.39
3.3 V LVTTL/LVCMOS – Schmitt trigger	3.3	–	25.51
2.5 V LVCMOS	2.5	–	5.76
2.5 V LVCMOS – Schmitt trigger	2.5	–	7.16
1.8 V LVCMOS	1.8	–	2.72
1.8 V LVCMOS – Schmitt trigger	1.8	–	2.80
1.5 V LVCMOS (JESD8-11)	1.5	–	2.08
1.5 V LVCMOS (JESD8-11) – Schmitt trigger	1.5	–	2.00
3.3 V PCI	3.3	–	18.82
3.3 V PCI – Schmitt trigger	3.3	–	20.12
3.3 V PCI-X	3.3	–	18.82
3.3 V PCI-X – Schmitt trigger	3.3	–	20.12
Voltage-Referenced			
3.3 V GTL	3.3	2.90	8.23
2.5 V GTL	2.5	2.13	4.78
3.3 V GTL+	3.3	2.81	4.14
2.5 V GTL+	2.5	2.57	3.71
HSTL (I)	1.5	0.17	2.03
HSTL (II)	1.5	0.17	2.03
SSTL2 (I)	2.5	1.38	4.48
SSTL2 (II)	2.5	1.38	4.48
SSTL3 (I)	3.3	3.21	9.26
SSTL3 (II)	3.3	3.21	9.26
Differential			
LVDS	2.5	2.26	1.50
LVPECL	3.3	5.71	2.17

Notes:

1. *P_{DC7} is the static power (where applicable) measured on V_{CCI}.*
2. *P_{AC9} is the total dynamic power measured on V_{CC} and V_{CCI}.*

Table 3-12 • **Summary of I/O Input Buffer Power (per pin)—Default I/O Software Settings (continued)**

	V_{CCI} (V)	Static Power P_{DC7} (mW)[1]	Dynamic Power P_{AC9} (µW/MHz)[2]
Applicable to Advanced I/O Banks			
Single-Ended			
3.3 V LVTTL/LVCMOS	3.3	–	16.69
2.5 V LVCMOS	2.5	–	5.12
1.8 V LVCMOS	1.8	–	2.13
1.5 V LVCMOS (JESD8-11)	1.5	–	1.45
3.3 V PCI	3.3	–	18.11
3.3 V PCI-X	3.3	–	18.11
Differential			
LVDS	2.5	2.26	1.20
LVPECL	3.3	5.72	1.87
Applicable to Standard I/O Banks			
3.3 V LVTTL/LVCMOS	3.3	–	16.79
2.5 V LVCMOS	2.5	–	5.19
1.8 V LVCMOS	1.8	–	2.18
1.5 V LVCMOS (JESD8-11)	1.5	–	1.52

Notes:

1. *P_{DC7} is the static power (where applicable) measured on V_{CCI}.*
2. *P_{AC9} is the total dynamic power measured on V_{CC} and V_{CCI}.*

Table 3-13 • **Summary of I/O Output Buffer Power (per pin)—Default I/O Software Settings[1]**

	C_{LOAD} (pF)	V_{CCI} (V)	Static Power P_{DC8} (mW)[2]	Dynamic Power P_{AC10} (µW/MHz)[3]
Applicable to Pro I/O Banks				
Single-Ended				
3.3 V LVTTL/LVCMOS	35	3.3	–	474.70
2.5 V LVCMOS	35	2.5	–	270.73
1.8 V LVCMOS	35	1.8	–	151.78
1.5 V LVCMOS (JESD8-11)	35	1.5	–	104.55
3.3 V PCI	10	3.3	–	204.61
3.3 V PCI-X	10	3.3	–	204.61
Voltage-Referenced				
3.3 V GTL	10	3.3	–	24.08
2.5 V GTL	10	2.5	–	13.52
3.3 V GTL+	10	3.3	–	24.10
2.5 V GTL+	10	2.5	–	13.54
HSTL (I)	20	1.5	7.08	26.22
HSTL (II)	20	1.5	13.88	27.22
SSTL2 (I)	30	2.5	16.69	105.56
SSTL2 (II)	30	2.5	25.91	116.60
SSTL3 (I)	30	3.3	26.02	114.87
SSTL3 (II)	30	3.3	42.21	131.76
Differential				
LVDS	–	2.5	7.70	89.62
LVPECL	–	3.3	19.42	168.02
Applicable to Advanced I/O Banks				
Single-Ended				
3.3 V LVTTL / 3.3 V LVCMOS	35	3.3	–	468.67
2.5 V LVCMOS	35	2.5	–	267.48
1.8 V LVCMOS	35	1.8	–	149.46
1.5 V LVCMOS (JESD8-11)	35	1.5	–	103.12
3.3 V PCI	10	3.3	–	201.02
3.3 V PCI-X	10	3.3	–	201.02

Notes:

1. *Dynamic power consumption is given for standard load and software-default drive strength and output slew.*
2. *P_{DC8} is the static power (where applicable) measured on V_{CCI}.*
3. *P_{AC10} is the total dynamic power measured on V_{CC} and V_{CCI}.*

Table 3-13 • **Summary of I/O Output Buffer Power (per pin)—Default I/O Software Settings[1] (continued)**

	C_{LOAD} (pF)	V_{CCI} (V)	Static Power P_{DC8} (mW)[2]	Dynamic Power P_{AC10} (µW/MHz)[3]
Differential				
LVDS	–	2.5	7.74	88.92
LVPECL	–	3.3	19.54	166.52
Applicable to Standard I/O Banks				
Single-Ended				
3.3 V LVTTL / 3.3 V LVCMOS	35	3.3	–	431.08
2.5 V LVCMOS	35	2.5	–	247.36
1.8 V LVCMOS	35	1.8	–	128.46
1.5 V LVCMOS (JESD8-11)	35	1.5	–	89.46

Notes:

1. *Dynamic power consumption is given for standard load and software-default drive strength and output slew.*
2. *P_{DC8} is the static power (where applicable) measured on V_{CCI}.*
3. *P_{AC10} is the total dynamic power measured on V_{CC} and V_{CCI}.*

Dynamic Power Consumption of Various Internal Resources

Table 3-14 • **Different Components Contributing to the Dynamic Power Consumption in Fusion Devices**

Parameter	Definition	Power Supply		Device-Specific Dynamic Contributions				Units
		Name	Setting	AFS1500	AFS600	AFS250	AFS090	
P_{AC1}	Clock contribution of a Global Rib	V_{CC}	1.5 V	14.5	12.8	11	11	µW/MHz
P_{AC2}	Clock contribution of a Global Spine	V_{CC}	1.5 V	2.5	1.9	1.6	0.8	µW/MHz
P_{AC3}	Clock contribution of a VersaTile row	V_{CC}	1.5 V	0.81				µW/MHz
P_{AC4}	Clock contribution of a VersaTile used as a sequential module	V_{CC}	1.5 V	0.11				µW/MHz
P_{AC5}	First contribution of a VersaTile used as a sequential module	V_{CC}	1.5 V	0.07				µW/MHz
P_{AC6}	Second contribution of a VersaTile used as a sequential module	V_{CC}	1.5 V	0.29				µW/MHz
P_{AC7}	Contribution of a VersaTile used as a combinatorial module	V_{CC}	1.5 V	0.29				µW/MHz
P_{AC8}	Average contribution of a routing net	V_{CC}	1.5 V	0.70				µW/MHz
P_{AC9}	Contribution of an I/O input pin (standard dependent)	V_{CCI}	See Table 3-12 on page 3-18					
P_{AC10}	Contribution of an I/O output pin (standard dependent)	V_{CCI}	See Table 3-13 on page 3-20					
P_{AC11}	Average contribution of a RAM block during a read operation	V_{CC}	1.5 V	25				µW/MHz
P_{AC12}	Average contribution of a RAM block during a write operation	V_{CC}	1.5 V	30				µW/MHz
P_{AC13}	Dynamic Contribution for PLL	V_{CC}	1.5 V	2.6				µW/MHz
P_{AC15}	Contribution of NVM block during a read operation (F < 33MHz)	V_{CC}	1.5 V	358				µW/MHz
P_{AC16}	1st contribution of NVM block during a read operation (F > 33MHz)	V_{CC}	1.5 V	12.88				mW

Table 3-14 • **Different Components Contributing to the Dynamic Power Consumption in Fusion Devices**

Parameter	Definition	Power Supply		Device-Specific Dynamic Contributions				Units
		Name	Setting	AFS1500	AFS600	AFS250	AFS090	
P_{AC17}	2nd contribution of NVM block during a read operation (F > 33MHz)	V_{CC}	1.5 V	4.8				µW/MHz
P_{AC18}	Crystal Oscillator contribution	V_{CC33A}	3.3 V	0.63				mW
P_{AC19}	RC Oscillator contribution	V_{CC33A}	3.3 V	3.3				mW
P_{AC20}	Analog Block dynamic power contribution of ADC	V_{CC}	1.5 V	3				mW

Static Power Consumption of Various Internal Resources

Table 3-15 • **Different Components Contributing to the Static Power Consumption in Fusion Devices**

Parameter	Definition	Power Supply		Device-Specific Static Contributions				Units
				AFS1500	AFS600	AFS250	AFS090	
P_{DC1}	Core static power contribution in operating mode	V_{CC}	1.5 V	18	7.5	4.50	3.00	mW
P_{DC2}	Device static power contribution in standby mode	V_{CC33A}	3.3 V	0.66				mW
P_{DC3}	Device static power contribution in sleep mode	V_{CC33A}	3.3 V	0.03				mW
P_{DC4}	NVM static power contribution	V_{CC}	1.5 V	1.19				mW
P_{DC5}	Analog Block static power contribution of ADC	V_{CC33A}	3.3 V	8.25				mW
P_{DC6}	Analog Block static power contribution per Quad	V_{CC33A}	3.3 V	3.3				mW
P_{DC7}	Static contribution per input pin – standard dependent contribution	V_{CCI}	See Table 3-12 on page 3-18					
P_{DC8}	Static contribution per input pin – standard dependent contribution	V_{CCI}	See Table 3-13 on page 3-20					
P_{DC9}	Static contribution for PLL	V_{CC}	1.5 V	2.55				mW

Power Calculation Methodology

This section describes a simplified method to estimate power consumption of an application. For more accurate and detailed power estimations, use the SmartPower tool in the Libero IDE software.

The power calculation methodology described below uses the following variables:

- The number of PLLs as well as the number and the frequency of each output clock generated
- The number of combinatorial and sequential cells used in the design
- The internal clock frequencies
- The number and the standard of I/O pins used in the design
- The number of RAM blocks used in the design
- The number of NVM blocks used in the design
- The number of Analog Quads used in the design
- Toggle rates of I/O pins as well as VersaTiles—guidelines are provided in Table 3-16 on page 3-29.
- Enable rates of output buffers—guidelines are provided for typical applications in Table 3-17 on page 3-29.
- Read rate and write rate to the RAM—guidelines are provided for typical applications in Table 3-17 on page 3-29.
- Read rate to the NVM blocks

The calculation should be repeated for each clock domain defined in the design.

Methodology

Total Power Consumption—P_{TOTAL}

Operating Mode, Standby Mode, and Sleep Mode

$P_{TOTAL} = P_{STAT} + P_{DYN}$

P_{STAT} is the total static power consumption.

P_{DYN} is the total dynamic power consumption.

Total Static Power Consumption—P_{STAT}

Operating Mode

$P_{STAT} = P_{DC1} + (N_{NVM\text{-}BLOCKS} * P_{DC4}) + P_{DC5} + (N_{QUADS} * P_{DC6}) + (N_{INPUTS} * P_{DC7}) + (N_{OUTPUTS} * P_{DC8}) + (N_{PLLS} * P_{DC9})$

$N_{NVM\text{-}BLOCKS}$ is the number of NVM blocks available in the device.

N_{QUADS} is the number of Analog Quads used in the design.

N_{INPUTS} is the number of I/O input buffers used in the design.

$N_{OUTPUTS}$ is the number of I/O output buffers used in the design.

N_{PLLS} is the number of PLLs available in the device.

Standby Mode

$P_{STAT} = P_{DC2}$

Sleep Mode

$P_{STAT} = P_{DC3}$

Total Dynamic Power Consumption—P_{DYN}

Operating Mode

$P_{DYN} = P_{CLOCK} + P_{S\text{-}CELL} + P_{C\text{-}CELL} + P_{NET} + P_{INPUTS} + P_{OUTPUTS} + P_{MEMORY} + P_{PLL} + P_{NVM} + P_{XTL\text{-}OSC} + P_{RC\text{-}OSC} + P_{AB}$

Standby Mode

$P_{DYN} = P_{XTL-OSC}$

Sleep Mode

$P_{DYN} = 0$ W

Global Clock Dynamic Contribution—P_{CLOCK}

Operating Mode

$P_{CLOCK} = (P_{AC1} + N_{SPINE} * P_{AC2} + N_{ROW} * PAC3 + N_{S\text{-}CELL} * P_{AC4}) * F_{CLK}$

N_{SPINE} is the number of global spines used in the user design—guidelines are provided in Table 3-16 on page 3-29.

N_{ROW} is the number of VersaTile rows used in the design—guidelines are provided in Table 3-16 on page 3-29.

F_{CLK} is the global clock signal frequency.

$N_{S\text{-}CELL}$ is the number of VersaTiles used as sequential modules in the design.

Standby Mode and Sleep Mode

$P_{CLOCK} = 0$ W

Sequential Cells Dynamic Contribution—$P_{S\text{-}CELL}$

Operating Mode

$P_{S\text{-}CELL} = N_{S\text{-}CELL} * (P_{AC5} + (\alpha_1 / 2) * P_{AC6}) * F_{CLK}$

$N_{S\text{-}CELL}$ is the number of VersaTiles used as sequential modules in the design. When a multi-tile sequential cell is used, it should be accounted for as 1.

α_1 is the toggle rate of VersaTile outputs—guidelines are provided in Table 3-16 on page 3-29.

F_{CLK} is the global clock signal frequency.

Standby Mode and Sleep Mode

$P_{S\text{-}CELL} = 0$ W

Combinatorial Cells Dynamic Contribution—$P_{C\text{-}CELL}$

Operating Mode

$P_{C\text{-}CELL} = N_{C\text{-}CELL} * (\alpha_1 / 2) * P_{AC7} * F_{CLK}$

$N_{C\text{-}CELL}$ is the number of VersaTiles used as combinatorial modules in the design.

α_1 is the toggle rate of VersaTile outputs—guidelines are provided in Table 3-16 on page 3-29.

F_{CLK} is the global clock signal frequency.

Standby Mode and Sleep Mode

$P_{C\text{-}CELL} = 0$ W

Routing Net Dynamic Contribution—P_{NET}

Operating Mode

$P_{NET} = (N_{S\text{-}CELL} + N_{C\text{-}CELL}) * (\alpha_1 / 2) * P_{AC8} * F_{CLK}$

$N_{S\text{-}CELL}$ is the number VersaTiles used as sequential modules in the design.

$N_{C\text{-}CELL}$ is the number of VersaTiles used as combinatorial modules in the design.

α_1 is the toggle rate of VersaTile outputs—guidelines are provided in Table 3-16 on page 3-29.

F_{CLK} is the global clock signal frequency.

Standby Mode and Sleep Mode

P_{NET} = 0 W

I/O Input Buffer Dynamic Contribution—P_{INPUTS}

Operating Mode

$P_{INPUTS} = N_{INPUTS} * (\alpha_2 / 2) * P_{AC9} * F_{CLK}$

N_{INPUTS} is the number of I/O input buffers used in the design.

α_2 is the I/O buffer toggle rate—guidelines are provided in Table 3-16 on page 3-29.

F_{CLK} is the global clock signal frequency.

Standby Mode and Sleep Mode

P_{INPUTS} = 0 W

I/O Output Buffer Dynamic Contribution—$P_{OUTPUTS}$

Operating Mode

$P_{OUTPUTS} = N_{OUTPUTS} * (\alpha_2 / 2) * \beta_1 * P_{AC10} * F_{CLK}$

$N_{OUTPUTS}$ is the number of I/O output buffers used in the design.

α_2 is the I/O buffer toggle rate—guidelines are provided in Table 3-16 on page 3-29.

β_1 is the I/O buffer enable rate—guidelines are provided in Table 3-17 on page 3-29.

F_{CLK} is the global clock signal frequency.

Standby Mode and Sleep Mode

$P_{OUTPUTS}$ = 0 W

RAM Dynamic Contribution—P_{MEMORY}

Operating Mode

$P_{MEMORY} = (N_{BLOCKS} * P_{AC11} * \beta_2 * F_{READ\text{-}CLOCK}) + (N_{BLOCKS} * P_{AC12} * \beta_3 * F_{WRITE\text{-}CLOCK})$

N_{BLOCKS} is the number of RAM blocks used in the design.

$F_{READ\text{-}CLOCK}$ is the memory read clock frequency.

β_2 is the RAM enable rate for read operations—guidelines are provided in Table 3-17 on page 3-29.

β_3 the RAM enable rate for write operations—guidelines are provided in Table 3-17 on page 3-29.

$F_{WRITE\text{-}CLOCK}$ is the memory write clock frequency.

Standby Mode and Sleep Mode

P_{MEMORY} = 0 W

PLL/CCC Dynamic Contribution—P_{PLL}

Operating Mode

$P_{PLL} = P_{AC13} * F_{CLKOUT}$

F_{CLKIN} is the input clock frequency.

F_{CLKOUT} is the output clock frequency.[1]

Standby Mode and Sleep Mode

P_{PLL} = 0 W

1. *The PLL dynamic contribution depends on the input clock frequency, the number of output clock signals generated by the PLL, and the frequency of each output clock. If a PLL is used to generate more than one output clock, include each output clock in the formula output clock by adding its corresponding contribution ($P_{AC14} * F_{CLKOUT}$ product) to the total PLL contribution.*

Nonvolatile Memory Dynamic Contribution—P_{NVM}

Operating Mode

The NVM dynamic power consumption is a piecewise linear function of frequency.

$P_{NVM} = N_{NVM\text{-}BLOCKS} * \beta_4 * P_{AC15} * F_{READ\text{-}NVM}$ when $F_{READ\text{-}NVM} \leq 33$ MHz,

$P_{NVM} = N_{NVM\text{-}BLOCKS} * \beta_4 * (P_{AC16} + P_{AC17} * F_{READ\text{-}NVM})$when $F_{READ\text{-}NVM} > 33$ MHz

$N_{NVM\text{-}BLOCKS}$ is the number of NVM blocks used in the design (2 inAFS600).

β_4 is the NVM enable rate for read operations. Default is 0 (NVM mainly in idle state).

$F_{READ\text{-}NVM}$ is the NVM read clock frequency.

Standby Mode and Sleep Mode

$P_{NVM} = 0$ W

Crystal Oscillator Dynamic Contribution—$P_{XTL\text{-}OSC}$

Operating Mode

$P_{XTL\text{-}OSC} = P_{AC18}$

Standby Mode

$P_{XTL\text{-}OSC} = P_{AC18}$

Sleep Mode

$P_{XTL\text{-}OSC} = 0$ W

RC Oscillator Dynamic Contribution—$P_{RC\text{-}OSC}$

Operating Mode

$P_{RC\text{-}OSC} = P_{AC19}$

Standby Mode and Sleep Mode

$P_{RC\text{-}OSC} = 0$ W

Analog System Dynamic Contribution—P_{AB}

Operating Mode

$P_{AB} = P_{AC20}$

Standby Mode and Sleep Mode

$P_{AB} = 0$ W

Guidelines

Toggle Rate Definition

A toggle rate defines the frequency of a net or logic element relative to a clock. It is a percentage. If the toggle rate of a net is 100%, this means that the net switches at half the clock frequency. Below are some examples:

- The average toggle rate of a shift register is 100%, as all flip-flop outputs toggle at half of the clock frequency.
- The average toggle rate of an 8-bit counter is 25%:
  - Bit 0 (LSB) = 100%
  - Bit 1 = 50%
  - Bit 2 = 25%
  - …
  - Bit 7 (MSB) = 0.78125%
  - Average toggle rate = (100% + 50% + 25% + 12.5% + . . . 0.78125%) / 8.

Enable Rate Definition

Output enable rate is the average percentage of time during which tristate outputs are enabled. When non-tristate output buffers are used, the enable rate should be 100%.

Table 3-16 • **Toggle Rate Guidelines Recommended for Power Calculation**

Component	Definition	Guideline
α_1	Toggle rate of VersaTile outputs	10%
α_2	I/O buffer toggle rate	10%

Table 3-17 • **Enable Rate Guidelines Recommended for Power Calculation**

Component	Definition	Guideline
β_1	I/O output buffer enable rate	100%
β_2	RAM enable rate for read operations	12.5%
β_3	RAM enable rate for write operations	12.5%
β_4	NVM enable rate for read operations	0%

Example of Power Calculation

This example considers a shift register with 5,000 storage tiles, including a counter and memory that stores analog information. The shift register is clocked at 50 MHz and stores and reads information from a RAM.

The device used is a commercial AFS600 device operating in typical conditions.

The calculation below uses the power calculation methodology previously presented and shows how to determine the dynamic and static power consumption of resources used in the application.

Also included in the example is the calculation of power consumption in operating, standby, and sleep modes to illustrate the benefit of power-saving modes.

Global Clock Contribution—P_{CLOCK}

F_{CLK} = 50 MHz

Number of sequential VersaTiles: $N_{S\text{-}CELL}$ = 5,000

Estimated number of Spines: N_{SPINES} = 5

Estimated number of Rows: N_{ROW} = 313

Operating Mode

$P_{CLOCK} = (P_{AC1} + N_{SPINE} * P_{AC2} + N_{ROW} * PAC3 + N_{S\text{-}CELL} * P_{AC4}) * F_{CLK}$

$P_{CLOCK} = (0.0128 + 5 * 0.0019 + 313 * 0.00081 + 5{,}000 * 0.00011) * 50$

$P_{CLOCK} = 41.28$ mW

Standby Mode and Sleep Mode

$P_{CLOCK} = 0$ W

Logic—Sequential Cells, Combinational Cells, and Routing Net Contributions—$P_{S\text{-}CELL}$, $P_{C\text{-}CELL}$, and P_{NET}

F_{CLK} = 50 MHz

Number of sequential VersaTiles: $N_{S\text{-}CELL}$ = 5,000

Number of combinatorial VersaTiles: $N_{C\text{-}CELL}$ = 6,000

Estimated toggle rate of VersaTile outputs: α_1 = 0.1 (10%)

Operating Mode

$P_{S\text{-}CELL} = N_{S\text{-}CELL} * (P_{AC5} + (\alpha_1 / 2) * P_{AC6}) * F_{CLK}$

$P_{S\text{-}CELL} = 5{,}000 * (0.00007 + (0.1 / 2) * 0.00029) * 50$

$P_{S\text{-}CELL} = 21.13$ mW

$P_{C\text{-}CELL} = N_{C\text{-}CELL} * (\alpha_1 / 2) * P_{AC7} * F_{CLK}$

$P_{C\text{-}CELL} = 6{,}000 * (0.1 / 2) * 0.00029 * 50$

$P_{C\text{-}CELL} = 4.35$ mW

$P_{NET} = (N_{S\text{-}CELL} + N_{C\text{-}CELL}) * (\alpha_1 / 2) * P_{AC8} * F_{CLK}$

$P_{NET} = (5{,}000 + 6{,}000) * (0.1 / 2) * 0.0007 * 50$

$P_{NET} = 19.25$ mW

$P_{LOGIC} = P_{S\text{-}CELL} + P_{C\text{-}CELL} + P_{NET}$

$P_{LOGIC} = 21.13$ mW + 4.35 mW + 19.25 mW

$P_{LOGIC} = 44.73$ mW

Standby Mode and Sleep Mode

$P_{S\text{-}CELL}$ = 0 W

$P_{C\text{-}CELL}$ = 0 W

P_{NET} = 0 W

P_{LOGIC} = 0 W

I/O Input and Output Buffer Contribution—$P_{I/O}$

This example uses LVTTL 3.3 V I/O cells. The output buffers are 12 mA–capable, configured with high output slew and driving a 35 pF output load.

F_{CLK} = 50 MHz

Number of input pins used: N_{INPUTS} = 30

Number of output pins used: $N_{OUTPUTS}$ = 40

Estimated I/O buffer toggle rate: α_2 = 0.1 (10%)

Estimated IO buffer enable rate: β_1 = 1 (100%)

Operating Mode

$P_{INPUTS} = N_{INPUTS} * (\alpha_2 / 2) * P_{AC9} * F_{CLK}$

$P_{INPUTS} = 30 * (0.1 / 2) * 0.01739 * 50$

P_{INPUTS} = 1.30 mW

$P_{OUTPUTS} = N_{OUTPUTS} * (\alpha_2 / 2) * \beta_1 * P_{AC10} * F_{CLK}$

$P_{OUTPUTS} = 40 * (0.1 / 2) * 1 * 0.4747 * 50$

$P_{OUTPUTS}$ = 47.47 mW

$P_{I/O} = P_{INPUTS} + P_{OUTPUTS}$

$P_{I/O}$ = 1.30 mW + 47.47 mW

$P_{I/O}$ = 48.77 mW

Standby Mode and Sleep Mode

P_{INPUTS} = 0 W

$P_{OUTPUTS}$ = 0 W

$P_{I/O}$ = 0 W

RAM Contribution—P_{MEMORY}

Frequency of Read Clock: $F_{READ\text{-}CLOCK}$ = 10 MHz

Frequency of Write Clock: $F_{WRITE\text{-}CLOCK}$ = 10 MHz

Number of RAM blocks: N_{BLOCKS} = 20

Estimated RAM Read Enable Rate: β_2 = 0.125 (12.5%)

Estimated RAM Write Enable Rate: β_3 = 0.125 (12.5%)

Operating Mode

$P_{MEMORY} = (N_{BLOCKS} * P_{AC11} * \beta_2 * F_{READ\text{-}CLOCK}) + (N_{BLOCKS} * P_{AC12} * \beta_3 * F_{WRITE\text{-}CLOCK})$

$P_{MEMORY} = (20 * 0.025 * 0.125 * 10) + (20 * 0.030 * 0.125 * 10)$

P_{MEMORY} = 1.38 mW

Standby Mode and Sleep Mode

P_{MEMORY} = 0 W

PLL/CCC Contribution—P_{PLL}

PLL is not used in this application.

P_{PLL} = 0 W

Nonvolatile Memory—P_{NVM}

Nonvolatile memory is not used in this application.

P_{NVM} = 0 W

Crystal Oscillator—$P_{XTL-OSC}$

The application utilizes standby mode. The crystal oscillator is assumed to be active.

Operating Mode

$P_{XTL-OSC} = P_{AC18}$

$P_{XTL-OSC}$ = 0.63 mW

Standby Mode

$P_{XTL-OSC} = P_{AC18}$

$P_{XTL-OSC}$ = 0.63 mW

Sleep Mode

$P_{XTL-OSC}$ = 0 W

RC Oscillator—P_{RC-OSC}

Operating Mode

$P_{RC-OSC} = P_{AC19}$

P_{RC-OSC} = 3.30 mW

Standby Mode and Sleep Mode

P_{RC-OSC} = 0 W

Analog System—P_{AB}

Number of Quads used: N_{QUADS} = 4

Operating Mode

$P_{AB} = P_{AC20}$

P_{AB} = 3.00 mW

Standby Mode and Sleep Mode

P_{AB} = 0 W

Total Dynamic Power Consumption—P_{DYN}

Operating Mode

$P_{DYN} = P_{CLOCK} + P_{S-CELL} + P_{C-CELL} + P_{NET} + P_{INPUTS} + P_{OUTPUTS} + P_{MEMORY} + P_{PLL} + P_{NVM} + P_{XTL-OSC} + P_{RC-OSC} + P_{AB}$

P_{DYN} = 41.28 mW + 21.1 mW + 4.35 mW + 19.25 mW + 1.30 mW + 47.47 mW + 1.38 mW + 0 + 0 + 0.63 mW + 3.30 mW + 3.00 mW

P_{DYN} = 143.06 mW

Standby Mode

$P_{DYN} = P_{XTL-OSC}$

P_{DYN} = 0.63 mW

Sleep Mode

P_{DYN} = 0 W

Total Static Power Consumption—P_{STAT}

Number of Quads used: N_{QUADS} = 4

Number of NVM blocks available (AFS600): $N_{NVM\text{-}BLOCKS}$ = 2

Number of input pins used: N_{INPUTS} = 30

Number of output pins used: $N_{OUTPUTS}$ = 40

Operating Mode

$P_{STAT} = P_{DC1} + (N_{NVM\text{-}BLOCKS} * P_{DC4}) + P_{DC5} + (N_{QUADS} * P_{DC6}) + (N_{INPUTS} * P_{DC7}) + (N_{OUTPUTS} * P_{DC8})$

P_{STAT} = 7.50 mW + (2 * 1.19 mW) + 8.25 mW + (4 * 3.30 mW) + (30 * 0.00) + (40 * 0.00)

P_{STAT} = 31.33 mW

Standby Mode

$P_{STAT} = P_{DC2}$

P_{STAT} = 0.03 mW

Sleep Mode

$P_{STAT} = P_{DC3}$

P_{STAT} = 0.03 mW

Total Power Consumption—P_{TOTAL}

In operating mode, the total power consumption of the device is 174.39 mW:

$P_{TOTAL} = P_{STAT} + P_{DYN}$

P_{TOTAL} = 143.06 mW + 31.33 mW

P_{TOTAL} = 174.39 mW

In standby mode, the total power consumption of the device is limited to 0.66 mW:

$P_{TOTAL} = P_{STAT} + P_{DYN}$

P_{TOTAL} = 0.03 mW + 0.63 mW

P_{TOTAL} = 0.66 mW

In sleep mode, the total power consumption of the device drops as low as 0.03 mW:

$P_{TOTAL} = P_{STAT} + P_{DYN}$

P_{TOTAL} = 0.03 mW

Power Consumption

Table 3-18 • **Power Consumption**

Parameter	Description	Condition	Min.	Typical	Max.	Units
Crystal Oscillator						
$I_{STBXTAL}$	Standby Current of Crystal Oscillator			10		µA
$I_{DYNXTAL}$	Operating Current	RC		0.6		mA
		0.032–0.2		0.19		mA
		0.2–2.0		0.6		mA
		2.0–20.0		0.6		mA
RC Oscillator						
I_{DYNRC}	Operating Current			1		mA
ACM						
	Operating Current (fixed clock)			200		µA/MHz
	Operating Current (user clock)			30		µA
NVM System						
	NVM Array Operating Power	Idle		795		µA
		Read operation		See Table 3-15 on page 3-24.		See Table 3-15 on page 3-24.
		Erase		900		µA
		Write		900		µA
$P_{NVMCTRL}$	NVM Controller Operating Power			20		µW/MHz

Part Number and Revision Date

Part Number 51700092-015-1
Revised July 2009

List of Changes

The following table lists critical changes that were made in the current version of the document.

Previous Version	Changes in Current Version (v2.0)	Page
Preliminary v1.7 (October 2008)	The MicroBlade and Fusion datasheets have been combined. Pigeon Point information is new. CoreMP7 support was removed since it is no longer offered. –F was removed from the datasheet since it is no longer offered. The operating temperature was changed from ambient to junction to better reflect actual conditions of operations. Commercial: 0°C to 85°C Industrial: –40°C to 100°C The version number category was changed from Preliminary to Production, which means the datasheet contains information based on final characterization. The version number changed from Preliminary v1.7 to v2.0.	N/A
	$V_{CC33PMP}$ was added to Table 3-1 · Absolute Maximum Ratings. In addition, conditions for AV, AC, AG, and AT were also updated.	3-1
	$V_{CC33PMP}$ was added to Table 3-2 · Recommended Operating Conditions. In addition, conditions for AV, AC, AG, and AT were also updated.	3-3
	Table 3-5 · FPGA Programming, Storage, and Operating Limits was updated to include new data and the temperature ranges were changed. The notes were removed from the table.	3-5
	Table 3-6 · Package Thermal Resistance was updated to include new data.	3-7
	In EQ 3-4 to EQ 3-6, the junction temperature was changed from 110°C to 100°C.	3-8 to 3-9
	Table 3-8 · AFS1500 Quiescent Supply Current Characteristics through Table 3-11 · AFS090 Quiescent Supply Current Characteristics are new and have replaced the Quiescent Supply Current Characteristics (IDDQ) table.	3-10 to 3-16
	In Table 3-14 · Different Components Contributing to the Dynamic Power Consumption in Fusion Devices, the power supply for P_{AC9} and P_{AC10} were changed from VMV/V_{CC} to V_{CCI}.	3-22
	In Table 3-15 · Different Components Contributing to the Static Power Consumption in Fusion Devices, the power supply for P_{DC7} and P_{DC8} were changed from VMV/V_{CC} to V_{CCI}. P_{DC1} was updated from TBD to 18.	3-24
Advance v1.6 (August 2008)	The version number category was changed from Advance to Preliminary, which means the datasheet contains information based on simulation and/or initial characterization. The information is believed to be correct, but changes are possible.	N/A
Advance v1.4 (July 2008)	The title of the datasheet changed from Actel Programmable System Chips to Actel Fusion Mixed-Signal FPGAs. In addition, all instances of programmable system chip were changed to mixed-signal FPGA.	N/A
Advance v1.3 (July 2008)	In Table 3-8 · Quiescent Supply Current Characteristics (IDDQ)1, footnote references were updated for I_{DC2} and I_{DC3}. Footnote 3 and 4 were updated and footnote 5 is new.	3-11

Previous Version	Changes in Current Version (v2.0)	Page
Advance v1.1	Table 3-6 · Package Thermal Resistance was significantly updated	3-7
	Table 3-14 · Different Components Contributing to the Dynamic Power Consumption in Fusion Devices was significantly updated.	3-22
	Table 3-16 · Toggle Rate Guidelines Recommended for Power Calculation was significantly updated.	3-29
Advance v0.9	In Table 3-1 · Absolute Maximum Ratings, the AT for the Unpowered, ADC reset asserted or unconfigured parameter, –11 was changed to –0.4.	3-1
	The units column of Table 3-2 · Recommended Operating Conditions was incomplete in the previous version. V was added to all the rows. In addition, AT for the Unpowered, ADC reset asserted or unconfigured parameter, –10.5 was changed to –0.3. Note 6 was updated to include V_{CC15A}.	3-3
	In the title of Table 3-3 · Input Resistance of Analog Pads, Impedance was changed to Resistance.	3-4
	In Table 3-5 · FPGA Programming, Storage, and Operating Limits, note 2 is new. "Program" was removed from the table heading in the Retention column.	3-5
	The "PLL Behavior at Brownout Condition" section is new.	3-5
	Table 3-7 · Temperature and Voltage Derating Factors for Timing Delays was updated.	3-9
	In the Table 3-12 · Summary of I/O Input Buffer Power (per pin)—Default I/O Software Settings, the HSTL (I) for the Static Power PDC7 (mW) was changed from 0.1 to 0.17.	3-18
	The Table 3-14 · Different Components Contributing to the Dynamic Power Consumption in Fusion Devices was updated.	3-22
	The Table 3-15 · Different Components Contributing to the Static Power Consumption in Fusion Devices was updated.	3-24
	In the "PLL/CCC Dynamic Contribution—P_{PLL}" section, P_{AC14} was deleted.	3-27
Advance v0.8 (June 2007)	In Table 3-6 · Package Thermal Resistance, the data for the following device/packages were updated: AFS090-FG256 AFS250-FG256 AFS600-FG256 AFS1500-FG256 AFS600-FG484 AFS1500-FG484 AFS1500-FG676	3-7
Advance v0.7 (January 2007)	The VMV pins have now been tied internally with the V_{CCI} pins.	N/A
	The V_{COMPLF} pin description was deleted.	N/A
	Table 3-1 · Absolute Maximum Ratings, Table 3-2 · Recommended Operating Conditions, and Table 3-3 · Input Resistance of Analog Pads were updated.	3-1 to 3-4
	Table 3-5 · FPGA Programming, Storage, and Operating Limits was updated.	3-5
	P_{AC13} and P_{AC14} were updated in Table 3-14 · Different Components Contributing to the Dynamic Power Consumption in Fusion Devices.	3-22
	The Operating Mode for the "PLL/CCC Dynamic Contribution—P_{PLL}" section was updated.	3-27
	Table 3-18 · Power Consumption was updated to change the typical value of $I_{DYNXTAL}$ for 0.032–0.2 MHz to 0.19.	3-34

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.5 (June 2006)	Table 3-3 · Input Resistance of Analog Pads is new.	3-4
Advance v0.4 (April 2006)	The low power modes of operation were updated and clarified.	N/A
Advance v0.2 (April 2006)	Table 3-8 · Quiescent Supply Current Characteristics (IDDQ)1 was updated.	3-11
	Table 3-14 · Different Components Contributing to the Dynamic Power Consumption in Fusion Devices was updated.	3-22
	Table 3-14 · Different Components Contributing to the Dynamic Power Consumption in Fusion Devices was updated.	3-22
	The "Example of Power Calculation" was updated.	3-30
	The Analog System information was deleted from Table 3-18 · Power Consumption.	3-34

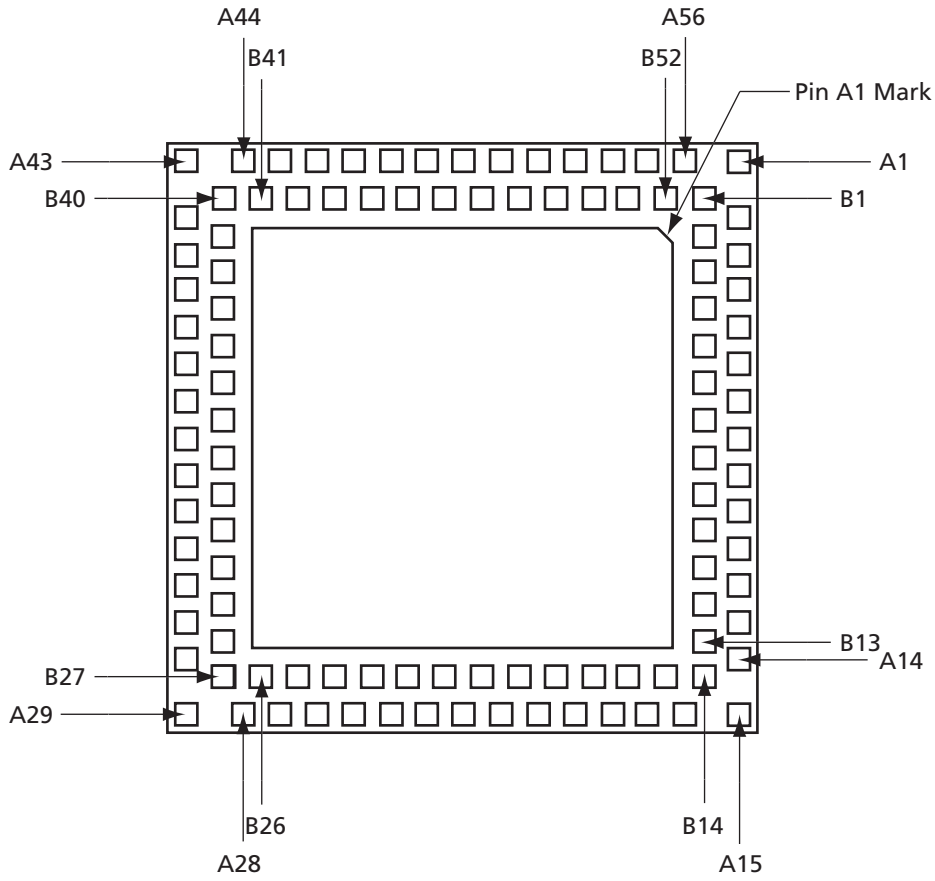
Actel Safety Critical, Life Support, and High-Reliability Applications Policy

The Actel products described in this advance status datasheet may not have completed Actel's qualification process. Actel may amend or enhance products during the product introduction and qualification process, resulting in changes in device functionality or performance. It is the responsibility of each customer to ensure the fitness of any Actel product (but especially a new product) for a particular purpose, including appropriateness for safety-critical, life-support, and other high-reliability applications. Consult Actel's Terms and Conditions for specific liability exclusions relating to life-support applications. A reliability report covering all of Actel's products is available on the Actel website at http://www.actel.com/documents/ORT_Report.pdf. Actel also offers a variety of enhanced qualification and lot acceptance screening procedures. Contact your local Actel sales office for additional reliability information.

Rate This Document

4 – Package Pin Assignments

108-Pin QFN



Note: The die attach paddle center of the package is tied to ground (GND).

Note

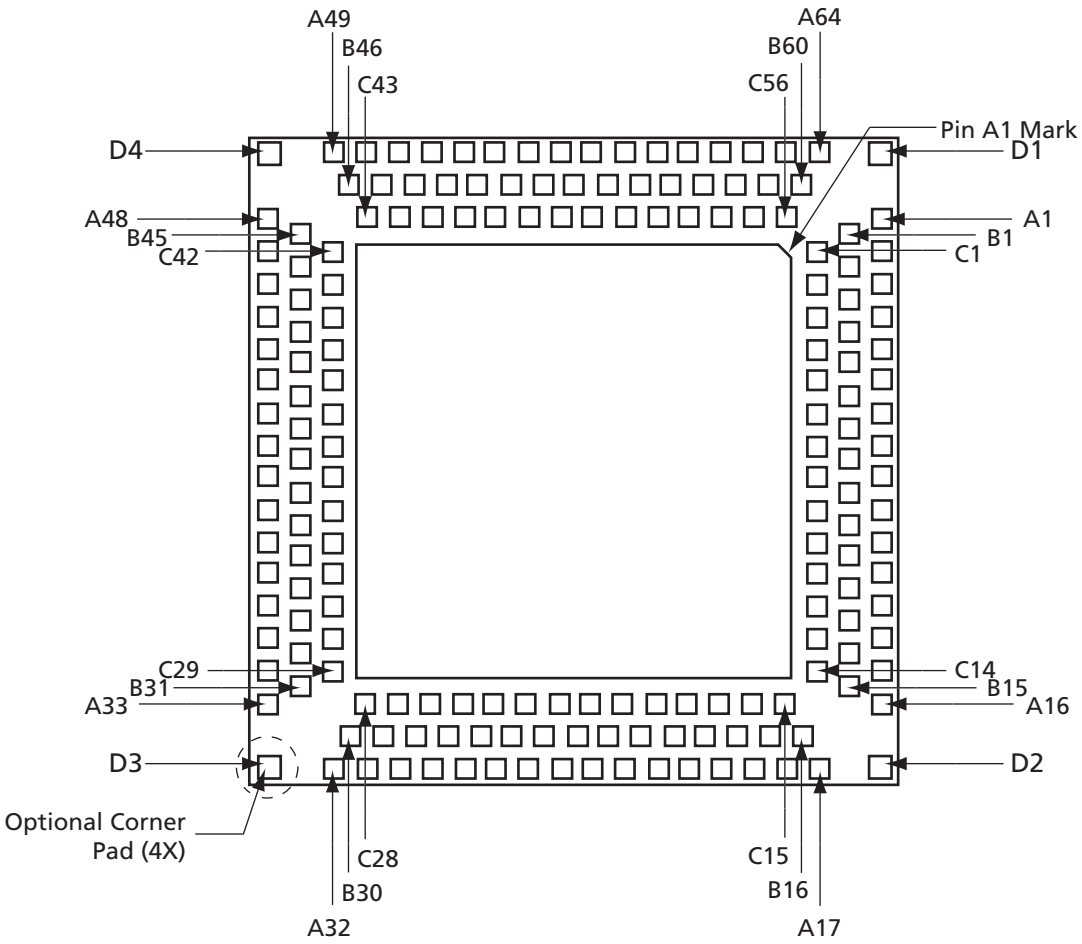
For Package Manufacturing and Environmental information, visit the Resource Center at http://www.actel.com/products/solutions/package/default.aspx.

108-Pin QFN	
Pin Number	**AFS090 Function**
A1	NC
A2	GNDQ
A3	GAA2/IO52PDB3V0
A4	GND
A5	GFA1/IO47PDB3V0
A6	GEB1/IO45PDB3V0
A7	VCCOSC
A8	XTAL2
A9	GEA1/IO44PPB3V0
A10	GEA0/IO44NPB3V0
A11	GEB2/IO42PDB3V0
A12	V_{CCNVM}
A13	V_{CC15A}
A14	PCAP
A15	NC
A16	GNDA
A17	AV0
A18	AG0
A19	ATRTN0
A20	AT1
A21	AC1
A22	AV2
A23	AG2
A24	AT2
A25	AT3
A26	AC3
A27	GNDAQ
A28	ADCGNDREF
A29	NC
A30	GNDA
A31	PTEM
A32	GNDNVM
A33	V_{PUMP}
A34	TCK
A35	TMS
A36	TRST
A37	GDB1/IO39PSB1V0
A38	GDC1/IO38PDB1V0

108-Pin QFN	
Pin Number	**AFS090 Function**
A39	GND
A40	GCB1/IO35PDB1V0
A41	GCB2/IO33PDB1V0
A42	GBA2/IO31PDB1V0
A43	NC
A44	GBA1/IO30RSB0V0
A45	GBB1/IO28RSB0V0
A46	GND
A47	V_{CC}
A48	GBC1/IO26RSB0V0
A49	IO21RSB0V0
A50	IO19RSB0V0
A51	IO09RSB0V0
A52	GAC0/IO04RSB0V0
A53	V_{CCI}B0
A54	GND
A55	GAB0/IO02RSB0V0
A56	GAA0/IO00RSB0V0
B1	V_{COMPLA}
B2	V_{CCI}B3
B3	GAB2/IO52NDB3V0
B4	V_{CCI}B3
B5	GFA0/IO47NDB3V0
B6	GEB0/IO45NDB3V0
B7	XTAL1
B8	GNDOSC
B9	GEC2/IO43PSB3V0
B10	GEA2/IO42NDB3V0
B11	V_{CC}
B12	GNDNVM
B13	NCAP
B14	$V_{CC33PMP}$
B15	V_{CC33N}
B16	GNDAQ
B17	AC0
B18	AT0
B19	AG1
B20	AV1

108-Pin QFN	
Pin Number	**AFS090 Function**
B21	AC2
B22	ATRTN1
B23	AG3
B24	AV3
B25	V_{CC33A}
B26	VAREF
B27	PUB
B28	V_{CC33A}
B29	PTBASE
B30	V_{CCNVM}
B31	V_{CC}
B32	TDI
B33	TDO
B34	V_{JTAG}
B35	GDC0/IO38NDB1V0
B36	V_{CCI}B1
B37	GCB0/IO35NDB1V0
B38	GCC2/IO33NDB1V0
B39	GBB2/IO31NDB1V0
B40	V_{CCI}B1
B41	GNDQ
B42	GBA0/IO29RSB0V0
B43	V_{CCI}B0
B44	GBB0/IO27RSB0V0
B45	GBC0/IO25RSB0V0
B46	IO20RSB0V0
B47	IO10RSB0V0
B48	GAC1/IO05RSB0V0
B49	GAB1/IO03RSB0V0
B50	V_{CC}
B51	GAA1/IO01RSB0V0
B52	V_{CCPLA}

180-Pin QFN



Note: The die attach paddle center of the package is tied to ground (GND).

Note

For Package Manufacturing and Environmental information, visit the Resource Center at http://www.actel.com/products/solutions/package/default.aspx.

180-Pin QFN		
Pin Number	AFS090 Function	AFS250 Function
A1	GNDQ	GNDQ
A2	V_{CCI}B3	V_{CCI}B3
A3	GAB2/IO52NDB3V0	IO74NDB3V0
A4	GFA2/IO51NDB3V0	IO71NDB3V0
A5	GFC2/IO50NDB3V0	IO69NPB3V0
A6	V_{CCI}B3	V_{CCI}B3
A7	GFA1/IO47PPB3V0	GFB1/IO67PPB3V0
A8	GEB0/IO45NDB3V0	NC
A9	XTAL1	XTAL1
A10	GNDOSC	GNDOSC
A11	GEC2/IO43PPB3V0	GEA1/IO61PPB3V0
A12	IO43NPB3V0	GEA0/IO61NPB3V0
A13	NC	V_{CCI}B3
A14	GNDNVM	GNDNVM
A15	PCAP	PCAP
A16	$V_{CC33PMP}$	$V_{CC33PMP}$
A17	NC	NC
A18	AV0	AV0
A19	AG0	AG0
A20	ATRTN0	ATRTN0
A21	AG1	AG1
A22	AC1	AC1
A23	AV2	AV2
A24	AT2	AT2
A25	AT3	AT3
A26	AC3	AC3
A27	AV4	AV4
A28	AC4	AC4
A29	AT4	AT4
A30	NC	AG5
A31	NC	AV5
A32	ADCGNDREF	ADCGNDREF
A33	V_{CC33A}	V_{CC33A}
A34	GNDA	GNDA
A35	PTBASE	PTBASE
A36	V_{CCNVM}	V_{CCNVM}

180-Pin QFN		
Pin Number	AFS090 Function	AFS250 Function
A37	V_{PUMP}	V_{PUMP}
A38	TDI	TDI
A39	TDO	TDO
A40	V_{JTAG}	V_{JTAG}
A41	GDB1/IO39PPB1V0	GDA1/IO54PPB1V0
A42	GDC1/IO38PDB1V0	GDB1/IO53PDB1V0
A43	V_{CC}	V_{CC}
A44	GCB0/IO35NPB1V0	GCB0/IO48NPB1V0
A45	GCC1/IO34PDB1V0	GCC1/IO47PDB1V0
A46	V_{CCI}B1	V_{CCI}B1
A47	GBC2/IO32PPB1V0	GBB2/IO41PPB1V0
A48	V_{CCI}B1	V_{CCI}B1
A49	NC	NC
A50	GBA0/IO29RSB0V0	GBB1/IO37RSB0V0
A51	V_{CCI}B0	V_{CCI}B0
A52	GBB0/IO27RSB0V0	GBC0/IO34RSB0V0
A53	GBC1/IO26RSB0V0	IO33RSB0V0
A54	IO24RSB0V0	IO29RSB0V0
A55	IO21RSB0V0	IO26RSB0V0
A56	V_{CCI}B0	V_{CCI}B0
A57	IO15RSB0V0	IO21RSB0V0
A58	IO10RSB0V0	IO13RSB0V0
A59	IO07RSB0V0	IO10RSB0V0
A60	GAC0/IO04RSB0V0	IO06RSB0V0
A61	GAB1/IO03RSB0V0	GAC1/IO05RSB0V0
A62	V_{CC}	V_{CC}
A63	GAA1/IO01RSB0V0	GAB0/IO02RSB0V0
A64	NC	NC
B1	V_{COMPLA}	V_{COMPLA}
B2	GAA2/IO52PDB3V0	GAC2/IO74PDB3V0
B3	GAC2/IO51PDB3V0	GFA2/IO71PDB3V0
B4	GFB2/IO50PDB3V0	GFB2/IO70PSB3V0
B5	V_{CC}	V_{CC}
B6	GFC0/IO49NDB3V0	GFC0/IO68NDB3V0
B7	GEB1/IO45PDB3V0	NC
B8	V_{CCOSC}	V_{CCOSC}

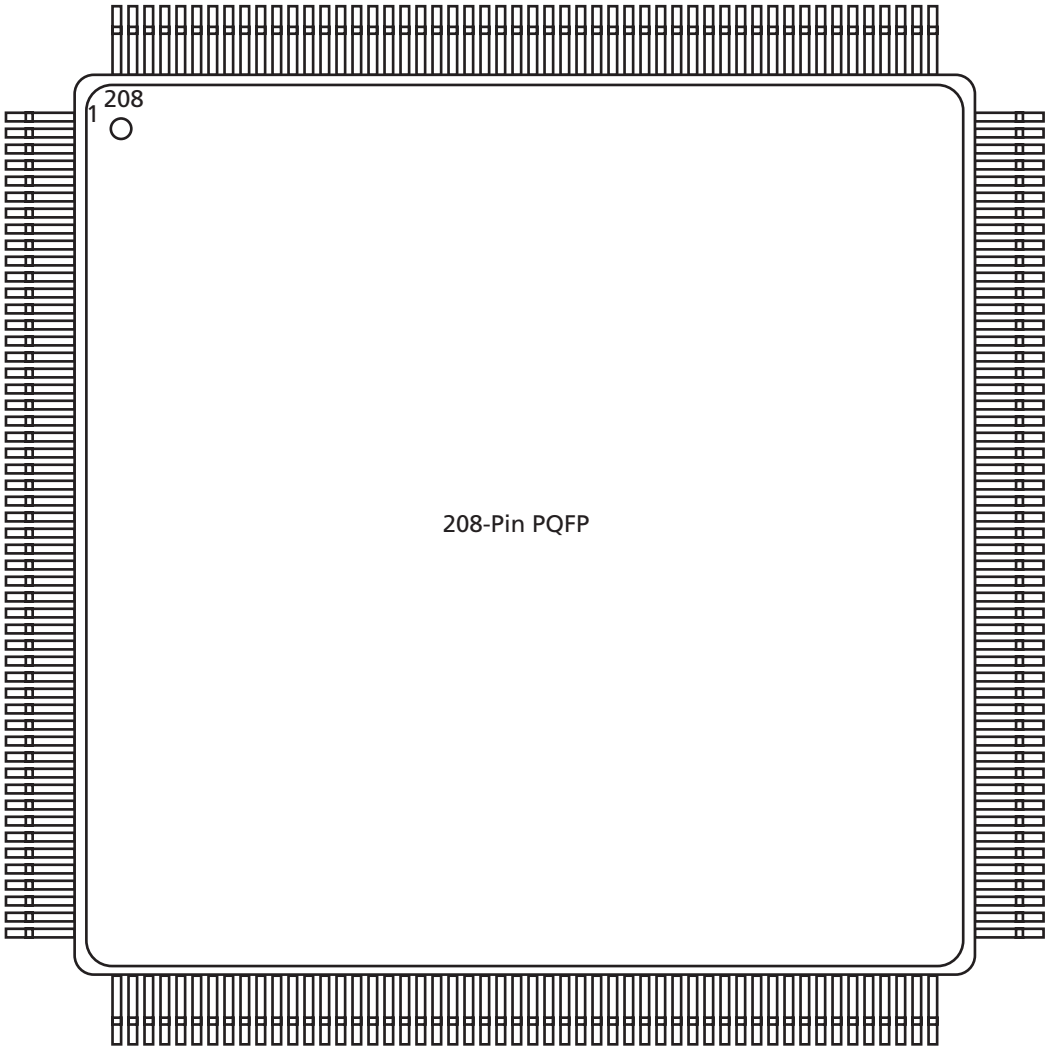
180-Pin QFN		
Pin Number	AFS090 Function	AFS250 Function
B9	XTAL2	XTAL2
B10	GEA0/IO44NDB3V0	GFA0/IO66NDB3V0
B11	GEB2/IO42PDB3V0	IO60NDB3V0
B12	V_{CC}	V_{CC}
B13	V_{CCNVM}	V_{CCNVM}
B14	V_{CC15A}	V_{CC15A}
B15	NCAP	NCAP
B16	VCC33N	VCC33N
B17	GNDAQ	GNDAQ
B18	AC0	AC0
B19	AT0	AT0
B20	AT1	AT1
B21	AV1	AV1
B22	AC2	AC2
B23	ATRTN1	ATRTN1
B24	AG3	AG3
B25	AV3	AV3
B26	AG4	AG4
B27	ATRTN2	ATRTN2
B28	NC	AC5
B29	V_{CC33A}	V_{CC33A}
B30	VAREF	VAREF
B31	PUB	PUB
B32	PTEM	PTEM
B33	GNDNVM	GNDNVM
B34	V_{CC}	V_{CC}
B35	TCK	TCK
B36	TMS	TMS
B37	TRST	TRST
B38	GDB2/IO41PSB1V0	GDA2/IO55PSB1V0
B39	GDC0/IO38NDB1V0	GDB0/IO53NDB1V0
B40	V_{CCI}B1	V_{CCI}B1
B41	GCA1/IO36PDB1V0	GCA1/IO49PDB1V0
B42	GCC0/IO34NDB1V0	GCC0/IO47NDB1V0
B43	GCB2/IO33PSB1V0	GBC2/IO42PSB1V0
B44	V_{CC}	V_{CC}

180-Pin QFN		
Pin Number	AFS090 Function	AFS250 Function
B45	GBA2/IO31PDB1V0	GBA2/IO40PDB1V0
B46	GNDQ	GNDQ
B47	GBA1/IO30RSB0V0	GBA0/IO38RSB0V0
B48	GBB1/IO28RSB0V0	GBC1/IO35RSB0V0
B49	V_{CC}	V_{CC}
B50	GBC0/IO25RSB0V0	IO31RSB0V0
B51	IO23RSB0V0	IO28RSB0V0
B52	IO20RSB0V0	IO25RSB0V0
B53	V_{CC}	V_{CC}
B54	IO11RSB0V0	IO14RSB0V0
B55	IO08RSB0V0	IO11RSB0V0
B56	GAC1/IO05RSB0V0	IO08RSB0V0
B57	V_{CCI}B0	V_{CCI}B0
B58	GAB0/IO02RSB0V0	GAC0/IO04RSB0V0
B59	GAA0/IO00RSB0V0	GAA1/IO01RSB0V0
B60	V_{CCPLA}	V_{CCPLA}
C1	NC	NC
C2	NC	V_{CCI}B3
C3	GND	GND
C4	NC	GFC2/IO69PPB3V0
C5	GFC1/IO49PDB3V0	GFC1/IO68PDB3V0
C6	GFA0/IO47NPB3V0	GFB0/IO67NPB3V0
C7	V_{CCI}B3	NC
C8	GND	GND
C9	GEA1/IO44PDB3V0	GFA1/IO66PDB3V0
C10	GEA2/IO42NDB3V0	GEC2/IO60PDB3V0
C11	NC	GEA2/IO58PSB3V0
C12	NC	NC
C13	GND	GND
C14	NC	NC
C15	NC	NC
C16	GNDA	GNDA
C17	NC	NC
C18	NC	NC
C19	NC	NC
C20	NC	NC

180-Pin QFN		
Pin Number	AFS090 Function	AFS250 Function
C21	AG2	AG2
C22	NC	NC
C23	NC	NC
C24	NC	NC
C25	NC	AT5
C26	GNDAQ	GNDAQ
C27	NC	NC
C28	NC	NC
C29	NC	NC
C30	NC	NC
C31	GND	GND
C32	NC	NC
C33	NC	NC
C34	NC	NC
C35	GND	GND
C36	GDB0/IO39NPB1V0	GDA0/IO54NPB1V0
C37	GDA1/IO37NSB1V0	GDC0/IO52NSB1V0
C38	GCA0/IO36NDB1V0	GCA0/IO49NDB1V0
C39	GCB1/IO35PPB1V0	GCB1/IO48PPB1V0
C40	GND	GND
C41	GCA2/IO32NPB1V0	IO41NPB1V0
C42	GBB2/IO31NDB1V0	IO40NDB1V0
C43	NC	NC
C44	NC	GBA1/IO39RSB0V0
C45	NC	GBB0/IO36RSB0V0
C46	GND	GND
C47	NC	IO30RSB0V0
C48	IO22RSB0V0	IO27RSB0V0
C49	GND	GND
C50	IO13RSB0V0	IO16RSB0V0
C51	IO09RSB0V0	IO12RSB0V0
C52	IO06RSB0V0	IO09RSB0V0
C53	GND	GND
C54	NC	GAB1/IO03RSB0V0
C55	NC	GAA0/IO00RSB0V0
C56	NC	NC

180-Pin QFN		
Pin Number	AFS090 Function	AFS250 Function
D1	NC	NC
D2	NC	NC
D3	NC	NC
D4	NC	NC

208-Pin PQFP



Note

For Package Manufacturing and Environmental information, visit the Resource Center at http://www.actel.com/products/solutions/package/default.aspx.

208-Pin PQFP		
Pin Number	AFS250 Function	AFS600 Function
1	V_{CCPLA}	V_{CCPLA}
2	V_{COMPLA}	V_{COMPLA}
3	GNDQ	GAA2/IO85PDB4V0
4	V_{CCI}B3	IO85NDB4V0
5	GAA2/IO76PDB3V0	GAB2/IO84PDB4V0
6	IO76NDB3V0	IO84NDB4V0
7	GAB2/IO75PDB3V0	GAC2/IO83PDB4V0
8	IO75NDB3V0	IO83NDB4V0
9	NC	IO77PDB4V0
10	NC	IO77NDB4V0
11	V_{CC}	IO76PDB4V0
12	GND	IO76NDB4V0
13	V_{CCI}B3	V_{CC}
14	IO72PDB3V0	GND
15	IO72NDB3V0	V_{CCI}B4
16	GFA2/IO71PDB3V0	GFA2/IO75PDB4V0
17	IO71NDB3V0	IO75NDB4V0
18	GFB2/IO70PDB3V0	GFC2/IO73PDB4V0
19	IO70NDB3V0	IO73NDB4V0
20	GFC2/IO69PDB3V0	V_{CCOSC}
21	IO69NDB3V0	XTAL1
22	V_{CC}	XTAL2
23	GND	GNDOSC
24	V_{CCI}B3	GFC1/IO72PDB4V0
25	GFC1/IO68PDB3V0	GFC0/IO72NDB4V0
26	GFC0/IO68NDB3V0	GFB1/IO71PDB4V0
27	GFB1/IO67PDB3V0	GFB0/IO71NDB4V0
28	GFB0/IO67NDB3V0	GFA1/IO70PDB4V0
29	V_{CCOSC}	GFA0/IO70NDB4V0
30	XTAL1	IO69PDB4V0
31	XTAL2	IO69NDB4V0
32	GNDOSC	V_{CC}
33	GEB1/IO62PDB3V0	GND
34	GEB0/IO62NDB3V0	V_{CCI}B4
35	GEA1/IO61PDB3V0	GEC1/IO63PDB4V0
36	GEA0/IO61NDB3V0	GEC0/IO63NDB4V0

208-Pin PQFP		
Pin Number	AFS250 Function	AFS600 Function
37	GEC2/IO60PDB3V0	GEB1/IO62PDB4V0
38	IO60NDB3V0	GEB0/IO62NDB4V0
39	GND	GEA1/IO61PDB4V0
40	V_{CCI}B3	GEA0/IO61NDB4V0
41	GEB2/IO59PDB3V0	GEC2/IO60PDB4V0
42	IO59NDB3V0	IO60NDB4V0
43	GEA2/IO58PDB3V0	V_{CCI}B4
44	IO58NDB3V0	GNDQ
45	V_{CC}	V_{CC}
45	V_{CC}	V_{CC}
46	V_{CCNVM}	V_{CCNVM}
47	GNDNVM	GNDNVM
48	GND	GND
49	V_{CC15A}	V_{CC15A}
50	PCAP	PCAP
51	NCAP	NCAP
52	$V_{CC33PMP}$	$V_{CC33PMP}$
53	VCC33N	VCC33N
54	GNDA	GNDA
55	GNDAQ	GNDAQ
56	NC	AV0
57	NC	AC0
58	NC	AG0
59	NC	AT0
60	NC	ATRTN0
61	NC	AT1
62	NC	AG1
63	NC	AC1
64	NC	AV1
65	AV0	AV2
66	AC0	AC2
67	AG0	AG2
68	AT0	AT2
69	ATRTN0	ATRTN1
70	AT1	AT3
71	AG1	AG3

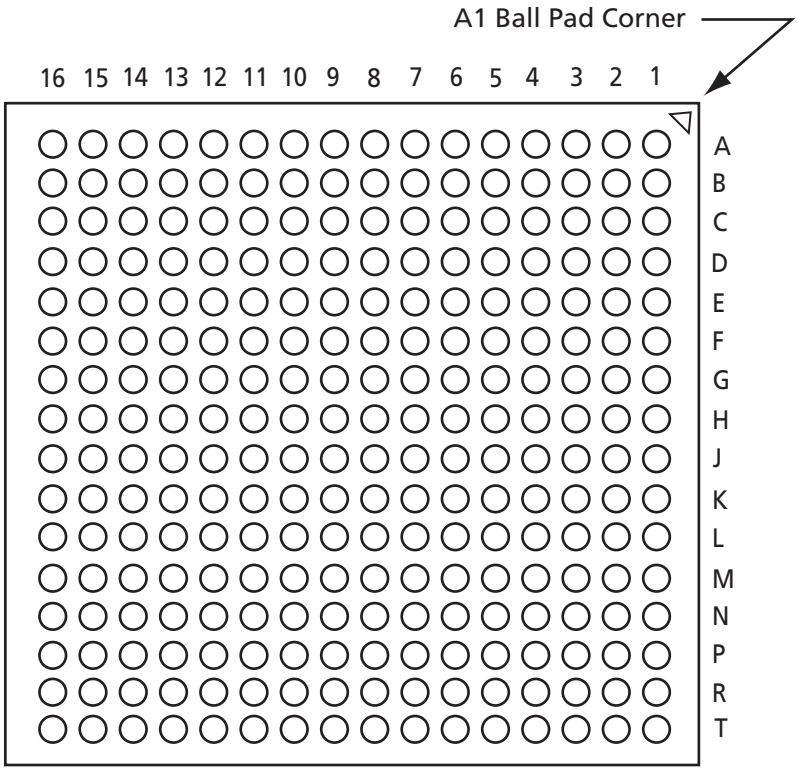
208-Pin PQFP		
Pin Number	AFS250 Function	AFS600 Function
72	AC1	AC3
73	AV1	AV3
74	AV2	AV4
75	AC2	AC4
76	AG2	AG4
77	AT2	AT4
78	ATRTN1	ATRTN2
79	AT3	AT5
80	AG3	AG5
81	AC3	AC5
82	AV3	AV5
83	AV4	AV6
84	AC4	AC6
85	AG4	AG6
86	AT4	AT6
87	ATRTN2	ATRTN3
88	AT5	AT7
89	AG5	AG7
90	AC5	AC7
91	AV5	AV7
92	NC	AV8
93	NC	AC8
94	NC	AG8
95	NC	AT8
96	NC	ATRTN4
97	NC	AT9
98	NC	AG9
99	NC	AC9
100	NC	AV9
101	GNDAQ	GNDAQ
102	V_{CC33A}	V_{CC33A}
103	ADCGNDREF	ADCGNDREF
104	VAREF	VAREF
105	PUB	PUB
106	V_{CC33A}	V_{CC33A}
107	GNDA	GNDA

208-Pin PQFP		
Pin Number	AFS250 Function	AFS600 Function
108	PTEM	PTEM
109	PTBASE	PTBASE
110	GNDNVM	GNDNVM
111	V_{CCNVM}	V_{CCNVM}
112	V_{CC}	V_{CC}
112	V_{CC}	V_{CC}
113	V_{PUMP}	V_{PUMP}
114	GNDQ	NC
115	V_{CCI}B1	TCK
116	TCK	TDI
117	TDI	TMS
118	TMS	TDO
119	TDO	TRST
120	TRST	V_{JTAG}
121	V_{JTAG}	IO57NDB2V0
122	IO57NDB1V0	GDC2/IO57PDB2V0
123	GDC2/IO57PDB1V0	IO56NDB2V0
124	IO56NDB1V0	GDB2/IO56PDB2V0
125	GDB2/IO56PDB1V0	IO55NDB2V0
126	V_{CCI}B1	GDA2/IO55PDB2V0
127	GND	GDA0/IO54NDB2V0
128	IO55NDB1V0	GDA1/IO54PDB2V0
129	GDA2/IO55PDB1V0	V_{CCI}B2
130	GDA0/IO54NDB1V0	GND
131	GDA1/IO54PDB1V0	V_{CC}
132	GDB0/IO53NDB1V0	GCA0/IO45NDB2V0
133	GDB1/IO53PDB1V0	GCA1/IO45PDB2V0
134	GDC0/IO52NDB1V0	GCB0/IO44NDB2V0
135	GDC1/IO52PDB1V0	GCB1/IO44PDB2V0
136	IO51NSB1V0	GCC0/IO43NDB2V0
137	V_{CCI}B1	GCC1/IO43PDB2V0
138	GND	IO42NDB2V0
139	V_{CC}	IO42PDB2V0
140	IO50NDB1V0	IO41NDB2V0
141	IO50PDB1V0	GCC2/IO41PDB2V0
142	GCA0/IO49NDB1V0	V_{CCI}B2

208-Pin PQFP		
Pin Number	AFS250 Function	AFS600 Function
143	GCA1/IO49PDB1V0	GND
144	GCB0/IO48NDB1V0	V_{CC}
145	GCB1/IO48PDB1V0	IO40NDB2V0
146	GCC0/IO47NDB1V0	GCB2/IO40PDB2V0
147	GCC1/IO47PDB1V0	IO39NDB2V0
148	IO42NDB1V0	GCA2/IO39PDB2V0
149	GBC2/IO42PDB1V0	IO31NDB2V0
150	V_{CCI}B1	GBB2/IO31PDB2V0
151	GND	IO30NDB2V0
152	V_{CC}	GBA2/IO30PDB2V0
153	IO41NDB1V0	V_{CCI}B2
154	GBB2/IO41PDB1V0	GNDQ
155	IO40NDB1V0	V_{COMPLB}
156	GBA2/IO40PDB1V0	V_{CCPLB}
157	GBA1/IO39RSB0V0	V_{CCI}B1
158	GBA0/IO38RSB0V0	GNDQ
159	GBB1/IO37RSB0V0	GBB1/IO27PPB1V1
160	GBB0/IO36RSB0V0	GBA1/IO28PPB1V1
161	GBC1/IO35RSB0V0	GBB0/IO27NPB1V1
162	V_{CCI}B0	GBA0/IO28NPB1V1
163	GND	V_{CCI}B1
164	V_{CC}	GND
165	GBC0/IO34RSB0V0	V_{CC}
166	IO33RSB0V0	GBC1/IO26PDB1V1
167	IO32RSB0V0	GBC0/IO26NDB1V1
168	IO31RSB0V0	IO24PPB1V1
169	IO30RSB0V0	IO23PPB1V1
170	IO29RSB0V0	IO24NPB1V1
171	IO28RSB0V0	IO23NPB1V1
172	IO27RSB0V0	IO22PPB1V0
173	IO26RSB0V0	IO21PPB1V0
174	IO25RSB0V0	IO22NPB1V0
175	V_{CCI}B0	IO21NPB1V0
176	GND	IO20PSB1V0
177	V_{CC}	IO19PSB1V0
178	IO24RSB0V0	IO14NSB0V1

208-Pin PQFP		
Pin Number	AFS250 Function	AFS600 Function
179	IO23RSB0V0	IO12PDB0V1
180	IO22RSB0V0	IO12NDB0V1
181	IO21RSB0V0	V_{CCI}B0
182	IO20RSB0V0	GND
183	IO19RSB0V0	V_{CC}
184	IO18RSB0V0	IO10PPB0V1
185	IO17RSB0V0	IO09PPB0V1
186	IO16RSB0V0	IO10NPB0V1
187	IO15RSB0V0	IO09NPB0V1
188	V_{CCI}B0	IO08PPB0V1
189	GND	IO07PPB0V1
190	V_{CC}	IO08NPB0V1
191	IO14RSB0V0	IO07NPB0V1
192	IO13RSB0V0	IO06PPB0V0
193	IO12RSB0V0	IO05PPB0V0
194	IO11RSB0V0	IO06NPB0V0
195	IO10RSB0V0	IO04PPB0V0
196	IO09RSB0V0	IO05NPB0V0
197	IO08RSB0V0	IO04NPB0V0
198	IO07RSB0V0	GAC1/IO03PDB0V0
199	IO06RSB0V0	GAC0/IO03NDB0V0
200	GAC1/IO05RSB0V0	V_{CCI}B0
201	V_{CCI}B0	GND
202	GND	V_{CC}
203	V_{CC}	GAB1/IO02PDB0V0
204	GAC0/IO04RSB0V0	GAB0/IO02NDB0V0
205	GAB1/IO03RSB0V0	GAA1/IO01PDB0V0
206	GAB0/IO02RSB0V0	GAA0/IO01NDB0V0
207	GAA1/IO01RSB0V0	GNDQ
208	GAA0/IO00RSB0V0	V_{CCI}B0

256-Pin FBGA



Note

For Package Manufacturing and Environmental information, visit the Resource Center at http://www.actel.com/products/solutions/package/default.aspx.

256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
A1	GND	GND	GND	GND
A2	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B0$
A3	GAB0/IO02RSB0V0	GAA0/IO00RSB0V0	GAA0/IO01NDB0V0	GAA0/IO01NDB0V0
A4	GAB1/IO03RSB0V0	GAA1/IO01RSB0V0	GAA1/IO01PDB0V0	GAA1/IO01PDB0V0
A5	GND	GND	GND	GND
A6	IO07RSB0V0	IO11RSB0V0	IO10PDB0V1	IO07PDB0V1
A7	IO10RSB0V0	IO14RSB0V0	IO12PDB0V1	IO13PDB0V2
A8	IO11RSB0V0	IO15RSB0V0	IO12NDB0V1	IO13NDB0V2
A9	IO16RSB0V0	IO24RSB0V0	IO22NDB1V0	IO24NDB1V0
A10	IO17RSB0V0	IO25RSB0V0	IO22PDB1V0	IO24PDB1V0
A11	IO18RSB0V0	IO26RSB0V0	IO24NDB1V1	IO29NDB1V1
A12	GND	GND	GND	GND
A13	GBC0/IO25RSB0V0	GBA0/IO38RSB0V0	GBA0/IO28NDB1V1	GBA0/IO42NDB1V2
A14	GBA0/IO29RSB0V0	IO32RSB0V0	IO29NDB1V1	IO43NDB1V2
A15	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B1$	$V_{CCI}B1$
A16	GND	GND	GND	GND
B1	V_{COMPLA}	V_{COMPLA}	V_{COMPLA}	V_{COMPLA}
B2	V_{CCPLA}	V_{CCPLA}	V_{CCPLA}	V_{CCPLA}
B3	GAA0/IO00RSB0V0	IO07RSB0V0	IO00NDB0V0	IO00NDB0V0
B4	GAA1/IO01RSB0V0	IO06RSB0V0	IO00PDB0V0	IO00PDB0V0
B5	NC	GAB1/IO03RSB0V0	GAB1/IO02PPB0V0	GAB1/IO02PPB0V0
B6	IO06RSB0V0	IO10RSB0V0	IO10NDB0V1	IO07NDB0V1
B7	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B0$
B8	IO12RSB0V0	IO16RSB0V0	IO18NDB1V0	IO22NDB1V0
B9	IO13RSB0V0	IO17RSB0V0	IO18PDB1V0	IO22PDB1V0
B10	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B1$	$V_{CCI}B1$
B11	IO19RSB0V0	IO27RSB0V0	IO24PDB1V1	IO29PDB1V1
B12	GBB0/IO27RSB0V0	GBC0/IO34RSB0V0	GBC0/IO26NPB1V1	GBC0/IO40NPB1V2
B13	GBC1/IO26RSB0V0	GBA1/IO39RSB0V0	GBA1/IO28PDB1V1	GBA1/IO42PDB1V2
B14	GBA1/IO30RSB0V0	IO33RSB0V0	IO29PDB1V1	IO43PDB1V2
B15	NC	NC	V_{CCPLB}	V_{CCPLB}
B16	NC	NC	V_{COMPLB}	V_{COMPLB}
C1	$V_{CCI}B3$	$V_{CCI}B3$	$V_{CCI}B4$	$V_{CCI}B4$
C2	GND	GND	GND	GND
C3	$V_{CCI}B3$	$V_{CCI}B3$	$V_{CCI}B4$	$V_{CCI}B4$
C4	NC	NC	$V_{CCI}B0$	$V_{CCI}B0$
C5	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B0$	$V_{CCI}B0$
C6	GAC1/IO05RSB0V0	GAC1/IO05RSB0V0	GAC1/IO03PDB0V0	GAC1/IO03PDB0V0

256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
C7	IO09RSB0V0	IO12RSB0V0	IO06NDB0V0	IO09NDB0V1
C8	IO14RSB0V0	IO22RSB0V0	IO16PDB1V0	IO23PDB1V0
C9	IO15RSB0V0	IO23RSB0V0	IO16NDB1V0	IO23NDB1V0
C10	IO22RSB0V0	IO30RSB0V0	IO25NDB1V1	IO31NDB1V1
C11	IO20RSB0V0	IO31RSB0V0	IO25PDB1V1	IO31PDB1V1
C12	V_{CCI}B0	V_{CCI}B0	V_{CCI}B1	V_{CCI}B1
C13	GBB1/IO28RSB0V0	GBC1/IO35RSB0V0	GBC1/IO26PPB1V1	GBC1/IO40PPB1V2
C14	V_{CCI}B1	V_{CCI}B1	V_{CCI}B2	V_{CCI}B2
C15	GND	GND	GND	GND
C16	V_{CCI}B1	V_{CCI}B1	V_{CCI}B2	V_{CCI}B2
D1	GFC2/IO50NPB3V0	IO75NDB3V0	IO84NDB4V0	IO124NDB4V0
D2	GFA2/IO51NDB3V0	GAB2/IO75PDB3V0	GAB2/IO84PDB4V0	GAB2/IO124PDB4V0
D3	GAC2/IO51PDB3V0	IO76NDB3V0	IO85NDB4V0	IO125NDB4V0
D4	GAA2/IO52PDB3V0	GAA2/IO76PDB3V0	GAA2/IO85PDB4V0	GAA2/IO125PDB4V0
D5	GAB2/IO52NDB3V0	GAB0/IO02RSB0V0	GAB0/IO02NPB0V0	GAB0/IO02NPB0V0
D6	GAC0/IO04RSB0V0	GAC0/IO04RSB0V0	GAC0/IO03NDB0V0	GAC0/IO03NDB0V0
D7	IO08RSB0V0	IO13RSB0V0	IO06PDB0V0	IO09PDB0V1
D8	NC	IO20RSB0V0	IO14NDB0V1	IO15NDB0V2
D9	NC	IO21RSB0V0	IO14PDB0V1	IO15PDB0V2
D10	IO21RSB0V0	IO28RSB0V0	IO23PDB1V1	IO37PDB1V2
D11	IO23RSB0V0	GBB0/IO36RSB0V0	GBB0/IO27NDB1V1	GBB0/IO41NDB1V2
D12	NC	NC	V_{CCI}B1	V_{CCI}B1
D13	GBA2/IO31PDB1V0	GBA2/IO40PDB1V0	GBA2/IO30PDB2V0	GBA2/IO44PDB2V0
D14	GBB2/IO31NDB1V0	IO40NDB1V0	IO30NDB2V0	IO44NDB2V0
D15	GBC2/IO32PDB1V0	GBB2/IO41PDB1V0	GBB2/IO31PDB2V0	GBB2/IO45PDB2V0
D16	GCA2/IO32NDB1V0	IO41NDB1V0	IO31NDB2V0	IO45NDB2V0
E1	GND	GND	GND	GND
E2	GFB0/IO48NPB3V0	IO73NDB3V0	IO81NDB4V0	IO118NDB4V0
E3	GFB2/IO50PPB3V0	IO73PDB3V0	IO81PDB4V0	IO118PDB4V0
E4	V_{CCI}B3	V_{CCI}B3	V_{CCI}B4	V_{CCI}B4
E5	NC	IO74NPB3V0	IO83NPB4V0	IO123NPB4V0
E6	NC	IO08RSB0V0	IO04NPB0V0	IO05NPB0V1
E7	GND	GND	GND	GND
E8	NC	IO18RSB0V0	IO08PDB0V1	IO11PDB0V1
E9	NC	NC	IO20NDB1V0	IO27NDB1V1
E10	GND	GND	GND	GND
E11	IO24RSB0V0	GBB1/IO37RSB0V0	GBB1/IO27PDB1V1	GBB1/IO41PDB1V2
E12	NC	IO50PPB1V0	IO33PSB2V0	IO48PSB2V0

256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
E13	V_{CCI}B1	V_{CCI}B1	V_{CCI}B2	V_{CCI}B2
E14	GCC2/IO33NDB1V0	IO42NDB1V0	IO32NDB2V0	IO46NDB2V0
E15	GCB2/IO33PDB1V0	GBC2/IO42PDB1V0	GBC2/IO32PDB2V0	GBC2/IO46PDB2V0
E16	GND	GND	GND	GND
F1	NC	NC	IO79NDB4V0	IO111NDB4V0
F2	NC	NC	IO79PDB4V0	IO111PDB4V0
F3	GFB1/IO48PPB3V0	IO72NDB3V0	IO76NDB4V0	IO112NDB4V0
F4	GFC0/IO49NDB3V0	IO72PDB3V0	IO76PDB4V0	IO112PDB4V0
F5	NC	NC	IO82PSB4V0	IO120PSB4V0
F6	GFC1/IO49PDB3V0	GAC2/IO74PPB3V0	GAC2/IO83PPB4V0	GAC2/IO123PPB4V0
F7	NC	IO09RSB0V0	IO04PPB0V0	IO05PPB0V1
F8	NC	IO19RSB0V0	IO08NDB0V1	IO11NDB0V1
F9	NC	NC	IO20PDB1V0	IO27PDB1V1
F10	NC	IO29RSB0V0	IO23NDB1V1	IO37NDB1V2
F11	NC	IO43NDB1V0	IO36NDB2V0	IO50NDB2V0
F12	NC	IO43PDB1V0	IO36PDB2V0	IO50PDB2V0
F13	NC	IO44NDB1V0	IO39NDB2V0	IO59NDB2V0
F14	NC	GCA2/IO44PDB1V0	GCA2/IO39PDB2V0	GCA2/IO59PDB2V0
F15	GCC1/IO34PDB1V0	GCB2/IO45PDB1V0	GCB2/IO40PDB2V0	GCB2/IO60PDB2V0
F16	GCC0/IO34NDB1V0	IO45NDB1V0	IO40NDB2V0	IO60NDB2V0
G1	GEC0/IO46NPB3V0	IO70NPB3V0	IO74NPB4V0	IO109NPB4V0
G2	V_{CCI}B3	V_{CCI}B3	V_{CCI}B4	V_{CCI}B4
G3	GEC1/IO46PPB3V0	GFB2/IO70PPB3V0	GFB2/IO74PPB4V0	GFB2/IO109PPB4V0
G4	GFA1/IO47PDB3V0	GFA2/IO71PDB3V0	GFA2/IO75PDB4V0	GFA2/IO110PDB4V0
G5	GND	GND	GND	GND
G6	GFA0/IO47NDB3V0	IO71NDB3V0	IO75NDB4V0	IO110NDB4V0
G7	GND	GND	GND	GND
G8	V_{CC}	V_{CC}	V_{CC}	V_{CC}
G9	GND	GND	GND	GND
G10	V_{CC}	V_{CC}	V_{CC}	V_{CC}
G11	GDA1/IO37NDB1V0	GCC0/IO47NDB1V0	GCC0/IO43NDB2V0	GCC0/IO62NDB2V0
G12	GND	GND	GND	GND
G13	IO37PDB1V0	GCC1/IO47PDB1V0	GCC1/IO43PDB2V0	GCC1/IO62PDB2V0
G14	GCB0/IO35NPB1V0	IO46NPB1V0	IO41NPB2V0	IO61NPB2V0
G15	V_{CCI}B1	V_{CCI}B1	V_{CCI}B2	V_{CCI}B2
G16	GCB1/IO35PPB1V0	GCC2/IO46PPB1V0	GCC2/IO41PPB2V0	GCC2/IO61PPB2V0
H1	GEB1/IO45PDB3V0	GFC2/IO69PDB3V0	GFC2/IO73PDB4V0	GFC2/IO108PDB4V0
H2	GEB0/IO45NDB3V0	IO69NDB3V0	IO73NDB4V0	IO108NDB4V0

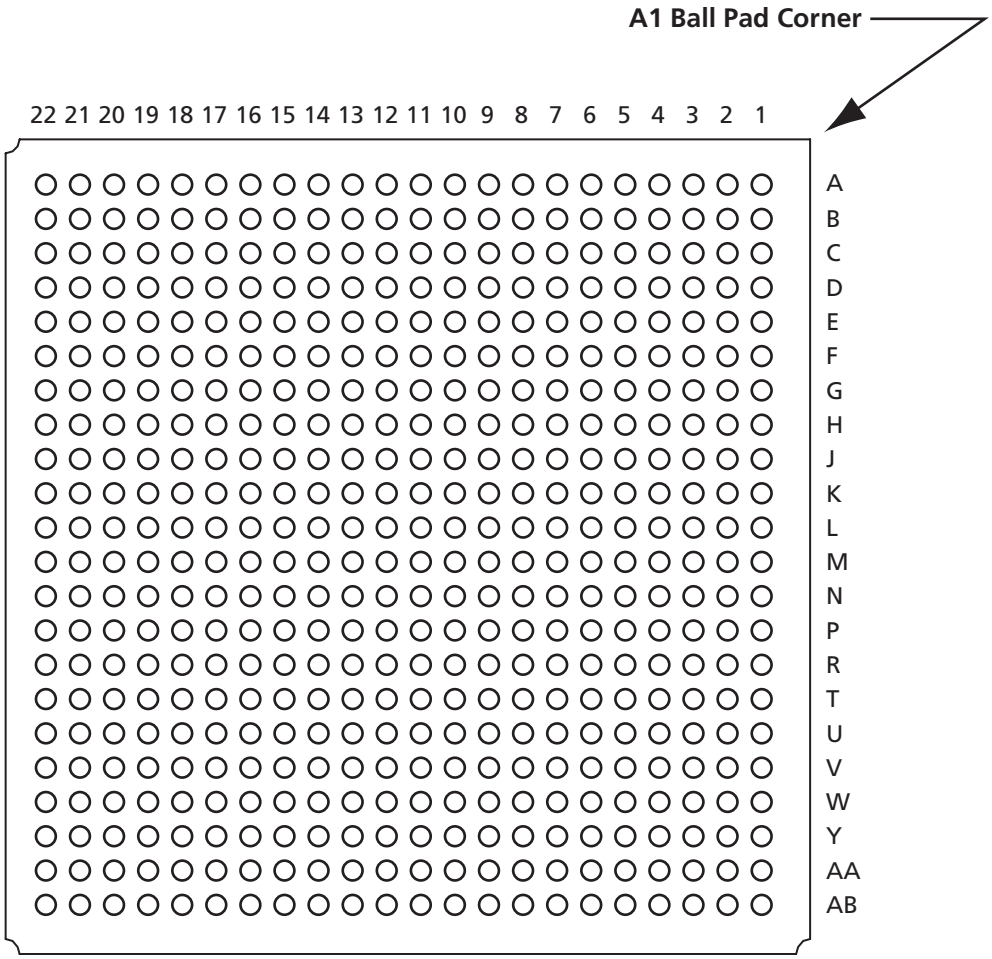
256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
H3	XTAL2	XTAL2	XTAL2	XTAL2
H4	XTAL1	XTAL1	XTAL1	XTAL1
H5	GNDOSC	GNDOSC	GNDOSC	GNDOSC
H6	V_{CCOSC}	V_{CCOSC}	V_{CCOSC}	V_{CCOSC}
H7	V_{CC}	V_{CC}	V_{CC}	V_{CC}
H8	GND	GND	GND	GND
H9	V_{CC}	V_{CC}	V_{CC}	V_{CC}
H10	GND	GND	GND	GND
H11	GDC0/IO38NDB1V0	IO51NDB1V0	IO47NDB2V0	IO69NDB2V0
H12	GDC1/IO38PDB1V0	IO51PDB1V0	IO47PDB2V0	IO69PDB2V0
H13	GDB1/IO39PDB1V0	GCA1/IO49PDB1V0	GCA1/IO45PDB2V0	GCA1/IO64PDB2V0
H14	GDB0/IO39NDB1V0	GCA0/IO49NDB1V0	GCA0/IO45NDB2V0	GCA0/IO64NDB2V0
H15	GCA0/IO36NDB1V0	GCB0/IO48NDB1V0	GCB0/IO44NDB2V0	GCB0/IO63NDB2V0
H16	GCA1/IO36PDB1V0	GCB1/IO48PDB1V0	GCB1/IO44PDB2V0	GCB1/IO63PDB2V0
J1	GEA0/IO44NDB3V0	GFA0/IO66NDB3V0	GFA0/IO70NDB4V0	GFA0/IO105NDB4V0
J2	GEA1/IO44PDB3V0	GFA1/IO66PDB3V0	GFA1/IO70PDB4V0	GFA1/IO105PDB4V0
J3	IO43NDB3V0	GFB0/IO67NDB3V0	GFB0/IO71NDB4V0	GFB0/IO106NDB4V0
J4	GEC2/IO43PDB3V0	GFB1/IO67PDB3V0	GFB1/IO71PDB4V0	GFB1/IO106PDB4V0
J5	NC	GFC0/IO68NDB3V0	GFC0/IO72NDB4V0	GFC0/IO107NDB4V0
J6	NC	GFC1/IO68PDB3V0	GFC1/IO72PDB4V0	GFC1/IO107PDB4V0
J7	GND	GND	GND	GND
J8	V_{CC}	V_{CC}	V_{CC}	V_{CC}
J9	GND	GND	GND	GND
J10	V_{CC}	V_{CC}	V_{CC}	V_{CC}
J11	GDC2/IO41NPB1V0	IO56NPB1V0	IO56NPB2V0	IO83NPB2V0
J12	NC	GDB0/IO53NPB1V0	GDB0/IO53NPB2V0	GDB0/IO80NPB2V0
J13	NC	GDA1/IO54PDB1V0	GDA1/IO54PDB2V0	GDA1/IO81PDB2V0
J14	GDA0/IO40PDB1V0	GDC1/IO52PPB1V0	GDC1/IO52PPB2V0	GDC1/IO79PPB2V0
J15	NC	IO50NPB1V0	IO51NSB2V0	IO77NSB2V0
J16	GDA2/IO40NDB1V0	GDC0/IO52NPB1V0	GDC0/IO52NPB2V0	GDC0/IO79NPB2V0
K1	NC	IO65NPB3V0	IO67NPB4V0	IO92NPB4V0
K2	$V_{CCI}B3$	$V_{CCI}B3$	$V_{CCI}B4$	$V_{CCI}B4$
K3	NC	IO65PPB3V0	IO67PPB4V0	IO92PPB4V0
K4	NC	IO64PDB3V0	IO65PDB4V0	IO96PDB4V0
K5	GND	GND	GND	GND
K6	NC	IO64NDB3V0	IO65NDB4V0	IO96NDB4V0
K7	V_{CC}	V_{CC}	V_{CC}	V_{CC}
K8	GND	GND	GND	GND

256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
K9	V_{CC}	V_{CC}	V_{CC}	V_{CC}
K10	GND	GND	GND	GND
K11	NC	GDC2/IO57PPB1V0	GDC2/IO57PPB2V0	GDC2/IO84PPB2V0
K12	GND	GND	GND	GND
K13	NC	GDA0/IO54NDB1V0	GDA0/IO54NDB2V0	GDA0/IO81NDB2V0
K14	NC	GDA2/IO55PPB1V0	GDA2/IO55PPB2V0	GDA2/IO82PPB2V0
K15	V_{CCI}B1	V_{CCI}B1	V_{CCI}B2	V_{CCI}B2
K16	NC	GDB1/IO53PPB1V0	GDB1/IO53PPB2V0	GDB1/IO80PPB2V0
L1	NC	GEC1/IO63PDB3V0	GEC1/IO63PDB4V0	GEC1/IO90PDB4V0
L2	NC	GEC0/IO63NDB3V0	GEC0/IO63NDB4V0	GEC0/IO90NDB4V0
L3	NC	GEB1/IO62PDB3V0	GEB1/IO62PDB4V0	GEB1/IO89PDB4V0
L4	NC	GEB0/IO62NDB3V0	GEB0/IO62NDB4V0	GEB0/IO89NDB4V0
L5	NC	IO60NDB3V0	IO60NDB4V0	IO87NDB4V0
L6	NC	GEC2/IO60PDB3V0	GEC2/IO60PDB4V0	GEC2/IO87PDB4V0
L7	GNDA	GNDA	GNDA	GNDA
L8	AC0	AC0	AC2	AC2
L9	AV2	AV2	AV4	AV4
L10	AC3	AC3	AC5	AC5
L11	PTEM	PTEM	PTEM	PTEM
L12	TDO	TDO	TDO	TDO
L13	V_{JTAG}	V_{JTAG}	V_{JTAG}	V_{JTAG}
L14	NC	IO57NPB1V0	IO57NPB2V0	IO84NPB2V0
L15	GDB2/IO41PPB1V0	GDB2/IO56PPB1V0	GDB2/IO56PPB2V0	GDB2/IO83PPB2V0
L16	NC	IO55NPB1V0	IO55NPB2V0	IO82NPB2V0
M1	GND	GND	GND	GND
M2	NC	GEA1/IO61PDB3V0	GEA1/IO61PDB4V0	GEA1/IO88PDB4V0
M3	NC	GEA0/IO61NDB3V0	GEA0/IO61NDB4V0	GEA0/IO88NDB4V0
M4	V_{CCI}B3	V_{CCI}B3	V_{CCI}B4	V_{CCI}B4
M5	NC	IO58NPB3V0	IO58NPB4V0	IO85NPB4V0
M6	NC	NC	AV0	AV0
M7	NC	NC	AC1	AC1
M8	AG1	AG1	AG3	AG3
M9	AC2	AC2	AC4	AC4
M10	AC4	AC4	AC6	AC6
M11	NC	AG5	AG7	AG7
M12	V_{PUMP}	V_{PUMP}	V_{PUMP}	V_{PUMP}
M13	V_{CCI}B1	V_{CCI}B1	V_{CCI}B2	V_{CCI}B2
M14	TMS	TMS	TMS	TMS

256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
M15	TRST	TRST	TRST	TRST
M16	GND	GND	GND	GND
N1	GEB2/IO42PDB3V0	GEB2/IO59PDB3V0	GEB2/IO59PDB4V0	GEB2/IO86PDB4V0
N2	GEA2/IO42NDB3V0	IO59NDB3V0	IO59NDB4V0	IO86NDB4V0
N3	NC	GEA2/IO58PPB3V0	GEA2/IO58PPB4V0	GEA2/IO85PPB4V0
N4	$V_{CC33PMP}$	$V_{CC33PMP}$	$V_{CC33PMP}$	$V_{CC33PMP}$
N5	V_{CC15A}	V_{CC15A}	V_{CC15A}	V_{CC15A}
N6	NC	NC	AG0	AG0
N7	AC1	AC1	AC3	AC3
N8	AG3	AG3	AG5	AG5
N9	AV3	AV3	AV5	AV5
N10	AG4	AG4	AG6	AG6
N11	NC	NC	AC8	AC8
N12	GNDA	GNDA	GNDA	GNDA
N13	V_{CC33A}	V_{CC33A}	V_{CC33A}	V_{CC33A}
N14	V_{CCNVM}	V_{CCNVM}	V_{CCNVM}	V_{CCNVM}
N15	TCK	TCK	TCK	TCK
N16	TDI	TDI	TDI	TDI
P1	V_{CCNVM}	V_{CCNVM}	V_{CCNVM}	V_{CCNVM}
P2	GNDNVM	GNDNVM	GNDNVM	GNDNVM
P3	GNDA	GNDA	GNDA	GNDA
P4	NC	NC	AC0	AC0
P5	NC	NC	AG1	AG1
P6	NC	NC	AV1	AV1
P7	AG0	AG0	AG2	AG2
P8	AG2	AG2	AG4	AG4
P9	GNDA	GNDA	GNDA	GNDA
P10	NC	AC5	AC7	AC7
P11	NC	NC	AV8	AV8
P12	NC	NC	AG8	AG8
P13	NC	NC	AV9	AV9
P14	ADCGNDREF	ADCGNDREF	ADCGNDREF	ADCGNDREF
P15	PTBASE	PTBASE	PTBASE	PTBASE
P16	GNDNVM	GNDNVM	GNDNVM	GNDNVM
R1	V_{CCI}B3	V_{CCI}B3	V_{CCI}B4	V_{CCI}B4
R2	PCAP	PCAP	PCAP	PCAP
R3	NC	NC	AT1	AT1
R4	NC	NC	AT0	AT0

256-Pin FBGA				
Pin Number	**AFS090 Function**	**AFS250 Function**	**AFS600 Function**	**AFS1500 Function**
R5	AV0	AV0	AV2	AV2
R6	AT0	AT0	AT2	AT2
R7	AV1	AV1	AV3	AV3
R8	AT3	AT3	AT5	AT5
R9	AV4	AV4	AV6	AV6
R10	NC	AT5	AT7	AT7
R11	NC	AV5	AV7	AV7
R12	NC	NC	AT9	AT9
R13	NC	NC	AG9	AG9
R14	NC	NC	AC9	AC9
R15	PUB	PUB	PUB	PUB
R16	$V_{CCI}B1$	$V_{CCI}B1$	$V_{CCI}B2$	$V_{CCI}B2$
T1	GND	GND	GND	GND
T2	NCAP	NCAP	NCAP	NCAP
T3	VCC33N	VCC33N	VCC33N	VCC33N
T4	NC	NC	ATRTN0	ATRTN0
T5	AT1	AT1	AT3	AT3
T6	ATRTN0	ATRTN0	ATRTN1	ATRTN1
T7	AT2	AT2	AT4	AT4
T8	ATRTN1	ATRTN1	ATRTN2	ATRTN2
T9	AT4	AT4	AT6	AT6
T10	ATRTN2	ATRTN2	ATRTN3	ATRTN3
T11	NC	NC	AT8	AT8
T12	NC	NC	ATRTN4	ATRTN4
T13	GNDA	GNDA	GNDA	GNDA
T14	V_{CC33A}	V_{CC33A}	V_{CC33A}	V_{CC33A}
T15	VAREF	VAREF	VAREF	VAREF
T16	GND	GND	GND	GND

484-Pin FBGA



Note

For Package Manufacturing and Environmental information, visit the Resource Center at http://www.actel.com/products/solutions/package/default.aspx.

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
A1	GND	GND
A2	V_{CC}	NC
A3	GAA1/IO01PDB0V0	GAA1/IO01PDB0V0
A4	GAB0/IO02NDB0V0	GAB0/IO02NDB0V0
A5	GAB1/IO02PDB0V0	GAB1/IO02PDB0V0
A6	IO07NDB0V1	IO07NDB0V1
A7	IO07PDB0V1	IO07PDB0V1
A8	IO10PDB0V1	IO09PDB0V1
A9	IO14NDB0V1	IO13NDB0V2
A10	IO14PDB0V1	IO13PDB0V2
A11	IO17PDB1V0	IO24PDB1V0
A12	IO18PDB1V0	IO26PDB1V0
A13	IO19NDB1V0	IO27NDB1V1
A14	IO19PDB1V0	IO27PDB1V1
A15	IO24NDB1V1	IO35NDB1V2
A16	IO24PDB1V1	IO35PDB1V2
A17	GBC0/IO26NDB1V1	GBC0/IO40NDB1V2
A18	GBA0/IO28NDB1V1	GBA0/IO42NDB1V2
A19	IO29NDB1V1	IO43NDB1V2
A20	IO29PDB1V1	IO43PDB1V2
A21	V_{CC}	NC
A22	GND	GND
AA1	V_{CC}	NC
AA2	GND	GND
AA3	V_{CCI}B4	V_{CCI}B4
AA4	V_{CCI}B4	V_{CCI}B4
AA5	PCAP	PCAP
AA6	AG0	AG0
AA7	GNDA	GNDA
AA8	AG1	AG1
AA9	AG2	AG2
AA10	GNDA	GNDA
AA11	AG3	AG3
AA12	AG6	AG6
AA13	GNDA	GNDA

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
AA14	AG7	AG7
AA15	AG8	AG8
AA16	GNDA	GNDA
AA17	AG9	AG9
AA18	VAREF	VAREF
AA19	V_{CCI}B2	V_{CCI}B2
AA20	PTEM	PTEM
AA21	GND	GND
AA22	V_{CC}	NC
AB1	GND	GND
AB2	V_{CC}	NC
AB3	NC	IO94NSB4V0
AB4	GND	GND
AB5	VCC33N	VCC33N
AB6	AT0	AT0
AB7	ATRTN0	ATRTN0
AB8	AT1	AT1
AB9	AT2	AT2
AB10	ATRTN1	ATRTN1
AB11	AT3	AT3
AB12	AT6	AT6
AB13	ATRTN3	ATRTN3
AB14	AT7	AT7
AB15	AT8	AT8
AB16	ATRTN4	ATRTN4
AB17	AT9	AT9
AB18	V_{CC33A}	V_{CC33A}
AB19	GND	GND
AB20	NC	IO76NPB2V0
AB21	V_{CC}	NC
AB22	GND	GND
B1	V_{CC}	NC
B2	GND	GND
B3	GAA0/IO01NDB0V0	GAA0/IO01NDB0V0
B4	GND	GND

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
B5	IO05NDB0V0	IO04NDB0V0
B6	IO05PDB0V0	IO04PDB0V0
B7	GND	GND
B8	IO10NDB0V1	IO09NDB0V1
B9	IO13PDB0V1	IO11PDB0V1
B10	GND	GND
B11	IO17NDB1V0	IO24NDB1V0
B12	IO18NDB1V0	IO26NDB1V0
B13	GND	GND
B14	IO21NDB1V0	IO31NDB1V1
B15	IO21PDB1V0	IO31PDB1V1
B16	GND	GND
B17	GBC1/IO26PDB1V1	GBC1/IO40PDB1V2
B18	GBA1/IO28PDB1V1	GBA1/IO42PDB1V2
B19	GND	GND
B20	V_{CCPLB}	V_{CCPLB}
B21	GND	GND
B22	V_{CC}	NC
C1	IO82PDB4V0	IO121PDB4V0
C2	NC	IO122PSB4V0
C3	IO00NDB0V0	IO00NDB0V0
C4	IO00PDB0V0	IO00PDB0V0
C5	$V_{CCI}B0$	$V_{CCI}B0$
C6	IO06NDB0V0	IO05NDB0V1
C7	IO06PDB0V0	IO05PDB0V1
C8	$V_{CCI}B0$	$V_{CCI}B0$
C9	IO13NDB0V1	IO11NDB0V1
C10	IO11PDB0V1	IO14PDB0V2
C11	$V_{CCI}B0$	$V_{CCI}B0$
C12	$V_{CCI}B1$	$V_{CCI}B1$
C13	IO20NDB1V0	IO29NDB1V1
C14	IO20PDB1V0	IO29PDB1V1
C15	$V_{CCI}B1$	$V_{CCI}B1$
C16	IO25NDB1V1	IO37NDB1V2
C17	GBB0/IO27NDB1V1	GBB0/IO41NDB1V2

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
C18	$V_{CCI}B1$	$V_{CCI}B1$
C19	V_{COMPLB}	V_{COMPLB}
C20	GBA2/IO30PDB2V0	GBA2/IO44PDB2V0
C21	NC	IO48PSB2V0
C22	GBB2/IO31PDB2V0	GBB2/IO45PDB2V0
D1	IO82NDB4V0	IO121NDB4V0
D2	GND	GND
D3	IO83NDB4V0	IO123NDB4V0
D4	GAC2/IO83PDB4V0	GAC2/IO123PDB4V0
D5	GAA2/IO85PDB4V0	GAA2/IO125PDB4V0
D6	GAC0/IO03NDB0V0	GAC0/IO03NDB0V0
D7	GAC1/IO03PDB0V0	GAC1/IO03PDB0V0
D8	IO09NDB0V1	IO10NDB0V1
D9	IO09PDB0V1	IO10PDB0V1
D10	IO11NDB0V1	IO14NDB0V2
D11	IO16NDB1V0	IO23NDB1V0
D12	IO16PDB1V0	IO23PDB1V0
D13	NC	IO32NPB1V1
D14	IO23NDB1V1	IO34NDB1V1
D15	IO23PDB1V1	IO34PDB1V1
D16	IO25PDB1V1	IO37PDB1V2
D17	GBB1/IO27PDB1V1	GBB1/IO41PDB1V2
D18	$V_{CCI}B2$	$V_{CCI}B2$
D19	NC	IO47PPB2V0
D20	IO30NDB2V0	IO44NDB2V0
D21	GND	GND
D22	IO31NDB2V0	IO45NDB2V0
E1	IO81NDB4V0	IO120NDB4V0
E2	IO81PDB4V0	IO120PDB4V0
E3	$V_{CCI}B4$	$V_{CCI}B4$
E4	GAB2/IO84PDB4V0	GAB2/IO124PDB4V0
E5	IO85NDB4V0	IO125NDB4V0
E6	GND	GND
E7	$V_{CCI}B0$	$V_{CCI}B0$
E8	NC	IO08NDB0V1

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
E9	NC	IO08PDB0V1
E10	GND	GND
E11	IO15NDB1V0	IO22NDB1V0
E12	IO15PDB1V0	IO22PDB1V0
E13	GND	GND
E14	NC	IO32PPB1V1
E15	NC	IO36NPB1V2
E16	V_{CCI}B1	V_{CCI}B1
E17	GND	GND
E18	NC	IO47NPB2V0
E19	IO33PDB2V0	IO49PDB2V0
E20	V_{CCI}B2	V_{CCI}B2
E21	IO32NDB2V0	IO46NDB2V0
E22	GBC2/IO32PDB2V0	GBC2/IO46PDB2V0
F1	IO80NDB4V0	IO118NDB4V0
F2	IO80PDB4V0	IO118PDB4V0
F3	NC	IO119NSB4V0
F4	IO84NDB4V0	IO124NDB4V0
F5	GND	GND
F6	V_{COMPLA}	V_{COMPLA}
F7	V_{CCPLA}	V_{CCPLA}
F8	V_{CCI}B0	V_{CCI}B0
F9	IO08NDB0V1	IO12NDB0V1
F10	IO08PDB0V1	IO12PDB0V1
F11	V_{CCI}B0	V_{CCI}B0
F12	V_{CCI}B1	V_{CCI}B1
F13	IO22NDB1V0	IO30NDB1V1
F14	IO22PDB1V0	IO30PDB1V1
F15	V_{CCI}B1	V_{CCI}B1
F16	NC	IO36PPB1V2
F17	NC	IO38NPB1V2
F18	GND	GND
F19	IO33NDB2V0	IO49NDB2V0
F20	IO34PDB2V0	IO50PDB2V0
F21	IO34NDB2V0	IO50NDB2V0

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
F22	IO35PDB2V0	IO51PDB2V0
G1	IO77PDB4V0	IO115PDB4V0
G2	GND	GND
G3	IO78NDB4V0	IO116NDB4V0
G4	IO78PDB4V0	IO116PDB4V0
G5	V_{CCI}B4	V_{CCI}B4
G6	NC	IO117PDB4V0
G7	V_{CCI}B4	V_{CCI}B4
G8	GND	GND
G9	IO04NDB0V0	IO06NDB0V1
G10	IO04PDB0V0	IO06PDB0V1
G11	IO12NDB0V1	IO16NDB0V2
G12	IO12PDB0V1	IO16PDB0V2
G13	NC	IO28NDB1V1
G14	NC	IO28PDB1V1
G15	GND	GND
G16	NC	IO38PPB1V2
G17	NC	IO53PDB2V0
G18	V_{CCI}B2	V_{CCI}B2
G19	IO36PDB2V0	IO52PDB2V0
G20	IO36NDB2V0	IO52NDB2V0
G21	GND	GND
G22	IO35NDB2V0	IO51NDB2V0
H1	IO77NDB4V0	IO115NDB4V0
H2	IO76PDB4V0	IO113PDB4V0
H3	V_{CCI}B4	V_{CCI}B4
H4	IO79NDB4V0	IO114NDB4V0
H5	IO79PDB4V0	IO114PDB4V0
H6	NC	IO117NDB4V0
H7	GND	GND
H8	V_{CC}	V_{CC}
H9	V_{CCI}B0	V_{CCI}B0
H10	GND	GND
H11	V_{CCI}B0	V_{CCI}B0
H12	V_{CCI}B1	V_{CCI}B1

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
H13	GND	GND
H14	V_{CCI}B1	V_{CCI}B1
H15	GND	GND
H16	GND	GND
H17	NC	IO53NDB2V0
H18	IO38PDB2V0	IO57PDB2V0
H19	GCA2/IO39PDB2V0	GCA2/IO59PDB2V0
H20	V_{CCI}B2	V_{CCI}B2
H21	IO37NDB2V0	IO54NDB2V0
H22	IO37PDB2V0	IO54PDB2V0
J1	NC	IO112PPB4V0
J2	IO76NDB4V0	IO113NDB4V0
J3	GFB2/IO74PDB4V0	GFB2/IO109PDB4V0
J4	GFA2/IO75PDB4V0	GFA2/IO110PDB4V0
J5	NC	IO112NPB4V0
J6	NC	IO104PDB4V0
J7	NC	IO111PDB4V0
J8	V_{CCI}B4	V_{CCI}B4
J9	GND	GND
J10	V_{CC}	V_{CC}
J11	GND	GND
J12	V_{CC}	V_{CC}
J13	GND	GND
J14	V_{CC}	V_{CC}
J15	V_{CCI}B2	V_{CCI}B2
J16	GCB2/IO40PDB2V0	GCB2/IO60PDB2V0
J17	NC	IO58NDB2V0
J18	IO38NDB2V0	IO57NDB2V0
J19	IO39NDB2V0	IO59NDB2V0
J20	GCC2/IO41PDB2V0	GCC2/IO61PDB2V0
J21	NC	IO55PSB2V0
J22	IO42PDB2V0	IO56PDB2V0
K1	GFC2/IO73PDB4V0	GFC2/IO108PDB4V0
K2	GND	GND
K3	IO74NDB4V0	IO109NDB4V0

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
K4	IO75NDB4V0	IO110NDB4V0
K5	GND	GND
K6	NC	IO104NDB4V0
K7	NC	IO111NDB4V0
K8	GND	GND
K9	V_{CC}	V_{CC}
K10	GND	GND
K11	V_{CC}	V_{CC}
K12	GND	GND
K13	V_{CC}	V_{CC}
K14	GND	GND
K15	GND	GND
K16	IO40NDB2V0	IO60NDB2V0
K17	NC	IO58PDB2V0
K18	GND	GND
K19	NC	IO68NPB2V0
K20	IO41NDB2V0	IO61NDB2V0
K21	GND	GND
K22	IO42NDB2V0	IO56NDB2V0
L1	IO73NDB4V0	IO108NDB4V0
L2	V_{CCOSC}	V_{CCOSC}
L3	V_{CCI}B4	V_{CCI}B4
L4	XTAL2	XTAL2
L5	GFC1/IO72PDB4V0	GFC1/IO107PDB4V0
L6	V_{CCI}B4	V_{CCI}B4
L7	GFB1/IO71PDB4V0	GFB1/IO106PDB4V0
L8	V_{CCI}B4	V_{CCI}B4
L9	GND	GND
L10	V_{CC}	V_{CC}
L11	GND	GND
L12	V_{CC}	V_{CC}
L13	GND	GND
L14	V_{CC}	V_{CC}
L15	V_{CCI}B2	V_{CCI}B2
L16	IO48PDB2V0	IO70PDB2V0

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
L17	V_{CCI}B2	V_{CCI}B2
L18	IO46PDB2V0	IO69PDB2V0
L19	GCA1/IO45PDB2V0	GCA1/IO64PDB2V0
L20	V_{CCI}B2	V_{CCI}B2
L21	GCC0/IO43NDB2V0	GCC0/IO62NDB2V0
L22	GCC1/IO43PDB2V0	GCC1/IO62PDB2V0
M1	NC	IO103PDB4V0
M2	XTAL1	XTAL1
M3	V_{CCI}B4	V_{CCI}B4
M4	GNDOSC	GNDOSC
M5	GFC0/IO72NDB4V0	GFC0/IO107NDB4V0
M6	V_{CCI}B4	V_{CCI}B4
M7	GFB0/IO71NDB4V0	GFB0/IO106NDB4V0
M8	V_{CCI}B4	V_{CCI}B4
M9	V_{CC}	V_{CC}
M10	GND	GND
M11	V_{CC}	V_{CC}
M12	GND	GND
M13	V_{CC}	V_{CC}
M14	GND	GND
M15	V_{CCI}B2	V_{CCI}B2
M16	IO48NDB2V0	IO70NDB2V0
M17	V_{CCI}B2	V_{CCI}B2
M18	IO46NDB2V0	IO69NDB2V0
M19	GCA0/IO45NDB2V0	GCA0/IO64NDB2V0
M20	V_{CCI}B2	V_{CCI}B2
M21	GCB0/IO44NDB2V0	GCB0/IO63NDB2V0
M22	GCB1/IO44PDB2V0	GCB1/IO63PDB2V0
N1	NC	IO103NDB4V0
N2	GND	GND
N3	IO68PDB4V0	IO101PDB4V0
N4	NC	IO100NPB4V0
N5	GND	GND
N6	NC	IO99PDB4V0
N7	NC	IO97PDB4V0

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
N8	GND	GND
N9	GND	GND
N10	V_{CC}	V_{CC}
N11	GND	GND
N12	V_{CC}	V_{CC}
N13	GND	GND
N14	V_{CC}	V_{CC}
N15	GND	GND
N16	GDB2/IO56PDB2V0	GDB2/IO83PDB2V0
N17	NC	IO78PDB2V0
N18	GND	GND
N19	IO47NDB2V0	IO72NDB2V0
N20	IO47PDB2V0	IO72PDB2V0
N21	GND	GND
N22	IO49PDB2V0	IO71PDB2V0
P1	GFA1/IO70PDB4V0	GFA1/IO105PDB4V0
P2	GFA0/IO70NDB4V0	GFA0/IO105NDB4V0
P3	IO68NDB4V0	IO101NDB4V0
P4	IO65PDB4V0	IO96PDB4V0
P5	IO65NDB4V0	IO96NDB4V0
P6	NC	IO99NDB4V0
P7	NC	IO97NDB4V0
P8	V_{CCI}B4	V_{CCI}B4
P9	V_{CC}	V_{CC}
P10	GND	GND
P11	V_{CC}	V_{CC}
P12	GND	GND
P13	V_{CC}	V_{CC}
P14	GND	GND
P15	V_{CCI}B2	V_{CCI}B2
P16	IO56NDB2V0	IO83NDB2V0
P17	NC	IO78NDB2V0
P18	GDA1/IO54PDB2V0	GDA1/IO81PDB2V0
P19	GDB1/IO53PDB2V0	GDB1/IO80PDB2V0
P20	IO51NDB2V0	IO73NDB2V0

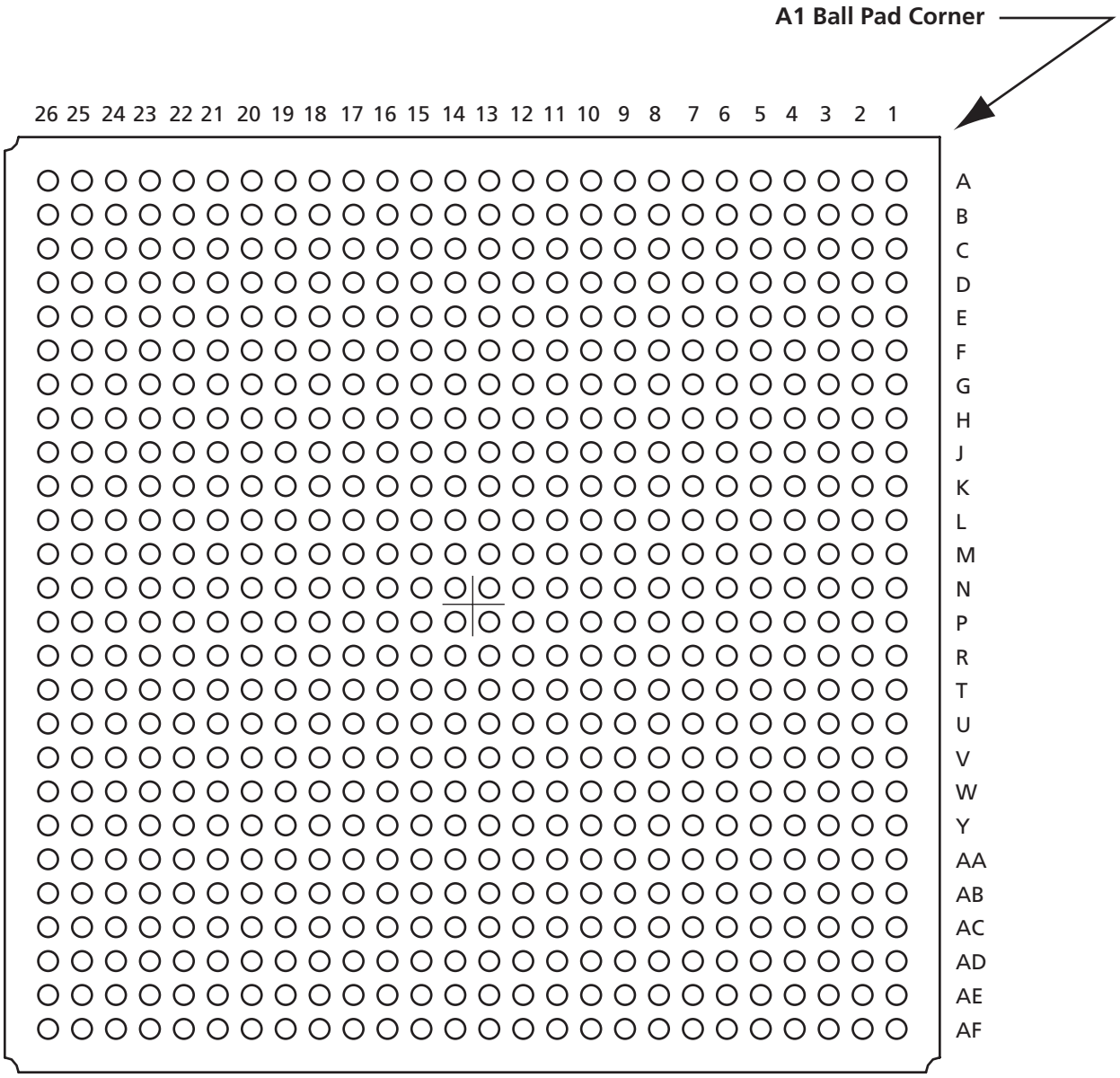
484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
P21	IO51PDB2V0	IO73PDB2V0
P22	IO49NDB2V0	IO71NDB2V0
R1	IO69PDB4V0	IO102PDB4V0
R2	IO69NDB4V0	IO102NDB4V0
R3	V_{CCI}B4	V_{CCI}B4
R4	IO64PDB4V0	IO91PDB4V0
R5	IO64NDB4V0	IO91NDB4V0
R6	NC	IO92PDB4V0
R7	GND	GND
R8	GND	GND
R9	V_{CC33A}	V_{CC33A}
R10	GNDA	GNDA
R11	V_{CC33A}	V_{CC33A}
R12	GNDA	GNDA
R13	V_{CC33A}	V_{CC33A}
R14	GNDA	GNDA
R15	V_{CC}	V_{CC}
R16	GND	GND
R17	NC	IO74NDB2V0
R18	GDA0/IO54NDB2V0	GDA0/IO81NDB2V0
R19	GDB0/IO53NDB2V0	GDB0/IO80NDB2V0
R20	V_{CCI}B2	V_{CCI}B2
R21	IO50NDB2V0	IO75NDB2V0
R22	IO50PDB2V0	IO75PDB2V0
T1	NC	IO100PPB4V0
T2	GND	GND
T3	IO66PDB4V0	IO95PDB4V0
T4	IO66NDB4V0	IO95NDB4V0
T5	V_{CCI}B4	V_{CCI}B4
T6	NC	IO92NDB4V0
T7	GNDNVM	GNDNVM
T8	GNDA	GNDA
T9	NC	NC
T10	AV4	AV4
T11	NC	NC

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
T12	AV5	AV5
T13	AC5	AC5
T14	NC	NC
T15	GNDA	GNDA
T16	NC	IO77PPB2V0
T17	NC	IO74PDB2V0
T18	V_{CCI}B2	V_{CCI}B2
T19	IO55NDB2V0	IO82NDB2V0
T20	GDA2/IO55PDB2V0	GDA2/IO82PDB2V0
T21	GND	GND
T22	GDC1/IO52PDB2V0	GDC1/IO79PDB2V0
U1	IO67PDB4V0	IO98PDB4V0
U2	IO67NDB4V0	IO98NDB4V0
U3	GEC1/IO63PDB4V0	GEC1/IO90PDB4V0
U4	GEC0/IO63NDB4V0	GEC0/IO90NDB4V0
U5	GND	GND
U6	V_{CCNVM}	V_{CCNVM}
U7	V_{CCI}B4	V_{CCI}B4
U8	V_{CC15A}	V_{CC15A}
U9	GNDA	GNDA
U10	AC4	AC4
U11	V_{CC33A}	V_{CC33A}
U12	GNDA	GNDA
U13	AG5	AG5
U14	GNDA	GNDA
U15	PUB	PUB
U16	V_{CCI}B2	V_{CCI}B2
U17	TDI	TDI
U18	GND	GND
U19	IO57NDB2V0	IO84NDB2V0
U20	GDC2/IO57PDB2V0	GDC2/IO84PDB2V0
U21	NC	IO77NPB2V0
U22	GDC0/IO52NDB2V0	GDC0/IO79NDB2V0
V1	GEB1/IO62PDB4V0	GEB1/IO89PDB4V0
V2	GEB0/IO62NDB4V0	GEB0/IO89NDB4V0

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
V3	$V_{CCI}B4$	$V_{CCI}B4$
V4	GEA1/IO61PDB4V0	GEA1/IO88PDB4V0
V5	GEA0/IO61NDB4V0	GEA0/IO88NDB4V0
V6	GND	GND
V7	$V_{CC33PMP}$	$V_{CC33PMP}$
V8	NC	NC
V9	V_{CC33A}	V_{CC33A}
V10	AG4	AG4
V11	AT4	AT4
V12	ATRTN2	ATRTN2
V13	AT5	AT5
V14	V_{CC33A}	V_{CC33A}
V15	NC	NC
V16	V_{CC33A}	V_{CC33A}
V17	GND	GND
V18	TMS	TMS
V19	V_{JTAG}	V_{JTAG}
V20	$V_{CCI}B2$	$V_{CCI}B2$
V21	TRST	TRST
V22	TDO	TDO
W1	NC	IO93PDB4V0
W2	GND	GND
W3	NC	IO93NDB4V0
W4	GEB2/IO59PDB4V0	GEB2/IO86PDB4V0
W5	IO59NDB4V0	IO86NDB4V0
W6	AV0	AV0
W7	GNDA	GNDA
W8	AV1	AV1
W9	AV2	AV2
W10	GNDA	GNDA
W11	AV3	AV3
W12	AV6	AV6
W13	GNDA	GNDA
W14	AV7	AV7
W15	AV8	AV8

484-Pin FBGA		
Pin Number	AFS600 Function	AFS1500 Function
W16	GNDA	GNDA
W17	AV9	AV9
W18	$V_{CCI}B2$	$V_{CCI}B2$
W19	NC	IO68PPB2V0
W20	TCK	TCK
W21	GND	GND
W22	NC	IO76PPB2V0
Y1	GEC2/IO60PDB4V0	GEC2/IO87PDB4V0
Y2	IO60NDB4V0	IO87NDB4V0
Y3	GEA2/IO58PDB4V0	GEA2/IO85PDB4V0
Y4	IO58NDB4V0	IO85NDB4V0
Y5	NCAP	NCAP
Y6	AC0	AC0
Y7	V_{CC33A}	V_{CC33A}
Y8	AC1	AC1
Y9	AC2	AC2
Y10	V_{CC33A}	V_{CC33A}
Y11	AC3	AC3
Y12	AC6	AC6
Y13	V_{CC33A}	V_{CC33A}
Y14	AC7	AC7
Y15	AC8	AC8
Y16	V_{CC33A}	V_{CC33A}
Y17	AC9	AC9
Y18	ADCGNDREF	ADCGNDREF
Y19	PTBASE	PTBASE
Y20	GNDNVM	GNDNVM
Y21	V_{CCNVM}	V_{CCNVM}
Y22	V_{PUMP}	V_{PUMP}

676-Pin FBGA



Note

For Package Manufacturing and Environmental information, visit the Resource Center at http://www.actel.com/products/solutions/package/default.aspx.

676-Pin FBGA	
Pin Number	AFS1500 Function
A1	NC
A2	GND
A3	NC
A4	NC
A5	GND
A6	NC
A7	NC
A8	GND
A9	IO17NDB0V2
A10	IO17PDB0V2
A11	GND
A12	IO18NDB0V2
A13	IO18PDB0V2
A14	IO20NDB0V2
A15	IO20PDB0V2
A16	GND
A17	IO21PDB0V2
A18	IO21NDB0V2
A19	GND
A20	IO39NDB1V2
A21	IO39PDB1V2
A22	GND
A23	NC
A24	NC
A25	GND
A26	NC
AA1	NC
AA2	V_{CCI}B4
AA3	IO93PDB4V0
AA4	GND
AA5	IO93NDB4V0
AA6	GEB2/IO86PDB4V0
AA7	IO86NDB4V0
AA8	AV0
AA9	GNDA
AA10	AV1

676-Pin FBGA	
Pin Number	AFS1500 Function
AA11	AV2
AA12	GNDA
AA13	AV3
AA14	AV6
AA15	GNDA
AA16	AV7
AA17	AV8
AA18	GNDA
AA19	AV9
AA20	V_{CCI}B2
AA21	IO68PPB2V0
AA22	TCK
AA23	GND
AA24	IO76PPB2V0
AA25	V_{CCI}B2
AA26	NC
AB1	GND
AB2	NC
AB3	GEC2/IO87PDB4V0
AB4	IO87NDB4V0
AB5	GEA2/IO85PDB4V0
AB6	IO85NDB4V0
AB7	NCAP
AB8	AC0
AB9	V_{CC33A}
AB10	AC1
AB11	AC2
AB12	V_{CC33A}
AB13	AC3
AB14	AC6
AB15	V_{CC33A}
AB16	AC7
AB17	AC8
AB18	V_{CC33A}
AB19	AC9
AB20	ADCGNDREF

676-Pin FBGA	
Pin Number	AFS1500 Function
AB21	PTBASE
AB22	GNDNVM
AB23	V_{CCNVM}
AB24	V_{PUMP}
AB25	NC
AB26	GND
AC1	NC
AC2	NC
AC3	NC
AC4	GND
AC5	V_{CCI}B4
AC6	V_{CCI}B4
AC7	PCAP
AC8	AG0
AC9	GNDA
AC10	AG1
AC11	AG2
AC12	GNDA
AC13	AG3
AC14	AG6
AC15	GNDA
AC16	AG7
AC17	AG8
AC18	GNDA
AC19	AG9
AC20	VAREF
AC21	V_{CCI}B2
AC22	PTEM
AC23	GND
AC24	NC
AC25	NC
AC26	NC
AD1	NC
AD2	NC
AD3	GND
AD4	NC

676-Pin FBGA	
Pin Number	AFS1500 Function
AD5	IO94NPB4V0
AD6	GND
AD7	VCC33N
AD8	AT0
AD9	ATRTN0
AD10	AT1
AD11	AT2
AD12	ATRTN1
AD13	AT3
AD14	AT6
AD15	ATRTN3
AD16	AT7
AD17	AT8
AD18	ATRTN4
AD19	AT9
AD20	V_{CC33A}
AD21	GND
AD22	IO76NPB2V0
AD23	NC
AD24	GND
AD25	NC
AD26	NC
AE1	GND
AE2	GND
AE3	NC
AE4	NC
AE5	NC
AE6	NC
AE7	NC
AE8	NC
AE9	GNDA
AE10	NC
AE11	NC
AE12	GNDA
AE13	NC
AE14	NC

676-Pin FBGA	
Pin Number	AFS1500 Function
AE15	GNDA
AE16	NC
AE17	NC
AE18	GNDA
AE19	NC
AE20	NC
AE21	NC
AE22	NC
AE23	NC
AE24	NC
AE25	GND
AE26	GND
AF1	NC
AF2	GND
AF3	NC
AF4	NC
AF5	NC
AF6	NC
AF7	NC
AF8	NC
AF9	V_{CC33A}
AF10	NC
AF11	NC
AF12	V_{CC33A}
AF13	NC
AF14	NC
AF15	V_{CC33A}
AF16	NC
AF17	NC
AF18	V_{CC33A}
AF19	NC
AF20	NC
AF21	NC
AF22	NC
AF23	NC
AF24	NC

676-Pin FBGA	
Pin Number	AFS1500 Function
AF25	GND
AF26	NC
B1	GND
B2	GND
B3	NC
B4	NC
B5	NC
B6	$V_{CCI}B0$
B7	NC
B8	NC
B9	$V_{CCI}B0$
B10	IO15NDB0V2
B11	IO15PDB0V2
B12	$V_{CCI}B0$
B13	IO19NDB0V2
B14	IO19PDB0V2
B15	$V_{CCI}B1$
B16	IO25NDB1V0
B17	IO25PDB1V0
B18	$V_{CCI}B1$
B19	IO33NDB1V1
B20	IO33PDB1V1
B21	$V_{CCI}B1$
B22	NC
B23	NC
B24	NC
B25	GND
B26	GND
C1	NC
C2	NC
C3	GND
C4	NC
C5	GAA1/IO01PDB0V0
C6	GAB0/IO02NDB0V0
C7	GAB1/IO02PDB0V0
C8	IO07NDB0V1

676-Pin FBGA	
Pin Number	AFS1500 Function
C9	IO07PDB0V1
C10	IO09PDB0V1
C11	IO13NDB0V2
C12	IO13PDB0V2
C13	IO24PDB1V0
C14	IO26PDB1V0
C15	IO27NDB1V1
C16	IO27PDB1V1
C17	IO35NDB1V2
C18	IO35PDB1V2
C19	GBC0/IO40NDB1V2
C20	GBA0/IO42NDB1V2
C21	IO43NDB1V2
C22	IO43PDB1V2
C23	NC
C24	GND
C25	NC
C26	NC
D1	NC
D2	NC
D3	NC
D4	GND
D5	GAA0/IO01NDB0V0
D6	GND
D7	IO04NDB0V0
D8	IO04PDB0V0
D9	GND
D10	IO09NDB0V1
D11	IO11PDB0V1
D12	GND
D13	IO24NDB1V0
D14	IO26NDB1V0
D15	GND
D16	IO31NDB1V1
D17	IO31PDB1V1
D18	GND

676-Pin FBGA	
Pin Number	AFS1500 Function
D19	GBC1/IO40PDB1V2
D20	GBA1/IO42PDB1V2
D21	GND
D22	V_{CCPLB}
D23	GND
D24	NC
D25	NC
D26	NC
E1	GND
E2	IO122NPB4V0
E3	IO121PDB4V0
E4	IO122PPB4V0
E5	IO00NDB0V0
E6	IO00PDB0V0
E7	V_{CCI}B0
E8	IO05NDB0V1
E9	IO05PDB0V1
E10	V_{CCI}B0
E11	IO11NDB0V1
E12	IO14PDB0V2
E13	V_{CCI}B0
E14	V_{CCI}B1
E15	IO29NDB1V1
E16	IO29PDB1V1
E17	V_{CCI}B1
E18	IO37NDB1V2
E19	GBB0/IO41NDB1V2
E20	V_{CCI}B1
E21	V_{COMPLB}
E22	GBA2/IO44PDB2V0
E23	IO48PPB2V0
E24	GBB2/IO45PDB2V0
E25	NC
E26	GND
F1	NC
F2	V_{CCI}B4

676-Pin FBGA	
Pin Number	AFS1500 Function
F3	IO121NDB4V0
F4	GND
F5	IO123NDB4V0
F6	GAC2/IO123PDB4V0
F7	GAA2/IO125PDB4V0
F8	GAC0/IO03NDB0V0
F9	GAC1/IO03PDB0V0
F10	IO10NDB0V1
F11	IO10PDB0V1
F12	IO14NDB0V2
F13	IO23NDB1V0
F14	IO23PDB1V0
F15	IO32NPB1V1
F16	IO34NDB1V1
F17	IO34PDB1V1
F18	IO37PDB1V2
F19	GBB1/IO41PDB1V2
F20	V_{CCI}B2
F21	IO47PPB2V0
F22	IO44NDB2V0
F23	GND
F24	IO45NDB2V0
F25	V_{CCI}B2
F26	NC
G1	NC
G2	IO119PPB4V0
G3	IO120NDB4V0
G4	IO120PDB4V0
G5	V_{CCI}B4
G6	GAB2/IO124PDB4V0
G7	IO125NDB4V0
G8	GND
G9	V_{CCI}B0
G10	IO08NDB0V1
G11	IO08PDB0V1
G12	GND

676-Pin FBGA	
Pin Number	AFS1500 Function
G13	IO22NDB1V0
G14	IO22PDB1V0
G15	GND
G16	IO32PPB1V1
G17	IO36NPB1V2
G18	V_{CCI}B1
G19	GND
G20	IO47NPB2V0
G21	IO49PDB2V0
G22	V_{CCI}B2
G23	IO46NDB2V0
G24	GBC2/IO46PDB2V0
G25	IO48NPB2V0
G26	NC
H1	GND
H2	NC
H3	IO118NDB4V0
H4	IO118PDB4V0
H5	IO119NPB4V0
H6	IO124NDB4V0
H7	GND
H8	V_{COMPLA}
H9	V_{CCPLA}
H10	V_{CCI}B0
H11	IO12NDB0V1
H12	IO12PDB0V1
H13	V_{CCI}B0
H14	V_{CCI}B1
H15	IO30NDB1V1
H16	IO30PDB1V1
H17	V_{CCI}B1
H18	IO36PPB1V2
H19	IO38NPB1V2
H20	GND
H21	IO49NDB2V0
H22	IO50PDB2V0

676-Pin FBGA	
Pin Number	AFS1500 Function
H23	IO50NDB2V0
H24	IO51PDB2V0
H25	NC
H26	GND
J1	NC
J2	V_{CCI}B4
J3	IO115PDB4V0
J4	GND
J5	IO116NDB4V0
J6	IO116PDB4V0
J7	V_{CCI}B4
J8	IO117PDB4V0
J9	V_{CCI}B4
J10	GND
J11	IO06NDB0V1
J12	IO06PDB0V1
J13	IO16NDB0V2
J14	IO16PDB0V2
J15	IO28NDB1V1
J16	IO28PDB1V1
J17	GND
J18	IO38PPB1V2
J19	IO53PDB2V0
J20	V_{CCI}B2
J21	IO52PDB2V0
J22	IO52NDB2V0
J23	GND
J24	IO51NDB2V0
J25	V_{CCI}B2
J26	NC
K1	NC
K2	NC
K3	IO115NDB4V0
K4	IO113PDB4V0
K5	V_{CCI}B4
K6	IO114NDB4V0

676-Pin FBGA	
Pin Number	AFS1500 Function
K7	IO114PDB4V0
K8	IO117NDB4V0
K9	GND
K10	V_{CC}
K11	V_{CCI}B0
K12	GND
K13	V_{CCI}B0
K14	V_{CCI}B1
K15	GND
K16	V_{CCI}B1
K17	GND
K18	GND
K19	IO53NDB2V0
K20	IO57PDB2V0
K21	GCA2/IO59PDB2V0
K22	V_{CCI}B2
K23	IO54NDB2V0
K24	IO54PDB2V0
K25	NC
K26	NC
L1	GND
L2	NC
L3	IO112PPB4V0
L4	IO113NDB4V0
L5	GFB2/IO109PDB4V0
L6	GFA2/IO110PDB4V0
L7	IO112NPB4V0
L8	IO104PDB4V0
L9	IO111PDB4V0
L10	V_{CCI}B4
L11	GND
L12	V_{CC}
L13	GND
L14	V_{CC}
L15	GND
L16	V_{CC}

676-Pin FBGA	
Pin Number	AFS1500 Function
L17	V_{CCI}B2
L18	GCB2/IO60PDB2V0
L19	IO58NDB2V0
L20	IO57NDB2V0
L21	IO59NDB2V0
L22	GCC2/IO61PDB2V0
L23	IO55PPB2V0
L24	IO56PDB2V0
L25	IO55NPB2V0
L26	GND
M1	NC
M2	V_{CCI}B4
M3	GFC2/IO108PDB4V0
M4	GND
M5	IO109NDB4V0
M6	IO110NDB4V0
M7	GND
M8	IO104NDB4V0
M9	IO111NDB4V0
M10	GND
M11	V_{CC}
M12	GND
M13	V_{CC}
M14	GND
M15	V_{CC}
M16	GND
M17	GND
M18	IO60NDB2V0
M19	IO58PDB2V0
M20	GND
M21	IO68NPB2V0
M22	IO61NDB2V0
M23	GND
M24	IO56NDB2V0
M25	V_{CCI}B2
M26	IO65PDB2V0

676-Pin FBGA	
Pin Number	AFS1500 Function
N1	NC
N2	NC
N3	IO108NDB4V0
N4	V_{CCOSC}
N5	V_{CCI}B4
N6	XTAL2
N7	GFC1/IO107PDB4V0
N8	V_{CCI}B4
N9	GFB1/IO106PDB4V0
N10	V_{CCI}B4
N11	GND
N12	V_{CC}
N13	GND
N14	V_{CC}
N15	GND
N16	V_{CC}
N17	V_{CCI}B2
N18	IO70PDB2V0
N19	V_{CCI}B2
N20	IO69PDB2V0
N21	GCA1/IO64PDB2V0
N22	V_{CCI}B2
N23	GCC0/IO62NDB2V0
N24	GCC1/IO62PDB2V0
N25	IO66PDB2V0
N26	IO65NDB2V0
P1	NC
P2	NC
P3	IO103PDB4V0
P4	XTAL1
P5	V_{CCI}B4
P6	GNDOSC
P7	GFC0/IO107NDB4V0
P8	V_{CCI}B4
P9	GFB0/IO106NDB4V0
P10	V_{CCI}B4

676-Pin FBGA	
Pin Number	AFS1500 Function
P11	V_{CC}
P12	GND
P13	V_{CC}
P14	GND
P15	V_{CC}
P16	GND
P17	V_{CCI}B2
P18	IO70NDB2V0
P19	V_{CCI}B2
P20	IO69NDB2V0
P21	GCA0/IO64NDB2V0
P22	V_{CCI}B2
P23	GCB0/IO63NDB2V0
P24	GCB1/IO63PDB2V0
P25	IO66NDB2V0
P26	IO67PDB2V0
R1	NC
R2	V_{CCI}B4
R3	IO103NDB4V0
R4	GND
R5	IO101PDB4V0
R6	IO100NPB4V0
R7	GND
R8	IO99PDB4V0
R9	IO97PDB4V0
R10	GND
R11	GND
R12	V_{CC}
R13	GND
R14	V_{CC}
R15	GND
R16	V_{CC}
R17	GND
R18	GDB2/IO83PDB2V0
R19	IO78PDB2V0
R20	GND

676-Pin FBGA	
Pin Number	AFS1500 Function
R21	IO72NDB2V0
R22	IO72PDB2V0
R23	GND
R24	IO71PDB2V0
R25	V_{CCI}B2
R26	IO67NDB2V0
T1	GND
T2	NC
T3	GFA1/IO105PDB4V0
T4	GFA0/IO105NDB4V0
T5	IO101NDB4V0
T6	IO96PDB4V0
T7	IO96NDB4V0
T8	IO99NDB4V0
T9	IO97NDB4V0
T10	V_{CCI}B4
T11	V_{CC}
T12	GND
T13	V_{CC}
T14	GND
T15	V_{CC}
T16	GND
T17	V_{CCI}B2
T18	IO83NDB2V0
T19	IO78NDB2V0
T20	GDA1/IO81PDB2V0
T21	GDB1/IO80PDB2V0
T22	IO73NDB2V0
T23	IO73PDB2V0
T24	IO71NDB2V0
T25	NC
T26	GND
U1	NC
U2	NC
U3	IO102PDB4V0
U4	IO102NDB4V0

676-Pin FBGA	
Pin Number	AFS1500 Function
U5	V_{CCI}B4
U6	IO91PDB4V0
U7	IO91NDB4V0
U8	IO92PDB4V0
U9	GND
U10	GND
U11	V_{CC33A}
U12	GNDA
U13	V_{CC33A}
U14	GNDA
U15	V_{CC33A}
U16	GNDA
U17	V_{CC}
U18	GND
U19	IO74NDB2V0
U20	GDA0/IO81NDB2V0
U21	GDB0/IO80NDB2V0
U22	V_{CCI}B2
U23	IO75NDB2V0
U24	IO75PDB2V0
U25	NC
U26	NC
V1	NC
V2	V_{CCI}B4
V3	IO100PPB4V0
V4	GND
V5	IO95PDB4V0
V6	IO95NDB4V0
V7	V_{CCI}B4
V8	IO92NDB4V0
V9	GNDNVM
V10	GNDA
V11	NC
V12	AV4
V13	NC
V14	AV5

676-Pin FBGA	
Pin Number	AFS1500 Function
V15	AC5
V16	NC
V17	GNDA
V18	IO77PPB2V0
V19	IO74PDB2V0
V20	V_{CCI}B2
V21	IO82NDB2V0
V22	GDA2/IO82PDB2V0
V23	GND
V24	GDC1/IO79PDB2V0
V25	V_{CCI}B2
V26	NC
W1	GND
W2	IO94PPB4V0
W3	IO98PDB4V0
W4	IO98NDB4V0
W5	GEC1/IO90PDB4V0
W6	GEC0/IO90NDB4V0
W7	GND
W8	V_{CCNVM}
W9	VCCIB4
W10	V_{CC15A}
W11	GNDA
W12	AC4
W13	V_{CC33A}
W14	GNDA
W15	AG5
W16	GNDA
W17	PUB
W18	V_{CCI}B2
W19	TDI
W20	GND
W21	IO84NDB2V0
W22	GDC2/IO84PDB2V0
W23	IO77NPB2V0
W24	GDC0/IO79NDB2V0

676-Pin FBGA	
Pin Number	**AFS1500 Function**
W25	NC
W26	GND
Y1	NC
Y2	NC
Y3	GEB1/IO89PDB4V0
Y4	GEB0/IO89NDB4V0
Y5	V_{CCI}B4
Y6	GEA1/IO88PDB4V0
Y7	GEA0/IO88NDB4V0
Y8	GND
Y9	$V_{CC33PMP}$
Y10	NC
Y11	V_{CC33A}
Y12	AG4
Y13	AT4
Y14	ATRTN2
Y15	AT5
Y16	V_{CC33A}
Y17	NC
Y18	V_{CC33A}
Y19	GND
Y20	TMS
Y21	V_{JTAG}
Y22	VCCIB2
Y23	TRST
Y24	TDO
Y25	NC
Y26	NC

Part Number and Revision Date

Part Number 51700092-016-1
Revised July 2009

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v2.0)	Page
Preliminary v1.7 (October 2008)	The version number category was changed from Preliminary to Production, which means the datasheet contains information based on final characterization. The version number changed from Preliminary v1.7 to v2.0.	N/A
	"180-Pin QFN" table was updated to remove the duplicates of pins B12 and B34.	4-4
Advance v1.6 (August 2008)	The version number category was changed from Advance to Preliminary, which means the datasheet contains information based on simulation and/or initial characterization. The information is believed to be correct, but changes are possible.	N/A
Advance v1.4 (July 2008)	The title of the datasheet changed from Actel Programmable System Chips to Actel Fusion Mixed-Signal FPGAs. In addition, all instances of programmable system chip were changed to mixed-signal FPGA.	N/A
Advance v1.1 (May 2008)	The "108-Pin QFN"figure was updated. D1 to D4 are new and the figure was changed to bottom view. The note below the figure is new.	4-1
	The "180-Pin QFN"figure was updated. D1 to D4 are new and the figure was changed to bottom view. The note below the figure is new.	4-3
Advance v0.9 October 2007	This change table states that in the "208-Pin PQFP" table listed under the Advance v0.8 changes, the AFS090 device had a pin change. That is incorrect. Pin 102 was updated for AFS250 and AFS600. The function name changed from $V_{CC33ACAP}$ to V_{CC33A}.	4-8
Advance v0.8 (June 2007)	In the "108-Pin QFN" table, the function changed from $V_{CC33ACAP}$ to V_{CC33A} for the following pin: B25	4-2
	In the "180-Pin QFN" table, the function changed from $V_{CC33ACAP}$ to V_{CC33A} for the following pins: AFS090: B29 AFS250: B29	4-4
	In the "208-Pin PQFP" table, the function changed from $V_{CC33ACAP}$ to V_{CC33A} for the following pins: AFS090: 102 AFS250: 102	4-8
	In the "256-Pin FBGA" table, the function changed from $V_{CC33ACAP}$ to V_{CC33A} for the following pins: AFS090: T14 AFS250: T14 AFS600: T14 AFS1500: T14	4-12

Previous Version	Changes in Current Version (v2.0)	Page
Advance v0.8 (continued)	In the "484-Pin FBGA" table, the function changed from $V_{CC33ACAP}$ to V_{CC33A} for the following pins: AFS600: AB18 AFS1500: AB18	4-20
	In the "676-Pin FBGA" table, the function changed from $V_{CC33ACAP}$ to V_{CC33A} for the following pins: AFS1500: AD20	4-28
Advance v0.7 (January 2007)	The VMV pins have now been tied internally with the V_{CCI} pins.	N/A
	The AFS090"108-Pin QFN" table was updated.	4-2
	The AFS090 and AFS250 devices were updated in the "108-Pin QFN" table.	4-2
	The AFS250 device was updated in the "208-Pin PQFP" table.	4-8
Advance v0.7 (continued)	The AFS600 device was updated in the "208-Pin PQFP" table.	4-8
	The AFS090, AFS250, AFS600, and AFS1500 devices were updated in the "256-Pin FBGA" table.	4-12
	The AFS600 and AFS1500 devices were updated in the "484-Pin FBGA" table.	4-20
	The AFS600 device was updated in the "676-Pin FBGA" table.	4-28
Advance v0.5 (June 2006)	The heading was incorrect in the "208-Pin PQFP" table. It should be AFS250 and not AFS090.	4-8
Advance v0.4 (April 2006)	The "256-Pin FBGA" table for the AFS1500 is new.	4-12
Advance v0.2 (April 2006)	The "108-Pin QFN" table for the AFS090 device is new.	4-2
	The "180-Pin QFN" table for the AFS090 device is new.	4-4
	The "208-Pin PQFP" table for the AFS090 device is new.	4-8
	The "256-Pin FBGA" table for the AFS090 device is new.	4-12
	The "256-Pin FBGA" table for the AFS250 device is new.	4-12

Datasheet Categories

Categories

In order to provide the latest information to designers, some datasheets are published before data has been fully characterized. Datasheets are designated as "Product Brief," "Advance," and "Production". The definition of these categories are as follows:

Product Brief

The product brief is a summarized version of a datasheet (advance or production) and contains general product information. This document gives an overview of specific device and family information.

Advance

This version contains initial estimated information based on simulation, other products, devices, or speed grades. This information can be used as estimates, but not for production. This label only applies to the DC and Switching Characteristics chapter of the datasheet and will only be used when the data has not been fully characterized.

Unmarked (production)

This version contains information that is considered to be final.

Export Administration Regulations (EAR)

The products described in this document are subject to the Export Administration Regulations (EAR). They could require an approved export license prior to export from the United States. An export includes release of product or disclosure of technology to a foreign national inside or outside the United States.

Actel Safety Critical, Life Support, and High-Reliability Applications Policy

The Actel products described in this advance status document may not have completed Actel's qualification process. Actel may amend or enhance products during the product introduction and qualification process, resulting in changes in device functionality or performance. It is the responsibility of each customer to ensure the fitness of any Actel product (but especially a new product) for a particular purpose, including appropriateness for safety-critical, life-support, and other high-reliability applications. Consult Actel's Terms and Conditions for specific liability exclusions relating to life-support applications. A reliability report covering all of Actel's products is available on the Actel website at http://www.actel.com/documents/ORT_Report.pdf. Actel also offers a variety of enhanced qualification and lot acceptance screening procedures. Contact your local Actel sales office for additional reliability information.

Section II – User's Guide

Actel
POWER MATTERS

Low-Power Flash Technology

Actel®
POWER MATTERS

1 – FPGA Array Architecture in Low-Power Flash Devices

Device Architecture

Advanced Flash Switch

Unlike SRAM FPGAs, the low-power flash devices use a live-at-power-up ISP flash switch as their programming element. Flash cells are distributed throughout the device to provide nonvolatile, reconfigurable programming to connect signal lines to the appropriate VersaTile inputs and outputs. In the flash switch, two transistors share the floating gate, which stores the programming information (Figure 1-1). One is the sensing transistor, which is only used for writing and verification of the floating gate voltage. The other is the switching transistor. The latter is used to connect or separate routing nets, or to configure VersaTile logic. It is also used to erase the floating gate. Dedicated high-performance lines are connected as required using the flash switch for fast, low-skew, global signal distribution throughout the device core. Maximum core utilization is possible for virtually any design. The use of the flash switch technology also removes the possibility of firm errors, which are increasingly common in SRAM-based FPGAs.

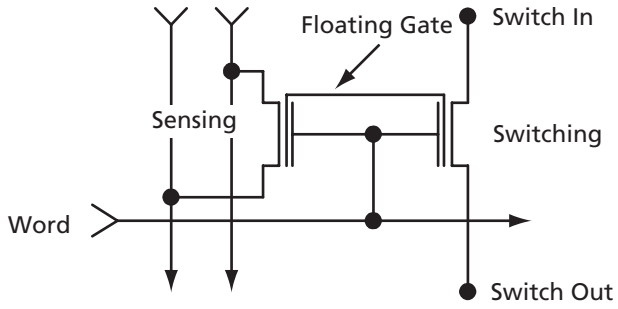


Figure 1-1 • **Flash-Based Switch**

FPGA Array Architecture Support

The flash FPGAs listed in Table 1-1 support the architecture features described in this document.

Table 1-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO®	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC®3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 1-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

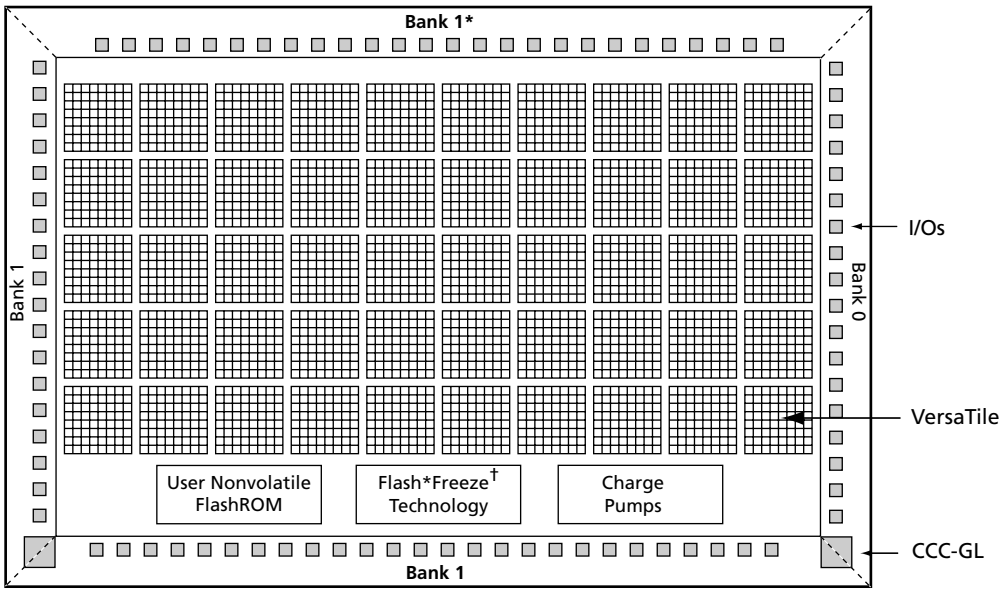
In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 1-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Device Overview

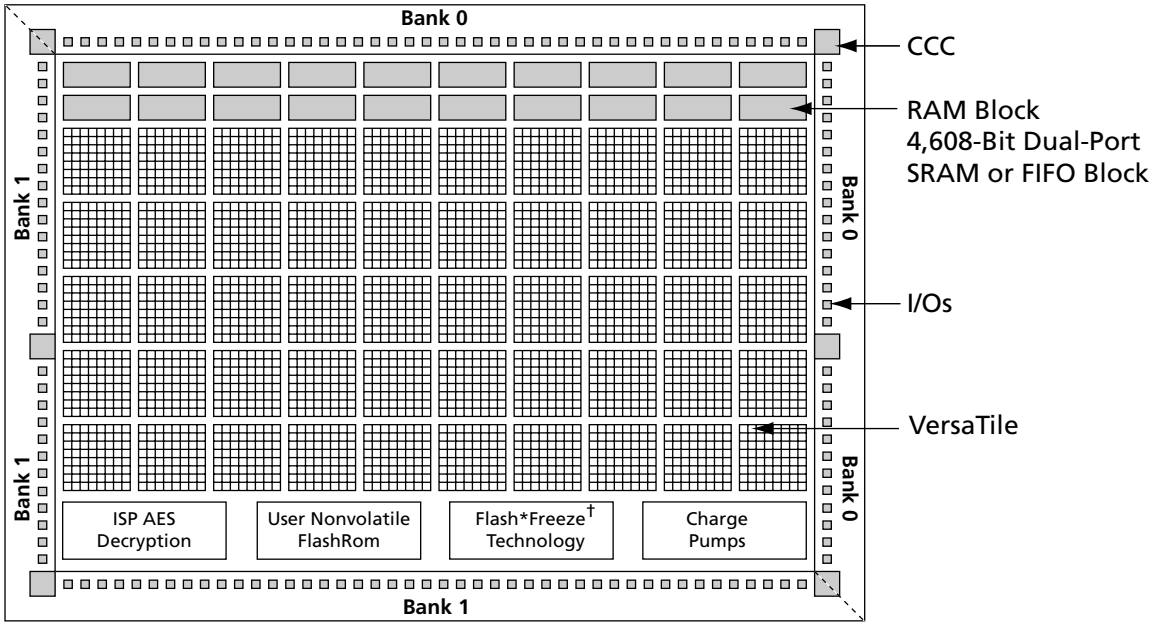
The low-power flash devices consist of multiple distinct programmable architectural features (Figure 1-5 on page 1-5 through Figure 1-7 on page 1-6):

- FPGA fabric/core (VersaTiles)
- Routing and clock resources (VersaNets)
- FlashROM
- Dedicated SRAM and/or FIFO
  - 30 k gate and smaller device densities do not support SRAM or FIFO.
  - Automotive devices do not support FIFO operation.
- I/O structures
- Flash*Freeze technology and low-power modes



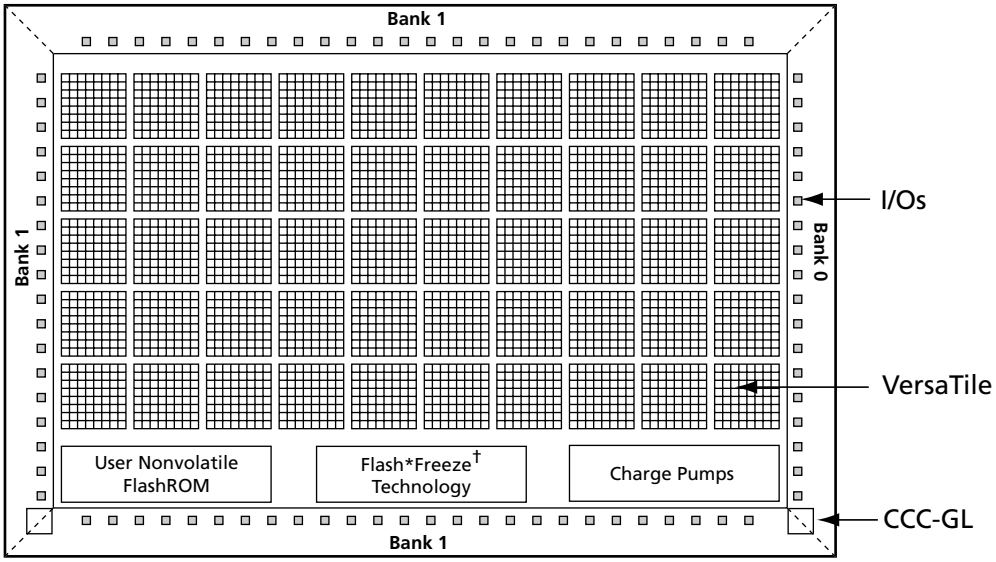
Notes: ** Bank 0 for the 30 k devices*
*† Flash*Freeze mode is supported on IGLOO devices.*

Figure 1-2 • **IGLOO and ProASIC3 nano Device Architecture Overview with Two I/O Banks (applies to 10 k and 30 k device densities, excluding IGLOO PLUS devices)**



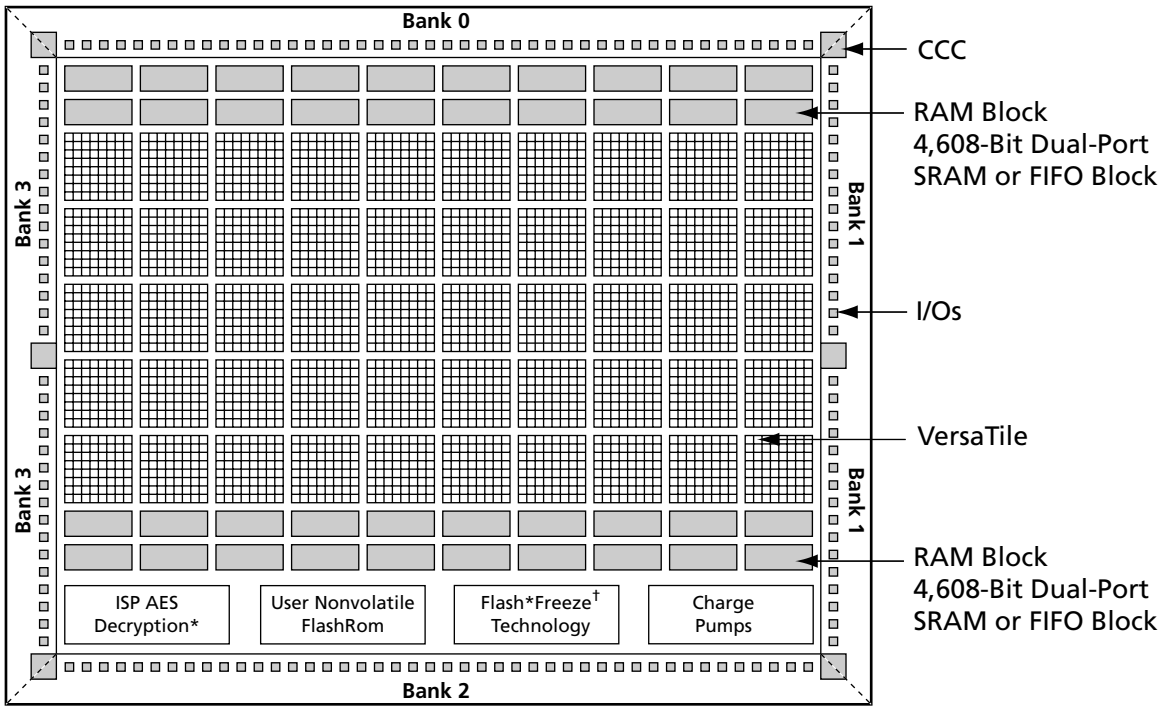
Note: † *Flash*Freeze mode is supported on IGLOO devices.*

Figure 1-3 • **IGLOO Device Architecture Overview with Two I/O Banks with RAM and PLL (60 k and 125 k gate densities)**



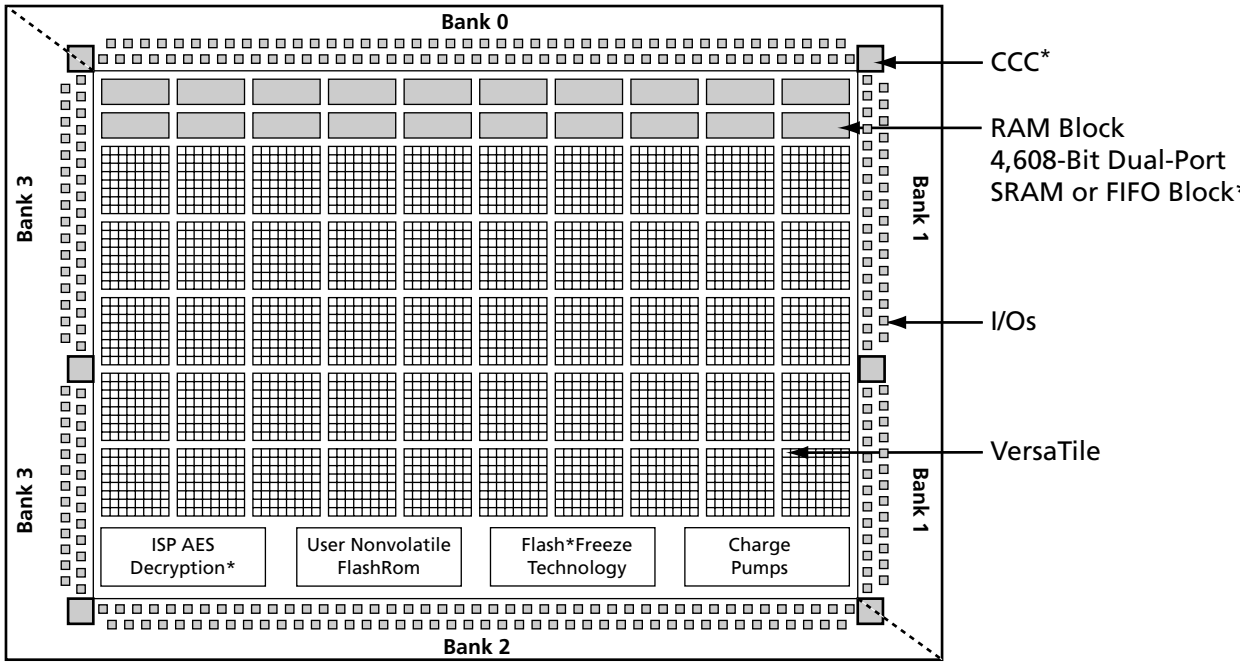
Note: † *Flash*Freeze mode is supported on IGLOO devices.*

Figure 1-4 • **IGLOO Device Architecture Overview with Three I/O Banks (AGLN015, AGLN020, A3PN015, and A3PN020)**



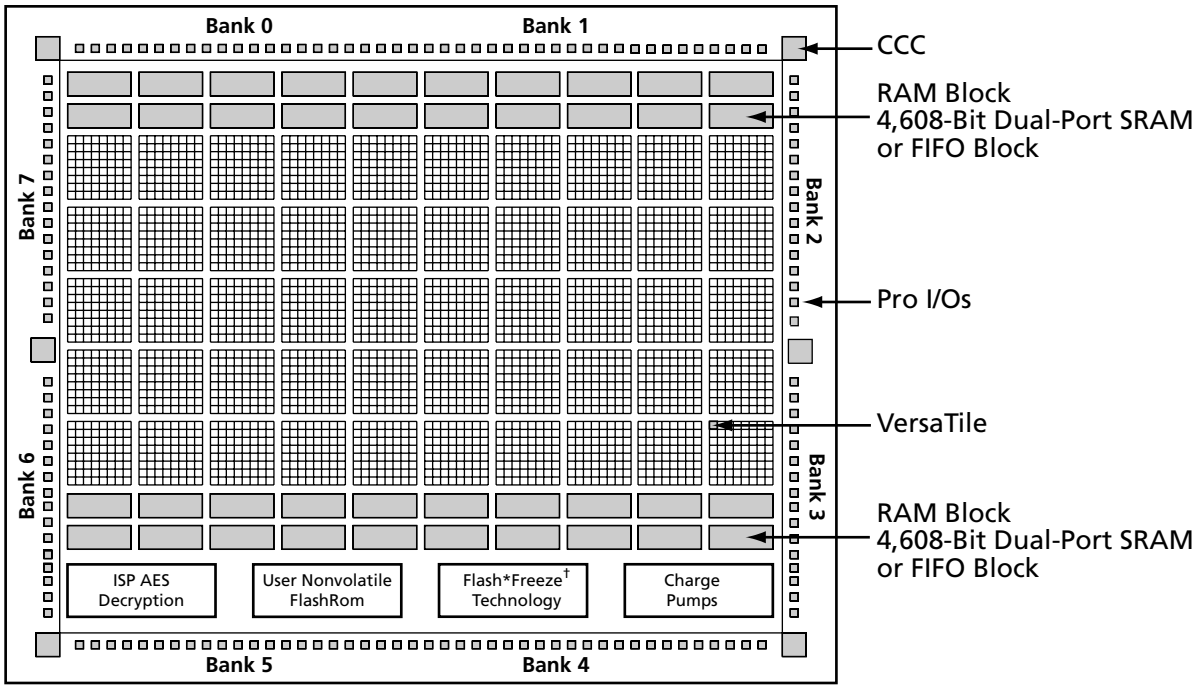
*Note: Flash*Freeze technology only applies to IGLOO and ProASIC3L families.*

Figure 1-5 • **IGLOO, IGLOO nano, ProASIC3 nano, and ProASIC3/L Device Architecture Overview with Four I/O Banks (AGL600 device is shown)**



*Note: * AGLP030 does not contain a PLL or support AES security.*

Figure 1-6 • **IGLOO PLUS Device Architecture Overview with Four I/O Banks**



*Note: Flash*Freeze technology only applies to IGLOOe devices.*

Figure 1-7 • **IGLOOe and ProASIC3E Device Architecture Overview (AGLE600 device is shown)**

Core Architecture

VersaTile

The proprietary IGLOO and ProASIC3 device architectures provide granularity comparable to gate arrays. The device core consists of a sea-of-VersaTiles architecture.

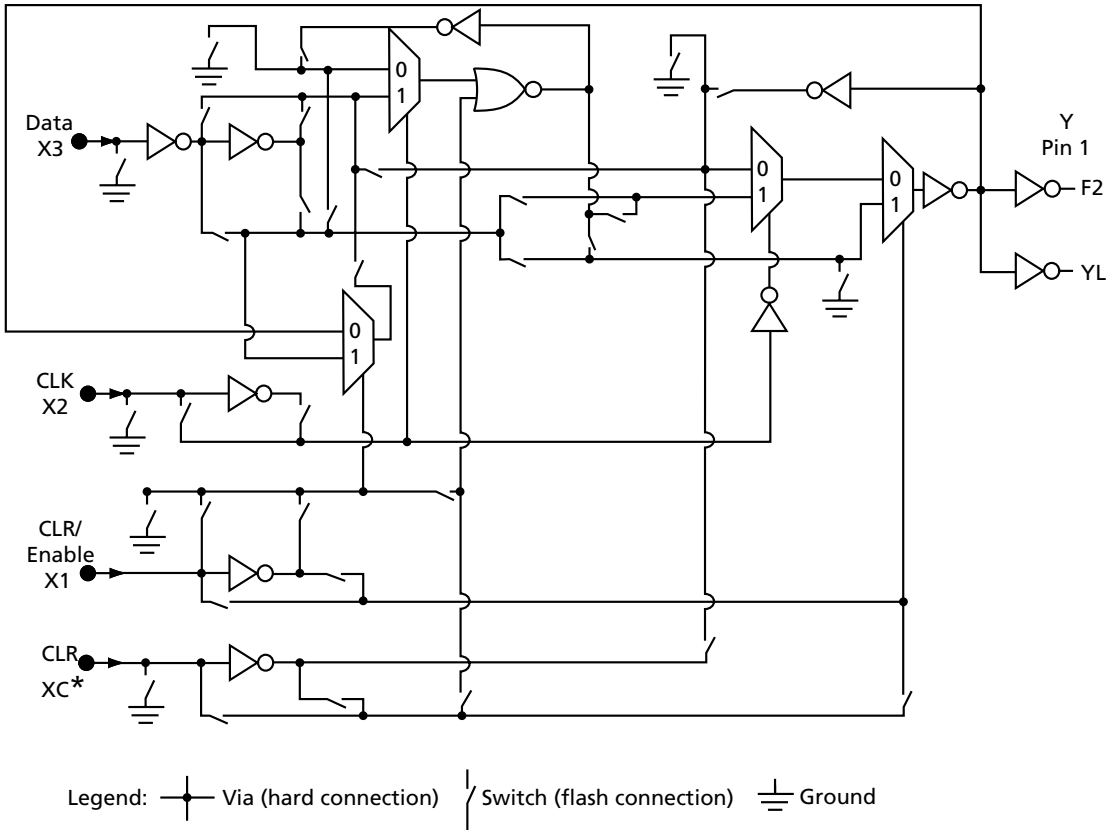
As illustrated in Figure 1-8, there are four inputs in a logic VersaTile cell, and each VersaTile can be configured using the appropriate flash switch connections:

- Any 3-input logic function
- Latch with clear or set
- D-flip-flop with clear or set
- Enable D-flip-flop with clear or set (on a 4th input)

VersaTiles can flexibly map the logic and sequential gates of a design. The inputs of the VersaTile can be inverted (allowing bubble pushing), and the output of the tile can connect to high-speed, very-long-line routing resources. VersaTiles and larger functions can be connected with any of the four levels of routing hierarchy.

When the VersaTile is used as an enable D-flip-flop, SET/CLR is supported by a fourth input. The SET/CLR signal can only be routed to this fourth input over the VersaNet (global) network. However, if, in the user's design, the SET/CLR signal is not routed over the VersaNet network, a compile warning message will be given, and the intended logic function will be implemented by two VersaTiles instead of one.

The output of the VersaTile is F2 when the connection is to the ultra-fast local lines, or YL when the connection is to the efficient long-line or very-long-line resources.



* *This input can only be connected to the global clock distribution network.*

Figure 1-8 • **Low-Power Flash Device Core VersaTile**

Array Coordinates

During many place-and-route operations in the Actel Designer software tool, it is possible to set constraints that require array coordinates. Table 1-2 provides array coordinates of core cells and memory blocks for IGLOO and ProASIC3 devices. Table 1-3 provides the information for IGLOO PLUS devices. Table 1-4 on page 1-9 provides the information for IGLOO nano and ProASIC3 nano devices. The array coordinates are measured from the lower left (0, 0). They can be used in region constraints for specific logic groups/blocks, designated by a wildcard, and can contain core cells, memories, and I/Os.

I/O and cell coordinates are used for placement constraints. Two coordinate systems are needed because there is not a one-to-one correspondence between I/O cells and core cells. In addition, the I/O coordinate system changes depending on the die/package combination. It is not listed in Table 1-2. The Designer ChipPlanner tool provides the array coordinates of all I/O locations. I/O and cell coordinates are used for placement constraints. However, I/O placement is easier by package pin assignment.

Figure 1-9 on page 1-9 illustrates the array coordinates of a 600 k gate device. For more information on how to use array coordinates for region/placement constraints, see the *Designer User's Guide* or online help (available in the software) for software tools.

Table 1-2 • **IGLOO and ProASIC3 Array Coordinates**

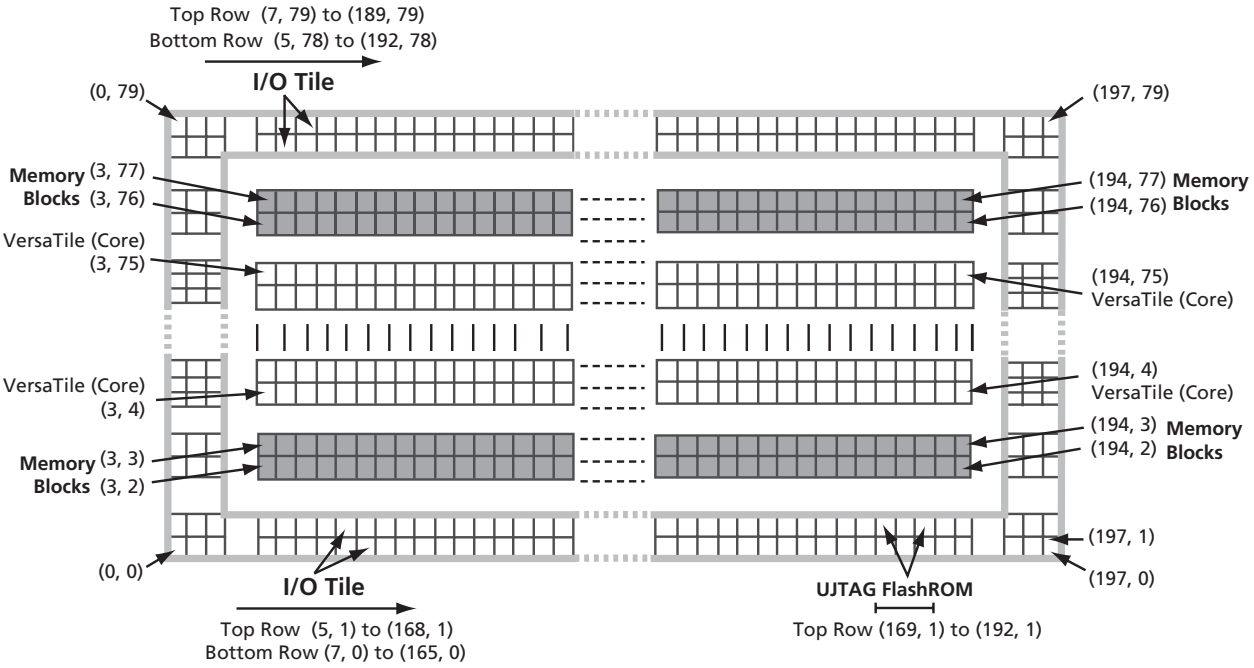
Device		VersaTiles				Memory Rows		Entire Die	
		Min.		Max.		Bottom	Top	Min.	Max.
IGLOO	ProASIC3 / ProASIC3L	x	y	x	y	(x, y)	(x, y)	(x, y)	(x, y)
AGL015	A3P015	3	2	34	13	None	None	(0, 0)	(37, 15)
AGL030	A3P030	3	3	66	13	None	None	(0, 0)	(69, 15)
AGL060	A3P060	3	2	66	25	None	(3, 26)	(0, 0)	(69, 29)
AGL125	A3P125	3	2	130	25	None	(3, 26)	(0, 0)	(133, 29)
AGL250	A3P250/L	3	2	130	49	None	(3, 50)	(0, 0)	(133, 53)
AGL400	A3P400	3	2	194	49	None	(3, 50)	(0, 0)	(197, 53)
AGL600	A3P600/L	3	4	194	75	(3, 2)	(3, 76)	(0, 0)	(197, 79)
AGL1000	A3P1000/L	3	4	258	99	(3, 2)	(3, 100)	(0, 0)	(261, 103)
AGLE600	A3PE600/L, RT3PE600L	3	4	194	75	(3, 2)	(3, 76)	(0, 0)	(197, 79)
	A3PE1500	3	4	322	123	(3, 2)	(3, 124)	(0, 0)	(325, 127)
AGLE3000	A3PE3000/L, RT3PE3000L	3	6	450	173	(3, 2) or (3, 4)	(3, 174) or (3, 176)	(0, 0)	(453, 179)

Table 1-3 • **IGLOO PLUS Array Coordinates**

Device	VersaTiles				Memory Rows		Entire Die	
	Min.		Max.		Bottom	Top	Min.	Max.
IGLOO PLUS	x	y	x	y	(x, y)	(x, y)	(x, y)	(x, y)
AGLP030	2	3	67	13	None	None	(0, 0)	(69, 15)
AGLP060	2	2	67	25	None	(3, 26)	(0, 0)	(69, 29)
AGLP125	2	2	131	25	None	(3, 26)	(0, 0)	(133, 29)

Table 1-4 • **IGLOO nano and ProASIC3 nano Array Coordinates**

Device		VersaTiles		Memory Rows		Entire Die	
		Min.	Max.	Bottom	Top	Min.	Max.
IGLOO nano	ProASIC3 nano	(x, y)	(x, y)	(x, y)	(x, y)	(x, y)	(x, y)
AGLN010	A3P010	(0, 2)	(32, 5)	None	None	(0, 0)	(34, 5)
AGLN015	A3PN015	(0, 2)	(32, 9)	None	None	(0, 0)	(34, 9)
AGLN020	A3PN020	(0, 2)	32, 13)	None	None	(0, 0)	(34, 13)
AGLN060	A3PN060	(3, 2)	(66, 25)	None	(3, 26)	(0, 0)	(69, 29)
AGLN125	A3PN125	(3, 2)	(130, 25)	None	(3, 26)	(0, 0)	(133, 29)
AGLN250	A3PN250	(3, 2)	(130, 49)	None	(3, 50)	(0, 0)	(133, 49)



Note: The vertical I/O tile coordinates are not shown. West-side coordinates are {(0, 2) to (2, 2)} to {(0, 77) to (2, 77)}; east-side coordinates are {(195, 2) to (197, 2)} to {(195, 77) to (197, 77)}.

Figure 1-9 • **Array Coordinates for AGL600, AGLE600, A3P600, and A3PE600**

Routing Architecture

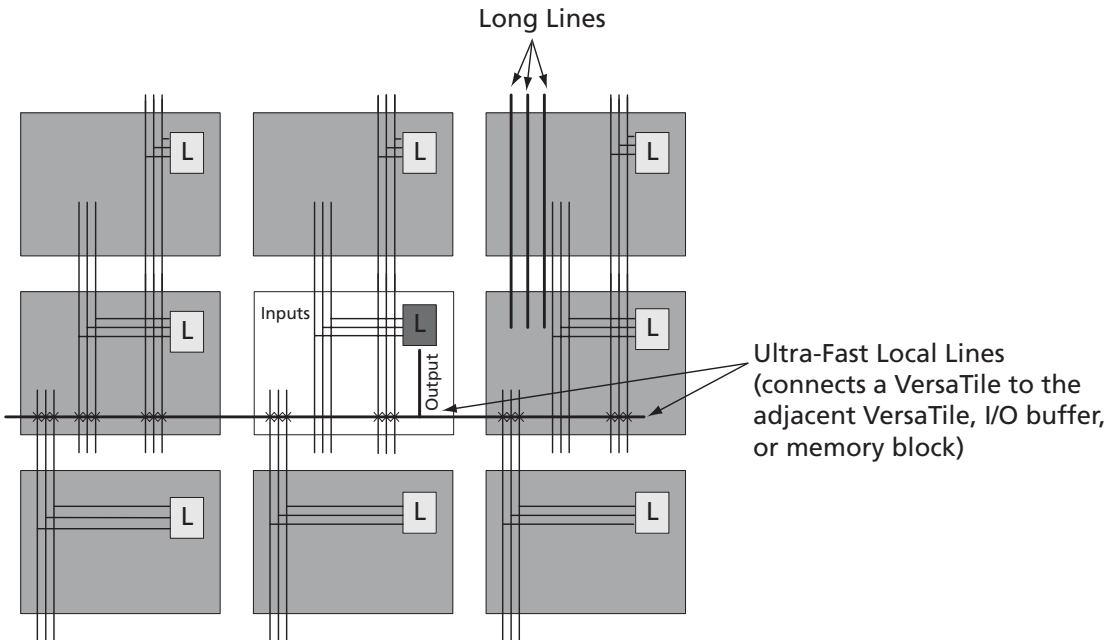
The routing structure of low-power flash devices is designed to provide high performance through a flexible four-level hierarchy of routing resources: ultra-fast local resources; efficient long-line resources; high-speed, very-long-line resources; and the high-performance VersaNet networks.

The ultra-fast local resources are dedicated lines that allow the output of each VersaTile to connect directly to every input of the eight surrounding VersaTiles (Figure 1-10 on page 1-10). The exception to this is that the SET/CLR input of a VersaTile configured as a D-flip-flop is driven only by the VersaTile global network.

The efficient long-line resources provide routing for longer distances and higher-fanout connections. These resources vary in length (spanning one, two, or four VersaTiles), run both vertically and horizontally, and cover the entire device (Figure 1-11 on page 1-11). Each VersaTile can drive signals onto the efficient long-line resources, which can access every input of every VersaTile. Routing software automatically inserts active buffers to limit loading effects.

The high-speed, very-long-line resources, which span the entire device with minimal delay, are used to route very long or high-fanout nets: length ±12 VersaTiles in the vertical direction and length ±16 in the horizontal direction from a given core VersaTile (Figure 1-12 on page 1-11). Very long lines in low-power flash devices have been enhanced over those in previous ProASIC families. This provides a significant performance boost for long-reach signals.

The high-performance VersaNet global networks are low-skew, high-fanout nets that are accessible from external pins or internal logic. These nets are typically used to distribute clocks, resets, and other high-fanout nets requiring minimum skew. The VersaNet networks are implemented as clock trees, and signals can be introduced at any junction. These can be employed hierarchically, with signals accessing every input of every VersaTile. For more details on VersaNets, refer to *Global Resources in Actel Low-Power Flash Devices.*



Note: Input to the core cell for the D-flip-flop set and reset is only available via the VersaNet global network connection.

Figure 1-10 • **Ultra-Fast Local Lines Connected to the Eight Nearest Neighbors**

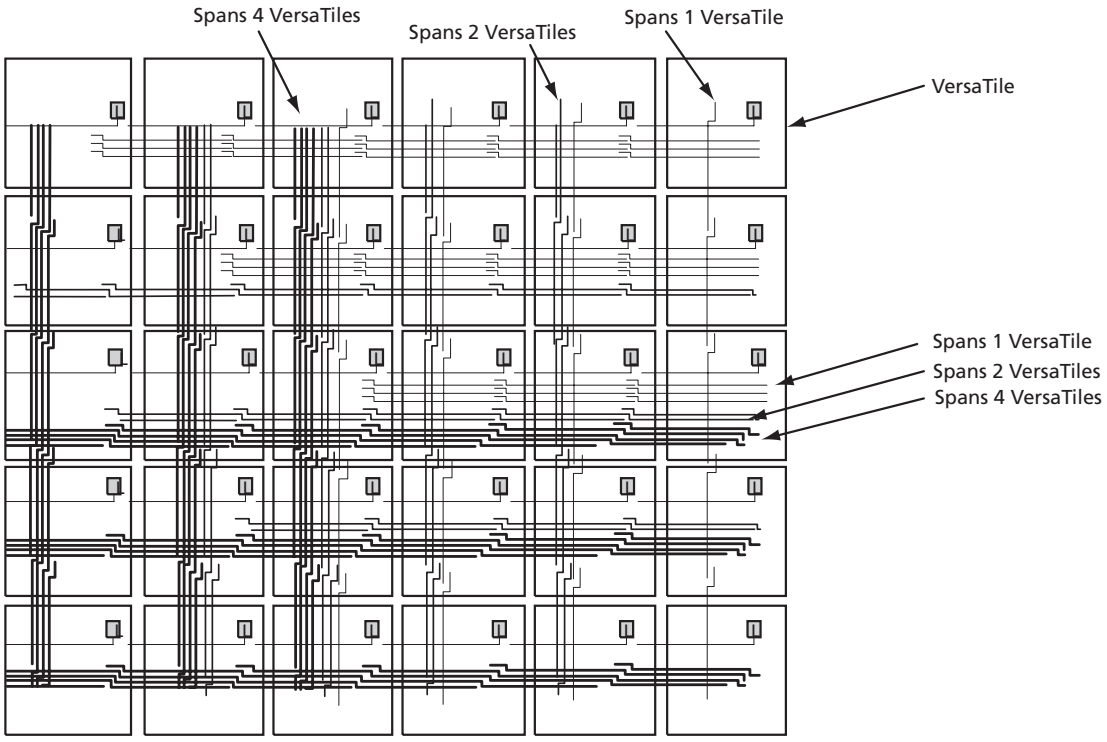


Figure 1-11 • **Efficient Long-Line Resources**

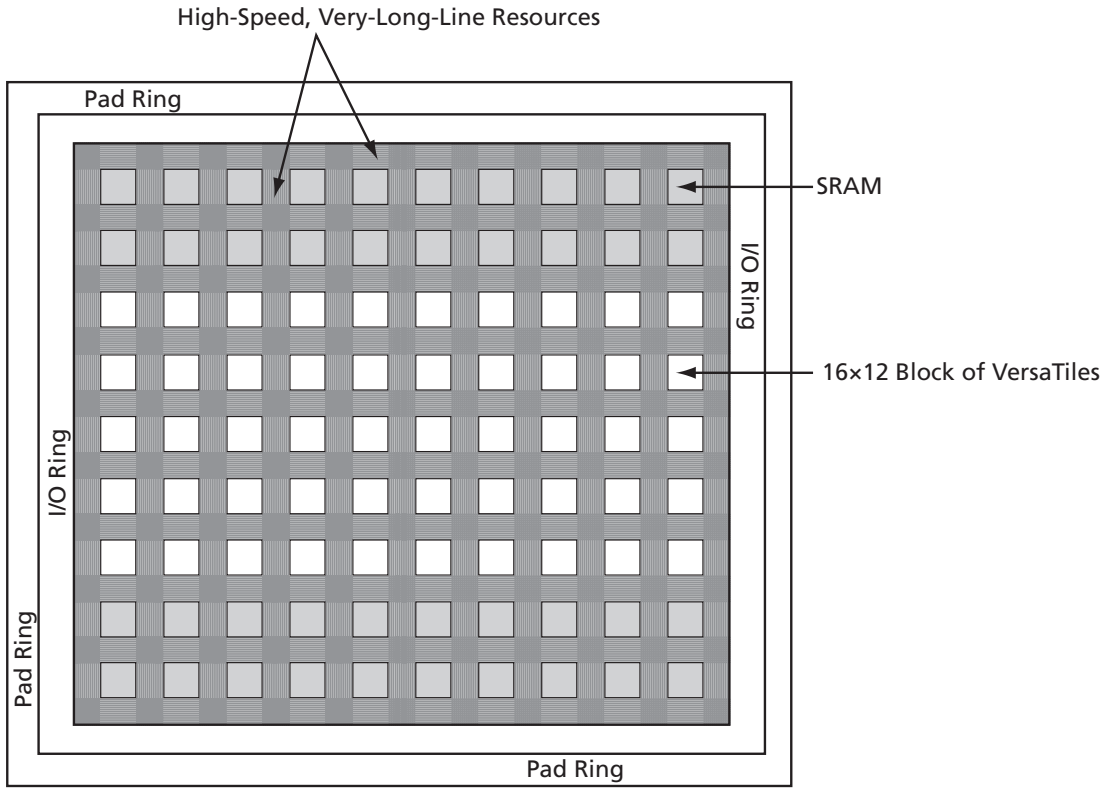


Figure 1-12 • **Very-Long-Line Resources**

Related Documents

Handbook Documents

Global Resources in Actel Low-Power Flash Devices

http://www.actel.com/documents/LPD_Glorbal_HBs.pdf

User's Guides

Designer User's Guide

http://www.actel.com/documents/designer_ug.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet. To improve usability for customers, the device architecture information has now been split into handbook sections, which also include usage information. No technical changes were made to the content unless explicitly listed.

Part Number 51700094-002-4

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 1-1 · Flash-Based FPGAs.	1-2
	Figure 1-2 · IGLOO and ProASIC3 nano Device Architecture Overview with Two I/O Banks (applies to 10 k and 30 k device densities, excluding IGLOO PLUS devices) through Figure 1-5 · IGLOO, IGLOO nano, ProASIC3 nano, and ProASIC3/L Device Architecture Overview with Four I/O Banks (AGL600 device is shown) are new.	1-3, 1-4
	Table 1-4 · IGLOO nano and ProASIC3 nano Array Coordinates is new.	1-9
v1.2 (June 2008)	The title of this document was changed from "Core Architecture of IGLOO and ProASIC3 Devices" to "FPGA Array Architecture in Low-Power Flash Devices."	1-1
	The "FPGA Array Architecture Support" section was revised to include new families and make the information more concise.	1-2
	Table 1-2 · IGLOO and ProASIC3 Array Coordinates was updated to include Military ProASIC3/EL and RT ProASIC3 devices.	1-8
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 1-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	1-2
v1.0 (January 2008)	Table 1-1 · Flash-Based FPGAs and the accompanying text was updated to include the IGLOO PLUS family. The "IGLOO Terminology" section and "Device Overview" section are new.	1-2
	The "Device Overview" section was updated to note that 15 k devices do not support SRAM or FIFO.	1-3
	Figure 1-6 · IGLOO PLUS Device Architecture Overview with Four I/O Banks is new.	1-5
	Table 1-2 · IGLOO and ProASIC3 Array Coordinates was updated to add A3P015 and AGL015.	1-8
	Table 1-3 · IGLOO PLUS Array Coordinates is new.	1-8

Global Resources and Clock Conditioning

Actel
POWER MATTERS

2 – Global Resources in Actel Low-Power Flash Devices

Introduction

Actel IGLOO,® Fusion, and ProASIC®3 FPGA devices offer a powerful, low-delay VersaNet global network scheme and have extensive support for multiple clock domains. In addition to the Clock Conditioning Circuits (CCCs) and phase-locked loops (PLLs), there is a comprehensive global clock distribution network called a VersaNet global network. Each logical element (VersaTile) input and output port has access to these global networks. The VersaNet global networks can be used to distribute low-skew clock signals or high-fanout nets. In addition, these highly segmented VersaNet global networks offer users the flexibility to create low-skew local networks using spines. This document describes VersaNet global networks and discusses how to assign signals to these global networks and spines in a design flow. Details concerning low-power flash device PLLs are described in *Clock Conditioning Circuits in IGLOO and ProASIC3 Devices*. This document describes the low-power flash devices' global architecture and uses of these global networks in designs.

Global Architecture

Low-power flash devices offer powerful and flexible control of circuit timing through the use of analog circuitry. Each chip has up to six CCCs, some with PLLs.

- In IGLOOe, ProASIC3EL, and ProASIC3E devices, all CCCs have PLLs—hence, 6 PLLs per device.
- In IGLOO, IGLOO nano, IGLOO PLUS, ProASIC3, and ProASIC3L devices, the west CCC contains a PLL core (except in 10 k through 30 k devices).
- In Fusion devices, the west CCC also contains a PLL core. In the two larger devices (AFS600 and AFS1500), the west and east CCCs each contain a PLL.

Each PLL includes delay lines, a phase shifter (0°, 90°, 180°, 270°), and clock multipliers/dividers. Each CCC has all the circuitry needed for the selection and interconnection of inputs to the VersaNet global network. The east and west CCCs each have access to three VersaNet global lines on each side of the chip (six global lines total). The CCCs at the four corners each have access to three quadrant global lines in each quadrant of the chip (except in 10 k through 30 k gate devices).

The nano 10 k, 15 k, and 20 k devices support four VersaNet global resources, and 30 k devices support six global resources. The 10 k through 30 k devices have simplified CCCs called CCC-GLs.

The flexible use of the VersaNet global network allows the designer to address several design requirements. User applications that are clock-resource-intensive can easily route external or gated internal clocks using VersaNet global routing networks. Designers can also drastically reduce delay penalties and minimize resource usage by mapping critical, high-fanout nets to the VersaNet global network.

The following sections give an overview of the VersaNet global network, the structure of the global network, and the clock aggregation feature that enables a design to have very low clock skew using spines.

Global Resource Support in Flash-Based Devices

The flash FPGAs listed in Table 2-1 support the global resources and the functions described in this document.

Table 2-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
	IGLOO nano	The industry's lowest-power, smallest-size solution
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO products as listed in Table 2-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 2-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

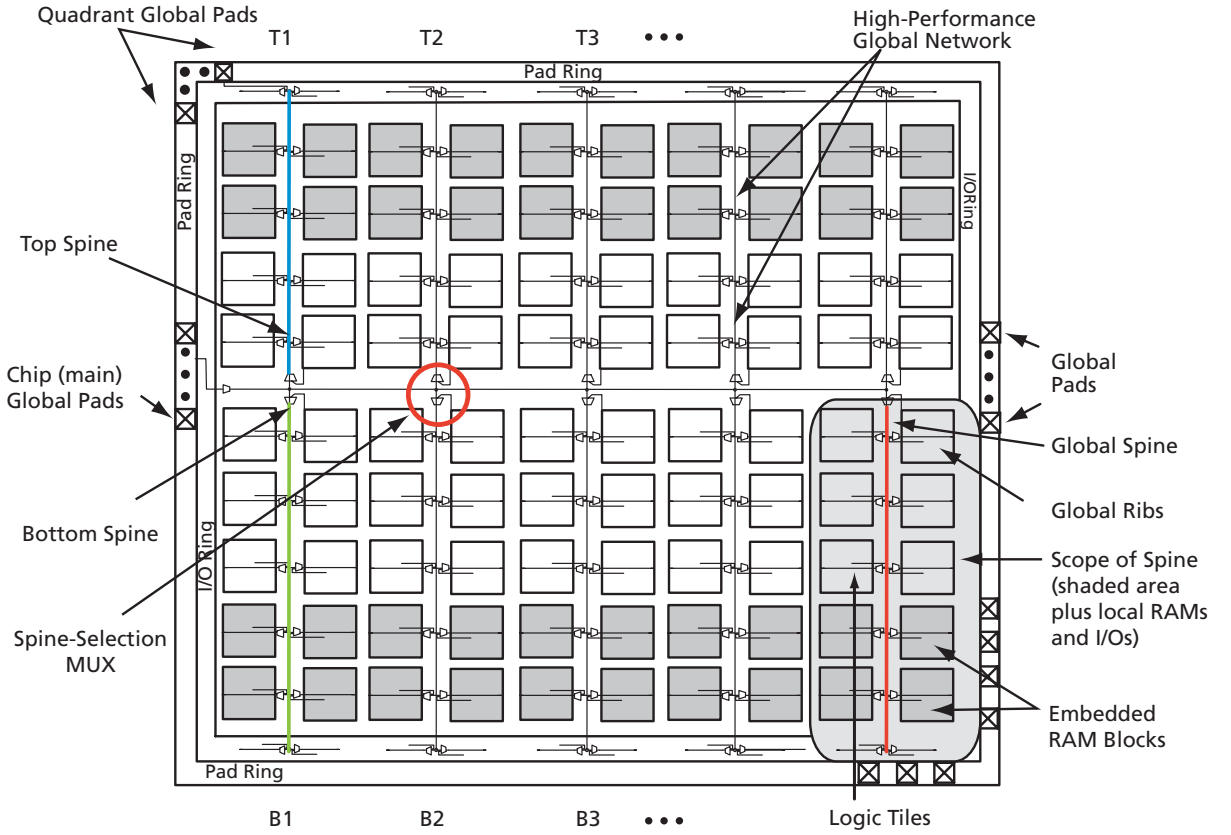
VersaNet Global Network Distribution

One of the architectural benefits of low-power flash architecture is the set of powerful, low-delay VersaNet global networks that can access the VersaTiles, SRAM, and I/O tiles of the device. Each device offers a chip global network with six global lines (except for nano 10 k, 15 k, and 20 k gate devices) that are distributed from the center of the FPGA array. In addition, each device (except the 10 k through 30 k gate device) has four quadrant global networks, each with three regional global line resources. These quadrant global networks can only drive a signal inside their own quadrant. Each core VersaTile has access to nine global line resources—three quadrant and six chip-wide (main) global networks—and a total of 18 globals are available on the device (3 × 4 regional from each quadrant and 6 global).

Figure 2-1 shows an overview of the VersaNet global network and device architecture for devices 60 k and above. Figure 2-2 and Figure 2-3 on page 2-4 show simplified VersaNet global networks.

The VersaNet global networks are segmented and consist of VersaNet global networks, spines, global ribs, and global multiplexers (MUXes), as shown in Figure 2-1. The global networks are driven from the global rib at the center of the die or quadrant global networks at the north or south side of the die. The global network uses the MUX trees to access the spine, and the spine uses the clock ribs to access the VersaTile. Access is available to the chip or quadrant global networks and the spines through the global MUXes. Access to the spine using the global MUXes is explained in the "Spine Architecture" section on page 2-5.

These VersaNet global networks offer fast, low-skew routing resources for high-fanout nets, including clock signals. In addition, these highly segmented global networks offer users the flexibility to create low-skew local networks using spines for up to 252 internal/external clocks or other high-fanout nets in low-power flash devices. Optimal usage of these low-skew networks can result in significant improvement in design performance.



Note: Not applicable to 10 k through 30 k gate devices

Figure 2-1 • **Overview of VersaNet Global Network and Device Architecture**

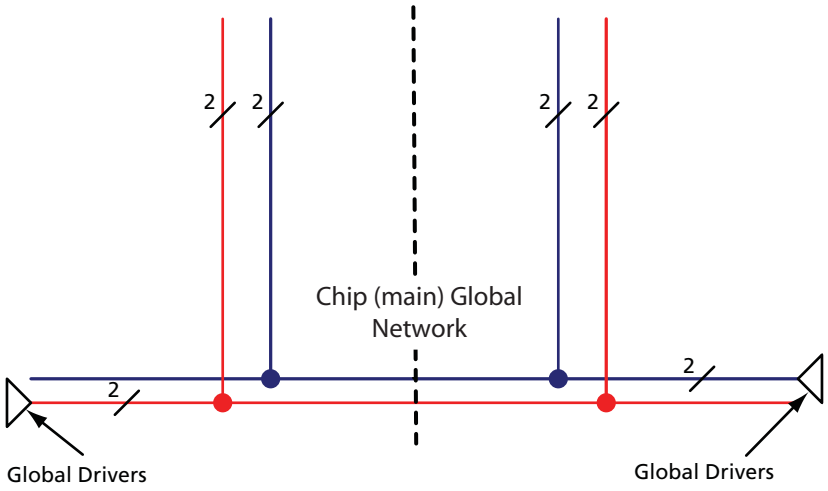


Figure 2-2 • **Simplified VersaNet Global Network (30 k gates and below)**

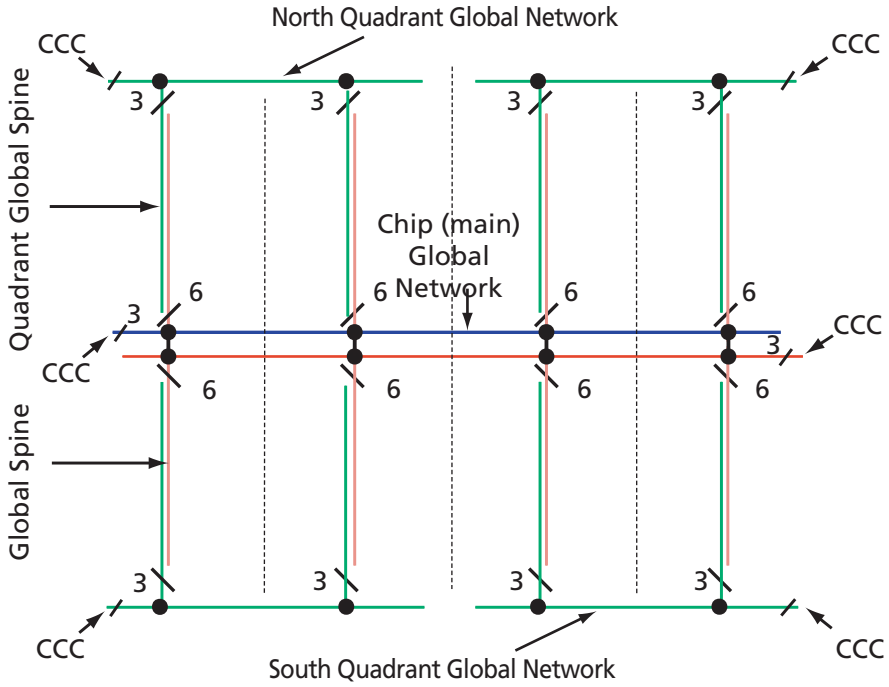


Figure 2-3 • **Simplified VersaNet Global Network (60 k gates and above)**

Spine Architecture

The low-power flash device architecture allows the VersaNet global networks to be segmented. Each of these networks contains spines (the vertical branches of the global network tree) and ribs that can reach all the VersaTiles inside its region. The nine spines available in a vertical column reside in global networks with two separate regions of scope: the quadrant global network, which has three spines, and the chip (main) global network, which has six spines. Note that the number of quadrant globals and globals/spines per tree varies depending on the specific device. Refer to Table 2-2 on page 2-6 for the clocking resources available for each device. The spines are the vertical branches of the global network tree, shown in Figure 2-3. Each spine in a vertical column of a chip (main) global network is further divided into two spine segments of equal lengths: one in the top and one in the bottom half of the die (except in 10 k through 30 k gate devices).

Top and bottom spine segments radiating from the center of a device have the same height. However, just as in the ProASIC^{PLUS}® family, signals assigned only to the top and bottom spine cannot access the middle two rows of the die. The spines for quadrant clock networks do not cross the middle of the die and cannot access the middle two rows of the architecture.

Each spine and its associated ribs cover a certain area of the device (the "scope" of the spine; see Figure 2-3). Each spine is accessed by the dedicated global network MUX tree architecture, which defines how a particular spine is driven—either by the signal on the global network from a CCC, for example, or by another net defined by the user. Details of the chip (main) global network spine-selection MUX are presented in Figure 2-5 on page 2-8. The spine drivers for each spine are located in the middle of the die.

Quadrant spines can be driven from user I/Os on the north and south sides of the die. The ability to drive spines in the quadrant global networks can have a significant effect on system performance for high-fanout inputs to a design. Access to the top quadrant spine regions is from the top of the die, and access to the bottom quadrant spine regions is from the bottom of the die. The A3PE3000 device has 28 clock trees and each tree has nine spines; this flexible global network architecture enables users to map up to 252 different internal/external clocks in an A3PE3000 device.

Table 2-2 • **Globals/Spines/Rows for IGLOO and ProASIC3 Devices**

ProASIC3/ ProASIC3L Devices	IGLOO Devices	Chip Globals	Quadrant Globals (4×3)	Clock Trees	Globals/ Spines per Tree	Total Spines per Device	VersaTiles in Each Tree	Total VersaTiles	Rows in Each Spine
A3PN010	AGLN010	4	0	1	0	0	260	260	4
A3PN015	AGLN015	4	0	1	0	0	384	384	6
A3PN020	AGLN020	4	0	1	0	0	520	520	6
A3PN060	AGLN060	6	12	4	9	36	384	1,536	12
A3PN125	AGLN125	6	12	8	9	72	384	3,072	12
A3PN250	AGLN250	6	12	8	9	72	768	6,144	24
A3P015	AGL015	6	0	1	9	9	384	384	12
A3P030	AGL030	6	0	2	9	18	384	768	12
A3P060	AGL060	6	12	4	9	36	384	1,536	12
A3P125	AGL125	6	12	8	9	72	384	3,072	12
A3P250/L	AGL250	6	12	8	9	72	768	6,144	24
A3P400	AGL400	6	12	12	9	108	768	9,216	24
A3P600/L	AGL600	6	12	12	9	108	1,152	13,824	36
A3P1000/L	AGL1000	6	12	16	9	144	1,536	24,576	48
A3PE600/L	AGLE600	6	12	12	9	108	1,120	13,440	35
A3PE1500		6	12	20	9	180	1,888	37,760	59
A3PE3000/L	AGLE3000	6	12	28	9	252	2,656	74,368	83

Table 2-3 • **Globals/Spines/Rows for IGLOO PLUS Devices**

IGLOO PLUS Devices	Chip Globals	Quadrant Globals (4×3)	Clock Trees	Globals/ Spines per Tree	Total Spines per Device	VersaTiles in Each Tree	Total VersaTiles	Rows in Each Spine
AGLP030	6	0	2	9	18	384*	792	12
AGLP060	6	12	4	9	36	384*	1,584	12
AGLP125	6	12	8	9	72	384*	3,120	12

*Note: *Clock trees that are located at far left and far right will support more VersaTiles.*

Table 2-4 • **Globals/Spines/Rows for Fusion Devices**

Fusion Device	Chip Globals	Quadrant Globals (4×3)	Clock Trees	Globals/ Spines per Tree	Total Spines per Device	VersaTiles in Each Tree	Total VersaTiles	Rows in Each Spine
AFS090	6	12	6	9	54	384	2,304	12
AFS250	6	12	8	9	72	768	6,144	24
AFS600	6	12	12	9	108	1,152	13,824	36
AFS1500	6	12	20	9	180	1,920	38,400	60

Spine Access

The physical location of each spine is identified by the letter 'T' (top) or 'B' (bottom) and an accompanying number (T*n* or B*n*). The number *n* indicates the horizontal location of the spine; 1 refers to the first spine on the left side of the die. Since there are six chip spines in each spine tree, there are up to six spines available for each combination of 'T' (or 'B') and *n* (for example, six T1 spines). Similarly, there are three quadrant spines available for each combination of 'T' (or 'B') and *n* (for example, four T1 spines), as shown in Figure 2-4.

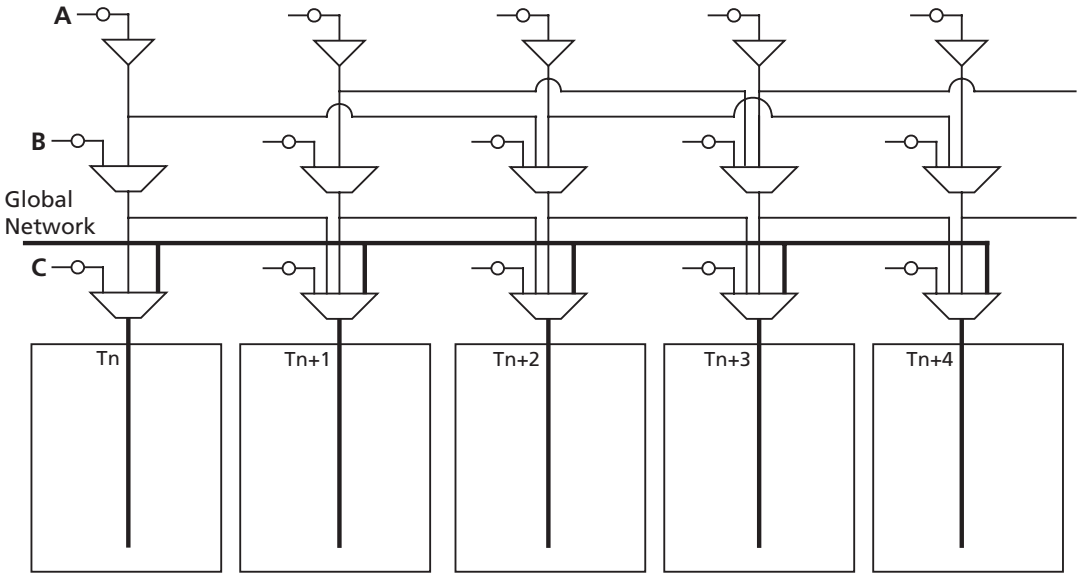


Figure 2-4 • **Chip Global Aggregation**

Spines are also called local clocks, and are accessed by the dedicated global MUX architecture. These MUXes define how a particular spine is driven. Refer to Figure 2-5 on page 2-8 for the global MUX architecture. The MUXes for each chip global spine are located in the middle of the die. Access to the top and bottom chip global spine is available from the middle of the die. There is no control dependency between the top and bottom spines. If a top spine, T1, of a chip global network is assigned to a net, B1 is not wasted and can be used by the global clock network. The signal assigned only to the top or bottom spine cannot access the middle two rows of the architecture. However, if a spine is using the top and bottom at the same time (T1 and B1, for instance), the previous restriction is lifted.

The MUXes for each quadrant global spine are located in the north and south sides of the die. Access to the top and bottom quadrant global spines is available from the north and south sides of the die. Since the MUXes for quadrant spines are located in the north and south sides of the die, you should not try to drive T1 and B1 quadrant spines from the same signal.

Using Clock Aggregation

Clock aggregation allows for multi-spine clock domains to be assigned using hardwired connections, without adding any extra skew. A MUX tree, shown in Figure 2-5, provides the necessary flexibility to allow long lines, local resources, or I/Os to access domains of one, two, or four global spines. Signal access to the clock aggregation system is achieved through long-line resources in the central rib in the center of the die, and also through local resources in the north and south ribs, allowing I/Os to feed directly into the clock system. As Figure 2-6 indicates, this access system is contiguous.

There is no break in the middle of the chip for the north and south I/O VersaNet access. This is different from the quadrant clocks located in these ribs, which only reach the middle of the rib.

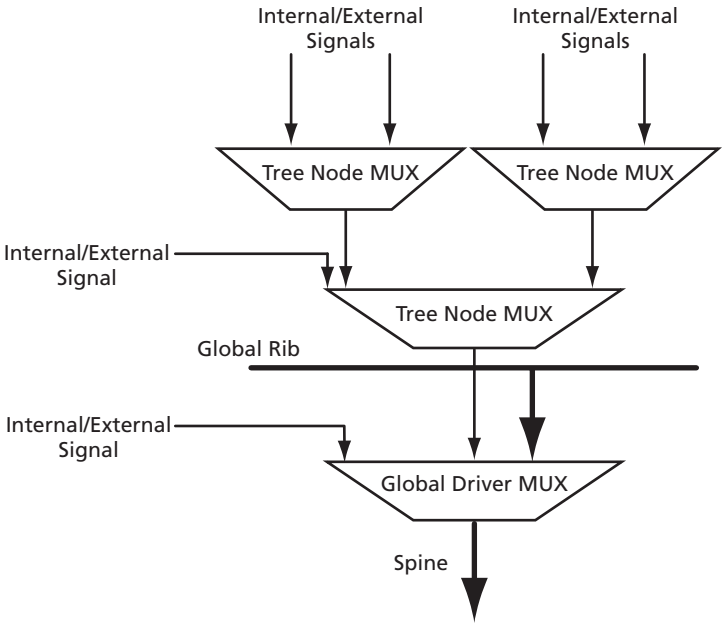


Figure 2-5 • **Spine Selection MUX of Global Tree**

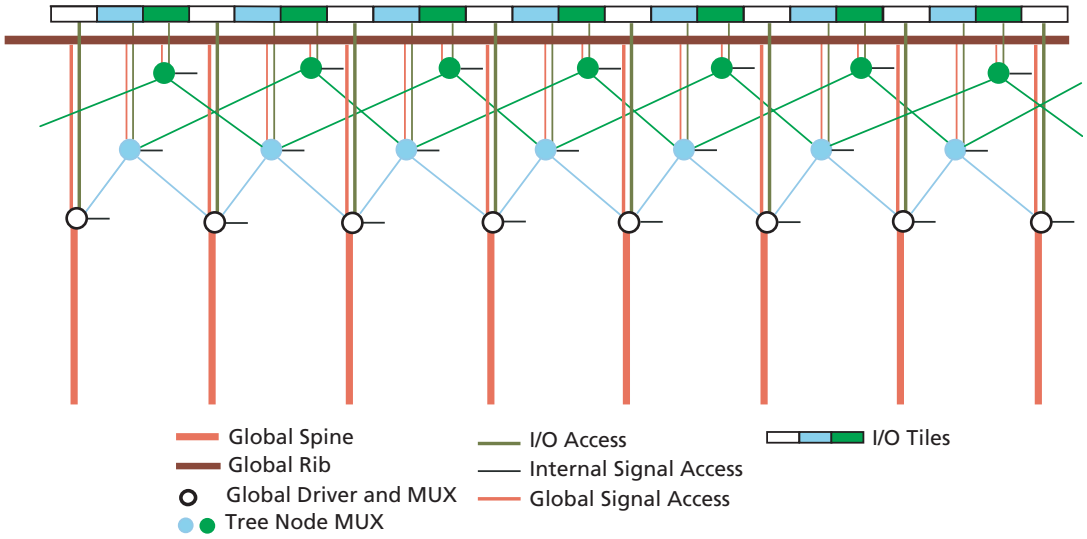


Figure 2-6 • **Clock Aggregation Tree Architecture**

Clock Aggregation Architecture

This clock aggregation feature allows a balanced clock tree, which improves clock skew. The physical regions for clock aggregation are defined from left to right and shift by one spine. For chip global networks, there are three types of clock aggregation available, as shown in Figure 2-7:

- Long lines that can drive up to four adjacent spines
- Long lines that can drive up to two adjacent spines
- Long lines that can drive one spine

There are three types of clock aggregation available for the quadrant spines, as shown in Figure 2-7:

- I/Os or local resources that can drive up to four adjacent spines
- I/Os or local resources that can drive up to two adjacent spines
- I/Os or local resources that can drive one spine
- As an example, A3PE600 and AFS600 devices have twelve spine locations: T1, T2, T3, T4, T5, T6, B1, B2, B3, B4, B5, and B6. Table 2-5 shows the clock aggregation you can have in A3PE600 and AFS600.

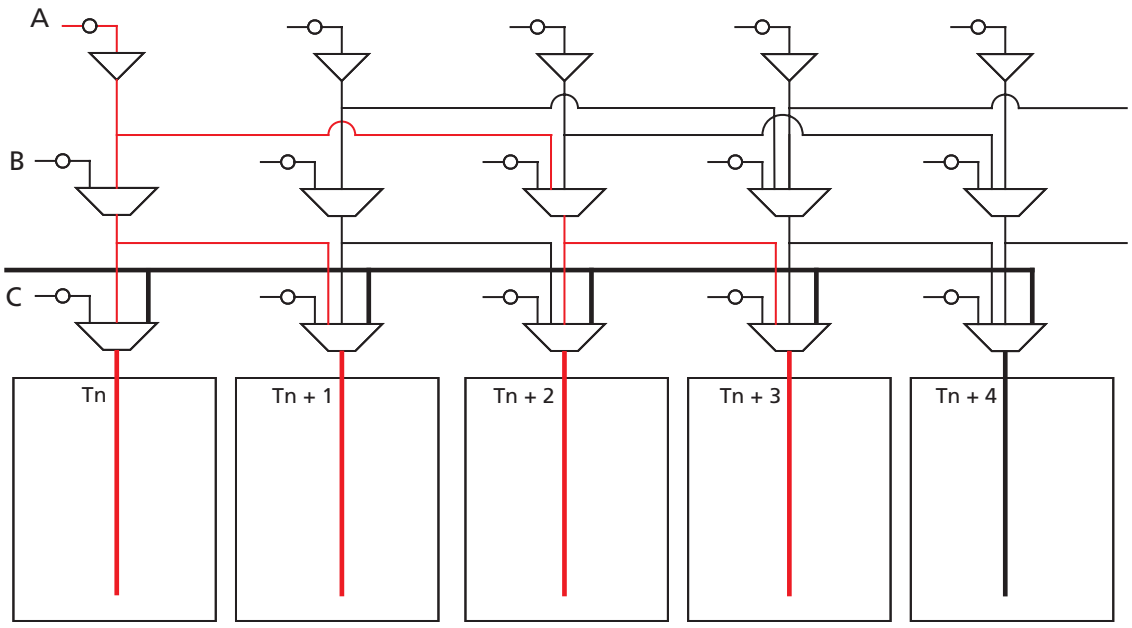


Figure 2-7 • **Four Spines Aggregation**

Table 2-5 • **Spine Aggregation in A3PE600 or AFS600**

Clock Aggregation	Spine
1 spine	T1, T2, T3, T4, T5, T6, B1, B2, B3, B4, B5, B6
2 spines	T1:T2, T2:T3, T3:T4, T4:T5, T5:T6, B1:B2, B2:B3, B3:B4, B4:B5, B5:B6
4 spines	B1:B4, B2:B5, B3:B6, T1:T4, T2:T5, T3:T6

The clock aggregation for the quadrant spines can cross over from the left to right quadrant, but not from top to bottom. The quadrant spine assignment T1:T4 is legal, but the quadrant spine assignment T1:B1 is not legal. Note that this clock aggregation is hardwired. You can always assign signals to spine T1 and B2 by instantiating a buffer, but this may add skew in the signal.

I/O Banks and Global I/Os

The following sections give an overview of naming conventions and other related I/O information.

Naming of Global I/Os

In low-power flash devices, the global I/Os have access to certain clock conditioning circuitry and have direct access to the global network. Additionally, the global I/Os can be used as regular I/Os, since they have identical capabilities to those of regular I/Os. Due to the comprehensive and flexible nature of the I/Os in low-power flash devices, a naming scheme is used to show the details of the I/O. The global I/O uses the generic name Gmn/IOuxwByVz. Refer to the I/O Structure section of the handbook for the device that you are using for more information on this naming convention.

Figure 2-8 represents the global input pins connection to the northwest CCC or northwest quadrant global networks for a low-power flash device. Each global buffer, as well as the PLL reference clock, can be driven from one of the following:

- 3 dedicated single-ended I/Os using a hardwired connection
- 2 dedicated differential I/Os using a hardwired connection
- The FPGA core

Since each bank can have a different I/O standard, the user should be careful to choose the correct global I/O for the design. There are 54 global pins available to access 18 global networks. For the single-ended and voltage-referenced I/O standards, you can use any of these three available I/Os to access the global network. For differential I/O standards such as LVDS and LVPECL, the I/O macro needs to be placed on GAA0 and GAA1 or a similar location. The unassigned global I/Os can be used as regular I/Os. Note that pin names starting with GF and GC are associated with the chip global networks, and GA, GB, GD, and GE are used for quadrant global networks.

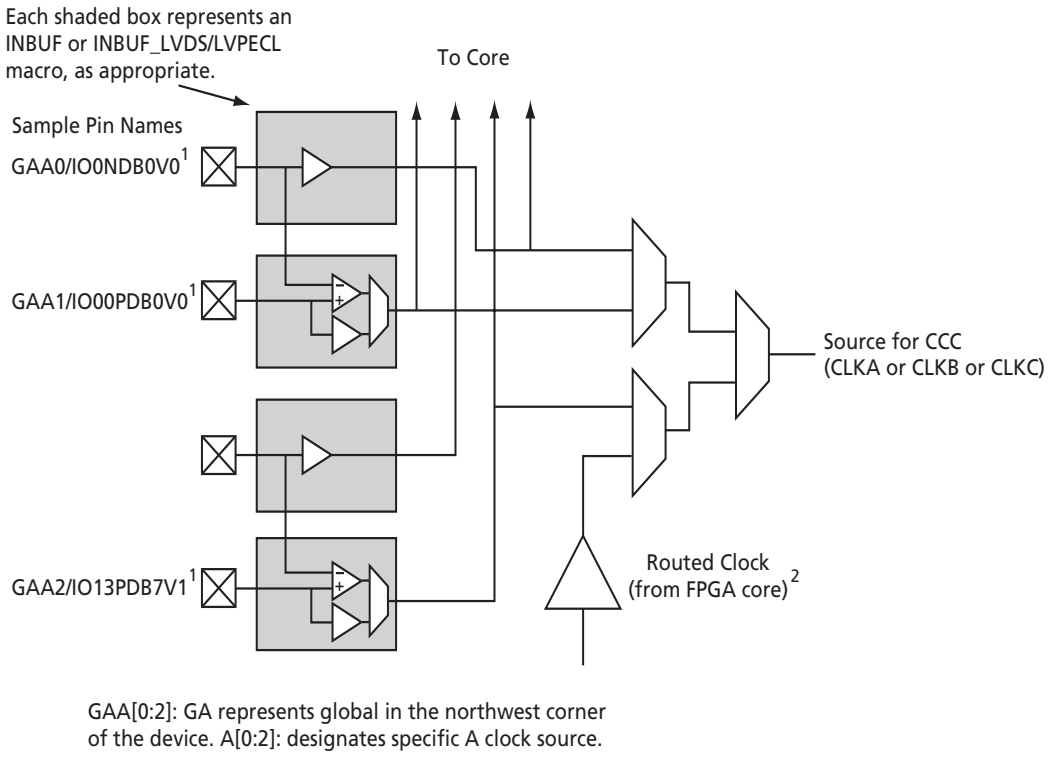


Figure 2-8 • **Global I/O Overview**

Unused Global I/O Configuration

The unused clock inputs behave similarly to the unused Pro I/Os. The Actel Designer software automatically configures the unused global pins as inputs with pull-up resistors if they are not used as regular I/O.

I/O Banks and Global I/O Standards

In low-power flash devices, any I/O or internal logic can be used to drive the global network. However, only the global macro placed at the global pins will use the hardwired connection between the I/O and global network. Global signal (signal driving a global macro) assignment to I/O banks is no different from regular I/O assignment to I/O banks with the exception that you are limited to the pin placement location available. Only global signals compatible with both the V_{CCI} and V_{REF} standards can be assigned to the same bank.

Design Recommendations

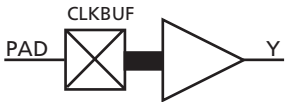
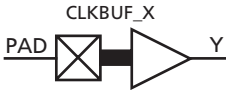
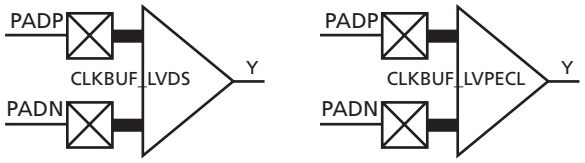
The following sections provide design flow recommendations for using a global network in a design.

- "Global Macros and I/O Standards"
- "Using Global Macros in Synplicity" on page 2-13
- "Global Promotion and Demotion Using PDC" on page 2-14
- "Spine Assignment" on page 2-15
- "Designer Flow for Global Assignment" on page 2-16
- "Simple Design Example" on page 2-18
- "Global Management in PLL Design" on page 2-20
- "Using Spines of Occupied Global Networks" on page 2-21

Global Macros and I/O Standards

Low-power flash devices have six chip global networks and four quadrant clock networks. However, the same clock macros are used for assigning signals to chip globals and quadrant globals. Depending on the clock macro placement or assignment in the Physical Design Constraint (PDC) file or MultiView Navigator (MVN), the signal will use the chip global network or quadrant network. Table 2-6 on page 2-12 lists the clock macros available for low-power flash devices. Refer to the *IGLOO, Fusion and ProASIC3 Macro Library Guide* for details.

Table 2-6 • **Clock Macros**

Macro Name	Description	Symbol
CLKBUF	Input macro for Clock Network	
CLKBUF_x	Input macro for Clock Network with specific I/O standard	
CLKBUF_LVDS/ LVPECL	LVDS or LVPECL input macro for Clock Network	
CLKINT	Internal clock interface	
CLKBIBUF	Bidirectional macro with input dedicated to routed Clock Network	

Use these available macros to assign a signal to the global network. In addition to these global macros, PLL and CLKDLY macros can also drive the global networks. Use I/O–standard–specific clock macros (CLKBUF_x) to instantiate a specific I/O standard for the global signals. Table 2-7 shows the list of these I/O–standard–specific macros. Note that if you use these I/O–standard–specific clock macros, you cannot change the I/O standard later in the design stage. If you use the regular CLKBUF macro, you can use MVN or the PDC file in Designer to change the I/O standard. The default I/O standard for CLKBUF is LVTTL in the current Actel Libero® Integrated Design Environment (IDE) and Designer software.

Table 2-7 • **I/O Standards within CLKBUF**

Name	Description
CLKBUF_LVCMOS5	LVCMOS clock buffer with 5.0 V CMOS voltage level
CLKBUF_LVCMOS33	LVCMOS clock buffer with 3.3 V CMOS voltage level
CLKBUF_LVCMOS25	LVCMOS clock buffer with 2.5 V CMOS voltage level[1]
CLKBUF_LVCMOS18	LVCMOS clock buffer with 1.8 V CMOS voltage level
CLKBUF_LVCMOS15	LVCMOS clock buffer with 1.5 V CMOS voltage level
CLKBUF_LVCMOS12	LVCMOS clock buffer with 1.2 V CMOS voltage level
CLKBUF_PCI	PCI clock buffer
CLKBUF_PCIX	PCIX clock buffer
CLKBUF_GTL25	GTL clock buffer with 2.5 V CMOS voltage level[1]
CLKBUF_GTL33	GTL clock buffer with 3.3 V CMOS voltage level[1]

Notes:

1. *Supported in only the IGLOOe, ProASIC3E, AFS600, and AFS1500 devices*
2. *By default, the CLKBUF macro uses the 3.3 V LVTTL I/O technology.*

Table 2-7 • **I/O Standards within CLKBUF (continued)**

Name	Description
CLKBUF_GTLP25	GTL+ clock buffer with 2.5 V CMOS voltage level[1]
CLKBUF_GTLP33	GTL+ clock buffer with 3.3 V CMOS voltage level[1]
CLKBUF_ HSTL _I	HSTL Class I clock buffer[1]
CLKBUF_ HSTL _II	HSTL Class II clock buffer[1]
CLKBUF_SSTL2_I	SSTL2 Class I clock buffer[1]
CLKBUF_SSTL2_II	SSTL2 Class II clock buffer[1]
CLKBUF_SSTL3_I	SSTL3 Class I clock buffer[1]
CLKBUF_SSTL3_II	SSTL3 Class II clock buffer[1]

Notes:

1. *Supported in only the IGLOOe, ProASIC3E, AFS600, and AFS1500 devices*
2. *By default, the CLKBUF macro uses the 3.3 V LVTTL I/O technology.*

The current synthesis tool libraries only infer the CLKBUF or CLKINT macros in the netlist. All other global macros must be instantiated manually into your HDL code. The following is an example of CLKBUF_LVCMOS25 global macro instantiations that you can copy and paste into your code:

VHDL

```
component clkbuf_lvcmos25
  port (pad : in std_logic; y : out std_logic);
end component

begin
-- concurrent statements
u2 : clkbuf_lvcmos25 port map (pad =>  ext_clk, y => int_clk);
end
```

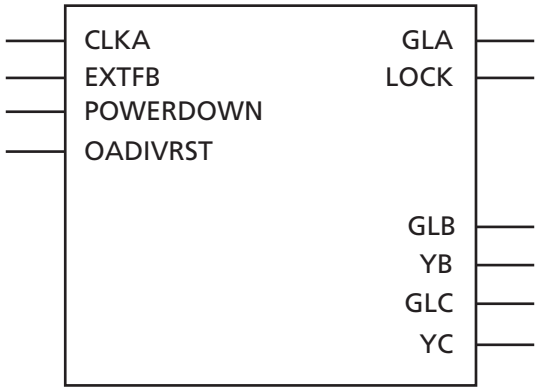
Verilog

```
module design (______);

input _____;
output ______;

clkbuf_lvcmos25 u2 (.y(int_clk), .pad(ext_clk);

endmodule
```

Using Global Macros in Synplicity

The Synplify® synthesis tool automatically inserts global buffers for nets with high fanout during synthesis. By default, Synplicity® puts six global macros (CLKBUF or CLKINT) in the netlist, including any global instantiation or PLL macro. Synplify always honors your global macro instantiation. If you have a PLL (only primary output is used) in the design, Synplify adds five more global buffers in the netlist. Synplify uses the following global counting rule to add global macros in the netlist:

1. CLKBUF: 1 global buffer
2. CLKINT: 1 global buffer
3. CLKDLY: 1 global buffer
4. PLL: 1 to 3 global buffers
   - GLA, GLB, GLC, YB, and YC are counted as 1 buffer.
   - GLB or YB is used or both are counted as 1 buffer.
   - GLC or YC is used or both are counted as 1 buffer.



Note: OAVDIVRST exists only in the Fusion PLL.

Figure 2-9 • **PLLs in Low-Power Flash Devices**

You can use the syn_global_buffers attribute in Synplify to specify a maximum number of global macros to be inserted in the netlist. This can also be used to restrict the number of global buffers inserted. In the Synplicity 8.1 version, a new attribute, syn_global_minfanout, has been added for low-power flash devices. This enables you to promote only the high-fanout signal to global. However, be aware that you can only have six signals assigned to chip global networks, and the rest of the global signals should be assigned to quadrant global networks. So, if the netlist has 18 global macros, the remaining 12 global macros should have fanout that allows the instances driven by these globals to be placed inside a quadrant.

Global Promotion and Demotion Using PDC

The HDL source file or schematic is the preferred place for defining which signals should be assigned to a clock network using clock macro instantiation. This method is preferred because it is guaranteed to be honored by the synthesis tools and Designer software and stop any replication on this net by the synthesis tool. Note that a signal with fanout may have logic replication if it is not promoted to global during synthesis. In that case, the user cannot promote that signal to global using PDC. See Synplicity Help for details on using this attribute. To help you with global management, Designer allows you to promote a signal to a global network or demote a global macro to a regular macro from the user netlist using the compile options and/or PDC commands.

The following are the PDC constraints you can use to promote a signal to a global network:

1. PDC syntax to promote a regular net to a chip global clock:

   ```
   assign_global_clock –net netname
   ```

   The following will happen during promotion of a regular signal to a global network:
   - If the net is external, the net will be driven by a CLKINT inserted automatically by Compile.
   - The I/O macro will not be changed to CLKBUF macros.
   - If the net is an internal net, the net will be driven by a CLKINT inserted automatically by Compile.
2. PDC syntax to promote a net to a quadrant clock:

   ```
   assign_local_clock –net netname –type quadrant UR|UL|LR|LL
   ```

   This follows the same rule as the chip global clock network.

The following PDC command demotes the clock nets to regular nets.

```
unassign_global_clock -net netname
```

The following will happen during demotion of a global signal to regular nets:

- CLKBUF_x becomes INBUF_x; CLKINT is removed from the netlist.
- The essential global macro, such as the output of the Clock Conditioning Circuit, cannot be demoted.
- No automatic buffering will happen.

Since no automatic buffering happens when a signal is demoted, this net may have a high delay due to large fanout. This may have a negative effect on the quality of the results. Actel recommends that the automatic global demotion only be used on small-fanout nets. Use clock networks for high-fanout nets to improve timing and routability.

Spine Assignment

The low-power flash device architecture allows the global networks to be segmented and used as clock spines. These spines, also called local clocks, enable the use of PDC or MVN to assign a signal to a spine.

PDC syntax to promote a net to a spine/local clock:

```
assign_local_clock –net netname –type [quadrant|chip] Tn|Bn|Tn:Bm
```

If the net is driven by a clock macro, Designer automatically demotes the clock net to a regular net before it is assigned to a spine. Nets driven by a PLL or CLKDLY macro cannot be assigned to a local clock.

When assigning a signal to a spine or quadrant global network using PDC (pre-compile), the Designer software will legalize the shared instances. The number of shared instances to be legalized can be controlled by compile options. If these networks are created in MVN (only quadrant globals can be created), no legalization is done (as it is post-compile). Designer does not do legalization between non-clock nets.

As an example, consider two nets, net_clk and net_reset, driving the same flip-flop. The following PDC constraints are used:

```
assign_local_clock –net net_clk –type chip T3
assign_local_clock –net net_reset –type chip T1:T2
```

During Compile, Designer adds a buffer in the reset net and places it in the T1 or T2 region, and places the flip-flop in the T3 spine region (Figure 2-10).

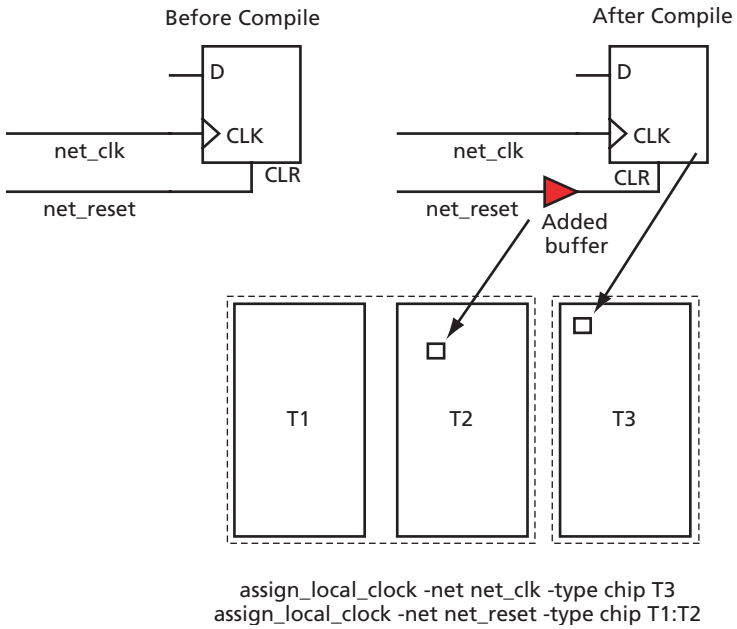


Figure 2-10 • **Adding a Buffer for Shared Instances**

You can control the maximum number of shared instances allowed for the legalization to take place using the Compile Option dialog box shown in Figure 2-11. Refer to Libero IDE / Designer online help for details on the Compile Option dialog box. A large number of shared instances most likely indicates a floorplanning problem that you should address.

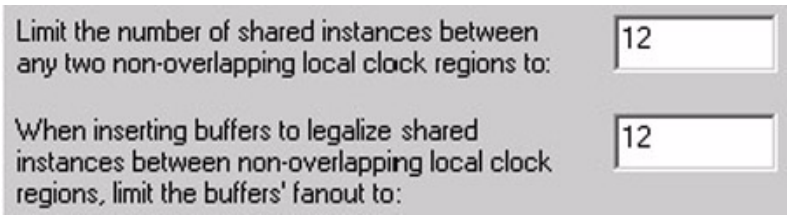


Figure 2-11 • **Shared Instances in the Compile Option Dialog Box**

Designer Flow for Global Assignment

To achieve the desired result, pay special attention to global management during synthesis and place-and-route. The current Synplify tool does not insert more than six global buffers in the netlist by default. Thus, the default flow will not assign any signal to the quadrant global network. However, you can use attributes in Synplify and increase the default global macro assignment in the netlist. Designer v6.2 supports automatic quadrant global assignment, which was not available in Designer v6.1. Layout will make the choice to assign the correct signals to global. However, you can also utilize PDC and perform manual global assignment to overwrite any automatic assignment. The following step-by-step suggestions guide you in the layout of your design and help you improve timing in Designer:

1. Run Compile and check the Compile report. The Compile report has global information in the "Device Utilization" section that describes the number of chip and quadrant signals in the design. A "Net Report" section describes chip global nets, quadrant global nets, local clock nets, a list of nets listed by fanout, and net candidates for local clock assignment. Review this information. Note that YB or YC are counted as global only when they are used in isolation; if you use YB only and not GLB, this net is not shown in the global/quadrant nets report. Instead, it appears in the Global Utilization report.
2. If some signals have a very high fanout and are candidates for global promotion, promote those signals to global using the compile options or PDC commands. Figure 2-12 on page 2-17 shows the Globals Management section of the compile options. Select **Promote regular nets whose fanout is greater than** and enter a reasonable value for fanouts.

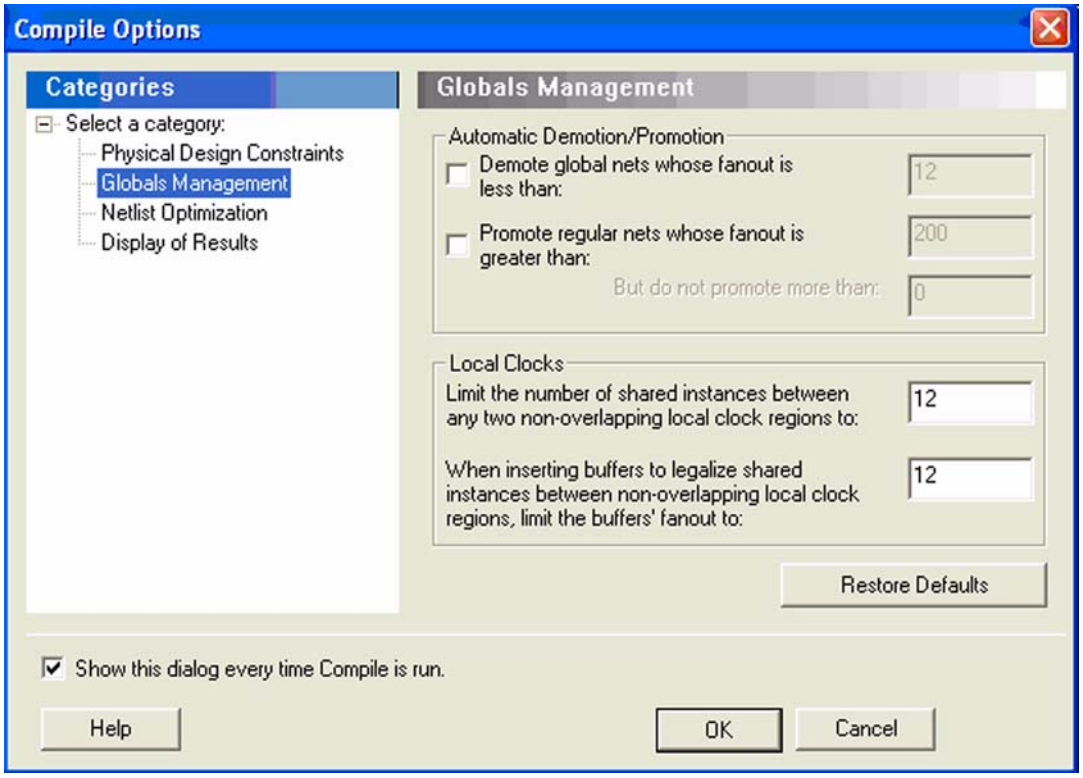


Figure 2-12 • **Globals Management GUI in Designer**

3. Occasionally, the synthesis tool assigns a global macro to clock nets, even though the fanout is significantly less than other asynchronous signals. Select **Demote global nets whose fanout is less than** and enter a reasonable value for fanouts. This frees up some global networks from the signals that have very low fanouts. This can also be done using PDC.
4. Use local clocks for the signals that do not need to go to the whole chip but should have low skew. This local clocks assignment can only be done using PDC.
5. Assign the I/O buffer using MVN if you have fixed I/O assignment. As shown in Figure 2-7 on page 2-9, there are three sets of global pins that have a hardwired connection to each global network. Do not try to put multiple CLKBUF macros in these three sets of global pins. For example, do not assign two CLKBUFs to GAA0x and GAA2x pins.
6. You must click **Commit** at the end of MVN assignment. This runs the pre-layout checker and checks the validity of global assignment.
7. Always run Compile with the **Keep existing physical constraints** option on. This uses the quadrant clock network assignment in the MVN assignment and checks if you have the desired signals on the global networks.
8. Run Layout and check the timing.

Simple Design Example

Consider a design consisting of six building blocks (shift registers) and targeted for an A3PE600-PQ208 (Figure 2-10 on page 2-15). The example design consists of two PLLs (PLL1 has GLA only; PLL2 has both GLA and GLB), a global reset (ACLR), an enable (EN_ALL), and three external clock domains (QCLK1, QCLK2, and QCLK3) driving the different blocks of the design. Note that the PQ208 package only has two PLLs (which access the chip global network). Because of fanout, the global reset and enable signals need to be assigned to the chip global resources. There is only one free chip global for the remaining global (QCLK1, QCLK2, QCLK3). Place two of these signals on the quadrant global resource. The design example demonstrates manually assignment of QCLK1 and QCLK2 to the quadrant global using the PDC command.

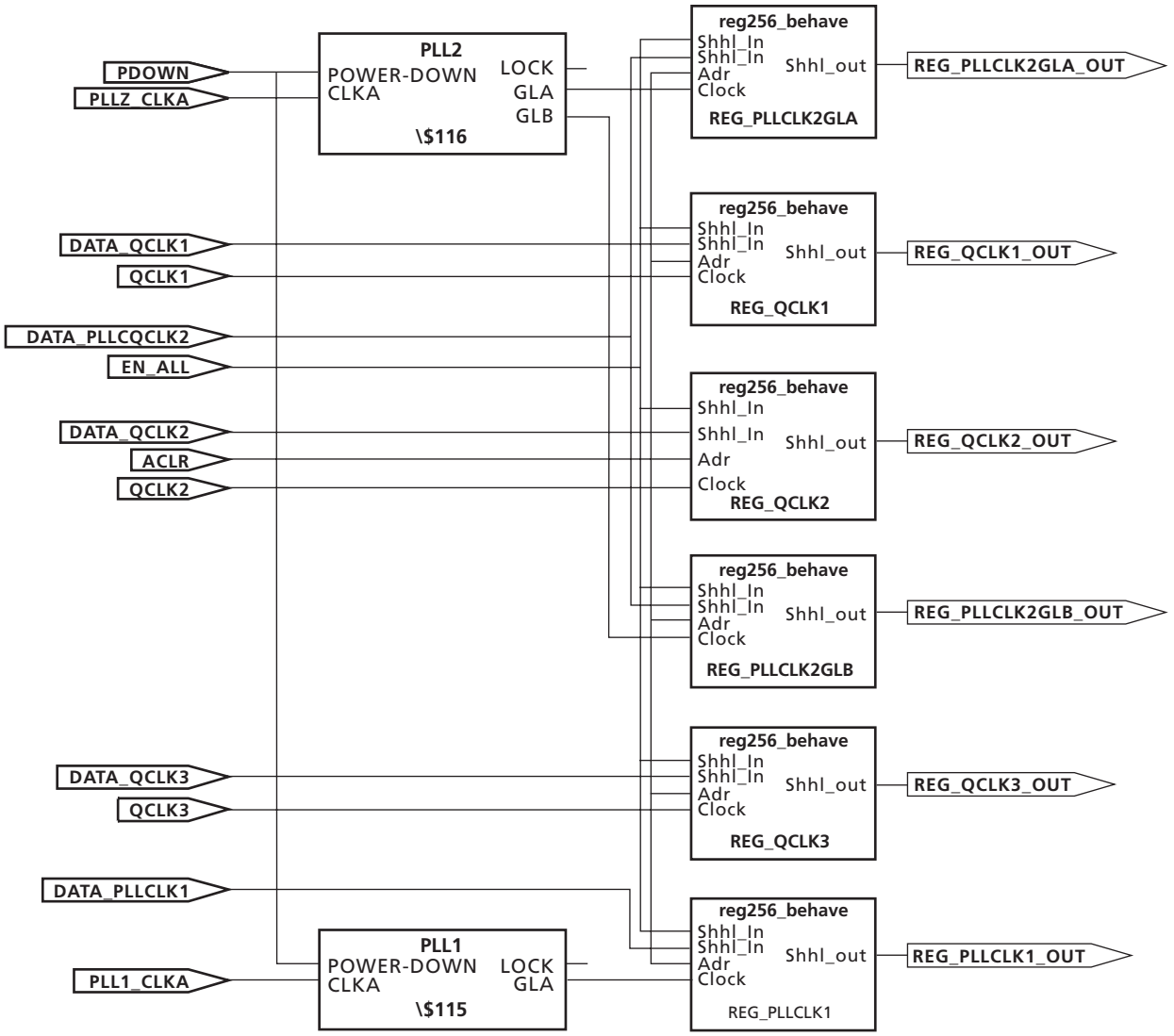


Figure 2-13 • **Block Diagram of the Global Management Example Design**

Step 1

Run Synthesis with default options. The Synplicity log shows the following device utilization:

Cell usage:

	cell count	area	count*area
DFN1E1C1	1536	2.0	3072.0
BUFF	278	1.0	278.0
INBUF	10	0.0	0.0
VCC	9	0.0	0.0
GND	9	0.0	0.0
OUTBUF	6	0.0	0.0
CLKBUF	3	0.0	0.0
PLL	2	0.0	0.0
TOTAL	1853		3350.0

Step 2

Run Compile with the **Promote regular nets whose fanout is greater than** option selected in Designer; you will see the following in the Compile report:

```
Device utilization report:
==========================
CORE               Used:   1536  Total:  13824   (11.11%)
IO (W/ clocks)     Used:     19  Total:    147   (12.93%)
Differential IO    Used:      0  Total:     65    (0.00%)
GLOBAL             Used:      8  Total:     18   (44.44%)
PLL                Used:      2  Total:      2  (100.00%)
RAM/FIFO           Used:      0  Total:     24    (0.00%)
FlashROM           Used:      0  Total:      1    (0.00%)
.....................
The following nets have been assigned to a global resource:
Fanout  Type           Name
-------------------------
1536    INT_NET        Net    : EN_ALL_c
                       Driver: EN_ALL_pad_CLKINT
                       Source: AUTO PROMOTED
1536    SET/RESET_NET  Net    : ACLR_c
                       Driver: ACLR_pad_CLKINT
                       Source: AUTO PROMOTED
256     CLK_NET        Net    : QCLK1_c
                       Driver: QCLK1_pad_CLKINT
                       Source: AUTO PROMOTED
256     CLK_NET        Net    : QCLK2_c
                       Driver: QCLK2_pad_CLKINT
                       Source: AUTO PROMOTED
256     CLK_NET        Net    : QCLK3_c
                       Driver: QCLK3_pad_CLKINT
                       Source: AUTO PROMOTED
256     CLK_NET        Net    : $1N14
                       Driver: $1I5/Core
                       Source: ESSENTIAL
256     CLK_NET        Net    : $1N12
                       Driver: $1I6/Core
                       Source: ESSENTIAL
256     CLK_NET        Net    : $1N10
                       Driver: $1I6/Core
                       Source: ESSENTIAL
```

Designer will promote five more signals to global due to high fanout. There are eight signals assigned to global networks.

During Layout, Designer will assign two of the signals to quadrant global locations.

Step 3 (optional)

You can also assign the QCLK1_c and QCLK2_c nets to quadrant regions using the following PDC commands:

```
assign_local_clock –net QCLK1_c  –type quadrant UL
assign_local_clock –net QCLK2_c  –type quadrant LL
```

Step 4

Import this PDC with the netlist and run Compile again. You will see the following in the Compile report:

```
The following nets have been assigned to a global resource:
Fanout  Type          Name
--------------------------
1536    INT_NET       Net   : EN_ALL_c
                      Driver: EN_ALL_pad_CLKINT
                      Source: AUTO PROMOTED
1536    SET/RESET_NET Net   : ACLR_c
                      Driver: ACLR_pad_CLKINT
                      Source: AUTO PROMOTED
256     CLK_NET       Net   : QCLK3_c
                      Driver: QCLK3_pad_CLKINT
                      Source: AUTO PROMOTED
256     CLK_NET       Net   : $1N14
                      Driver: $1I5/Core
                      Source: ESSENTIAL
256     CLK_NET       Net   : $1N12
                      Driver: $1I6/Core
                      Source: ESSENTIAL
256     CLK_NET       Net   : $1N10
                      Driver: $1I6/Core
                      Source: ESSENTIAL
The following nets have been assigned to a quadrant clock resource using PDC:
Fanout  Type          Name
--------------------------
256     CLK_NET       Net   : QCLK1_c
                      Driver: QCLK1_pad_CLKINT
                      Region: quadrant_UL
256     CLK_NET       Net   : QCLK2_c
                      Driver: QCLK2_pad_CLKINT
                      Region: quadrant_LL
```

Step 5

Run Layout.

Global Management in PLL Design

This section describes the legal global network connections to PLLs in the low-power flash devices. For detailed information on using PLLs, refer to *Clock Conditioning Circuits in IGLOO and ProASIC3 Devices*. Actel recommends that you use the dedicated global pins to directly drive the reference clock input of the associated PLL for reduced propagation delays and clock distortion. However, low-power flash devices offer the flexibility to connect other signals to reference clock inputs. Each PLL is associated with three global networks (Figure 2-8 on page 2-10). There are some limitations, such as when trying to use the global and PLL at the same time:

- If you use a PLL with only primary output, you can still use the remaining two free global networks.
- If you use three globals associated with a PLL location, you cannot use the PLL on that location.
- If the YB or YC output is used standalone, it will occupy one global, even though this signal does not go to the global network.

Using Spines of Occupied Global Networks

When a signal is assigned to a global network, the flash switches are programmed to set the MUX select lines (explained in the "Clock Aggregation Architecture" section on page 2-9) to drive the spines of that network with the global net. However, if the global net is restricted from reaching into the scope of a spine, the MUX drivers of that spine are available for other high-fanout or critical signals (Figure 2-14).

For example, if you want to limit the CLK1_c signal to the left half of the chip and want to use the right side of the same global network for CLK2_c, you can add the following PDC commands:

```
define_region -name region1 -type inclusive 0 0 34 29
assign_net_macros region1 CLK1_c
assign_local_clock –net CLK2_c –type chip B2
```

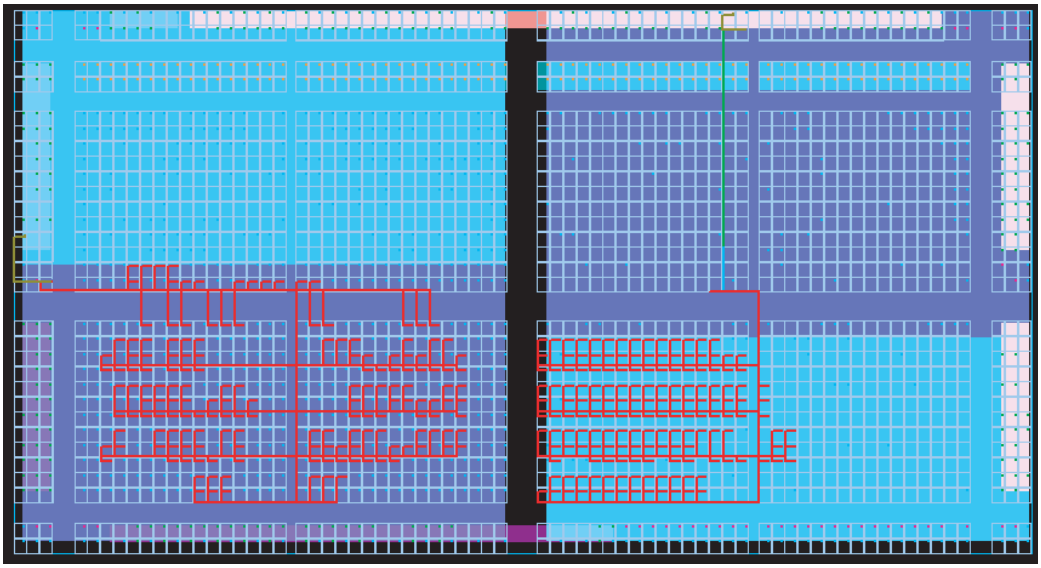


***Figure 2-14* • Design Example Using Spines of Occupied Global Networks**

Conclusion

IGLOO, Fusion, and ProASIC3 devices contain 18 global networks: 6 chip global networks and 12 quadrant global networks. These global networks can be segmented into local low-skew networks called spines. The spines provide low-skew networks for the high-fanout signals of a design. These allow you up to 252 different internal/external clocks in an A3PE3000 device. This document describes the architecture for the global network, plus guidelines and methodologies in assigning signals to globals and spines.

Related Documents

Handbook Documents

Clock Conditioning Circuits in IGLOO and ProASIC3 Devices
http://www.actel.com/LPD_CCC_HBs.pdf
I/O Structures in IGLOO PLUS Devices
http://www.actel.com/documents/IGLOOPLUS_IO_HBs.pdf
I/O Structures in IGLOO and ProASIC3 Devices
http://www.actel.com/documents/IGLOO_PA3_IO_HBs.pdf
I/O Structures in IGLOOe and ProASIC3E Devices
http://www.actel.com/documents/IGLOOe_PA3E_IO_HBs.pdf

User's Guides

IGLOO, Fusion, and ProASIC3 Macro Library Guide
http://www.actel.com/documents/pa3_libguide_ug.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-005-4
Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.4)	Page
v1.3 (October 2008)	The "Global Architecture" section was updated to include 10 k devices, and to include information about VersaNet global support for IGLOO nano devices.	2-1
	The Table 2-1 · Flash-Based FPGAs was updated to include IGLOO nano and ProASIC3 nano devices.	2-2
	The "VersaNet Global Network Distribution" section was updated to include 10 k devices and to note an exception in global lines for nano devices.	2-3
	Figure 2-2 · Simplified VersaNet Global Network (30 k gates and below) is new.	2-4
	The "Spine Architecture" section was updated to clarify support for 10 k and nano devices.	2-5
	Table 2-2 · Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated to include IGLOO nano and ProASIC3 nano devices.	2-6
	The figure in the CLKBUF_LVDS/LVPECL row of Table 2-6 · Clock Macros was updated to change CLKBIBUF to CLKBUF.	2-12
v1.2 (June 2008)	A third bullet was added to the beginning of the "Global Architecture" section: In Fusion devices, the west CCC also contains a PLL core. In the two larger devices (AFS600 and AFS1500), the west and east CCCs each contain a PLL.	2-1
	The "Global Resource Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	2-2
	Table 2-2 · Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated to include A3PE600/L in the device column.	2-6
	Table note 1 was revised in Table 2-7 · I/O Standards within CLKBUF to include AFS600 and AFS1500.	2-12
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 2-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	2-2
v1.0 (January 2008)	The "Global Architecture" section was updated to include the IGLOO PLUS family. The bullet was revised to include that the west CCC does not contain a PLL core in 15 k and 30 k devices. Instances of "A3P030 and AGL030 devices" were replaced with "15 k and 30 k gate devices."	2-1
	Table 2-1 · Flash-Based FPGAs and the accompanying text was updated to include the IGLOO PLUS family. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	2-2
	The "VersaNet Global Network Distribution" section, "Spine Architecture" section, the note in Figure 2-1 · Overview of VersaNet Global Network and Device Architecture, and the note in Figure 2-3 · Simplified VersaNet Global Network (60 k gates and above) were updated to include mention of 15 k gate devices.	2-3, 2-4

Previous Version	Changes in Current Version (v1.4)	Page
v1.0 (continued)	Table 2-2 · Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated to add the A3P015 device, and to revise the values for clock trees, globals/spines per tree, and globals/spines per device for the A3P030 and AGL030 devices.	2-6
	Table 2-3 · Globals/Spines/Rows for IGLOO PLUS Devices is new.	2-6
	CLKBUF_LVCMOS12 was added to Table 2-7 · I/O Standards within CLKBUF.	2-12
	The "Handbook Documents" section was updated to include the three different I/O Structures chapters for ProASIC3 and IGLOO device families.	2-22
51900087-1/3.05	Figure 2-3 · Simplified VersaNet Global Network (60 k gates and above) was updated.	2-4
	The "Naming of Global I/Os" section was updated.	2-10
	The "Using Global Macros in Synplicity" section was updated.	2-13
	The "Global Promotion and Demotion Using PDC" section was updated.	2-14
	The "Designer Flow for Global Assignment" section was updated.	2-16
	The "Simple Design Example" section was updated.	2-18
51900087-0/1.05	Table 2-2 · Globals/Spines/Rows for IGLOO and ProASIC3 Devices was updated.	2-6

3 – Clock Conditioning Circuits in Low-Power Flash Devices and Mixed-Signal FPGAs

Introduction

This document outlines the following device information: Clock Conditioning Circuit (CCC) features, PLL core specifications, functional descriptions, software configuration information, detailed usage information, recommended board-level considerations, and other considerations concerning clock conditioning circuits and global networks in low-power flash devices or mixed-signal FPGAs.

Overview of Clock Conditioning Circuitry

In Fusion, IGLOO,® and ProASIC®3 devices, the CCCs are used to implement frequency division, frequency multiplication, phase shifting, and delay operations. The CCCs are available in six chip locations—each of the four chip corners and the middle of the east and west chip sides. For device-specific variations, refer to the "Device-Specific Layout" section on page 3-17.

The CCC is composed of the following:

- PLL core
- 3 phase selectors
- 6 programmable delays and 1 fixed delay that advances/delays phase
- 5 programmable frequency dividers that provide frequency multiplication/division (not shown in Figure 3-5 on page 3-10 because they are automatically configured based on the user's required frequencies)
- 1 dynamic shift register that provides CCC dynamic reconfiguration capability

Figure 3-1 provides a simplified block diagram of the physical implementation of the building blocks in each of the CCCs.

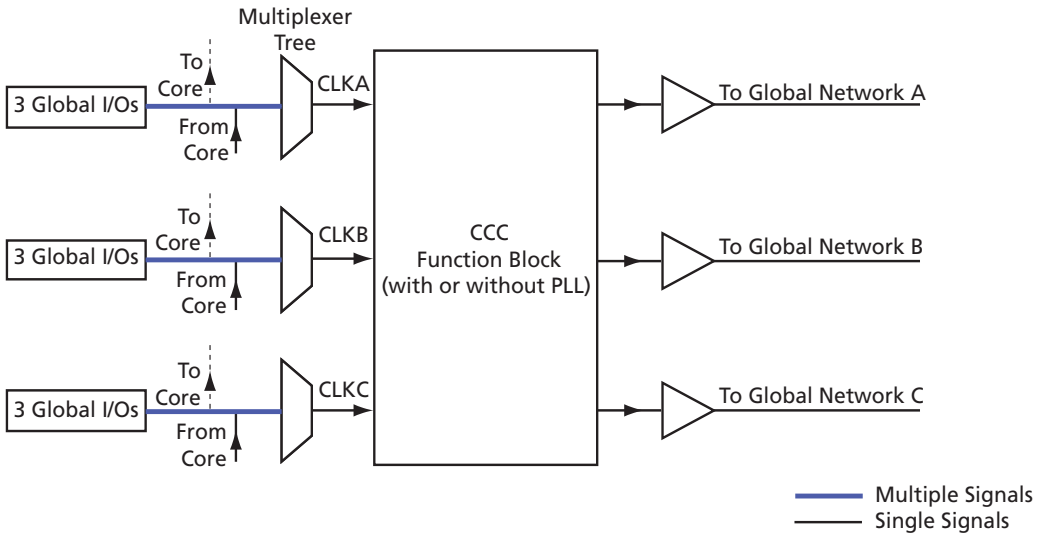


Figure 3-1 • **Overview of the CCCs Offered in Fusion, IGLOO, and ProASIC3**

Each CCC can implement up to three independent global buffers (with or without programmable delay) or a PLL function (programmable frequency division/multiplication, phase shift, and delays) with up to three global outputs. Unused global outputs of a PLL can be used to implement independent global buffers, up to a maximum of three global outputs for a given CCC.

CCC Programming

The CCC block is fully configurable, either via flash configuration bits set in the programming bitstream or through an asynchronous interface. This asynchronous dedicated shift register interface is dynamically accessible from inside the low-power flash devices to permit parameter changes, such as PLL divide ratios and delays, during device operation.

To increase the versatility and flexibility of the clock conditioning system, the CCC configuration is determined either by the user during the design process, with configuration data being stored in flash memory as part of the device programming procedure, or by writing data into a dedicated shift register during normal device operation.

This latter mode allows the user to dynamically reconfigure the CCC without the need for core programming. The shift register is accessed through a simple serial interface. Refer to *UJTAG Applications in Actel's Low-Power Flash Devices* or the application note *Using Global Resources in Actel Fusion Devices*.

Global Resources

Low-power flash and mixed-signal devices provide three global routing networks (GLA, GLB, and GLC) for each of the CCC locations. There are potentially many I/O locations; each global I/O location can be chosen from only one of three possibilities. This is controlled by the multiplexer tree circuitry in each global network. Once the I/O location is selected, the user has the option to utilize the CCCs before the signals are connected to the global networks. The CCC in each location (up to six) has the same structure, so generating the CCC macros is always done with an identical software GUI. The CCCs in the corner locations drive the quadrant global networks, and the CCCs in the middle of the east and west chip sides drive the chip global networks. The quadrant global networks span only a quarter of the device, while the chip global networks span the entire device. For more details on global resources offered in low-power flash devices, refer to *Global Resources in Actel Low-Power Flash Devices*.

A global buffer can be placed in any of the three global locations (CLKA-GLA, CLKB-GLB, or CLKC-GLC) of a given CCC. A PLL macro uses the CLKA CCC input to drive its reference clock. It uses the GLA and, optionally, the GLB and GLC global outputs to drive the global networks. A PLL macro can also drive the YB and YC regular core outputs. The GLB (or GLC) global output cannot be reused if the YB (or YC) output is used. Refer to the "PLL Macro Signal Descriptions" section on page 3-8 for more information.

Each global buffer, as well as the PLL reference clock, can be driven from one of the following:

- 3 dedicated single-ended I/Os using a hardwired connection
- 2 dedicated differential I/Os using a hardwired connection
- The FPGA core

CCC Support in Actel's Flash Devices

The flash FPGAs listed in Table 3-1 support the CCC feature and the functions described in this document.

Table 3-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
	IGLOO nano	The industry's lowest-power, smallest-size solution
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 3-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 3-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Global Buffers with No Programmable Delays

Access to the global / quadrant global networks can be configured directly from the global I/O buffer, bypassing the CCC functional block (as indicated by the dotted lines in Figure 3-1 on page 3-1). Internal signals driven by the FPGA core can use the global / quadrant global networks by connecting via the routed clock input of the multiplexer tree.

There are many specific CLKBUF macros supporting the wide variety of single-ended I/O inputs (CLKBUF) and differential I/O standards (CLKBUF_LVDS/LVPECL) in the low-power flash families. They are used when connecting global I/Os directly to the global/quadrant networks.

When an internal signal needs to be connected to the global/quadrant network, the CLKINT macro is used to connect the signal to the routed clock input of the network's MUX tree.

To utilize direct connection from global I/Os or from internal signals to the global/quadrant networks, CLKBUF, CLKBUF_LVPECL/LVDS, and CLKINT macros are used (Figure 3-2).

- The CLKBUF and CLKBUF_LVPECL/LVDS[1] macros are composite macros that include an I/O macro driving a global buffer, which uses a hardwired connection.
- The CLKBUF, CLKBUF_LVPECL/LVDS[1] and CLKINT macros are pass-through clock sources and do not use the PLL or provide any programmable delay functionality.
- The CLKINT macro provides a global buffer function driven internally by the FPGA core.

The available CLKBUF macros are described in the *IGLOO, Fusion, and ProASIC3 Macro Library Guide*.

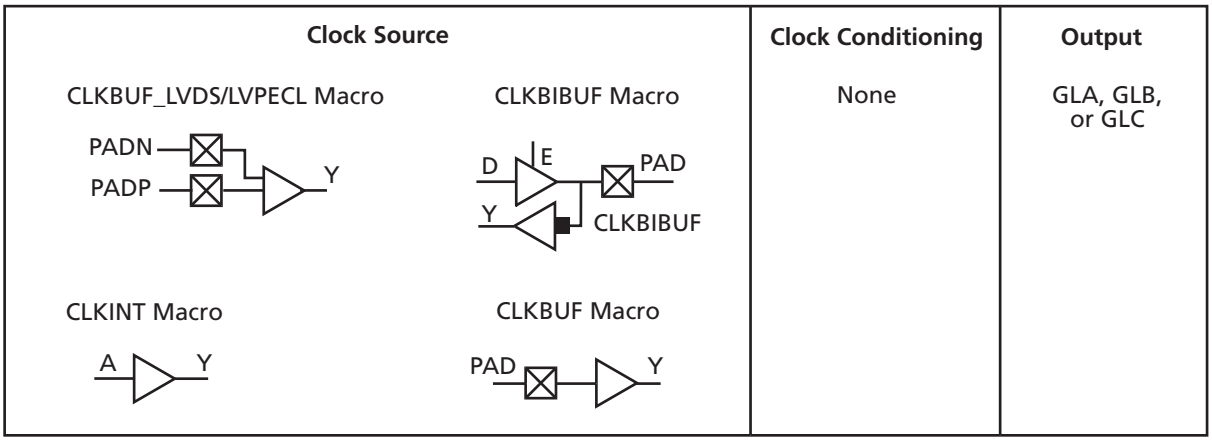


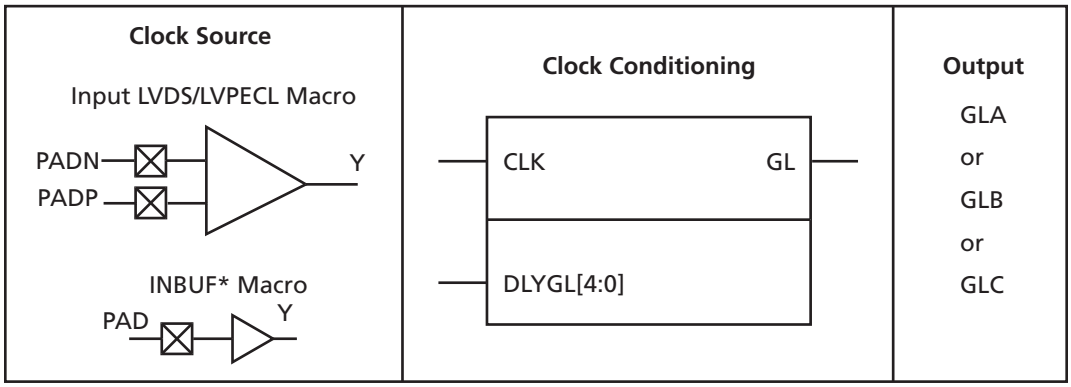
Figure 3-2 • **CCC Options: Global Buffers with No Programmable Delay**

Global Buffer with Programmable Delay

Clocks requiring clock adjustments can utilize the programmable delay cores before connecting to the global / quadrant global networks. A maximum of 18 CCC global buffers can be instantiated in a device—three per CCC and up to six CCCs per device.

Each CCC functional block contains a programmable delay element for each of the global networks (up to three), and users can utilize these features by using the corresponding macro (Figure 3-3 on page 3-5).

1. *B-LVDS and M-LVDS are supported with the LVDS macro.*



Note: For INBUF driving a PLL macro or CLKDLY macro, the I/O will be hard-routed to the CCC; i.e., will be placed by software to a dedicated Global I/O.*

Figure 3-3 • **CCC Options: Global Buffers with Programmable Delay**

The CLKDLY macro is a pass-through clock source that does not use the PLL, but provides the ability to delay the clock input using a programmable delay. The CLKDLY macro takes the selected clock input and adds a user-defined delay element. This macro generates an output clock phase shift from the input clock.

The CLKDLY macro can be driven by an INBUF* macro to create a composite macro, where the I/O macro drives the global buffer (with programmable delay) using a hardwired connection. In this case, the software will automatically place the dedicated global I/O in the appropriate locations. Many specific INBUF macros support the wide variety of single-ended and differential I/O standards supported by the low-power flash family. The available INBUF macros are described in the *IGLOO, Fusion, and ProASIC3 Macro Library Guide.*

The CLKDLY macro can be driven directly from the FPGA core. The CLKDLY macro can also be driven from an I/O that is routed through the FPGA regular routing fabric. In this case, users must instantiate a special macro, PLLINT, to differentiate the clock input driven by the hardwired I/O connection.

The visual CLKDLY configuration in the SmartGen area of the Actel Libero® Integrated Design Environment (IDE) and Designer tools allows the user to select the desired amount of delay and configures the delay elements appropriately. SmartGen also allows the user to select the input clock source. SmartGen will automatically instantiate the special macro, PLLINT, when needed.

CLKDLY Macro Signal Descriptions

The CLKDLY macro supports one input and one output. Each signal is described in Table 3-2.

Table 3-2 • **Input and Output Description of the CLKDLY Macro**

Signal	Name	I/O	Description
CLK	Reference Clock	Input	Reference clock input
GL	Global Output	Output	Primary output clock to respective global/quadrant clock networks

CLKDLY Macro Usage

When a CLKDLY macro is used in a CCC location, the programmable delay element is used to allow the clock delays to go to the global network. In addition, the user can bypass the PLL in a CCC location integrated with a PLL, but use the programmable delay that is associated with the global network by instantiating the CLKDLY macro. The same is true when using programmable delay elements in a CCC location with no PLLs (the user needs to instantiate the CLKDLY macro). There is no difference between the programmable delay elements used for the PLL and the CLKDLY macro. The CCC will be configured to use the programmable delay elements in accordance with the macro instantiated by the user.

As an example, if the PLL is not used in a particular CCC location, the designer is free to specify up to three CLKDLY macros in the CCC, each of which can have its own input frequency and delay adjustment options. If the PLL core is used, assuming output to only one global clock network, the other two global clock networks are free to be used by either connecting directly from the global inputs or connecting from one or two CLKDLY macros for programmable delay.

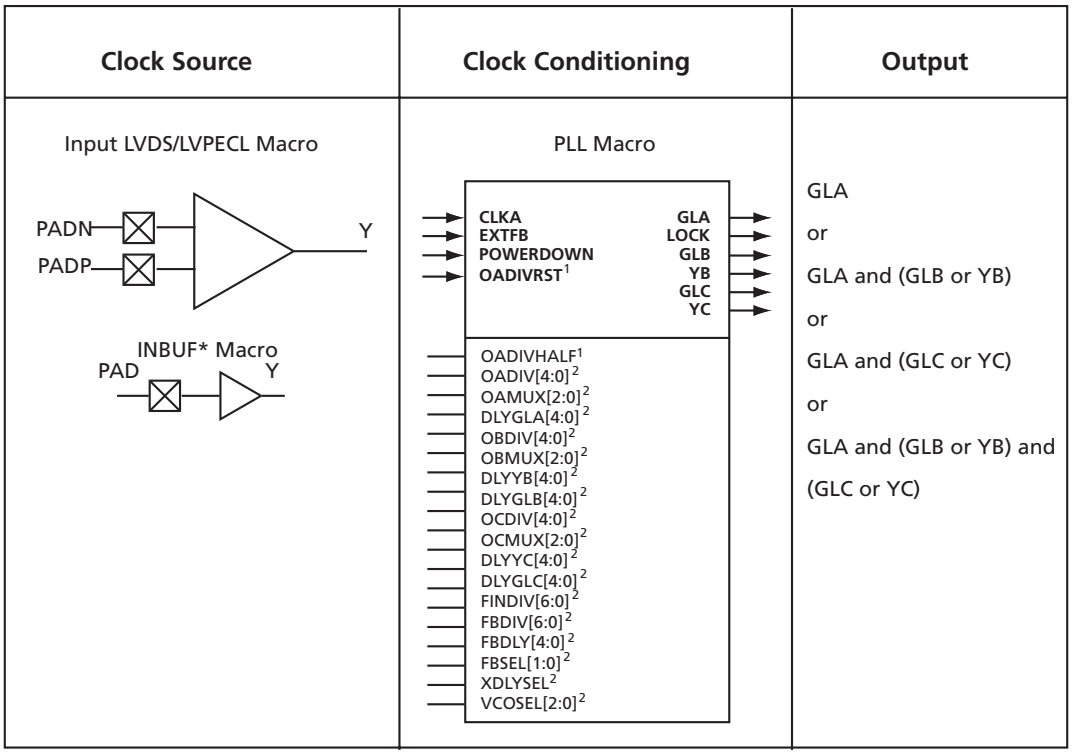
The programmable delay elements are shown in the block diagram of the PLL block shown in Figure 3-5 on page 3-10. Note that any CCC locations with no PLL present contain only the programmable delay blocks going to the global networks (labeled "Programmable Delay Type 2"). Refer to the "Clock Delay Adjustment" section on page 3-25 for a description of the programmable delay types used for the PLL. Also refer to Table 3-13 on page 3-31 for Programmable Delay Type 1 step delay values, and Table 3-14 on page 3-32 for Programmable Delay Type 2 step delay values. CCC locations with a PLL present can be configured to utilize only the programmable delay blocks (Programmable Delay Type 2) going to the global networks A, B, and C.

Global network A can be configured to use only the programmable delay element (bypassing the PLL) if the PLL is not used in the design. Figure 3-5 on page 3-10 shows a block diagram of the PLL, where the programmable delay elements are used for the global networks (Programmable Delay Type 2).

Global Buffers with PLL Function

Clocks requiring frequency synthesis or clock adjustments can utilize the PLL core before connecting to the global / quadrant global networks. A maximum of 18 CCC global buffers can be instantiated in a device—three per CCC and up to six CCCs per device. Each PLL core can generate up to three global/quadrant clocks, while a clock delay element provides one.

The PLL functionality of the clock conditioning block is supported by the PLL macro.



Notes:

1. *For Fusion only.*
2. *Refer to the IGLOO, Fusion, and ProASIC3 Macro Library Guide for more information.*
3. *For INBUF* driving a PLL macro or CLKDLY macro, the I/O will be hard-routed to the CCC; i.e., will be placed by software to a dedicated Global I/O.*

Figure 3-4 • **CCC Options: Global Buffers with PLL**

The PLL macro provides five derived clocks (three independent) from a single reference clock. The PLL macro also provides power-down input and lock output signals. The additional inputs shown on the macro are configuration settings, which are configured through the use of SmartGen. For manual setting of these bits refer to the *IGLOO, Fusion, and ProASIC3 Macro Library Guide* for details.

Figure 3-5 on page 3-10 illustrates the various clock output options and delay elements.

PLL Macro Signal Descriptions

The PLL macro supports two inputs and up to six outputs. Table 3-3 gives a description of each signal.

Table 3-3 • **Input and Output Signals of the PLL Block**

Signal	Name	I/O	Description
CLKA	Reference Clock	Input	Reference clock input for PLL core; input clock for primary output clock, GLA
OADIVRST	Reset Signal for the Output Divider A	Input	For Fusion only. OADIVRST can be used when you bypass the PLL core (i.e., OAMUX = 001). The purpose of the OADIVRST signals is to reset the output of the final clock divider to synchronize it with the input to that divider when the PLL is bypassed. The signal is active on a low to high transition. The signal must be low for at least one divider input. If PLL core is used, this signal is "don't care" and the internal circuitry will generate the reset signal for the synchronization purpose.
OADIVHALF	Output A Division by Half	Input	For Fusion only. Active high. Division by half feature. This feature can only be used when users bypass the PLL core (i.e., OAMUX = 001) and the RC Oscillator (RCOSC) drives the CLKA input. This can be used to divide the 100 MHz RC oscillator by a factor of 1.5, 2.5, 3.5, 4.5 ... 14.5). Refer to Table 3-17 on page 3-33 for more information.
EXTFB	External Feedback	Input	Allows an external signal to be compared to a reference clock in the PLL core's phase detector.
POWERDOWN	Power Down	Input	Active low input that selects power-down mode and disables the PLL. With the POWERDOWN signal asserted, the PLL core sends 0 V signals on all of the outputs.
GLA	Primary Output	Output	Primary output clock to respective global/quadrant clock networks
GLB	Secondary 1 Output	Output	Secondary 1 output clock to respective global/quadrant clock networks
YB	Core 1 Output	Output	Core 1 output clock to local routing network
GLC	Secondary 2 Output	Output	Secondary 2 output clock to respective global/quadrant clock networks
YC	Core 2 Output	Output	Core 2 output clock to local routing network
LOCK	PLL Lock Indicator	Output	Active high signal indicating that steady-state lock has been achieved between CLKA and the PLL feedback signal

Input Clock

The inputs to the input reference clock (CLKA) of the PLL can come from global input pins, regular I/O pins, or internally from the core. For Fusion families, the input reference clock can also be from the embedded RC oscillator or crystal oscillator.

Global Output Clocks

GLA (Primary), GLB (Secondary 1), and GLC (Secondary 2) are the outputs of Global Multiplexer 1, Global Multiplexer 2, and Global Multiplexer 3, respectively. These signals (GLx) can be used to drive the high-speed global and quadrant networks of the low-power flash devices.

A global multiplexer block consists of the input routing for selecting the input signal for the GLx clock and the output multiplexer, as well as delay elements associated with that clock.

Core Output Clocks

YB and YC are known as Core Outputs and can be used to drive internal logic without using global network resources. This is especially helpful when global network resources must be conserved and utilized for other timing-critical paths.

YB and YC are identical to GLB and GLC, respectively, with the exception of a higher selectable final output delay. The SmartGen PLL Wizard will configure these outputs according to user specifications and can enable these signals with or without the enabling of Global Output Clocks.

The above signals can be enabled in the following output groupings in both internal and external feedback configurations of the static PLL:

- One output – GLA only
- Two outputs – GLA + (GLB and/or YB)
- Three outputs – GLA + (GLB and/or YB) + (GLC and/or YC)

PLL Macro Block Diagram

As illustrated, the PLL supports three distinct output frequencies from a given input clock. Two of these (GLB and GLC) can be routed to the B and C global network access, respectively, and/or routed to the device core (YB and YC).

There are five delay elements to support phase control on all five outputs (GLA, GLB, GLC, YB, and YC).

There are delay elements in the feedback loop that can be used to advance the clock relative to the reference clock.

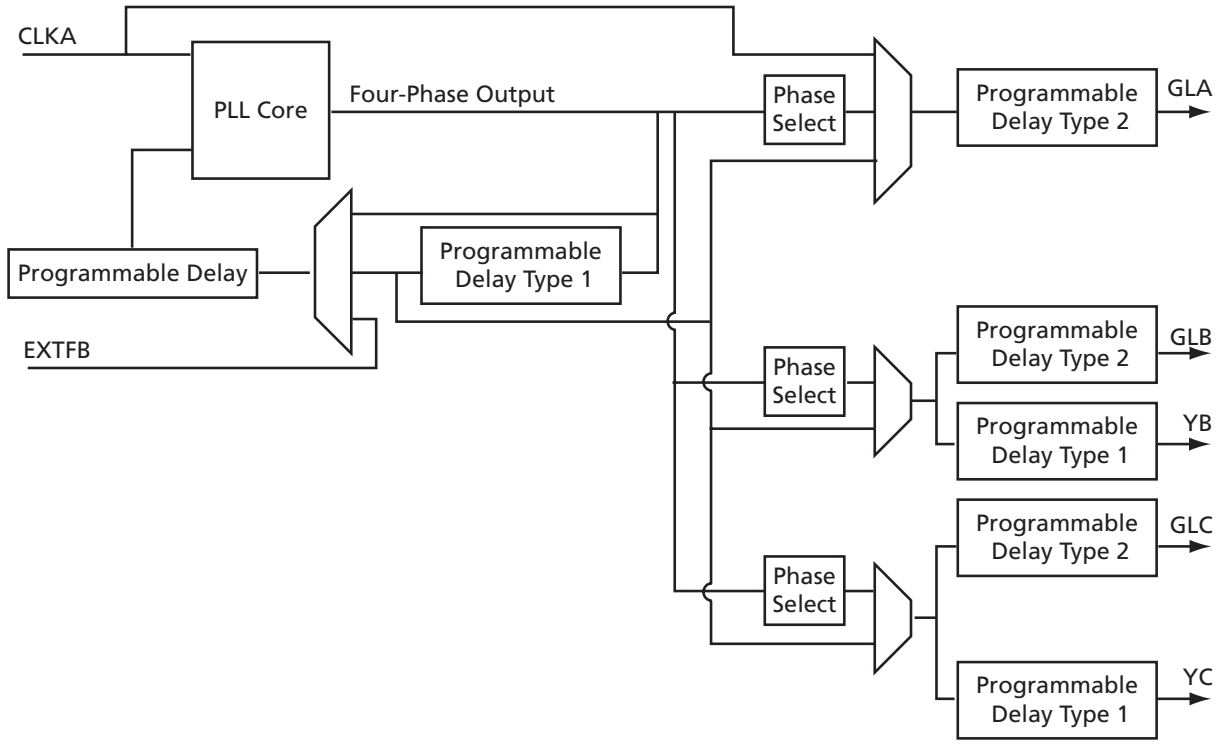
The PLL macro reference clock can be driven in the following ways:

1. By an INBUF* macro to create a composite macro, where the I/O macro drives the global buffer (with programmable delay) using a hardwired connection. In this case, the I/O must be placed in one of the dedicated global I/O locations.
2. Directly from the FPGA core.
3. From an I/O that is routed through the FPGA regular routing fabric. In this case, users must instantiate a special macro, PLLINT, to differentiate from the hardwired I/O connection described earlier.

During power-up, the PLL outputs will toggle around the maximum frequency of the voltage-controlled oscillator (VCO) gear selected. Toggle frequencies can range from 40 MHz to 250 MHz. This will continue as long as the clock input (CLKA) is constant (HIGH or LOW). This can be prevented by LOW assertion of the POWERDOWN signal.

The visual PLL configuration in SmartGen, a component of the Libero IDE and Designer tools, will derive the necessary internal divider ratios based on the input frequency and desired output frequencies selected by the user.

SmartGen also allows the user to select the various delays and phase shift values necessary to adjust the phases between the reference clock (CLKA) and the derived clocks (GLA, GLB, GLC, YB, and YC). SmartGen allows the user to select the input clock source. SmartGen automatically instantiates the special macro, PLLINT, when needed.



Note: Clock divider and clock multiplier blocks are not shown in this figure or in SmartGen. They are automatically configured based on the user's required frequencies.

Figure 3-5 • **CCC with PLL Block**

Global Input Selections

Low-power flash devices provide the flexibility of choosing one of the three global input pad locations available to connect to a CCC functional block or to a global / quadrant global network. Figure 3-6 and Figure 3-7 on page 3-11 show the detailed architecture of each global input structure for 30 k gate devices and below, as well as 60 k gate devices and above, respectively. For 60 k gate devices and above (Figure 3-6), if the single-ended I/O standard is chosen, there is flexibility to choose one of the global input pads (the first, second, and fourth input). Once chosen, the other I/O locations are used as regular I/Os. If the differential I/O standard is chosen, the first and second inputs are considered as paired, and the third input is paired with a regular I/O.

The user then has the choice of selecting one of the two sets to be used as the clock input source to the CCC functional block. There is also the option to allow an internal clock signal to feed the global network or the CCC functional block. A multiplexer tree selects the appropriate global input for routing to the desired location. Note that the global I/O pads do not need to feed the global network; they can also be used as regular I/O pads.

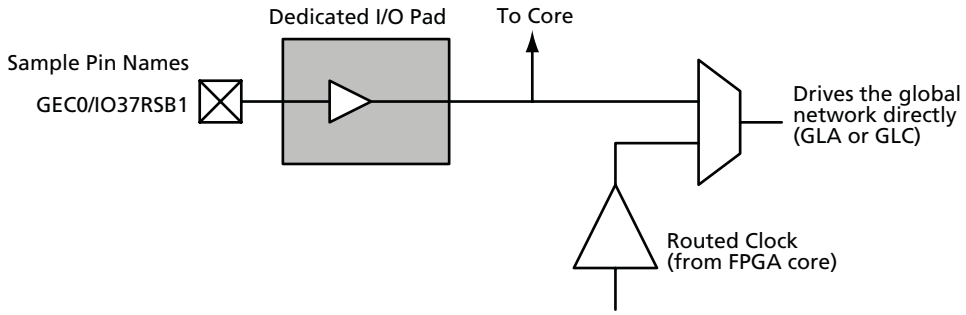
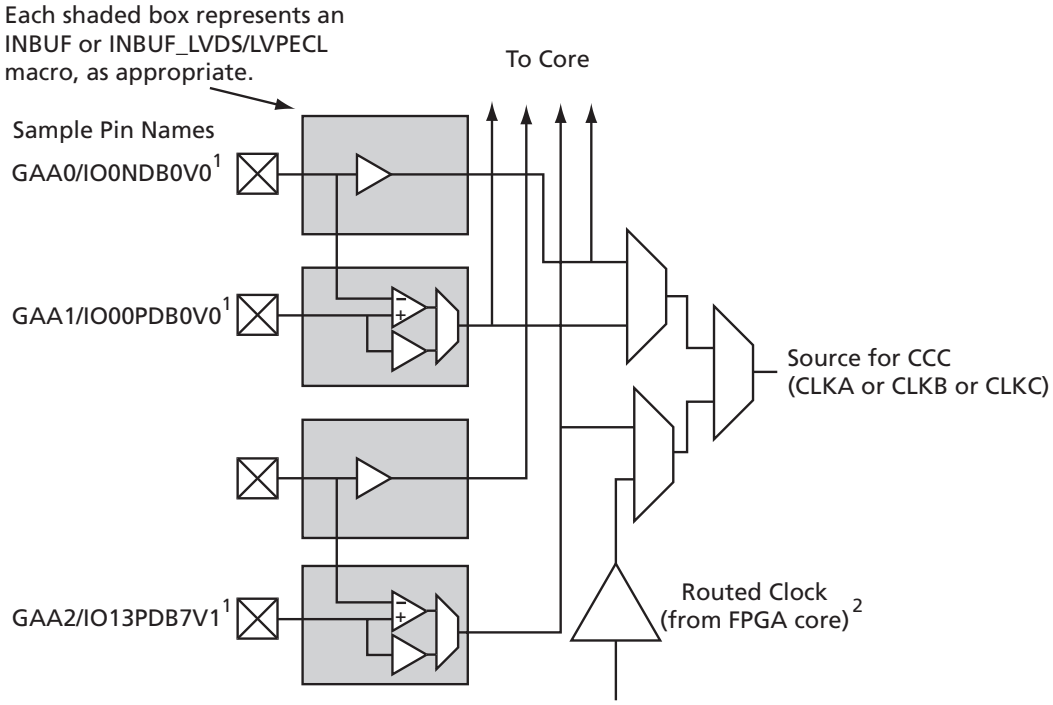


Figure 3-6 • **Clock Input Sources (30 k gates devices and below)**



GAA[0:2]: GA represents global in the northwest corner of the device. A[0:2]: designates specific A clock source.

Notes:

1. *Represents the global input pins. Globals have direct access to the clock conditioning block and are not routed via the FPGA fabric. Refer to User I/O Naming Conventions in I/O Structures in IGLOO and ProASIC3 Devices.*
2. *Instantiate the routed clock source input as follows:*
   a) *Connect the output of a logic element to the clock input of a PLL, CLKDLY, or CLKINT macro.*
   b) *Do not place a clock source I/O (INBUF or INBUF_LVPECL/LVDS/B-LVDS/M-LVDS/DDR) in a relevant global pin location.*

Figure 3-7 • **Clock Input Sources Including CLKBUF, CLKBUF_LVDS/LVPECL, and CLKINT (60 k gates devices and above)**

Each global buffer, as well as the PLL reference clock, can be driven from one of the following:

- 3 dedicated single-ended I/Os using a hardwired connection
- 2 dedicated differential I/Os using a hardwired connection
- The FPGA core

Since the architecture of the devices varies as size increases, the following list details I/O types supported for globals:

IGLOO and ProASIC3

- LVDS-based clock sources are available only on 250 k gate devices and above.
- 60 k and 125 k gate devices support single-ended clock sources only.
- 15 k and 30 k gate devices support these inputs for CCC only and do not contain a PLL.
- nano devices:
  - 10 k, 15 k, and 20 k devices do not contain PLLs in the CCCs, and support only CLKBUF and CLKINT.
  - 60 k, 125 k, and 250 k devices support one PLL in the middle left CCC position. In the absence of the PLL, this CCC can be used by CLKBUF, CLKINT, and CLKDLY macros. The corner CCCs support CLKBUF, CLKINT, and CLKDLY.

Fusion

- AFS600 and AFS1500: All single-ended, differential, and voltage-referenced I/O standards (Pro I/O).
- AFS090 and AFS250: All single-ended and differential I/O standards.

Clock Sources for PLL and CLKDLY Macros

The input reference clock (CLKA for a PLL macro, CLK for a CLKDLY macro) can be accessed from different sources via the associated clock multiplexer tree. Each CCC has the option of choosing the source of the input clock from one of the following:

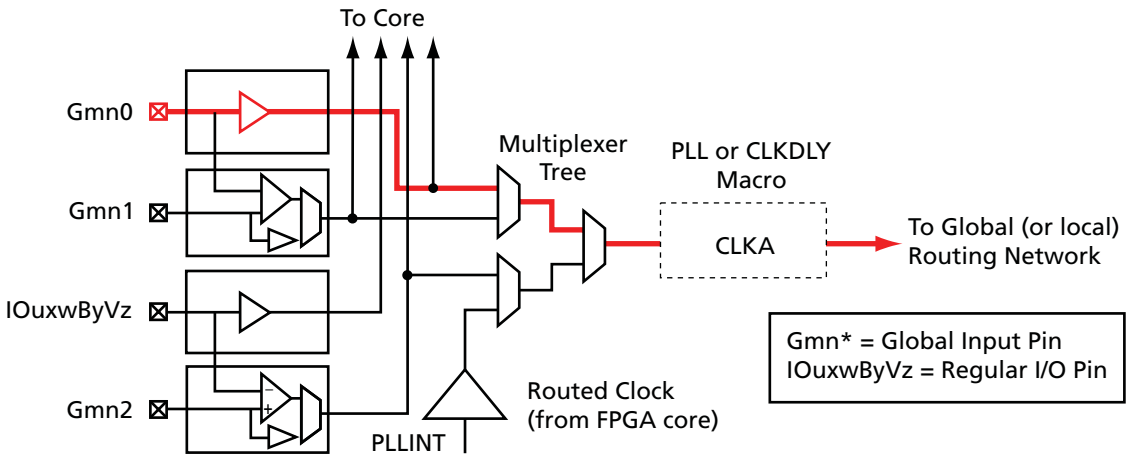
- Hardwired I/O
- External I/O
- Core Logic
- RC Oscillator (Fusion only)
- Crystal Oscillator (Fusion only)

The SmartGen macro builder tool allows users to easily create the PLL and CLKDLY macros with the desired settings. Actel strongly recommends using SmartGen to generate the CCC macros.

Hardwired I/O Clock Source

Hardwired I/O refers to global input pins that are hardwired to the multiplexer tree, which directly accesses the CCC global buffers. These global input pins have designated pin locations and are indicated with the I/O naming convention *Gmn* (*m* refers to any one of the positions where the PLL core is available, and *n* refers to any one of the three global input MUXes and the pin number of the associated global location, *m*). Choosing this option provides the benefit of directly connecting to the CCC reference clock input, which provides less delay. See Figure 3-8 on page 3-13 for an example illustration of the connections, shown in red. If a CLKDLY macro is initiated to utilize the programmable delay element of the CCC, the clock input can be placed at one of nine dedicated global input pin locations. In other words, if Hardwired I/O is chosen as the input source, the user can decide to place the input pin in one of the GmA0, GmA1, GmA2, GmB0, GmB1, GmB2, GmC0, GmC1, or GmC2 locations of the low-power flash devices. When a PLL macro is used to utilize the

PLL core in a CCC location, the clock input of the PLL can only be connected to one of three GmA* global pin locations: GmA0, GmA1, or GmA2.



Note: Fusion CCCs have additional source selections (RCOSC, XTAL).

Figure 3-8 • **Illustration of Hardwired I/O (global input pins) Usage for IGLOO and ProASIC3 devices 60 k Gates and Larger**

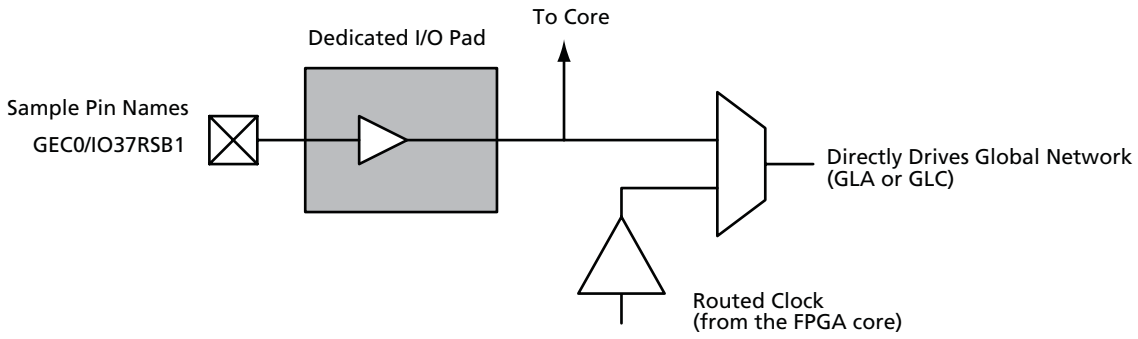


Figure 3-9 • **Illustration of Hardwired I/O (global input pins) Usage for IGLOO and ProASIC3 devices 30 k Gates and Smaller**

External I/O Clock Source

External I/O refers to regular I/O pins. The clock source is instantiated with one of the various INBUF options and accesses the CCCs via internal routing. The user has the option of assigning this input to any of the I/Os labeled with the I/O convention *IOuxwByVz*. Refer to *User I/O Naming Conventions in I/O Structures in IGLOO and ProASIC3 Devices,* and for Fusion, refer to the *Fusion Mixed-Signal Programmable System Chip* datasheet for more information. Figure 3-10 gives a brief explanation of external I/O usage. Choosing this option provides the freedom of selecting any user I/O location but introduces additional delay because the signal connects to the routed clock input through internal routing before connecting to the CCC reference clock input.

For the External I/O option, the routed signal would be instantiated with a PLLINT macro before connecting to the CCC reference clock input. This instantiation is conveniently done automatically by SmartGen when this option is selected. Actel recommends using the SmartGen tool to generate the CCC macro. The instantiation of the PLLINT macro results in the use of the routed clock input of the I/O to connect to the PLL clock input. If not using SmartGen, manually instantiate a PLLINT macro before the PLL reference clock to indicate that the regular I/O driving the PLL reference clock should be used (see Figure 3-10 on page 3-14 for an example illustration of the connections, shown in red).

In the above two options, the clock source must be instantiated with one of the various INBUF macros. The reference clock pins of the CCC functional block core macros must be driven by regular input macros (INBUFs), not clock input macros.

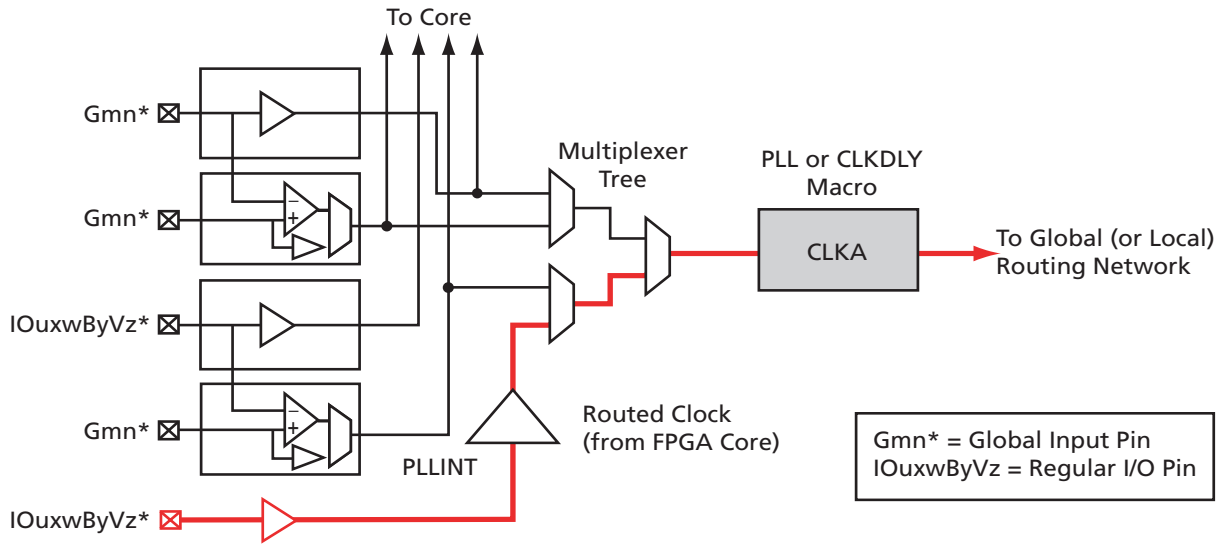


Figure 3-10 • **Illustration of External I/O Usage**

For Fusion devices, the input reference clock can also be from the embedded RC oscillator and crystal oscillator. In this case, the CCC configuration is the same as the hardwired I/O clock source, and users are required to instantiate the RC oscillator or crystal oscillator macro and connect its output to the input reference clock of the CCC block.

Core Logic Clock Source

Core logic refers to internal routed nets. Internal routed signals access the CCC via the FPGA Core Fabric. Similar to the External I/O option, whenever the clock source comes internally from the core itself, the routed signal is instantiated with a PLLINT macro before connecting to the CCC clock input (see Figure 3-11 for an example illustration of the connections, shown in red).

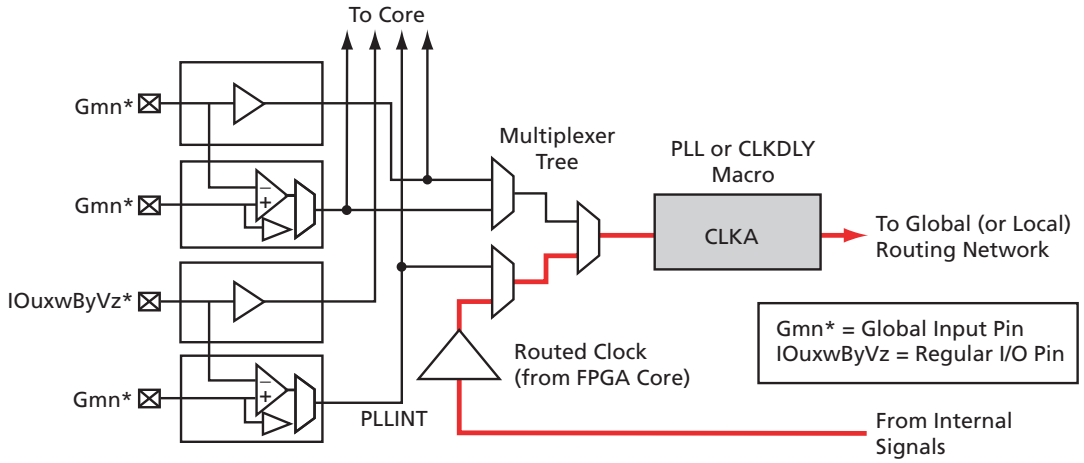


Figure 3-11 • **Illustration of Core Logic Usage**

For Fusion devices, the input reference clock can also be from the embedded RC oscillator and crystal oscillator. In this case, the CCC configuration is the same as the hardwired I/O clock source, and users are required to instantiate the RC oscillator or crystal oscillator macro and connect its output to the input reference clock of the CCC block.

Available I/O Standards

Table 3-4 • **Available I/O Standards within CLKBUF and CLKBUF_LVDS/LVPECL Macros**

CLKBUF_LVCMOS5
CLKBUF_LVCMOS33 [1]
CLKBUF_LVCMOS25 [2]
CLKBUF_LVCMOS18
CLKBUF_LVCMOS15
CLKBUF_PCI
CLKBUF_PCIX [3]
CLKBUF_GTL25 [2,3]
CLKBUF_GTL33 [2,3]
CLKBUF_GTLP25 [2,3]
CLKBUF_GTLP33 [2,3]
CLKBUF_HSTL_I [2,3]
CLKBUF_HSTL_II [2,3]
CLKBUF_SSTL3_I [2,3]
CLKBUF_SSTL3_II [2,3]
CLKBUF_SSTL2_I [2,3]
CLKBUF_SSTL2_II [2,3]
CLKBUF_LVDS [4]
CLKBUF_LVPECL

Notes:

1. *By default, the CLKBUF macro uses 3.3 V LVTTL I/O technology. For more details, refer to the* IGLOO, Fusion, and ProASIC3 Macro Library Guide.
2. *I/O standards only supported in ProASIC3E and IGLOOe families.*
3. *I/O standards only supported in the following Fusion devices: AFS600 and AFS1500.*
4. *B-LVDS and M-LVDS standards are supported by CLKBUF_LVDS.*

Global Synthesis Constraints

The Synplify® synthesis tool, by default, allows six clocks in a design for Fusion, IGLOO, and ProASIC3. When more than six clocks are needed in the design, a user synthesis constraint attribute, syn_global_buffers, can be used to control the maximum number of clocks (up to 18) that can be inferred by the synthesis engine.

High-fanout nets will be inferred with clock buffers and/or internal clock buffers. If the design consists of CCC global buffers, they are included in the count of clocks in the design.

The subsections below discuss the clock input source (global buffers with no programmable delays) and the clock conditioning functional block (global buffers with programmable delays and/or PLL function) in detail.

Device-Specific Layout

Two kinds of CCCs are offered in low-power flash devices: CCCs with integrated PLLs, and CCCs without integrated PLLs (simplified CCCs). Table 3-5 lists the number of CCCs in various devices.

Table 3-5 • **Number of CCCs by Device Size and Package**

Device		Package	CCCs with Integrated PLLs	CCCs without Integrated PLLs (simplified CCC)
ProASIC3	**IGLOO**			
A3PN010	AGLN010	All	0	2
A3PN015	AGLN015	All	0	2
A3PN020	AGLN020	All	0	2
	AGLN060	CS81	0	6
A3PN060	AGLN060	All other packages	1	5
	AGLN125	CS81	0	6
A3PN125	AGLN125	All other packages	1	5
	AGLN250	CS81	0	6
A3PN250	AGLN250	All other packages	1	5
A3P015	AGL015	All	0	2
A3P030	AGL030/AGLP030	All	0	2
	AGL060/AGLP060	CS121/CS201	0	6
A3P060	AGL060/AGLP060	All other packages	1	5
A3P125	AGL125/AGLP125	All	1	5
A3P250/L	AGL250	All	1	5
A3P400	AGL400	All	1	5
A3P600/L	AGL600	All	1	5
A3P1000/L	AGL1000	All	1	5
A3PE600	AGLE600	PQ208	2	4
A3PE600/L		All other packages	6	0
A3PE1500		PQ208	2	4
A3PE1500		All other packages	6	0
A3PE3000/L		PQ208	2	4
A3PE3000/L	AGLE3000	All other packages	6	0
Fusion Devices				
AFS090		All	1	5
AFS250, M1AFS250		All	1	5
AFS600, M7AFS600, M1AFS600		All	2	4
AFS1500, M1AFS1500		All	2	4

Note: nano 10 k, 15 k, and 20 k offer 6 global MUXes instead of CCCs.

This section outlines the following device information: CCC features, PLL core specifications, functional descriptions, software configuration information, detailed usage information, recommended board-level considerations, and other considerations concerning global networks in low-power flash devices.

Clock Conditioning Circuits with Integrated PLLs

Each of the CCCs with integrated PLLs includes the following:

- 1 PLL core, which consists of a phase detector, a low-pass filter, and a four-phase voltage-controlled oscillator
- 3 global multiplexer blocks that steer signals from the global pads and the PLL core onto the global networks
- 6 programmable delays and 1 fixed delay for time advance/delay adjustments
- 5 programmable frequency divider blocks to provide frequency synthesis (automatically configured by the SmartGen macro builder tool)

Clock Conditioning Circuits without Integrated PLLs

There are two types of simplified CCCs without integrated PLLs in low-power flash devices.

1. The simplified CCC with programmable delays, which is composed of the following:
   – 3 global multiplexer blocks that steer signals from the global pads and the programmable delay elements onto the global networks
   – 3 programmable delay elements to provide time delay adjustments
2. The simplified CCC (referred to as CCC-GL) without programmable delay elements, which is composed of the following:
   – A global multiplexer block that steer signals from the global pads onto the global networks

CCC Locations

CCCs located in the middle of the east and west sides of the device access the three VersaNet global networks on each side (six total networks), while the four CCCs located in the four corners access three quadrant global networks (twelve total networks). See Figure 3-12.

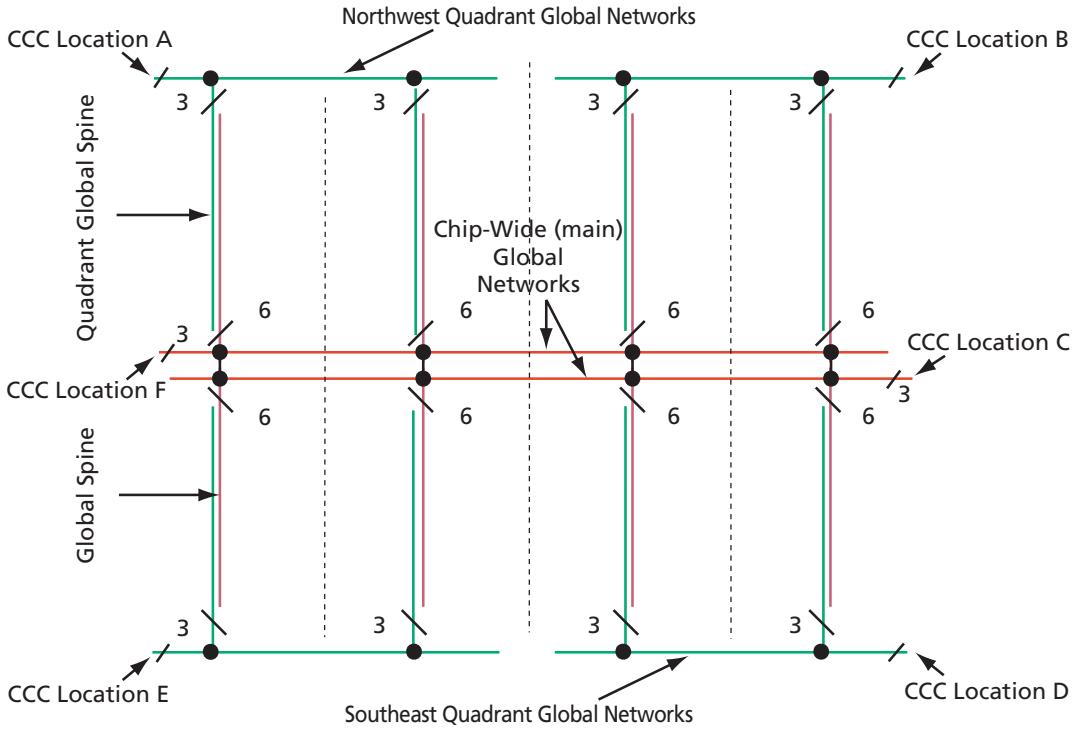


***Figure 3-12* • Global Network Architecture for 60 k Gate Devices and Above**

The following explains the locations of the CCCs in IGLOO and ProASIC3 devices:

In Figure 3-14 on page 3-21 through Figure 3-15 on page 3-21, CCCs with integrated PLLs are indicated in red, and simplified CCCs are indicated in yellow. There is a letter associated with each location of the CCC, in clockwise order. The upper left corner CCC is named "A," the upper right is named "B," and so on. These names finish up at the middle left with letter "F."

IGLOO and ProASIC3 CCC Locations

In all IGLOO and ProASIC3 devices (except 10 k through 30 k gate devices, which do not contain PLLs), six CCCs are located in the same positions as the IGLOOe and ProASIC3E CCCs. Only one of the CCCs has an integrated PLL and is located in the middle of the west (middle left) side of the device. The other five CCCs are simplified CCCs and are located in the four corners and the middle of the east side of the device (Figure 3-13).

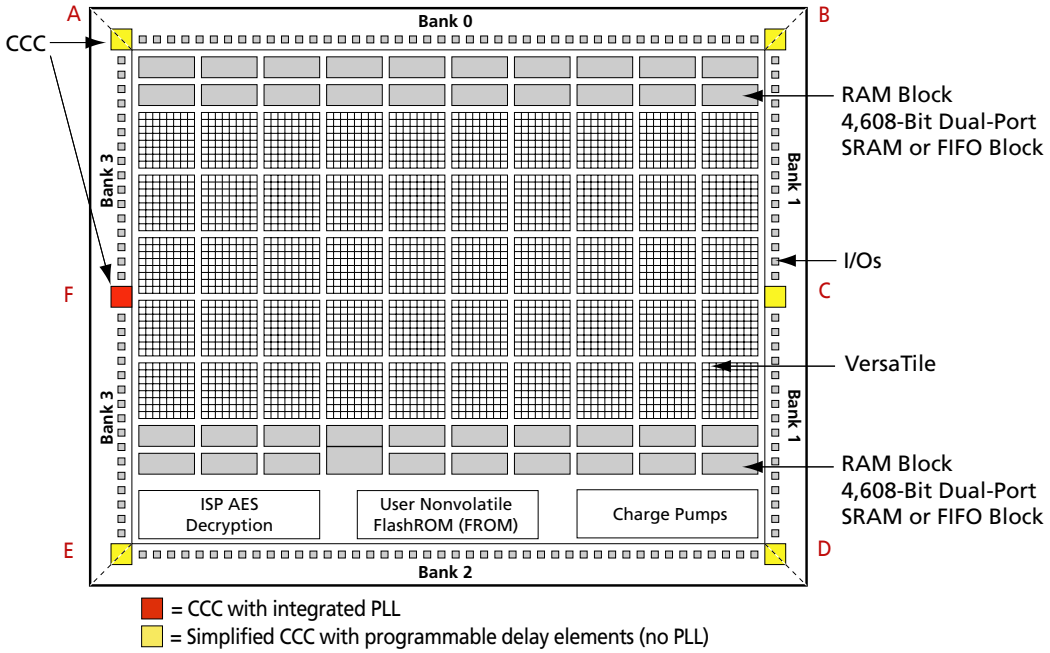


Figure 3-13 • **CCC Locations in IGLOO and ProASIC3 Family Devices (except 10 k through 30 k gate devices)**

Note: The number and architecture of the banks are different for some devices.

10 k through 30 k gate devices do not support PLL features. In these devices, there are two CCC-GLs at the lower corners (one at the lower right, and one at the lower left). These CCC-GLs do not have programmable delays.

IGLOOe and ProASIC3E CCC Locations

IGLOOe and ProASIC3E devices have six CCCs—one in each of the four corners and one each in the middle of the east and west sides of the device (Figure 3-14).

All six CCCs are integrated with PLLs, except in PQFP-208 package devices. PQFP-208 package devices also have six CCCs, of which two include PLLs and four are simplified CCCs. The CCCs with PLLs are implemented in the middle of the east and west sides of the device (middle right and middle left). The simplified CCCs without PLLs are located in the four corners of the device (Figure 3-15).

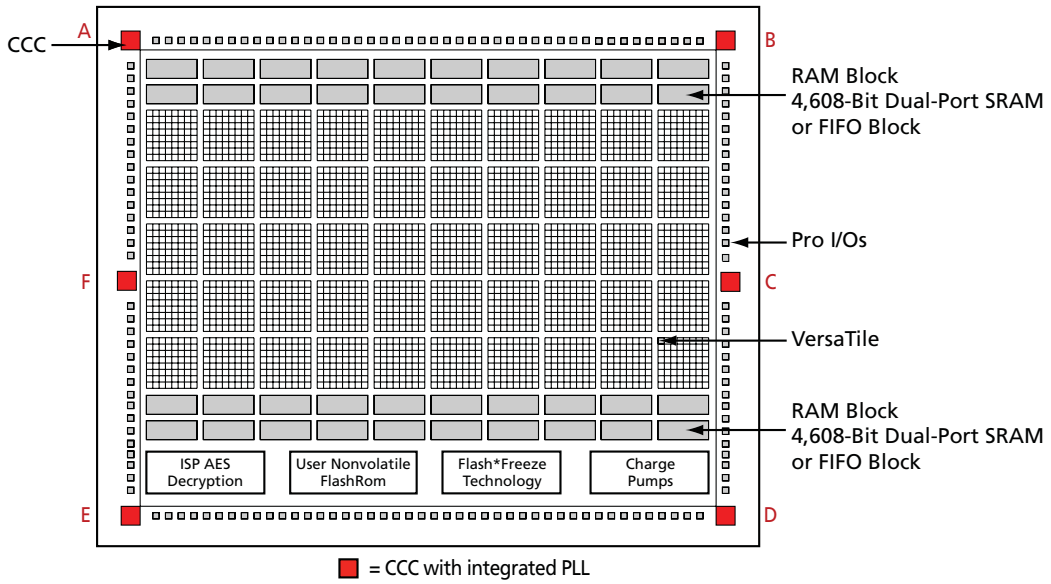


Figure 3-14 • **CCC Locations in IGLOOe and ProASIC3E Family Devices (except PQFP-208 package)**

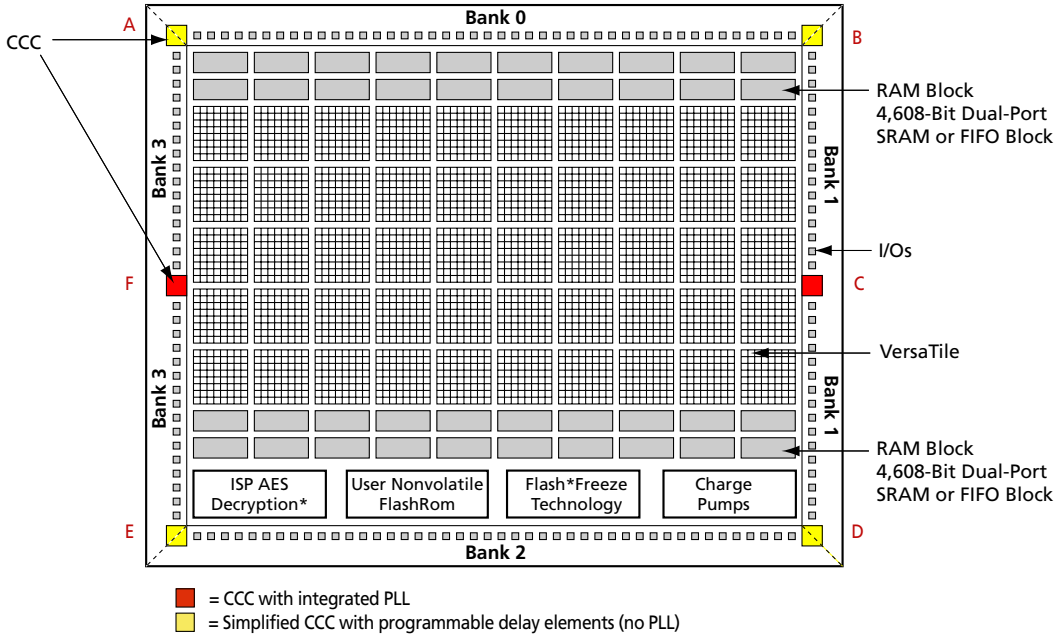


Figure 3-15 • **CCC Locations in ProASIC3E Family Devices (PQFP-208 package)**

Fusion CCC Locations

Fusion devices have six CCCs: one in each of the four corners and one each in the middle of the east and west sides of the device (Figure 3-16 and Figure 3-17). The device can have one integrated PLL in the middle of the west side of the device or two integrated PLLs in the middle of the east and west sides of the device (middle right and middle left).

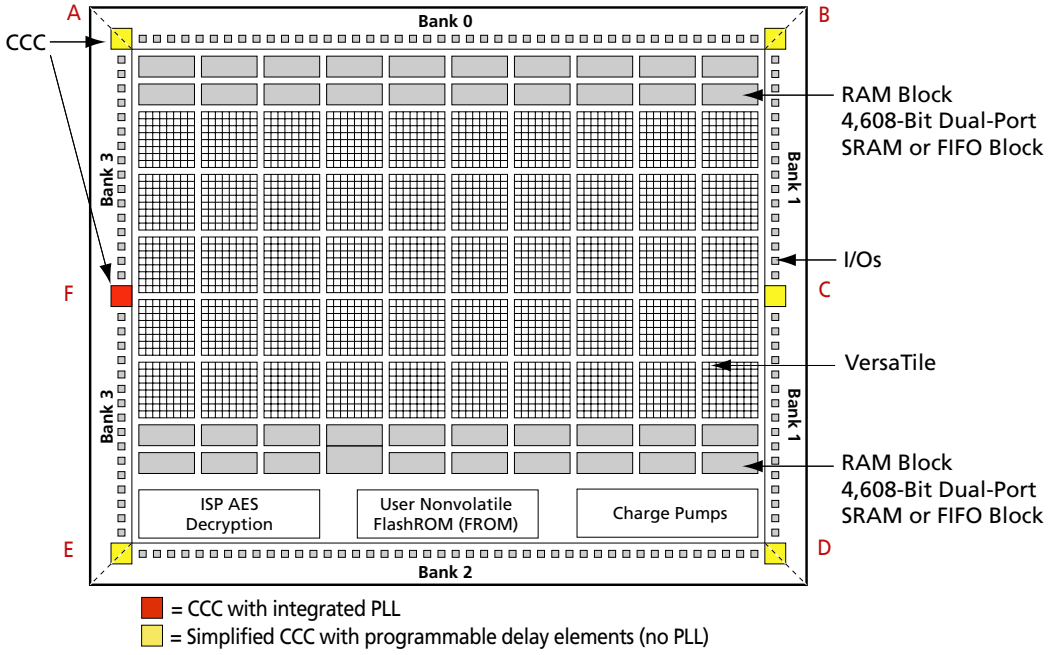


Figure 3-16 • **CCC Locations in Fusion Family Devices (AFS090, AFS250, M1AFS250)**

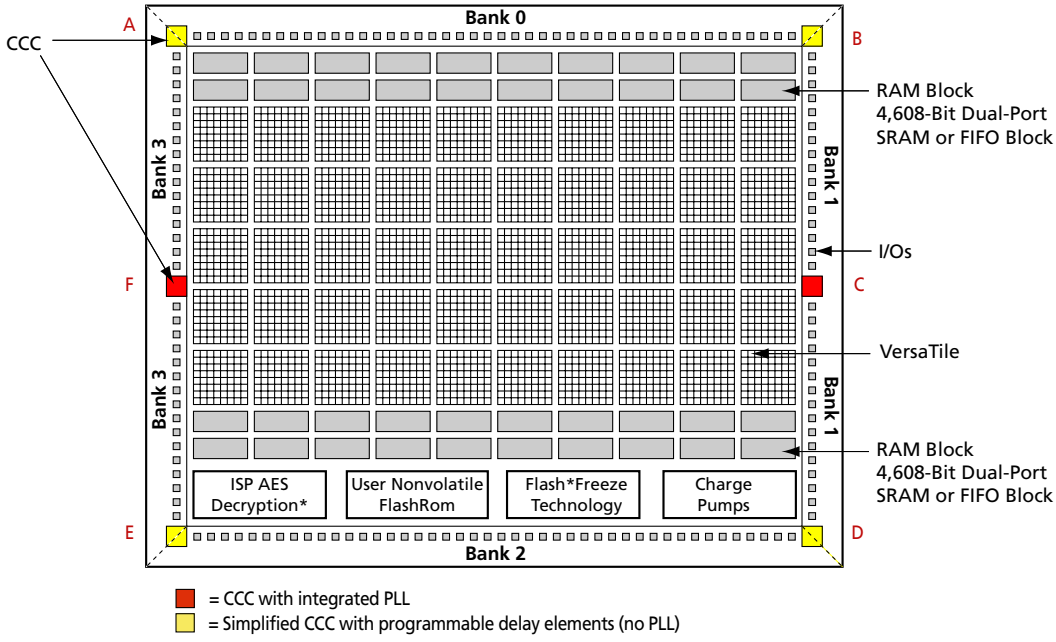


Figure 3-17 • **CCC Locations in Fusion Family Devices (except AFS090, AFS250, M1AFS250)**

PLL Core Specifications

PLL core specifications can be found in the DC and Switching Characteristics chapter of the appropriate family datasheet.

Loop Bandwidth

Common design practice for systems with a low-noise input clock is to have PLLs with small loop bandwidths to reduce the effects of noise sources at the output. Table 3-6 shows the PLL loop bandwidth, providing a measure of the PLL's ability to track the input clock and jitter.

Table 3-6 • **–3 dB Frequency of the PLL**

	Minimum (T_a = +125°C, V_{CCA} = 1.4 V)	Typical (T_a = +25°C, V_{CCA} = 1.5 V)	Maximum (T_a = –55°C, V_{CCA} = 1.6 V)
–3 dB Frequency	15 kHz	25 kHz	45 kHz

PLL Core Operating Principles

This section briefly describes the basic principles of PLL operation. The PLL core is composed of a phase detector (PD), a low-pass filter (LPF), and a four-phase voltage-controlled oscillator (VCO). Figure 3-18 illustrates a basic single-phase PLL core with a divider and delay in the feedback path.

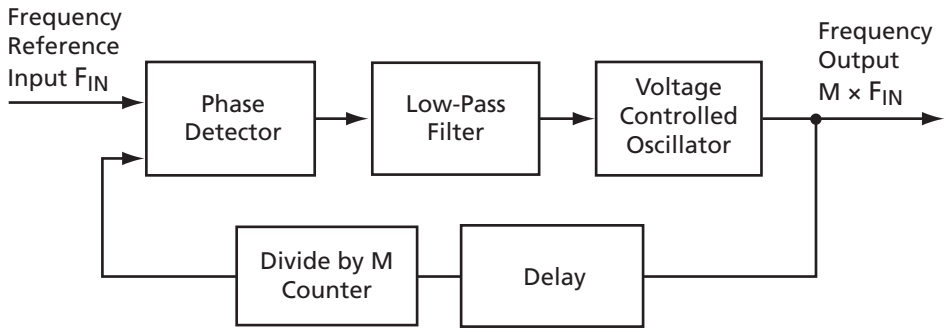


Figure 3-18 • **Simplified PLL Core with Feedback Divider and Delay**

The PLL is an electronic servo loop that phase-aligns the PD feedback signal with the reference input. To achieve this, the PLL dynamically adjusts the VCO output signal according to the average phase difference between the input and feedback signals.

The first element is the PD, which produces a voltage proportional to the phase difference between its inputs. A simple example of a digital phase detector is an Exclusive-OR gate. The second element, the LPF, extracts the average voltage from the phase detector and applies it to the VCO. This applied voltage alters the resonant frequency of the VCO, thus adjusting its output frequency.

Consider Figure 3-18 with the feedback path bypassing the divider and delay elements. If the LPF steadily applies a voltage to the VCO such that the output frequency is identical to the input frequency, this steady-state condition is known as lock. Note that the input and output phases are also identical. The PLL core sets a LOCK output signal HIGH to indicate this condition.

Should the input frequency increase slightly, the PD detects the frequency/phase difference between its reference and feedback input signals. Since the PD output is proportional to the phase difference, the change causes the output from the LPF to increase. This voltage change increases the resonant frequency of the VCO and increases the feedback frequency as a result. The PLL dynamically adjusts in this manner until the PD senses two phase-identical signals and steady-state lock is achieved. The opposite (decreasing PD output signal) occurs when the input frequency decreases.

Now suppose the feedback divider is inserted in the feedback path. As the division factor M (shown in Figure 3-19) is increased, the average phase difference increases. The average phase difference will cause the VCO to increase its frequency until the output signal is phase-identical to the input after undergoing division. In other words, lock in both frequency and phase is achieved when the output frequency is M times the input. Thus, clock division in the feedback path results in multiplication at the output.

A similar argument can be made when the delay element is inserted into the feedback path. To achieve steady-state lock, the VCO output signal will be delayed by the input period *less* the feedback delay. For periodic signals, this is equivalent to time-advancing the output clock by the feedback delay.

Another key parameter of a PLL system is the acquisition time. Acquisition time is the amount of time it takes for the PLL to achieve lock (i.e., phase-align the feedback signal with the input reference clock). For example, suppose there is no voltage applied to the VCO, allowing it to operate at its free-running frequency. Should an input reference clock suddenly appear, a lock would be established within the maximum acquisition time.

Functional Description

This section provides detailed descriptions of PLL block functionality: clock dividers and multipliers, clock delay adjustment, phase adjustment, and dynamic PLL configuration.

Clock Dividers and Multipliers

The PLL block contains five programmable dividers. Figure 3-19 shows a simplified PLL block.

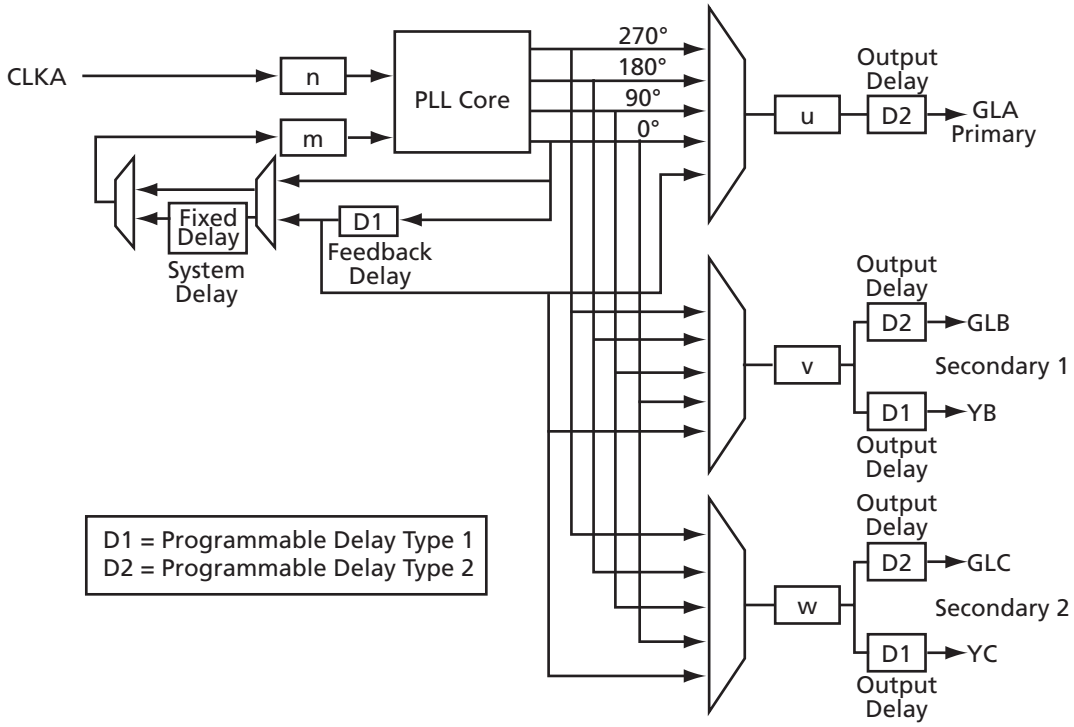


***Figure 3-19* • PLL Block Diagram**

Dividers *n* and *m* (the input divider and feedback divider, respectively) provide integer frequency division factors from 1 to 128. The output dividers *u*, *v*, and *w* provide integer division factors from 1 to 32. Frequency scaling of the reference clock CLKA is performed according to the following formulas:

$$f_{GLA} = f_{CLKA} \times m / (n \times u) – \text{GLA Primary PLL Output Clock}$$

EQ 3-1

$$f_{GLB} = f_{YB} = f_{CLKA} \times m / (n \times v) – \text{GLB Secondary 1 PLL Output Clock(s)}$$

EQ 3-2

$$f_{GLC} = f_{YC} = f_{CLKA} \times m / (n \times w) – \text{GLC Secondary 2 PLL Output Clock(s)}$$

EQ 3-3

SmartGen provides a user-friendly method of generating the configured PLL netlist, which includes automatically setting the division factors to achieve the closest possible match to the requested frequencies. Since the five output clocks share the *n* and *m* dividers, the achievable output frequencies are interdependent and related according to the following formula:

$$f_{GLA} = f_{GLB} \times (v / u) = f_{GLC} \times (w / u)$$

EQ 3-4

Clock Delay Adjustment

There are a total of seven configurable delay elements implemented in the PLL architecture.

Two of the delays are located in the feedback path, entitled System Delay and Feedback Delay. System Delay provides a fixed delay of 2 ns (typical), and Feedback Delay provides selectable delay values from 0.6 ns to 5.56 ns in 160 ps increments (typical). For PLLs, delays in the feedback path will effectively advance the output signal from the PLL core with respect to the reference clock. Thus, the System and Feedback delays generate negative delay on the output clock. Additionally, each of these delays can be independently bypassed if necessary.

The remaining five delays perform traditional time delay and are located at each of the outputs of the PLL. Besides the fixed global driver delay of 0.755 ns for each of the global networks, the global multiplexer outputs (GLA, GLB, and GLC) each feature an additional selectable delay value from 0.025 ns to 0.76 ns in the first step, and then to 5.56 ns in 160 ps increments. The additional YB and YC signals have access to a selectable delay from 0.6 ns to 5.56 ns in 160 ps increments (typical). This is the same delay value as the CLKDLY macro. It is similar to CLKDLY, which bypasses the PLL core just to take advantage of the phase adjustment option with the delay value.

The following parameters must be taken into consideration to achieve minimum delay at the outputs (GLA, GLB, GLC, YB, and YC) relative to the reference clock: routing delays from the PLL core to CCC outputs, core outputs and global network output delays, and the feedback path delay. The feedback path delay acts as a time advance of the input clock and will offset any delays introduced beyond the PLL core output. The routing delays are determined from back-annotated simulation and are configuration-dependent.

Phase Adjustment

The output from the PLL core can be phase-adjusted with respect to the reference input clock, CLKA. The user can select a 0°, 90°, 180°, or 270° phase shift independently for each of the outputs YA, GLB/YB, and GLC/YC. Note that each of these phase-adjusted signals might also undergo further frequency division and/or time adjustment via the remaining dividers and delays located at the outputs of the PLL.

Dynamic PLL Configuration

The CCCs can be configured both statically and dynamically.

In addition to the ports available in the Static CCC, the Dynamic CCC has the dynamic shift register signals that enable dynamic reconfiguration of the CCC. With the Dynamic CCC, the ports CLKB and CLKC are also exposed. All three clocks (CLKA, CLKB, and CLKC) can be configured independently.

The CCC block is fully configurable. The following two sources can act as the CCC configuration bits.

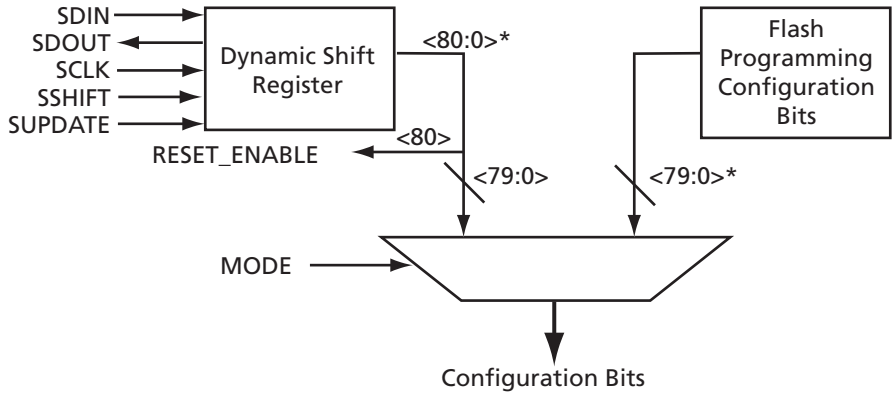
Flash Configuration Bits

The flash configuration bits are the configuration bits associated with programmed flash switches. These bits are used when the CCC is in static configuration mode. Once the device is programmed, these bits cannot be modified. They provide the default operating state of the CCC.

Dynamic Shift Register Outputs

This source does not require core reprogramming and allows core-driven dynamic CCC reconfiguration. When the dynamic register drives the configuration bits, the user-defined core circuit takes full control over SDIN, SDOUT, SCLK, SSHIFT, and SUPDATE. The configuration bits can consequently be dynamically changed through shift and update operations in the serial register interface. Access to the logic core is accomplished via the dynamic bits in the specific tiles assigned to the PLLs.

Figure 3-20 illustrates a simplified block diagram of the MUX architecture in the CCCs.



* For Fusion, bit <88:81> is also needed.

***Figure 3-20* • The CCC Configuration MUX Architecture**

The selection between the flash configuration bits and the bits from the configuration register is made using the MODE signal shown in Figure 3-20. If the MODE signal is logic HIGH, the dynamic shift register configuration bits are selected. There are 81 control bits to configure the different functions of the CCC.

Each group of control bits is assigned a specific location in the configuration shift register. For a list of the 81 configuration bits (C[80:0]) in the CCC and a description of each, refer to "PLL Configuration Bits Description" on page 3-28. The configuration register can be serially loaded with the new configuration data and programmed into the CCC using the following ports:

- SDIN: The configuration bits are serially loaded into a shift register through this port. The LSB of the configuration data bits should be loaded first.
- SDOUT: The shift register contents can be shifted out (LSB first) through this port using the shift operation.
- SCLK: This port should be driven by the shift clock.
- SSHIFT: The active-high shift enable signal should drive this port. The configuration data will be shifted into the shift register if this signal is HIGH. Once SSHIFT goes LOW, the data shifting will be halted.

- SUPDATE: The SUPDATE signal is used to configure the CCC with the new configuration bits when shifting is complete.

To access the configuration ports of the shift register (SDIN, SDOUT, SSHIFT, etc.), the user should instantiate the CCC macro in his design with appropriate ports. Actel recommends that users choose SmartGen to generate the CCC macros with the required ports for dynamic reconfiguration.

Users must familiarize themselves with the architecture of the CCC core and its input, output, and configuration ports to implement the desired delay and output frequency in the CCC structure. Figure 3-21 shows a model of the CCC with configurable blocks and switches.

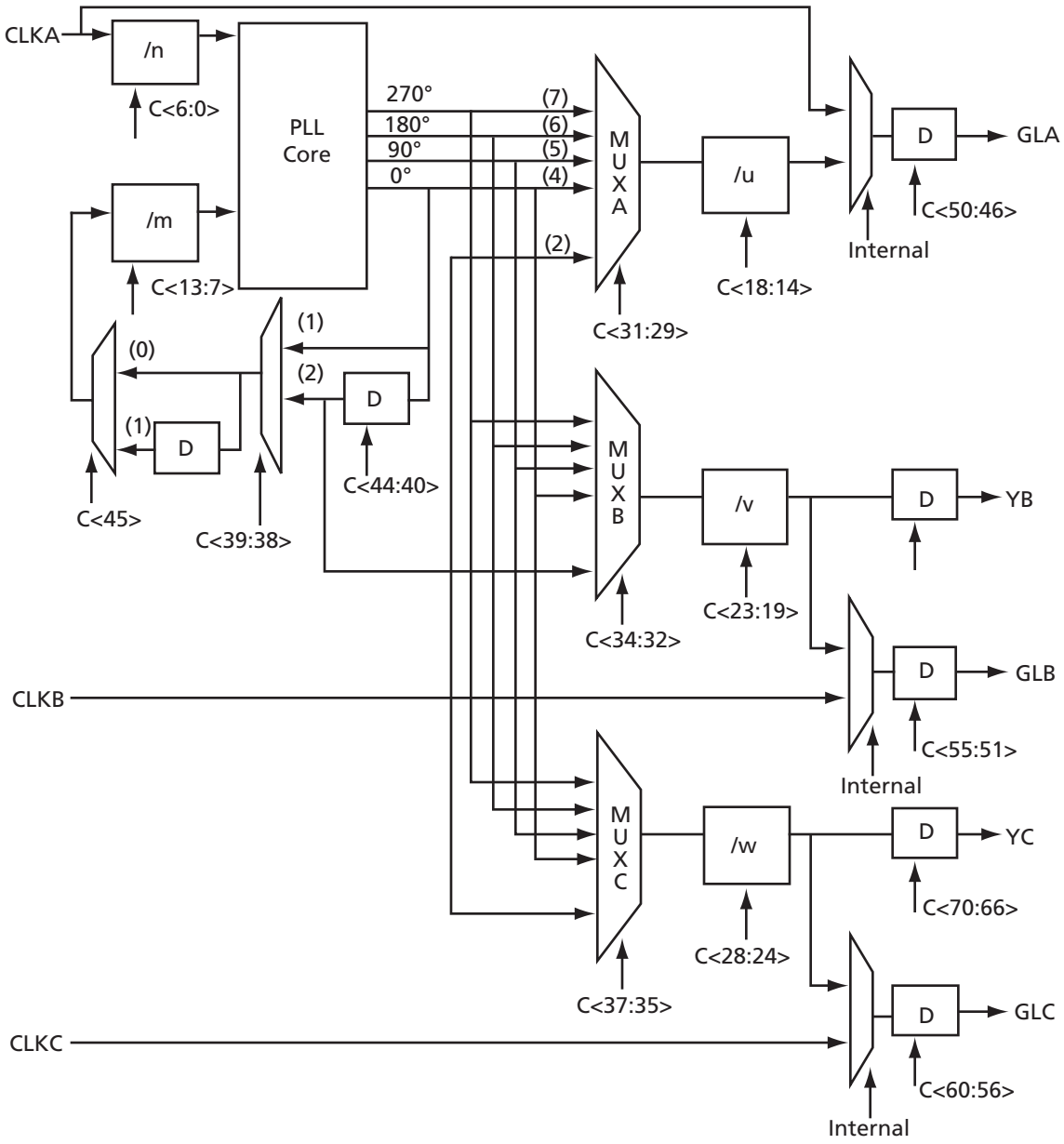


***Figure 3-21* • CCC Block Control Bits – Graphical Representation of Assignments**

Loading the Configuration Register

The most important part of CCC dynamic configuration is to load the shift register properly with the configuration bits. There are different ways to access and load the configuration shift register:

- JTAG interface
- Logic core
- Specific I/O tiles

JTAG Interface

The JTAG interface requires no additional I/O pins. The JTAG TAP controller is used to control the loading of the CCC configuration shift register.

Low-power flash devices provide a user interface macro between the JTAG pins and the device core logic. This macro is called UJTAG. A user should instantiate the UJTAG macro in his design to access the configuration register ports via the JTAG pins.

For more information on CCC dynamic reconfiguration using UJTAG, refer to *UJTAG Applications in Actel's Low-Power Flash Devices.*

Logic Core

If the logic core is employed, the user must design a module to provide the configuration data and control the shifting and updating of the CCC configuration shift register. In effect, this is a user-designed TAP controller, which requires additional chip resources.

Specific I/O Tiles

If specific I/O tiles are used for configuration, the user must provide the external equivalent of a TAP controller. This does not require additional core resources but does use pins.

Shifting the Configuration Data

To enter a new configuration, all 81 bits must shift in via SDIN. After all bits are shifted, SSHIFT must go LOW and SUPDATE HIGH to enable the new configuration. For simulation purposes, bits <71:73> and <77:80> are "don't care."

The SUPDATE signal must be LOW during any clock cycle where SSHIFT is active. After SUPDATE is asserted, it must go back to the LOW state until a new update is required.

PLL Configuration Bits Description

Table 3-7 • **Configuration Bit Descriptions for the CCC Blocks**

Config. Bits	Signal	Name	Description
<88:87>	GLMUXCFG [1:0][1]	NGMUX configuration	The configuration bits specify the input clocks to the NGMUX (refer to Table 3-16 on page 3-32).[2]
86	OCDIVHALF[1]	Division by half	When the PLL is bypassed, the 100 MHz RC oscillator can be divided by the divider factor in Table 3-17 on page 3-33.
85	OBDIVHALF[1]	Division by half	When the PLL is bypassed, the 100 MHz RC oscillator can be divided by a 0.5 factor (refer to Table 3-17 on page 3-33).
84	OADIVHALF[1]	Division by half	When the PLL is bypassed, the 100 MHz RC oscillator can be divided by certain 0.5 factor (refer to Table 3-15 on page 3-32).

Notes:

1. *The <88:81> configuration bits are only for the Fusion dynamic CCC.*
2. *This value depends on the input clock source, so Layout must complete before these bits can be set. After completing Layout in Designer, generate the "CCC_Configuration" report by choosing* ***Tools*** *>* ***Report*** *>* ***CCC_Configuration****. The report contains the appropriate settings for these bits.*

Table 3-7 • **Configuration Bit Descriptions for the CCC Blocks (continued)**

Config. Bits	Signal	Name	Description
83	RXCSEL[1]	CLKC input selection	Select the CLKC input clock source between RC oscillator and crystal oscillator (refer to Table 3-15 on page 3-32).[2]
82	RXBSEL[1]	CLKB input selection	Select the CLKB input clock source between RC oscillator and crystal oscillator (refer to Table 3-15 on page 3-32).[2]
81	RXASEL[1]	CLKA input selection	Select the CLKA input clock source between RC oscillator and crystal oscillator (refer to Table 3-15 on page 3-32).[2]
80	RESETEN	Reset Enable	Enables (active high) the synchronization of PLL output dividers after dynamic reconfiguration (SUPDATE). The Reset Enable signal is READ-ONLY and should not be modified via dynamic reconfiguration.
79	DYNCSEL	Clock Input C Dynamic Select	Configures clock input C to be sent to GLC for dynamic control.[2]
78	DYNBSEL	Clock Input B Dynamic Select	Configures clock input B to be sent to GLB for dynamic control.[2]
77	DYNASEL	Clock Input A Dynamic Select	Configures clock input A for dynamic PLL configuration.[2]
<76:74>	VCOSEL[2:0]	VCO Gear Control	Three-bit VCO Gear Control for four frequency ranges (refer to Table 3-18 on page 3-33 and Table 3-19 on page 3-33).
73	STATCSEL	MUX Select on Input C	MUX selection for clock input C[2]
72	STATBSEL	MUX Select on Input B	MUX selection for clock input B[2]
71	STATASEL	MUX Select on Input A	MUX selection for clock input A[2]
<70:66>	DLYC[4:0]	YC Output Delay	Sets the output delay value for YC.
<65:61>	DLYB[4:0]	YB Output Delay	Sets the output delay value for YB.
<60:56>	DLYGLC[4:0]	GLC Output Delay	Sets the output delay value for GLC.
<55:51>	DLYGLB[4:0]	GLB Output Delay	Sets the output delay value for GLB.
<50:46>	DLYGLA[4:0]	Primary Output Delay	Primary GLA output delay
45	XDLYSEL	System Delay Select	When selected, inserts System Delay in the feedback path in Figure 3-19 on page 3-24.
<44:40>	FBDLY[4:0]	Feedback Delay	Sets the feedback delay value for the feedback element in Figure 3-19 on page 3-24.
<39:38>	FBSEL[1:0]	Primary Feedback Delay Select	Controls the feedback MUX: no delay, include programmable delay element, or use external feedback.
<37:35>	OCMUX[2:0]	Secondary 2 Output Select	Selects from the VCO's four phase outputs for GLC/YC.

Notes:

1. *The <88:81> configuration bits are only for the Fusion dynamic CCC.*
2. *This value depends on the input clock source, so Layout must complete before these bits can be set. After completing Layout in Designer, generate the "CCC_Configuration" report by choosing **Tools** > **Report** > **CCC_Configuration**. The report contains the appropriate settings for these bits.*

Table 3-7 • **Configuration Bit Descriptions for the CCC Blocks (continued)**

Config. Bits	Signal	Name	Description
<34:32>	OBMUX[2:0]	Secondary 1 Output Select	Selects from the VCO's four phase outputs for GLB/YB.
<31:29>	OAMUX[2:0]	GLA Output Select	Selects from the VCO's four phase outputs for GLA.
<28:24>	OCDIV[4:0]	Secondary 2 Output Divider	Sets the divider value for the GLC/YC outputs. Also known as divider *w* in Figure 3-19 on page 3-24. The divider value will be OCDIV[4:0] + 1.
<23:19>	OBDIV[4:0]	Secondary 1 Output Divider	Sets the divider value for the GLB/YB outputs. Also known as divider *v* in Figure 3-19 on page 3-24. The divider value will be OBDIV[4:0] + 1.
<18:14>	OADIV[4:0]	Primary Output Divider	Sets the divider value for the GLA output. Also known as divider *u* in Figure 3-19 on page 3-24. The divider value will be OADIV[4:0] + 1.
<13:7>	FBDIV[6:0]	Feedback Divider	Sets the divider value for the PLL core feedback. Also known as divider *m* in Figure 3-19 on page 3-24. The divider value will be FBDIV[6:0] + 1.
<6:0>	FINDIV[6:0]	Input Divider	Input Clock Divider (/n). Sets the divider value for the input delay on CLKA. The divider value will be FINDIV[6:0] + 1.

Notes:

1. *The <88:81> configuration bits are only for the Fusion dynamic CCC.*
2. *This value depends on the input clock source, so Layout must complete before these bits can be set. After completing Layout in Designer, generate the "CCC_Configuration" report by choosing* ***Tools*** *>* ***Report*** *>* ***CCC_Configuration****. The report contains the appropriate settings for these bits.*

Table 3-8 to Table 3-14 on page 3-32 provide descriptions of the configuration data for the configuration bits.

Table 3-8 • **Input Clock Divider, FINDIV[6:0] (/n)**

FINDIV<6:0> State	Divisor	New Frequency Factor
0	1	1.00000
1	2	0.50000
⋮	⋮	⋮
127	128	0.0078125

Table 3-9 • **Feedback Clock Divider, FBDIV[6:0] (/m)**

FBDIV<6:0> State	Divisor	New Frequency Factor
0	1	1
1	2	2
⋮	⋮	⋮
127	128	128

Table 3-10 • Output Frequency Dividers
***A* Output Divider, OADIV <4:0> (/u);**
***B* Output Divider, OBDIV <4:0> (/v);**
***C* Output Divider, OCDIV <4:0> (/w)**

OADIV<4:0>; OBDIV<4:0>; CDIV<4:0> State	Divisor	New Frequency Factor
0	1	1.00000
1	2	0.50000
⋮	⋮	⋮
31	32	0.03125

Table 3-11 • MUXA, MUXB, MUXC

OAMUX<2:0>; OBMUX<2:0>; OCMUX<2:0> State	MUX Input Selected
0	None. Six-input MUX and PLL are bypassed. Clock passes only through global MUX and goes directly into HC ribs.
1	Not available
2	PLL feedback delay line output
3	Not used
4	PLL VCO 0° phase shift
5	PLL VCO 90° phase shift
6	PLL VCO 180° phase shift
7	PLL VCO 270° phase shift

Table 3-12 • 2-Bit Feedback MUX

FBSEL<1:0> State	MUX Input Selected
0	Ground. Used for power-down mode in power-down logic block.
1	PLL VCO 0° phase shift
2	PLL delayed VCO 0° phase shift
3	N/A

Table 3-13 • Programmable Delay Selection for Feedback Delay and Secondary Core Output Delays

FBDLY<4:0>; DLYYB<4:0>; DLYYC<4:0> State	Delay Value
0	Typical delay = 600 ps
1	Typical delay = 760 ps
2	Typical delay = 920 ps
⋮	⋮
31	Typical delay = 5.56 ns

Table 3-14 • **Programmable Delay Selection for Global Clock Output Delays**

DLYGLA<4:0>; DLYGLB<4:0>; DLYGLC<4:0> State	Delay Value
0	Typical delay = 225 ps
1	Typical delay = 760 ps
2	Typical delay = 920 ps
⋮	⋮
31	Typical delay = 5.56 ns

Table 3-15 • **Fusion Dynamic CCC Clock Source Selection**

RXASEL	DYNASEL	Source of CLKA
1	0	RC Oscillator
1	1	Crystal Oscillator
RXBSEL	**DYNBSEL**	**Source of CLKB**
1	0	RC Oscillator
1	1	Crystal Oscillator
RXBSEL	**DYNCSEL**	**Source of CLKC**
1	0	RC Oscillator
1	1	Crystal Oscillator

Table 3-16 • **Fusion Dynamic CCC NGMUX Configuration**

GLMUXCFG<1:0>	NGMUX Select Signal	Supported Input Clocks to NGMUX
00	0	GLA
	1	GLC
01	0	GLA
	1	GLINT
10	0	GLC
	1	GLINT

Table 3-17 • **Fusion Dynamic CCC Division by Half Configuration**

OADIVHALF / OBDIVHALF / OCDIVHALF	OADIV<4:0> / OBDIV<4:0> / OCDIV<4:0> (in decimal)	Divider Factor	Input Clock Frequency	Output Clock Frequency (MHz)
1	2	1.5	100 MHz RC Oscillator	66.7
	4	2.5		40.0
	6	3.5		28.6
	8	4.5		22.2
	10	5.5		18.2
	12	6.5		15.4
	14	7.5		13.3
	16	8.5		11.8
	18	9.5		10.5
	20	10.5		9.5
	22	11.5		8.7
	24	12.5		8.0
	26	13.5		7.4
	28	14.5		6.9
0	0–31	1–32	Other Clock Sources	Depends on other divider settings

Table 3-18 • **Configuration Bit <76:75> / VCOSEL<2:1> Selection for All Families**

	VCOSEL[2:1]							
	00		01		10		11	
Voltage	Min. (MHz)	Max. (MHz)	Min. (MHz)	Max. (MHz)	Min. (MHz)	Max. (MHz)	Min. (MHz)	Max. (MHz)
IGLOO and IGLOO PLUS								
1.2 V ± 5%	24	35	30	70	60	140	135	160
1.5 V ± 5%	24	43.75	30	87.5	60	175	135	250
ProASIC3L, RT ProASIC3, and Military ProASIC3/L								
1.2 V ± 5%	24	35	30	70	60	140	135	250
1.5 V ± 5%	24	43.75	30	70	60	175	135	350
ProASIC3 and Fusion								
1.5 V ± 5%	24	43.75	33.75	87.5	67.5	175	135	350

Table 3-19 • **Configuration Bit <74> / VCOSEL<0> Selection for All Families**

VCOSEL[0]	Description
0	Fast PLL lock acquisition time with high tracking jitter. Refer to the corresponding datasheet for specific value and definition.
1	Slow PLL lock acquisition time with low tracking jitter. Refer to the corresponding datasheet for specific value and definition.

Software Configuration

SmartGen automatically generates the desired CCC functional block by configuring the control bits, and allows the user to select two CCC modes: Static PLL and Delayed Clock (CLKDLY).

Static PLL Configuration

The newly implemented Visual PLL Configuration Wizard feature provides the user a quick and easy way to configure the PLL with the desired settings (Figure 3-22). The user can invoke SmartGen to set the parameters and generate the netlist file with the appropriate flash configuration bits set for the CCCs. As mentioned in "PLL Macro Block Diagram" on page 3-9, the input reference clock CLKA can be configured to be driven by Hardwired I/O, External I/O, or Core Logic. The user enters the desired settings for all the parameters (output frequency, output selection, output phase adjustment, clock delay, feedback delay, and system delay). Notice that the actual values (divider values, output frequency, delay values, and phase) are shown to aid the user in reaching the desired design frequency in real time. These values are typical-case data. Best- and worst-case data can be observed through static timing analysis in SmartTime within Designer.

For dynamic configuration, the CCC parameters are defined using either the external JTAG port or an internally defined serial interface via the built-in dynamic shift register. This feature provides the ability to compensate for changes in the external environment.

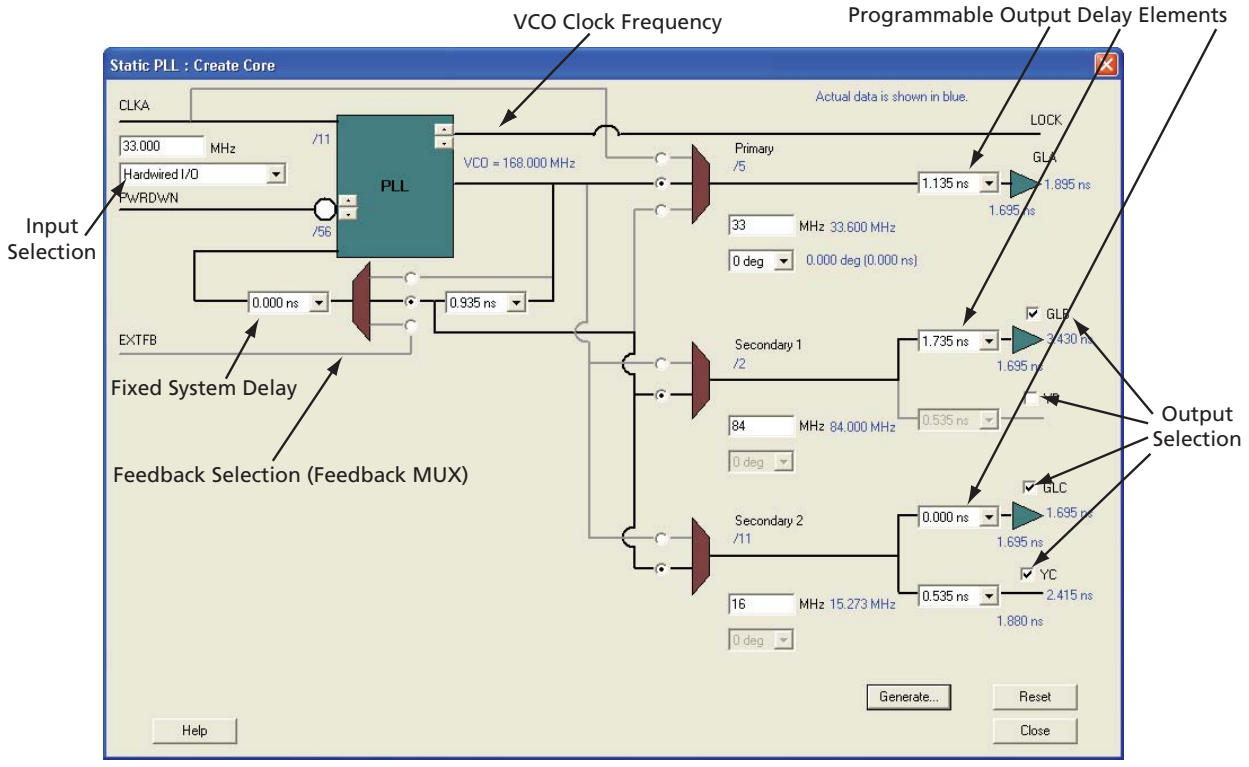


Figure 3-22 • **Visual PLL Configuration Wizard**

Feedback Configuration

The PLL provides both internal and external feedback delays. Depending on the configuration, various combinations of feedback delays can be achieved.

Internal Feedback Configuration

This configuration essentially sets the feedback multiplexer to route the VCO output of the PLL core as the input to the feedback of the PLL. The feedback signal can be processed with the fixed system and the adjustable feedback delay, as shown in Figure 3-23. The dividers are automatically configured by SmartGen based on the user input.

Indicated below is the System Delay pull-down menu. The System Delay can be bypassed by setting it to 0. When set, it adds a 2 ns delay to the feedback path (which results in delay advancement of the output clock by 2 ns).

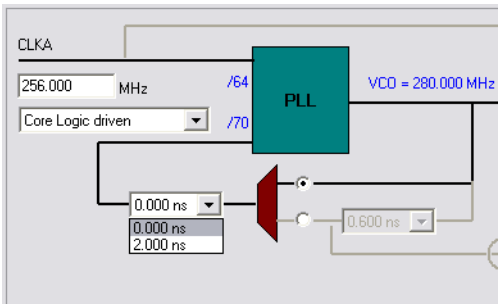


***Figure 3-23* • Internal Feedback with Selectable System Delay**

Figure 3-24 shows the controllable Feedback Delay. If set properly in conjunction with the fixed System Delay, the total output delay can be advanced significantly.

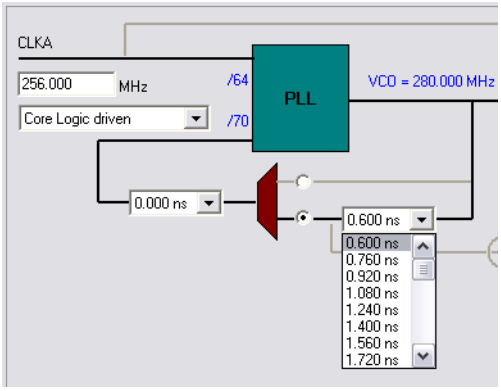


***Figure 3-24* • Internal Feedback with Selectable Feedback Delay**

External Feedback Configuration

For certain applications, such as those requiring generation of PCB clocks that must be matched with existing board delays, it is useful to implement an external feedback, EXTFB. The Phase Detector of the PLL core will receive CLKA and EXTFB as inputs. EXTFB may be processed by the fixed System Delay element as well as the *M* divider element. The EXTFB option is currently not supported.

After setting all the required parameters, users can generate one or more PLL configurations with HDL or EDIF descriptions by clicking the **Generate** button. SmartGen gives the option of saving session results and messages in a log file:

```
****************
Macro Parameters
****************

Name                            : test_pll
Family                          : ProASIC3E
Output Format                   : VHDL
Type                            : Static PLL
Input Freq(MHz)                 : 10.000
CLKA Source                     : Hardwired I/O
Feedback Delay Value Index      : 1
Feedback Mux Select             : 2
XDLY Mux Select                 : No
Primary Freq(MHz)               : 33.000
Primary PhaseShift              : 0
Primary Delay Value Index       : 1
Primary Mux Select              : 4
Secondary1 Freq(MHz)            : 66.000
Use GLB                         : YES
Use YB                          : YES
GLB Delay Value Index           : 1
YB Delay Value Index            : 1
Secondary1 PhaseShift           : 0
Secondary1 Mux Select           : 4
Secondary2 Freq(MHz)            : 101.000
Use GLC                         : YES
Use YC                          : NO
GLC Delay Value Index           : 1
YC Delay Value Index            : 1
Secondary2 PhaseShift           : 0
Secondary2 Mux Select           : 4

…
…
…

Primary Clock frequency 33.333
Primary Clock Phase Shift 0.000
Primary Clock Output Delay from CLKA 0.180

Secondary1 Clock frequency 66.667
Secondary1 Clock Phase Shift 0.000
Secondary1 Clock Global Output Delay from CLKA 0.180
Secondary1 Clock Core Output Delay from CLKA 0.625

Secondary2 Clock frequency 100.000
Secondary2 Clock Phase Shift 0.000
Secondary2 Clock Global Output Delay from CLKA 0.180
```

Below is an example Verilog HDL description of a legal PLL core configuration generated by SmartGen:

```
module test_pll(POWERDOWN,CLKA,LOCK,GLA);
```

```
input POWERDOWN, CLKA;
output  LOCK, GLA;

    wire VCC, GND;

    VCC VCC_1_net(.Y(VCC));
    GND GND_1_net(.Y(GND));
    PLL Core(.CLKA(CLKA), .EXTFB(GND), .POWERDOWN(POWERDOWN),
        .GLA(GLA), .LOCK(LOCK), .GLB(), .YB(), .GLC(), .YC(),
        .OADIV0(GND), .OADIV1(GND), .OADIV2(GND), .OADIV3(GND),
        .OADIV4(GND), .OAMUX0(GND), .OAMUX1(GND), .OAMUX2(VCC),
        .DLYGLA0(GND), .DLYGLA1(GND), .DLYGLA2(GND), .DLYGLA3(GND)
        , .DLYGLA4(GND), .OBDIV0(GND), .OBDIV1(GND), .OBDIV2(GND),
        .OBDIV3(GND), .OBDIV4(GND), .OBMUX0(GND), .OBMUX1(GND),
        .OBMUX2(GND), .DLYYB0(GND), .DLYYB1(GND), .DLYYB2(GND),
        .DLYYB3(GND), .DLYYB4(GND), .DLYGLB0(GND), .DLYGLB1(GND),
        .DLYGLB2(GND), .DLYGLB3(GND), .DLYGLB4(GND), .OCDIV0(GND),
        .OCDIV1(GND), .OCDIV2(GND), .OCDIV3(GND), .OCDIV4(GND),
        .OCMUX0(GND), .OCMUX1(GND), .OCMUX2(GND), .DLYYC0(GND),
        .DLYYC1(GND), .DLYYC2(GND), .DLYYC3(GND), .DLYYC4(GND),
        .DLYGLC0(GND), .DLYGLC1(GND), .DLYGLC2(GND), .DLYGLC3(GND)
        , .DLYGLC4(GND), .FINDIV0(VCC), .FINDIV1(GND), .FINDIV2(
       VCC), .FINDIV3(GND), .FINDIV4(GND), .FINDIV5(GND),
        .FINDIV6(GND), .FBDIV0(VCC), .FBDIV1(GND), .FBDIV2(VCC),
        .FBDIV3(GND), .FBDIV4(GND), .FBDIV5(GND), .FBDIV6(GND),
        .FBDLY0(GND), .FBDLY1(GND), .FBDLY2(GND), .FBDLY3(GND),
        .FBDLY4(GND), .FBSEL0(VCC), .FBSEL1(GND), .XDLYSEL(GND),
        .VCOSEL0(GND), .VCOSEL1(GND), .VCOSEL2(GND));
    defparam Core.VCOFREQUENCY = 33.000;
endmodule
```

The "PLL Configuration Bits Description" section on page 3-28 provides descriptions of the PLL configuration bits for completeness. The configuration bits are shown as busses only for purposes of illustration. They will actually be broken up into individual pins in compilation libraries and all simulation models. For example, the FBSEL[1:0] bus will actually appear as pins FBSEL1 and FBSEL0. The setting of these select lines for the static PLL configuration is performed by the software and is completely transparent to the user.

Dynamic PLL Configuration

To generate a dynamically reconfigurable CCC, the user should select **Dynamic CCC** in the configuration section of the SmartGen GUI (Figure 3-25). This will generate both the CCC core and the configuration shift register / control bit MUX.

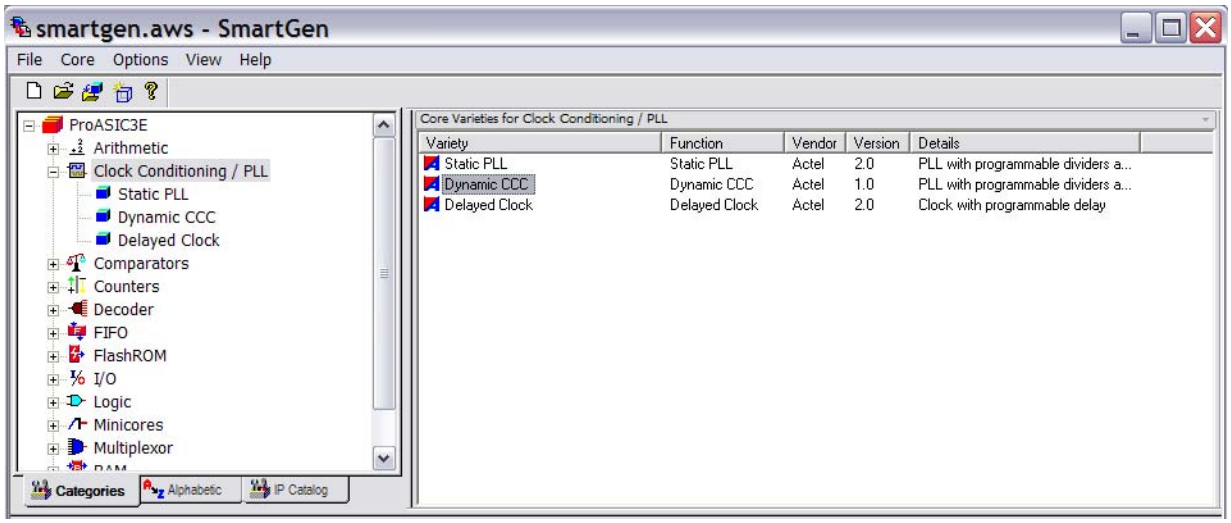


Figure 3-25 • **SmartGen GUI**

Even if dynamic configuration is selected in SmartGen, the user must still specify the static configuration data for the CCC (Figure 3-26). The specified static configuration is used whenever the MODE signal is set to LOW and the CCC is required to function in the static mode. The static configuration data can be used as the default behavior of the CCC where required.

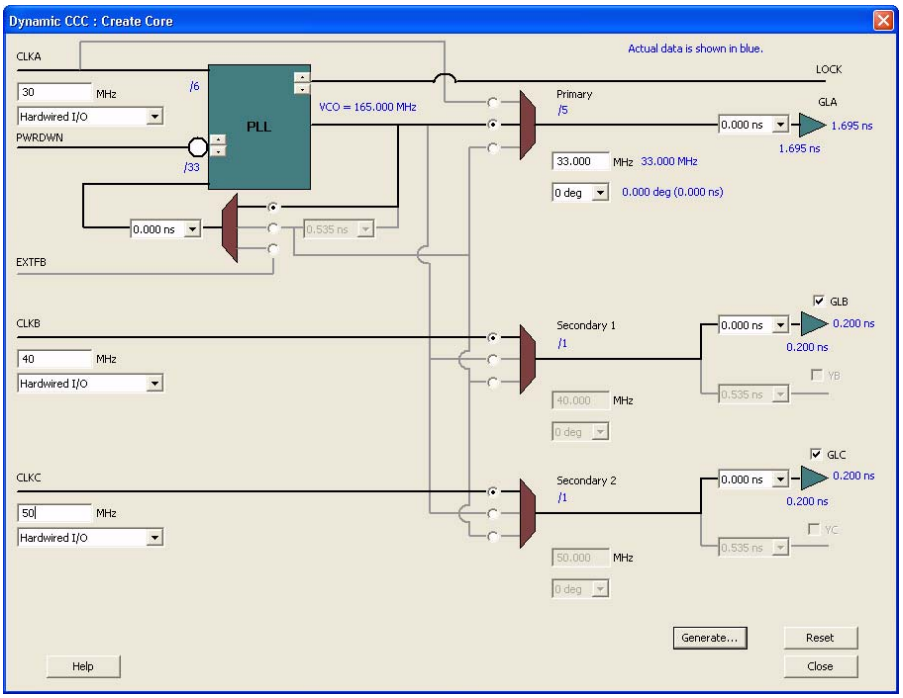


Figure 3-26 • **Dynamic CCC Configuration in SmartGen**

When SmartGen is used to define the configuration that will be shifted in via the serial interface, SmartGen prints out the values of the 81 configuration bits. For ease of use, several configuration bits are automatically inferred by SmartGen when the dynamic PLL core is generated; however, <71:73> (STATASEL, STATBSEL, STATCSEL) and <77:79> (DYNASEL, DYNBSEL, DYNCSEL) depend on the input clock source of the corresponding CCC. Users must first run Layout in Designer to determine the exact setting for these ports. After Layout is complete, generate the "CCC_Configuration" report by choosing **Tools** > **Reports** > **CCC_Configuration** in the Designer software. Refer to "PLL Configuration Bits Description" on page 3-28 for descriptions of the PLL configuration bits. For simulation purposes, bits <71:73> and <78:80> are "don't care." Therefore, it is strongly suggested that SmartGen be used to generate the correct configuration bit settings for the dynamic PLL core.

After setting all the required parameters, users can generate one or more PLL configurations with HDL or EDIF descriptions by clicking the **Generate** button. SmartGen gives the option of saving session results and messages in a log file:

```
****************
Macro Parameters
****************

Name                            : dyn_pll_hardio
Family                          : ProASIC3E
Output Format                   : VERILOG
Type                            : Dynamic CCC
Input Freq(MHz)                 : 30.000
CLKA Source                     : Hardwired I/O
Feedback Delay Value Index      : 1
Feedback Mux Select             : 1
XDLY Mux Select                 : No
Primary Freq(MHz)               : 33.000
Primary PhaseShift              : 0
Primary Delay Value Index       : 1
Primary Mux Select              : 4
Secondary1 Freq(MHz)            : 40.000
Use GLB                         : YES
Use YB                          : NO
GLB Delay Value Index           : 1
YB Delay Value Index            : 1
Secondary1 PhaseShift           : 0
Secondary1 Mux Select           : 0
Secondary1 Input Freq(MHz)      : 40.000
CLKB Source                     : Hardwired I/O
Secondary2 Freq(MHz)            : 50.000
Use GLC                         : YES
Use YC                          : NO
GLC Delay Value Index           : 1
YC Delay Value Index            : 1
Secondary2 PhaseShift           : 0
Secondary2 Mux Select           : 0
Secondary2 Input Freq(MHz)      : 50.000
CLKC Source                     : Hardwired I/O

Configuration Bits:
FINDIV[6:0]     0000101
FBDIV[6:0]      0100000
OADIV[4:0]      00100
OBDIV[4:0]      00000
OCDIV[4:0]      00000
OAMUX[2:0]      100
OBMUX[2:0]      000
OCMUX[2:0]      000
FBSEL[1:0]      01
FBDLY[4:0]      00000
XDLYSEL         0
DLYGLA[4:0]     00000
```

```
DLYGLB[4:0]     00000
DLYGLC[4:0]     00000
DLYYB[4:0]      00000
DLYYC[4:0]      00000
VCOSEL[2:0]     100

Primary Clock Frequency 33.000
Primary Clock Phase Shift 0.000
Primary Clock Output Delay from CLKA 1.695

Secondary1 Clock Frequency 40.000
Secondary1 Clock Phase Shift 0.000
Secondary1 Clock Global Output Delay from CLKB 0.200

Secondary2 Clock Frequency 50.000
Secondary2 Clock Phase Shift 0.000
Secondary2 Clock Global Output Delay from CLKC 0.200

#####################################
# Dynamic Stream Data
#####################################
-------------------------------------
|NAME    |SDIN     |VALUE   |TYPE     |
-------------------------------------
|FINDIV  |[6:0]    |0000101 |EDIT     |
|FBDIV   |[13:7]   |0100000 |EDIT     |
|OADIV   |[18:14]  |00100   |EDIT     |
|OBDIV   |[23:19]  |00000   |EDIT     |
|OCDIV   |[28:24]  |00000   |EDIT     |
|OAMUX   |[31:29]  |100     |EDIT     |
|OBMUX   |[34:32]  |000     |EDIT     |
|OCMUX   |[37:35]  |000     |EDIT     |
|FBSEL   |[39:38]  |01      |EDIT     |
|FBDLY   |[44:40]  |00000   |EDIT     |
|XDLYSEL |[45]     |0       |EDIT     |
|DLYGLA  |[50:46]  |00000   |EDIT     |
|DLYGLB  |[55:51]  |00000   |EDIT     |
|DLYGLC  |[60:56]  |00000   |EDIT     |
|DLYYB   |[65:61]  |00000   |EDIT     |
|DLYYC   |[70:66]  |00000   |EDIT     |
|STATASEL|[71]     |X       |MASKED   |
|STATBSEL|[72]     |X       |MASKED   |
|STATCSEL|[73]     |X       |MASKED   |
|VCOSEL  |[76:74]  |100     |EDIT     |
|DYNASEL |[77]     |X       |MASKED   |
|DYNBSEL |[78]     |X       |MASKED   |
|DYNCSEL |[79]     |X       |MASKED   |
|RESETEN |[80]     |1       |READONLY |
```

Below is the resultant Verilog HDL description of a legal dynamic PLL core configuration generated by SmartGen:

```
module dyn_pll_macro(POWERDOWN, CLKA, LOCK, GLA, GLB, GLC, SDIN, SCLK, SSHIFT, SUPDATE,
  MODE, SDOUT, CLKB, CLKC);

input POWERDOWN, CLKA;
output  LOCK, GLA, GLB, GLC;
input  SDIN, SCLK, SSHIFT, SUPDATE, MODE;
output  SDOUT;
input  CLKB, CLKC;

  wire VCC, GND;

  VCC VCC_1_net(.Y(VCC));
  GND GND_1_net(.Y(GND));
```

```
  DYNCCC Core(.CLKA(CLKA), .EXTFB(GND), .POWERDOWN(POWERDOWN), .GLA(GLA), .LOCK(LOCK),
    .CLKB(CLKB), .GLB(GLB), .YB(), .CLKC(CLKC), .GLC(GLC), .YC(), .SDIN(SDIN),
    .SCLK(SCLK), .SSHIFT(SSHIFT), .SUPDATE(SUPDATE), .MODE(MODE), .SDOUT(SDOUT),
    .OADIV0(GND), .OADIV1(GND), .OADIV2(VCC), .OADIV3(GND), .OADIV4(GND), .OAMUX0(GND),
    .OAMUX1(GND), .OAMUX2(VCC), .DLYGLA0(GND), .DLYGLA1(GND), .DLYGLA2(GND),
    .DLYGLA3(GND), .DLYGLA4(GND), .OBDIV0(GND), .OBDIV1(GND), .OBDIV2(GND),
    .OBDIV3(GND), .OBDIV4(GND), .OBMUX0(GND), .OBMUX1(GND), .OBMUX2(GND), .DLYYB0(GND),
    .DLYYB1(GND), .DLYYB2(GND), .DLYYB3(GND), .DLYYB4(GND), .DLYGLB0(GND),
    .DLYGLB1(GND), .DLYGLB2(GND), .DLYGLB3(GND), .DLYGLB4(GND), .OCDIV0(GND),
    .OCDIV1(GND), .OCDIV2(GND), .OCDIV3(GND), .OCDIV4(GND), .OCMUX0(GND), .OCMUX1(GND),
    .OCMUX2(GND), .DLYYC0(GND), .DLYYC1(GND), .DLYYC2(GND), .DLYYC3(GND), .DLYYC4(GND),
    .DLYGLC0(GND), .DLYGLC1(GND), .DLYGLC2(GND), .DLYGLC3(GND), .DLYGLC4(GND),
    .FINDIV0(VCC), .FINDIV1(GND), .FINDIV2(VCC), .FINDIV3(GND), .FINDIV4(GND),
    .FINDIV5(GND), .FINDIV6(GND), .FBDIV0(GND), .FBDIV1(GND), .FBDIV2(GND),
    .FBDIV3(GND), .FBDIV4(GND), .FBDIV5(VCC), .FBDIV6(GND), .FBDLY0(GND), .FBDLY1(GND),
    .FBDLY2(GND), .FBDLY3(GND), .FBDLY4(GND), .FBSEL0(VCC), .FBSEL1(GND),
    .XDLYSEL(GND), .VCOSEL0(GND), .VCOSEL1(GND), .VCOSEL2(VCC));
  defparam Core.VCOFREQUENCY = 165.000;

endmodule
```

Delayed Clock Configuration

The CLKDLY macro can be generated with the desired delay and input clock source (Hardwired I/O, External I/O, or Core Logic), as in Figure 3-27.

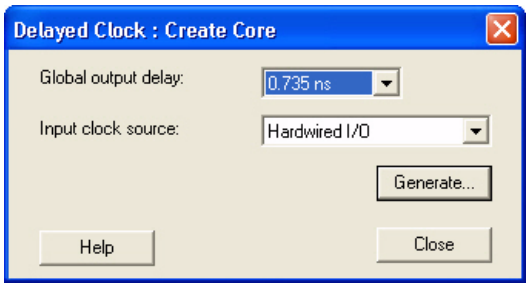


***Figure 3-27* • Delayed Clock Configuration Dialog Box**

After setting all the required parameters, users can generate one or more PLL configurations with HDL or EDIF descriptions by clicking the **Generate** button. SmartGen gives the option of saving session results and messages in a log file:

```
***************
Macro Parameters
***************

Name                            : delay_macro
Family                          : ProASIC3
Output Format                   : Verilog
Type                            : Delayed Clock
Delay Index                     : 2
CLKA Source                     : Hardwired I/O

Total Clock Delay = 0.935 ns.

The resultant CLKDLY macro Verilog netlist is as follows:

module delay_macro(GL,CLK);

output GL;
input  CLK;
```

```
  wire VCC, GND;

  VCC VCC_1_net(.Y(VCC));
  GND GND_1_net(.Y(GND));
  CLKDLY Inst1(.CLK(CLK), .GL(GL), .DLYGL0(VCC), .DLYGL1(GND), .DLYGL2(VCC),
    .DLYGL3(GND), .DLYGL4(GND));

endmodule
```

Detailed Usage Information

Clock Frequency Synthesis

Deriving clocks of various frequencies from a single reference clock is known as frequency synthesis. The PLL has an input frequency range from 1.5 to 350 MHz. This frequency is automatically divided down to a range between 1.5 MHz and 5.5 MHz by input dividers (not shown in Figure 3-18 on page 3-23) between PLL macro inputs and PLL phase detector inputs. The VCO output is capable of an output range from 24 to 350 MHz. With dividers before the input to the PLL core and following the VCO outputs, the VCO output frequency can be divided to provide the final frequency range from 0.75 to 350 MHz. Using SmartGen, the dividers are automatically set to achieve the closest possible matches to the specified output frequencies.

Users should be cautious when selecting the desired PLL input and output frequencies and the I/O buffer standard used to connect to the PLL input and output clocks. Depending on the I/O standards used for the PLL input and output clocks, the I/O frequencies have different maximum limits. Refer to the family datasheets for specifications of maximum I/O frequencies for supported I/O standards. Desired PLL input or output frequencies will not be achieved if the selected frequencies are higher than the maximum I/O frequencies allowed by the selected I/O standards. Users should be careful when selecting the I/O standards used for PLL input and output clocks. Performing post-layout simulation can help detect this type of error, which will be identified with pulse width violation errors. Users are strongly encouraged to perform post-layout simulation to ensure the I/O standard used can provide the desired PLL input or output frequencies. Users can also choose to cascade PLLs together to achieve the high frequencies needed for their applications. Details of cascading PLLs are discussed in the "Cascading CCCs" section on page 3-47.

In SmartGen, the actual generated frequency (under typical operating conditions) will be displayed beside the requested output frequency value. This provides the ability to determine the exact frequency that can be generated by SmartGen, in real time. The log file generated by SmartGen is a useful tool in determining how closely the requested clock frequencies match the user specifications. For example, assume a user specifies 101 MHz as one of the secondary output frequencies. If the best output frequency that could be achieved were 100 MHz, the log file generated by SmartGen would indicate the actual generated frequency.

Simulation Verification

The integration of the generated PLL and CLKDLY modules is similar to any VHDL component or Verilog module instantiation in a larger design; i.e., there is no special requirement that users need to take into account to successfully synthesize their designs.

For simulation purposes, users need to refer to the VITAL or Verilog library that includes the functional description and associated timing parameters. Refer to the Software Tools section of the Actel website to obtain the family simulation libraries. If Actel Designer is installed, these libraries are stored in the following locations:

<Designer_Installation_Directory>\lib\vtl\95\proasic3.vhd

<Designer_Installation_Directory>\lib\vtl\95\proasic3e.vhd

<Designer_Installation_Directory>\lib\vlog\proasic3.v

<Designer_Installation_Directory>\lib\vlog\proasic3e.v

For Libero IDE users, there is no need to compile the simulation libraries, as they are conveniently pre-compiled in the Model*Sim*® Actel simulation tool.

The following is an example of a PLL configuration utilizing the clock frequency synthesis and clock delay adjustment features. The steps include generating the PLL core with SmartGen, performing simulation for verification with Model*Sim*, and performing static timing analysis with SmartTime in Designer.

Parameters of the example PLL configuration:

Input Frequency – 20 MHz

Primary Output Requirement – 20 MHz with clock advancement of 3.02 ns

Secondary 1 Output Requirement – 40 MHz with clock delay of 2.515 ns

Figure 3-28 shows the SmartGen settings. Notice that the overall delays are calculated automatically, allowing the user to adjust the delay elements appropriately to obtain the desired delays.

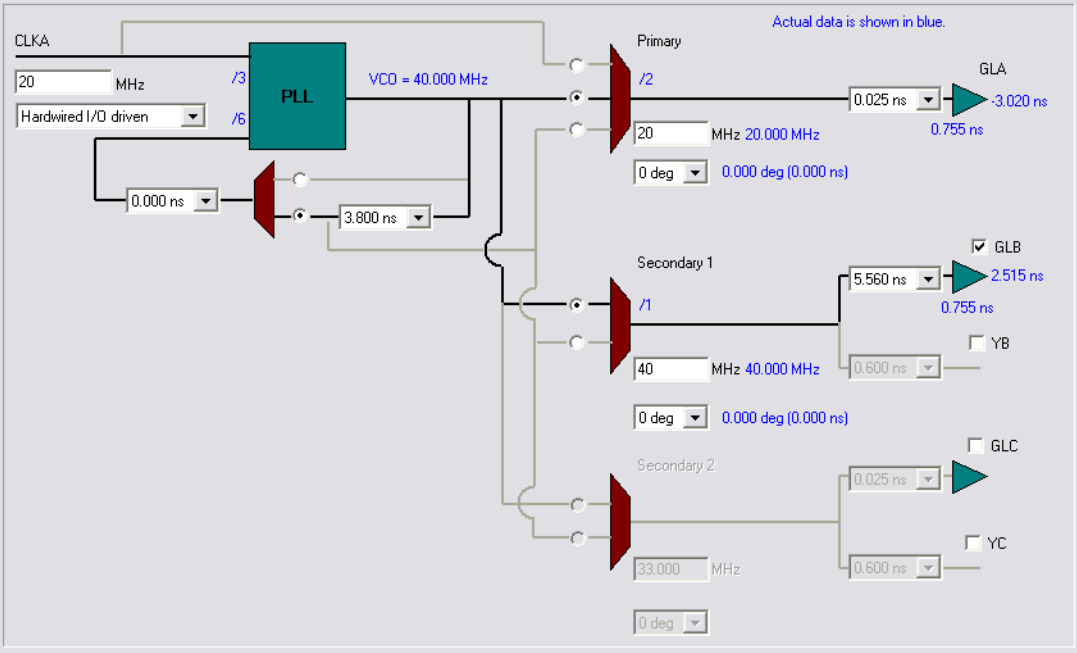


Figure 3-28 • **SmartGen Settings**

After confirming the correct settings, generate a structural netlist of the PLL and verify PLL core settings by checking the log file:

```
Name                              : test_pll_delays
Family                            : ProASIC3E
Output Format                     : VHDL
Type                              : Static PLL
Input Freq(MHz)                   : 20.000
CLKA Source                       : Hardwired I/O
Feedback Delay Value Index        : 21
Feedback Mux Select               : 2
XDLY Mux Select                   : No
Primary Freq(MHz)                 : 20.000
Primary PhaseShift                : 0
Primary Delay Value Index         : 1
Primary Mux Select                : 4
Secondary1 Freq(MHz)              : 40.000
Use GLB                           : YES
Use YB                            : NO
```

```
…
…
…
Primary Clock frequency 20.000
Primary Clock Phase Shift 0.000
Primary Clock Output Delay from CLKA -3.020

Secondary1 Clock frequency 40.000
Secondary1 Clock Phase Shift 0.000
Secondary1 Clock Global Output Delay from CLKA 2.515
```

Next, perform simulation in Model*Sim* to verify the correct delays. Figure 3-29 shows the simulation results. The delay values match those reported in the SmartGen PLL Wizard.

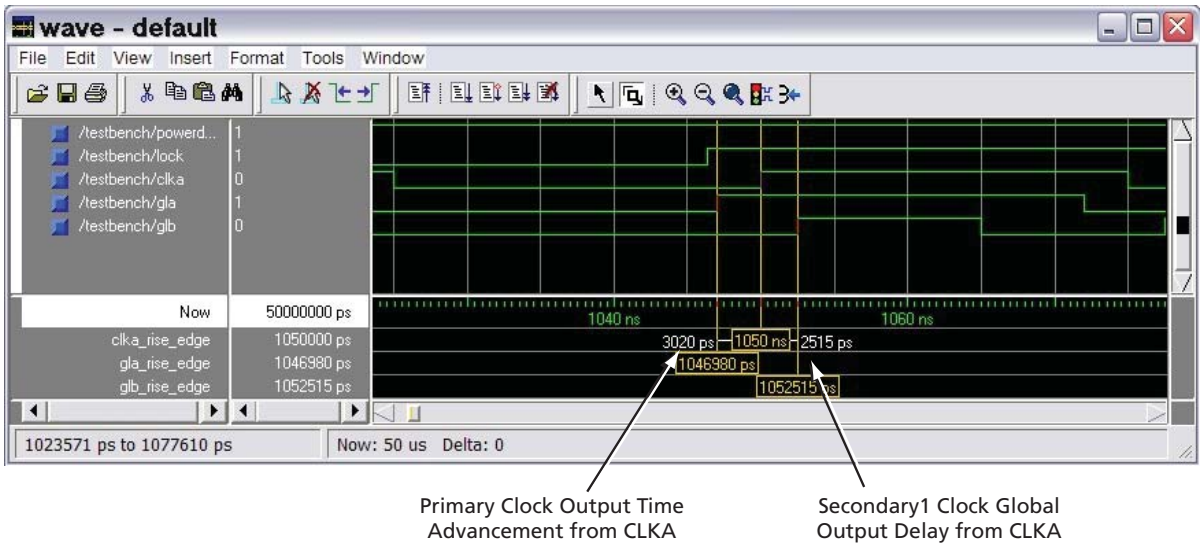
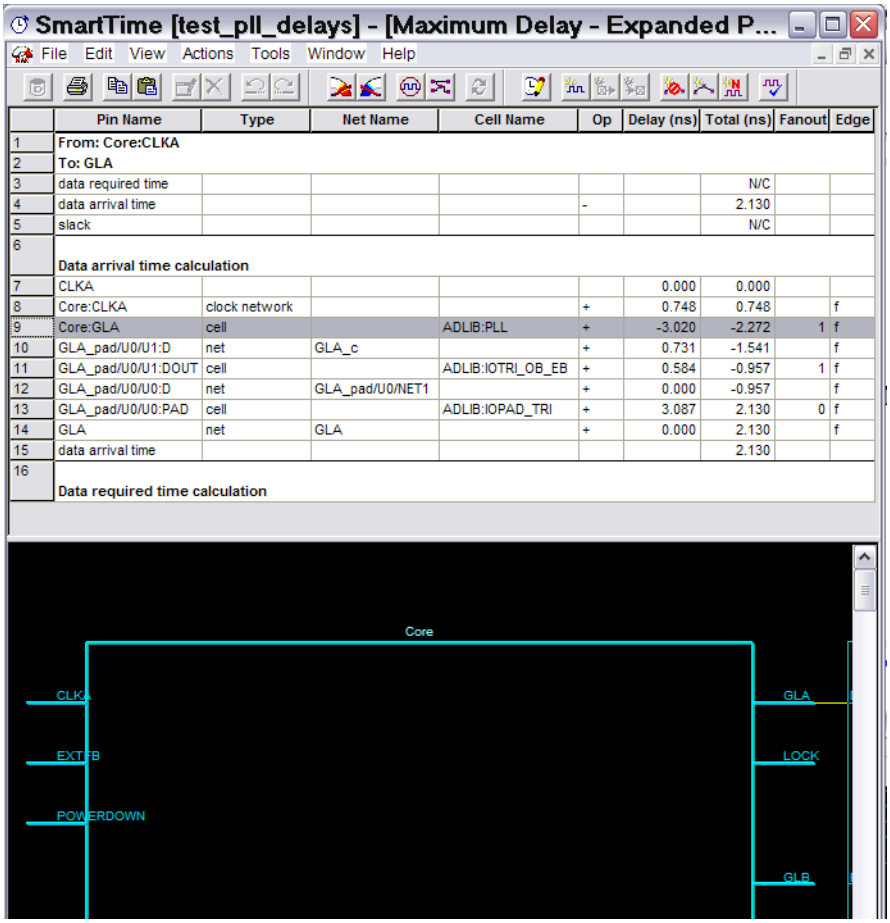


Figure 3-29 • **Model*Sim* Simulation Results**

The timing can also be analyzed using SmartTime in Designer. The user should import the synthesized netlist to Designer, perform Compile and Layout, and then invoke SmartTime. Go to **Tools** > **Options** and change the maximum delay operating conditions to **Typical Case**. Then expand the Clock-to-Out paths of GLA and GLB and the individual components of the path delays are shown. The path of GLA is shown in Figure 3-30 on page 3-45 displaying the same delay value.



	Pin Name	Type	Net Name	Cell Name	Op	Delay (ns)	Total (ns)	Fanout	Edge
1	From: Core:CLKA								
2	To: GLA								
3	data required time						N/C		
4	data arrival time				-		2.130		
5	slack						N/C		
6	Data arrival time calculation								
7	CLKA					0.000	0.000		
8	Core:CLKA	clock network			+	0.748	0.748		f
9	Core:GLA	cell		ADLIB:PLL	+	-3.020	-2.272	1	f
10	GLA_pad/U0/U1:D	net	GLA_c		+	0.731	-1.541		f
11	GLA_pad/U0/U1:DOUT	cell		ADLIB:IOTRI_OB_EB	+	0.584	-0.957	1	f
12	GLA_pad/U0/U0:D	net	GLA_pad/U0/NET1		+	0.000	-0.957		f
13	GLA_pad/U0/U0:PAD	cell		ADLIB:IOPAD_TRI	+	3.087	2.130	0	f
14	GLA	net	GLA		+	0.000	2.130		f
15	data arrival time						2.130		
16	Data required time calculation								

Figure 3-30 • **Static Timing Analysis Using SmartTime**

Place-and-Route Stage Considerations

Several considerations must be noted to properly place the CCC macros for layout.

For CCCs with clock inputs configured with the Hardwired I/O–Driven option:

- PLL macros must have the clock input pad coming from one of the GmA* locations.
- CLKDLY macros must have the clock input pad coming from one of the Global I/Os.

If a PLL with a Hardwired I/O input is used at a CCC location and a Hardwired I/O–Driven CLKDLY macro is used at the same CCC location, the clock input of the CLKDLY macro must be chosen from one of the GmB* or GmC* pin locations. If the PLL is not used or is an External I/O–Driven or Core Logic–Driven PLL, the clock input of the CLKDLY macro can be sourced from the GmA*, GmB*, or GmC* pin locations.

For CCCs with clock inputs configured with the External I/O–Driven option, the clock input pad can be assigned to any regular I/O location (IO******** pins). Note that since global I/O pins can also be used as regular I/Os, regardless of CCC function (CLKDLY or PLL), clock inputs can also be placed in any of these I/O locations.

By default, the Designer layout engine will place global nets in the design at one of the six chip globals. When the number of globals in the design is greater than six, the Designer layout engine will automatically assign additional globals to the quadrant global networks of the low-power flash devices. If the user wishes to decide which global signals should be assigned to chip globals (six available) and which to the quadrant globals (three per quadrant for a total of 12 available), the assignment can be achieved with PinEditor, ChipPlanner, or by importing a placement

constraint file. Layout will fail if the global assignments are not allocated properly. See the "Physical Constraints for Quadrant Clocks" section for information on assigning global signals to the quadrant clock networks.

Promoted global signals will be instantiated with CLKINT macros to drive these signals onto the global network. This is automatically done by Designer when the Auto-Promotion option is selected. If the user wishes to assign the signals to the quadrant globals instead of the default chip globals, this can done by using ChipPlanner, by declaring a physical design constraint (PDC), or by importing a PDC file.

Physical Constraints for Quadrant Clocks

If it is necessary to promote global clocks (CLKBUF, CLKINT, PLL, CLKDLY) to quadrant clocks, the user can define PDCs to execute the promotion. PDCs can be created using PDC commands (pre-compile) or the MultiView Navigator (MVN) interface (post-compile). The advantage of using the PDC flow over the MVN flow is that the Compile stage is able to automatically promote any regular net to a global net before assigning it to a quadrant. There are three options to place a quadrant clock using PDC commands:

- Place a clock core (not hardwired to an I/O) into a quadrant clock location.
- Place a clock core (hardwired to an I/O) into an I/O location (set_io) or an I/O module location (set_location) that drives a quadrant clock location.
- Assign a net driven by a regular net or a clock net to a quadrant clock using the following command:

  ```
  assign_local_clock -net <net name> -type quadrant <quadrant clock region>
  ```

  where

  `<net name>` is the name of the net assigned to the local user clock region.

  `<quadrant clock region>` defines which quadrant the net should be assigned to. Quadrant clock regions are defined as UL (upper left), UR (upper right), LL (lower left), and LR (lower right).

Note: If the net is a regular net, the software inserts a CLKINT buffer on the net.

For example:

```
assign_local_clock -net localReset -type quadrant UR
```

Keep in mind the following when placing quadrant clocks using MultiView Navigator:

Hardwired I/O–Driven CCCs

- Find the associated clock input port under the Ports tab, and place the input port at one of the Gmn* locations using PinEditor or I/O Attribute Editor, as shown in Figure 3-31.

Figure 3-31 • **Port Assignment for a CCC with Hardwired I/O Clock Input**

- Use quadrant global region assignments by finding the clock net associated with the CCC macro under the Nets tab and creating a quadrant global region for the net, as shown in Figure 3-32.

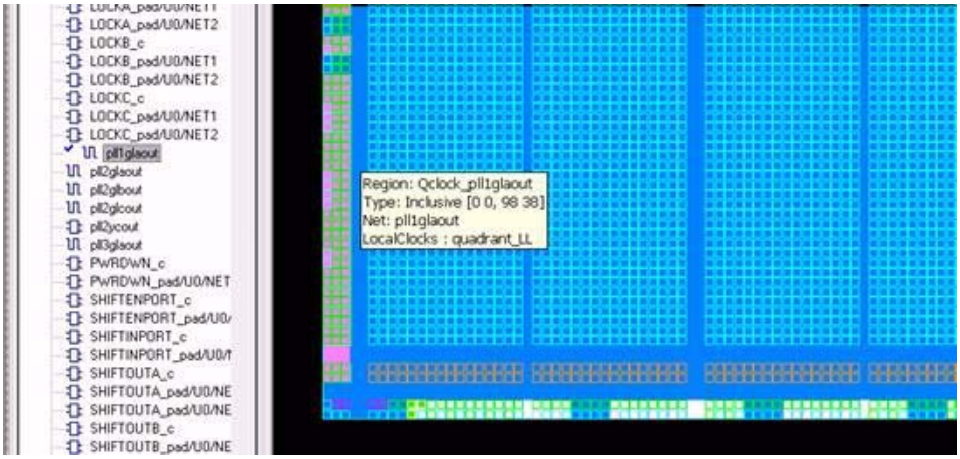


***Figure 3-32* • Quadrant Clock Assignment for a Global Net**

External I/O–Driven CCCs

The above-mentioned recommendation for proper layout techniques will ensure the correct assignment. It is possible that, especially with External I/O–Driven CCC macros, placement of the CCC macro in a desired location may not be achieved. For example, assigning an input port of an External I/O–Driven CCC near a particular CCC location does not guarantee global assignments to the desired location. This is because the clock inputs of External I/O–Driven CCCs can be assigned to any I/O location; therefore, it is possible that the CCC connected to the clock input will be routed to a location other than the one closest to the I/O location, depending on resource availability and placement constraints.

Clock Placer

The clock placer is a placement engine for low-power flash devices that places global signals on the chip global and quadrant global networks. Based on the clock assignment constraints for the chip global and quadrant global clocks, it will try to satisfy all constraints, as well as creating quadrant clock regions when necessary. If the clock placer fails to create the quadrant clock regions for the global signals, it will report an error and stop Layout.

The user must ensure that the constraints set to promote clock signals to quadrant global networks are valid.

Cascading CCCs

The CCCs in low-power flash devices can be cascaded. Cascading CCCs can help achieve more accurate PLL output frequency results than those achievable with a single CCC. In addition, this technique is useful when the user application requires the output clock of the PLL to be a multiple of the reference clock by an integer greater than the maximum feedback divider value of the PLL (divide by 128) to achieve the desired frequency.

For example, the user application may require a 280 MHz output clock using a 2 MHz input reference clock, as shown in Figure 3-33 on page 3-48.

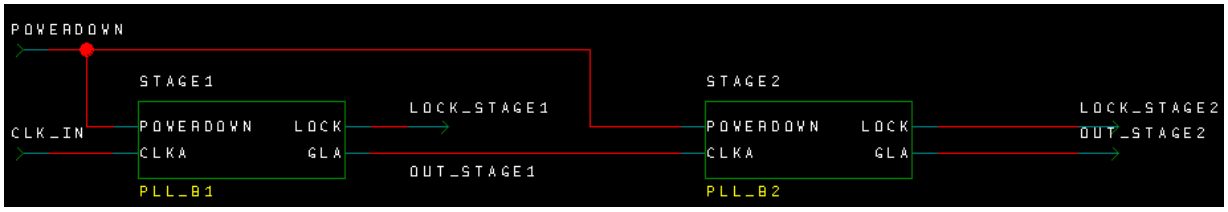


Figure 3-33 • **Cascade PLL Configuration**

Using internal feedback, we know from EQ 3-1 on page 3-25 that the maximum achievable output frequency from the primary output is

$$f_{GLA} = f_{CLKA} \times m / (n \times u) = 2\ MHz \times 128 / (1 \times 1) = 256\ MHz$$

EQ 3-5

Figure 3-34 shows the settings of the initial PLL. When configuring the initial PLL, specify the input to be either Hardwired I/O–Driven or External I/O–Driven. This generates a netlist with the initial PLL routed from an I/O. Do not specify the input to be Core Logic–Driven, as this prohibits the connection from the I/O pin to the input of the PLL.

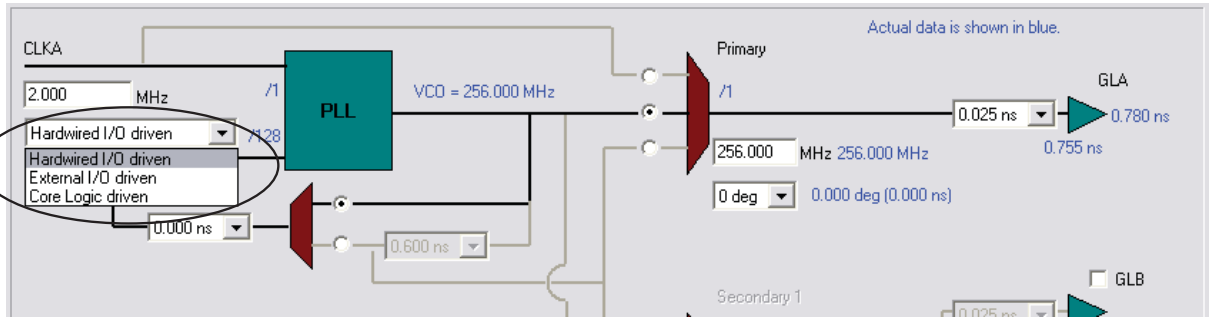


Figure 3-34 • **First-Stage PLL Showing Input of 2 MHz and Output of 256 MHz**

A second PLL can be connected serially to achieve the required frequency. EQ 3-1 on page 3-25 to EQ 3-3 on page 3-25 are extended as follows:

$$f_{GLA2} = f_{GLA} \times m_2 / (n_2 \times u_2) = f_{CLKA1} \times m_1 \times m_2 / (n_1 \times u_1 \times n_2 \times u_2) \text{ – Primary PLL Output Clock}$$

EQ 3-6

$$f_{GLB2} = f_{YB2} = f_{CLKA1} \times m_1 \times m_2 / (n_1 \times n_2 \times v_1 \times v_2) \text{ – Secondary 1 PLL Output Clock(s)}$$

EQ 3-7

$$f_{GLC2} = f_{YC2} = f_{CLKA1} \times m_1 \times m_2 / (n_1 \times n_2 \times w_1 \times w_2) \text{ – Secondary 2 PLL Output Clock(s)}$$

EQ 3-8

In the example, the final output frequency (f_{output}) from the primary output of the second PLL will be as follows (EQ 3-9):

$$f_{output} = f_{GLA2} = f_{GLA} \times m_2 / (n_2 \times u_2) = 256\ MHz \times 70 / (64 \times 1) = 280\ MHz$$

EQ 3-9

Figure 3-35 on page 3-49 shows the settings of the second PLL. When configuring the second PLL (or any subsequent-stage PLLs), specify the input to be Core Logic–Driven. This generates a netlist with the second PLL routed internally from the core. Do not specify the input to be Hardwired I/O–Driven or External I/O–Driven, as these options prohibit the connection from the output of the first PLL to the input of the second PLL.

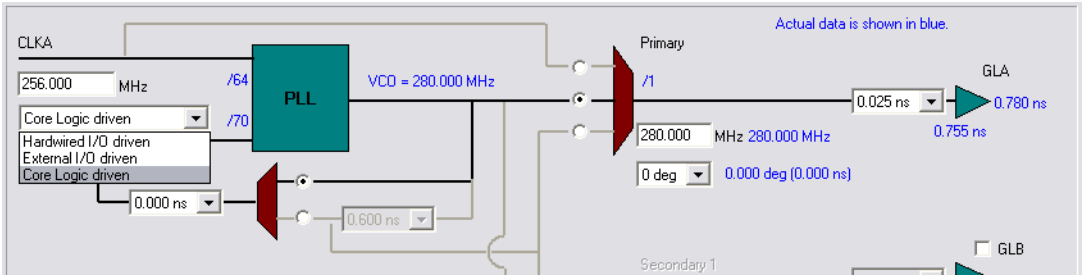


Figure 3-35 • **Second-Stage PLL Showing Input of 256 MHz from First Stage and Final Output of 280 MHz**

Figure 3-36 shows the simulation results, where the first PLL's output period is 3.9 ns (~256 MHz), and the stage 2 (final) output period is 3.56 ns (~280 MHz).

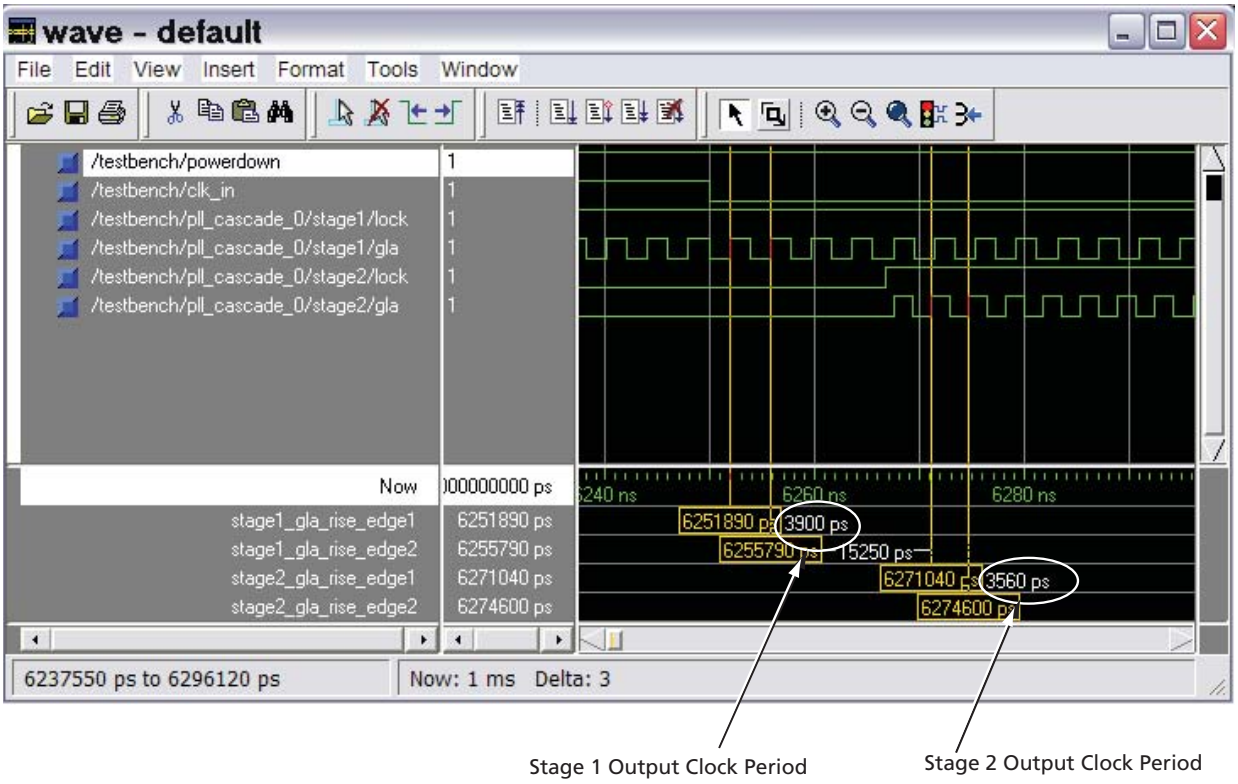


Figure 3-36 • **Model*Sim* Simulation Results**

Recommended Board-Level Considerations

The power to the PLL core is supplied by $V_{CCPLA/B/C/D/E/F}$ (V_{CCPLx}), and the associated ground connections are supplied by $V_{COMPLA/B/C/D/E/F}$ (V_{COMPLx}). When the PLLs are not used, the Actel Designer place-and-route tool automatically disables the unused PLLs to lower power consumption. The user should tie unused V_{CCPLx} and V_{COMPLx} pins to ground. Optionally, the PLL can be turned on/off during normal device operation via the POWERDOWN port (see Table 3-3 on page 3-8).

PLL Power Supply Decoupling Scheme

The PLL core is designed to tolerate noise levels on the PLL power supply as specified in the datasheets. When operated within the noise limits, the PLL will meet the output peak-to-peak jitter specifications specified in the datasheets. User applications should always ensure the PLL power supply is powered from a noise-free or low-noise power source.

However, in situations where the PLL power supply noise level is higher than the tolerable limits, various decoupling schemes can be designed to suppress noise to the PLL power supply. An example is provided in Figure 3-37. The V_{CCPLx} and V_{COMPLx} pins correspond to the PLL analog power supply and ground.

Actel strongly recommends that two ceramic capacitors (10 nF in parallel with 100 nF) be placed close to the power pins (less than 1 inch away). A third generic 10 µF electrolytic capacitor is recommended for low-frequency noise and should be placed farther away due to its large physical size. Actel recommends that a 6.8 µH inductor be placed between the supply source and the capacitors to filter out any low-/medium- and high-frequency noise. In addition, the PCB layers should be controlled so the V_{CCPLx} and V_{COMPLx} planes have the minimum separation possible, thus generating a good-quality RF capacitor.

For more recommendations, refer to the *Board-Level Considerations* application note.

Recommended 100 nF capacitor:

- Producer BC Components, type X7R, 100 nF, 16 V
- BC Components part number: 0603B104K160BT
- Digi-Key part number: BC1254CT-ND
- Digi-Key part number: BC1254TR-ND

Recommended 10 nF capacitor:

- Surface-mount ceramic capacitor
- Producer BC Components, type X7R, 10 nF, 50 V
- BC Components part number: 0603B103K500BT
- Digi-Key part number: BC1252CT-ND
- Digi-Key part number: BC1252TR-ND

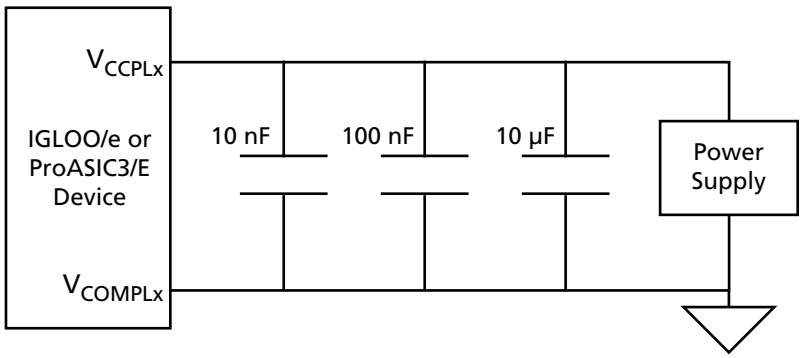


***Figure 3-37* • Decoupling Scheme for One PLL (should be replicated for each PLL used)**

Conclusion

The advanced CCCs of the IGLOO and ProASIC3 devices are ideal for applications requiring precise clock management. They integrate easily with the internal low-skew clock networks and provide flexible frequency synthesis, clock deskewing, and/or time-shifting operations.

Related Documents

Application Notes

Board-Level Considerations

http://www.actel.com/documents/BoardLevelCons_AN.pdf

Handbook Documents

UJTAG Applications in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_UJTAG_HBs.pdf

Global Resources in Actel Low-Power Flash Devices

http://www.actel.com/documents/LPD_Global_HBs.pdf

User I/O Naming Conventions in I/O Structures in IGLOO and ProASIC3 Devices

http://www.actel.com/documents/IGLOO_PA3_IO_HBs.pdf

User's Guides

IGLOO, Fusion, and ProASIC3 Macro Library Guide

http://www.actel.com/documents/pa3_libguide_ug.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-006-4

Revised December 2008

List of Changes

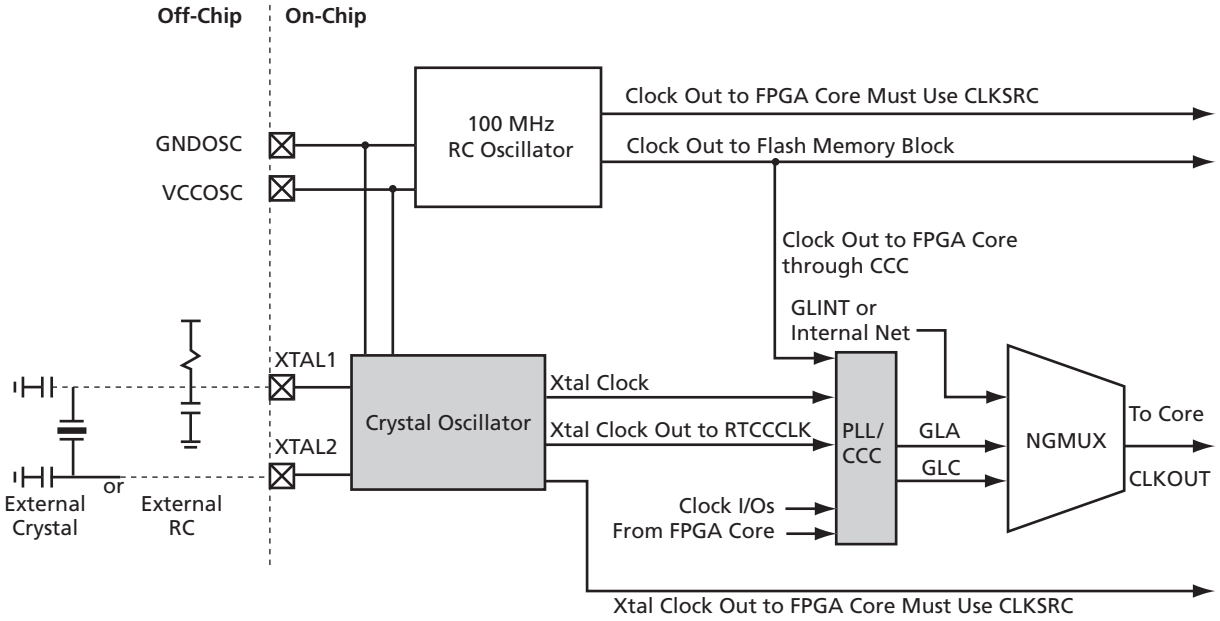
The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in the Current Version (v1.4)	Page
v1.3 (October 2008)	The "CCC Support in Actel's Flash Devices" section was updated to include IGLOO nano and ProASIC3 nano devices.	3-3
	Figure 3-2 · CCC Options: Global Buffers with No Programmable Delay was revised to add the CLKBIBUF macro.	3-4
	The description of the reference clock was revised in Table 3-2 · Input and Output Description of the CLKDLY Macro.	3-5
	Figure 3-6 · Clock Input Sources (30 k gates devices and below) is new. Figure 3-7 · Clock Input Sources Including CLKBUF, CLKBUF_LVDS/LVPECL, and CLKINT (60 k gates devices and above) applies to 60 k gate devices and above.	3-11
	The "IGLOO and ProASIC3" section was updated to include information for IGLOO nano devices.	3-12
	A note regarding Fusion CCCs was added to Figure 3-8 · Illustration of Hardwired I/O (global input pins) Usage for IGLOO and ProASIC3 devices 60 k Gates and Larger and the name of the figure was changed from Figure 4-8 · Illustration of Hardwired I/O (global input pins) Usage. Figure 3-9 · Illustration of Hardwired I/O (global input pins) Usage for IGLOO and ProASIC3 devices 30 k Gates and Smaller is new.	3-13
	Table 3-5 · Number of CCCs by Device Size and Package was updated to include IGLOO nano and ProASIC3 nano devices. Entries were added to note differences for the CS81, CS121, and CS201 packages.	3-17
	The "Clock Conditioning Circuits without Integrated PLLs" section was rewritten.	3-18
	The "IGLOO and ProASIC3 CCC Locations" section was updated for nano devices.	3-20
	Figure 4-13 · CCC Locations in the 15 k and 30 k Gate Devices was deleted.	4-20
v1.2 (June 2008)	This document was updated to include Fusion and RT ProASIC3 device information. Please review the document very carefully.	N/A
	The "CCC Support in Actel's Flash Devices" section was updated.	3-3
	In the "Global Buffer with Programmable Delay" section, the following sentence was changed from: "In this case, the I/O must be placed in one of the dedicated global I/O locations." To "In this case, the software will automatically place the dedicated global I/O in the appropriate locations."	3-4
	Figure 3-4 · CCC Options: Global Buffers with PLL was updated to include OADIVRST and OADIVHALF.	3-7
	In Figure 3-5 · CCC with PLL Block "fixed delay" was changed to "programmable delay".	3-7
	Table 3-3 · Input and Output Signals of the PLL Block was updated to include OADIVRST and OADIVHALF descriptions.	3-8
	Table 3-7 · Configuration Bit Descriptions for the CCC Blocks was updated to include configuration bits 88 to 81. Note 2 is new. In addition, the description for bit <76:74> was updated.	3-28
	Table 3-15 · Fusion Dynamic CCC Clock Source Selection and Table 3-16 · Fusion Dynamic CCC NGMUX Configuration are new.	3-32

Previous Version	Changes in the Current Version (v1.4)	Page
v1.2 (continued)	Table 3-17 · Fusion Dynamic CCC Division by Half Configuration and Table 3-18 · Configuration Bit <76:75> / VCOSEL<2:1> Selection for All Families are new.	3-33
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 3-1 · Overview of the CCCs Offered in Fusion, IGLOO, and ProASIC3: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	3-1
v1.0 (January 2008)	Table 3-1 · Flash-Based FPGAs and the associated text were updated to include the IGLOO PLUS family. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	3-3
	The "Global Input Selections" section was updated to include 15 k gate devices as supported I/O types for globals, for CCC only.	3-10
	Table 3-5 · Number of CCCs by Device Size and Package was revised to include ProASIC3L, IGLOO PLUS, A3P015, AGL015, AGLP030, AGLP060, and AGLP125.	3-17
	The "IGLOO and ProASIC3 CCC Locations" section was revised to include 15 k gate devices in the exception statements, as they do not contain PLLs.	3-20
51900133-0/5.06	Information about unlocking the PLL was removed from the "Dynamic PLL Configuration" section.	3-26
	In the "Dynamic PLL Configuration" section, information was added about running Layout and determining the exact setting of the ports.	3-38
	In Table 3-7 · Configuration Bit Descriptions for the CCC Blocks, the following bits were updated to delete "transport to the user" and reference the footnote at the bottom of the table: 79 to 71.	3-28

4 – Fusion Clock Resources

The Actel Fusion® mixed-signal FPGA family of devices has a wide variety of on-chip clocking peripherals, as shown in Figure 4-1. The on-chip resources enable the creation, manipulation, and distribution of clock signals both internally and externally. An integrated internal RC oscillator produces a 100 MHz clock without external components. For systems that require more precise clock signals, the Fusion mixed-signal FPGA family also supports an on-chip crystal oscillator circuit. In conjunction with the crystal oscillator circuit, the on-chip Real-Time Counter (RTC) provides timed wake-up or power-up sequences and thus the ability to supply time and date stamps.



Note: This is a simplified block diagram of the clocking resources within the Fusion device. For details regarding the global networks and clocking resources, refer to the Actel Fusion Mixed-Signal FPGAs datasheet. For details regarding the PLL/CCC, refer to the Clock Conditioning Circuits in Low-Power Flash Devices and Mixed-Signal FPGAs section of the handbook.

Figure 4-1 • **Fusion Mixed-Signal FPGA Clocking System**

Like the ProASIC®3/E family of flash-based FPGAs, the flash-based Fusion device integrates up to two phase-locked loop (PLL) cores in the embedded Clock Conditioning Circuits (CCCs). The PLL can be clocked from the internal RC oscillator, crystal oscillator, or any other internal signal. The integrated PLL provides the capability to alter the clock source by multiplying, dividing, synchronizing, advancing, or delaying the signal.

The Fusion mixed-signal FPGA also integrates one No-Glitch MUX (NGMUX) for each PLL. The NGMUX enables designers to switch between multiple clock sources without introducing glitches into the clock network.

This chapter includes the following sections:

- "Internal RC Oscillator" on page 4-2
- "Crystal Oscillator (XTLOSC)" on page 4-6
- "No-Glitch Multiplexer (NGMUX)" on page 4-15
- "Real-Time Counter (RTC)" on page 4-23

For information on using the CCC and PLL, refer to the *Clock Conditioning Circuits in Low-Power Flash Devices and Mixed-Signal FPGAs* section of the handbook.

Internal RC Oscillator

The internal RC oscillator is an on-chip free-running clock source capable of generating a 100 MHz source without external components. Using the internal RC oscillator in conjunction with the integrated PLL enables designers to generate clocks of varying frequency and phase, clocking both on- and off-chip resources.

The *Actel Fusion Mixed-Signal FPGAs* datasheet contains both timing and accuracy characteristics for the internal RC oscillator peripheral.

RC Oscillator Usage

The internal RC oscillator is capable of driving any of the clock macros (i.e., a static or dynamic PLL) directly after instantiation. To drive a macro in the FPGA core, the RC oscillator must first be routed through the CLKSRC macro. See the examples below on manually instantiating the RCOSC and CLKSRC macros. SmartGen can also be used to implement these macros. For more information on using SmartGen, refer to the *SmartGen, FlashROM, ASB, and Flash Memory System Builder User's Guide*.

Example: RC Oscillator Driving Clock Macros

The following example manually instantiates the internal RC oscillator using the RCOSC macro, and connects the 100 MHz clock output to the input pin of a SmartGen-generated PLL. Since the 100 MHz clock output does not connect to FPGA core logic, the CLKSRC macro is not needed.

Verilog

```
module myClocks (
  NSYSRESET,
  CLK25MHZ
);

  input NSYSRESET;
  output CLK25MHZ;

  wire CLK100;

  RCOSC uRCOSC (
    .CLKOUT (CLK100)
  );

  myPLL myPLL1 (
    .POWERDOWN (1'b1),
    .CLKA (CLK100),
    .LOCK (),
    .GLA  (CLK25MHZ),
    .OADIVRST (NSYSRESET)
  );

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity myClocks is
  port (
    NSYSRESET: in std_logic;
    CLK25MHZ: out std_logic
  );
end entity myClocks;

architecture myClocks is
  signal CLK100 : std_logic;

  component RCOSC
  port (
    CLKOUT: out std_logic
  );

  component myPLL
  port (
    POWERDOWN: in std_logic;
    CLKA: out std_logic;
    LOCK: out std_logic;
    GLA: out std_logic;
    OADIVRST: in std_logic
  );

begin
  uRCOSC : RCOSC
  port map (
    CLKOUT => CLK100
  );

  myPLL1 : myPLL
  port map (
    POWERDOWN => '1',
    CLKA => CLK100,
    LOCK => open,
    GLA => CLK25MHZ,
    OADIVRST => NSYSRESET
  );

end architecture myClocks;
```

Example: RC Oscillator Driving FPGA Core Logic

The following example manually instantiates the internal RC oscillator and connects the 100 MHz clock output to the FPGA core logic, which in turn generates a 25 MHz clock. Both the RCOSC and CLKSRC macros are used, RCOSC to instantiate the internal RC oscillator and CLKSRC to connect the RCOSC output to FPGA core logic.

Verilog

```
module myClock (
  NSYSRESET,
  CLK25MHZ
);

  input NSYSRESET;
  output CLK25MHZ;

  wire CLK100, SYSCLK;

  reg [1:0] iCOUNT;

  RCOSC uRCOSC (
    .CLKOUT (CLK100)
  );

  CLKSRC uCLKSRC (
    .A  (CLK100),
    .Y  (SYSCLK)
  );

  always @ (negedge NSYSRESET or posedge SYSCLK)
  begin
    if (NSYSRESET == 1'b0)
      iCOUNT = 2'b0;
    else iCOUNT = iCOUNT + 1'b1;
  end

  assign CLK25MHZ = iCOUNT[1];

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity myClock is
  port (
    NSYSRESET: in std_logic;
    CLK25MHZ: out std_logic
  );
end entity myClock;

architecture myClock is
  signal CLK100 : std_logic;
  signal SYSCLK : std_logic;

  component RCOSC
  port (
    CLKOUT: out std_logic
  );

  component CLKSRC
  port (
    A: in std_logic;
    Y: out std_logic
  );

begin

  uRCOSC : RCOSC
  port map (
    CLKOUT => CLK100
  );

  uCLKSRC : CLKSRC
  port map (
    A => CLK100,
    Y => SYSCLK
  );

  process (NSYSRESET, SYSCLK)
  variable iCOUNT: std_logic_vector(1 downto 0);
  begin
  if (NSYSRESET = '0')
    iCOUNT := (others => '0');
  elsif (SYSCLK'event and SYSCLK = '1')
    iCOUNT := iCOUNT + '1';
  end

  CLK25MHZ <= iCOUNT(1);

end architecture myClock;
```

RC Oscillator Tips and Package Connections

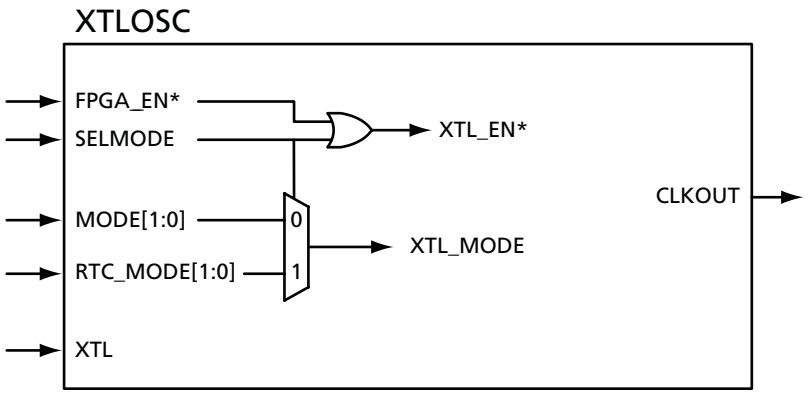
Although the internal RC oscillator is only a one-port macro, the GNDOSC and VCCOSC package pins must be connected externally to provide a power and ground source for the resource and the crystal oscillator. If neither the RC nor the crystal oscillator is used, refer to the *Actel Fusion Mixed-Signal FPGAs* datasheet for accurate termination guidelines.

Crystal Oscillator (XTLOSC)

The crystal oscillator (XTLOSC) generates the clock from an external crystal. The output of the XTLOSC CLKOUT signal can be selected as an input to the PLL. Refer to the *Clock Conditioning Circuits in Low-Power Flash Devices and Mixed-Signal FPGAs* section of the handbook for more details. The XTLOSC can operate in normal operation and standby mode (RTC is running and 1.5 V is not present).

In normal operation, the internal FPGA_EN signal is 1 as long as 1.5 V is present for V_{CC}. The internal enable signal for the crystal oscillator, XTL_EN, is enabled since FPGA_EN is asserted. The XTL_MODE signal can use MODE or RTC_MODE, depending on SELMODE.

During standby, 1.5 V is not available. FPGA_EN is 0 and SELMODE must be asserted in order for XTL_EN to be enabled. Hence XTL_MODE relies on RTC_MODE. SELMODE and RTC_MODE must be connected to RTCXTLSEL and RTCXTLMODE from the AB, respectively, for correct operation during standby. Refer to the "Real-Time Counter (RTC)" section on page 4-23 for a detailed description.



*Note: *Internal signal; does not exist in macro.*

Figure 4-2 • **XTLOSC Macro**

Table 4-1 • **XTLOSC Signals Description**

Signal Name	Width	Direction	Function
XTL_EN*	1		Enables the crystal. Active high.
XTL_MODE*	2		Settings for the crystal clock for different frequencies: **Value** **Modes** **Frequency Range** b'00 RC network 32 kHz to 4 MHz b'01 Low gain 32 to 200 kHz b'10 Medium gain 0.20 to 2.0 MHz b'11 High gain 2.0 to 20.0 MHz
SELMODE	1	IN	Selects the source of XTL_MODE and also enables the XTL_EN. Connect from RTCXTLSEL from AB. 0: For normal operation or sleep mode of operation XTL_EN depends on FPGA_EN, XTL_MODE depends on MODE 1: For Standby mode of operation XTL_EN is enabled, XTL_MODE depends on RTC_MODE
RTC_MODE[1:0]	2	IN	Settings for the crystal clock for different frequency ranges. XTL_MODE uses RTC_MODE when SELMODE is 1.
MODE[1:0]	2	IN	Settings for the crystal clock for different frequency ranges. XTL_MODE uses MODE when SELMODE is 0. In standby, MODE inputs will be 0s.
FPGA_EN*	1	IN	0 when 1.5 V is not present for V_{CC} 1 when 1.5 V is present for V_{CC}
XTL	1	IN	Crystal clock source
CLKOUT	1	OUT	Crystal clock output

*Note: *Internal signal and does not exist in macro*

The crystal oscillator can be configured in one of the four modes:

- RC network, 32 kHz to 4 MHz
- Low gain, 32 to 200 kHz
- Medium gain, 0.20 to 2.0 MHz
- High gain, 2.0 to 20.0 MHz

In RC network mode, the XTAL1 pin is connected to an RC circuit, as shown in Figure 4-1 on page 4-1. The XTAL2 pin should be left floating. The RC value can be chosen based on Figure 4-3 for any desired frequency between 32 kHz and 4 MHz. The RC network mode can also accommodate an external clock source on XTAL1 instead of an RC circuit.

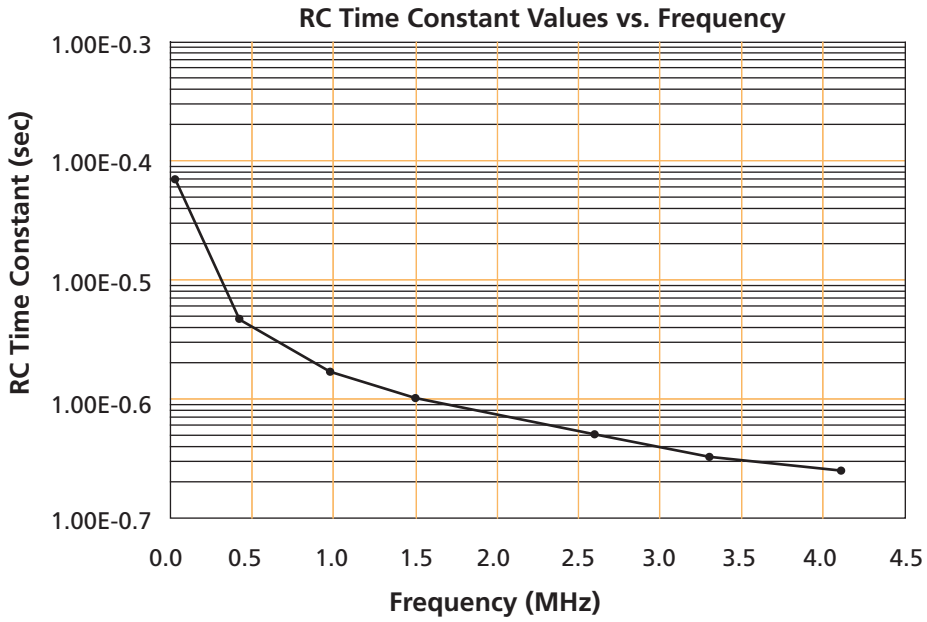


Figure 4-3 • **Crystal Oscillator: RC Time Constant Values vs. Frequency (typical)**

In low gain, medium gain, and high gain, an external crystal component or ceramic resonator can be added onto XTAL1 and XTAL2, as shown in Figure 4-1 on page 4-1.

Example: Crystal Oscillator Driving the Real-Time Counter

The following example manually instantiates the crystal oscillator using the XTLOSC macro and connects the external 32.768 kHz crystal output to the RTC in the Analog Block (AB). Since the 32.768 kHz clock output does not connect to FPGA core logic, the CLKSRC macro is not needed. The examples below assumes that the Analog Configuration MUX (ACM) has been previously configured and is controlling the functionality of the RTC. For more information on the ACM, refer to *Designing the Fusion Analog System*.

Verilog

```
module myRTC (
  CLK10MHZ,
  CLK32kHz
);

  input CLK10MHZ;
  input CLK32kHz;

  wire iRTCCLK, iRTCSELMODE;
  wire [1:0] iRTCMODE;

  wire iACMCLK, iACMWEN, iACMRESET;
  wire [7:0] iACMADDR, iACMRDATA, iACMWDATA;
```

```
  XTLOSC uXTLOSC (
    .XTL  (CLK32kHz),
    .CLKOUT (iRTCCLK),
    .SELMODE (iRTCSELMODE),
    .MODE (2'b0),
    .RTCMODE (iRTCMODE)
  );

  AB uAB (
    // Note: Several of the Analog Block signals
    //  have been omitted from this
    //  example, only the critical signals
    //  are present.

    .SYSCLK (CLK10MHZ),

    .ACMCLK (iACMCLK),
    .ACMWEN (iACMWEN),
    .ACMRESET (iACMRESET),
    .ACMWDATA (iACMWDATA),
    .ACMADDR (iACMADDR),
    .ACMRDATA (iACMRDATA),

    .RTCCLK (iRTCCLK),
    .RTCXTLSEL (iRTCSELMODE),
    .RTCXTLMODE (iRTCMODE),
    .RTCMATCH (),
    .RTCPSMMATCH ()
  );

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity myRTC is
  port (
    CLK10MHZ: in std_logic;
    CLK32kHz: in std_logic
  );
end entity myRTC;

architecture myRTC is
  signal iRTCCLK : std_logic;
  signal iRTCSELMODE : std_logic;
  signal iRTCMODE : std_logic_vector(1 downto 0);

  signal iACMCLK : std_logic;
  signal iACMRESET : std_logic;
  signal iACMWEN : std_logic;
  signal iACMADDR : std_logic_vector(7 downto 0;
  signal iACMRDATA : std_logic_vector(7 downto 0;
  signal iACMWDATA : std_logic_vector(7 downto 0;

  component XTLOSC
  port (
    XTL: in std_logic;
    CLKOUT: out std_logic;
    SELMODE: in std_logic_vector(1 downto 0);
    RTCMODE: in std_logic_vector(1 downto 0);
    MODE: in std_logic_vector(1 downto 0)
  );
```

```
AB
  -- Note: Several of the Analog Block signals
  --  have been omitted from this
  --  example, only the critical signals
  --  are present.
  port (
    SYSCLK: in std_logic;

    ACMCLK: in std_logic;
    ACMWEN: in std_logic;
    ACMRESET: in std_logic;
    ACMADDR: in std_logic_vector(7 downto 0);
    ACMWDATA: in std_logic_vector(7 downto 0);
    ACMRDATA: out std_logic_vector(7 downto 0);

    RTCCLK: in std_logic;
    RTCXTLSEL: out std_logic;
    RTCXTLMODE: out std_logic_vector(1 downto 0);
    RTCMATCH: out std_logic;
    RTCPSMATCH: out std_logic
  );

begin

  uXTLOSC : XTLOSC
  port map (
    XTL => CLK32kHz,
    CLKOUT => iRTCCLK,
    SELMODE => iRTCSELMODE,
    MODE => “00”,
    RTCMODE => iRTCMODE
  );

  uAB : AB
  -- Note: Several of the Analog Block signals
  --  have been omitted from this
  --  example, only the critical signals
  --  are present.
  port map (
    SYSCLK => CLK10MHZ,

    ACMCLK => iACMCLK,
    ACMWEN => iACMWEN,
    ACMRESET => iACMRESET,
    ACMWDATA => iACMWDATA,
    ACMADDR => iACMADDR,
    ACMRDATA => iACMRDATA,

    RTCCLK => iRTCCLK,
    RTCXTLSEL => iRTCSELMODE,
    RTCXTLMODE => iRTCMODE,
    RTCMATCH => open,
    RTCPSMMATCH => open
  );

end architecture myRTC;
```

Example: Crystal Oscillator Driving Clock Macros

The following example manually instantiates the crystal oscillator using the XTLOSC macro and connects the external crystal to the input pin of a SmartGen-generated PLL. Since the external crystal does not connect to FPGA core logic, the CLKSRC macro is not needed.

Verilog

```
module myXTAL (
  CLK50MHZ,
  NSYSRESET,
  XTAL10MHZ
);

  input XTAL10MHZ;
  input NSYSRESET;
  output CLK50MHZ;

  wire iXTLCLK;

  XTLOSC uXTLOSC (
    .XTL  (XTAL10MHZ),
    .CLKOUT (iXTLCLK),
    .SELMODE (1'b0),
    .MODE (2'b11),
    .RTCMODE (2'b0)
  );

  myPLL myPLL1 (
    .POWERDOWN (1'b1),
    .CLKA (iXTLCLK),
    .LOCK (),
    .GLA  (CLK50MHZ),
    .OADIVRST (NSYSRESET)
  );

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity myXTAL is
  port (
    CLK50MHZ: out std_logic;
    NSYSRESET: in std_logic;
    XTAL10MHZ: in std_logic
  );
end entity myXTAL;

architecture myXTAL is
  signal iXTLCLK : std_logic;

  component XTLOSC
  port (
    XTL: in std_logic;
    CLKOUT: out std_logic;
    SELMODE: in std_logic_vector(1 downto 0);
    RTCMODE: in std_logic_vector(1 downto 0);
    MODE: in std_logic_vector(1 downto 0)
  );

  component myPLL
  port (
    POWERDOWN: in std_logic;
    CLKA: out std_logic;
    LOCK: out std_logic;
    GLA: out std_logic;
    OADIVRST: in std_logic
  );

begin
  uXTLOSC : XTLOSC
  port map (
    XTL => XTAL10MHZ,
    CLKOUT => iXTLCLK,
    SELMODE => '0',
    MODE => "11",
    RTCMODE => "00"
  );

  myPLL1 : myPLL
  port map (
    POWERDOWN => '1',
    CLKA => iXTLCLK,
    LOCK => open,
    GLA => CLK50MHZ,
    OADIVRST => NSYSRESET
  );

end architecture myXTAL;
```

Example: Crystal Oscillator Driving FPGA Core Logic

The following example manually instantiates the crystal oscillator and connects the external crystal output to the FPGA core logic, which in turn generates a clock divided by four. Both the XTLOSC and CLKSRC macros are used, XTLOSC to instantiate the crystal oscillator and CLKSRC to connect the XTLOSC output to FPGA core logic.

Verilog

```
module myCLKDIV (
  CLKDIV4,
  NSYSRESET,
  XTAL10MHZ
);

  input XTAL10MHZ;
  input NSYSRESET;
  output CLKDIV4;

  wire iXTLCLK;
  wire SYSCLK;

  reg [1:0] iCOUNT;

  XTLOSC uXTLOSC (
    .XTL  (XTAL10MHZ),
    .CLKOUT (iXTLCLK),
    .SELMODE (1'b0),
    .MODE (2'b11),
    .RTCMODE (2'b0)
  );

  CLKSRC uCLKSRC (
    .A  (iXTLCLK),
    .Y  (SYSCLK)
  );

  always @ (negedge NSYSRESET or posedge SYSCLK)
  begin
    if (NSYSRESET == 1'b0)
      iCOUNT = 2'b0;
    else iCOUNT = iCOUNT + 1'b1;
  end

  assign CLKDIV4 = iCOUNT[1];

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity myCLKDIV is
  port (
    CLKDIV4: out std_logic;
    NSYSRESET: in std_logic;
    XTAL10MHZ: in std_logic
  );
end entity myCLKDIV;

architecture myCLKDIV is
  signal iXTLCLK : std_logic;
  signal SYSCLK : std_logic;

  component XTLOSC
  port (
    XTL: in std_logic;
    CLKOUT: out std_logic;
    SELMODE: in std_logic_vector(1 downto 0);
    RTCMODE: in std_logic_vector(1 downto 0);
    MODE: in std_logic_vector(1 downto 0)
  );

  component CLKSRC
  port (
    A: in std_logic;
    Y: out std_logic
  );

begin

  uXTLOSC : XTLOSC
  port map (
    XTL => XTAL10MHZ,
    CLKOUT => iXTLCLK,
    SELMODE => '0',
    MODE => "11",
    RTCMODE => "00"
  );

  uCLKSRC : CLKSRC
  port map (
    A => iXTLCLK,
    Y => SYSCLK
  );

  process (NSYSRESET, SYSCLK)
  variable iCOUNT: std_logic_vector(1 downto 0);
  begin
    if (NSYSRESET = '0')
      iCOUNT := (others => '0');
    elsif (SYSCLK'event and SYSCLK = '1')
      iCOUNT := iCOUNT + '1';
  end

  CLKDIV4 <= iCOUNT(1);

end architecture myCLKDIV;
```

Crystal Oscillator Tips and Package Connections

When using the crystal oscillator, the GNDOSC and VCCOSC external package pins must be connected to provide power and ground sources for this resource and the internal RC oscillator (Table 4-2). If neither the RC nor the crystal oscillator is used, refer to the *Actel Fusion Mixed-Signal FPGAs* datasheet for accurate termination guidelines.

In addition, the XTL pin of the XTLOSC macro is connected to the XTAL1 package pin, the crystal oscillator circuit input. When using an external RC network the XTAL2 package pin must be left unconnected.

Table 4-2 • **Crystal Oscillator Tips and Package Connections**

Signal Name	Direction	Description
XTAL1	Input	Input to crystal oscillator circuit. This pin is used to connect the external crystal, ceramic resonator, RC network, or external clock input. When using an external crystal or ceramic oscillator, Actel recommends using external capacitors (refer to the *Actel Fusion Mixed-Signal FPGAs* datasheet for the recommended capacitor values). If using an external RC network or clock input, use XTAL1 and leave XTAL2 unconnected.
XTAL2	Input	Input to crystal oscillator circuit. This pin is used to connect the external crystal, ceramic resonator, RC network, or external clock input. When using an external crystal or ceramic oscillator, Actel recommends using external capacitors (refer to the *Actel Fusion Mixed-Signal FPGAs* datasheet for the recommended capacitor values). If using an external RC network or clock input, use XTAL1 and leave XTAL2 unconnected.
VCCOSC	Input	External power supply (3.3 V) for both the integrated RC and crystal oscillator circuits
GNDOSC	Input	External ground supply for both the integrated RC and crystal oscillator circuits

No-Glitch Multiplexer (NGMUX)

Up to two No-Glitch Multiplexers, positioned downstream from the PLL/CCC blocks, are integrated into the Fusion device, as shown in Figure 4-4. The NGMUX provides a special switching sequence between two asynchronous clock domains, which avoids generating any unwanted narrow glitch pulses. It switches between two different clock sources and the output goes to the global network, as shown in Figure 4-5 on page 4-16.

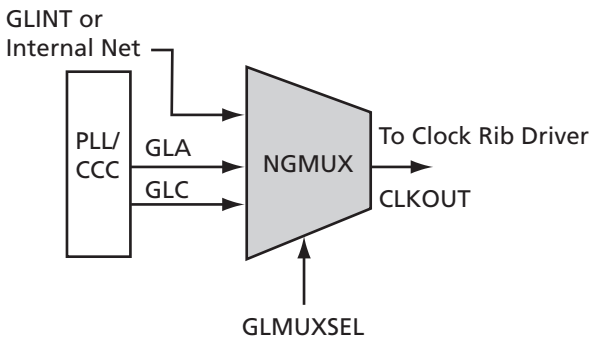


Figure 4-4 • **No-Glitch MUX**

Figure 4-5 • **No-Glitch MUX Macro**

The NGMUX allows the following inputs: PLL outputs GLA and GLC, and GLINT or any internal net. However, there are some restrictions on these signals, which are described in "NGMUX Usage" on page 4-18.

NGMUX Modes of Operation

The signals driving NGMUX have the same specifications as the output clock of the PLL/CCC. The following examples show various scenarios for the switching sequence between two asynchronous clock domains CLK0 and CLK1.

Case 1: Both CLK0 and CLK1 Active

When both the CLK0 and CLK1 inputs to the NGMUX are active, the switching sequence between the two clock sources (from CLK0 to CLK1) is as below. An example is shown in Figure 4-6.

1. A transition on S initiates the clock source switch.
2. GL drives one last complete CLK0 positive pulse (i.e., one rising edge followed by one falling edge).
3. GL stays LOW until the second rising edge of CLK1 occurs.
4. At the second CLK1 rising edge, GL continuously delivers CLK1.

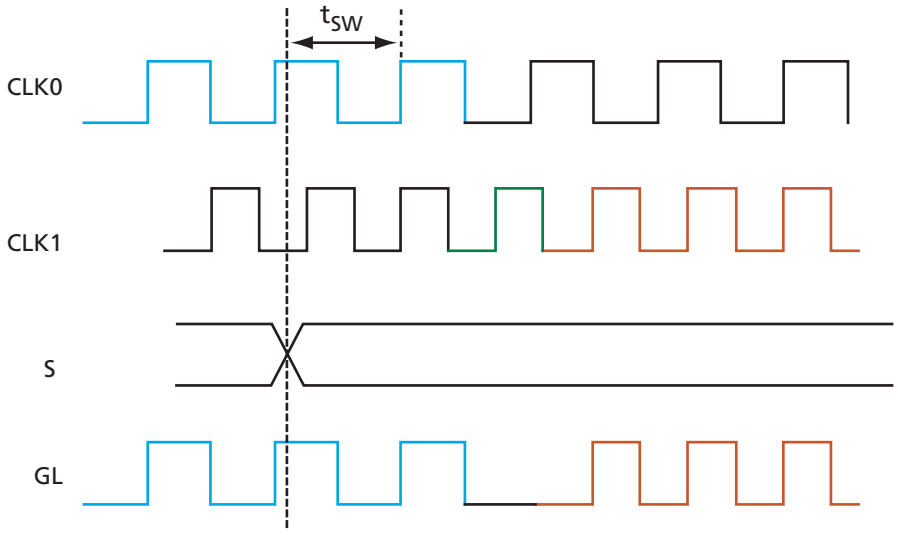


Figure 4-6 • **Active CLK0/CLK1 Inputs to NGMUX**

Note: Min. t_{sw} = 0.05 ns at 35°C (typical conditions)

Case 2: CLK0 Stopped or at Very Low Frequency

If CLK0 stops or runs at a very low frequency after S transition, the timeout circuitry inside the FPGA is used. The sequence of switching between the two clock sources (from CLK0 to CLK1) is described and illustrated below.

Case 2A: No Rising CLK0 Edge

If CLK0 does not have a rising edge before the seventh CLK1 rising edge, the switching sequence between the two clock sources (from CLK0 to CLK1) is as shown in Figure 4-7.

1. At the seventh CLK1 rising edge, GL will go LOW until the ninth CLK1 rising edge.
2. At the ninth CLK1 rising edge, GL will continuously deliver the CLK1 signal.

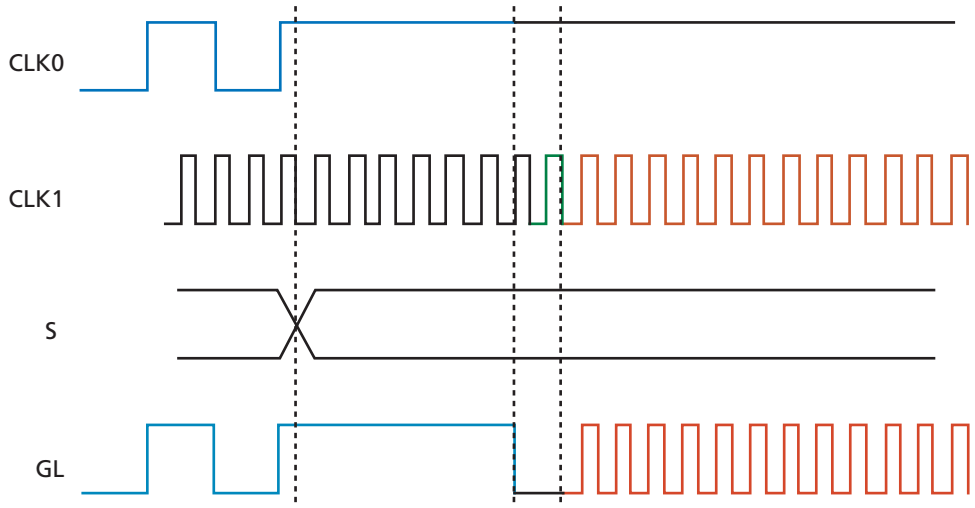


Figure 4-7 • **Low-Frequency CLK0 after S Transition, No Rising CLK0 Edge before Seventh Rising CLK1 Edge**

Case 2B: No Falling CLK0 Edge

If a CLK0 rising edge occurs before the seventh CLK1 rising edge but a CLK0 falling edge does not occur before the fifteenth CLK1 rising edge, the sequence of switching between the two clock sources (from CLK0 to CLK1) is as shown in Figure 4-8.

1. At the fifteenth CLK1 rising edge, GL will go LOW until the seventeenth CLK1 rising edge.
2. At the seventeenth CLK1 rising edge, GL will continuously deliver the CLK1 signal.

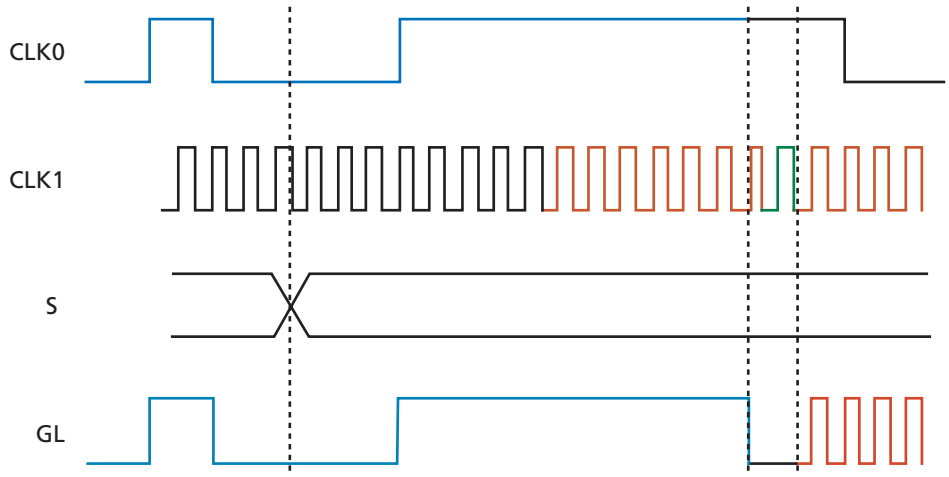


Figure 4-8 • **Low-Frequency CLK0 after S Transition, Rising CLK0 Edge before Seventh Rising CLK1 Edge**

NGMUX Usage

The software implementation of the NGMUX has been simplified to a 2:1 multiplexer, as shown in Figure 4-5 on page 4-16. The two clock input ports are CLK0 and CLK1, and the output clock port is GL. The allowable inputs to the NGMUX are as follows:

- The GLA and GLC outputs of a PLL
- The GLA output of a PLL and GLINT (the fanout of GLINT must be 1)
- The GLA output of a PLL and an internal net

SmartGen can also be used to implement these macros. For more information on using SmartGen, refer to the *SmartGen, FlashROM, Analog System Builder, and Flash Memory System Builder User's Guide*.

The following example manually instantiates the No-Glitch Multiplexer using the NGMUX macro and connects the CLK0 and CLK1 ports to the output ports of a SmartGen-generated PLL.

Verilog

```
module myCLKMUX (
  SYSCLK,
  CLK75MHZ,
  CLKSEL
);

  input CLKSEL;
  input CLK75MHZ;
  output SYSCLK;

  wire iGLA, iGLC;

  NGMUX uNGMUX (
    .CLK0 (iGLA),
    .CLK1 (iGLC),
    .S  (CLKSEL),
    .GL  (SYSCLK)
  );

  myPLL myPLL1 (
    .POWERDOWN (1'b1),
    .CLKA ( CLK75MHZ),
    .LOCK (),
    .GLA  ( iGLA),
    .GLC  (iGLC),
    .OADIVRST (NSYSRESET)
  );

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity my CLKMUX is
  port (
    SYSCLK: out std_logic;
    CLK75MHZ: in std_logic;
    CLKSEL: in std_logic
  );
end entity my CLKMUX;

architecture my CLKMUX is
  signal iGLA : std_logic;
  signal iGLC : std_logic;

  component NGMUX
  port (
    CLK0: in std_logic;
    CLK1: in std_logic;
    S: in std_logic;
    GL : out std_logic
  );

  component myPLL
  port (
    POWERDOWN: in std_logic;
    CLKA: out std_logic;
    LOCK: out std_logic;
    GLA: out std_logic;
    GL C: out std_logic;
    OADIVRST: in std_logic
  );

begin
  uNGMUX : NGMUX
  port map (
    CLK0 => iGLA,
    CLK1 => iGLC,
    S => CLKSEL,
    GL => SYSCLK
  );

  myPLL1 : myPLL
  port map (
    POWERDOWN => '1',
    CLKA => CLK75MHZ,
    LOCK => open,
    GLA => iGLA,
    GL C => iGLC,
    OADIVRST => NSYSRESET
  );

end architecture my CLKMUX;
```

NGMUX Tips

The following design considerations are recommended when using the NGMUX:

- The NGMUX has a fixed design location and is intended to be placed downstream from the PLL.
- Hardwire the NGMUX CLK0 input to PLL output GLA, as GLA must drive the NGMUX CLK0 input. GLA can only have a fanout of one, as the GLA global driver is used for the NGMUX output.
- If the two inputs to the NGMUX are PLL outputs GLA and GLC, you may lose the global network driver for GLC because it is consumed by the PLL output. Since the global network in Fusion is segmented, local clock networks can be used even though the whole global network is not available.
- Since the NGMUX macro has a fixed location (downstream from the PLL), routing delay can occur when the input to NGMUX comes from a regular net.
- The NGMUX GL output uses the GLA global network.

NGMUX Timing Analysis

Timing analysis verifies the functionality of the design with timing information. To check the design functionality for NGMUX, designers should check both the static and dynamic timing analyses as follows.

NGMUX Static Timing Analysis

Static timing analysis on both clock inputs is performed separately using the SmartTime tool in Designer (Figure 4-9). Run setup and hold checks on the source pins of CLK0 and CLK1.

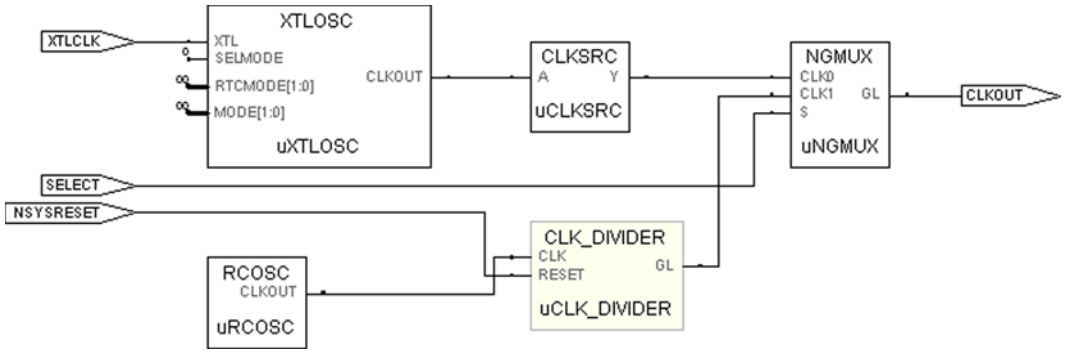


Figure 4-9 • **Static Timing Analysis Example**

Figure 4-9 on page 4-20 shows how the two inputs of the NGMUX are connected from the uCLK_DIVIDER and uCLKSRC components. In this instance, setup and hold time checks in SmartTime are done for both clocks, uCLK_DIVIDER/Inst1:GL and uCLKSRC:Y, as shown in Figure 4-10.

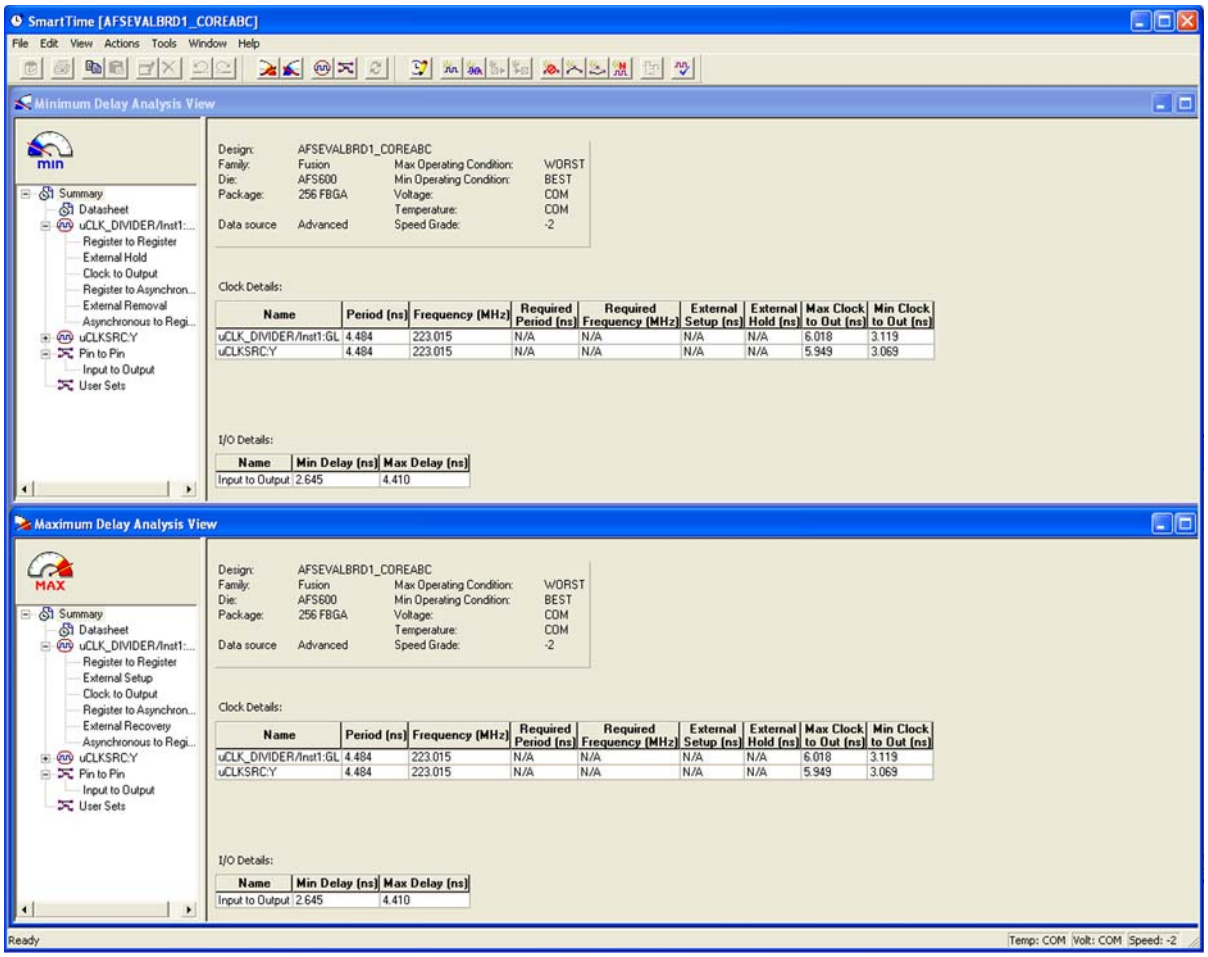


Figure 4-10 • **Checking Setup and Hold Times for Clocks**

NGMUX Dynamic Timing Analysis

For dynamic timing analysis, run a back-annotated timing simulation using the ModelSim® tool, and check the NGMUX signals, as shown in Figure 4-11.

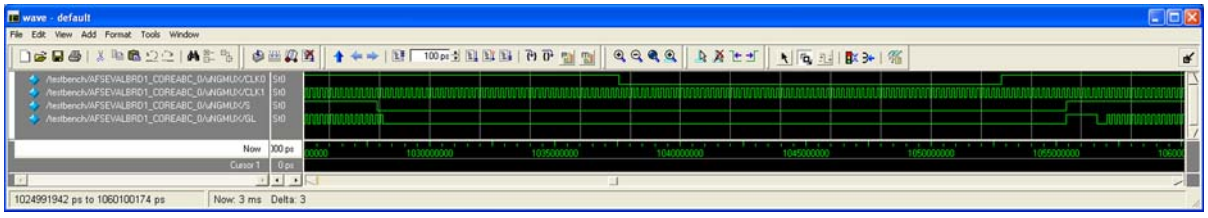


Figure 4-11 • **Dynamic Timing Analysis for NGMUX Signals**

A transition of S from High to Low initiates a switch to CLK0, and from Low to High initiates a switch to CLK1. The output of the NGMUX is undefined if S switches again before the previous switch operation has completed.

NGMUX Connections

The NGMUX has two input ports that are intended to be connected to clock signals. Connect the CLK0 input to a net driven by a clock signal. The fanout of the net connecting CCC:GLA to NGMUX:CLK0 should always be one. The driver of this clock signal will automatically be placed in the GLA tile for the CCC location in which the NGMUX is placed.

There are two possible connections for the NGMUX:CLK1 input:

1. The CLK1 port can be driven by a clock signal. The fanout of the net connecting CCC:GLC to NGMUX:CLK1 should always be one. The driver will automatically be placed in the CCC:GLC tile for the location in which the NGMUX is placed.
2. The CLK1 port can also be driven by a routed net; in this case, there is no restriction on the placement or fanout of the logic/net driving NGMUX:CLK1.

The integrated Fusion oscillators cannot drive the NGMUX directly, as they do not produce the clock signals. You must connect them to the NGMUX inputs through a valid clock macro (i.e., CLKSRC), refer to "Internal RC Oscillator" on page 4-2 and "Crystal Oscillator (XTLOSC)" on page 4-6 for more information.

NGMUX Placement

The NGMUX macros are placed automatically during layout. The NGMUX macros can be manually placed by doing the following:

1. **Placing NGMUX.** One of the two available locations for NGMUX must be chosen: TILE5 of the central CCC locations (shown in yellow in Figure 4-12).
2. **Placing the driver for CLK0.** The driver for CLK0 has to be a CCC macro that can be placed in the CCC:GLA tile (shown in green in Figure 4-12). The CCC macros that can be placed here are as follows:
   - CLKBUF (only non-V_{REF} versions)
   - CLKBIBUF
   - CLKSRC
   - CLKDLY
   - CLKDIVDLY
   - PLL
3. **Placing the driver for CLK1.** If the driver for CLK1 is a CCC macro, it infers a hardwired connection. This macro must be placed in the CCC:GLC tile (shown in pink in Figure 4-12) and has the same CCC macro restrictions as CLK0.

   When the CLK1 port drives a PLL:GLC instance, the PLL:GLA instance of the same PLL must become the driver for CLK0.

   When the CCC macro is driven from the hardwired I/O, placing the I/O controls the placement of the CCC macro.

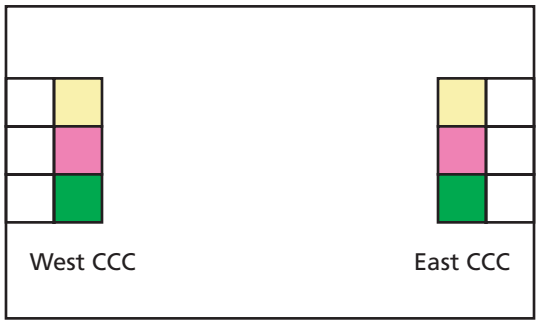


Figure 4-12 • **NGMUX Interconnects**

Real-Time Counter (RTC)

The addition of the real-time counter enables the Fusion mixed-signal FPGA to support both standby and sleep modes of operation, greatly reducing power consumption in many applications. The RTC also provides implementation of a time and date calendar, enabling embedded systems to log data with time and date stamps.

RTC Usage

The RTC resides within the Fusion Analog Block and has the following features and requirements:

- The RTC must be driven by the crystal oscillator circuit, and the crystal oscillator must be configured to operate in RTC mode.
- The MATCH signal on the output of the RTC system asserts when the value in the counter matches the value specified in the match register.
- There is an optional output RTCPSMMATCH that is triggered on a match. The RTCPSMMATCH signal can be used to signal the internal voltage regulator to power up/down and must be connected to the RTCPSM macro so the voltage regulator actives when the MATCH signal is asserted.

The MATCH signal asserts when the counter is equal to the value contained in MATCHREG. MATCHREG is a 40-bit register located in the ACM.

The RTC count register (COUNTER) can be preloaded with a zero or non-zero start value. The default value is zero. This is also a 40-bit register located in the ACM.

The control/status register (CTRL_STAT) is an 8-bit register located within the ACM that defines the operation of the RTC. The control register can reset the RTC, enabling operation to begin with all zeroes in the counter. The RTC can be configured to clear upon a match with the match register, or it can continue to count while the match signal is still asserted. Designers can also enable the Fusion device to power on at a specific time or at periodic intervals. For more information on the CTRL_STAT register, refer to the *Actel Fusion Mixed-Signal FPGAs* datasheet.

SmartGen can also be used to implement these macros. For more information on using SmartGen, refer to the *SmartGen, FlashROM, Analog System Builder, and Flash Memory System Builder User's Guide*.

The following example manually instantiates the crystal oscillator using the XTLOSC macro and connects the external 32.768 KHz crystal output to the RTC in the AB. Since the 32.768 KHz clock output is not connected to FPGA core logic, the CLKSRC macro is not needed. The example below assumes that the ACM has been previously configured and is controlling the functionality of the RTC. For more information on the ACM refer to the *Designing the Fusion Analog System*.

Verilog

```
module myRTC (
  CLK10MHZ,
  CLK32kHz
);

  input CLK10MHZ;
  input CLK32kHz;

  wire iRTCCLK, iRTCSELMODE;
  wire [1:0] iRTCMODE;

  wire iACMCLK, iACMWEN, iACMRESET;
  wire [7:0] iACMADDR, iACMRDATA, iACMWDATA;

  XTLOSC uXTLOSC (
    .XTL  (CLK32kHz),
    .CLKOUT (iRTCCLK),
    .SELMODE (iRTCSELMODE),
    .MODE (2'b0),
    .RTCMODE (iRTCMODE)
  );

  AB uAB (
    // Note: Several of the Analog Block signals
    //  have been omitted from this
    //  example, only the critical signals
    //  are present.

    .SYSCLK (CLK10MHZ),

    .ACMCLK (iACMCLK),
    .ACMWEN (iACMWEN),
    .ACMRESET (iACMRESET),
    .ACMWDATA (iACMWDATA),
    .ACMADDR (iACMADDR),
    (iACMRDATA),

    .RTCCLK (iRTCCLK),
    .RTCXTLSEL (iRTCSELMODE),
    .RTCXTLMODE (iRTCMODE),
    .RTCMATCH (),
    .RTCPSMMATCH ()
  );

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;

entity myRTC is
  port (
    CLK10MHZ: in std_logic;
    CLK32kHz: in std_logic
  );
end entity my RTC;

architecture my RTC is
  signal iRTCCLK : std_logic;
  signal iRTCSELMODE : std_logic;
  signal iRTCMODE : std_logic_vector(1 downto 0);

  signal iACMCLK : std_logic;
  signal iACMRESET : std_logic;
  signal iACMWEN : std_logic;
  signal iACMADDR : std_logic_vector(7 downto 0;
  signal iACMRDATA : std_logic_vector(7 downto 0;
  signal iACMWDATA : std_logic_vector(7 downto 0;

  component XTLOSC
  port (
    XTL: in std_logic;
    CLKOUT: out std_logic;
    SELMODE: in std_logic_vector(1 downto 0);
    RTCMODE: in std_logic_vector(1 downto 0);
    MODE: in std_logic_vector(1 downto 0)
  );

  component AB
  -- Note: Several of the Analog Block signals
  --  have been omitted from this
  --  example, only the critical signals
  --  are present.
  port (
    SYSCLK: in std_logic;

    ACMCLK: in std_logic;
    ACMWEN: in std_logic;
    ACMRESET: in std_logic;
    ACMADDR: in std_logic_vector(7 downto 0);
    ACMWDATA: in std_logic_vector(7 downto 0);
    ACMRDATA: out std_logic_vector(7 downto 0);

    RTCCLK: in std_logic;
    RTCXTLSEL: out std_logic;
    RTCXTLMODE: out std_logic_vector(1 downto 0);
    RTCMATCH: out std_logic;
    RTCPSMATCH: out std_logic
  );

begin

  uXTLOSC : XTLOSC
  port map (
    XTL => CLK32kHz,
    CLKOUT => iRTCCLK,
    SELMODE => iRTCSELMODE,
    MODE => “00”,
    RTCMODE => iRTCMODE
  );
```

```
  uAB : AB
  -- Note: Several of the Analog Block signals
  --  have been omitted from this
  --  example, only the critical signals
  --  are present.
  port map (
    SYSCLK => CLK10MHZ,

    ACMCLK => iACMCLK,
    ACMWEN => iACMWEN,
    ACMRESET => iACMRESET,
    ACMWDATA => iACMWDATA,
    ACMADDR => iACMADDR,
    ACMRDATA => iACMRDATA,

    RTCCLK => iRTCCLK,
    RTCXTLSEL => iRTCSELMODE,
    RTCXTLMODE => iRTCMODE,
    RTCMATCH => open,
    RTCPSMMATCH => open
  );

end architecture myRTC;
```

RTC Tips

The following design considerations are advised when using the RTC:

- When the RTC is not configured to reset the counter when a match occurs, the time interval between active RTCMATCH occurrences is equal to the total cumulative time count of the 40-bit RTC. In other words, the counter must overflow and reach the MATCHREG value again to create an active RTCMATCH output. The time required for the counter to overflow would not be practical for most applications; Actel recommends that the counter be reset upon a match condition if the RTCMATCH signal is needed.
- Each bit of the 40-bit COUNTER is compared to each bit of the 40-bit MATCHREG via XNOR gates, and the result is stored in the MATCHBITS register, enabling the designer to check whether an individual bit match has occurred.
- The location of the RTC registers within the ACM is shown in Table 4-3.

Table 4-3 • **Location of the RTC within the Analog Configuration MUX**

ACM_ADDR[7:0]	Register Name	Description
0x40	COUNTER0	Counter Bits [7:0]
0x41	COUNTER1	Counter Bits [15:8]
0x42	COUNTER2	Counter Bits [23:16]
0x43	COUNTER3	Counter Bits [31:24]
0x44	COUNTER4	Counter Bits [39:32]
0x48	MATCHREG0	Match Register Bits [7:0]
0x49	MATCHREG1	Match Register Bits [15:8]
0x4A	MATCHREG2	Match Register Bits [23:16]
0x4B	MATCHREG3	Match Register Bits [31:24]
0x4C	MATCHREG4	Match Register Bits [39:32]
0x50	MATCHBITS0	Individual Match Bits [7:0]
0x51	MATCHBITS1	Individual Match Bits [15:8]
0x52	MATCHBITS2	Individual Match Bits [23:16]
0x53	MATCHBITS3	Individual Match Bits [31:24]
0x54	MATCHBITS4	Individual Match Bits [39:32]
0x58	CTRL_STAT	Control / Status Register Bits
0x59	TEST_REG	Test Register

RTC Interconnection

Figure 4-13 shows the interconnection between the RTC and the various components in the Fusion device. If any hardwired input is not used, connect it to GND, and leave unused outputs floating (see the "Verilog" section on page 4-24 and the "VHDL" section on page 4-25 for an example of unused outputs left floating). For all hardwired connections, the fanout of the net connecting the two hardwired pins must be one.

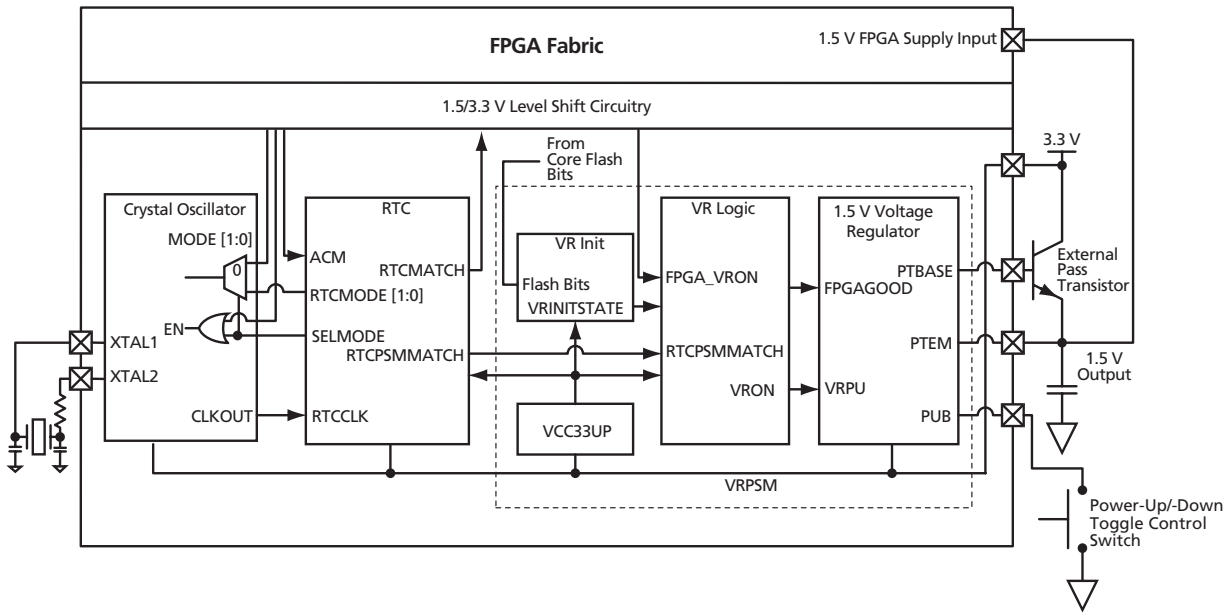


Figure 4-13 • **RTC Interconnection Diagram**

Related Documents

Handbook Documents

Actel Fusion Mixed-Signal FPGAs datasheet
http://www.actel.com/documents/Fusion_DS.pdf
Clock Conditioning Circuits in Low-Power Flash Devices and Mixed-Signal FPGAs
http://www.actel.com/documents/LPD_CCC_HBs.pdf
Designing the Fusion Analog System
http://www.actel.com/documents/Fusion_Analog_HBs.pdf

User's Guides

SmartGen, FlashROM, ASB, and Flash Memory System Builder User's Guide
http://www.actel.com/documents/genguide_ug.pdf

Part Number and Revision Date

Part Number 51700092-003-1
Revised August 2009

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.1)	Page
v1.0 (November 2007)	The "Crystal Oscillator" section was revised to remove Table 4-1 · RC Oscillator Tips and Package Connections and Table 4-2 · Crystal Oscillator Mode Settings. The text of the "Crystal Oscillator Usage" section was replaced with new text and Figure 4-2 · XTLOSC Macro was replaced.	4-6
	Figure 4-3 · Crystal Oscillator: RC Time Constant Values vs. Frequency (typical) is new.	4-8

Embedded Memories

Actel®
POWER MATTERS

5 – Fusion Embedded Flash Memory Blocks

Actel has not only utilized flash memory technology to configure the Actel Fusion® FPGA core tiles, but Fusion also offers up to four embedded flash Blocks (FBs), each 2 Mbits in density, for system initialization and general data storage. The embedded flash memory blocks are accessible by both on-chip and off-chip resources. Internally, the FPGA core fabric can directly access the FB's data bus. Externally, the embedded flash memory is accessible via the JTAG port or through interfaces implemented within the FPGA fabric: AMBA AHB (Advanced Microcontroller Bus Architecture Advanced High-Performance Bus), CoreCFI (Common Flash Interface), or a proprietary interface. It can be partitioned along page boundaries, giving designers better control over memory space usage, and the ability to individually protect each memory space against page loss, page overwrite, and external access. It also offers priority action control of the memory operations during simultaneous access requests.

The embedded flash memory can be configured using the Flash Memory System Builder in Actel Libero® Integrated Design Environment (IDE), using the CoreConsole IP Deployment Platform (IDP) for microprocessor usage, and manually through RTL. The following sections discuss design usage details for each method of configuration.

Using the Embedded Flash Memory for Initialization

The Flash Memory System Builder in Libero IDE enables designers to easily configure the embedded flash memory via a simple GUI interface. The GUI interface provides FPGA designers the ability to quickly partition and configure the embedded flash memory for initialization or general data storage purposes. This section discusses the uses of the initialization IP offered in Libero IDE as a block within the embedded Flash Memory System.

The Flash Memory System Builder offers the ability to support five types of clients. A client is a design block whose functionality is dependent on or configured by the data stored in the embedded flash memory. These clients also utilize the custom IP functionality added to the flash memory control logic, only available through the Flash Memory Block System Builder. Two of the client types are used to add general-purpose data storage capability to the Flash Memory System. These clients are described in the "Using the Embedded Flash Memory for General Data Storage" section on page 5-22. The other three clients are described in the following subsections. As clients are added to the Flash Memory System Builder, the Flash Memory System is partitioned and configured accordingly. Once each client's memory partition is configured, the HDL code is generated for the Flash Memory System and is ready to be interfaced with the design project. Figure 5-1 on page 5-2 portrays the embedded flash with initialization IP system interconnects to its clients.

The Flash Memory Block System Builder supports the following initialization clients:

- The Analog System Client is used to configure the flash memory for the storage of the Fusion Analog System initialization and configuration parameters, which are stored in the spare pages of the flash memory.
- The RAM Initialization Client is used to create and configure a flash memory partition to store RAM initialization data to be reloaded at power-up for context-saving applications.
- The Standalone Initialization Client is used to create and configure a general-purpose flash memory partition to store initialization data interfaced to, for example, a RAM/FIFO or ROM emulation. The initialization interface must be custom-designed.

Each client spans a minimum of one page (128 bytes) and can span up to 2,048 pages, depending on the number of free pages available. The Analog System Client itself does not take any of the regular pages; it is stored entirely in the reserved (spare) pages.

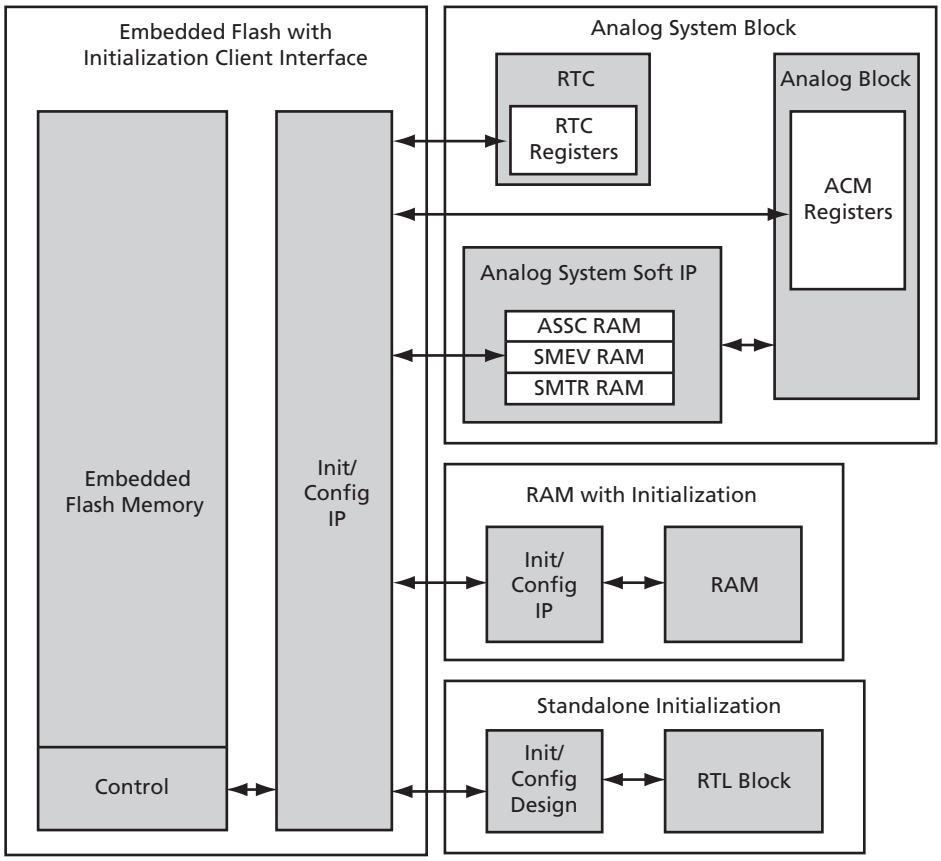


Figure 5-1 • **Embedded Flash with Initialization Client System Interconnects**

Embedded Flash Initialization IP Interface

The Flash Memory System Builder has the capability to generate the embedded flash with an integrated initialization IP circuit. The initialization circuit was designed to read data from embedded flash and store its contents in usable volatile registers or RAM for quick and easy accesses by the on-chip systems. The initialization circuit includes a common interface between the flash memory block and its supported clients. Specific write enable or data control signals are included in the circuit to control the write and read accesses between the clients and embedded flash.

The Analog System Client and RAM Initialization Client are the two key clients that interface directly to the flash initialization circuit. A user design block can also be interfaced to the embedded flash for initialization with the use of the Standalone Initialization Client. However, the initialization interface bridge must be designed by the user and added to the user block design.

Once the initialization clients and flash memory have been configured, the HDL is generated per the specifications entered in the Libero IDE GUI. The embedded flash memory module includes a common initialization interface between all clients and the control signals specific to the client. Table 5-1 on page 5-3 includes a list of all the common initialization interface ports and their descriptions.

Table 5-1 • **Flash Initialization Interface Common Client Ports**

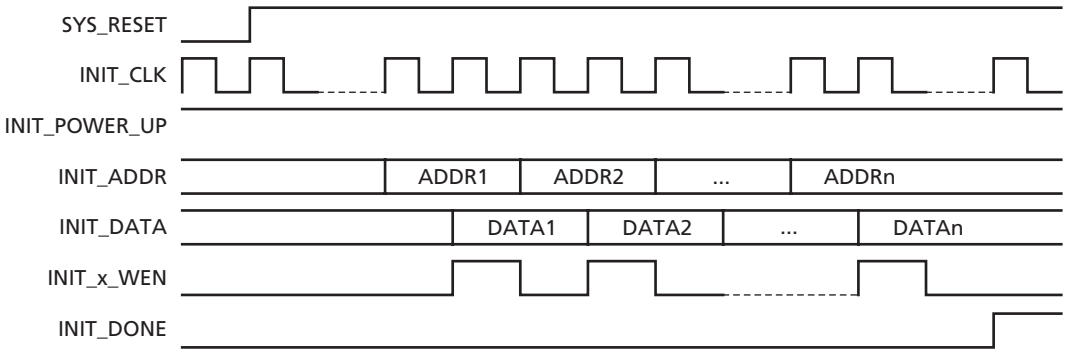
Flash Port	Direction	Description
SYS_RESET	Input	Asynchronous active-low system reset signal that will hold the flash memory in an initial state until released. This input is often common to the FPGA design's main reset.
INIT_CLK	Input	Initialization clock input whose maximum operating frequency is 10 MHz; it is rising-edge-active. For synchronous operation between the flash and its clients, its source clock should be common.
INIT_POWER_UP	Input	Active-high input to the flash, used to activate the initialization IP circuitry at power-up. INIT_POWER_UP must be HIGH at least until INIT_DONE transitions from LOW to HIGH. It can typically be tied HIGH unless performing on-demand updates to the system volatile registers as described in the "Managing the Initialization Process for Power Management" section on page 5-6. If being controlled, its transitions must be synchronous to INIT_CLK.
INIT_DONE	Output	Active-high initialization-done signal. Upon flash SYS_RESET, INIT_DONE defaults to LOW and remains LOW during the initialization activity. It transitions HIGH synchronously to INIT_CLK once the initialization process completes.
INIT_ADDR	Output	This 9-bit initialization address bus is common to the entire flash initialization system. During initialization and the save-to-flash activity, INIT_ADDR synchronously transitions with the rising edge of INIT_CLK. After flash SYS_RESET, INIT_ADDR defaults to 0x000. For each initialization client, INIT_ADDR begins at 0x000 and sequentially increments through all addresses in the flash page(s) assigned to the client's memory partition.
INIT_DATA	Output	The 9-bit initialization data bus is common to the entire flash initialization system. During the initialization activity, INIT_DATA synchronously transitions with the rising edge of INIT_CLK. After flash SYS_RESET, INIT_DATA defaults to 0x000. Depending on the client being initialized, either all nine bits or only the eight least significant bits are utilized as data.
INIT_x_WEN	Output	Active-high write enable (chip select) control signals from flash to the Analog System's volatile registers. INIT_x_WEN is a generic description for multiple signals to the Analog Block (AB). Refer to the "Analog System Client" section on page 5-7 for name specifics. Each signal is a 1-bit port from the Flash Block. After flash SYS_RESET, all signals default to LOW. During initialization activity, each signal synchronously transitions with the rising edge of INIT_CLK. Only one signal is activated at a time. Once activated, it will remain active until all addresses in the defined memory space are accessed.

Table 5-1 • **Flash Initialization Interface Common Client Ports (continued)**

Flash Port	Direction	Description
<RAM_CLIENT>_x_DAT_VAL	Output	Active-high data valid control signal (chip select) from the flash to the RAM block(s). <RAM_CLIENT>_x_DAT_VAL is a generic description for multiple signals to the RAM block. Refer to the "RAM Initialization Client" section on page 5-10 for name specifics. Each signal is a 1-bit port from the Flash Block. After flash SYS_RESET, all signals default to LOW. During initialization or save-to-flash activity, each signal synchronously transitions with the rising edge of INIT_CLK. Only one signal is activated at a time. Once activated, it will remain active until all addresses in the client's defined memory space are accessed.
<CLIENT_NAME>_<CHIP_SELECT>	Output	Active-high chip select for the initialization client from the flash to the defined block. <CLIENT_NAME>_<CHIP_SELECT> is a generic description for multiple signals to the block to be initialized. Refer to the "Standalone Initialization Client" section on page 5-14 for name specifics. Each signal is a 1-bit port from the Flash Block. After flash SYS_RESET, all signals default to LOW. During initialization or save-to-flash activity, each signal synchronously transitions with the rising edge of INIT_CLK. Only one signal is activated at a time. Once activated, it will remain active until all addresses in the client's defined memory space are accessed.

At power-up, after the flash reset is inactive, the initialization circuit reads the partitions from flash and writes the data to volatile memory. The INIT_POWER_UP signal must be HIGH, and INIT_DONE LOW, to trigger this process. If all clients are to initialize only at power-up, the INIT_POWER_UP signal can be tied HIGH. Once the initialization process begins, the INIT_DONE output flag is asserted LOW. During its LOW state, the main system design can use the INIT_DONE signal as a mask to prevent the system design from accessing the Analog System or the other clients being initialized during this process. None of the initialization clients should be used, including the on-chip ADC, until the initialization process completes.

The initialization circuit first initializes the Analog System block, followed by the RAM and other clients. For each client being initialized, a chip select or data valid control signal is produced by the Flash Memory System. As shown in Figure 5-2 on page 5-5, the INIT_x_WEN signal pulses HIGH when the write action to the volatile memory can be synchronously executed. Once the initialization process completes, the INIT_DONE signal transitions from LOW to HIGH, indicating that the process has completed and normal device operation can proceed. Figure 5-3 on page 5-5 shows a simplified simulation of the initialization process at power-up. The specific write enable or data valid signals are shown pulsing, activating the initialization process for each particular client in the system.



Note: INIT_x_WEN is a generic description of the write enable (or chip select) output signal behavior from flash to the volatile memory. Refer to the individual client sections for the exact signal names.

Figure 5-2 • **Basic Initialization Timing Diagram**

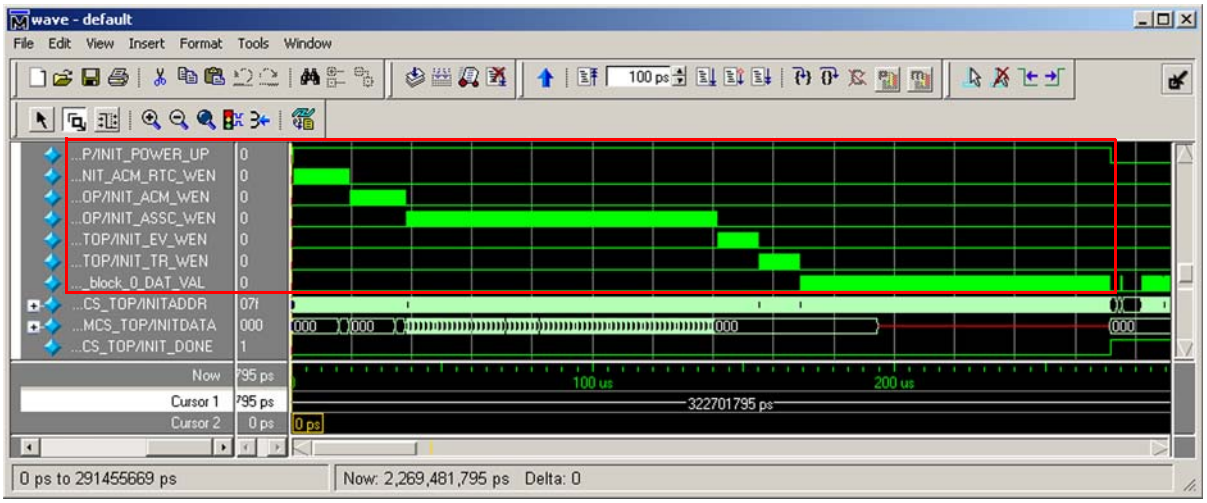


Figure 5-3 • **Complete Initialization Process at Power-Up**

Clock Configuration Options

The initialization process must operate at a frequency of 10 MHz or less. For a synchronous design, the flash module INIT_CLK input should also be connected to the flash initialization client's clock input pins. The Fusion No-Glitch MUX (NGMUX) macro can be used to switch between the slower frequency for the initialization process (INIT_CLK network) and a higher frequency that will utilize the full performance of the embedded Flash Block. Figure 5-4 shows an example of NGMUX usage in a system design.

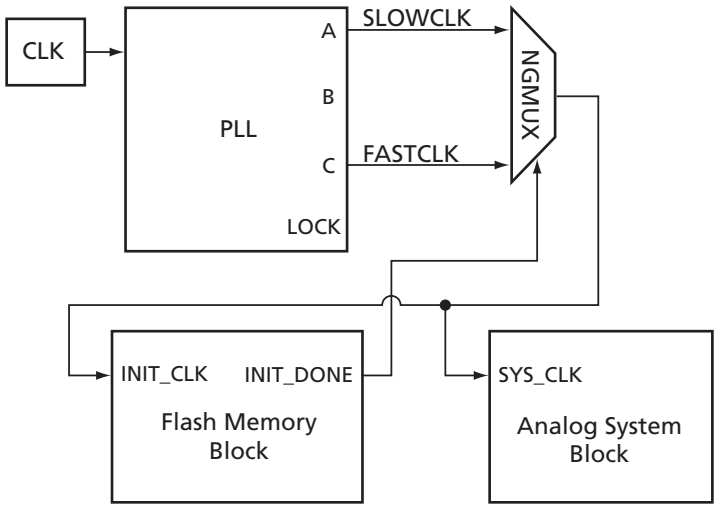


Figure 5-4 • **Analog System and Flash Memory Block NGMUX Example**

Managing the Initialization Process for Power Management

The initialization process can be managed to perform on-demand context (data) save and reload for power management purposes so critical data will not be lost in sleep or standby low-power modes. When entering a low-power mode, the main system design can perform a context save, for example, from RAM to flash. Once the context save is complete, the FPGA can be powered off or placed in low-power state without losing its critical data. Once the FPGA is to resume its normal operation, the initialization process begins performing the context reload operation with the preserved data from the last context save. For a complete Fusion context save and reload reference design, refer to the *Context Save and Reload* application note.

The Flash Initialization IP circuitry has the built-in capability to perform the context save and reload operations via the RAM Initialization Client and Standalone Initialization Client. The CLIENT_UPDATE input to the flash module is used to control the context save operation, and the INIT_POWER_UP signal is used to control the context reload. The "RAM Initialization Client" section on page 5-10 provides details on performing the context save operation.

After FPGA power-up, if the INIT_POWER_UP signal is HIGH, the initialization process will occur, performing the context reload. However, if an on-demand context reload operation is needed in the application, the initialization circuitry can be manipulated to perform addition context reload operations without powering the device down. The key is to clear the INIT_DONE signal so the HIGH INIT_POWER_UP can trigger a new context reload. This can be achieved by creating a controlled flash SYS_RESET signal (FLASH_RESET).

FLASH_RESET should be a registered active-low signal with a reset that is the main system reset for the design. If Fusion's internal Voltage Regulator Power Supply Monitor (VRPSM) is being used, a power-on reset pulse can be generated by connecting a simple external RC circuit to the 3.3 V power supply. For an internal power-on reset, if the PLL is being used in the design, the PLL lock signal can also serve as the FPGA system reset. After a system reset, the FLASH_RESET should default to HIGH. Once a command is received by the control logic to perform a context reload, the FLASH_RESET signal should be synchronously pulsed LOW, clearing the INIT_DONE signal and triggering a new initialization process, which performs the context reload. The initialization

process will, however, initialize all its clients connected to the initialization interface. The write enable or data valid control signals to those clients that should not be updated must be masked to LOW, preventing the writes from occurring. This is especially recommended for the Analog System module. Refer to the *Context Save and Reload* application note for complete implementation details. The INIT_DONE signal should also be masked to HIGH for the Analog System.

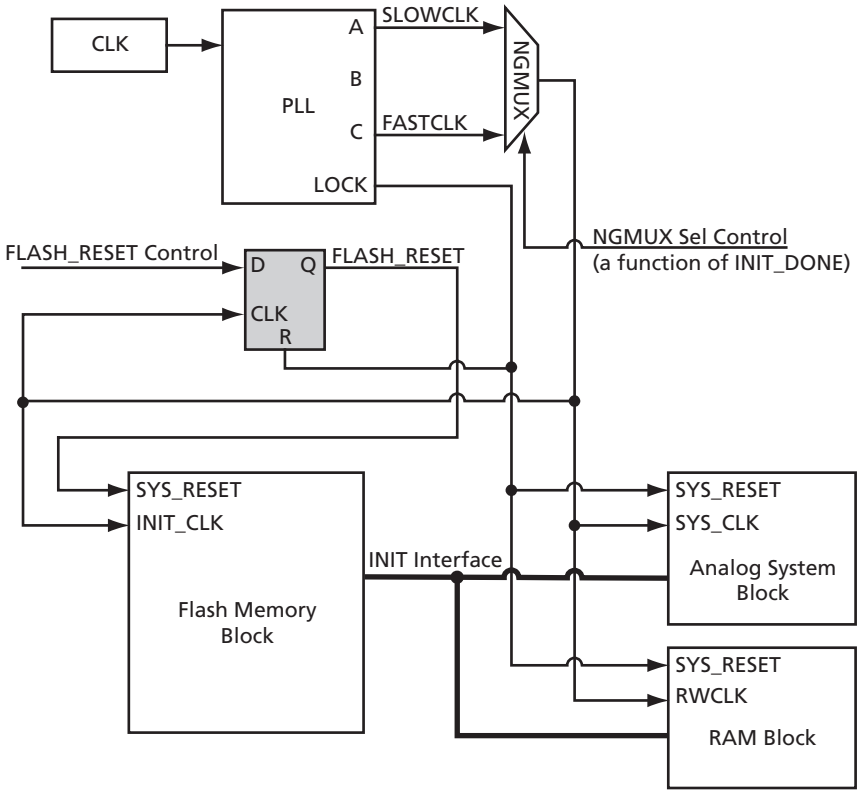


Figure 5-5 • **Reset and Clock Connection Diagram for On-Demand Context Reload**

Analog System Client

When creating the Analog System in Libero IDE using the Analog System Builder, a configuration file is generated and its data stored in the spare pages within the embedded flash memory during FPGA programming. The Flash Memory Analog System Client is used to create the memory partitions to store this configuration data.

The Analog System uses the embedded flash memory to hold the nonvolatile configuration data for the analog subsystem. After power-up and during the initialization process, the flash memory is read and the data stored in the Analog System's volatile register or RAM blocks within the analog subsystem.

The analog subsystem functions initialized during the initialization process are as follows (if selected during the Analog System configuration):

- Analog Configuration MUX (ACM)
- Programmable Real-Time Counter (RTC)
- ADC Sample Sequence Controller (ASSC)
- System Monitor Evaluation Phase State Machine (SMEV)
- System Monitor Transition Phase State Machine (SMTR)

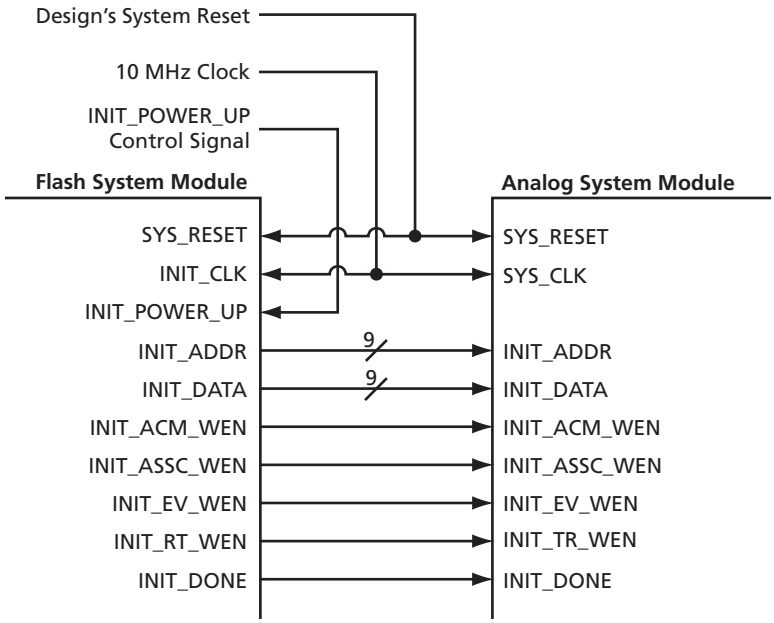
Analog System and Flash Memory Interconnects

The Flash Initialization IP circuitry includes a common interface connected to all clients, as described in Table 5-1 on page 5-3. However, each client includes interconnects that serve purposes specific to the client, such as the write enable signals to the Analog System register and RAM blocks. The write enable signals are active only during the initialization process triggered by the HIGH state of INIT_POWER_UP and the LOW state of INIT_DONE. Table 5-2 describes all Analog System Client–specific signals from the Flash Initialization IP circuit to the Analog System block.

Table 5-2 • **Analog System Client-Specific Flash Ports**

Flash Port	Direction	Description
INIT_ACM_RTC_WEN	Output	Active-high RTC peripheral chip select and write enable control signals from flash to the Analog System's RTC volatile registers. After flash SYS_RESET, INIT_ACM_RTC_WEN defaults to LOW. During the initialization activity, INIT_ACM_RTC_WEN synchronously transitions with the rising edge of INIT_CLK. Once activated, it will remain active until all addresses in the RTC memory space are accessed. No other chip select signals will be activated until the INIT_ACM_RTC_WEN activity completes.
INIT_ACM_WEN	Output	Active-high ACM registers' chip select and write enable control signals from flash to the Analog System's ACM volatile registers. After flash SYS_RESET, INIT_ACM_WEN defaults to LOW. During the initialization activity, INIT_ACM_WEN synchronously transitions with the rising edge of INIT_CLK. Once activated, it will remain active until all addresses in the ACM memory space are accessed. No other chip select signals will be activated until the INIT_ACM_WEN activity completes.
INIT_ASSC_WEN	Output	Active-high ASSC memory chip select and write enable control signals from flash to the Analog System's ACM RAM block. After flash SYS_RESET, INIT_ASSC_WEN defaults to LOW. During the initialization activity, INIT_ASSC_WEN synchronously transitions with the rising edge of INIT_CLK. Once activated, it will remain active until all addresses in the ACM memory space are accessed. No other chip select signals will be activated until the INIT_ASSC_WEN activity completes.
INIT_EV_WEN	Output	Active-high SMEV memory chip select and write enable control signals from flash to the Analog System's SMEV RAM block. After flash SYS_RESET, INIT_EV_WEN defaults to LOW. During the initialization activity, INIT_EV_WEN synchronously transitions with the rising edge of INIT_CLK. Once activated, it will remain active until all addresses in the SMEV memory space are accessed. No other chip select signals will be activated until the INIT_EV_WEN activity completes.
INIT_TR_WEN	Output	Active-high SMTR memory chip select and write enable control signals from flash to the Analog System's SMTR RAM block. After flash SYS_RESET, INIT_TR_WEN defaults to LOW. During the initialization activity, INIT_TR_WEN synchronously transitions with the rising edge of INIT_CLK. Once activated, it will remain active until all addresses in the SMTR memory space are accessed. No other chip select signals will be activated until the INIT_TR_WEN activity completes.

Figure 5-6 is a connection diagram showing the interconnects between the flash memory and the Analog System. If other flash memory clients exist in the system, they will also be connected to the common initialization interface ports.



Note: The above names represent the actual port names for the flash and Analog System HDL modules.

Figure 5-6 • **Analog System and Flash Memory Initialization Interface Connection Diagram**

Figure 5-7 is a timing diagram showing the write activity from flash memory to the Analog System's ASSC RAM block. At the positive edge of clock, the initialization circuit synchronously generates the address of the data to be written. At the following positive edge of the clock, the data is produced and INIT_ASSC_WEN is pulsed HIGH. The INIT_ASSC_WEN signal to the Analog System serves as an ASSC RAM chip select and write enable. The data is then synchronously written to the ASSC RAM memory on the next positive edge of the clock.

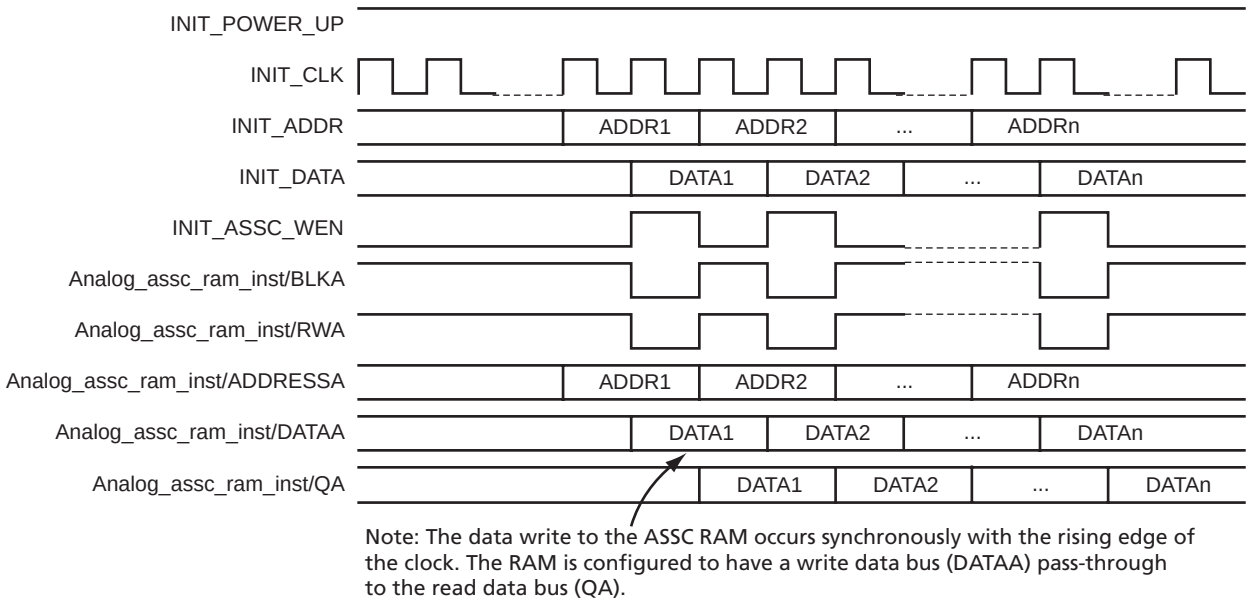


Figure 5-7 • **Initialization Interface Writing to ASSC RAM**

RAM Initialization Client

The RAM Initialization Client allows the user to create a flash memory partition to permanently store the RAM initialization data. After power-up, during the initialization process, the data stored in the RAM initialization flash memory partition is read, and its data is written to the RAM blocks. The Flash Memory System Builder allows for multiple RAM initialization flash memory partitions, giving the designer better flexibility over the system design's critical data. A JTAG read and/or write protection option is also available for each RAM Initialization Client.

When creating the RAM in Libero IDE, the **RAM with Initialization** option must be selected before generating the RAM so the initialization IP is included in the RAM block. The RAM initialization core's configuration GUI allows for the configuration of both two-port and dual-port RAMs with combined or separate read and write clocks. It also provides the ability to save the RAM contents back into flash via the **Enable On-Demand Save to Flash Memory** option in the RAM with Initialization configuration GUI. The initialization interface for the RAM shares the RAM's clocks; therefore, during the initialization or save-to-flash process, the RAM clock frequency must be no greater than 10 MHz.

RAM and Flash Memory Initialization Interconnects

The Flash Initialization IP circuitry includes a common interface connected to all clients, as described in Table 5-1 on page 5-3. However, each client includes interconnects that serve purposes specific to the client, such as the save-to-flash interface signals of the RAM with Initialization core IP. These signals are active only during the initialization process triggered by the HIGH state of INIT_POWER_UP while INIT_DONE is LOW, and during a save-to-flash process triggered by a HIGH state of CLIENT_UPDATE. Table 5-3 describes all signals from the Flash Initialization IP circuit specific to RAM with Initialization.

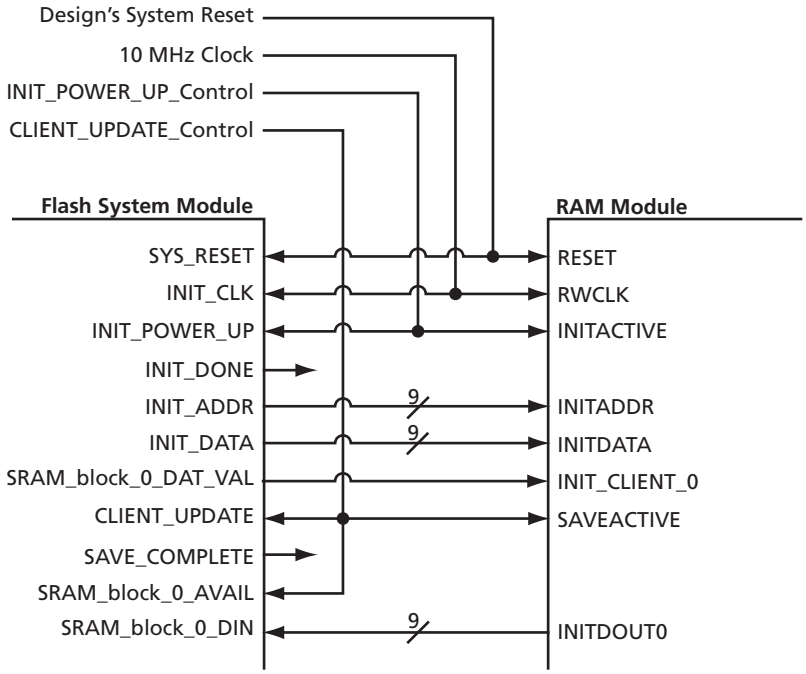
Table 5-3 • **RAM with Initialization Client-Specific Flash Ports**

Flash Port	Direction	Description
<RAM_CLIENT>_x_DAT_VAL	Output	Active-high data valid signal that behaves as a chip select for the RAM client. The number of data valid signals (labeled 0 to *x*) depends on the number of RAM blocks used to satisfy the RAM configuration selected in Libero IDE. Each signal is a 1-bit output port of the Flash Block. After flash SYS_RESET, the data valid signal(s) default to LOW. <RAM_CLIENT>_x_DAT_VAL synchronously transitions with the rising edge of INIT_CLK during initialization and save-to-flash activity. Once activated, it will remain active until all addresses in the RAM memory space are accessed. No other chip select signals will be activated until all <RAM_CLIENT>_x_DAT_VAL activity completes.
CLIENT_UPDATE	Input	Active-high control input signal used to trigger the save-to-flash process, which reads data from RAM and stores it to flash for context saving. CLIENT_UPDATE is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. CLIENT_UPDATE must be LOW at power-up and held LOW until the initialization process completes (INIT_DONE transitions from LOW to HIGH with INIT_POWER_UP held HIGH). To trigger the save-to-flash activity, CLIENT_UPDATE must be transitioned from LOW to HIGH synchronously with INIT_CLK. It must be held HIGH until the save-to-flash activity completes (SAVE_COMPLETE pulses HIGH). Once complete, CLIENT_UPDATE must then be transitioned LOW synchronously with INIT_CLK.

Table 5-3 • **RAM with Initialization Client-Specific Flash Ports (continued)**

Flash Port	Direction	Description
SAVE_COMPLETE	Output	Active-high output flag used to identify when the on-demand save-to-flash memory operation completes. SAVE_COMPLETE is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. After flash SYS_RESET, SAVE_COMPLETE defaults to LOW. The save-to-flash activity is triggered by the HIGH state of CLIENT_UPDATE. Once complete, SAVE_COMPLETE will transition HIGH and back LOW synchronously with INIT_CLK.
<RAM_CLIENT>_x_AVAIL	Input	Active-high data available input that behaves as a chip-select to the RAM-Initialization flash memory partition. The number is data valid signals (labeled '0' to 'x') depend on the number of RAM blocks used to satisfy the RAM configuration selected in Libero IDE. Each signal is a 1-bit input port of the Flash Block. <RAM_CLIENT>_x_AVAIL is only available if the "Enable On-Demand save to Flash Memory" feature was selected in the Libero IDE GUI. These input(s) can be connected directly to the CLIENT_UPDATE control input port. <RAM_CLIENT>_x_AVAIL must be Low at power-up and held Low the save to flash activity is triggered by the High state of CLIENT_UPDATE. <RAM_CLIENT>_x_AVAIL must transition High and held High until the save to flash activity completes. Once complete, <RAM_CLIENT>_x_AVAIL must be transitioned Low synchronously with INIT_CLK.
<RAM_CLIENT>_block_x_DIN	Input	The 9-bit save-to-flash data bus that supplies the data read from RAM to be written to flash. <RAM_CLIENT>_block_x_DIN is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. The number of data busses (labeled 0 to *x*) depends on the number of RAM blocks used to satisfy the RAM configuration selected in Libero IDE. Each data bus is a 9-bit input port of the Flash Block. During save-to-flash activity, <RAM_CLIENT>_block_x_DIN synchronously transitions with the rising edge of the RAM block's RWCLK, RCLK, or port-B clock (depending on the RAM configuration). After flash SYS_RESET, <RAM_CLIENT>_block_x_DIN defaults to 0x000. Depending on the client configuration, either all nine bits or only the eight least significant bits are utilized as data.

Figure 5-8 is a connection diagram showing the interconnects between the flash memory and RAM initialization ports. If other flash memory clients exist in the system, they will also be connected to the common initialization interface ports.



Note: The above names represent the actual port names for the flash and RAM HDL modules.

Figure 5-8 • **Two-Port RAM and Flash Memory Initialization Interface Connection Diagram**

Figure 5-9 is a timing diagram showing the write activity from flash memory to the RAM block through the initialization interface. At the positive edge of the clock, the initialization circuit synchronously generates the address of the data to be written. At the following positive edge of the clock, the data is produced and the SRAM_block_0_DAT_VAL signal is pulsed HIGH. The SRAM_block_0_DAT_VAL signal to the RAM serves as a chip select and write enable. The data is then synchronously written to the RAM on the next positive edge of the clock.

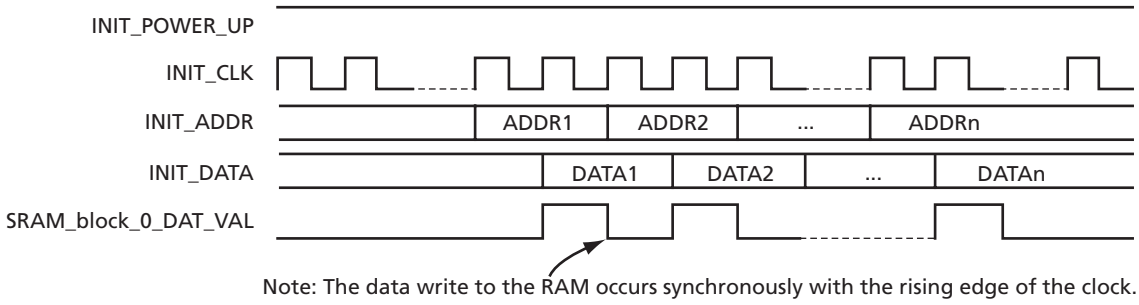


Figure 5-9 • **RAM Initialization Process Timing Detail**

Figure 5-10 on page 5-13 is a simulation example showing the complete save-to-flash process. To start the process, at the positive edge of the clock, CLIENT_UPDATE is transitioned HIGH. The flash initialization circuit provides the read address from RAM on the positive edge of the clock, as shown in Figure 5-11 on page 5-13. At the following positive edge of the clock, the data is read and put out from INITDOUT0. The flash initialization circuit then stores the data in the page buffer at the next positive edge of the clock. Once all data has been read from RAM and stored in flash, the

SAVE_COMPLETE signal is synchronously pulsed HIGH at the positive edge of the clock, holding a HIGH state for several clock cycles.

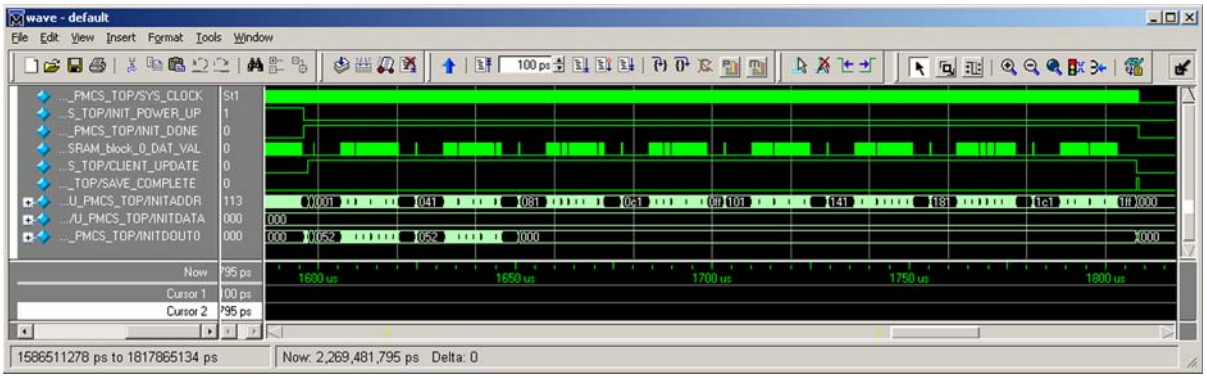


Figure 5-10 • **Complete Save-to-Flash Process**

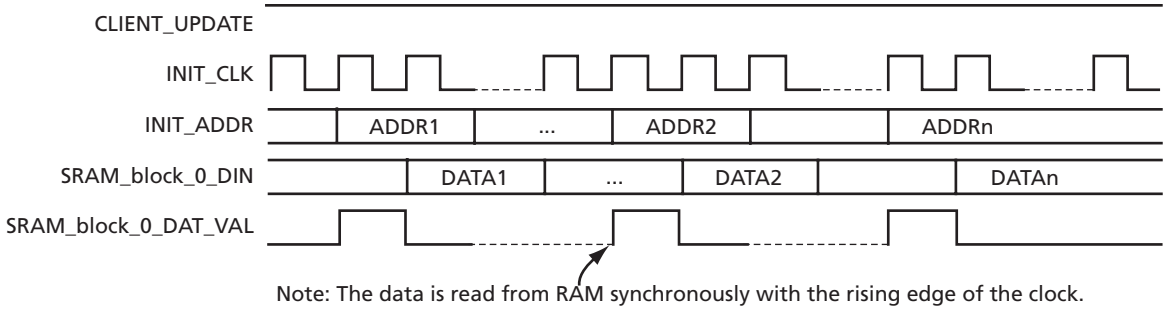


Figure 5-11 • **Save-to-Flash Timing Details**

Two-Port vs. Dual-Port RAM with Initialization Clock Configurations

The RAM with Initialization core IP can be configured with a single clock for both read and write operations, or with separate read and write clocks. Since the RAM initialization IP blocks share the standard RAM clock ports, the 10 MHz maximum frequency of the initialization interface must be considered when defining the system clocks.

For both the two-port and dual-port RAM with Initialization core configurations, if a single read and write clock is selected, the NGMUX macro can be used to switch between the slower frequency for the initialization or save-to-flash process and the faster frequency used to perform normal RAM accesses. Figure 5-5 on page 5-7 gives a possible clock configuration using the NGMUX as described.

For the two-port RAM with Initialization core configuration, if separate read and write clock ports are selected, the write clock port is used by the RAM during the initialization process. If the on-demand save-to-flash feature is enabled, the read clock port is used by the RAM during the save-to-flash process, and the read data bus is configured as non-pipelined.

For the dual-port RAM with Initialization core configuration, if the separate port-A and port-B clock option is selected, the port-A clock is used to perform the RAM write actions during the initialization process. If the on-demand save-to-flash feature is enabled, the port-B clock is used to perform the RAM read actions during the save-to-flash process, and the output data bus on port-B is configured as non-pipelined. If on-demand save-to-flash is not enabled, port-B is free to run at any desired frequency within the RAM operating range.

Both the initialization and save-to-flash processes must operate at a frequency no greater than 10 MHz. The NGMUX can be used to switch between a slower and a faster clock. For a synchronous

design, Actel recommends that a common clock source be used for all initialization modules, as shown in Figure 5-12.

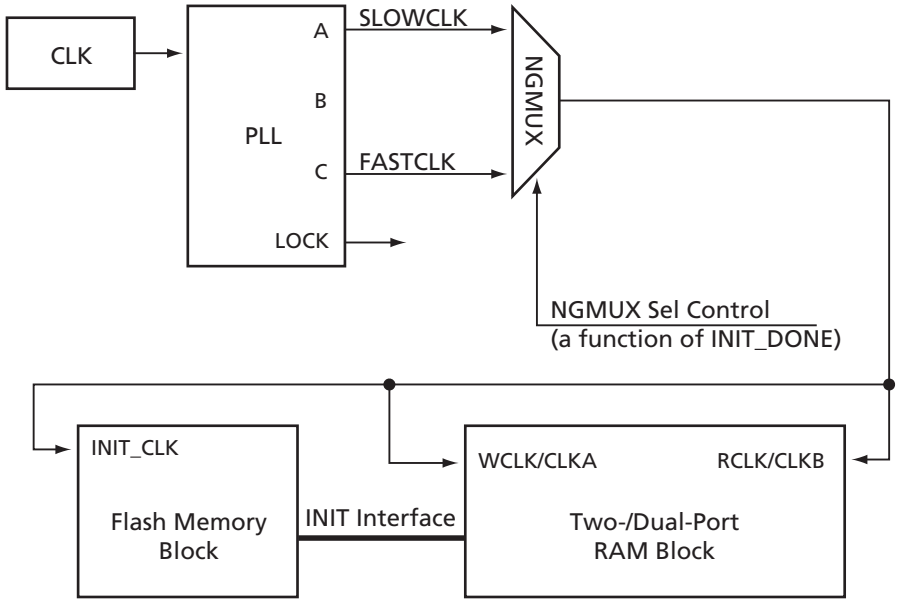


***Figure 5-12* • RAM Initialization Two-Clock Configuration Diagram**

Standalone Initialization Client

The standalone Flash Memory System initialization IP initializes all its clients with the data stored in the associated flash memory partition during the initialization process at power-up. The initialization client provides the ability to create a flash memory partition to initialize a desired block's volatile memory or registers. Once the partition has been created, access is given to the flash memory's initialization IP interface ports specific to the client's needs.

The Flash Memory System Builder's Initialization Client Libero IDE GUI allows the user to specify the client's name, starting address, word size, and memory-contents file. The memory-contents file supplies the initialization data values to be stored in the client's flash memory partition during FPGA programming. The user can also select the **Enable On-Demand Save to Flash Memory** feature, and protect the flash memory partition from JTAG access. And the user can name the client's chip select and save request ports.

Initialization Client Flash Memory Ports

The Flash Initialization IP circuitry includes a common interface connected to all clients, as described in Table 5-1 on page 5-3. However, each client includes interconnects that serve purposes specific to the client, such as the save-to-flash interface signals. These signals are active only during the initialization process triggered by the HIGH state of INIT_POWER_UP while INIT_DONE is LOW, and during a save-to-flash process triggered by a HIGH state of CLIENT_UPDATE. Table 5-4 on page 5-15 describes all Standalone Initialization Client–specific signals from the Flash Initialization IP circuit.

Table 5-4 • **Initialization Client-Specific Flash Ports**

Flash Port	Direction	Description
<CLIENT_NAME>_<CHIP_SELECT>	Output	Active-high chip select for the Standalone Initialization Client. After flash SYS_RESET, <CLIENT_NAME>_<CHIP_SELECT> defaults to LOW. During initialization and save-to-flash activity, <CLIENT_NAME>_<CHIP_SELECT> synchronously transitions with the rising edge of INIT_CLK. Once activated, it will remain active until all addresses in the client's memory space are accessed. No other chip select signals will be activated until the <CLIENT_NAME>_<CHIP_SELECT> activity completes.
CLIENT_UPDATE	Input	Active-high control input signal used to trigger the save-to-flash process, which reads data from client and stores it to flash for context saving. CLIENT_UPDATE is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. CLIENT_UPDATE must be LOW at power-up and held LOW until the initialization process completes (INIT_DONE transitions from LOW to HIGH with INIT_POWER_UP held HIGH). To trigger the save-to-flash activity, CLIENT_UPDATE must be transitioned from LOW to HIGH synchronously with INIT_CLK. It must be held HIGH until the save-to-flash activity completes (SAVE_COMPLETE pulses HIGH). Once complete, CLIENT_UPDATE must then be transitioned LOW synchronously with INIT_CLK.
SAVE_COMPLETE	Output	Active-high output flag used to identify when the on-demand save-to-flash memory operation completes. SAVE_COMPLETE is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. After flash SYS_RESET, SAVE_COMPLETE defaults to LOW. Save-to-flash activity is triggered by the HIGH state of CLIENT_UPDATE. Once complete, SAVE_COMPLETE will transition HIGH and back LOW synchronously with INIT_CLK.

Table 5-4 • **Initialization Client-Specific Flash Ports (continued)**

Flash Port	Direction	Description
<CLIENT_NAME>_<SAVE_REQUEST>	Input	Active-high save request input that behaves as a chip select to the initialization client's flash memory partition. The <CLIENT_NAME>_<SAVE_REQUEST> signal is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. These input(s) can be connected directly to the CLIENT_UPDATE control input port. The <CLIENT_NAME>_<SAVE_REQUEST> signal must be LOW at power-up and held LOW. Save-to-flash activity is triggered by the HIGH state of the CLIENT_UPDATE signal. The <CLIENT_NAME>_<SAVE_REQUEST> signal must transition HIGH and be held HIGH until the save-to-flash activity completes. <CLIENT_NAME>_<SAVE_REQUEST> must be transitioned LOW synchronously with INIT_CLK once complete.
<CLIENT_NAME>_DIN	Input	The 9-bit save-to-flash data bus that supplies the data read from initialization client to be written to flash. <CLIENT_NAME>_DIN is only available if the **Enable On-Demand Save to Flash Memory** feature was selected in the Libero IDE GUI. During the save-to-flash activity, <CLIENT_NAME>_DIN synchronously transitions with the rising edge of INIT_CLK. After flash SYS_RESET, <CLIENT_NAME>_DIN should default to 0x000. Depending on the client configuration, either all nine bits or only the eight least significant bits are utilized as data.

Initialization Client Usages

The Standalone Initialization Client provides designers the utmost flexibility, providing access to the Flash Initialization IP interface to set initial values—for example, for the Fusion synchronous FIFO, for ROM emulation, and for CoreABC (AMBA Bus Controller) IP instruction memory space. An initialization wrapper for the initialization clients must be generated by the designer (except when used with the CoreABC IP). An example of the initialization wrapper design for the Fusion synchronous FIFO is described in the "Fusion FIFO with Initialization Example" section on page 5-17.

To perform ROM emulation, the flash memory's initialization client and RAM blocks are required. A wrapper must be created providing the Flash Initialization IP interface write access to the RAM. The user interface should only include the read access ports of the RAM (RADDR, RD, RCLK, and REN). Depending on the data bus width, proper bus width handling must be taken into consideration in the wrapper design, since the initialization interface writes nine bits of data per chip select. Within the wrapper, the RAM write access ports may be set to a default state tying the WEN port HIGH. The WCLK port should be connected to the 10 MHz initialization clock network.

In some cases, the RAM with Initialization core IP may be the simplest approach for ROM emulation. The initialization interface IP is already included in the RAM core module. When generating the RAM with Initialization block, the two-port RAM with separate read and write clock ports is a recommended configuration. When creating the flash memory RAM Initialization Client partition, the **Enable On-Demand Save to Flash Memory** option should be cleared. All user RAM write access ports may be set to a default state, tying off the WEN port to HIGH. The WCLK port should be connected to the 10 MHz initialization clock network.

CoreABC is a simple, low-gate-count controller that uses the Flash Memory System's initialization client to initialize at power-up the RAM used for instruction code execution. A complete design example can be found in the *Design Example* section of *Interfacing with the Fusion Analog System: Processor/Microcontroller Interface*.

Fusion FIFO with Initialization Example

When creating the FIFO with Initialization design, the flash memory initialization client and synchronous FIFO blocks are required. A wrapper must be created providing the Flash Initialization IP interface write access to the FIFO, and the save-to-flash interface read access. Depending on the data bus width, proper bus width handling must be taken into consideration in the wrapper design, since the initialization interface writes nine bits of data per chip select. Figure 5-13 shows the selected synchronous FIFO configuration used in this example.

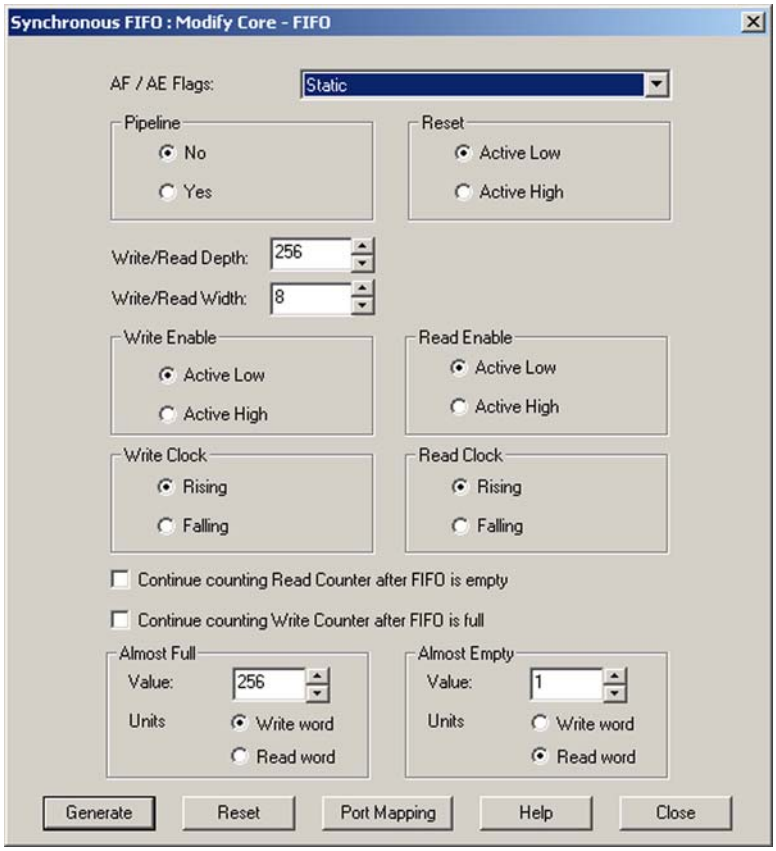


Figure 5-13 • **Synchronous FIFO Configuration**

The Flash Memory System Builder's initialization client is used to generate the flash memory partition for the FIFO. Figure 5-14 shows the required initialization client configuration based on the FIFO configuration shown in Figure 5-13 on page 5-17.

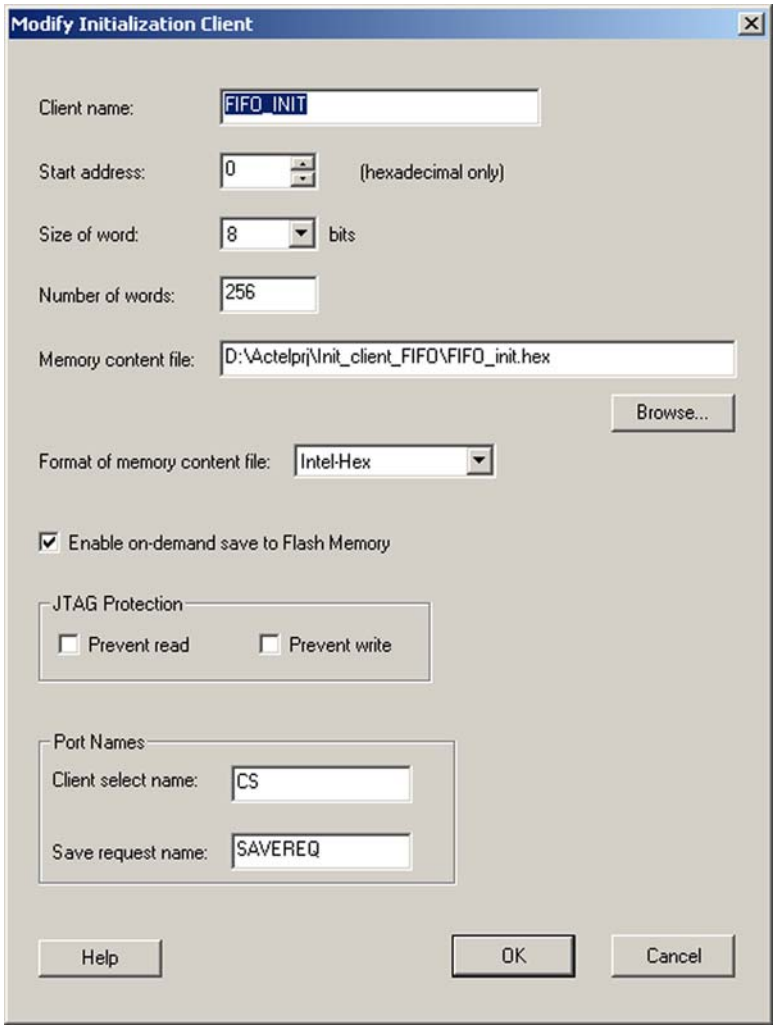
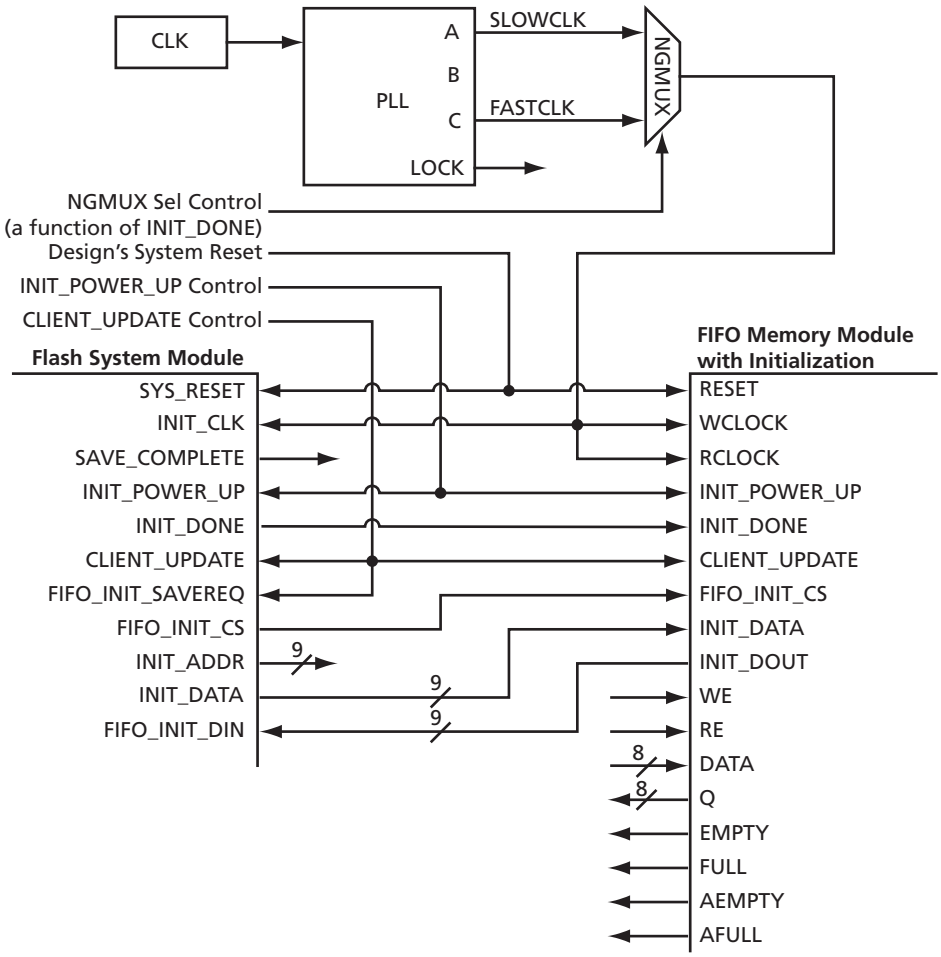


Figure 5-14 • **Initialization Client Configuration for the FIFO**

Once the flash memory block with the FIFO initialization partition has been generated, the following initialization ports, given in Verilog, are added to the Flash Block:

```
input         INIT_CLK;
input         SYS_RESET;
input         INIT_POWER_UP;
output        INIT_DONE;
output [8:0]  INIT_DATA;
output [7:0]  INIT_ADDR;
output        FIFO_INIT_CS;
input         FIFO_INIT_SAVEREQ;
input  [8:0]  FIFO_INIT_DIN;
output        SAVE_COMPLETE;
input         CLIENT_UPDATE;
```

The FIFO with Initialization interface wrapper must connect to the above ports. The INIT_ADDR port is not used in this design, since the Flash Block is being interfaced to a FIFO. Figure 5-15 shows the connections between the Flash Block and the FIFO with Initialization wrapper.



Note: The above names represent the actual port names for the flash and FIFO HDL modules.

Figure 5-15 • **FIFO with Initialization Interface Connection Diagram**

The FIFO with Initialization wrapper design should multiplex the control signals from the flash memory's initialization circuit with the FIFO's user access ports for a dual read and write access FIFO configuration. The following, given in Verilog, describes the main wrapper design:

```
assign FIFO_DATA = (INIT_POWER_UP & !INIT_DONE) ? INIT_DATA[7:0] : DATA;
assign FIFO_WEN = (INIT_POWER_UP & !INIT_DONE) ? !FIFO_INIT_CS : WE;
assign FIFO_REN = CLIENT_UPDATE ? !FIFO_INIT_CS : RE;
assign INIT_DOUT = {1'b0,Q};
FIFO U_FIFO(
  .DATA(FIFO_DATA),
  .Q(Q),
  .WE(FIFO_WEN),
  .RE(FIFO_REN),
  .WCLOCK(WCLOCK),
  .RCLOCK(RCLOCK),
  .FULL(FULL),
  .EMPTY(EMPTY),
  .RESET(RESET),
  .AEMPTY(AEMPTY),
  .AFULL(AFULL));
```

Figure 5-16 is a simulation example showing the FIFO's complete initialization process. A HIGH state on INIT_POWER_UP while INIT_DONE is LOW will trigger the initialization process. During this process, the FIFO should be in an empty state. The FIFO_INIT_CS signal is pulsed with the positive edge of the clock when a FIFO write operation should occur, as shown in Figure 5-17. With every HIGH state of FIFO_INIT_CS, INIT_DATA has valid data to be written into the FIFO. At the positive edge of the clock while FIFO_INIT_CS is HIGH, the data is synchronously written. Once all data values have been written into the FIFO, the FIFO's AFULL flag and the flash memory's INIT_DONE signals synchronously transition HIGH. The user must take care not to access the FIFO via the user access ports or the save-to-flash interface unless the INIT_DONE signal is HIGH.

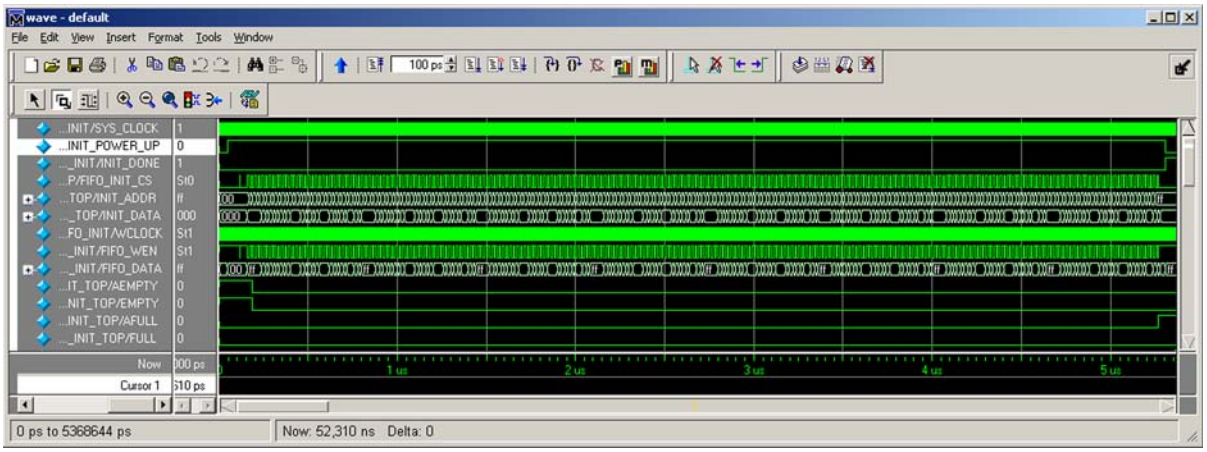
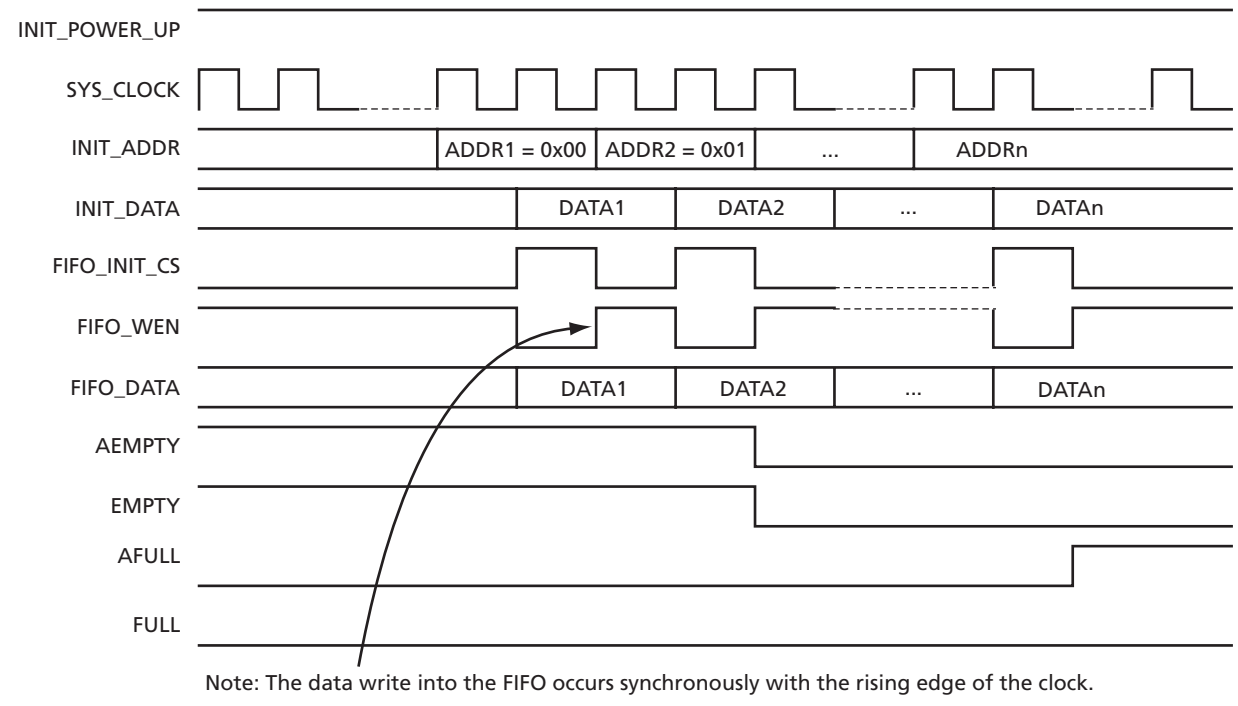


Figure 5-16 • **Complete FIFO Initialization Process**



Note: The data write into the FIFO occurs synchronously with the rising edge of the clock.

Figure 5-17 • **FIFO Initialization Write Operation Timing Details**

Figure 5-18 on page 5-21 is a simulation example showing the FIFO's complete save-to-flash process. To start the process, CLIENT_UPDATE is transitioned HIGH at the positive edge of the clock.

However, the user must take care not to activate CLIENT_UPDATE while INIT_DONE is LOW or if the FIFO is not full. The AFULL flag should be used to monitor the filled state of the FIFO, since the FULL flag will not transition HIGH unless the entire physical FIFO memory has been filled. In this example, although the FIFO is configured as 256×8 and all locations are written, the FIFO4K18 macro is instantiated for this configuration. The FIFO_INIT_CS signal is pulsed with the positive edge of clock when a FIFO read operation should occur, as shown in Figure 5-19. Once all data values have been read from the FIFO and stored in flash, the FIFO EMPTY signal and the flash memory SAVE_COMPLETE signal are synchronously transitioned HIGH at the positive edge of the clock. The SAVE_COMPLETE signal will hold a HIGH state for several clock cycles before transitioning LOW.

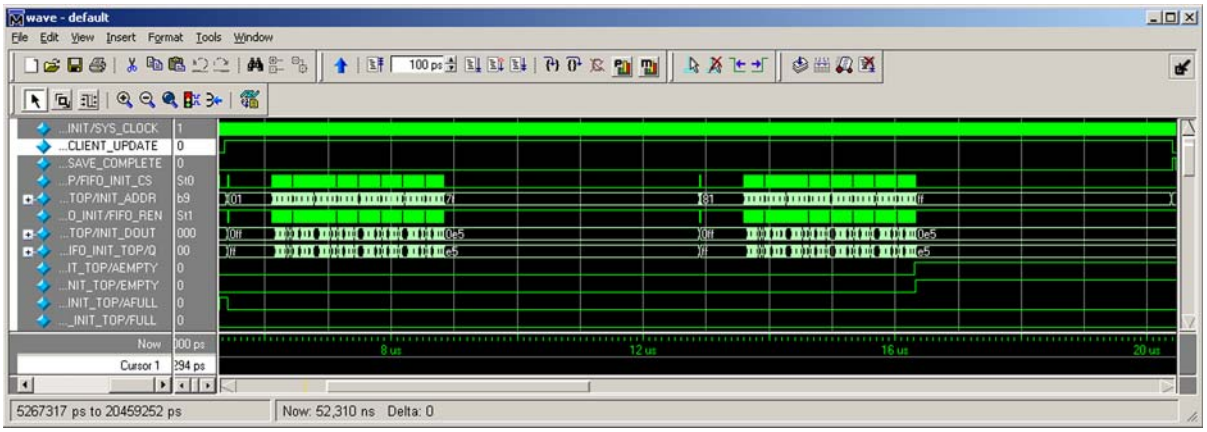


***Figure 5-18* • Complete FIFO Save-to-Flash Process**

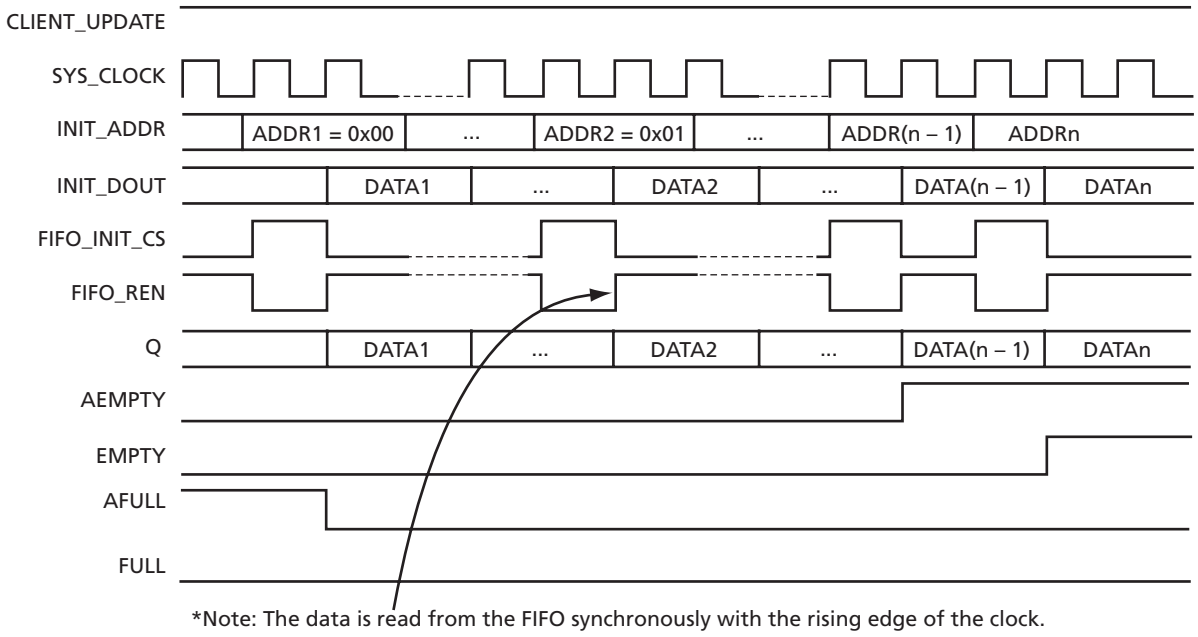


***Figure 5-19* • FIFO Save-to-Flash Read Operation Timing Details**

Using the Embedded Flash Memory for General Data Storage

A key feature of the embedded flash memory is its ability to be used as general data storage by the FPGA fabric. The nonvolatile nature of the flash allows for permanent storage of the design system's key parameters and variables. Fusion's embedded flash memory blocks can be instantiated in a design for general data storage via the following methods:

- The embedded flash memory macro can be instantiated directly into the RTL design in text or through SmartDesign.
- The Flash Memory System Builder's Data Storage Client in Libero IDE can be used to configure the embedded flash memory to be used for general-purpose data storage.
- CoreConsole IDP can be used to interface the embedded flash memory with the Common Flash Interface via the CoreCFI IP. The Flash Memory System Builder in Libero IDE is then used to configure the embedded flash memory to be paired with CoreCFI.

The following sections discuss the basic flash memory operations available to Fusion's embedded flash memory, and its three general data storage usages. Details related to the silicon implementation of the embedded flash memory block, including timing characteristics, can be found in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet.

Flash Memory Macro and Interface

Fusion contains up to four embedded flash memory blocks, each 2 Mbits in density. In an RTL design, a single embedded flash memory block corresponds to a single instance of the embedded flash memory block (NVM) macro. The embedded flash memory block is referenced as "FB" in this section and in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet. However, the macro used to instantiate an embedded flash memory block in a design is named "NVM" and will be referenced as such throughout this section.

The NVM macro can be instantiated directly into an RTL design without the intervention of Libero IDE's Flash Memory System Builder. With the exception of a RESET operation, all operations on an NVM instance are synchronous to the rising edge of the clock. Table 5-5 contains a complete list of the input and output ports for the NVM macro.

Table 5-5 • **Flash Block Macro Port Descriptions**

Flash Port	Direction	Description
RESET	Input	Asynchronous active-low reset input signal. Holds the Flash Block's control logic in an initial state until released.
CLK	Input	Flash memory input clock whose maximum clock period is dictated by $t_{MPWCLKNVM}$. All memory operations and status are synchronous to the rising edge of this clock.
ADDR[17:0]	Input	The 18-bit byte-based Flash Block input address bus. ADDR must transition synchronously with the rising edge of CLK. The minimum addressable data size is 8 bits. For a data width of 16 bits, ADDR[0] is ignored and ADDR[1] becomes the lowest-order address; for a data width of 32 bits, ADDR[1:0] is ignored and ADDR[2] becomes the lowest-order address.
WEN	Input	Active-high write enable control input signal used for writing data into the flash memory Page Buffer. WEN must transition synchronously with the rising edge of CLK.
PROGRAM	Input	Active-high program operation control input signal used for writing the contents of the Page Buffer into the flash memory array page addressed. PROGRAM must transition synchronously with the rising edge of CLK.
REN	Input	Active-high read enable control input signal used for read data from the flash memory array, Page Buffer, block buffer, or status registers. REN must transition synchronously with the rising edge of CLK.

Table 5-5 • **Flash Block Macro Port Descriptions (continued)**

Flash Port	Direction	Description
READNEXT	Input	Active-high read-next operation control input signal. This feature loads the next block relative to that stored in the block buffer from the flash memory array while reads from the block buffer are being performed. READNEXT must be asserted along with REN to initiate the read-next operation, and must transition synchronously with the rising edge of CLK.
RD[31:0]	Output	The 32-bit output read data bus. The data on RD transitions synchronously with the rising edge of CLK. After REN has been asserted, issuing a data read operation, all data must be sampled from RD when BUSY is not asserted (when LOW). Data put out on RD are LSB-oriented. Upon a RESET, RD is initialized to zero.
WD[31:0]	Input	The 32-bit input write data bus. Data put in on WD must be LSB-oriented; any unused pins must be grounded or driven LOW. The data on WD must transition synchronously with the rising edge of CLK.
DATAWIDTH[1:0]	Input	The 2-bit RD and WD data bus width control input signal. DATAWIDTH must transition synchronously with the rising edge of CLK. • If DATAWIDTH is '00', the data busses contain one byte of data forming an 8-bit word (RD/WD[7:0]). • If DATAWIDTH is '01', the data busses contain two bytes of data forming a 16-bit word (RD/WD[15:0]). • If DATAWIDTH is '1x', the data busses contain four bytes of data forming a 32-bit word (RD/WD[31:0]).
PIPE	Input	Active-high pipeline stage control input signal. When asserted, a pipeline stage is added to the read data output. Read operations complete in five cycles instead of the typical four. The addition of the pipeline stage is recommended to be used for CLK frequencies greater than 50 MHz. PIPE must transition synchronously with the rising edge of CLK and be asserted along with REN.
PAGESTATUS	Input	Active-high page status control input signal. When asserted, a page status read operation is initiated. PAGESTATUS must transition synchronously with the rising edge of CLK and be asserted along with REN.
ERASEPAGE	Input	Active-high page erase control input signal asserted when the addressed page is to be programmed with all zeros. ERASEPAGE must transition synchronously with the rising edge of CLK.
DISCARDPAGE	Input	Active-high discard page control input signal asserted when the contents of the Page Buffer are to be discarded so that a new page write can be started. DISCARDPAGE must transition synchronously with the rising edge of CLK.
AUXBLOCK	Input	Active-high auxiliary block select input signal asserted when the page address is used to access the auxiliary block within the page addressed. AUXBLOCK must transition synchronously with the rising edge of CLK.
SPAREPAGE	Input	Active-high spare page select input signal asserted when the sector addressed is used to access the spare page within that sector. SPAREPAGE must transition synchronously with the rising edge of CLK.
UNPROTECTPAGE	Input	Active-high unprotect page control input signal used to clear the protection of the addressed page. UNPROTECTPAGE must transition synchronously with the rising edge of CLK.

Table 5-5 • **Flash Block Macro Port Descriptions (continued)**

Flash Port	Direction	Description
OVERWRITEPAGE	Input	Active-high overwrite page control input signal asserted when the page addressed is to be overwritten, if writable, with the contents of the Page Buffer. OVERWRITEPAGE must transition synchronously with the rising edge of CLK and must be asserted along with PROGRAM.
OVERWRITEPROTECT	Input	Active-high overwrite protect control input signal used to change the protection bit of the addressed page. OVERWRITEPROTECT must transition synchronously with the rising edge of CLK and must be asserted along with PROGRAM or ERASEPAGE.
PAGELOSSPROTECT	Input	Active-high page-loss protect control input signal used to prevent writes to any other page except the current addressed page in the Page Buffer, until the page is either discarded or programmed. PAGELOSSPROTECT must transition synchronously with the rising edge of CLK and be asserted along with PROGRAM or ERASEPAGE.
LOCKREQUEST	Input	Active-high lock request control input signal asserted when user access (including JTAG) to the flash memory array is to be prevented. LOCKREQUEST must transition synchronously with the rising edge of CLK.
BUSY	Output	Active-high busy control output signal used to indicate when the flash memory is performing an operation. BUSY transitions synchronously with the rising edge of CLK. Upon a RESET, BUSY pulses HIGH for several cycles before settling to a LOW state.
STATUS[1:0]	Output	The 2-bit flash memory operation status output signals used to indicate the status of the last completed operation. STATUS transitions synchronously with the rising edge of CLK. Upon a RESET, STATUS is initialized to 0x00. • When STATUS is '00', it indicates that the last operation completed successfully. • When STATUS is '01' after a read operation, it indicates that a single error was detected during the last completed operation and was corrected. • When STATUS is '01' after a write operation, it indicates that the last completed operation addressed a write-protected page. • When STATUS is '01' after an erase-page/program operation, it indicates that the Page Buffer was unmodified during the last completed operation. • When STATUS is '10' after a read operation, it indicates that two or more errors were detected during the last completed operation. • When STATUS is '10' after an erase-page/program operation, it indicates that the compare operation failed during the last completed operation. • When STATUS is '11' after a write operation, it indicates that the attempt to write to another page before programming the current page was made during the last completed operation.

Below are the Verilog and VHDL representations of an NVM macro instantiation. For simulation purposes, the NVM macro may reference an Actel Memory File used to preload the memory with user-defined data. This is achieved by overriding the MEMORYFILE parameter in the simulation model during the NVM instantiation. The following is the NVM macro instantiation in Verilog and VHDL with the MEMORYFILE parameter override.

Verilog NVM Macro Instance

```
NVM U_NVM (
  .CLK (CLK),
  .RESET (RESET),
  .ADDR (ADDR),
  .REN (REN),
  .WEN (WEN),
  .READNEXT (READNEXT),
  .ERASEPAGE (ERASEPAGE),
  .PROGRAM (PROGRAM),
  .DATAWIDTH (DATAWIDTH),
  .RD (RD),
  .WD (WD),
  .BUSY (BUSY),
  .STATUS (STATUS),
  .SPAREPAGE (SPAREPAGE),
  .AUXBLOCK (AUXBLOCK),
  .UNPROTECTPAGE (UNPROTECTPAGE),
  .DISCARDPAGE (DISCARDPAGE),
  .OVERWRITEPROTECT (OVERWRITEPROTECT),
  .PAGELOSSPROTECT (PAGELOSSPROTECT),
  .PAGESTATUS (PAGESTATUS),
  .OVERWRITEPAGE (OVERWRITEPAGE),
  .PIPE (PIPE),
  .LOCKREQUEST (LOCKREQUEST)
);
```

Verilog NVM Macro Instance with Memory File

```
NVM #( .MEMORYFILE("<FLASH_INIT_FILE_NAME>.mem") U_NVM (
  ...
);
```

VHDL NVM Macro Instance with Memory File

```
architecture DEF_ARCH of <FLASH_NAME> is

component NVM

  generic (MEMORYFILE:string := "");

  port(
    ...
  );
end component;

begin

NVM_INST : NVM
  generic map(MEMORYFILE => "<FLASH_INIT_FILE_NAME>.mem")

  port map(
    ...
  );

end DEF_ARCH;
```

The memory array declared in the simulation model stores data that is one block wide. It is 64k×140 bits. The addressing scheme for accessing this array consists of 16 bits, as shown in Figure 5-20.

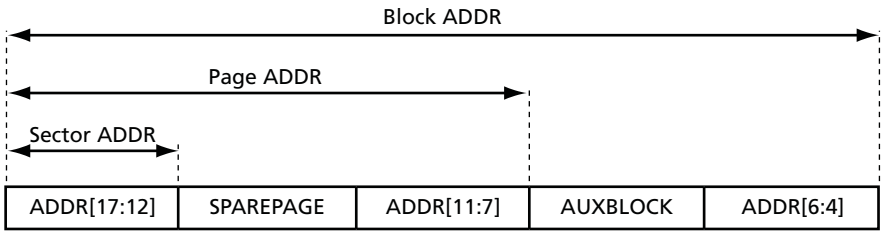


Figure 5-20 • **Addressing Scheme for Accessing the Flash Memory Array**

ADDR[17:0] is the embedded flash memory interface address; SPAREPAGE and AUXBLOCK are input signals. The memory file for preloading the Flash Array consists of strings of address and data in hexadecimal notation with address delimiters ('@'), and must conform to the rules given below:

1. Each line must contain a string of fixed length (35 characters) and start with an '@' if it corresponds to an address.
2. Each line following the address line corresponds to a block of data starting at the block address specified in the address line. This applies until the next line with an address specifier ('@') is encountered.
3. Each data block consists of 35 hex characters. Hex[31:0] are the data characters corresponding to 16 bytes of user data, where Hex[1:0] corresponds to Byte0 and Hex[31:30] corresponds to Byte15. Hex[34:32] are ECC-related bits and must be addressed manually.

Based on these rules, the format looks like the following:

```
@Block_Address_0
Block_Data_0 ( required )
Block_Data_1 ( optional )
Block_Data_2 ( optional )
...
...
Block_Data_8 ( Aux block data for this page, optional )
@Block_Address_n
Block_Data_n ( required )
Block_Data_n+1 ( optional )
Block_Data_n+2 ( optional )
...
...
...
Block_Data_n+8 ( Aux block data for this page, optional )
```

A typical memory file looks like the following:

```
@000...0000     // Beginning with @, start address in hex format. 0s to be padded
                // between @ and hex address to get a string of length 35.
ab101fd01...    // 35 hexadecimal characters corresponding to each block of flash memory
                // block cell
eab9c4......
@0004030....    // Start address for next data stream
c805489e....    // 35 hexadecimal characters corresponding to each block of flash memory
                // block cell
96986391
```

Data Storage Client Interface

The Flash Memory System Builder's Data Storage Client in Libero IDE can be used to configure the embedded flash memory to be used as general-purpose data storage. The Data Storage Client allows the user to create a flash memory partition, configure the address and data busses, and select an initialization file containing the flash array's initial values. JTAG read and/or write protections can also be added to prevent external access to the embedded flash memory array contents. Figure 5-21 shows the Data Storage Configuration window in Libero IDE.

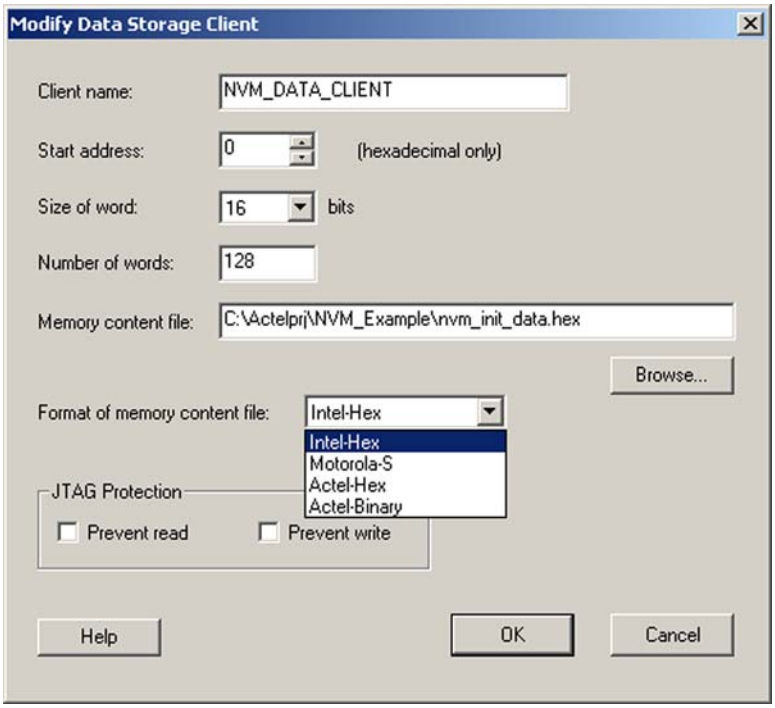


Figure 5-21 • **Data Storage Client Configuration Window**

The Data Storage Client spans a minimum of one page (128 bytes) and can go up to 2,048 pages, depending on the number of free pages available. The starting address for the Data Storage Client must be set to a value along the page boundaries—e.g., 0x00, 0x80, 0x100, etc. The word size for the read and write data busses can be either 8, 16, or 32 bits. The total number of words can be anywhere from one to 262,144 for 8-bit words, one to 131,072 for 16-bit words, and one to 65,536 for 32-bit words.

Once the Data Storage Client configuration is complete and the Flash Memory System Builder's IP is generated, the user can then instantiate the embedded flash memory system in the design. The overall ports list for the embedded flash memory system module may vary depending on the total number of clients added to the Flash Memory System. Refer to the "Using the Embedded Flash Memory for Initialization" section on page 5-1 and the "Common Flash Interface Data Client" section on page 5-31 for additional details regarding the other clients available in the Flash Memory System Builder. Table 5-6 on page 5-28 describes the Data Storage Client–specific ports.

Table 5-6 • **Data Storage Client–Specific Port Descriptions**

Flash Port	Direction	Description
USER_RESET	Input	Asynchronous active-low reset input signal. Holds the Flash Block's control logic in an initial state until released.
USER_CLK	Input	Flash memory input clock whose maximum clock period is dictated by $t_{MPWCLKNVM}$. All memory operations and status are synchronous to the rising edge of this clock.
USER_ADD[17:0]	Input	The 18-bit byte-based Flash Block input address bus. USER_ADD must transition synchronously with the rising edge of USER_CLK. For a data width of 16 bits, USER_ADD[0] is ignored and USER_ADD[1] becomes the lowest-order address; for a data width of 32 bits, USER_ADD[1:0] is ignored and USER_ADD[2] becomes the lowest-order address.
USER_WRITE	Input	Active-high write enable control input signal used for writing data into the flash memory Page Buffer. USER_WRITE must transition synchronously with the rising edge of USER_CLK.
USER_PROGRAM	Input	Active-high program operation control input signal used for writing the contents of the Page Buffer into the flash memory array page addressed. USER_PROGRAM must transition synchronously with the rising edge of USER_CLK.
USER_READ	Input	Active-high read enable control input signal used for read data from the flash memory array, Page Buffer, block buffer, or status registers. USER_READ must transition synchronously with the rising edge of USER_CLK.
USER_READ_NEXT	Input	Active-high read-next operation control input signal. This feature loads the next block relative to that stored in the block buffer from the flash memory array while reads from the block buffer are being performed. USER_READ_NEXT must be asserted along with USER_READ to initiate the read-next operation, and must transition synchronously with the rising edge of USER_CLK.
USER_DOUT[7:0], [15:0], or [31:0]	Output	The 8-, 16-, or 32-bit output read data bus. The data on USER_DOUT transitions synchronously with the rising edge of USER_CLK. After USER_READ has been asserted issuing a data read operation, all data must be sampled from USER_DOUT when USER_NVM_BUSY is not asserted (when LOW). Data put out on USER_DOUT are LSB-oriented. Upon a RESET, RD is initialized to zero. Upon a USER_RESET, RD is initialized to zero.
USER_DATA[7:0], [15:0], or [31:0]	Input	The 8-, 16-, or 32-bit input write data bus. Data put in on USER_DATA must be LSB-oriented; any unused pins must be grounded or driven LOW. The data on USER_DATA must transition synchronously with the rising edge of USER_CLK.

Table 5-6 • **Data Storage Client–Specific Port Descriptions (continued)**

Flash Port	Direction	Description
USER_WIDTH or USER_WIDTH[1:0]	Input	The 1- or 2-bit USER_DOUT and USER_DATA data bus width control input signal. DATAWIDTH must transition synchronously with the rising edge of USER_CLK. • If USER_WIDTH is '00', the data busses contain one byte of data forming an 8-bit word (USER_DOUT/USER_DATA[7:0]). • If USER_WIDTH is '01', the data busses contain two bytes of data forming a 16-bit word (USER_DOUT/USER_DATA[15:0]). • If USER_WIDTH is '1x', the data busses contain four bytes of data forming a 32-bit word (USER_DOUT/USER_DATA[31:0]). For a 1-bit USER_WIDTH: If USER_WIDTH is 1, the data busses are 16 bits wide; otherwise, they are 8 bits wide.
USER_PAGE_STATUS	Input	Active-high page status control input signal. When asserted, a page status read operation is initiated. USER_PAGE_STATUS must transition synchronously with the rising edge of USER_CLK and must be asserted along with USER_READ.
USER_ERASE_PAGE	Input	Active-high page erase control input signal asserted when the addressed page is to be programmed with all zeros. USER_ERASE_PAGE must transition synchronously with the rising edge of USER_CLK.
USER_DISCARD_PAGE	Input	Active-high discard page control input signal asserted when the contents of the Page Buffer are to be discarded so a new page write can be started. USER_DISCARD_PAGE must transition synchronously with the rising edge of USER_CLK.
USER_OVERWRITE_PAGE	Input	Active-high overwrite page control input signal asserted when the page addressed is to be overwritten, if writable, with the contents of the Page Buffer. USER_OVERWRITE_PAGE must transition synchronously with the rising edge of USER_CLK and must be asserted along with USER_PROGRAM.
USER_AUX_BLOCK	Input	Active-high auxiliary block select input signal asserted when the page address is used to access the auxiliary block within the page addressed. USER_AUX_BLOCK must transition synchronously with the rising edge of USER_CLK.
USER_SPARE_PAGE	Input	Active-high spare page select input signal asserted when the sector addressed is used to access the spare page within that sector. USER_SPARE_PAGE must transition synchronously with the rising edge of USER_CLK.
USER_UNPROT_PAGE	Input	Active-high unprotect page control input signal used to clear the protection of the addressed page. USER_UNPROT_PAGE must transition synchronously with the rising edge of USER_CLK.
USER_OVERWRITE_PROT	Input	Active-high overwrite protect control input signal used to change the protection bit of the addressed page. USER_OVERWRITE_PROT must transition synchronously with the rising edge of USER_CLK and must be asserted along with USER_PROGRAM or USER_ERASE_PAGE.

Table 5-6 • **Data Storage Client–Specific Port Descriptions (continued)**

Flash Port	Direction	Description
USER_PAGELOSS_PROT	Input	Active-high page-loss protect control input signal used to prevent writes to any other page than the current addressed page in the Page Buffer, until the page is either discarded or programmed. USER_PAGELOSS_PROT must transition synchronously with the rising edge of USER_CLK and must be asserted along with USER_PROGRAM or USER_ERASE_PAGE.
USER_LOCK	Input	Active-high lock request control input signal asserted when user access (including JTAG) to the flash memory array is to be prevented. USER_LOCK must transition synchronously with the rising edge of USER_CLK.
USER_NVM_BUSY	Output	Active-high busy control output signal used to indicate when the flash memory is performing an operation. USER_NVM_BUSY transitions synchronously with the rising edge of USER_CLK. Upon a USER_RESET, USER_NVM_BUSY pulses HIGH for several cycles before settling to a LOW state.
USER_NVM_STATUS[1:0]	Output	The 2-bit flash memory operation status output signals used to indicate the status of the last completed operation. USER_NVM_STATUS transitions synchronously with the rising edge of USER_CLK. Upon a USER_RESET, USER_NVM_STATUS is initialized to 0x00. • When USER_NVM_STATUS is '00', it indicates that the last operation completed successfully. • When USER_NVM_STATUS is '01' after a read operation, it indicates that a single error was detected during the last completed operation and was corrected. • When USER_NVM_STATUS is '01' after a write operation, it indicates that the last completed operation addressed a write-protected page. • When USER_NVM_STATUS is '01' after an erase-page/program operation, it indicates that the Page Buffer was unmodified during the last completed operation. • When USER_NVM_STATUS is '10' after a read operation, it indicates that two or more errors were detected during the last completed operation. • When USER_NVM_STATUS is '10' after an erase-page/program operation, it indicates that the compare operation failed during the last completed operation. • When USER_NVM_STATUS is '11' after a write operation, it indicates that an attempt to write to another page before programming the current page was made during the last completed operation.

Common Flash Interface Data Client

The CoreCFI IP is available to designers through Actel's CoreConsole IDP. CoreCFI provides an industry-standard interface to the embedded flash memory blocks within the Fusion family of FPGAs. The CoreCFI IP module is targeted to provide a functional subset of the Common Flash Interface standards with a design emphasis given to minimizing design size.

Using CoreConsole IDP to generate the CoreCFI IP HDL code, users must perform the following steps:

1. Open CoreConsole IDP through Libero IDE and add CoreCFI to the CoreConsole project.
2. Configure the CoreCFI IP. Figure 5-22 shows the CoreCFI IP Configuration GUI window in CoreConsole. The **FPGA Family** selected must be Fusion. The **Size (address width)** can be 6 to 18 bits. The number of address bits selected indicates the amount of the embedded flash memory accessible through CoreCFI—e.g., 10 bits = 1 kB, 12 bits = 4 kB, 18 bits = 256 kB.
3. Using the **Auto-Stitch to Top Level** feature of CoreConsole, bring all ports to the top on the block so that CoreCFI can be interfaced to the embedded flash memory module and other controlling circuits.

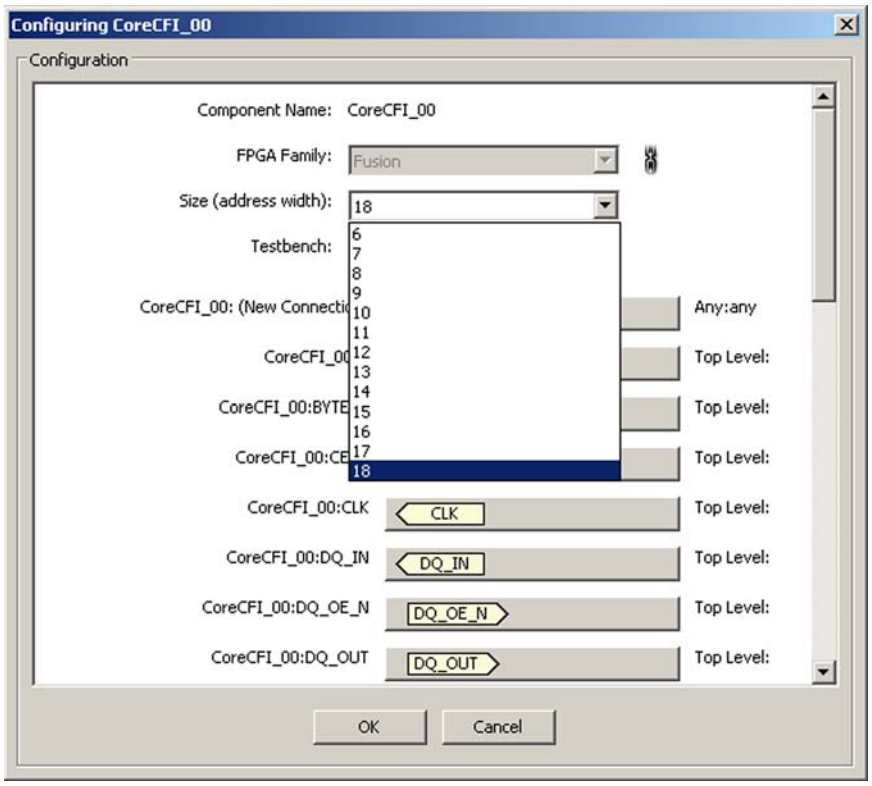


Figure 5-22 • **CoreCFI IP Configuration Window**

Once the CoreCFI IP is configured, use the **Save and Generate** function to deploy the HDL code and other related files, including the query data memory file. The query data memory file contains the Actel-specific 16-bit Command Set and Control Interface ID code as well as the embedded flash memory parameters and interface configuration data. Refer to the *CoreConsole User's Guide* and *CoreCFI Handbook* for additional configuration details.

The Flash Memory System Builder's CFI Data Client in Libero IDE is used to store the query data for the CoreCFI IP into the reserved (spare) pages of the embedded flash memory. This client does not take up any of the 2,048 pages available to designers for initialization or general data storage use.

Figure 5-23 on page 5-32 shows the CFI Data Client Configuration window in Libero IDE. Using the dialog box, the location of the query data memory file generated by CoreConsole during CoreCFI IP

deployment must be specified. The memory file format is Actel Binary and should not be modified by the user. CoreConsole deploys the query data memory file to the following directory location:

C:\Actelprj\<Libero_Project>\coreconsole\common\CORECFI\rtl\vlog\test\user\corecfi_query.mem

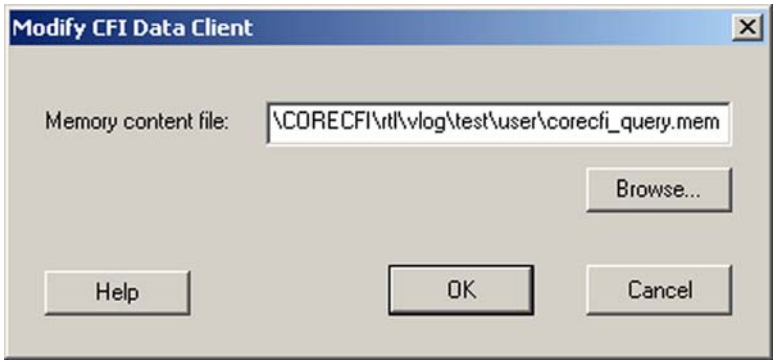


Figure 5-23 • **CFI Data Client Configuration Window**

During CoreCFI IP deployment, CoreConsole generates a sample HDL top file, named *corecfi_chip*, that instantiates both CoreCFI and the embedded flash memory system. Figure 5-24 illustrates an example CoreCFI IP system, complete with all interconnects between the main system blocks. The interface between CoreCFI and the embedded flash memory is through the Data Storage Client interface ports, as listed in Table 5-6 on page 5-28. Table 5-7 on page 5-33 describes the CoreCFI-specific ports.

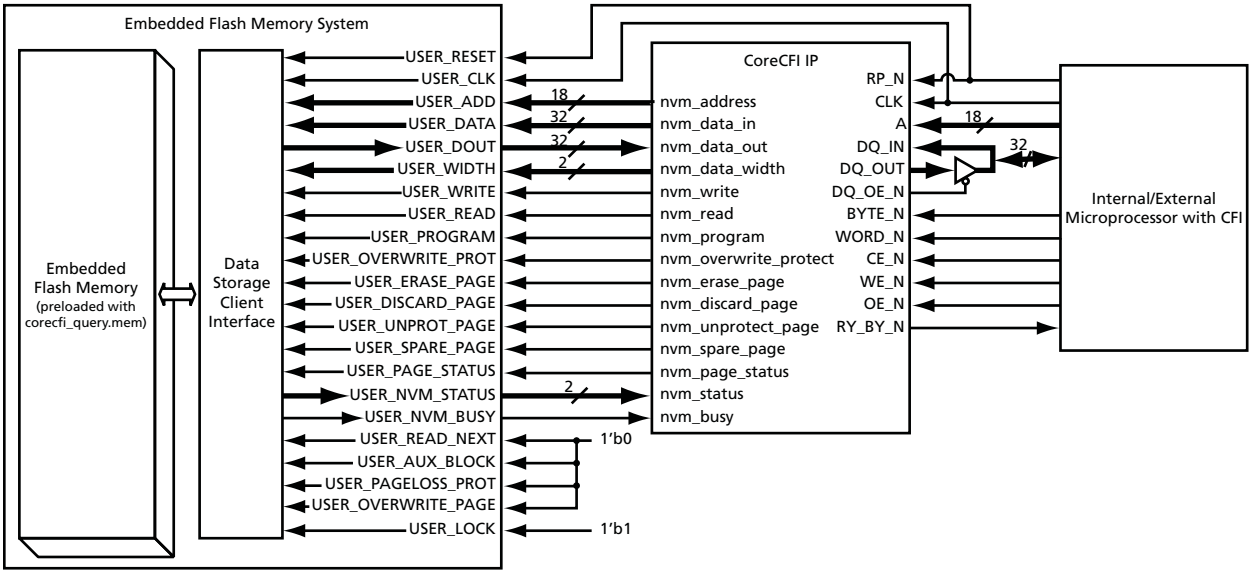


Figure 5-24 • **CoreCFI IP System Diagram**

Table 5-7 • **CoreCFI-Specific Port Descriptions**

CoreCFI Port	Direction	Description
RP_N	Input	Asynchronous active-low reset input signal that holds CoreCFI in an initial state until released. It is recommended that RP_N not be asserted while RY_BY_N is asserted; otherwise, the embedded flash memory may be damaged. When RP_N is asserted, CoreCFI is placed in Read Array mode, the status register is set to 0x80 (ready), and the data bus ports are tristated.
CLK	Input	Flash memory input clock whose maximum clock period is dictated by $t_{MPWCLKNVM}$. All operations and status are synchronous to the rising edge of this clock.
CE_N	Input	Active-low chip enable control input signal. When asserted, CoreCFI is enabled. CE_N must transition synchronously with the rising edge of CLK and must be asserted along with WE_N.
WE_N	Input	Active-low write enable control input signal used to initiate a write operation. WE_N must transition synchronously with the rising edge of CLK and must be asserted along with CE_N. If CE_N and WE_N are not both asserted, the write command will be ignored. A write command takes one clock cycle to execute; both CE_N and WE_N must be deasserted upon completion.
OE_N	Input	Active-low output enable control input signal used to control the direction of the CFI bidirectional data bus, and asserted during a read command. However, the bidirectional CFI data bus as listed in the CFI specification is split into an input data bus (DQ_IN) and an output data bus (DQ_OUT) in CoreCFI. The bidirectional CFI data bus must be formed outside of CoreCFI, where the direction control signal is an output of CoreCFI (DQ_OE_N), as shown in Figure 5-24 on page 5-32, and is a function of OE_N. OE_N must transition synchronously with the rising edge of CLK and must be asserted along with CE_N. If CE_N and OE_N are not both asserted, the read command will be ignored.
A[17:0]	Input	Active-high 18-bit input address bus used to address a location in the Flash Array during a command execution. For a data width of 16 bits, A[0] is ignored and A[1] becomes the lowest-order address; for a data width of 32 bits, A[1:0] is ignored and A[2] becomes the lowest-order address.
WORD_N	Input	The 2-bit {WORD_N, BYTE_N} data bus width control input signal. Both WORD_N and BYTE_N must transition synchronously with the rising edge of CLK and must be asserted together. • If 'x0', the data bus contains one byte of data forming an 8-bit word (DQ_IN/OUT [7:0]). • If '01', the data bus contains two bytes of data forming a 16-bit word (DQ_IN/OUT [15:0]). • If '11', the data bus contains four bytes of data forming a 32-bit word (DQ_IN/OUT [31:0]).
BYTE_N	Input	
DQ_OE_N	Output	Active-low output enable control output signal used to control the direction of the CFI bidirectional data bus formed external to the CoreCFI IP, as shown in Figure 5-24 on page 5-32. DQ_OE_N transitions synchronously with the rising edge of CLK and is asserted when both CE_N and OE_N are asserted. Upon an RP_N assertion, DQ_OE_N is initialized to zero.

Table 5-7 • **CoreCFI-Specific Port Descriptions (continued)**

CoreCFI Port	Direction	Description
DQ_IN[31:0]	Input	The 32-bit input data bus used during a write command. The data on DQ_IN must transition synchronously with the rising edge of CLK. Data put in on DQ_IN must be LSB-oriented. Configuration depends on the state of {WORD_N, BYTE_N}: if in the 8-bit word mode, the DQ_IN[31:8] ports are ignored; if in the 16-bit word mode, the DQ_IN[31:16] ports are ignored. As shown in Figure 5-24 on page 5-32, DQ_IN should be connected to the CFI bidirectional bus.
DQ_OUT[31:0]	Output	The 32-bit output data bus used during a read command. The data on DQ_OUT transitions synchronously with the rising edge of CLK. Data put out on DQ_OUT is LSB-oriented. Configuration depends on the state of {WORD_N, BYTE_N}: if in the 8-bit word mode, the DQ_OUT[31:8] ports are ignored; if in the 16-bit word mode, the DQ_OUT[31:16] ports are ignored. As shown in Figure 5-24 on page 5-32, DQ_OUT should be connected to the CFI bidirectional bus. Upon an RP_N assertion, DQ_OUT is initialized to zero.

CoreCFI supports the Read Query, Read, Automatic Erase, Automatic Write, Lock, and Status CFI operations. The command descriptions are summarized in Table 5-8 and Table 5-9 on page 5-35. Refer to the *CoreCFI Handbook* (available on the Actel website or in CoreConsole) for the CFI command details.

Table 5-8 • **Supported CFI Command Descriptions**

Command	Description
Read Query	The Read Query command causes CoreCFI to load the query database from a spare page of the embedded flash memory. Query data is always supplied on the least significant 8 bits of DQ_OUT. The address of the query data starts at 10h in 32-bit, 20h in 16-bit, or 40h in 8-bit mode.
Read ID Codes	The Read ID Codes command causes CoreCFI to load either the manufacturer code, die size code, or page lock status from the embedded flash memory onto DQ_OUT. The identifier codes returned are either values stored in the query data spare page or the lock status of a page in the Flash Array.
Read Array	The Read Array command causes CoreCFI to be placed in read array mode, where the content of the addressed location of the Flash Array is loaded onto DQ_OUT. Upon a reset, CoreCFI is initialized to the read array mode state.
Read Status	The Read Status command causes CoreCFI to load the status register onto DQ_OUT. The status register provides the status of the last write, erase, or lock command execution.
Clear Status	The Clear Status command causes CoreCFI to clear registered status register bits.
Erase Page	The Erase Page command causes CoreCFI to erase the addressed page of the Flash Array. A page erase activity fills the contents of a page with zeros.
Single Write	The Single Write command causes CoreCFI to write the data placed on DQ_IN to the addressed location of the Flash Array. This command reads the entire addressed page, modifies the address location, and programs the page into the Flash Array. If more than one location is to be modified, the Multiple Write command should be used.
Multiple Write	The Multiple Write command causes CoreCFI to be placed in the page write mode, where the contents of DQ_IN are written to the Page Buffer of the embedded flash memory. If the write activity exceeds the page boundary, the data written will wrap to the top of the page. Once all values are written into the page, the page is written into the Flash Array.
Page Lock	The Page Lock command causes CoreCFI to lock the addressed page, preventing any erase or write commands to the page from executing.
Page Unlock	The Page Unlock command causes CoreCFI to unlock the addressed page, allowing all erase or write commands to the page to execute.

Table 5-9 • **CFI Command Algorithm Summary**

Command	No. of Bus Cycles	First Bus Cycle			Second Bus Cycle		
		Operation	Address	Data	Operation	Address	Data
Read Query	2	Write	X	0x98	Read	Query Address	Query Data
Read ID Codes	2	Write	X	0x90	Read	Identifier Address	Identifier Data
Read Array	1 or 2	Write	X	0xFF	Read	Array Address	Array Data
Read Status	2	Write	X	0x70	Read	X	Status Data
Clear Status	1	Write	X	0x50	–	–	–
Erase Page	2	Write	Page Address	0x20	Write	Page Address	D0h
Single-Write	2	Write	Page Address	0x40	Write	Array Address	Array Data
Multi-Write	2	Write	Page Address	0xE8	Write	Page Address	N = Num. of elements – 1
Page Lock	2	Write	X	0x60	Write	Page Address	0x01
Page Unlock	2	Write	X	0x60	Write	Page Address	0xD0

Flash Operation Priority

The embedded flash memory has a built-in priority for operations when multiple actions are requested simultaneously. Table 5-10 shows the operation priority order—priority 0 is the highest. Access to the embedded flash memory is controlled by the BUSY (USER_BUSY in the Data Storage Client interface) signal. The BUSY output is synchronous to the CLK (USER_CLK in the Data Storage Client interface) signal. The embedded flash memory operations are only accepted in cycles where BUSY is not asserted (LOW).

Table 5-10 • **Flash Memory Operation Priority**

Operation	Priority
System Initialization	0 (highest)
Flash Memory Reset	1
Read	2
Write	3
Erase Page	4
Program	5
Unprotect Page	6
Discard Page	7

The system initialization operation is the highest in the priority order. The system initialization occurs upon a system reset of a Fusion FPGA. All FPGA operations should be halted during the system initialization process. Refer to the "Using the Embedded Flash Memory for Initialization" section on page 5-1 for additional information.

If read and write operations are performed simultaneously, for example, the read operation takes precedence over the write operation. The write data supplied during this operation is ignored, and the data remains unchanged. Also, if an erase page and a write operation are performed

simultaneously, the write operation takes precedence over the erase page operation. The write data supplied during this operation executes. All other priority order situations behave similarly.

Flash Busy Signal Handling

The BUSY (USER_BUSY in the Data Storage Client interface) signal is one of the most important signals in the embedded flash memory; all operations on the embedded flash memory should be designed based on the BUSY signal. The BUSY signal is asserted HIGH whenever a flash operation is in progress, then deasserted after the operation is complete. To shorten simulation run time, the run time of the different operations (Write/Program/Erase/Read) is shortened in the simulation model. Therefore, the system design should be based on the BUSY signal assertion and deassertion status instead of counting the operation cycles for each operation. All flash memory inputs are ignored while BUSY is asserted. Inputs can be updated for the next operation at the rising edge of the clock with no hold time requirement.

During an embedded flash memory reset, the contents of the flash memory control logic block, such as the contents of the Block Buffer and Page Buffer, are cleared. Once RESET (USER_RESET in the Data Storage Client interface) is asserted LOW, the BUSY signal is asserted HIGH. After RESET is deasserted, the BUSY signal is deasserted approximately 25 µs later. Therefore, the system design should accommodate this BUSY period by monitoring the BUSY signal status. All operations can be executed only after the BUSY signal is deasserted.

For continuous operations, like the continuous reads for microprocessor instruction executions, the flash clock may be adjusted to compensate for the flash memory BUSY period—for example, during the loading of a new page from the Flash Array into the page buffer. Typically, the flash clock is sourced by the microprocessor system clock in an application. SmartTime, Actel's gate-level static timing analysis tool available in Actel's Designer software, will provide the maximum frequency for the system design. This maximum frequency will be adjusted to compensate for these BUSY periods.

Write Operations and Page Programming

The embedded flash memory offers a write operations class of commands. These commands include the page buffer write, discard page, program page, erase page, and overwrite page operations. All write commands are page-based operations. The embedded flash memory write operations modify the contents of both the Block and Page Buffers. As shown in Figure 5-25, the Block and Page Buffers are sub-blocks of the embedded flash memory and consist of volatile registers.

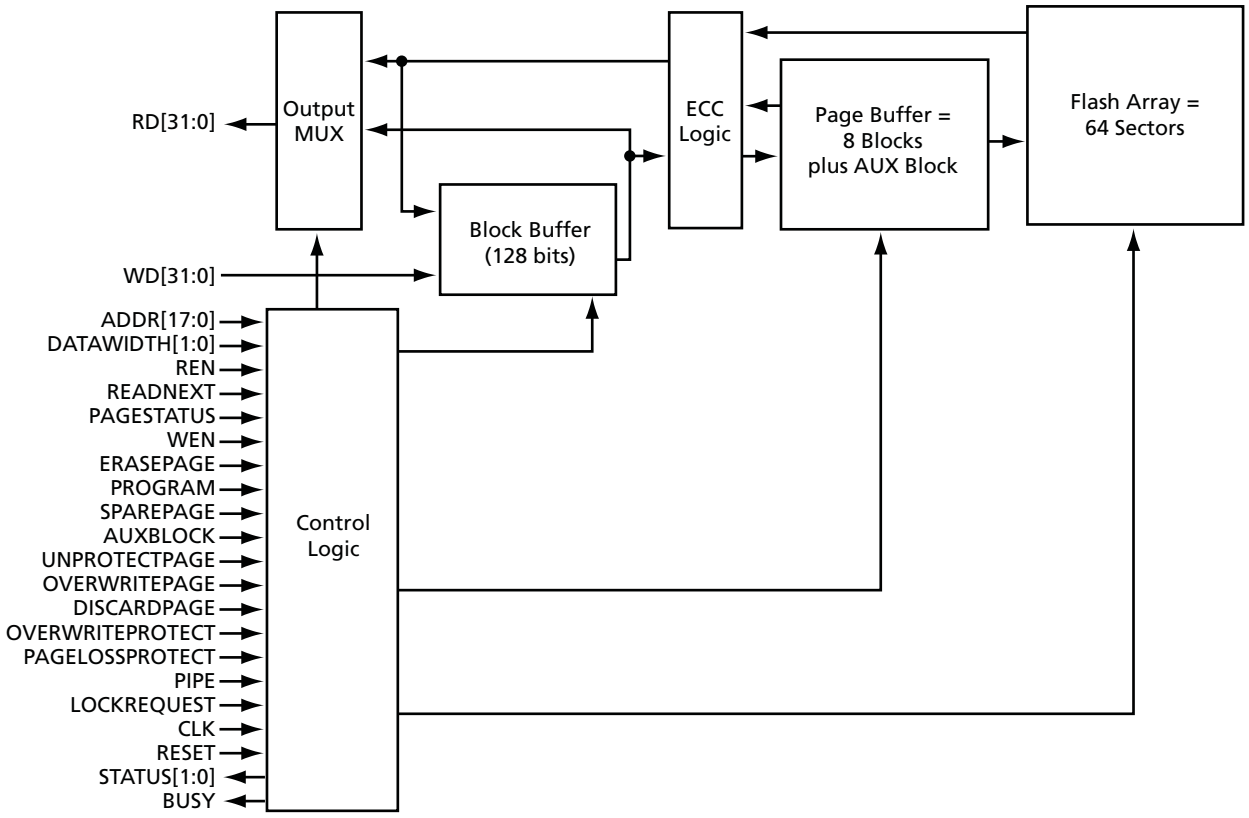


Figure 5-25 • **Flash Memory Block Diagram**

A write operation to a location in a page that is not already in the Page Buffer will cause the page to be read from the Flash Array and stored in the Page Buffer. The number of BUSY cycles required to complete the page buffer load is variable. The block that was addressed during the write operation will be loaded into the Block Buffer, and the data written by WD (USER_DATA in the Data Storage Client interface) will overwrite the data in the Block Buffer. After the data is written to the Block Buffer, the Block Buffer is then written to the Page Buffer to keep both buffers in sync. Subsequent writes to the same block will overwrite both the Block and Page Buffers without incurring BUSY cycles. A write operation to another block in the page will cause the addressed block to be loaded from the Page Buffer and into the Block Buffer, and the write will be performed as previously described. The Block Buffer load will incur four BUSY cycles (five cycles with PIPE asserted). The contents of the Page Buffer will be stored into the Flash Array only when a program page operation is executed. During the program page operation execution, the BUSY (USER_BUSY in the Data Storage Client interface) signal will be asserted HIGH for ~8 ms until the page programming completes. Figure 5-26 on page 5-38 is the timing diagram for a program page operation.

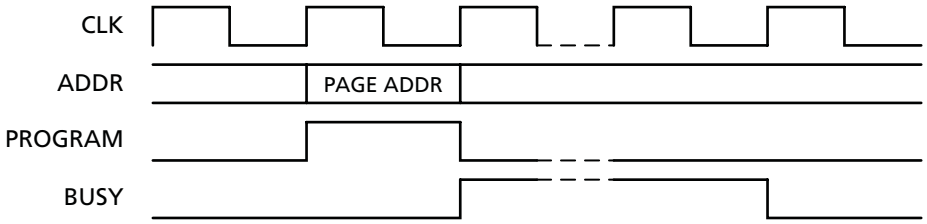


Figure 5-26 • **Program Page Operation Timing Diagram**

A page write operation is initiated by asserting the WEN (USER_WRITE in the Data Storage Client interface) signal HIGH. The page write operation automatically triggers a Block or Page Buffer load operation when a change in the block or page address is detected. During a Block Buffer load operation, the embedded flash memory logic loads the contents of the addressed block from the Page Buffer into the Block Buffer volatile registers. During a Page Buffer load operation, the embedded flash memory logic loads the contents of the addressed page from the Flash Array into the Page Buffer volatile registers. The BUSY signal is asserted HIGH during both loading processes. Figure 5-27 is the timing diagram for a page write operation. Any page being written using a page write or program page operation that is overwrite-protected will result in the STATUS signals being set to '01'; the page data stored in the Page Buffer or Flash Array are left unchanged. During a page write operation, the protected page is detected during both the page and block buffer loading processes.

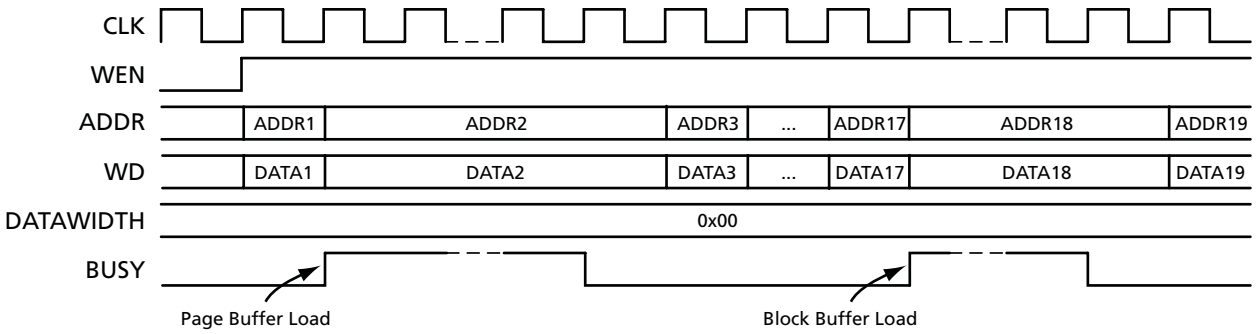


Figure 5-27 • **Page Write Operation Timing Diagram**

Writing to more than one page without executing a program page operation before changing the page address will result in the loss of the Page Buffer data. The page-loss protection option can be enabled to protect against the loss of the Page Buffer data by asserting the PAGELOSSPROTECT (USER_PAGELOSS_PROT in the Data Storage Client interface) signal HIGH during a program or erase page operation. Any page that is page-loss-protected will result in the STATUS (USER_NVM_STATUS in the Data Storage Client interface) signals being set to '11' when an attempt is made to write to a new page leaving the Page Buffer unchanged. The PAGELOSSPROTECT signal can be tied permanently HIGH; it is only sampled when the PROGRAM or ERASEPAGE signals are asserted HIGH. Actel recommends always enabling the page-loss protection option. Figure 5-28 on page 5-39 is the timing diagram showing the page-loss protection fault STATUS update.

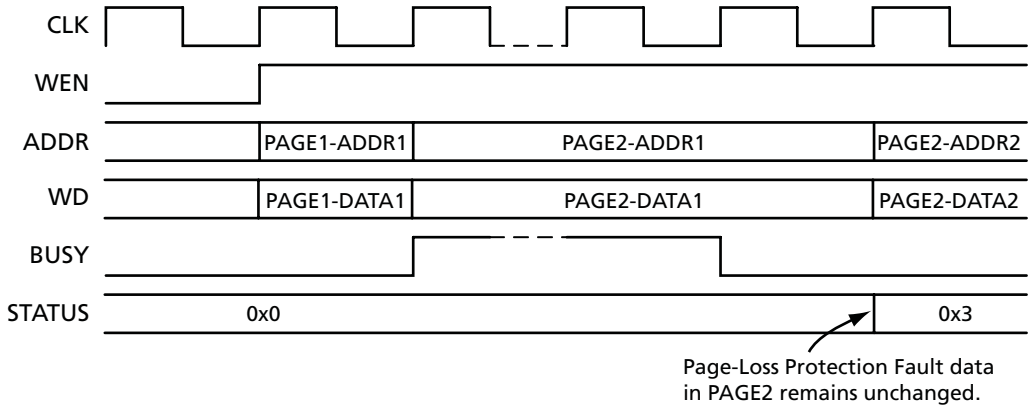


Figure 5-28 • **Page-Loss Protection Fault Timing Diagram**

To discard the contents of the modified Page Buffer, the discard page operation can be initiated by asserting the DISCARDPAGE (USER_DISCARD_PAGE in the Data Storage Client interface) signal HIGH for one clock cycle. This command will result in the Page Buffer being marked as unmodified. The BUSY signal will remain asserted until the discard page operation completes.

The erase page operation erases the addressed Flash Array page by filling the Page Buffer volatile registers with all zeros and issuing a program page operation. It is initiated by asserting the ERASEPAGE (USER_ERASE_PAGE in the Data Storage Client interface) signal HIGH while addressing the page to be erased. During the erase page operation execution, the BUSY signal will be asserted HIGH until the operation completes. Both the erase page and page write operations require fewer cycles when executing on the same page rather than a new page. Any page that is overwrite-protected will result in the STATUS signals being set to '01' when an attempt is made to erase the page, and the addressed page's data will be left unchanged.

The overwrite page operation can be used to overwrite any addressed page in the Flash Array with the contents of the Page Buffer. The operation can be initiated by asserting the OVERWRITEPAGE (USER_OVERWRITE_PAGE in the Data Storage Client interface) signal HIGH during a program page operation. Any page that is overwrite-protected will result in the STATUS signals being set to '01' when an attempt is made to program a page with OVERWRITEPAGE asserted, and the addressed page's data will be left unchanged. Figure 5-29 is the timing diagram showing the overwrite protection fault STATUS update.

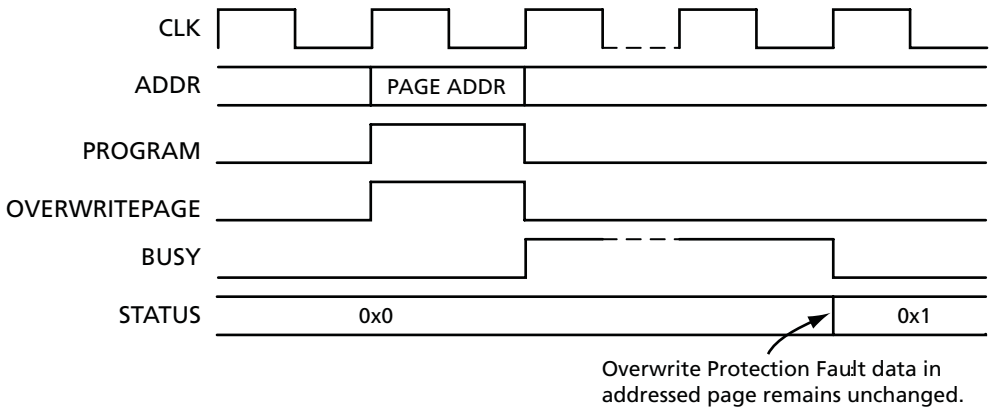


Figure 5-29 • **Overwrite Protection Fault Timing Diagram**

The overwrite protect mechanism is used to protect the contents of the selected Flash Array's pages from being overwritten. Asserting the OVERWRITEPROTECT (USER_OVERWRITE_PROT in the Data Storage Client interface) signal HIGH when a program page operation is undertaken will set the overwrite protection option for the addressed page. OVERWRITEPROTECT can be held HIGH if multiple pages are to be overwritten; it is only sampled when the PROGRAM or ERASEPAGE signals are asserted HIGH. OVERWRITEPROTECT is ignored in all other operations. Any page that is overwrite-protected will result in the STATUS signals being set to '01' when an attempt is made to either write, program, or erase the protected page, as shown in Figure 5-30.

To clear the overwrite protect option for a given page, the unprotect page operation must be performed, and the page must be programmed with the OVERWRITEPROTECT pin cleared to save the new protection settings. An unprotect page operation is initiated by asserting the UNPROTECTPAGE (USER_UNPROT_PAGE in the Data Storage Client interface) signal HIGH while addressing the page. If the addressed page is not in the Page Buffer, the unprotect page operation will trigger a Page Buffer load operation. The load operation occurs only if the current page in the Page Buffer was programmed into the Flash Array or is not page-loss-protected. During the unprotect page operation execution, the BUSY signal will be asserted HIGH until the operation completes. If either the OVERWRITEPROTECT or UNPROTECTPAGE signal is asserted, the other must be deasserted. The unprotect page operation may result in the STATUS signals being set to '01' when the page has a single-bit correctable error, '10' for a double-bit uncorrectable error, or '11' when the Page Buffer has encountered a page-loss protection situation, during the operation execution. Figure 5-30 is the timing diagram showing the page-loss protection fault STATUS update during an unprotect page operation.

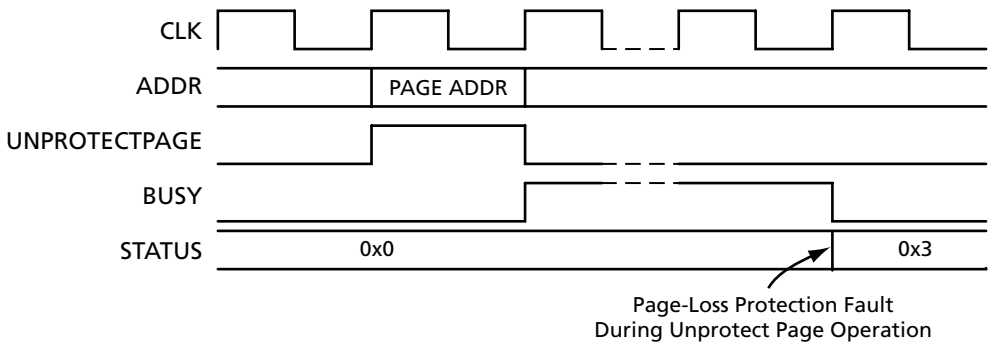


Figure 5-30 • **Unprotect Page Operation Page-Loss Protection Fault Timing Diagram**

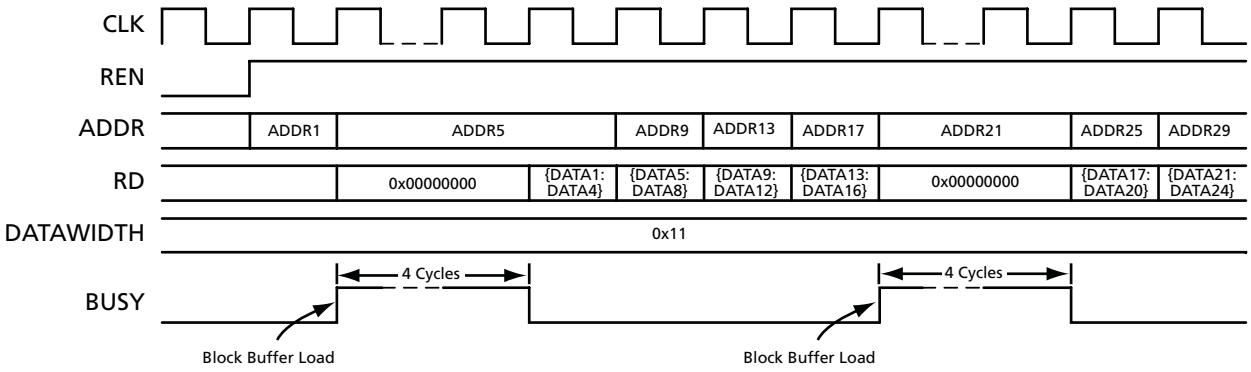
Read Operations

Read operations are designed to read data from the Flash Array and page status registers. The read operations support read operations with and without read-next or pipeline stage enabled. All read commands are page-based operations. The embedded flash memory read operation reads the contents of the Block Buffers, which are loaded from either the Page Buffer or the Flash Array. The Block and Page Buffers are sub-blocks of the embedded flash memory, consisting of volatile registers. Refer to Figure 5-25 on page 5-37 for the flash memory block diagram.

A read operation to a location in a page that is not already stored in the Page Buffer will cause the data from the Flash Array to be read and stored directly into the Block Buffer. The BUSY (USER_BUSY in the Data Storage Client interface) signal is asserted HIGH during the Block Buffer loading process for approximately four or five clock cycles. Any subsequent blocks addressed within the same page will be filled with data from the Flash Array with the same BUSY period consequence. However, a read operation to a location already stored in the Page Buffer is loaded into the Block Buffer without a BUSY period.

A read operation is initiated by asserting the REN (USER_READ in the Data Storage Client interface) signal HIGH. If the Block Buffer load is from the Flash Array, the BUSY signal is asserted HIGH for approximately four or five clock cycles during the Block Buffer loading process. The contents of the block that was addressed during the read operation will be placed on the RD (USER_DOUT in the

Data Storage Client interface) output data bus. For frequencies greater that 50 MHz, a pipeline stage before the data read is placed on the RD data bus may be added by asserting the PIPE signal HIGH along with REN. If the pipeline stage is enabled, the BUSY signal is asserted for five clock cycles during each Block Buffer load process; otherwise, it is asserted for four cycles. Figure 5-31 is the timing diagram for a page read operation.



Note: With 16- and 32-bit RD data bus widths, bytes are placed on the bus in big-endian byte order.

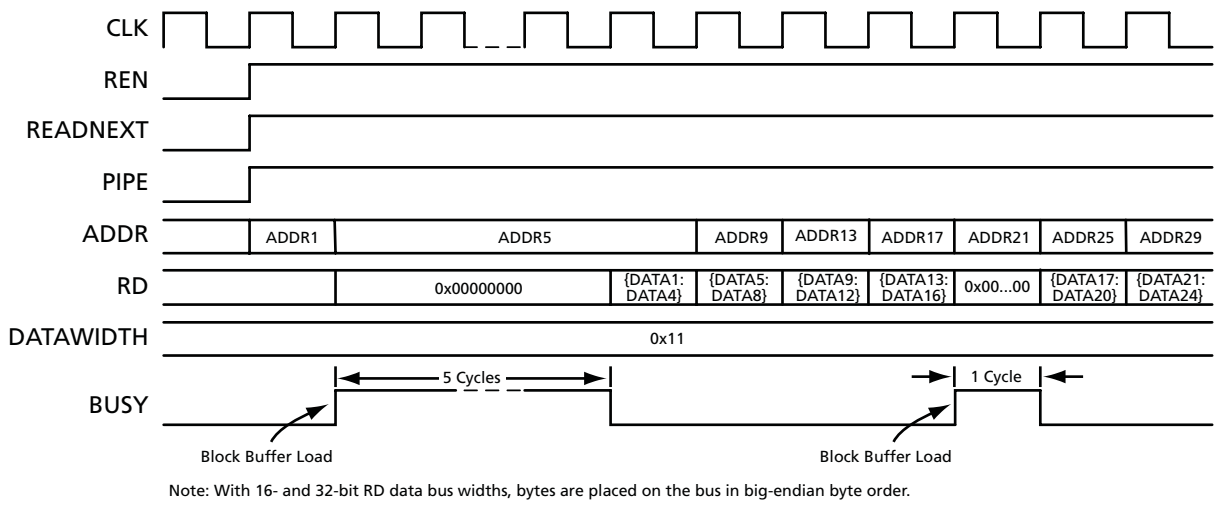
Figure 5-31 • **Page Read Operation Timing Diagram**

The read-next operation is a feature by which the next block to the current block in the Block Buffer is read from the Flash Array while performing reads from the Block Buffer, to minimize BUSY wait states during a sequential read operation. It is enabled by asserting the READNEXT (USER_READ_NEXT in the Data Storage Client interface) signal HIGH along with REN. Since the read-next operation executes look-ahead reads, it is performed in a predetermined manner, as follows:

- When reading within a page, the next block fetched will be the next in linear address.
- When reading the last data block of a page, it will fetch the first block of the next page.
- When reading spare pages, it will read the first block of the next sector's spare page.
- When reading the last sector, it will wrap around to sector 0.
- When reading the auxiliary blocks, it will read the next linear page's auxiliary block.

When an address on the ADDR (USER_ADD in the Data Storage Client interface) bus does not agree with the predetermined look-ahead address, there is a time penalty for this access. The embedded flash memory must complete the current look-ahead read before starting the next. The worst-case

BUSY period is a total of nine cycles before data is delivered. Figure 5-32 is the timing diagram for a pipeline-staged read-next page read operation.



Note: With 16- and 32-bit RD data bus widths, bytes are placed on the bus in big-endian byte order.

Figure 5-32 • **Pipeline-Staged Read-Next Page Read Operation Timing Diagram**

The status registers of each page of the embedded flash memory can be read by asserting the PAGESTATUS (USER_PAGE_STATUS in the Data Storage Client interface) signal along with REN. The contents of the addressed page's status register will be driven onto the RD data bus. The format of the data returned by a page status read is shown in Table 5-11. Figure 5-33 on page 5-43 is the timing diagram for a page status register read operation.

Table 5-11 • **Page Status Register Bit Definitions**

Bit(s)	Name	Description
[31:8]	Write Count	The number of times the page addressed has been programmed or erased
[7:4]	Reserved	Reads as 0.
[3]	Over-Threshold	Over-threshold indicator. See the "Program Operation" section of the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet for details.
[2]	Read Protect	The read protect bit for the page set via the JTAG interface. If 1, the page is read-protected.
[1]	Write Protect	The write protect bit for the page set via the JTAG interface. If 1, the page is write-protected.
[0]	Overwrite Protect	The overwrite protect bit used to protect the page from being inadvertently overwritten. The bit must be set by asserting the OVERWRITEPROTECT signal during a program operation. The page cannot be overwritten without first performing an unprotect page operation. Refer to the "Write Operations and Page Programming" section on page 5-37 for additional information.

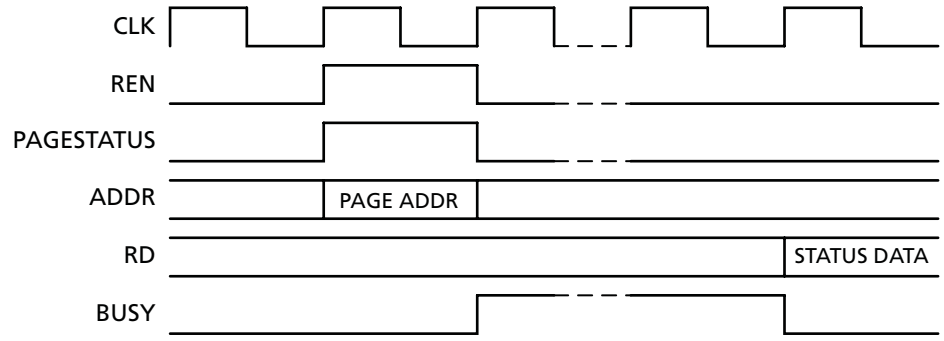


***Figure 5-33* • Page Status Register Read Operation**

Note on Updating the Contents of Flash

The possibility of data corruption due to a programming interruption is common to flash technology, and precautions must be taken when updating the data contents of any flash memory, including Fusion's embedded flash memory block. An interruption may occur, for example, as a result of a loss of power to the Fusion FPGA or an unexpected reset operation of the Flash Block control logic. Therefore, it is recommended that the appropriate measure in the application be taken to prevent such interruptions from occurring by adding power-down ramp control circuitry, low voltage detection circuitry, etc., to allow for the 8 ms program and erase page operations to complete their execution.

If an interruption of the write or read operation occurs, the immediate data stored in the Block or Page Buffer is lost, but the Flash Array's data (a sub-block of Flash Block holding the nonvolatile data contents, as shown in Figure 5-25 on page 5-37) remains valid. However, if an interruption of a program or erase operation occurs, the page addressed during the interruption may be left in a locked state. Although it may be that no physical damage to the Flash Array will have occurred, data corruption of the page is likely to have occurred, resulting in the possible corruption of the page's auxiliary block control data. Only the page addressed at the time of the interruption may incur data corruption; no other page should be affected.

A Flash Block's page contains eight blocks of user data and one auxiliary block. The auxiliary block is mostly used for storing control data. The auxiliary block control bits of concern are the write protect, read protect, overwrite protect, and write count bits. If the protection bits are corrupted, in most cases, reprogramming the embedded flash memory contents with the programming (STAPL) file will restore the protection bits to their predefined states; however, the page's write count will be lost. If the overwrite protection bit is the only bit of concern, the application can set/clear this bit by using the OVERWRITEPROTECT or UNPROTECTPAGE input signals to the NVM macro, as described in the "Write Operations and Page Programming" section on page 5-37.

Microprocessor/Microcontroller Interface

The embedded flash memory can be interfaced to a microprocessor or microcontroller for use as its nonvolatile instruction execution memory space and data storage memory space. To enable the embedded flash memory to communicate with the microprocessor's or microcontroller's bus architecture, an interface bridge must be developed and added to the design. Actel offers, as a DirectCore IP through CoreConsole IDP, the CoreAhbNvm IP, which contains a hardware bridge between the embedded flash memory block and the industry-standard Advanced Microcontroller Bus Architecture's high-performance system backbone bus (AHB). Microprocessors can then access data stored in the flash memory via the software-driven CFI command protocol. The following sections describe CoreAhbNvm's design configuration and CFI commands.

CoreAhbNvm IP Configuration

CoreConsole IDP is used to develop a microcontroller design for Actel FPGAs, utilizing the ARM7,™ Cortex-M1, and CoreABC microprocessor or microcontroller IP offerings, together with the various AMBA-AHB and AMBA-APB (Advanced Peripheral Bus) IP peripherals. For Fusion FPGA designs, CoreAhbNvm can be paired with either the ARM7 (CoreMP7 IP) or Cortex-M1 microprocessors. The microprocessors must communicate with CoreAhbNvm through an AHB master, also available through CoreConsole as CoreAHB and CoreAHBLite.

When using CoreAhbNvm's embedded flash memory as the microprocessor's instruction execution memory, CoreAhbNvm must be connected as CoreAHB or CoreAHBLite's slave-0 (AHBmslave0) port. After system reset, CoreAhbNvm defaults to the CFI's read-array mode for the continuous reading of the instruction code stored in memory. Since data is read from the embedded flash memory's page buffer, when changing the instruction address to a new page in flash, the microprocessor must allow for the time required to load a new flash memory page into its page buffer before capturing the data. Therefore, the microprocessor and AHB-master system clock must be reduced appropriately, such that continuous reads can be performed without needing to pause for page buffer loading. When performing static timing analysis using SmartTime, the maximum operating frequency of the microprocessor's system clock is typically reduced to, for example, 15 MHz for the ARM7.

When using CoreAhbNvm's embedded flash memory as the microprocessor's nonvolatile data storage, CoreAhbNvm must be connected to any slave port other than slave-0 (AHBmslave1–AHBmslave15), reserving slave-0 for the instruction execution memory. Note that the Remap input to CoreAHB and CoreAHBLite is used to swap between the slave-0 (AHBmslave0) and slave-1 (AHBmslave1) ports. The Remap feature of the AHB architecture is typically used to swap boot memory spaces (from flash to RAM and vice versa). Therefore, be sure to plan the AHB system carefully taking all AHB slave configurations into consideration. For additional details, refer to the *CoreAHB*, *CoreAHBLite*, and *CoreRemap* datasheets.

Once the entire microcontroller design is complete, CoreConsole's "Save & Generate" operation will produce the RTL code and testbench for the design and save it in a project directory, which then is imported into Libero IDE. Libero IDE can then be used to complete the Fusion FPGA design flow. Refer to the *CoreConsole User's Guide* and the *Libero IDE User's Guide* for additional usage information. Figure 5-34 on page 5-45 is an example of a simple Fusion CoreMP7 microcontroller CoreConsole project, with CoreAhbNvm used as its instruction execution memory.

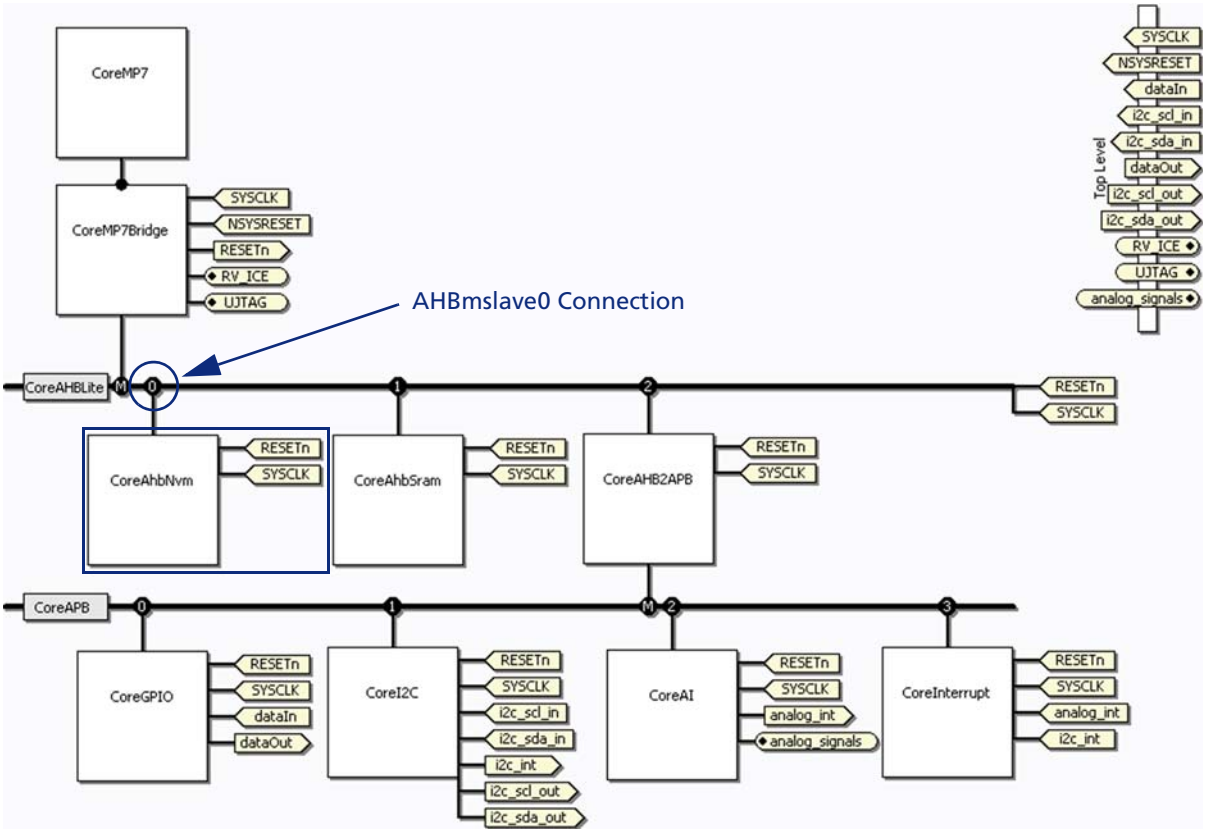


Figure 5-34 • **Simple Fusion CoreConsole Project with CoreAhbNvm Used as the Instruction Execution Memory**

CoreConsole allows designers to easily configure the IP though a simple GUI. Figure 5-35 shows the CoreAhbNvm IP CoreConsole configuration GUI window.

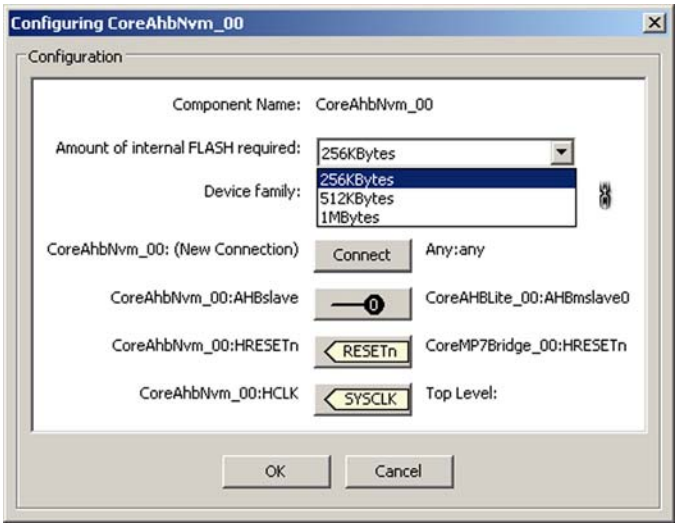


Figure 5-35 • **CoreAhbNvm CoreConsole Configuration GUI**

CoreAhbNvm can be configured to include the embedded flash as a 256-kbyte, 512-kbyte, or 1-Mbyte flash memory data space. By default, CoreAhbNvm is configured to include a 256-kbyte flash memory. Up to four CoreAhbNvm blocks can be instantiated for a single Fusion design, depending on the targeted Fusion FPGA device. Be sure to size the flash memory for each CoreAhbNvm block appropriately, such that the total number of Flash Block instances does not exceed the total number available on the targeted Fusion FPGA device. Table 5-12 lists the required number of Flash Block instances for each flash memory space size configuration option. Cross-reference the required number of Flash Block instances with the number of blocks available in the targeted Fusion device, as found in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet.

Table 5-12 • **Flash Memory Space Size vs. Required Number of Memory Block Instances**

Flash Memory Size Option	Number of Flash Block Instances
256 kbytes	1
512 kbytes	2
1 Mbytes	4

The CoreAhbNvm port signals are described in Table 5-13. If CoreAhbNvm is paired with either the on-chip ARM7 or Cortex-M1, CoreConsole's Auto Stitch feature can be used to make all the required interconnects between CoreAhbNvm and CoreAHB/CoreAHBLite. CoreAhbNvm, however, can also be used as an extension of an external microprocessor's nonvolatile memory space by placing all its AHB interface signals at the top of the chip design. CoreAhbNvm can then be connected and controlled by an external AHB master.

Table 5-13 • **CoreAhbNvm Port Signal Descriptions**

Signal	Direction	Description
HCLK	Input Bus Clock	Rising-edge-active AHB clock, which times all bus transfers and all signal timings
HRESETn	Input Reset	Active-low asynchronous bus reset signal used to reset the system and the bus. This is the only active-low AHB signal.
HTRANS[1:0]	Input	Transfer control input signals that indicate the type of the current transfer: 00 – Idle 01 – Busy 10 – Non-Sequential 11 – Sequential HTRANS transitions synchronously with the rising edge of HCLK.
HADDR[19:0]	Input	The 32-bit AHB system address bus from the AHB master. HADDR transitions synchronously with the rising edge of HCLK.
HWRITE	Input	Transfer direction control signal. When HIGH, this signal indicates a write transfer; when LOW, a read transfer. HWRITE transitions synchronously with the rising edge of HCLK.
HSIZE[2:0]	Input	Indicates the size of the transfer, which can be byte (8-bit), halfword (16-bit), or word (32-bit). HSIZE transitions synchronously with the rising edge of HCLK.
HWDATA[31:0]	Input	32-bit write input data bus from the AHB master. HWDATA transitions synchronously with the rising edge of HCLK.
HREADYIN	Input	Active-high ready signal input from all other AHB slaves. HREADYIN transitions synchronously with the rising edge of HCLK.
HSEL	Input	Active-high slave select signal input, which is a combinatorial decode of HADDR. Indicates that this slave is currently being selected. HSEL transitions synchronously with the rising edge of HCLK.

Table 5-13 • **CoreAhbNvm Port Signal Descriptions (continued)**

Signal	Direction	Description
HRDATA[31:0]	Output	32-bit read output data bus written back to the AHB master. HRDATA transitions synchronously with the rising edge of HCLK. Upon an HRESETn reset, the HRDATA output is zero.
HREADY	Output	Transfer-done control output signal. When HIGH, the HREADY signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer. HREADY transitions synchronously with the rising edge of HCLK. Upon an HRESETn reset, the HREADY output is HIGH.
HRESP[1:0]	Output	Transfer response output signals, which have the following meanings: 00 – Okay 01 – Error 10 – Retry 11 – Split HRESP transitions synchronously with the rising edge of HCLK. Upon an HRESETn reset, the HRESP output is in an Okay state (0x00).

Supported CFI Commands

The data stored within CoreAhbNvm's flash memory blocks for instruction execution or data storage can be accessed by the microprocessor or microcontroller via the software-driven Common Flash Interface. The CFI is a command-driven interface standard used to standardize the low-level flash software algorithms. The CoreAhbNvm IP supports a subset of the CFI commands. The supported commands are summarized in Table 5-14.

Table 5-14 • **Supported CFI Command Descriptions**

Command	No. of Bus Cycles	First Bus Cycle			Second Bus Cycle		
		Operation	Address	Data	Operation	Address	Data
Read Status	2	Write	X	0x70	Read	X	Status Data
Clear Status	1	Write	X	0x50	–	–	–
Read Array	1 or 2	Write	X	0xFF	Read	Array Address	Array Data
Erase Page	2	Write	Page Address	0x20	Write	Page Address	D0h
Single Write	2	Write	Page Address	0x40	Write	Array Address	Array Data
Multi-Write	2	Write	Page Address	0xE8	Write	Page Address	No. of elements – 1

Upon a system reset (LOW on HRESETn), CoreAhbNvm defaults to the read-array operation mode to support the microprocessor instruction execution usage. No write command is needed to place the CoreAhbNvm IP in the read-array mode as instructed in Table 5-14.

As part of the CFI, a status register is used to provide feedback to the software regarding command execution. During each command, the status register should be read to ensure that the flash memory is not busy executing a command (bit 7 set to 1) and that the command executed appropriately. All new commands are ignored while the flash memory is busy (bit 7 set to 0). The status register also includes the Program/Erase, Write, and Read/Protection Error flags (bits 5, 4, and 1, respectively). All three error flags are registered; therefore, once an error is detected, the status register must be cleared using the Clear Status command. Once a CFI command is issued and bit 7 (the Ready flag) is set to 1, the error flags should be checked to ensure that the completed operation did not incur an error. Table 5-15 on page 5-48 includes a bit description of the status register flags.

Table 5-15 • **Status Register Bit Descriptions**

Bit(s)	Flag	Description
7	Ready	Active-high Ready flag used to indicate when the flash memory is ready to receive new commands or is busy processing a command's operation. If Ready is set to 1, the flash memory is ready to receive new commands, and the previous command's operation is complete. If Ready is set to 0, the flash memory is busy processing the command's operation. No new commands should be issued until the Ready flag is at 1. Upon an HRESETn reset, the Ready flag defaults to 1.
6	–	–
5	Program or Erase Error	Active-high Program or Erase Error flag used to indicate whether an error occurred during a program or erase operation. If the Program or Erase Error flag is set to 1, the flash memory incurred an error during a program or erase operation. A locked page access will cause a program or erase operation to fail, triggering the error flag to transition to 1. If the flag transitions to 1, it will remain there until cleared by the Clear Status command operation. Upon an HRESETn reset, the Program or Erase Error flag defaults to 0.
4	Write Error	Active-high Write Error flag used to indicate whether an error occurred during a write operation. If the Write Error flag is set to 1, an error occurred during the page buffer write operation. If the Write Error flag is set to 0, the page buffer write operation was successful. If the flag transitions to 1, it will remain there until cleared by the Clear Status command operation. Upon an HRESETn reset, the Write Error flag defaults to 0.
3–2	–	–
1	Read or Protection Error	The Read or Protection Error flag is used to indicate whether an error occurred during a read, program, or erase operation. If a read operation completed and the Read Error flag is set to 1, an error occurred during the read operation. If a program/erase operation completed and the Protection Error flag is set to 1, the page program or erase operation failed because the page being accessed is protected. If the flag is set to 0, the read, program, or erase operation completed successfully. If the flag transitions to 1, it will remain there until cleared by the Clear Status command operation. Upon an HRESETn reset, the Read or Protection Error flag defaults to 0.
0	–	–

Read Status Command

The status register contains flags used to inform the user when the flash is ready for the next operation or when an error occurred with the last operation performed. Prior to issuing any new array write or read commands to the CFI control logic, the Ready flag (bit 7 of the status register) should be checked. If Ready is set to 1, the flash memory is ready to receive a new command. If Ready is set to 0, the flash memory is busy performing an operation and all new commands issued will be ignored. Once the Ready flag is set to 1, the Program/Erase, Write, and Read/Protection Error flags (bits 5, 4, and 1, respectively) should be checked to ensure that the completed operation did not incur an error. All three error flags are registered; therefore, once an error is detected, the status register must be cleared using the Clear Status command.

Read Status command execution is performed as follows:

1. Issue the Read Status command by writing the command value 0x70 to any location in the flash array.

   ```
   [Any Array Address] = 0x70
   ```

2. Once the command has been issued, read and store the contents of the status register to a temporary variable. To read the contents of the status register, issue a read operation at any address in the flash array. The value read will be the contents of the status register.

   ```
   Status_Register_Contents = [Any Array Address]
   ```

Clear Status Command

The Clear Status command clears the Program/Erase, Write, and Read/Protection Error flags (bits 5, 4, and 1, respectively) of the status register. Clear Status command execution is performed in a single step:

Issue the Clear Status command by writing the command value 0x50 to any location in the flash array.

```
[Any Array Address] = 0x50
```

Read Array Command

The Read Array command is used to place CoreAhbNvm in read-array mode. Once CoreAhbNvm is in read-array mode, it will remain in this mode until a new CFI command is issued. Upon an HRESETn reset, CoreAhbNvm defaults to read-array mode to support microprocessor instruction execution usage. When performing a continuous read and a page boundary has been crossed, the embedded flash memory must reload the page buffer with the new flash page being accessed. In this situation, either the Ready flag of the status register must be monitored or the system clock frequency must be adjusted to allow for the loading of the page buffer. If monitoring the Ready flag, once it is at 1, the Read Error flag should be checked to ensure that the completed operation did not incur an error. The Read Error flag is registered; therefore, once an error is detected, the status register must be cleared using the Clear Status command.

Read Array command execution is performed as follows:

1. Issue the Read Array command by writing the command value 0xFF to any location in the Flash Array.

   ```
   [Any Array Address] = 0xFF
   ```

2. Once the command has been issued, the contents of the array can be read by issuing a read operation at the selected array address in flash.

   ```
   Array_Contents = [Array Address]
   ```

If monitoring the Ready flag, perform the following:

3. The Ready flag (bit 7) of the status register must be monitored to determine when the read operation completes. Using the Read Status command, the status register should be polled until Ready flag is set to 1, signaling that the write operation has completed.

   ```
   Ready = Status_Register_Contents & 0x80
   ```

4. The Read Error flag (bit 1) of the status register should also be checked to determine if an error occurred during the read operation. Once the Ready flag is set to 1, the status of the flag can be read.

   ```
   Read_Error = Status_Register_Contents & 0x02
   ```

Erase Page Command

The Erase Page command is used to erase an entire page by writing 0x00 to all locations of the selected page in the embedded flash memory array. Once the Erase Page command has completed, the Protection and Erase Error flags (bits 1 and 5 of the status register) should be checked to determine whether an erase error occurred and whether the page was protected.

Erase Page command execution is performed as follows:

1. Issue the Erase Page command by writing the command value 0x20 to any location within the flash array page to be erased.

   ```
   [Page Address] = 0x20
   ```

2. Issue the Erase Confirm command by writing the command value 0xD0, addressing any location within the flash array page to be erased. Any other command issued will abort the Erase Page command.

   ```
   [Page Address] = 0xD0
   ```

3. The Ready flag (bit 7) of the status register must be monitored to determine when the erase operation completes. Using the Read Status command, the status register should be polled until the Ready flag is at 1, signaling that the write operation has completed.

   ```
   Ready = Status_Register_Contents & 0x80
   ```

4. The Protection and Erase Error flags (bits 1 and 5) of the status register should also be checked to determine whether an error occurred during the write operation. Once the Ready flag is at 1, the status of both flags can be read.

   ```
   Protection_Error = Status_Register_Contents & 0x02
   Erase_Error = Status_Register_Contents & 0x20
   ```

Single Write Command

The Single Write command is used to write a single byte, halfword, or word to the embedded flash memory. When performing a Single Write command operation, the entire page buffer will be written to the embedded flash memory. Therefore, users should avoid a Single Write command operation when more than one location in a page must be written; a Multi-Write command operation should be used wherever possible. Once the write command has completed, the Write and Program Error flags (bits 4 and 5 of the status register) should be check to determine whether an error occurred.

Single Write command execution is performed as follows:

1. Issue the Single Write command by writing the command value 0x40 to any location within the flash array page. It may be simplest to write the command value to the array address of the data to be written.

   ```
   [Page Address] = 0x40
   ```

2. Once the command has been issued, a second write operation must be issued. The write operation consists of writing the array data contents to the array address.

   ```
   [Array Address] = New_Array_Data
   ```

3. The Ready flag (bit 7) of the status register must be monitored to determine when the write operation completes. Using the Read Status command, the status register should be polled until the Ready flag is at 1, signaling that the write operation has completed.

   ```
   Ready = Status_Register_Contents & 0x80
   ```

4. The Write and Program Error flags (bits 4 and 5) of the status register should also be checked to determine whether an error occurred during the write operation. Once the Ready flag is at 1, the status of both flags can be captured.

   ```
   Write_Error = Status_Register_Contents & 0x10
   Program_Error = Status_Register_Contents & 0x20
   ```

Multi-Write Command

The Multi-Write command is used to write multiple consecutive bytes, half-words, or words to a single page of the embedded flash memory. Initially, the flash memory page is read from the array and loaded into the flash memory's page buffer. *N*, the number of data elements (bytes, words, or double words) to be written to the array, minus one, is then written to CoreAhbNvm and serves as the maximum number of data elements used by the internal counter. The expected N count ranges are N = 00h to N = 7Fh (e.g., 1 to 128 bytes) in 8-bit mode, N = 00h to N = 3Fh in 16-bit mode, and N = 00h to N = 1Fh in 32-bit mode. All data is written into the page buffer sequentially, starting from the first address of the page data is written to. If N exceeds the starting address plus N addresses of the selected page, the data writes will wrap around onto the top of the current page stored in the page buffer. Once all data values are written into the page buffer, the Program Buffer Confirm command (0xD0) is expected to be issued at the next write cycle after the last data element is written; any other command issued at this point in the sequence will prevent the programming of the page buffer into the array (the Multi-Write command will be aborted). Once the program operation has completed, the Write and Program Error flags (bits 4 and 5 of the status register) should be checked to determine whether an error occurred.

Multi-Write command execution is performed as follows:

1. Issue the Multi-Write command by writing the command value 0xE8, addressing the flash page to be written. It may be simplest to write the command value to the starting address of the sequence of data values to be written.

   ```
   [Page Address] = 0xE8
   ```

2. The Ready flag (bit 7) of the status register must be monitored to determine when the page buffer fill operation completes. Using the Read Status command, the status register should be polled until the Ready flag is at 1, signaling that the page buffer fill operation has completed.

   ```
   Ready = Status_Register_Contents & 0x80
   ```

3. Issue a write operation, writing the number of elements to be written to the flash array, minus one (N), to the page address. Again, it may be simplest to write N to the starting address.

   ```
   [Page Address] = N       // (N = number of elements - 1)
   ```

4. Once the command and number of elements has been issued, a sequence of write operations must be issued. The write operation consists of writing the array data contents to the array address. All data is written sequentially, beginning from the starting address, set by the address of the first data value written. Repeat this step until all N elements have been written into the page buffer.

   ```
   [Array Address] = New_Array_Data
   ```

5. Issue the Buffer Program Confirm command by writing the command value 0xD0, addressing the flash page to be written. Any other command issued will abort the programming of the page buffer.

   ```
   [Page Address] = 0xD0
   ```

6. The Ready flag (bit 7) of the status register must be monitored to determine when the page buffer write operation completes. Using the Read Status command, the status register should be polled until the Ready flag is at 1, signaling that the write operation has completed.

   ```
   Ready = Status_Register_Contents & 0x80
   ```

7. The Write and Program Error flags (bits 4 and 5) of the status register should also be checked to determine whether an error occurred during the write operation. Once the Ready flag is at 1, the status of both flags can be captured.

   ```
   Write_Error = Status_Register_Contents & 0x10
   Program_Error = Status_Register_Contents & 0x20
   ```

Part Number and Revision Date

Part Number 51700092-005-0
Revised November 2007

6 – FlashROM in Actel's Low-Power Flash Devices

Introduction

The Fusion, IGLOO,® and ProASIC®3 families of low-power flash-based devices have a dedicated nonvolatile FlashROM memory of 1,024 bits, which provides a unique feature in the FPGA market. The FlashROM can be read, modified, and written using the JTAG (or UJTAG) interface. It can be read but not modified from the FPGA core. Only low-power flash devices contain on-chip user nonvolatile memory (NVM).

Architecture of User Nonvolatile FlashROM

Low-power flash devices have 1 kbit of user-accessible nonvolatile flash memory on-chip that can be read from the FPGA core fabric. The FlashROM is arranged in eight banks of 128 bits (16 bytes) during programming. The 128 bits in each bank are addressable as 16 bytes during the read-back of the FlashROM from the FPGA core. Figure 6-1 shows the FlashROM logical structure.

The FlashROM can only be programmed via the IEEE 1532 JTAG port. It cannot be programmed directly from the FPGA core. When programming, each of the eight 128-bit banks can be selectively reprogrammed. The FlashROM can only be reprogrammed on a bank boundary. Programming involves an automatic, on-chip bank erase prior to reprogramming the bank. The FlashROM supports synchronous read. The address is latched on the rising edge of the clock, and the new output data is stable after the falling edge of the same clock cycle. For more information, refer to the timing diagrams in the DC and Switching Characteristics chapter of the appropriate datasheet. The FlashROM can be read on byte boundaries. The upper three bits of the FlashROM address from the FPGA core define the bank being accessed. The lower four bits of the FlashROM address from the FPGA core define which of the 16 bytes in the bank is being accessed.

		Byte Number in Bank 4 LSB of ADDR (READ)															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bank Number 3 MSB of ADDR (READ)	7																
	6																
	5																
	4																
	3																
	2																
	1																
	0																

Figure 6-1 • **FlashROM Architecture**

FlashROM Support in Flash-Based Devices

The flash FPGAs listed in Table 6-1 support the FlashROM feature and the functions described in this document.

Table 6-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 6-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 6-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

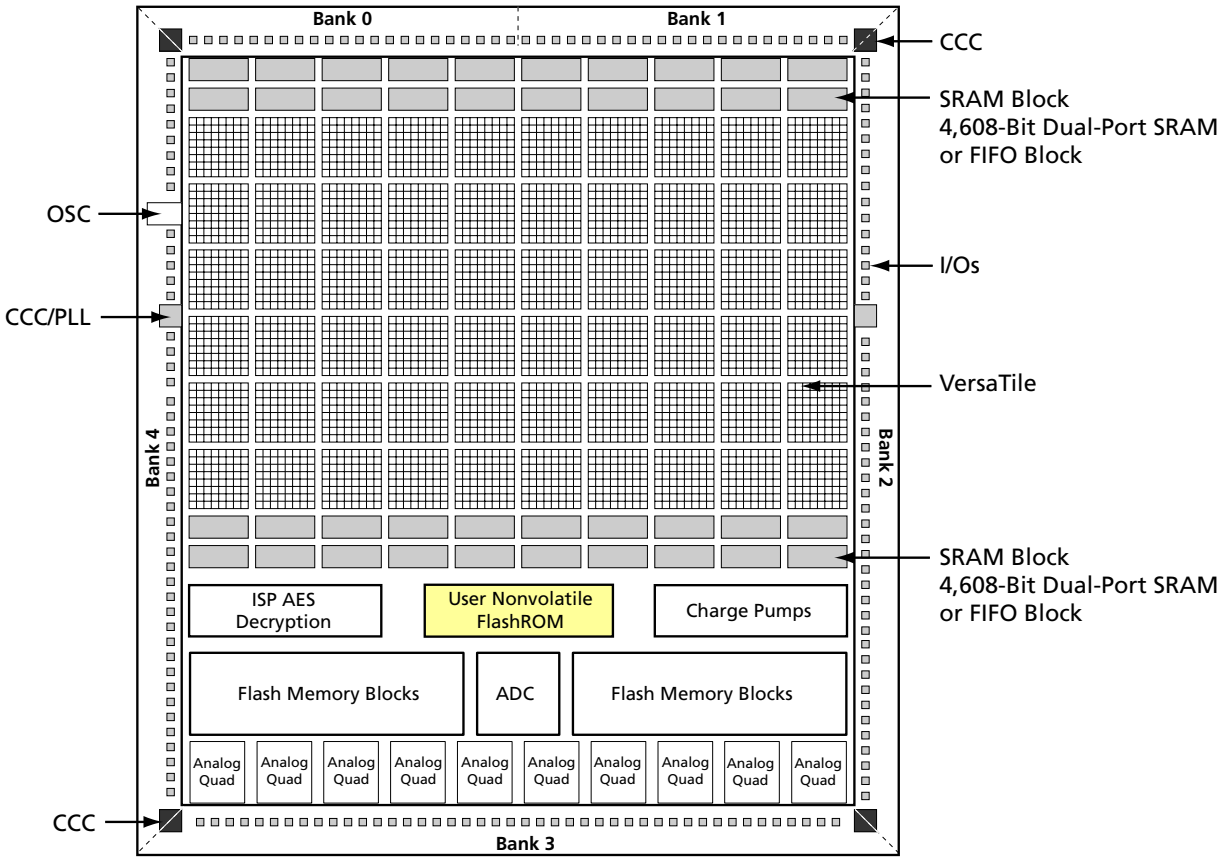


Figure 6-2 • **Fusion Device Architecture Overview (AFS600)**

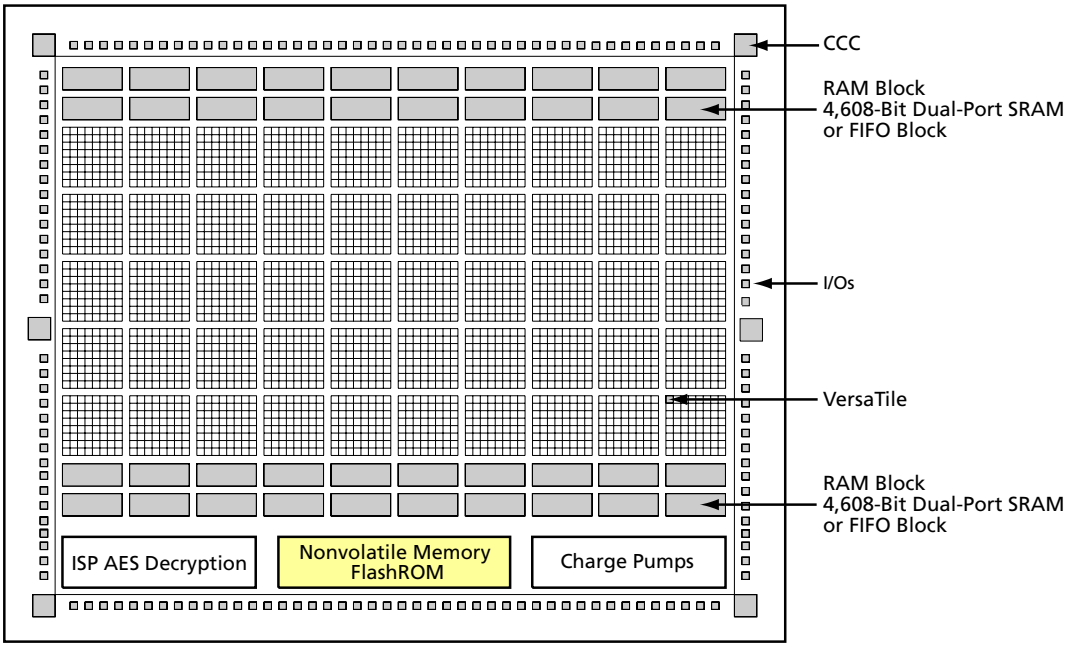


Figure 6-3 • **ProASIC3 and IGLOO Device Architecture**

FlashROM Applications

The SmartGen core generator is used to configure FlashROM content. You can configure each page independently. SmartGen enables you to create and modify regions within a page; these regions can be 1 to 16 bytes long (Figure 6-4).

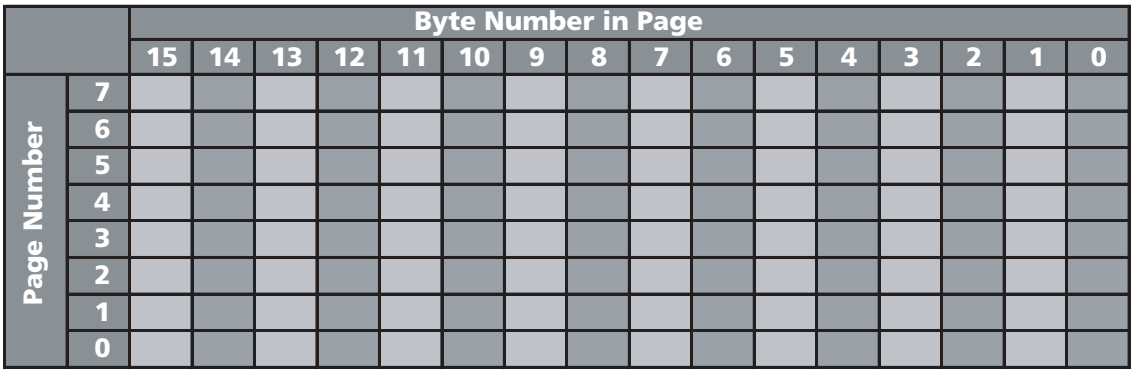


Figure 6-4 • **FlashROM Configuration**

The FlashROM content can be changed independently of the FPGA core content. It can be easily accessed and programmed via JTAG, depending on the security settings of the device. The SmartGen core generator enables each region to be independently updated (described in the "Programming and Accessing FlashROM" section on page 6-6). This enables you to change the FlashROM content on a per-part basis while keeping some regions "constant" for all parts. These features allow the FlashROM to be used in diverse system applications. Consider the following possible uses of FlashROM:

- Internet protocol (IP) addressing (wireless or fixed)
- System calibration settings
- Restoring configuration after unpredictable system power-down
- Device serialization and/or inventory control
- Subscription-based business models (e.g., set-top boxes)
- Secure key storage
- Asset management tracking
- Date stamping
- Version management

FlashROM Security

Low-power flash devices have an on-chip Advanced Encryption Standard (AES) decryption core, combined with an enhanced version of the Actel flash-based lock technology (FlashLock®). Together, they provide unmatched levels of security in a programmable logic device. This security applies to both the FPGA core and FlashROM content. These devices use the 128-bit AES (Rijndael) algorithm to encrypt programming files for secure transmission to the on-chip AES decryption core. The same algorithm is then used to decrypt the programming file. This key size provides approximately 3.4×10^{38} possible 128-bit keys. A computing system that could find a DES key in a second would take approximately 149 trillion years to crack a 128-bit AES key. The 128-bit FlashLock feature in low-power flash devices works via a FlashLock security Pass Key mechanism, where the user locks or unlocks the device with a user-defined key. Refer to *Security in Low-Power Flash Devices*.

If the device is locked with certain security settings, functions such as device read, write, and erase are disabled. This unique feature helps to protect against invasive and noninvasive attacks. Without the correct Pass Key, access to the FPGA is denied. To gain access to the FPGA, the device first must be unlocked using the correct Pass Key. During programming of the FlashROM or the FPGA core, you can generate the security header programming file, which is used to program the AES key and/or FlashLock Pass Key. The security header programming file can also be generated independently of the FlashROM and FPGA core content. The FlashLock Pass Key is not stored in the FlashROM.

Low-power flash devices with AES-based security allow for secure remote field updates over public networks such as the Internet, and ensure that valuable intellectual property (IP) remains out of the hands of IP thieves. Figure 6-5 shows this flow diagram.

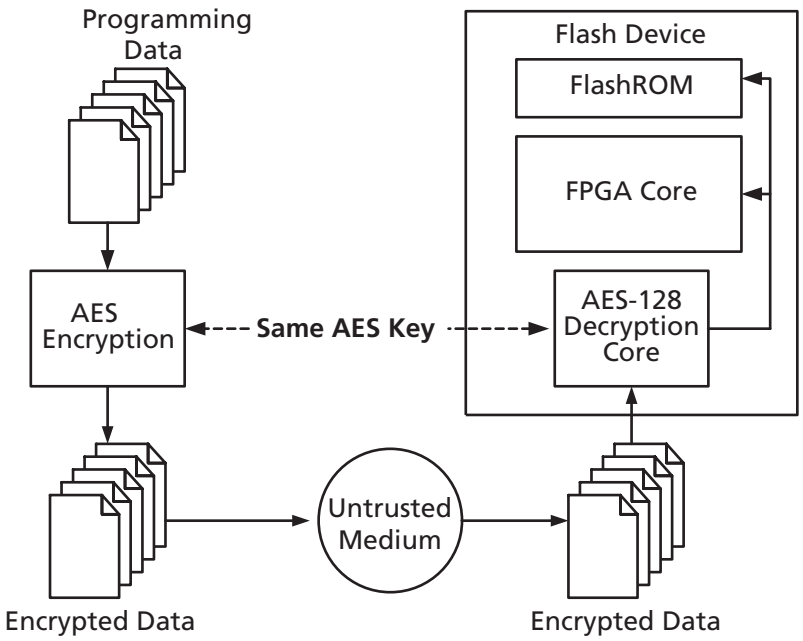


Figure 6-5 • **Programming FlashROM Using AES**

Programming and Accessing FlashROM

The FlashROM content can only be programmed via JTAG, but it can be read back selectively through the JTAG programming interface, the UJTAG interface, or via direct FPGA core addressing. The pages of the FlashROM can be made secure to prevent read-back via JTAG. In that case, read-back on these secured pages is only possible by the FPGA core fabric or via UJTAG.

A 7-bit address from the FPGA core defines which of the eight pages (three MSBs) is being read, and which of the 16 bytes within the selected page (four LSBs) are being read. The FlashROM content can be read on a random basis; the access time is 10 ns for a device supporting commercial specifications. The FPGA core will be powered down during writing of the FlashROM content. FPGA power-down during FlashROM programming is managed on-chip, and FPGA core functionality is not available during programming of the FlashROM. Table 6-2 summarizes various FlashROM access scenarios.

Table 6-2 • **FlashROM Read/Write Capabilities by Access Mode**

Access Mode	FlashROM Read	FlashROM Write
JTAG	Yes	Yes
UJTAG	Yes	No
FPGA core	Yes	No

Figure 6-6 shows the accessing of the FlashROM using the UJTAG macro. This is similar to FPGA core access, where the 7-bit address defines which of the eight pages (three MSBs) is being read and which of the 16 bytes within the selected page (four LSBs) are being read. Refer to *UJTAG Applications in Actel's Low-Power Flash Devices* for details on using the UJTAG macro to read the FlashROM.

Figure 6-7 on page 6-7 and Figure 6-8 on page 6-7 show the FlashROM access from the JTAG port. The FlashROM content can be read on a random basis. The three-bit address defines which page is being read or updated.

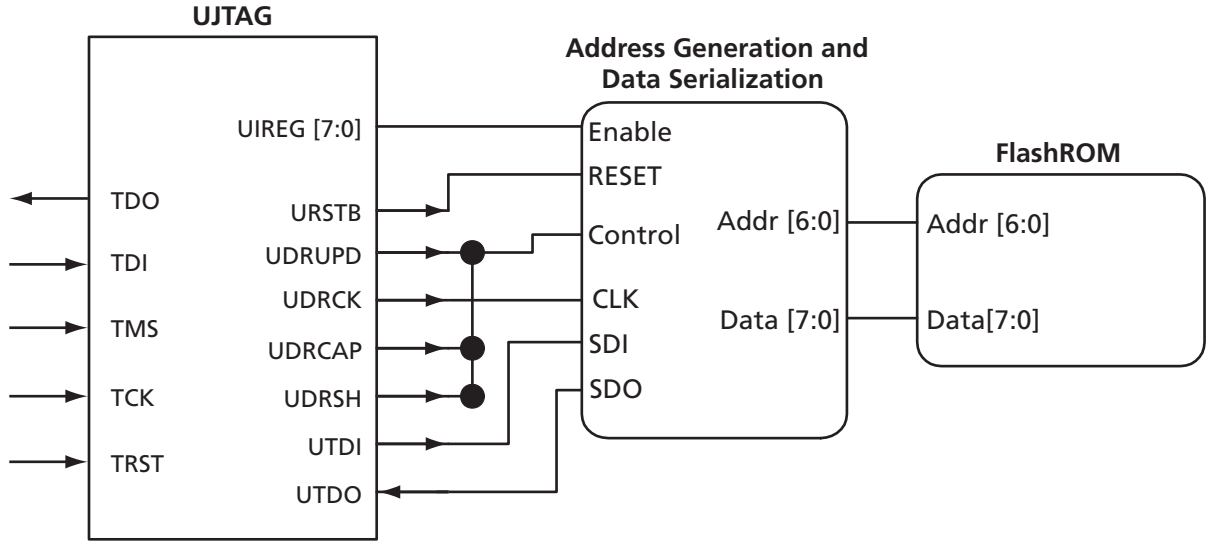


Figure 6-6 • **Block Diagram of Using UJTAG to Read FlashROM Contents**

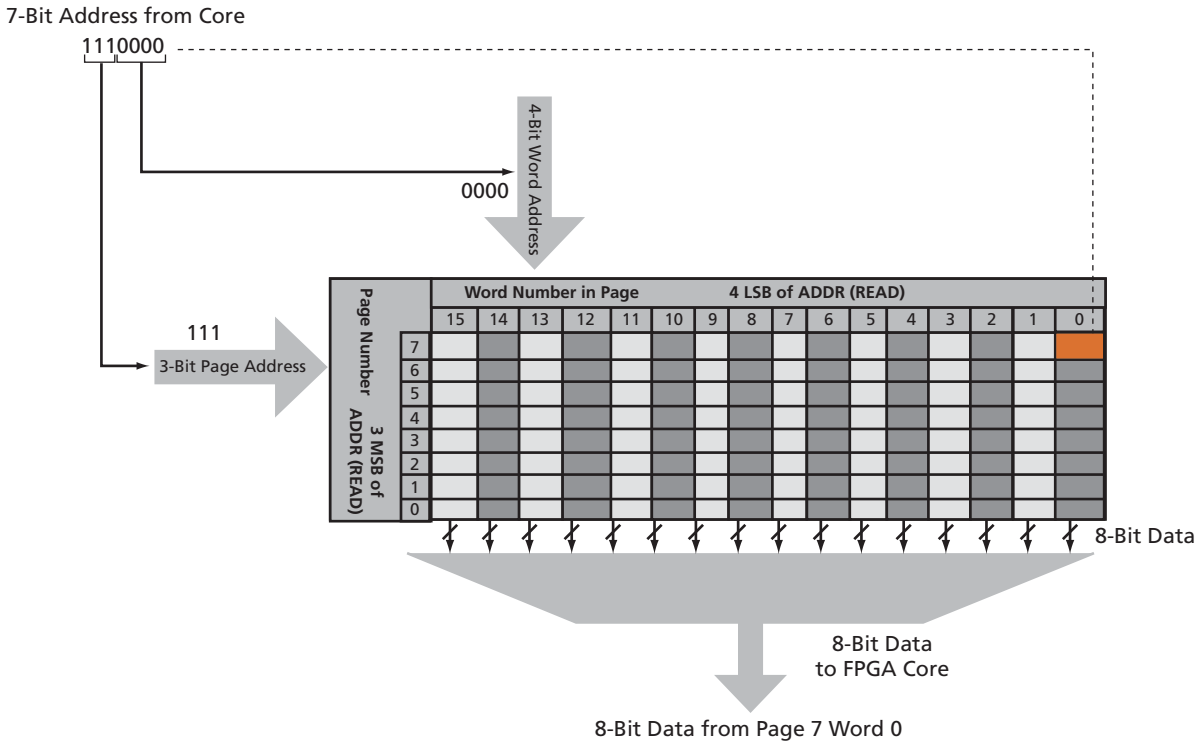


Figure 6-7 • **Accessing FlashROM Using FPGA Core**

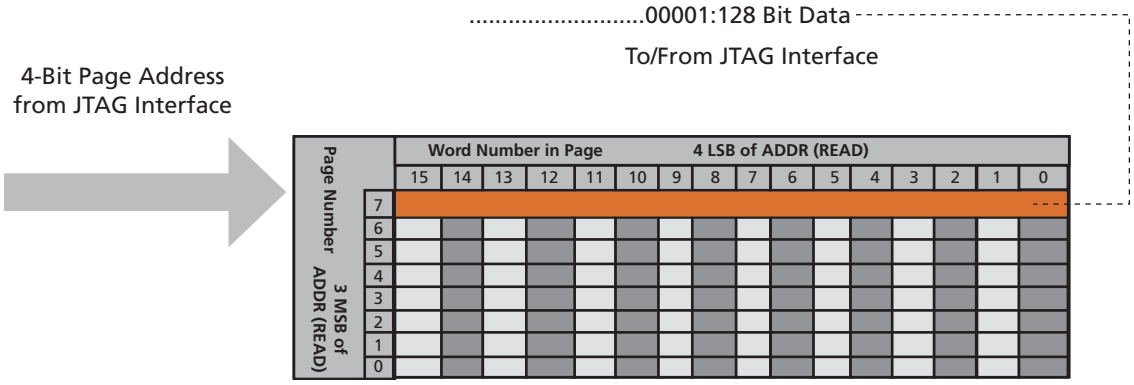


Figure 6-8 • **Accessing FlashROM Using JTAG Port**

FlashROM Design Flow

The Actel Libero® Integrated Design Environment (IDE) software has extensive FlashROM support, including FlashROM generation, instantiation, simulation, and programming. Figure 6-9 shows the user flow diagram. In the design flow, there are three main steps:

1. FlashROM generation and instantiation in the design
2. Simulation of FlashROM design
3. Programming file generation for FlashROM design

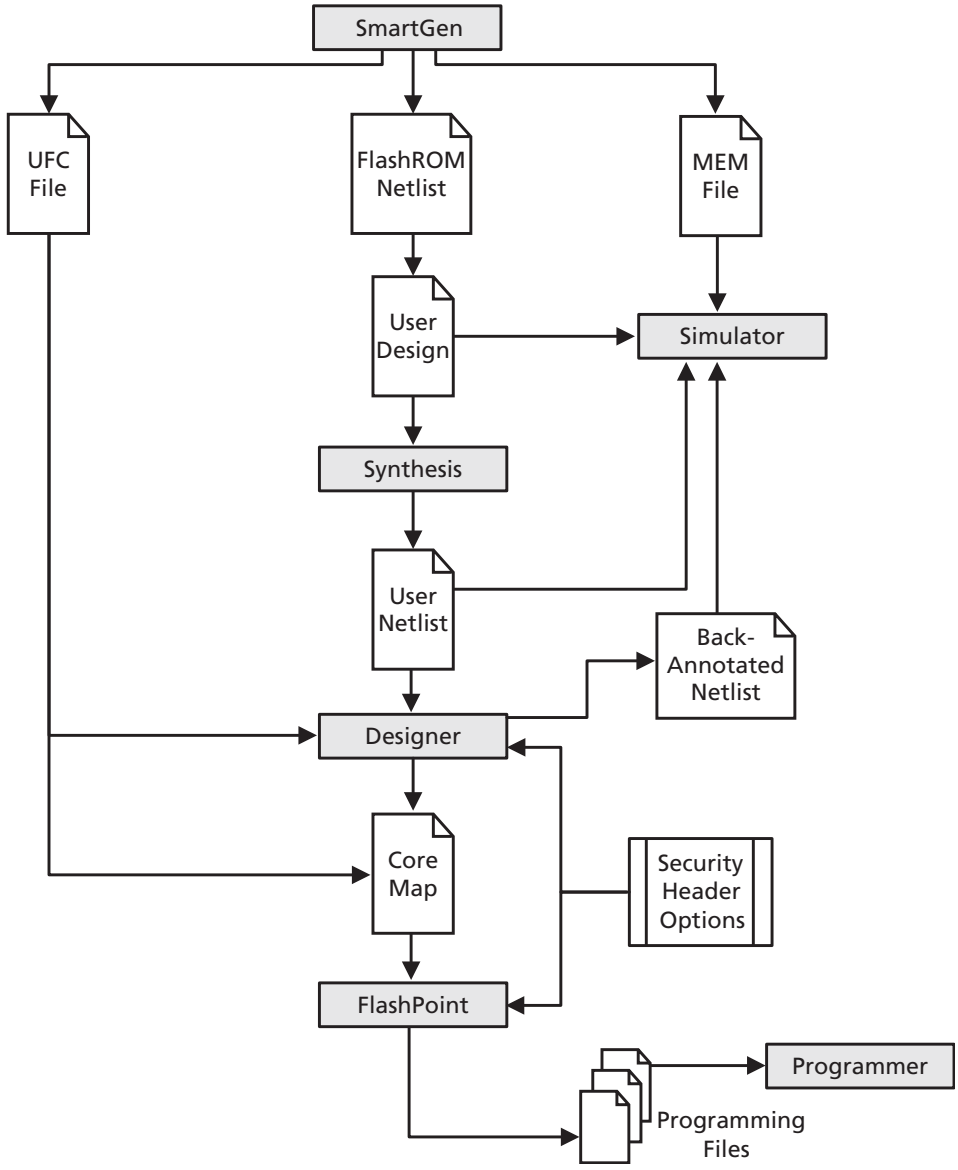


Figure 6-9 • **FlashROM Design Flow**

FlashROM Generation and Instantiation in the Design

The SmartGen core generator, available in Libero IDE and Designer, is the only tool that can be used to generate the FlashROM content. SmartGen has several user-friendly features to help generate the FlashROM contents. Instead of selecting each byte and assigning values, you can create a region within a page, modify the region, and assign properties to that region. The FlashROM user interface, shown in Figure 6-10 on page 6-9, includes the configuration grid, existing regions list, and properties field. The properties field specifies the region-specific information and defines the data used for that region. You can assign values to the following properties:

1. Static Fixed Data—Enables you to fix the data so it cannot be changed during programming time. This option is useful when you have fixed data stored in this region, which is required for the operation of the design in the FPGA. Key storage is one example.
2. Static Modifiable Data—Select this option when the data in a particular region is expected to be static data (such as a version number, which remains the same for a long duration but could conceivably change in the future). This option enables you to avoid changing the value every time you enter new data.
3. Read from File—This provides the full flexibility of FlashROM usage to the customer. If you have a customized algorithm for generating the FlashROM data, you can specify this setting. You can then generate a text file with data for as many devices as you wish to program, and load that into the FlashPoint programming file generation software to get programming files that include all the data. SmartGen will optionally pass the location of the file where the data is stored if the file is specified in SmartGen. Each text file has only one type of data format (binary, decimal, hex, or ASCII text). The length of each data file must be shorter than or equal to the selected region length. If the data is shorter than the selected region length, the most significant bits will be padded with 0s. For multiple text files for multiple regions, the first lines are for the first device. In SmartGen, **Load Sim. Value From File** allows you to load the first device data in the MEM file for simulation.
4. Auto Increment/Decrement—This scenario is useful when you specify the contents of FlashROM for a large number of devices in a series. You can specify the step value for the serial number and a maximum value for inventory control. During programming file generation, the actual number of devices to be programmed is specified and a start value is fed to the software.

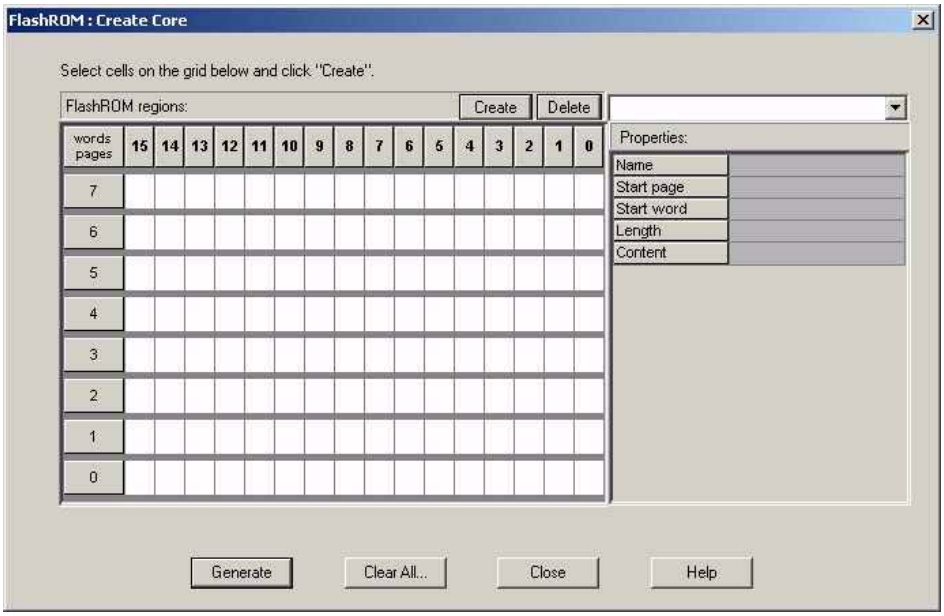


Figure 6-10 • **SmartGen GUI of the FlashROM**

SmartGen allows you to generate the FlashROM netlist in VHDL, Verilog, or EDIF format. After the FlashROM netlist is generated, the core can be instantiated in the main design like other SmartGen cores. Note that the macro library name for FlashROM is UFROM. The following is a sample FlashROM VHDL netlist that can be instantiated in the main design:

```
library ieee;
use ieee.std_logic_1164.all;
library fusion;

entity FROM_a is
  port( ADDR : in std_logic_vector(6 downto 0); DOUT : out std_logic_vector(7 downto 0));
end FROM_a;

architecture DEF_ARCH of  FROM_a is

  component UFROM
    generic (MEMORYFILE:string);
    port(DO0, DO1, DO2, DO3, DO4, DO5, DO6, DO7 : out std_logic;
      ADDR0, ADDR1, ADDR2, ADDR3, ADDR4, ADDR5, ADDR6 : in std_logic := 'U') ;
  end component;

  component GND
    port( Y : out std_logic);
  end component;

signal U_7_PIN2 : std_logic ;

begin

  GND_1_net : GND port map(Y => U_7_PIN2);
  UFROM0 : UFROM
  generic map(MEMORYFILE => "FROM_a.mem")
  port map(DO0 => DOUT(0), DO1 => DOUT(1), DO2 => DOUT(2), DO3 => DOUT(3), DO4 => DOUT(4),
    DO5 => DOUT(5), DO6 => DOUT(6), DO7 => DOUT(7), ADDR0 => ADDR(0), ADDR1 => ADDR(1),
    ADDR2 => ADDR(2), ADDR3 => ADDR(3), ADDR4 => ADDR(4), ADDR5 => ADDR(5),
    ADDR6 => ADDR(6));

end DEF_ARCH;
```

SmartGen generates the following files along with the netlist. These are located in the SmartGen folder for the Libero IDE project.

1. MEM (Memory Initialization) file
2. UFC (User Flash Configuration) file
3. Log file

The MEM file is used for simulation, as explained in the "Simulation of FlashROM Design" section. The UFC file, generated by SmartGen, has the FlashROM configuration for single or multiple devices and is used during STAPL generation. It contains the region properties and simulation values. Note that any changes in the MEM file will not be reflected in the UFC file. Do not modify the UFC to change FlashROM content. Instead, use the SmartGen GUI to modify the FlashROM content. See the "Programming File Generation for FlashROM Design" section on page 6-11 for a description of how the UFC file is used during the programming file generation. The log file has information regarding the file type and file location.

Simulation of FlashROM Design

The MEM file has 128 rows of 8 bits, each representing the contents of the FlashROM used for simulation. For example, the first row represents page 0, byte 0; the next row is page 0, byte 1; and so the pattern continues. Note that the three MSBs of the address define the page number, and the four LSBs define the byte number. So, if you send address 0000100 to FlashROM, this corresponds to the page 0 and byte 4 location, which is the fifth row in the MEM file. SmartGen defaults to 0s for any unspecified location of the FlashROM. Besides using the MEM file generated by SmartGen, you can create a binary file with 128 rows of 8 bits each and use this as a MEM file. Actel recommends that you use different file names if you plan to generate multiple MEM files. During simulation, Libero IDE passes the MEM file used as the generic file in the netlist, along with the design files and testbench. If you want to use different MEM files during simulation, you need to modify the generic file reference in the netlist.

```
.....
UFROM0: UFROM
--generic map(MEMORYFILE => "F:\Appsnotes\FROM\test_designs\testa\smartgen\FROM_a.mem")
--generic map(MEMORYFILE => "F:\Appsnotes\FROM\test_designs\testa\smartgen\FROM_b.mem")
.....
```

The VITAL and Verilog simulation models accept the generics passed by the netlist, read the MEM file, and perform simulation with the data in the file.

Programming File Generation for FlashROM Design

FlashPoint is the programming software used to generate the programming files for flash devices. Depending on the applications, you can use the FlashPoint software to generate a STAPL file with different FlashROM contents. In each case, optional AES decryption is available. To generate a STAPL file that contains the same FPGA core content and different FlashROM contents, the FlashPoint software needs an Array Map file for the core and UFC file(s) for the FlashROM. This final STAPL file represents the combination of the logic of the FPGA core and FlashROM content.

FlashPoint generates the STAPL files you can use to program the desired FlashROM page and/or FPGA core of the FPGA device contents. FlashPoint supports the encryption of the FlashROM content and/or FPGA Array configuration data. In the case of using the FlashROM for device serialization, a sequence of unique FlashROM contents will be generated. When generating a programming file with multiple unique FlashROM contents, you can specify in FlashPoint whether to include all FlashROM content in a single STAPL file or generate a different STAPL file for each FlashROM (Figure 6-11). The programming software (FlashPro) handles the single STAPL file that contains the FlashROM content from multiple devices. It enables you to program the FlashROM content into a series of devices sequentially (Figure 6-11). See the *FlashPro User's Guide* for information on serial programming.

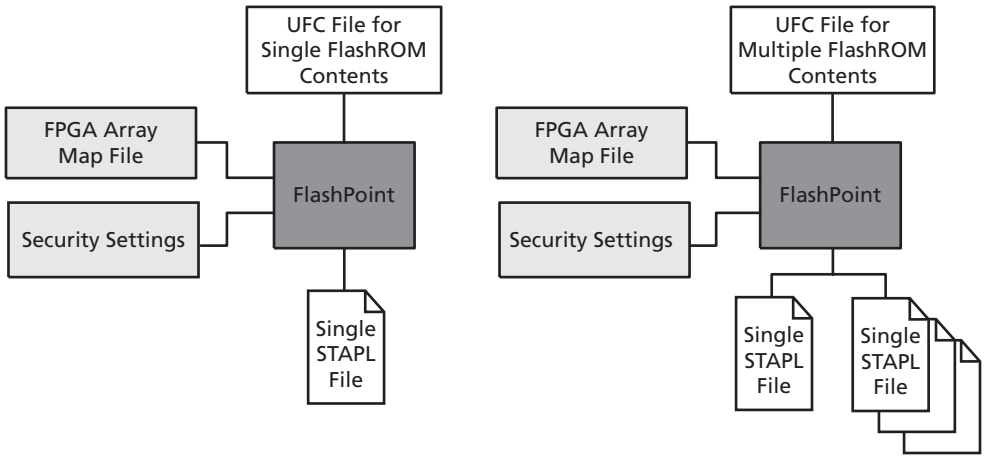


***Figure 6-11* • Single or Multiple Programming File Generation**

Figure 6-12 shows the programming file generator, which enables different STAPL file generation methods. When you select **Program FlashROM** and choose the UFC file, the FlashROM Settings window appears, as shown in Figure 6-13. In this window, you can select the FlashROM page you want to program and the data value for the configured regions. This enables you to use a different page for different programming files.

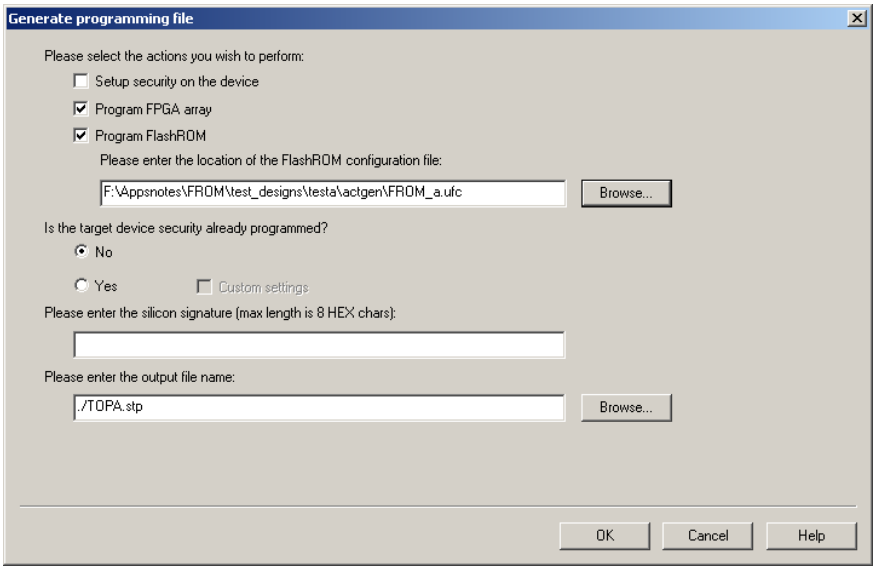


Figure 6-12 • **Programming File Generator**

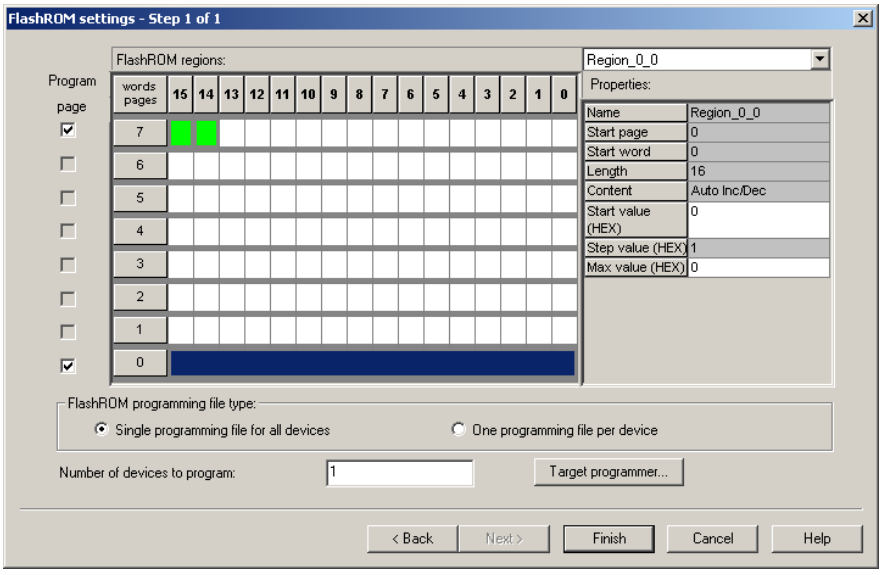


Figure 6-13 • **Setting FlashROM during Programming File Generation**

The programming hardware and software can load the FlashROM with the appropriate STAPL file. Programming software handles the single STAPL file that contains multiple FlashROM contents for multiple devices, and programs the FlashROM in sequential order (e.g., for device serialization).

This feature is supported in the programming software. After programming with the STAPL file, you can run **DEVICE_INFO** to check the FlashROM content.

DEVICE_INFO displays the FlashROM content, serial number, Design Name, and checksum, as shown below:

```
EXPORT IDCODE[32] = 123261CF
EXPORT SILSIG[32] = 00000000
User information :
CHECKSUM: 61A0
Design Name:        TOP
Programming Method: STAPL
Algorithm Version: 1
Programmer: UNKNOWN
=======================================
FlashROM Information :
EXPORT Region_7_0[128] = FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
=======================================
Security Setting :
Encrypted FlashROM Programming Enabled.
Encrypted FPGA Array Programming Enabled.
=======================================
```

The Libero IDE file manager recognizes the UFC and MEM files and displays them in the appropriate view. Libero IDE also recognizes the multiple programming files if you choose the option to generate multiple files for multiple FlashROM contents in Designer. These features enable a user-friendly flow for the FlashROM generation and programming in Libero IDE.

Custom Serialization Using FlashROM

You can use FlashROM for device serialization or inventory control by using the Auto Inc region or Read From File region. FlashPoint will automatically generate the serial number sequence for the Auto Inc region with the **Start Value**, **Max Value**, and **Step Value** provided. If you have a unique serial number generation scheme that you prefer, the Read From File region allows you to import the file with your serial number scheme programmed into the region. See the *FlashPro User's Guide* for custom serialization file format information.

The following steps describe how to perform device serialization or inventory control using FlashROM:

1. Generate FlashROM using SmartGen. From the **Properties** section in the **FlashROM Settings** dialog box, select the **Auto Inc** or **Read From File** region. For the **Auto Inc** region, specify the desired step value. You will not be able to modify this value in the FlashPoint software.
2. Go through the regular design flow and finish place-and-route.
3. Select **Programming File** in Designer and open **Generate Programming File** (Figure 6-12 on page 6-12).
4. Click **Program FlashROM**, browse to the UFC file, and click **Next**. The FlashROM Settings window appears, as shown in Figure 6-13 on page 6-12.
5. Select the FlashROM page you want to program and the data value for the configured regions. The STAPL file generated will contain only the data that targets the selected FlashROM page.
6. Modify properties for the serialization.
   - For the Auto Inc region, specify the **Start** and **Max** values.
   - For the Read From File region, select the file name of the custom serialization file.
7. Select the FlashROM programming file type you want to generate from the two options below:
   - Single programming file for all devices: generates one programming file with all FlashROM values.
   - One programming file per device: generates a separate programming file for each FlashROM value.

8. Enter the number of devices you want to program and generate the required programming file.
9. Open the programming software and load the programming file. The programming software, FlashPro3 and Silicon Sculptor II, supports the device serialization feature. If, for some reason, the device fails to program a part during serialization, the software allows you to reuse or skip the serial data. Refer to the *FlashPro User's Guide* for details.

Conclusion

The Fusion, IGLOO, and ProASIC3 families are the only FPGAs that offer on-chip FlashROM support. This document presents information on the FlashROM architecture, possible applications, programming, access through the JTAG and UJTAG interface, and integration into your design. In addition, the Libero IDE tool set enables easy creation and modification of the FlashROM content.

The nonvolatile FlashROM block in the FPGA can be customized, enabling multiple applications.

Additionally, the security offered by the low-power flash devices keeps both the contents of FlashROM and the FPGA design safe from system over-builders, system cloners, and IP thieves.

Related Documents

Handbook Documents

Security in Low-Power Flash Devices

www.actel.com/documents/LPD_Security_HBs.pdf

UJTAG Applications in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_UJTAG_HBs.pdf

User's Guides

FlashPro User's Guide

http://www.actel.com/documents/FlashPro_UG.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-007-4

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 6-1 · Flash-Based FPGAs.	6-2
v1.2 (June 2008)	The "FlashROM Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	6-2
	Figure 6-2 · Fusion Device Architecture Overview (AFS600) was replaced. Figure 6-5 · Programming FlashROM Using AES was revised to change "Fusion" to "Flash Device."	6-3, 6-5
	The *FlashPoint User's Guide* was removed from the "User's Guides" section, as its content is now part of the *FlashPro User's Guide*.	6-14
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 6-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	6-2
v1.0 (January 2008)	The chapter was updated to include the IGLOO PLUS family and information regarding 15 k gate devices. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	N/A

7 – SRAM and FIFO Memories in Actel's Low-Power Flash Devices

Introduction

As design complexity grows, greater demands are placed upon an FPGA's embedded memory. Actel Fusion,® IGLOO,® and ProASIC®3 devices provide the flexibility of true dual-port and two-port SRAM blocks. The embedded memory, along with built-in, dedicated FIFO control logic, can be used to create cascading RAM blocks and FIFOs without using additional logic gates.

IGLOO, IGLOO PLUS, and ProASIC3L FPGAs contain an additional feature that allows the device to be put in a low-power mode called Flash*Freeze. In this mode, the core draws minimal power (on the order of 2 to 127 µW) and still retains values on the embedded SRAM/FIFO and registers. Flash*Freeze technology allows the user to switch to Active mode on demand, thus simplifying power management and the use of SRAM/FIFOs.

Device Architecture

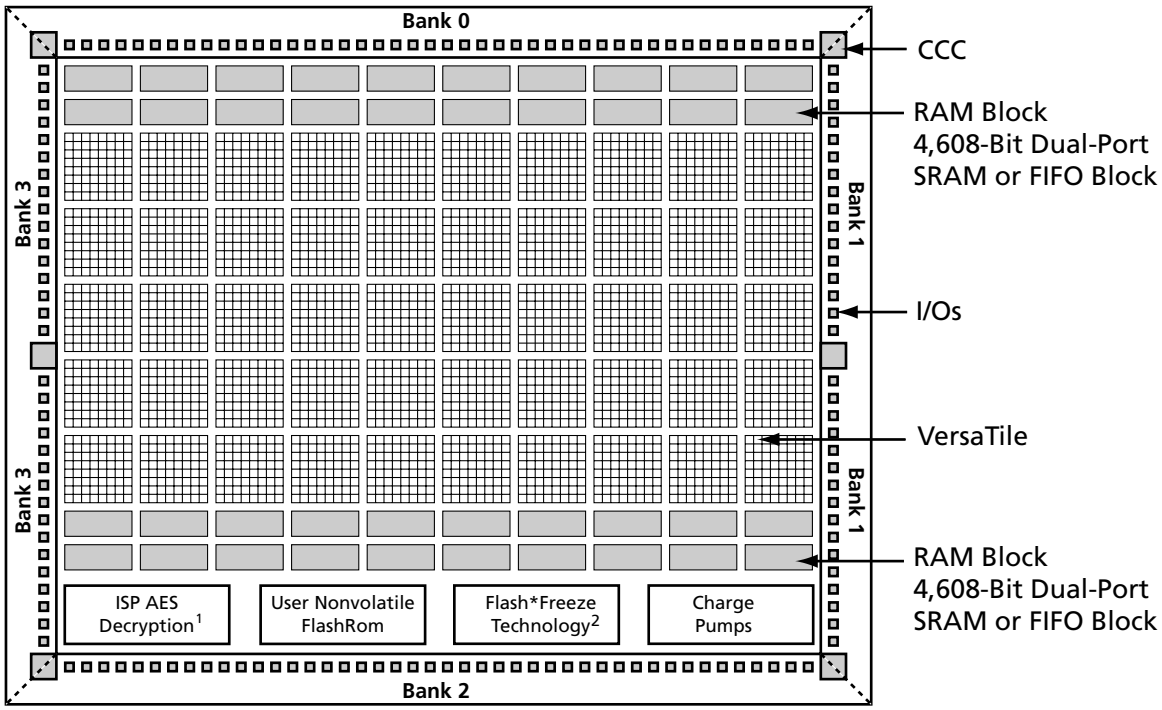
The low-power flash devices feature up to 504 kbits of RAM in 4,608-bit blocks (Figure 7-1 on page 7-2 and Figure 7-2 on page 7-3). The total embedded SRAM for each device can be found in the datasheets. These memory blocks are arranged along the top and bottom of the device to allow better access from the core and I/O (in some devices, they are only available on the north side of the device). Every RAM block has a flexible, hardwired, embedded FIFO controller, enabling the user to implement efficient FIFOs without sacrificing user gates.

In the IGLOO and ProASIC3 families of devices, the following memories are supported:

- 30 k gate devices and smaller do not support SRAM and FIFO.
- 60 k and 125 k gate devices support memories on the north side of the device only.
- 250 k devices and larger support memories on the north and south sides of the device.

In Fusion devices, the following memories are supported:

- AFS090 and AFS250 support memories on the north side of the device only.
- AFS600 and AFS1500 support memories on the north and south sides of the device.



Notes:

1. *AES decryption not supported in 30 k gate devices and smaller.*
2. *Flash*Freeze is supported in all IGLOO devices and the ProASIC3L devices.*

Figure 7-1 • **IGLOO and ProASIC3 Device Architecture Overview**

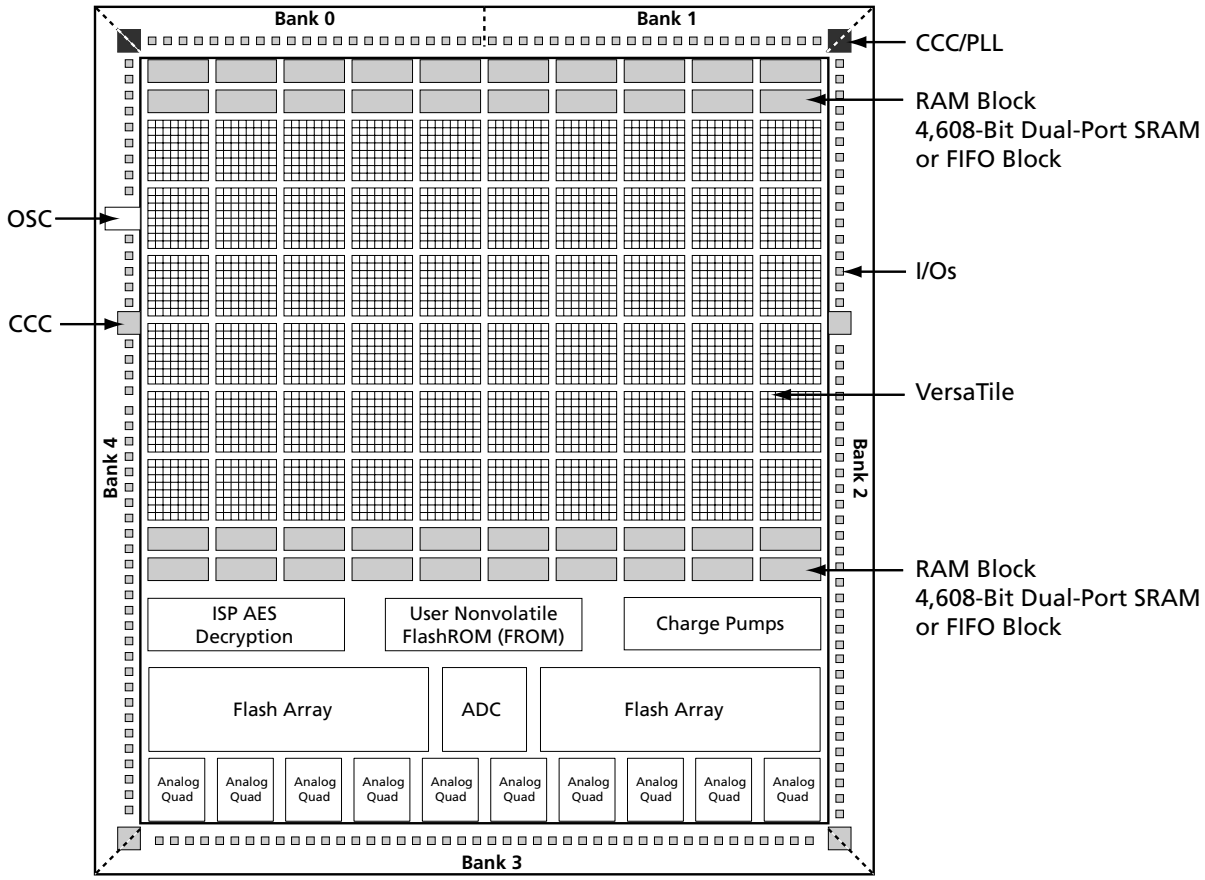


Figure 7-2 • **Fusion Device Architecture Overview (AFS600)**

SRAM/FIFO Support in Flash-Based Devices

The flash FPGAs listed in Table 7-1 support SRAM and FIFO blocks and the functions described in this document.

Table 7-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 7-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 7-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

SRAM and FIFO Architecture

To meet the needs of high-performance designs, the memory blocks operate strictly in synchronous mode for both read and write operations. The read and write clocks are completely independent, and each can operate at any desired frequency up to 250 MHz.

- 4k×1, 2k×2, 1k×4, 512×9 (dual-port RAM—2 read / 2 write or 1 read / 1 write)
- 512×9, 256×18 (2-port RAM—1 read / 1 write)
- Sync write, sync pipelined / nonpipelined read

Automotive ProASIC3 devices support single-port SRAM capabilities or dual-port SRAM only under specific conditions. Dual-port mode is supported if the clocks to the two SRAM ports are the same and 180° out of phase (i.e., the port A clock is the inverse of the port B clock). The Actel Libero® Integrated Design Environment (IDE) software macro libraries support a dual-port macro only. For use of this macro as a single-port SRAM, the inputs and clock of one port should be tied off (grounded) to prevent errors during design compile. For use in dual-port mode, the same clock with an inversion between the two clock pins of the macro should be used in the design to prevent errors during compile.

The memory block includes dedicated FIFO control logic to generate internal addresses and external flag logic (FULL, EMPTY, AFULL, AEMPTY).

Simultaneous dual-port read/write and write/write operations at the same address are allowed when certain timing requirements are met.

During RAM operation, addresses are sourced by the user logic, and the FIFO controller is ignored. In FIFO mode, the internal addresses are generated by the FIFO controller and routed to the RAM array by internal MUXes.

The low-power flash device architecture enables the read and write sizes of RAMs to be organized independently, allowing for bus conversion. For example, the write size can be set to 256×18 and the read size to 512×9.

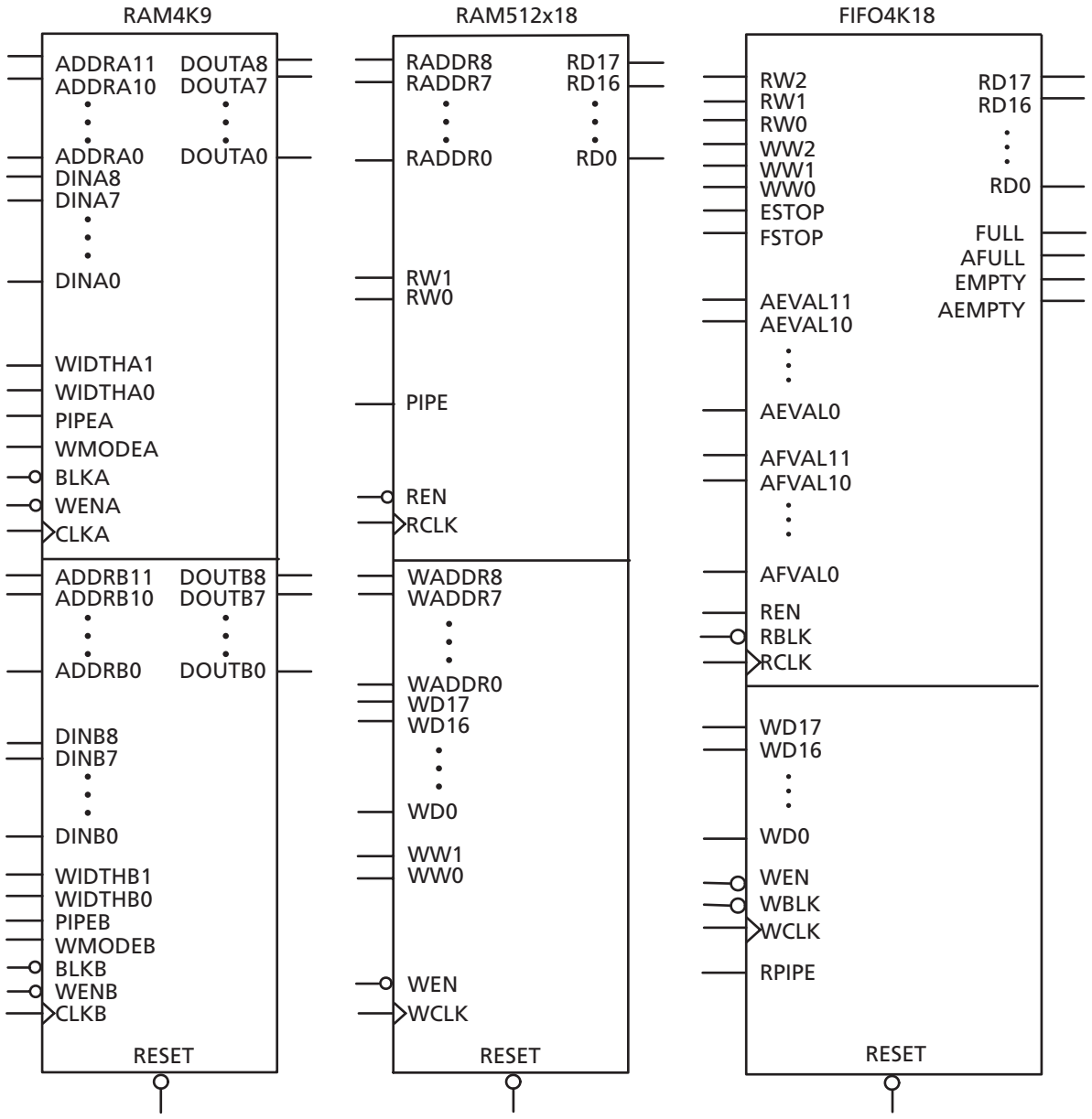
Both the write width and read width for the RAM blocks can be specified independently with the WW (write width) and RW (read width) pins. The different D×W configurations are 256×18, 512×9, 1k×4, 2k×2, and 4k×1. When widths of one, two, or four are selected, the ninth bit is unused. For example, when writing nine-bit values and reading four-bit values, only the first four bits and the second four bits of each nine-bit value are addressable for read operations. The ninth bit is not accessible.

Conversely, when writing four-bit values and reading nine-bit values, the ninth bit of a read operation will be undefined. The RAM blocks employ little-endian byte order for read and write operations.

Memory Blocks and Macros

Memory blocks can be configured with many different aspect ratios, but are generically supported in the macro libraries as one of two memory elements: RAM4K9 or RAM512X18. The RAM4K9 is configured as a true dual-port memory block, and the RAM512X18 is configured as a two-port memory block. Dual-port memory allows the RAM to both read from and write to either port independently. Two-port memory allows the RAM to read from one port and write to the other using a common clock or independent read and write clocks. If needed, the RAM4K9 blocks can be configured as two-port memory blocks. The memory block can be configured as a FIFO by combining the basic memory block with dedicated FIFO controller logic. The FIFO macro is named FIFO4KX18 (Figure 7-3 on page 7-6).

Clocks for the RAM blocks can be driven by the VersaNet (global resources) or by regular nets. When using local clock segments, the clock segment region that encompasses the RAM blocks can drive the RAMs. In the dual-port configuration (RAM4K9), each memory block port can be driven by either rising-edge or falling-edge clocks. Each port can be driven by clocks with different edges. Though only a rising-edge clock can drive the physical block itself, the Actel Designer software will automatically bubble-push the inversion to properly implement the falling-edge trigger for the RAM block.



Note: Automotive ProASIC3 devices restrict RAM4K9 to a single port or to dual ports with the same clock 180° out of phase (inverted) between clock pins. In single-port mode, inputs to port B should be tied to ground to prevent errors during compile. For FIFO4K18, the same clock 180° out of phase (inverted) between clock pins should be used.

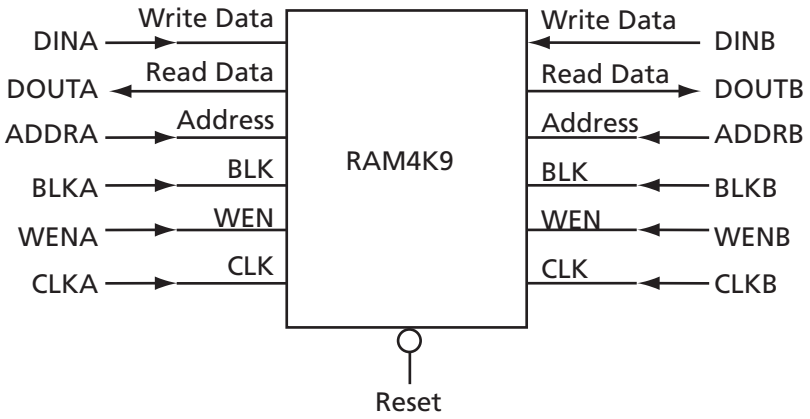
Figure 7-3 • **Supported Basic RAM Macros**

SRAM Features

RAM4K9 Macro

RAM4K9 is the dual-port configuration of the RAM block (Figure 7-4). The RAM4K9 nomenclature refers to both the deepest possible configuration and the widest possible configuration the dual-port RAM block can assume, and does not denote a possible memory aspect ratio. The RAM block can be configured to the following aspect ratios: 4,096×1, 2,048×2, 1,024×4, and 512×9. RAM4K9 is fully synchronous and has the following features:

- Two ports that allow fully independent reads and writes at different frequencies
- Selectable pipelined or nonpipelined read
- Active-low block enables for each port
- Toggle control between read and write mode for each port
- Active-low asynchronous reset
- Pass-through write data or hold existing data on output. In pass-through mode, the data written to the write port will immediately appear on the read port.
- Designer software will automatically facilitate falling-edge clocks by bubble-pushing the inversion to previous stages.



Note: For timing diagrams of the RAM signals, refer to the appropriate family datasheet.

Figure 7-4 • **RAM4K9 Simplified Configuration**

Signal Descriptions for RAM4K9

Note: Automotive ProASIC3 devices support single-port SRAM capabilities, or dual-port SRAM only under specific conditions. Dual-port mode is supported if the clocks to the two SRAM ports are the same and 180° out of phase (i.e., the port A clock is the inverse of the port B clock). Since Actel Libero IDE macro libraries support a dual-port macro only, certain modifications must be made. These are detailed below.

The following signals are used to configure the RAM4K9 memory element:

WIDTHA and WIDTHB

These signals enable the RAM to be configured in one of four allowable aspect ratios (Table 7-2 on page 7-8).

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, WIDTHB should be tied to ground.

Table 7-2 • **Allowable Aspect Ratio Settings for WIDTHA[1:0]**

WIDTHA[1:0]	WIDTHB[1:0]	D×W
00	00	4k×1
01	01	2k×2
10	10	1k×4
11	11	512×9

Note: The aspect ratio settings are constant and cannot be changed on the fly.

BLKA and BLKB

These signals are active-low and will enable the respective ports when asserted. When a BLKx signal is deasserted, that port's outputs hold the previous value.

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, BLKB should be tied to ground.

WENA and WENB

These signals switch the RAM between read and write modes for the respective ports. A LOW on these signals indicates a write operation, and a HIGH indicates a read.

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, WENB should be tied to ground.

CLKA and CLKB

These are the clock signals for the synchronous read and write operations. These can be driven independently or with the same driver.

Note: For Automotive ProASIC3 devices, dual-port mode is supported if the clocks to the two SRAM ports are the same and 180° out of phase (i.e., the port A clock is the inverse of the port B clock). For use of this macro as a single-port SRAM, the inputs and clock of one port should be tied off (grounded) to prevent errors during design compile.

PIPEA and PIPEB

These signals are used to specify pipelined read on the output. A LOW on PIPEA or PIPEB indicates a nonpipelined read, and the data appears on the corresponding output in the same clock cycle. A HIGH indicates a pipelined read, and data appears on the corresponding output in the next clock cycle.

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, PIPEB should be tied to ground. For use in dual-port mode, the same clock with an inversion between the two clock pins of the macro should be used in the design to prevent errors during compile.

WMODEA and WMODEB

These signals are used to configure the behavior of the output when the RAM is in write mode. A LOW on these signals makes the output retain data from the previous read. A HIGH indicates pass-through behavior, wherein the data being written will appear immediately on the output. This signal is overridden when the RAM is being read.

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, WMODEB should be tied to ground.

RESET

This active-low signal resets the control logic, forces the output hold state registers to zero, disables reads and writes from the SRAM block, and clears the data hold registers when asserted. It does not reset the contents of the memory array.

While the RESET signal is active, read and write operations are disabled. As with any asynchronous reset signal, care must be taken not to assert it too close to the edges of active read and write clocks.

ADDRA and ADDRB

These are used as read or write addresses, and they are 12 bits wide. When a depth of less than 4 k is specified, the unused high-order bits must be grounded (Table 7-3 on page 7-9).

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, ADDRB should be tied to ground.

Table 7-3 • **Address Pins Unused/Used for Various Supported Bus Widths**

D×W	ADDRx	
	Unused	Used
4k×1	None	[11:0]
2k×2	[11]	[10:0]
1k×4	[11:10]	[9:0]
512×9	[11:9]	[8:0]

Note: The "x" in ADDRx implies A or B.

DINA and DINB

These are the input data signals, and they are nine bits wide. Not all nine bits are valid in all configurations. When a data width less than nine is specified, unused high-order signals must be grounded (Table 7-4).

Note: When using the SRAM in single-port mode for Automotive ProASIC3 devices, DINB should be tied to ground.

DOUTA and DOUTB

These are the nine-bit output data signals. Not all nine bits are valid in all configurations. As with DINA and DINB, high-order bits may not be used (Table 7-4). The output data on unused pins is undefined.

Table 7-4 • **Unused/Used Input and Output Data Pins for Various Supported Bus Widths**

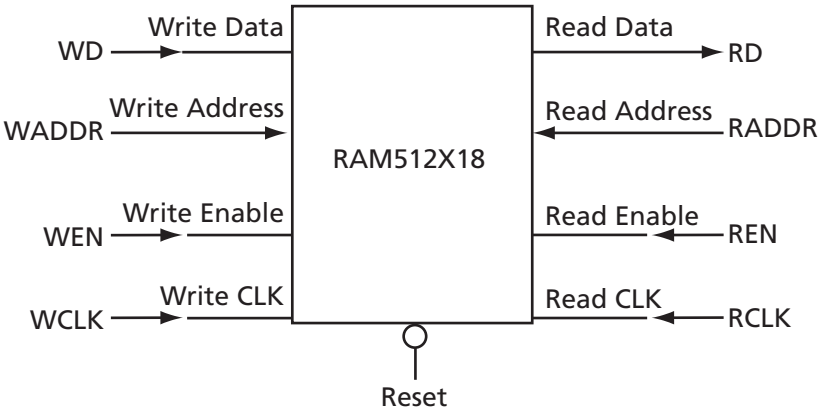
D×W	DINx/DOUTx	
	Unused	Used
4k×1	[8:1]	[0]
2k×2	[8:2]	[1:0]
1k×4	[8:4]	[3:0]
512×9	None	[8:0]

Note: The "x" in DINx or DOUTx implies A or B.

RAM512X18 Macro

RAM512X18 is the two-port configuration of the same RAM block (Figure 7-5 on page 7-10). Like the RAM4K9 nomenclature, the RAM512X18 nomenclature refers to both the deepest possible configuration and the widest possible configuration the two-port RAM block can assume. In two-port mode, the RAM block can be configured to either the 512×9 aspect ratio or the 256×18 aspect ratio. RAM512X18 is also fully synchronous and has the following features:

- Dedicated read and write ports
- Active-low read and write enables
- Selectable pipelined or nonpipelined read
- Active-low asynchronous reset
- Designer software will automatically facilitate falling-edge clocks by bubble-pushing the inversion to previous stages.



Note: For timing diagrams of the RAM signals, refer to the appropriate family datasheet.

Figure 7-5 • **512X18 Two-Port RAM Block Diagram**

Signal Descriptions for RAM512X18

RAM512X18 has slightly different behavior from RAM4K9, as it has dedicated read and write ports.

WW and RW

These signals enable the RAM to be configured in one of the two allowable aspect ratios (Table 7-5).

Table 7-5 • **Aspect Ratio Settings for WW[1:0]**

WW[1:0]	RW[1:0]	D×W
01	01	512×9
10	10	256×18
00, 11	00, 11	Reserved

WD and RD

These are the input and output data signals, and they are 18 bits wide. When a 512×9 aspect ratio is used for write, WD[17:9] are unused and must be grounded. If this aspect ratio is used for read, RD[17:9] are undefined.

WADDR and RADDR

These are read and write addresses, and they are nine bits wide. When the 256×18 aspect ratio is used for write or read, WADDR[8] and RADDR[8] are unused and must be grounded.

WCLK and RCLK

These signals are the write and read clocks, respectively. They can be clocked on the rising or falling edge of WCLK and RCLK.

WEN and REN

These signals are the write and read enables, respectively. They are both active-low by default. These signals can be configured as active-high.

RESET

This active-low signal resets the control logic, forces the output hold state registers to zero, disables reads and writes from the SRAM block, and clears the data hold registers when asserted. It does not reset the contents of the memory array.

While the RESET signal is active, read and write operations are disabled. As with any asynchronous reset signal, care must be taken not to assert it too close to the edges of active read and write clocks.

PIPE

This signal is used to specify pipelined read on the output. A LOW on PIPE indicates a nonpipelined read, and the data appears on the output in the same clock cycle. A HIGH indicates a pipelined read, and data appears on the output in the next clock cycle.

SRAM Usage

The following descriptions refer to the usage of both RAM4K9 and RAM512X18.

Clocking

The dual-port SRAM blocks are only clocked on the rising edge. SmartGen allows falling-edge-triggered clocks by adding inverters to the netlist, hence achieving dual-port SRAM blocks that are clocked on either edge (rising or falling). For dual-port SRAM, each port can be clocked on either edge and by separate clocks by port. Note that for Automotive ProASIC3, the same clock, with an inversion between the two clock pins of the macro, should be used in design to prevent errors during compile.

Low-power flash devices support inversion (bubble-pushing) throughout the FPGA architecture, including the clock input to the SRAM modules. Inversions added to the SRAM clock pin on the design schematic or in the HDL code will be automatically accounted for during design compile without incurring additional delay in the clock path.

The two-port SRAM can be clocked on the rising or falling edge of WCLK and RCLK.

If negative-edge RAM and FIFO clocking is selected for memory macros, clock edge inversion management (bubble-pushing) is automatically used within the development tools, without performance penalty.

Modes of Operation

There are two read modes and one write mode:

- Read Nonpipelined (synchronous—1 clock edge): In the standard read mode, new data is driven onto the RD bus in the same clock cycle following RA and REN valid. The read address is registered on the read port clock active edge, and data appears at RD after the RAM access time. Setting PIPE to OFF enables this mode.
- Read Pipelined (synchronous—2 clock edges): The pipelined mode incurs an additional clock delay from address to data but enables operation at a much higher frequency. The read address is registered on the read port active clock edge, and the read data is registered and appears at RD after the second read clock edge. Setting PIPE to ON enables this mode.
- Write (synchronous—1 clock edge): On the write clock active edge, the write data is written into the SRAM at the write address when WEN is HIGH. The setup times of the write address, write enables, and write data are minimal with respect to the write clock.

RAM Initialization

Each SRAM block can be individually initialized on power-up by means of the JTAG port using the UJTAG mechanism. The shift register for a target block can be selected and loaded with the proper bit configuration to enable serial loading. The 4,608 bits of data can be loaded in a single operation.

FIFO Features

The FIFO4KX18 macro is created by merging the RAM block with dedicated FIFO logic (Figure 7-6 on page 7-12). Since the FIFO logic can only be used in conjunction with the memory block, there is no separate FIFO controller macro. As with the RAM blocks, the FIFO4KX18 nomenclature does not refer to a possible aspect ratio, but rather to the deepest possible data depth and the widest possible data width. FIFO4KX18 can be configured into the following aspect ratios: 4,096×1, 2,048×2, 1,024×4, 512×9, and 256×18. In addition to being fully synchronous, the FIFO4KX18 also has the following features:

- Four FIFO flags: Empty, Full, Almost-Empty, and Almost-Full
- Empty flag is synchronized to the read clock
- Full flag is synchronized to the write clock
- Both Almost-Empty and Almost-Full flags have programmable thresholds

- Active-low asynchronous reset
- Active-low block enable
- Active-low write enable
- Active-high read enable
- Ability to configure the FIFO to either stop counting after the empty or full states are reached or to allow the FIFO counters to continue
- Designer software will automatically facilitate falling-edge clocks by bubble-pushing the inversion to previous stages.

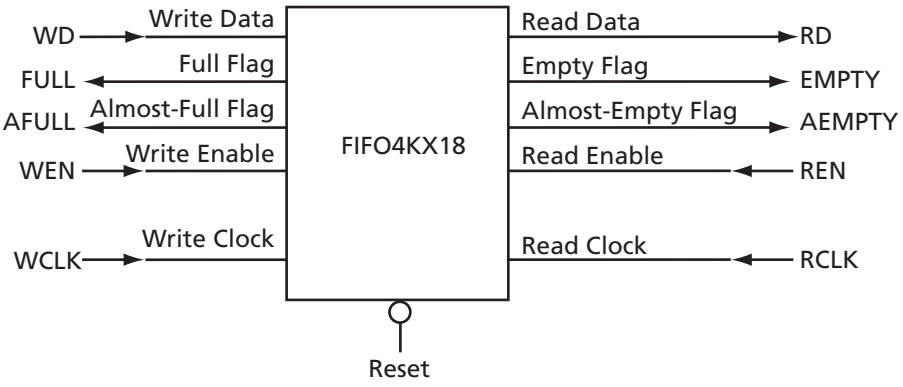


Figure 7-6 • **FIFO4KX18 Block Diagram**

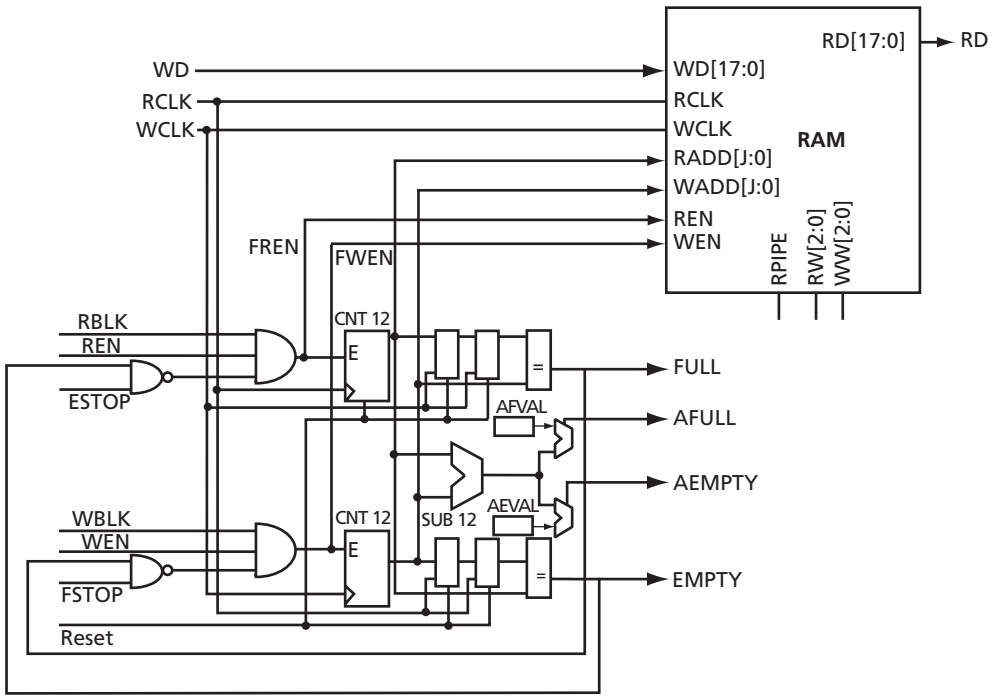


Figure 7-7 • **RAM Block with Embedded FIFO Controller**

The FIFOs maintain a separate read and write address. Whenever the difference between the write address and the read address is greater than or equal to the almost-full value (AFVAL), the Almost-Full flag is asserted. Similarly, the Almost-Empty flag is asserted whenever the difference between the write address and read address is less than or equal to the almost-empty value (AEVAL).

Due to synchronization between the read and write clocks, the Empty flag will deassert after the second read clock edge from the point that the write enable asserts. However, since the Empty flag is synchronized to the read clock, it will assert after the read clock reads the last data in the FIFO. Also, since the Full flag is dependent on the actual hardware configuration, it will assert when the actual physical implementation of the FIFO is full.

For example, when a user configures a 128×18 FIFO, the actual physical implementation will be a 256×18 FIFO element. Since the actual implementation is 256×18, the Full flag will not trigger until the 256×18 FIFO is full, even though a 128×18 FIFO was requested. For this example, the Almost-Full flag can be used instead of the Full flag to signal when the 128th data word is reached.

To accommodate different aspect ratios, the almost-full and almost-empty values are expressed in terms of data bits instead of data words. SmartGen translates the user's input, expressed in data words, into data bits internally. SmartGen allows the user to select the thresholds for the Almost-Empty and Almost-Full flags in terms of either the read data words or the write data words, and makes the appropriate conversions for each flag.

After the empty or full states are reached, the FIFO can be configured so the FIFO counters either stop or continue counting. For timing numbers, refer to the appropriate family datasheet.

Signal Descriptions for FIFO4K18

The following signals are used to configure the FIFO4K18 memory element:

WW and RW

These signals enable the FIFO to be configured in one of the five allowable aspect ratios (Table 7-6).

Table 7-6 • **Aspect Ratio Settings for WW[2:0]**

WW[2:0]	RW[2:0]	D×W
000	000	4k×1
001	001	2k×2
010	010	1k×4
011	011	512×9
100	100	256×18
101, 110, 111	101, 110, 111	Reserved

WBLK and RBLK

These signals are active-low and will enable the respective ports when LOW. When the RBLK signal is HIGH, that port's outputs hold the previous value.

WEN and REN

Read and write enables. WEN is active-low and REN is active-high by default. These signals can be configured as active-high or -low.

WCLK and RCLK

These are the clock signals for the synchronous read and write operations. These can be driven independently or with the same driver.

Note: For the Automotive ProASIC3 FIFO4K18, for the same clock, 180° out of phase (inverted) between clock pins should be used.

RPIPE

This signal is used to specify pipelined read on the output. A LOW on RPIPE indicates a nonpipelined read, and the data appears on the output in the same clock cycle. A HIGH indicates a pipelined read, and data appears on the output in the next clock cycle.

RESET

This active-low signal resets the control logic and forces the output hold state registers to zero when asserted. It does not reset the contents of the memory array (Table 7-7 on page 7-14).

While the RESET signal is active, read and write operations are disabled. As with any asynchronous RESET signal, care must be taken not to assert it too close to the edges of active read and write clocks.

WD

This is the input data bus and is 18 bits wide. Not all 18 bits are valid in all configurations. When a data width less than 18 is specified, unused higher-order signals must be grounded (Table 7-7).

RD

This is the output data bus and is 18 bits wide. Not all 18 bits are valid in all configurations. Like the WD bus, high-order bits become unusable if the data width is less than 18. The output data on unused pins is undefined (Table 7-7).

Table 7-7 • **Input Data Signal Usage for Different Aspect Ratios**

D×W	WD/RD Unused
4k×1	WD[17:1], RD[17:1]
2k×2	WD[17:2], RD[17:2]
1k×4	WD[17:4], RD[17:4]
512×9	WD[17:9], RD[17:9]
256×18	–

ESTOP, FSTOP

ESTOP is used to stop the FIFO read counter from further counting once the FIFO is empty (i.e., the EMPTY flag goes HIGH). A HIGH on this signal inhibits the counting.

FSTOP is used to stop the FIFO write counter from further counting once the FIFO is full (i.e., the FULL flag goes HIGH). A HIGH on this signal inhibits the counting.

For more information on these signals, refer to the "ESTOP and FSTOP Usage" section on page 7-15.

FULL, EMPTY

When the FIFO is full and no more data can be written, the FULL flag asserts HIGH. The FULL flag is synchronous to WCLK to inhibit writing immediately upon detection of a full condition and to prevent overflows. Since the write address is compared to a resynchronized (and thus time-delayed) version of the read address, the FULL flag will remain asserted until two WCLK active edges after a read operation eliminates the full condition.

When the FIFO is empty and no more data can be read, the EMPTY flag asserts HIGH. The EMPTY flag is synchronous to RCLK to inhibit reading immediately upon detection of an empty condition and to prevent underflows. Since the read address is compared to a resynchronized (and thus time-delayed) version of the write address, the EMPTY flag will remain asserted until two RCLK active edges after a write operation removes the empty condition.

For more information on these signals, refer to the "FIFO Flag Usage Considerations" section on page 7-15.

AFULL, AEMPTY

These are programmable flags and will be asserted on the threshold specified by AFVAL and AEVAL, respectively.

When the number of words stored in the FIFO reaches the amount specified by AEVAL while reading, the AEMPTY output will go HIGH. Likewise, when the number of words stored in the FIFO reaches the amount specified by AFVAL while writing, the AFULL output will go HIGH.

AFVAL, AEVAL

The AEVAL and AFVAL pins are used to specify the almost-empty and almost-full threshold values. They are 12-bit signals. For more information on these signals, refer to the "FIFO Flag Usage Considerations" section on page 7-15.

FIFO Usage

ESTOP and FSTOP Usage

The ESTOP pin is used to stop the read counter from counting any further once the FIFO is empty (i.e., the EMPTY flag goes HIGH). Likewise, the FSTOP pin is used to stop the write counter from counting any further once the FIFO is full (i.e., the FULL flag goes HIGH).

The FIFO counters in the device start the count at zero, reach the maximum depth for the configuration (e.g., 511 for a 512×9 configuration), and then restart at zero. An example application for ESTOP, where the read counter keeps counting, would be writing to the FIFO once and reading the same content over and over without doing another write.

FIFO Flag Usage Considerations

The AEVAL and AFVAL pins are used to specify the 12-bit AEMPTY and AFULL threshold values. The FIFO contains separate 12-bit write address (WADDR) and read address (RADDR) counters. WADDR is incremented every time a write operation is performed, and RADDR is incremented every time a read operation is performed. Whenever the difference between WADDR and RADDR is greater than or equal to AFVAL, the AFULL output is asserted. Likewise, whenever the difference between WADDR and RADDR is less than or equal to AEVAL, the AEMPTY output is asserted. To handle different read and write aspect ratios, AFVAL and AEVAL are expressed in terms of total data bits instead of total data words. When users specify AFVAL and AEVAL in terms of read or write words, the SmartGen tool translates them into bit addresses and configures these signals automatically. SmartGen configures the AFULL flag to assert when the write address exceeds the read address by at least a predefined value. In a 2k×8 FIFO, for example, a value of 1,500 for AFVAL means that the AFULL flag will be asserted after a write when the difference between the write address and the read address reaches 1,500 (there have been at least 1,500 more writes than reads). It will stay asserted until the difference between the write and read addresses drops below 1,500.

The AEMPTY flag is asserted when the difference between the write address and the read address is less than a predefined value. In the example above, a value of 200 for AEVAL means that the AEMPTY flag will be asserted when a read causes the difference between the write address and the read address to drop to 200. It will stay asserted until that difference rises above 200. Note that the FIFO can be configured with different read and write widths; in this case, the AFVAL setting is based on the number of write data entries, and the AEVAL setting is based on the number of read data entries. For aspect ratios of 512×9 and 256×18, only 4,096 bits can be addressed by the 12 bits of AFVAL and AEVAL. The number of words must be multiplied by 8 and 16 instead of 9 and 18. The SmartGen tool automatically uses the proper values. To avoid halfwords being written or read, which could happen if different read and write aspect ratios were specified, the FIFO will assert FULL or EMPTY as soon as at least one word cannot be written or read. For example, if a two-bit word is written and a four-bit word is being read, the FIFO will remain in the empty state when the first word is written. This occurs even if the FIFO is not completely empty, because in this case, a complete word cannot be read. The same is applicable in the full state. If a four-bit word is written and a two-bit word is read, the FIFO is full and one word is read. The FULL flag will remain asserted because a complete word cannot be written at this point.

Variable Aspect Ratio and Cascading

Variable aspect ratio and cascading allow users to configure the memory in the width and depth required. The memory block can be configured as a FIFO by combining the basic memory block with dedicated FIFO controller logic. The FIFO macro is named FIFO4KX18. Low-power flash device RAM can be configured as 1, 2, 4, 9, or 18 bits wide. By cascading the memory blocks, any multiple of those widths can be created. The RAM blocks can be from 256 to 4,096 bits deep, depending on the aspect ratio, and the blocks can also be cascaded to create deeper areas. Refer to the aspect ratios available for each macro cell in the "SRAM Features" section on page 7-7. The largest continuous configurable memory area is equal to half the total memory available on the device, because the RAM is separated into two groups, one on each side of the device.

The Actel SmartGen core generator will automatically configure and cascade both RAM and FIFO blocks. Cascading is accomplished using dedicated memory logic and does not consume user gates for depths up to 4,096 bits deep and widths up to 18, depending on the configuration. Deeper memory will utilize some user gates to multiplex the outputs.

Generated RAM and FIFO macros can be created as either structural VHDL or Verilog for easy instantiation into the design. Users of Actel Libero IDE can create a symbol for the macro and incorporate it into a design schematic.

Table 7-10 on page 7-17 shows the number of memory blocks required for each of the supported depth and width memory configurations, and for each depth and width combination. For example, a 256-bit deep by 32-bit wide two-port RAM would consist of two 256×18 RAM blocks. The first 18 bits would be stored in the first RAM block, and the remaining 14 bits would be implemented in the other 256×18 RAM block. This second RAM block would have four bits of unused storage. Similarly, a dual-port memory block that is 8,192 bits deep and 8 bits wide would be implemented using 16 memory blocks. The dual-port memory would be configured in a 4,096×1 aspect ratio. These blocks would then be cascaded two deep to achieve 8,192 bits of depth, and eight wide to achieve the eight bits of width.

Table 7-8 and Table 7-9 show the maximum potential width and depth configuration for each device. Note that 15 k and 30 k gate devices do not support RAM or FIFO.

Table 7-8 • **Memory Availability per IGLOO and ProASIC3 Device**

Device		RAM Blocks	Maximum Potential Width[1]		Maximum Potential Depth[2]	
IGLOO IGLOO nano IGLOO PLUS	ProASIC3 ProASIC3 nano ProASIC3L		Depth	Width	Depth	Width
AGL060 AGLN060 AGLP060	A3P060 A3PN060	4	256	72 (4×18)	16,384 (4,096×4)	1
AGL125 AGLN125 AGLP125	A3P125 A3PN125	8	256	144 (8×18)	32,768 (4,094×8)	1
AGL250 AGLN250	A3P250/L A3PN250	8	256	144 (8×18)	32,768 (4,096×8)	1
AGL400	A3P400	12	256	216 (12×18)	49,152 (4,096×12)	1
AGL600	A3P600/L	24	256	432 (24×18)	98,304 (4,096×24)	1
AGL1000	A3P1000/L	32	256	576 (32×18)	131,072 (4,096×32)	1
AGLE600	A3PE600	24	256	432 (24×18)	98,304 (4,096×24)	1
	A3PE1500	60	256	1,080 (60×18)	245,760 (4,096×60)	1
AGLE3000	A3PE3000/L	112	256	2,016 (112×18)	458,752 (4,096×112)	1

Notes:

1. *Maximum potential width uses the two-port configuration.*
2. *Maximum potential depth uses the dual-port configuration.*

Table 7-9 • **Memory Availability per Fusion Device**

Device	RAM Blocks	Maximum Potential Width[1]		Maximum Potential Depth[2]	
		Depth	Width	Depth	Width
AFS090	6	256	108 (6×18)	24,576 (4,094×6)	1
AFS250	8	256	144 (8×18)	32,768 (4,094×8)	1
AFS600	24	256	432 (24×18)	98,304 (4,096×24)	1
AFS1500	60	256	1,080 (60×18)	245,760 (4,096×60)	1

Notes:

1. *Maximum potential width uses the two-port configuration.*
2. *Maximum potential depth uses the dual-port configuration.*

Table 7-10 • **RAM and FIFO Memory Block Consumption**

			Depth									
			256		512	1,024	2,048	4,096	8,192	16,384	32,768	65,536
			Two-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port	Dual-Port
Width	1	Number Block	1	1	1	1	1	1	2	4	8	16 × 1
		Configuration	Any	Any	Any	1,024 × 4	2,048 × 2	4,096 × 1	2 × (4,096 × 1) Cascade Deep	4 × (4,096 × 1) Cascade Deep	8 × (4,096 × 1) Cascade Deep	16 × (4,096 × 1) Cascade Deep
	2	Number Block	1	1	1	1	1	2	4	8	16	32
		Configuration	Any	Any	Any	1,024×4	2,048 × 2	2 × (4,096 × 1) Cascaded Wide	4 × (4,096 × 1) Cascaded 2 Deep and 2 Wide	8 × (4,096 × 1) Cascaded 4 Deep and 2 Wide	16 × (4,096 × 1) Cascaded 8 Deep and 2 Wide	32 × (4,096 × 1) Cascaded 16 Deep and 2 Wide
	4	Number Block	1	1	1	1	2	4	8	16	32	64
		Configuration	Any	Any	Any	1,024 × 4	2 × (2,048 × 2) Cascaded Wide	4 × (4,096 × 1) Cascaded Wide	4 × (4,096 × 1) Cascaded 2 Deep and 4 Wide	16 × (4,096 × 1) Cascaded 4 Deep and 4 Wide	32 × (4,096 × 1) Cascaded 8 Deep and 4 Wide	64 × (4,096 × 1) Cascaded 16 Deep and 4 Wide
	8	Number Block	1	1	1	2	4	8	16	32	64	
		Configuration	Any	Any	Any	2 × (1,024 × 4) Cascaded Wide	4 × (2,048 × 2) Cascaded Wide	8 × (4,096 × 1) Cascaded Wide	16 × (4,096 × 1) Cascaded 2 Deep and 8 Wide	32 × (4,096 × 1) Cascaded 4 Deep and 8 Wide	64 × (4,096 × 1) Cascaded 8 Deep and 8 Wide	
	9	Number Block	1	1	1	2	4	8	16	32		
		Configuration	Any	Any	Any	2 × (512 × 9) Cascaded Deep	4 × (512 × 9) Cascaded Deep	8 × (512 × 9) Cascaded Deep	16 × (512 × 9) Cascaded Deep	32 × (512 × 9) Cascaded Deep		
	16	Number Block	1	1	1	4	8	16	32	64		
		Configuration	256 × 18	256 × 18	256 × 18	4 × (1,024 × 4) Cascaded Wide	8 × (2,048 × 2) Cascaded Wide	16 × (4,096 × 1) Cascaded Wide	32 × (4,096 × 1) Cascaded 2 Deep and 16 Wide	32 × (4,096 × 1) Cascaded 4 Deep and 16 Wide		
	18	Number Block	1	2	2	4	8	18	32			
		Configuration	256 × 8	2 × (512 × 9) Cascaded Wide	2 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded 2 Deep and 2 Wide	8 × (512 × 9) Cascaded 4 Deep and 2 Wide	16 × (512 × 9) Cascaded 8 Deep and 2 Wide	16 × (512 × 9) Cascaded 16 Deep and 2 Wide			
	32	Number Block	2	4	4	8	16	32	64			
		Configuration	2 × (256 × 18) Cascaded Wide	4 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded Wide	8 × (1,024 × 4) Cascaded Wide	16 × (2,048 × 2) Cascaded Wide	32 × (4,096 × 1) Cascaded Wide	64 × (4,096 × 1) Cascaded 2 Deep and 32 Wide			
	36	Number Block	2	4	4	8	16	32				
		Configuration	2 × (256 × 18) Cascaded Wide	4 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded Wide	4 × (512 × 9) Cascaded 2 Deep and 4 Wide	16 × (512 × 9) Cascaded 4 Deep and 4 Wide	16 × (512 × 9) Cascaded 8 Deep and 4 Wide				
	64	Number Block	4	8	8	16	32	64				
		Configuration	4 × (256 × 18) Cascaded Wide	8 × (512 × 9) Cascaded Wide	8 × (512 × 9) Cascaded Wide	16 × (1,024 × 4) Cascaded Wide	32 × (2,048 × 2) Cascaded Wide	64 × (4,096 × 1) Cascaded Wide				
	72	Number Block	4	8	8	16	32					
		Configuration	4 × (256 × 18) Cascaded Wide	8 × (512 × 9) Cascaded Wide	8 × (512 × 9) Cascaded Wide	16 × (512 × 9) Cascaded Wide	16 × (512 × 9) Cascaded 4 Deep and 8 Wide					

Note: Memory configurations represented by grayed cells are not supported.

Initializing the RAM/FIFO

The SRAM blocks can be initialized with data to use as a lookup table (LUT). Data initialization can be accomplished either by loading the data through the design logic or through the UJTAG interface. The UJTAG macro is used to allow access from the JTAG port to the internal logic in the device. By sending the appropriate initialization string to the JTAG Test Access Port (TAP) Controller, the designer can put the JTAG circuitry into a mode that allows the user to shift data into the array logic through the JTAG port using the UJTAG macro. For a more detailed explanation of the UJTAG macro, refer to *UJTAG Applications in Actel's Low-Power Flash Devices*.

A user interface is required to receive the user command, initialization data, and clock from the UJTAG macro. The interface must synchronize and load the data into the correct RAM block of the design. The main outputs of the user interface block are the following:

- Memory block chip select: Selects a memory block for initialization. The chip selects signals for each memory block that can be generated from different user-defined pockets or simple logic, such as a ring counter (see below).
- Memory block write address: Identifies the address of the memory cell that needs to be initialized.
- Memory block write data: The interface block receives the data serially from the UTDI port of the UJTAG macro and loads it in parallel into the write data ports of the memory blocks.
- Memory block write clock: Drives the WCLK of the memory block and synchronizes the write data, write address, and chip select signals.

Figure 7-8 shows the user interface between UJTAG and the memory blocks.

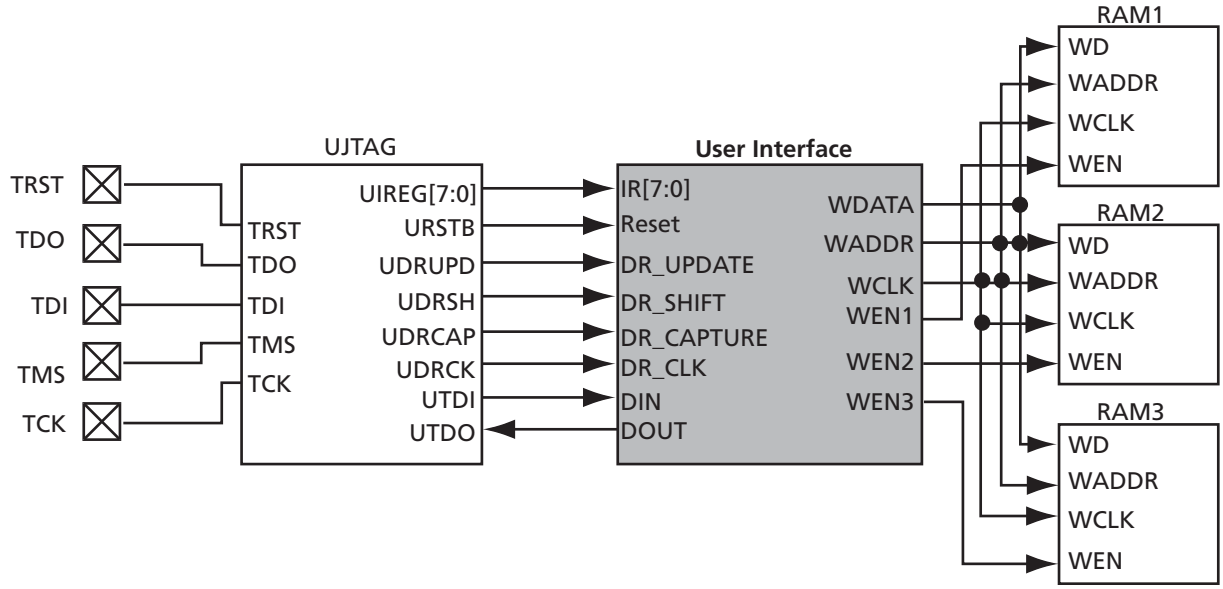


Figure 7-8 • **Interfacing TAP Ports and SRAM Blocks**

An important component of the interface between the UJTAG macro and the RAM blocks is a serial-in/parallel-out shift register. The width of the shift register should equal the data width of the RAM blocks. The RAM data arrives serially from the UTDI output of the UJTAG macro. The data must be shifted into a shift register clocked by the JTAG clock (provided at the UDRCK output of the UJTAG macro).

Then, after the shift register is fully loaded, the data must be transferred to the write data port of the RAM block. To synchronize the loading of the write data with the write address and write clock, the output of the shift register can be pipelined before driving the RAM block.

The write address can be generated in different ways. It can be imported through the TAP using a different instruction opcode and another shift register, or generated internally using a simple

counter. Using a counter to generate the address bits and sweep through the address range of the RAM blocks is recommended, since it reduces the complexity of the user interface block and the board-level JTAG driver.

Moreover, using an internal counter for address generation speeds up the initialization procedure, since the user only needs to import the data through the JTAG port.

The designer may use different methods to select among the multiple RAM blocks. Using counters along with demultiplexers is one approach to set the write enable signals. Basically, the number of RAM blocks needing initialization determines the most efficient approach. For example, if all the blocks are initialized with the same data, one enable signal is enough to activate the write procedure for all of them at the same time. Another alternative is to use different opcodes to initialize each memory block. For a small number of RAM blocks, using counters is an optimal choice. For example, a ring counter can be used to select from multiple RAM blocks. The clock driver of this counter needs to be controlled by the address generation process.

Once the addressing of one block is finished, a clock pulse is sent to the (ring) counter to select the next memory block.

Figure 7-9 illustrates a simple block diagram of an interface block between UJTAG and RAM blocks.

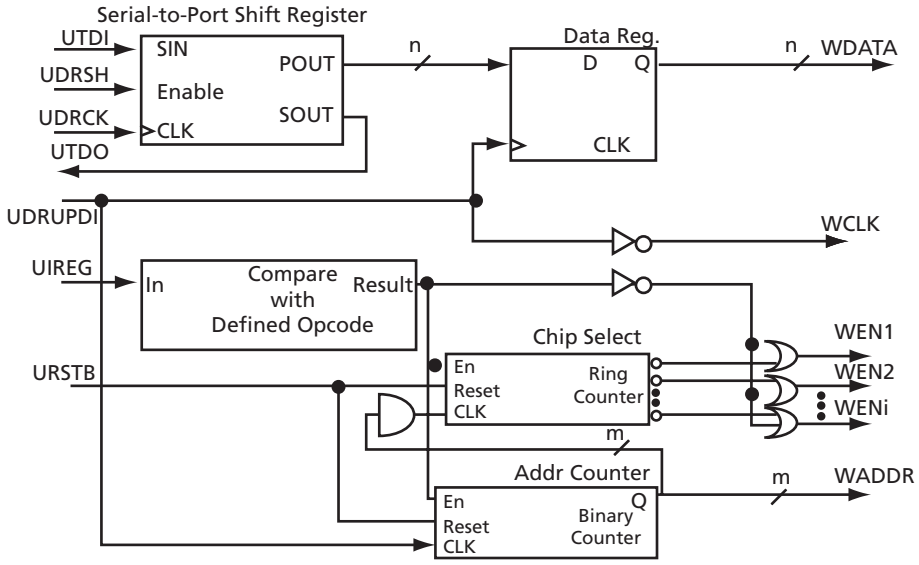


Figure 7-9 • **Block Diagram of a Sample User Interface**

In the circuit shown in Figure 7-9, the shift register is enabled by the UDRSH output of the UJTAG macro. The counters and chip select outputs are controlled by the value of the TAP Instruction Register. The comparison block compares the UIREG value with the "start initialization" opcode value (defined by the user). If the result is true, the counters start to generate addresses and activate the WEN inputs of appropriate RAM blocks.

The UDRUPD output of the UJTAG macro, also shown in Figure 7-9, is used for generating the write clock (WCLK) and synchronizing the data register and address counter with WCLK. UDRUPD is HIGH when the TAP Controller is in the Data Register Update state, which is an indication of completing the loading of one data word. Once the TAP Controller goes into the Data Register Update state, the UDRUPD output of the UJTAG macro goes HIGH. Therefore, the pipeline register and the address counter place the proper data and address on the outputs of the interface block. Meanwhile, WCLK is defined as the inverted UDRUPD. This will provide enough time (equal to the UDRUPD HIGH time) for the data and address to be placed at the proper ports of the RAM block before the rising edge of WCLK. The inverter is not required if the RAM blocks are clocked at the falling edge of the write clock. An example of this is described in the "Example of RAM Initialization" section on page 7-20.

Example of RAM Initialization

This section of the document presents a sample design in which a 4×4 RAM block is being initialized through the JTAG port. A test feature has been implemented in the design to read back the contents of the RAM after initialization to verify the procedure.

The interface block of this example performs two major functions: initialization of the RAM block and running a test procedure to read back the contents. The clock output of the interface is either the write clock (for initialization) or the read clock (for reading back the contents). The Verilog code for the interface block is included in the "Sample Verilog Code" section on page 7-21.

For simulation purposes, users can declare the input ports of the UJTAG macro for easier assignment in the testbench. However, the UJTAG input ports should not be declared on the top level during synthesis. If the input ports of the UJTAG are declared during synthesis, the synthesis tool will instantiate input buffers on these ports. The input buffers on the ports will cause Compile to fail in Designer.

Figure 7-10 shows the simulation results for the initialization step of the example design.

The CLK_OUT signal, which is the clock output of the interface block, is the inverted DR_UPDATE output of the UJTAG macro. It is clear that it gives sufficient time (while the TAP Controller is in the Data Register Update state) for the write address and data to become stable before loading them into the RAM block.

Figure 7-11 presents the test procedure of the example. The data read back from the memory block matches the written data, thus verifying the design functionality.

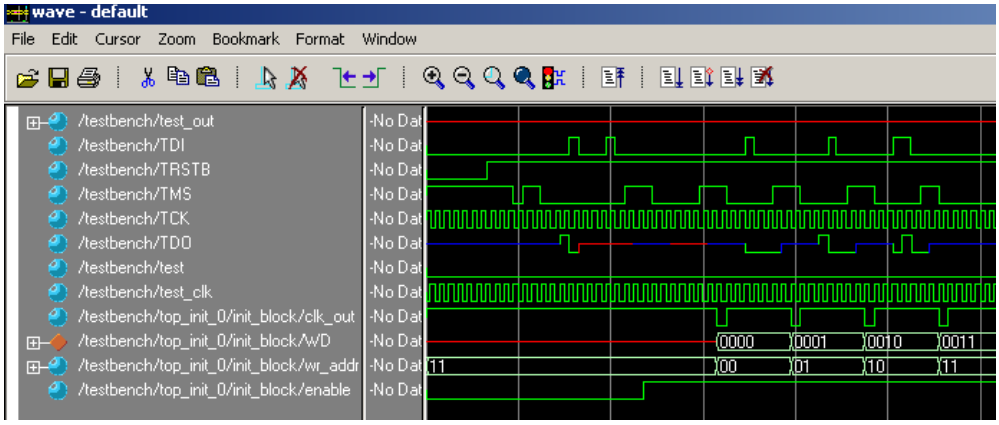


Figure 7-10 • **Simulation of Initialization Step**

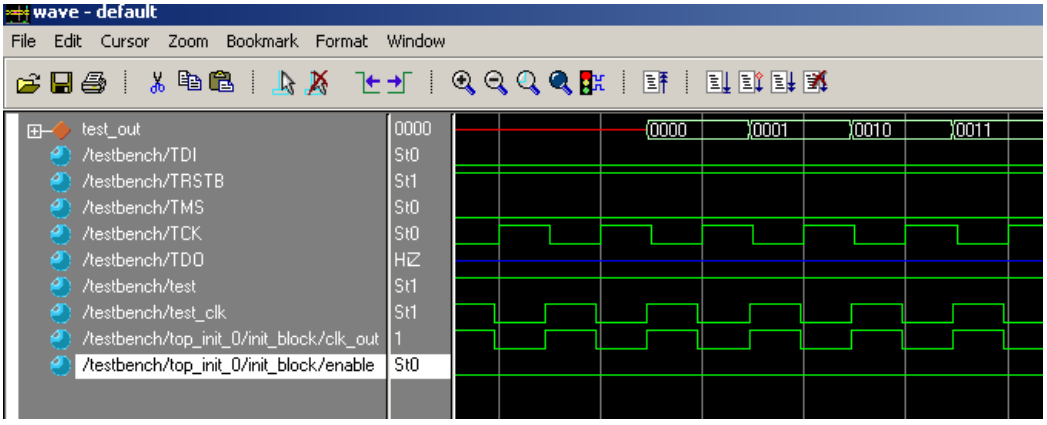


Figure 7-11 • **Simulation of the Test Procedure of the Example**

The ROM emulation application is based on RAM block initialization. If the user's main design has access only to the read ports of the RAM block (RADDR, RD, RCLK, and REN), and the contents of the RAM are already initialized through the TAP, then the memory blocks will emulate ROM functionality for the core design. In this case, the write ports of the RAM blocks are accessed only by the user interface block, and the interface is activated only by the TAP Instruction Register contents.

Users should note that the contents of the RAM blocks are lost in the absence of applied power. However, the 1 kbit of flash memory, FlashROM, in low-power flash devices can be used to retain data after power is removed from the device. Refer to *FlashROM in Actel's Low-Power Flash Devices* for more information.

Sample Verilog Code

Interface Block

```
`define Initialize_start 8'h22 //INITIALIZATION START COMMAND VALUE
`define Initialize_stop 8'h23 //INITIALIZATION START COMMAND VALUE

module interface(IR, rst_n, data_shift, clk_in, data_update, din_ser, dout_ser, test,
  test_out,test_clk,clk_out,wr_en,rd_en,write_word,read_word,rd_addr, wr_addr);

input [7:0] IR;
input [3:0] read_word; //RAM DATA READ BACK
input rst_n, data_shift, clk_in, data_update, din_ser; //INITIALIZATION SIGNALS
input test, test_clk; //TEST PROCEDURE CLOCK AND COMMAND INPUT
output [3:0] test_out; //READ DATA
output [3:0] write_word; //WRITE DATA
output [1:0] rd_addr; //READ ADDRESS
output [1:0] wr_addr; //WRITE ADDRESS
output dout_ser; //TDO DRIVER
output clk_out, wr_en, rd_en;

wire [3:0] write_word;
wire [1:0] rd_addr;
wire [1:0] wr_addr;
wire [3:0] Q_out;
wire enable, test_active;

reg clk_out;

//SELECT CLOCK FOR INITIALIZATION OR READBACK TEST
always @(enable or test_clk or data_update)
begin
  case ({test_active})
    1 : clk_out = test_clk ;
    0 : clk_out = !data_update;
    default : clk_out = 1'b1;
  endcase
end

assign test_active = test && (IR == 8'h23);
assign enable = (IR == 8'h22);
assign wr_en = !enable;
assign rd_en = !test_active;
assign test_out = read_word;
assign dout_ser = Q_out[3];

//4-bit SIN/POUT SHIFT REGISTER
shift_reg data_shift_reg (.Shiften(data_shift), .Shiftin(din_ser), .Clock(clk_in),
  .Q(Q_out));

//4-bit PIPELINE REGISTER
D_pipeline pipeline_reg (.Data(Q_out), .Clock(data_update), .Q(write_word));
```

```
//
addr_counter counter_1 (.Clock(data_update), .Q(wr_addr), .Aset(rst_n),
  .Enable(enable));
addr_counter counter_2 (.Clock(test_clk), .Q(rd_addr), .Aset(rst_n),
  .Enable( test_active));

endmodule
```

Interface Block / UJTAG Wrapper

This example is a sample wrapper, which connects the interface block to the UJTAG and the memory blocks.

```
// WRAPPER
module top_init (TDI, TRSTB, TMS, TCK, TDO, test, test_clk, test_ out);

input TDI, TRSTB, TMS, TCK;
output TDO;
input test, test_clk;
output [3:0] test_out;

wire [7:0] IR;
wire reset, DR_shift, DR_cap, init_clk, DR_update, data_in, data_out;
wire clk_out, wen, ren;
wire [3:0] word_in, word_out;
wire [1:0] write_addr, read_addr;

UJTAG UJTAG_U1 (.UIREG0(IR[0]), .UIREG1(IR[1]), .UIREG2(IR[2]), .UIREG3(IR[3]),
  .UIREG4(IR[4]), .UIREG5(IR[5]), .UIREG6(IR[6]), .UIREG7(IR[7]), .URSTB(reset),
  .UDRSH(DR_shift), .UDRCAP(DR_cap), .UDRCK(init_clk), .UDRUPD(DR_update),
  .UT-DI(data_in), .TDI(TDI), .TMS(TMS), .TCK(TCK), .TRSTB(TRSTB), .TDO(TDO),
  .UT-DO(data_out));
mem_block RAM_block (.DO(word_out), .RCLOCK(clk_out), .WCLOCK(clk_out), .DI(word_in),
  .WRB(wen), .RDB(ren), .WAD-DR(write_addr), .RADDR(read_addr));
interface init_block (.IR(IR), .rst_n(reset), .data_shift(DR_shift), .clk_in(init_clk),
  .data_update(DR_update), .din_ser(data_in), .dout_ser(data_out), .test(test),
  .test_out(test_out), .test_clk(test_clk), .clk_out(clk_out), .wr_en(wen),
  .rd_en(ren), .write_word(word_in), .read_word(word_out), .rd_addr(read_addr),
  .wr_addr(write_addr));

endmodule
```

Address Counter

```
module addr_counter (Clock, Q, Aset, Enable);

input Clock;
output [1:0] Q;
input Aset;
input Enable;

reg [1:0] Qaux;

always @(posedge Clock or negedge Aset)
begin
  if (!Aset) Qaux <= 2'b11;
  else if (Enable) Qaux <= Qaux + 1;
end

assign Q = Qaux;

endmodule
```

Pipeline Register

```
module D_pipeline (Data, Clock, Q);

input [3:0] Data;
input Clock;
output [3:0] Q;

reg [3:0] Q;

always @ (posedge Clock) Q <= Data;

endmodule
```

4x4 RAM Block (created by SmartGen Core Generator)

```
module mem_block(DI,DO,WADDR,RADDR,WRB,RDB,WCLOCK,RCLOCK);

input [3:0] DI;
output [3:0] DO;
input [1:0] WADDR, RADDR;
input WRB, RDB, WCLOCK, RCLOCK;

wire WEBP, WEAP, VCC, GND;

VCC VCC_1_net(.Y(VCC));
GND GND_1_net(.Y(GND));
INV WEBUBBLEB(.A(WRB), .Y(WEBP));
RAM4K9 RAMBLOCK0(.ADDRA11(GND), .ADDRA10(GND), .ADDRA9(GND), .ADDRA8(GND),
  .ADDRA7(GND), .ADDRA6(GND), .ADDRA5(GND), .ADDRA4(GND), .ADDRA3(GND), .ADDRA2(GND),
  .ADDRA1(RADDR[1]), .ADDRA0(RADDR[0]), .ADDRB11(GND), .ADDRB10(GND), .ADDRB9(GND),
  .ADDRB8(GND), .ADDRB7(GND), .ADDRB6(GND), .ADDRB5(GND), .ADDRB4(GND), .ADDRB3(GND),
  .ADDRB2(GND), .ADDRB1(WADDR[1]), .ADDRB0(WADDR[0]), .DINA8(GND), .DINA7(GND),
  .DINA6(GND), .DINA5(GND), .DINA4(GND), .DINA3(GND), .DINA2(GND), .DINA1(GND),
  .DINA0(GND), .DINB8(GND), .DINB7(GND), .DINB6(GND), .DINB5(GND), .DINB4(GND),
  .DINB3(DI[3]), .DINB2(DI[2]), .DINB1(DI[1]), .DINB0(DI[0]), .WIDTHA0(GND),
  .WIDTHA1(VCC), .WIDTHB0(GND), .WIDTHB1(VCC), .PIPEA(GND), .PIPEB(GND),
  .WMODEA(GND), .WMODEB(GND), .BLKA(WEAP), .BLKB(WEBP), .WENA(VCC), .WENB(GND),
  .CLKA(RCLOCK), .CLKB(WCLOCK), .RESET(VCC), .DOUTA8(), .DOUTA7(), .DOUTA6(),
  .DOUTA5(), .DOUTA4(), .DOUTA3(DO[3]), .DOUTA2(DO[2]), .DOUTA1(DO[1]),
  .DOUTA0(DO[0]), .DOUTB8(), .DOUTB7(), .DOUTB6(), .DOUTB5(), .DOUTB4(), .DOUTB3(),
  .DOUTB2(), .DOUTB1(), .DOUTB0());
INV WEBUBBLEA(.A(RDB), .Y(WEAP));

endmodule
```

Software Support

The SmartGen core generator is the easiest way to select and configure the memory blocks (Figure 7-12). SmartGen automatically selects the proper memory block type and aspect ratio, and cascades the memory blocks based on the user's selection. SmartGen also configures any additional signals that may require tie-off.

SmartGen will attempt to use the minimum number of blocks required to implement the desired memory. When cascading, SmartGen will configure the memory for width before configuring for depth. For example, if the user requests a 256×8 FIFO, SmartGen will use a 512×9 FIFO configuration, not 256×18.

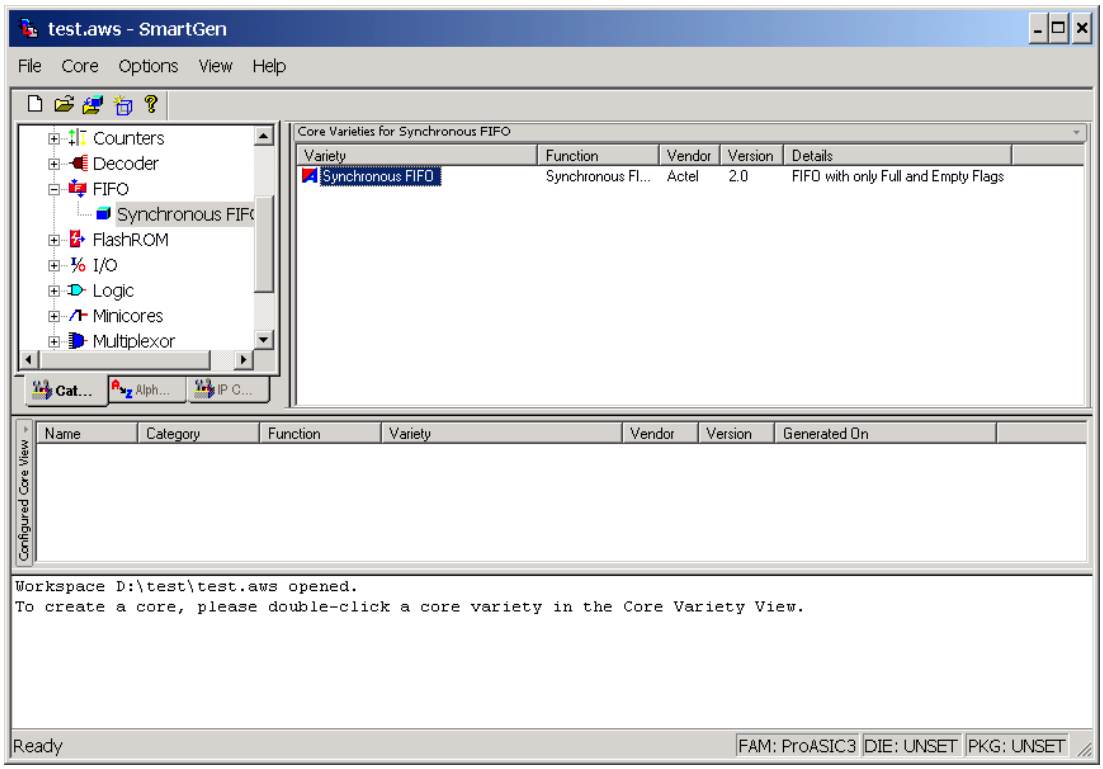


Figure 7-12 • **SmartGen Core Generator Interface**

SmartGen enables the user to configure the desired RAM element to use either a single clock for read and write, or two independent clocks for read and write. The user can select the type of RAM as well as the width/depth and several other parameters (Figure 7-13).

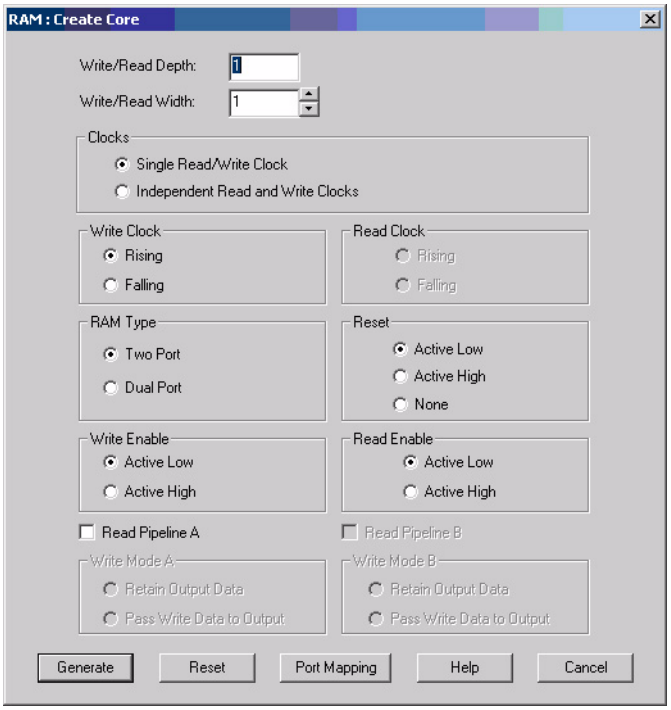


***Figure 7-13* • SmartGen Memory Configuration Interface**

SmartGen also has a Port Mapping option that allows the user to specify the names of the ports generated in the memory block (Figure 7-14).

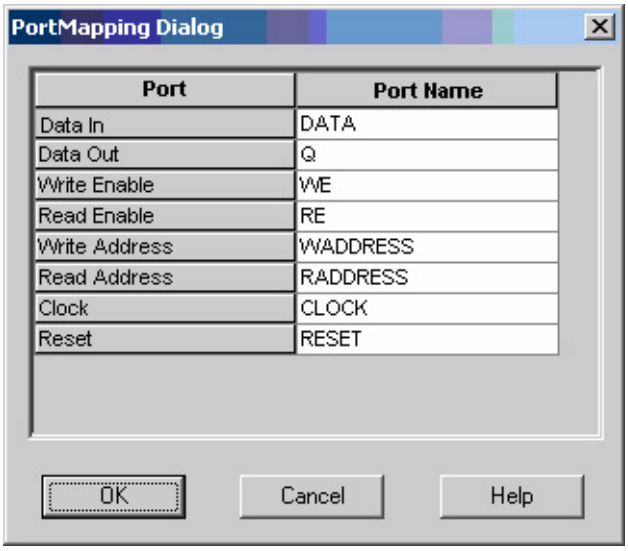


***Figure 7-14* • Port Mapping Interface for SmartGen-Generated Memory**

SmartGen also configures the FIFO according to user specifications. Users can select no flags, static flags, or dynamic flags. Static flag settings are configured using configuration flash and cannot be altered without reprogramming the device. Dynamic flag settings are determined by register values and can be altered without reprogramming the device by reloading the register values either from the design or through the UJTAG interface described in the "Initializing the RAM/FIFO" section on page 7-18.

SmartGen can also configure the FIFO to continue counting after the FIFO is full. In this configuration, the FIFO write counter will wrap after the counter is full and continue to write data. With the FIFO configured to continue to read after the FIFO is empty, the read counter will also wrap and re-read data that was previously read. This mode can be used to continually read back repeating data patterns stored in the FIFO (Figure 7-15).

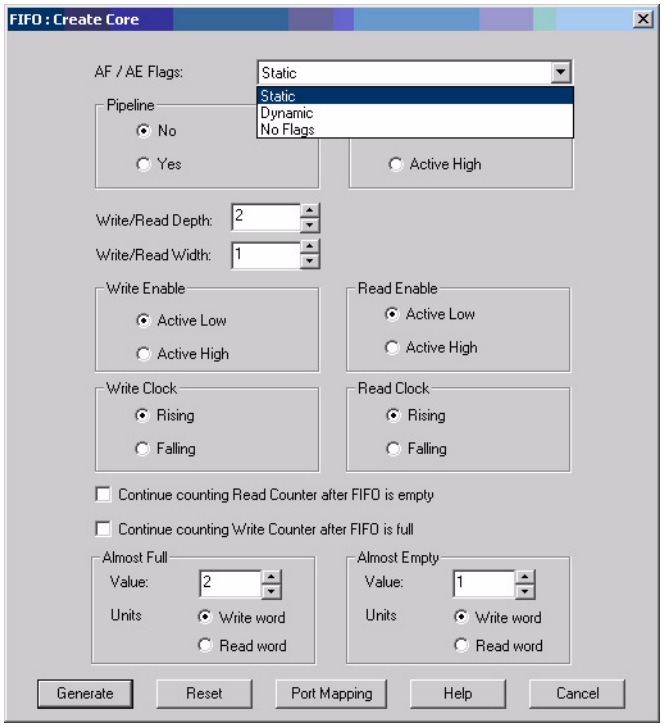


Figure 7-15 • **SmartGen FIFO Configuration Interface**

FIFOs configured using SmartGen can also make use of the port mapping feature to configure the names of the ports.

Limitations

Users should be aware of the following limitations when configuring SRAM blocks for low-power flash devices:

- SmartGen does not track the target device in a family, so it cannot determine if a configured memory block will fit in the target device.
- Dual-port RAMs with different read and write aspect ratios are not supported.
- Cascaded memory blocks can only use a maximum of 64 blocks of RAM.
- The Full flag of the FIFO is sensitive to the maximum depth of the actual physical FIFO block, not the depth requested in the SmartGen interface.

Conclusion

Fusion, IGLOO, and ProASIC3 devices provide users with extremely flexible SRAM blocks for most design needs, with the ability to choose between an easy-to-use dual-port memory or a wide-word two-port memory. Used with the built-in FIFO controllers, these memory blocks also serve as highly efficient FIFOs that do not consume user gates when implemented. The Actel SmartGen core generator provides a fast and easy way to configure these memory elements for use in designs.

Related Documents

Handbook Documents

UJTAG Applications in Actel's Low-Power Flash Devices

www.actel.com/documents/LPD_UJTAG_HBs.pdf

FlashROM in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_FlashROM_HBs.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-008-5

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in the Current Version (v1.5)	Page
v1.4 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 7-1 · Flash-Based FPGAs.	7-4
	IGLOO nano and ProASIC3 nano devices were added to Table 7-8 · Interfacing TAP Ports and SRAM Blocks.	7-18
v1.3 (August 2008)	The "SRAM/FIFO Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	7-4
	The "SRAM and FIFO Architecture" section was modified to remove "IGLOO and ProASIC3E" from the description of what the memory block includes, as this statement applies to all memory blocks.	7-5
	Wording in the "Clocking" section was revised to change "IGLOO and ProASIC3 devices support inversion" to "Low-power flash devices support inversion." The reference to IGLOO and ProASIC3 development tools in the last paragraph of the section was changed to refer to development tools in general.	7-11
	The "ESTOP and FSTOP Usage" section was updated to refer to FIFO counters in devices in general rather than only IGLOO and ProASIC3E devices.	7-15
v1.2 (June 2008)	The note was removed from Figure 7-7 · RAM Block with Embedded FIFO Controller and placed in the WCLK and RCLK description.	7-12
	The "WCLK and RCLK" description was revised.	7-13
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 7-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	7-4
v1.0 (January 2008)	The "Introduction" section was updated to include the IGLOO PLUS family.	7-1
	The "Device Architecture" section was updated to state that 15 k gate devices do not support SRAM and FIFO.	7-1
	The first note in Figure 7-1 · IGLOO and ProASIC3 Device Architecture Overview was updated to include mention of 15 k gate devices, and IGLOO PLUS was added to the second note.	7-3
	Table 7-1 · Flash-Based FPGAs and associated text were updated to include the IGLOO PLUS family. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	7-4
	The text introducing Table 7-8 · Memory Availability per IGLOO and ProASIC3 Device was updated to replace "A3P030 and AGL030" with "15 k and 30 k gate devices." Table 7-8 · Memory Availability per IGLOO and ProASIC3 Device was updated to remove AGL400 and AGLE1500 and include IGLOO PLUS and ProASIC3L devices.	7-16

Analog System

Actel
POWER MATTERS

8 – Designing the Fusion Analog System

Introduction

Actel Fusion® devices offer robust and flexible analog mixed-signal capability in addition to the high-performance flash FPGA fabric and flash memory block. The many built-in analog peripherals include a configurable 32:1 analog MUX, up to 10 independent MOSFET gate driver outputs, and a configurable analog-to-digital converter (ADC). Fusion also introduces the Analog Quad I/O structure; each Analog Quad consists of three analog inputs and one gate driver. Each Quad can be configured in various built-in circuit combinations, such as prescaler circuits, three-digital-input circuits, a current monitor circuit, or a temperature monitor circuit. Each prescaler has multiple scaling factors programmed by FPGA signals to support a large range of analog inputs with positive or negative polarity. When the current monitor circuit is selected, two adjacent analog inputs measure the voltage drop across a small external sense resistor. Built-in operational amplifiers amplify small voltage signals for accurate current measurement. One analog input in each Quad can be connected to an external temperature monitor diode. These components are used as the building blocks in designing an analog system.

The Analog Quad I/O configuration, ADC resolution, channel sampling sequence, and sampling rate are programmable and implemented in the FPGA logic using Designer and Actel Libero® Integrated Design Environment (IDE) software tool support. An overview of different design methodologies is covered in the *Fusion Design Solutions and Methodologies* chapter. This chapter gives a detailed description of the Analog Quads, ADC, and Analog Configuration MUX (ACM). It also covers the details of the analog system with the explanation and sample calculations of accuracy, sample rate, sample sequencing, acquisition time, ADC clocking, and prescaler selection.

Analog-to-Digital Converter Background

An analog-to-digital converter is used to capture discrete samples of a continuous analog voltage and provide a discrete binary representation of the signal.

Analog-to-digital converters are generally characterized in three ways:

- Input voltage range
- Resolution
- Bandwidth or conversion rate

The input voltage range of an ADC is determined by its reference voltage (V_{REF}). Actel Fusion™ devices include an internal 2.56 V reference, or the user can supply an external reference of up to 3.3 V. The following examples use the internal 2.56 V reference, so the full-scale input range of the ADC is 0 to 2.56 V. For input signal ranges less than or greater than V_{REF}, an analog scaling function such as that built into the Fusion Analog Quad Prescaler can be used to amplify or attenuate the input signal, thus matching the input voltage range of the ADC.

The resolution (LSB) of the ADC is a function of the number of binary bits in the converter. The ADC approximates the value of the input voltage using 2^n "steps," where *n* is the number of bits in the converter. Each step therefore represents $V_{REF} / 2^n$ volts. In the case of the Fusion ADC configured for 12-bit operation, the LSB is 2.56 V / 4096 = 0.625 mV.

Finally, bandwidth is an indication of the maximum number of conversions the ADC can perform each second. The bandwidth of an ADC is constrained by its architecture and several key performance characteristics. In addition, the bandwidth is limited by Fusion system considerations. See the "Sample Rate and Sample Sequence Calculation" section on page 8-8.

There are several popular ADC architectures, each with its own advantages and limitations. The analog-to-digital converter in Fusion devices is a switched-capacitor Successive Approximation Register (SAR) ADC. It supports 8-, 10-, and 12-bit modes of operation with a cumulative sample rate up to 600 k samples per second (ksps). Built-in bandgap circuitry offers 1% internal voltage reference accuracy, or an external reference voltage can be used.

As shown in Figure 8-1, a SAR ADC contains N capacitors with binary-weighted values.

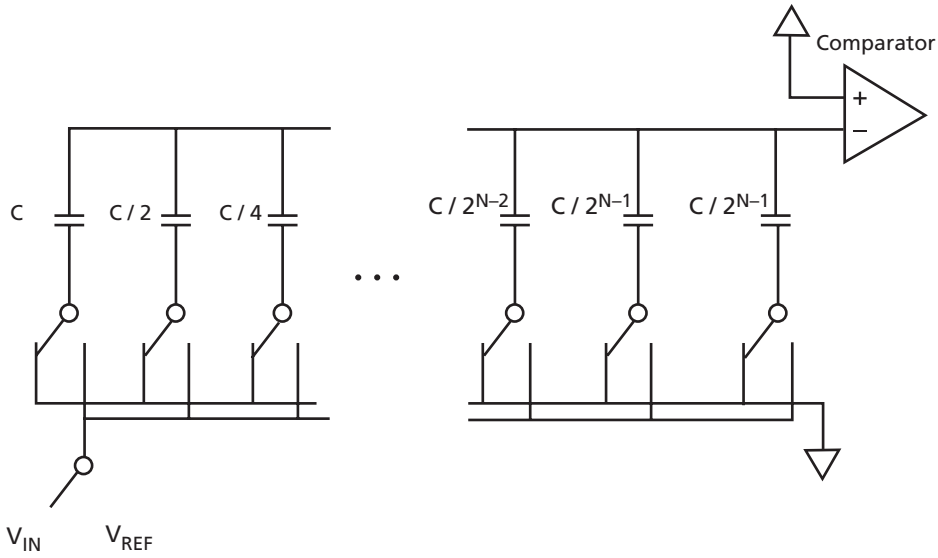


Figure 8-1 • **Example SAR ADC Architecture**

To begin a conversion, all of the capacitors are quickly discharged. Then V_{IN} is applied to all the capacitors for a period of time (acquisition time) during which the capacitors are charged to a value very close to V_{IN}. Then all of the capacitors are switched to ground, and thus $–V_{IN}$ is applied across the comparator.

Now the conversion process begins. First, C is switched to V_{REF}. Because of the binary weighting of the capacitors, the voltage at the input of the comparator is then $–V_{IN} + V_{REF} / 2$. If V_{IN} is greater than $V_{REF} / 2$, the output of the comparator is 1; otherwise, the comparator output is 0. A register is clocked to retain this value as the MSB of the result.

Next, if the MSB is 0, C is switched back to ground; otherwise, it remains connected to V_{REF}, and C / 2 is connected to V_{REF}. The result at the comparator input is now either $–V_{IN} + V_{REF} / 4$ or $–V_{IN} + 3 V_{REF} / 4$ (depending on the state of the MSB), and the comparator output now indicates the value of the next most significant bit. This bit is likewise registered, and the process continues for each subsequent bit until a conversion is completed. The conversion process requires some acquisition time plus N + 1 ADC clock cycles to complete.

This process results in a binary approximation of V_{IN}. Generally, there is a fixed interval T, the sampling period, between the samples. The inverse of the sampling period is often referred to as the sampling frequency $f_s = 1 / T$. The combined effect is illustrated in Figure 8-2.

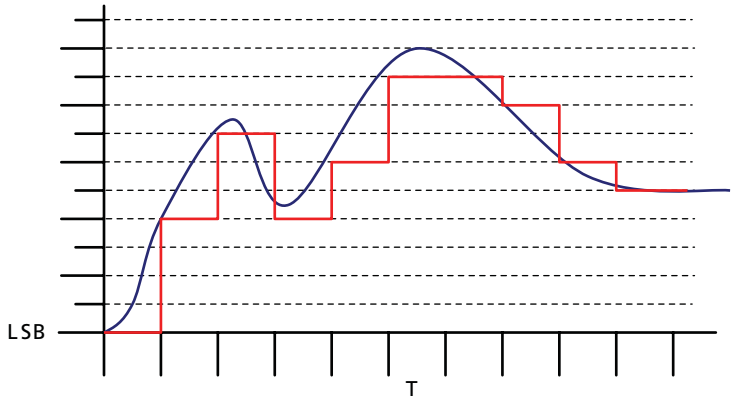


Figure 8-2 • **Analog-to-Digital Conversion Example**

Figure 8-2 demonstrates that if the signal changes faster than the sampling rate can accommodate, or if the actual value of V_{IN} falls between "counts" in the result, this information is lost during the conversion. There are several techniques that can be used to address these issues.

First, the sampling rate must be chosen to provide enough samples to adequately represent the input signal. Based on the Nyquist-Shannon Sampling Theorem, the minimum sampling rate must be at least twice the frequency of the highest frequency component in the target signal (Nyquist Frequency). For example, to re-create the frequency content of an audio signal with up to 22 kHz bandwidth, the user must sample it at a minimum of 44 ksps. However, as shown in Figure 8-2, significant post-processing of the data is required to interpolate the value of the waveform during the time between each sample.

Similarly, to re-create the amplitude variation of a signal, the signal must be sampled with adequate resolution. Continuing with the audio example, the dynamic range of the human ear (the ratio of the amplitude of the threshold of hearing to the threshold of pain) is generally accepted to be 135 dB, and the dynamic range of a typical symphony orchestra performance is around 85 dB. Most commercial recording media provide about 96 dB of dynamic range using 16-bit sample resolution. But 16-bit fidelity does not necessarily mean that you need a 16-bit ADC. As long as the input is sampled at or above the Nyquist Frequency, post-processing techniques can be used to interpolate intermediate values and reconstruct the original input signal to within desired tolerances.

If sophisticated digital signal processing (DSP) capabilities are available, the best results are obtained by implementing a reconstruction filter, which is used to interpolate many intermediate values with higher resolution than the original data. Interpolating many intermediate values, increases the effective number of samples, and higher resolution increases the effective number of bits in the sample. In many cases, however, it is not cost-effective or necessary to implement such a sophisticated reconstruction algorithm. For applications that do not require extremely fine reproduction of the input signal, alternative methods can enhance digital sampling results with relatively simple post-processing. The details of such techniques are out of the scope of this chapter; refer to the *Improving ADC Results Through Oversampling and Post-Processing of Data* white paper for more information.

ADC Clock

When the Fusion ADC is used, the ADC clock determines the sampling throughput, and the system clock determines the operating speed of the SmartGen IP and/or the user's design logic. This section examines the relationship between these two clocks and how the sampling rate is related to the accuracy. A simplified block diagram of the ADC is given in Figure 8-3.

The ADC clock has a maximum frequency of 10 MHz and can be derived from the system clock. To generate the ADC clock, the system clock is divided by a multiple of four. The exact multiple of four used is determined by the 8-bit user-configurable TVC[7:0] register (EQ 8-1).

$$\text{ADC Clock Frequency (MHz)} = \text{System Clock (MHz)} / (4 \times (\text{TVC_reg} + 1))$$

EQ 8-1

where TVC_reg is the TVC register value, from 0 to 255.

The TVC register setting is used to ensure that the ADC clock frequency does not exceed 10 MHz. Note that the 10 MHz maximum frequency for the ADC clock implies that a higher system clock frequency does not always result in a higher ADC clock frequency. For example, a 40 MHz system clock frequency enables a maximum ADC clock frequency of 10 MHz (TVC register value of 0), whereas a 50 MHz system frequency results in a slower maximum ADC clock frequency, 6.25 MHz, because a TVC register value of 0 would give an ADC clock frequency of 12.5 MHz—above the 10 MHz limit. Note that this 10 MHz limit means that a higher system clock frequency does not always result in a higher ADC clock frequency. For example, a 40 MHz system clock frequency enables a maximum ADC clock frequency of 10 MHz (TVC register value of 0). However, a 50 MHz system frequency results in a slower maximum ADC clock frequency because a TVC register value of 0 would give an ADC clock frequency of 12.5 MHz—above the 10 MHz limit. Setting the TVC register value to 1 in this case gives a maximum ADC clock frequency of 6.25 MHz.

In general, the performance of the ADC is the rate at which the ADC can acquire or sample the analog input and convert it into a digital value. Datasheet specifications define this in terms of samples per second (S/sec) or hertz (Hz). The inverse of the conversion time is the sampling rate for the channel. However, the sampling rate reported by SmartGen includes the ADC Sample Sequence Controller (ASSC) overhead time. This time, the turnaround time, defines how fast an ADC client can process data and give another start conversion signal. With no wait states, the ASSC turnaround time is 10 system clock cycles.

$$\text{Sample Rate} = \frac{1}{\text{Conversion Time} + \text{Turnaround Time}} (\text{S/sec})$$

EQ 8-2

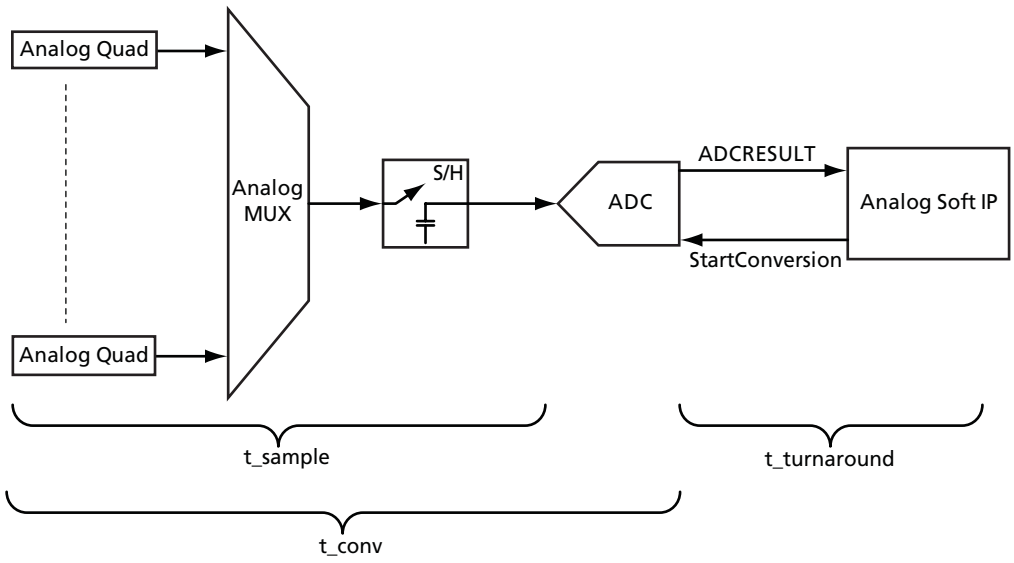


Figure 8-3 • **Simplified ADC Diagram**

The conversion time (t_conv) is the total time required to convert an analog input signal into a digital output (EQ 8-3).

t_conv = t_sync_read + t_sample + t_distrib + t_post_cal + t_sync_write

EQ 8-3

The components of EQ 8-3 are defined in Table 8-1.

Table 8-1 • **ADC Conversion Time Formula Elements**

Equation	Description
t_sync_read = sys_clk_period	• Time to latch the input data • sys_clk_period is the ADC interface clock (10 ns to 250 ns).
t_sample = (2 + STC) × adc_clock_period	• STC is the Sample Time Control in the SmartGen GUI. This changes the acquisition time t_sample (sample-and-hold time). STC[7:0] ranges from 0 to 255. • adc_clock_period is the ADC internal clock period (100 ns to 2 µs).
t_distrib = resolution × adc_clock_period	• Time of charge redistribution • adc_clock_period is the ADC internal clock period (100 ns to 2 µs). • Selectable 8-/10-/12-bit resolution mode
t_post_cal = 2 × adc_clock_period	• Time for post-calibration • adc_clock_period is the ADC internal clock period.
t_sync_write = sys_clk_period	• Time for latching the output data • sys_clk_period is the ADC interface clock period (10 ns to 250 ns).

Example 1

Given that only one channel is used without prescaler, the maximum sample rate of a channel can be calculated as follows:

System Clock Period = 1 / (40 MHz) = 25 ns

ADC Clock Period = 1 / (10 MHz) = 100 ns

Acquisition Time = t_sample = 0.4 µs

Resolution = 8

t_conv = t_sync_read + t_sample + t_distrib + t_post_cal + t_sync_write

t_conv = 25 ns + 400 ns + 8 × 100 ns + 2 × 100 ns + 25 ns = 1.45 µs

t_turnaround = 10 × sys_clk = 250 ns

$$\text{Sample Rate} = \frac{1}{1.45\ \mu s + 0.25\ \mu s}\text{ns} = 588\ \text{ksps}$$

Note: To avoid using the prescaler, the maximum voltage value must be set between 2.01 V and 2.56 V in the SmartGen GUI to configure the peripheral as a direct input.

Example 2

Given that only one channel is used with the prescaler, the maximum sample rate of a channel can be calculated as follows:

System Clock Period = 1 / (80 MHz) = 12.5 ns

ADC Clock Period = 1 / (10 MHz) = 100 ns

Acquisition Time = Settling time = 10 µs (max.)

Resolution = 8

t_conv = t_sync_read + t_sample + t_distrib + t_post_cal + t_sync_write

t_conv = 12.5 ns + 10000 ns + 8 × 100 ns + 2 × 100 ns + 12.5 ns = 11.025 µs

Add turnaround time: 11.025 µs + 0.125 µs = 11.15 µs

$$\text{Sample Rate} = \frac{1}{11.025\ \mu s + 0.125\ \mu s}\text{ns} = 89.68\ \text{ksps}$$

Example 3

Given that only one channel is used with the prescaler, the maximum sample rate of a channel can be calculated as follows:

System Clock Period = 1 / (40 MHz) = 25 ns

ADC Clock = 1 / (10 MHz) = 100 ns

Acquisition Time = Settling time = 10 µs (max.)

Resolution = 8

t_conv = t_sync_read + t_sample + t_distrib + t_post_cal + t_sync_write

t_conv = 25 ns + 10000 ns + 8 × 100 ns + 2 × 100 ns + 25 ns = 11.05 µs

Add turnaround time: 11.05 µs + 0.25 µs = 11.3 µs

$$\text{Sample Rate} = \frac{1}{11.05\ \mu s + 0.25\ \mu s}\text{ns} = 88.49\ \text{ksps}$$

Note that when the prescaler is used, a 10 µs settling/acquisition time is recommended for increased accuracy. SmartGen automatically computes values for the STC, clock divider setting (TVC), and ADC clock period. The goal of SmartGen is to meet the minimum sample time requirement with the highest possible ADC clock frequency, which implies a low TVC value and high STC value.

Sample Sequencing Overview

As described in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet and illustrated in Figure 8-4, there is one ADC in the Analog Block (AB) and up to ten Analog Quads, with three analog inputs each: AV, AC, and AT. The analog input to ADC is selected through a MUX architecture controlled by the CHNUMBER select input. FPGA fabric access to the CHNUMBER input of the AB provides users the flexibility to define custom sample sequencing among the Analog Quads.

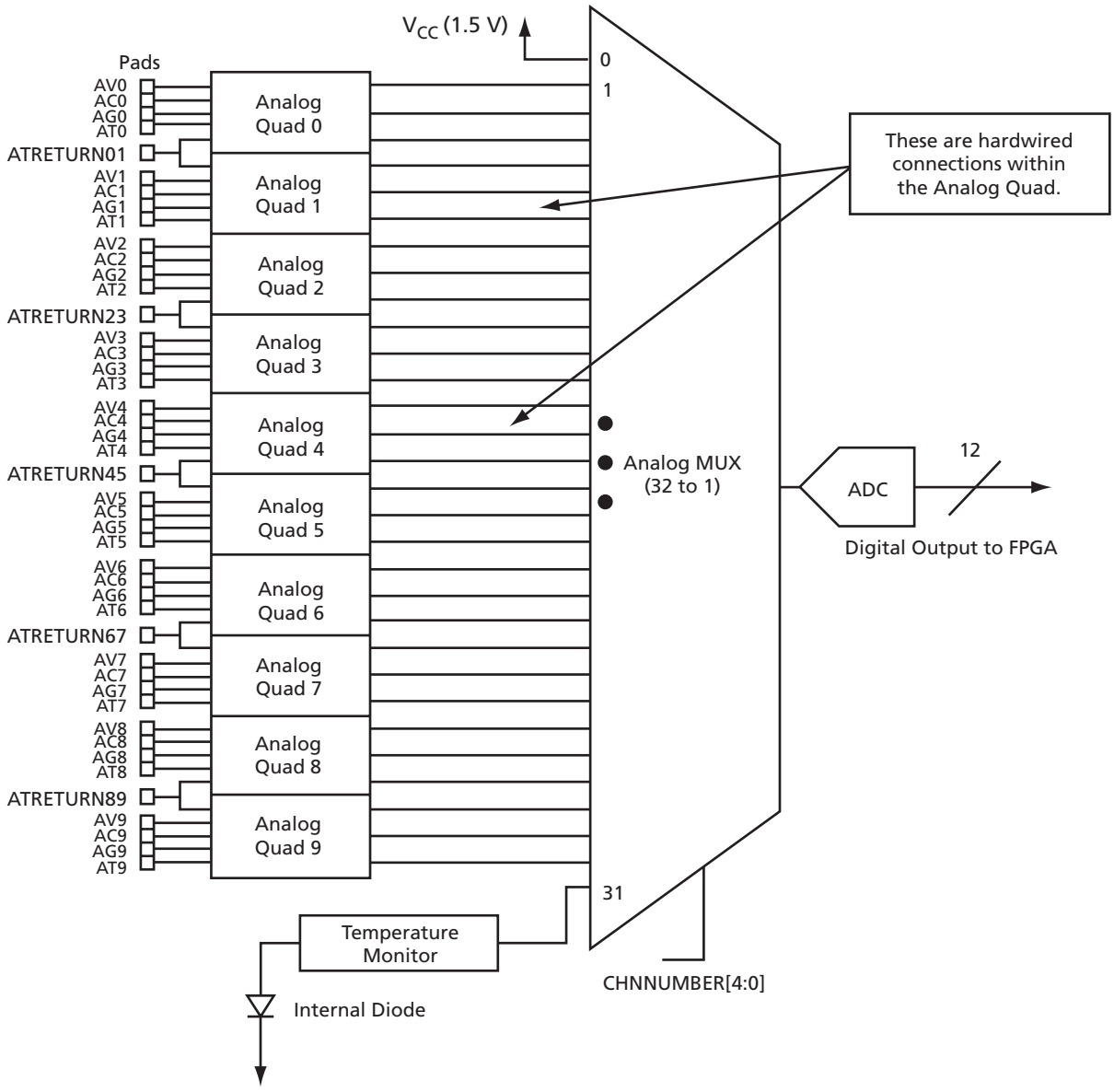


Figure 8-4 • **Analog Block ADC and MUX Architecture**

The flexibility of sample sequencing in the Fusion AB architecture enables conditional sequences and control of the sampling rate of each channel. For example, the designer can sample critical inputs (requiring a higher sampling rate) more often than non-critical inputs. It is also feasible for the design to change sampling sequence and/or rate of analog inputs during operation whenever required.

There is no automatic internal sequencing in the architecture shown in Figure 8-4 on page 8-7. Therefore, the CHNUMBER input of the MUX must be controlled and defined by the user's design at all points throughout its implementation (e.g., Smartgen IP, CoreAI (Analog Interface) register space, and custom logic).

Sample Rate and Sample Sequence Calculation

As the Fusion ADC can be shared among different channels (32 channels in all), the sample rate can be calculated based on the system sampling rate or per-channel sampling rate (EQ 8-4 and EQ 8-5).

$$\text{System Sampling Rate} = \frac{\text{Total \# of Samples}}{\text{Total (conversion + turnaround) Time of All Samples}} \quad \textit{EQ 8-4}$$

$$\text{Channel Sampling Rate} = \frac{\text{Total \# of Samples for a Channel}}{\text{Total \# All Samples}} \times \text{System Sampling Rate} \quad \textit{EQ 8-5}$$

Example: Equal Weight and Equal Conversion Time

Each channel has a conversion time of 2 µs, as shown in Figure 8-5.

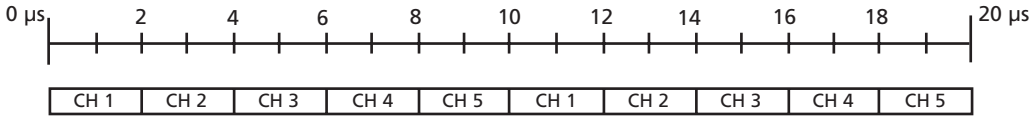


Figure 8-5 • **Equal Weight and Equal Conversion Time**

In this case, there are 10 samples, which take a total of 20 µs. Thus, the total system sampling rate is 10 / (20 µs) = 500 ksps.

Channel 1 sampling rate: (2 / 10) × 500 ksps = 100 ksps

Channel 2 sampling rate: (2 / 10) × 500 ksps = 100 ksps

Channel 3 sampling rate: (2 / 10) × 500 ksps = 100 ksps

Channel 4 sampling rate: (2 / 10) × 500 ksps = 100 ksps

Channel 5 sampling rate: (2 / 10) × 500 ksps = 100 ksps

Example: Unequal Weight and Equal Conversion Time

In this example, the channels are not equally weighted in the sampling sequences shown in Figure 8-6.

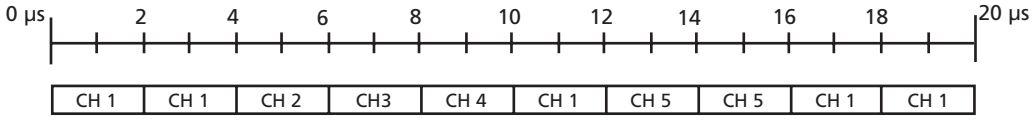


Figure 8-6 • **Unequal Weight and Equal Conversion Time**

In this case, there are 10 samples that take a total of 20 µs, giving a total system sampling rate of 500 ksps (as above). However, the individual channel sampling rates are different.

Channel 1 sampling rate: (5 / 10) × 500 ksps = 250 ksps

Channel 2 sampling rate: (1 / 10) × 500 ksps = 50 ksps

Channel 3 sampling rate: (1 / 10) × 500 ksps = 50 ksps

Channel 4 sampling rate: (1 / 10) × 500 ksps = 50 ksps

Channel 5 sampling rate: (2 / 10) × 500 ksps = 100 ksps

Example: Unequal Weight and Unequal Conversion Time

In this example, channels have different conversion times and are not equally weighted in the sampling sequence, as shown in Figure 8-7.

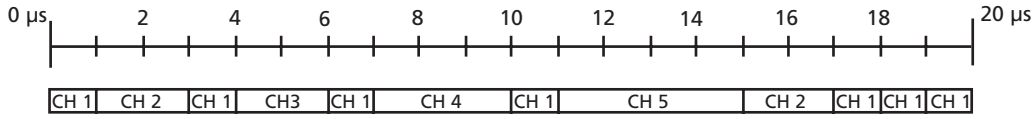


Figure 8-7 • **Unequal Weight and Unequal Conversion Time**

In this case, there are 12 samples in 20 µs, giving a total system sampling rate of 600 ksps.

Channel 1 sampling rate: (7 / 12) × 600 ksps = 349 ksps

Channel 2 sampling rate: (2 / 12) × 600 ksps = 99.6 ksps

Channel 3 sampling rate: (1 / 12) × 600 ksps = 49.8 ksps

Channel 4 sampling rate: (1 / 12) × 600 ksps = 49.8 ksps

Channel 5 sampling rate: (1 / 12) × 600 ksps = 49.8 ksps

Acquisition Time Calculation

Acquisition time (t_sample) specifies how long an analog input signal has to charge the internal capacitor array. Figure 8-8 shows a simplified internal input sampling mechanism of a SAR ADC. The internal impedance (Z_{INAD}), external source resistance (R_{source}), and sample capacitor (C_{INAD}) form a simple RC network. As a result, the accuracy of the ADC can be affected if the ADC is given insufficient time to charge the capacitor. To resolve this problem, the user can either reduce the source resistance or increase the sampling time by changing the acquisition time in the design or in the SmartGen GUI.

Using the ADC with Direct Input

When the Fusion ADC is driven by a direct input (the prescaler is not used), Actel recommends driving the analog input pin with low source impedance (R_{source}) for fast acquisition time. High source impedance (R_{source}) is acceptable, but the acquisition time will be increased.

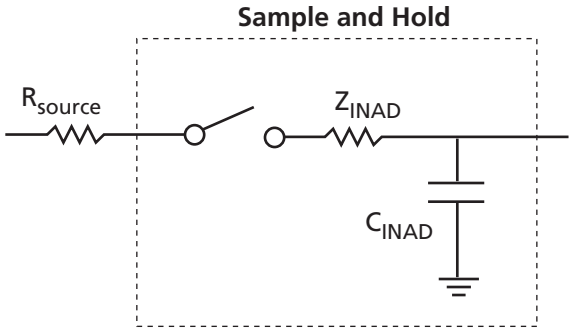


Figure 8-8 • **Simplified Sample and Hold Circuitry**

EQ 8-6 can be used to approximate the acquisition time that can be entered in SmartGen. In this equation, 5τ is used as an example to approximate the acquisition time, but it is application-dependent. Based on the acquisition time, SmartGen will provide the sample rate calculation using EQ 8-4 on page 8-8 and EQ 8-5 on page 8-8. If the actual acquisition time is higher than the software setting, the settling time error can affect the accuracy of the ADC, because the sampling capacitor is only partially charged within the given sampling cycle, referred to as the acquisition period (t_sample).

$$\text{Acquisition Time (t_sample)} \sim 5 \times (R_{source} + Z_{INAD}) \times C_{INAD}$$

EQ 8-6

Users can calculate the minimum actual acquisition time by using EQ 8-7:

$$V_{OUT} = V_{IN}(1 - e^{-t/RC})$$

EQ 8-7

For 0.5 LSB gain error, V_{out} should be replaced with (V_{in} – 0.5 × LSB Value):

$$(V_{IN} - 0.5 \times \text{LSB Value}) = V_{IN}(1 - e^{-t/RC})$$

EQ 8-8

where V_{IN} is the ADC reference voltage (V_{REFADC}).

Solving EQ 8-8,

$$t = RC \times \ln\left(\frac{V_{IN}}{0.5 \times \text{LSB Value}}\right)$$

EQ 8-9

where $R = Z_{INAD} + R_{source}$ and $C = C_{INAD}$.

Examples are given in Table 8-2 and Table 8-3.

Table 8-2 • **ADC Parameters – V_{IN} = 2.56 V, R = 4000 Ω (R_{source} ~ 0 Ω), and C = 18 pF**

Resolution (bits)	LSB Value (mV)	Min. Sample/Hold Time for 0.5 LSB (µs)
8	10	0.449
10	2.5	0.549
12	0.625	0.649

Table 8-3 • **ADC Parameters – V_{IN} = 3.3 V, R = 4000 Ω (R_{source} ~ 0 Ω), and C = 18 pF**

Resolution (bits)	LSB Value (mV)	Min. Sample/Hold Time for 0.5 LSB (µs)
8	12.891	0.449
10	3.223	0.549
12	0.806	0.649

Using the ADC with Built-In Prescaler

When using the prescaler, the user can achieve the highest sampling rate by providing a recommended 10 µs acquisition time, which is the maximum settling time when prescaler is used with the ADC. Using SmartGen, users can set the acquisition time in the software GUI so the corresponding sampling rate will be calculated automatically. Users with other design flows must ensure the control logic is allocating sufficient time for the ADC to perform a conversion that satisfies the accuracy requirement.

Prescaler Selection

As mentioned in the "Analog-to-Digital Converter Background" section on page 8-1, the analog input signals to the ADC must be mapped to the ADC reference voltage range. If the maximum value of an analog input voltage is greater than or less than the ADC reference voltage, the embedded prescaler feature can be used to amplify or attenuate the input voltage signal to match the input voltage range of the ADC. In SmartGen design flows, designers can enter the expected voltage range, and the software will configure the appropriate factors. If a design flow other than SmartGen is used, refer to Table 2-46, "Prescaler Control Truth Table," in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet to select the appropriate scaling factor.

Analog Configuration MUX (ACM)

The FPGA core uses the Analog Configuration MUX to interface with the Analog Quad configuration settings and Real-Time Counter (RTC) system. To use the Analog Quads, appropriate configuration settings (scaling factor, polarity, prescaler usage, etc.) must be set before using these features. Each Analog Quad has one byte of register space to store its configuration settings, and users can access these registers via the ACM, which is part of the Analog Block Macro. Similarly, the RTC counter register and match register can be accessed by the FPGA core via the ACM. Table 8-4 shows the ACM ports for the FPGA core to interface with the Analog Quads and RTC system.

In a SmartGen design flow, the configuration setting in the GUI is translated into the corresponding configuration bits for the Analog Quads and RTC registers. The configuration settings data is stored in the embedded flash memory, and when the device is powered up or reset, the SmartGen IP loads the configuration settings from the embedded flash memory into the corresponding registers. The Analog Quads and RTC systems are functional after this initialization stage; INIT_DONE (active high) from the SmartGen IP indicates the completion of the initialization stage.

With a different design flow, the Analog Quads and RTC registers must be configured. The configuration data can be stored in the embedded FlashROM, flash memory, or core logic tiles.

Table 8-4 • **ACM Interface**

Port Name	Type	Description
ACMWDATA[7:0]	Input	Writing data from the FPGA core to the analog system
ACMRDATA[7:0]	Output	Reading the data from the analog system to the FPGA core
ACMADDRESS[7:0]	Input	Address
ACMWEN	Input	0 for reading 1 for writing
ACMCLK	Input	Clock input from the FPGA (maximum frequency = 10 MHz)
ACMRESET	Input	Active-low asynchronous reset

Figure 8-9 and Figure 8-10 show the ACM Read and Write operations. Refer to Table 2-44, "Analog Configuration Multiplexer (ACM) Timing," in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet for the corresponding timing parameters.

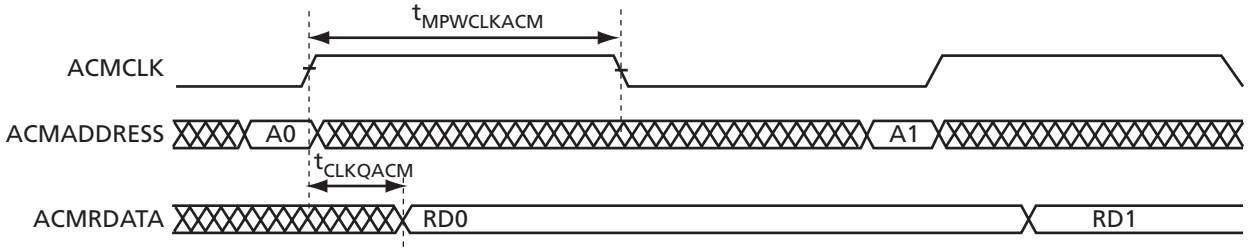


Figure 8-9 • **ACM Read Waveforms**

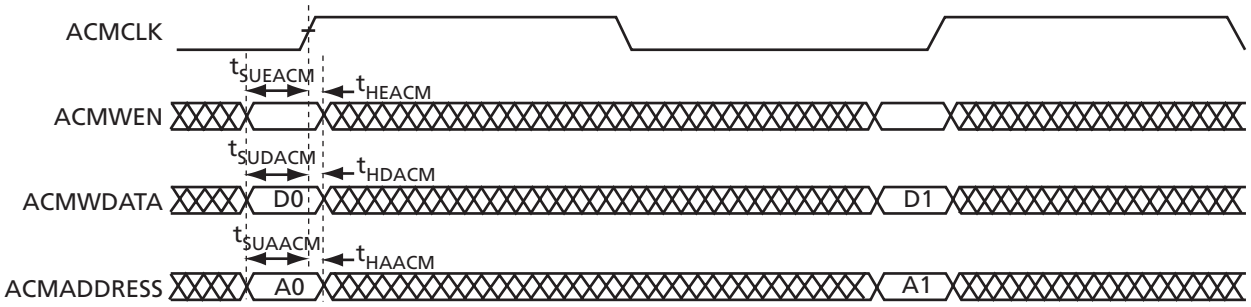
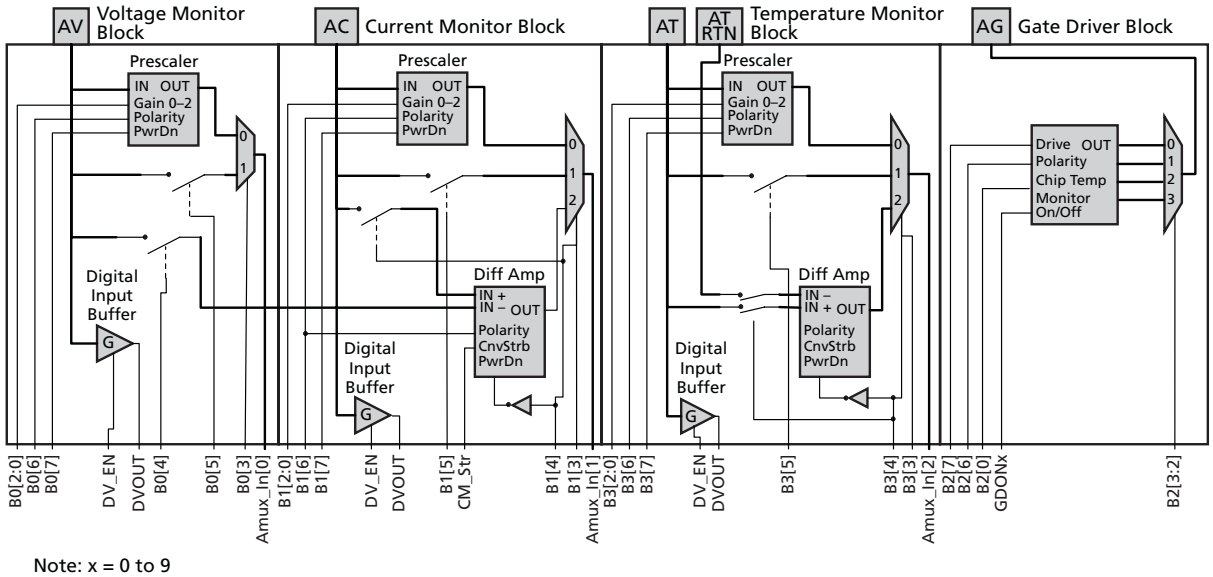


Figure 8-10 • **ACM Write Waveforms**

Figure 8-11 shows the block diagram of the Analog Block. The figure shows how the configuration byte (B0–4) is connected to each block and how each block is interfaced with the user-accessible Analog Block macro and the ADC. Note that the soft IP interface to the AB manages the configuration, making connectivity transparent to the user.



Note: x = 0 to 9

Figure 8-11 • **Analog Block**

Refer to the *Analog Quad ACM Byte Assignment* table in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet for the explanation of each bit setting and its corresponding configurations.

Refer to the *ACM Address Decode Table for Analog Quad* table in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet for the corresponding address for each Analog Quad and RTC register.

Part Number and Revision Date

Part Number 51700092-002-0

Revised November 2007

9 – Fusion Design Solutions and Methodologies

With a rich mixture of analog and digital features, coupled with the ability to build mixed-signal designs either in HDL or with a processor, the Actel Fusion® mixed-signal FPGA offers designers the ultimate in flexibility. However, with product flexibility comes design complexity. Actel offers several design solutions that make the design process simple for all users.

HDL Design with Analog System Soft IP

The Analog System Soft IP Design Flow is an IP-based design method for HDL designs, which establishes a backbone to interconnect the Fusion FPGA fabric, the Analog System, the embedded flash memory block, and other peripherals. Figure 9-1 on page 9-2 provides an overview of the design flow.

The Analog System Soft IP Design Flow offers a number of advantages to users. All the required soft IP cores are free. Sample sequence control, averaging/filtering and threshold response functions are built-in and specified by an intuitive GUI in Actel Libero® Integrated Design Environment (IDE). IP configuration and connectivity are tightly integrated into SmartDesign and Libero IDE, enabling users to rapidly and seamlessly implement the complete analog and peripheral interface. Users do not have to write up their own code to control the analog and flash memory systems.

Users have several options for integrating the Analog System into their design, but at the heart of it all are the Analog System Builder and Flash Memory System Builder from Libero IDE. The Analog System Builder creates VHDL or Verilog source code and the configuration file to be stored in the embedded flash memory. Users can invoke the Analog System Builder and Flash Memory System Builder, along with the generators for other Fusion and IP cores, from the Cores Catalog window in Libero IDE, or modify an existing configuration from within SmartDesign. Other Fusion-specific core generators in Libero IDE include the No-Glitch MUX (NGMUX), RC Oscillator, Crystal Oscillator, SRAM, FIFO, FlashROM, I/Os, and Voltage Regulator Power Supply Monitor (VRPSM).

SmartDesign is a unique block-diagram-based design entry tool introduced in Libero IDE v8.0 that gives users the capability to visually create block-level system designs and automatically abstract the result into synthesis-ready source code. Designers can build their entire design in SmartDesign. For Fusion designs, SmartDesign identifies required connections between blocks in the Analog System and automatically stitches them together. SmartDesign also provides a connectivity grid, which enables users to graphically perform port mapping across all the blocks in the design. If the Analog System is used in SmartDesign, SmartDesign will audit the connections and any updates made to the Analog System. For example, if the Analog System is regenerated, the tool will remind the user that the NVM client must be regenerated.

If users do not use SmartDesign, they can generate the Analog System and required embedded flash memory clients from the Libero IDE Cores Catalog window. They must then manually stitch the Fusion blocks together with the user HDL code.

Note: Any time the Analog System is regenerated, the Analog System client for the embedded flash memory must also be regenerated from the Flash Memory System Builder in Libero IDE.

After building the HDL design, users should take their design through the rest of the FPGA design flow—synthesis, post-synthesis simulation, and Designer functions including compile, place-and-route, package pin assignment, and static timing analysis—and then perform back-annotated simulation. Once the design is finalized and timing has been verified, the design is ready to be programmed into the FPGA using the FlashPro programmer and software. The "Design State Management in SmartDesign" section on page 9-3 and the "Changing Memory Content" section on page 9-3 discuss how to handle design iterations for Fusion within Libero IDE.

FlashPoint provides the interface to generate programming files for Fusion devices. The FlashROM, Fusion embedded flash memory, and security settings can be reset and reprogrammed using FlashPoint, which is integrated into both Designer and the FlashPro software.

With FlashPoint integrated with FlashPro, users can modify the contents of the embedded flash memory or the FlashROM without using Designer or Libero IDE. By default, Designer's program file generation from FlashPoint generates a programming database file (PDB) file instead of a STAPL file. A PDB file is required to enable users to make modifications using the FlashPro software. When users need to change the programming settings in FlashPro, they can simply click the **Configure PDB** button. All modifications are stored in the PDB file, and FlashPro uses the information to program the device with the appropriate settings. Libero IDE/Designer does not have to be open to make these modifications.

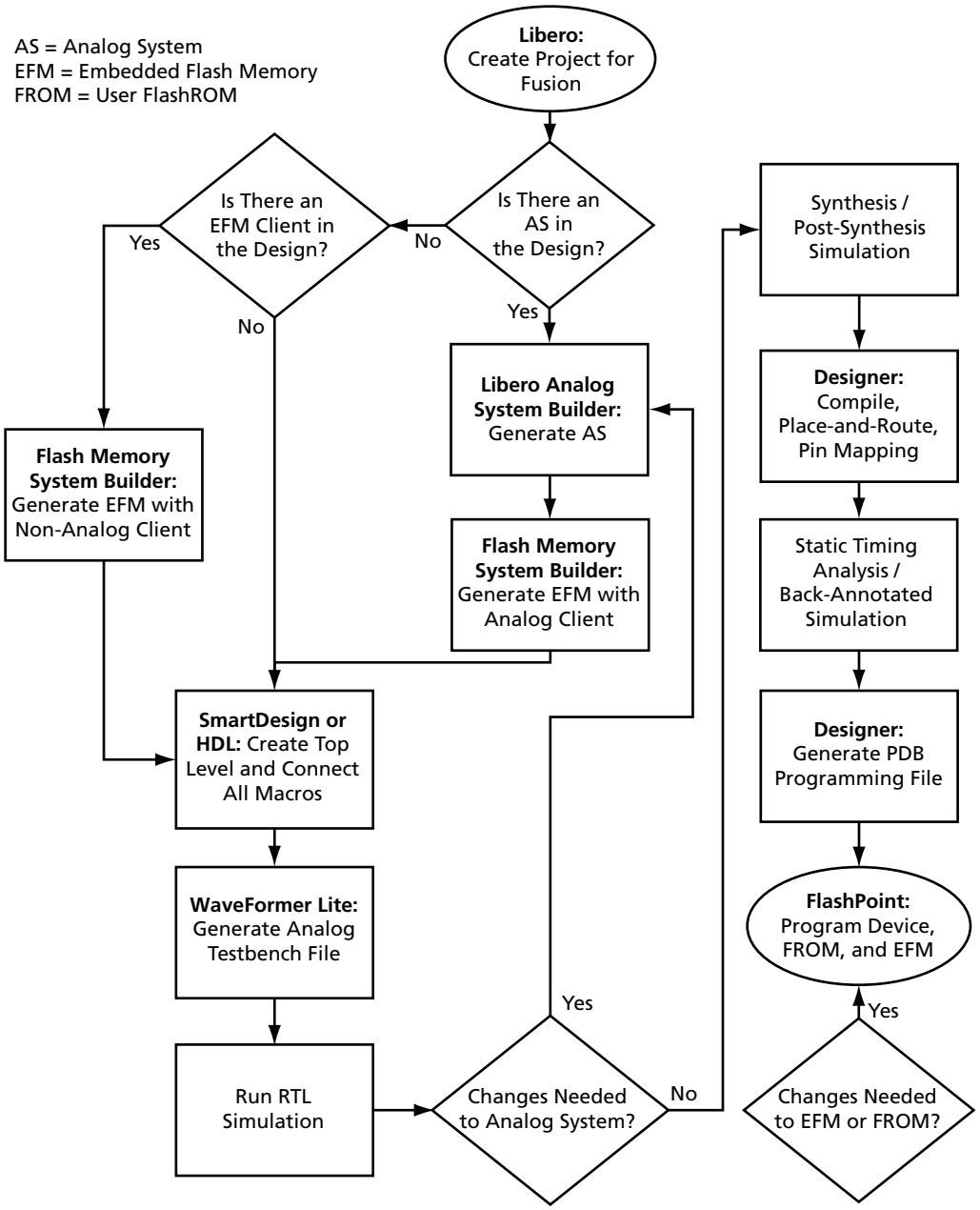


Figure 9-1 • **Analog System Soft IP Design Flow**

Design State Management in SmartDesign

When any component with instances in a SmartDesign design is changed, all instances of that component detect the change. If the change only affects the memory content, user changes do not affect the component's behavior or port interface, and the user's SmartDesign design does not need to be updated. If the change affects the behavior of the instantiated component but the change does not affect the component's port interface, the design must be resynthesized, but the SmartDesign design does not need to be updated.

If the port interface of the instantiated component is changed, the user must reconcile the new definition for all instances of the component and resolve any mismatches. If a port is deleted, SmartDesign will remove that port and clear all the connections to that port when the user reconciles all instances. If a new port is added to the component, instances of that component will contain the new port when the user reconciles all instances. The affected instances are identified in the SmartDesign design in the Connectivity Grid and the Canvas with an exclamation point. Right-click an instance and choose **Update With Latest Component**.

Note: For HDL modules instantiated in a SmartDesign design, if the modification causes syntax errors, SmartDesign does not detect the port changes. The changes will be recognized when the syntax errors are resolved.

Changing Memory Content

For certain cores such as Analog System Builder and Flash Memory, it is possible to change the configuration such that only the memory content used for programming is altered. In this case, Libero IDE only invalidates the programming file, but synthesis, compile, and place-and-route results remain valid.

When the user modifies the memory content of a core—such as Analog System Builder or RAM with Initialization—that is used by a Flash Memory core, the Flash Memory core indicates that one of its dependent components has changed and that it needs to be regenerated. This indication is shown in the Design Hierarchy or Files tab. In these cases, Libero IDE indicates that the programming file is out of date, but synthesis and place-and-route remain valid. The user only needs to regenerate the programming file in FlashPoint.

If any core is regenerated when the HDL file is not modified, the Libero IDE Project Manager design state will not invalidate synthesis or place-and-route results. For these scenarios, the new embedded NVM data file (EFC file) will be used to update the programming file within Libero IDE, or can be imported into FlashPro. Some specific cores are listed below:

- RAM with Initialization core – The memory content can be modified without invalidating synthesis.
- Analog System Builder core – The following can be modified without invalidating synthesis:
  - Existing flag settings: threshold levels, assertion/deassertion counts, OVER/UNDER type
  - Modifying sequence order or adding sequence operations
  - Changing acquisition times
  - Resistor value for the current monitor
  - RTC time settings
  - Gate driver source current
- Flash Memory System Builder core – The following can be modified without invalidating synthesis:
  - Modifying memory file or memory content for clients
  - JTAG protection for initialization clients

Microprocessor/Microcontroller Design

Actel offers several microprocessor and microcontroller solutions for customers, all of which are tightly integrated with Actel Libero IDE, optimized for Actel FPGA architecture, and supplied with a complete toolset for code compile and debug. Here is a summary of Actel's available solutions:

- **Cortex-M1:** The first ARM® processor developed specifically for implementation in FPGAs. Cortex-M1 is available without license fees or royalties for use in Actel M1 ProASIC3/E and Fusion devices.
- **CoreMP7:** CoreMP7 is a soft IP version of the ARM7TDMI-S™ that is optimized for use in Actel M7 Fusion and ProASIC3/E flash-based FPGAs. CoreMP7 is available with no license fees or royalties—bringing ARM7™ to the masses.
- **Core8051(s):** Both Core8051 and Core8051s are code-compatible with the industry-standard 8051 architecture, allowing designers to utilize existing code while shortening design time. Further, both are single-cycle execution architectures, executing one instruction per clock cycle. Core8051 has the traditional SFR memory space and includes the standard 8051 peripherals, and Core8051s replaces the traditional SFR interface with an Advanced Peripheral Bus (APB) interface, allowing the customization of the 8051 peripheral set.
- **CoreABC:** CoreABC (AMBA [Advanced Microcontroller Bus Architecture] Bus Controller) is the smallest and first RTL-programmable soft microcontroller available for FPGAs. The free controller resides on the APB, can be implemented in as few as 241 tiles, and can be used in the smallest Actel devices.
- **LEON3:** LEON3 is a 32-bit processor based on the SPARC V8 architecture, optimized for use in Actel FPGAs. A fault-tolerant version of the LEON3 processor is available for system-critical applications.
- **AMBA:** Actel supplies a full range of subsystem IP cores: AMBA bus interfaces, memory controllers, timers, and others. The subsystem IP connects to the processor via the AMBA bus and is available for free in CoreConsole.

For the above ARM-based processors and CoreABC, Actel offers CoreAI (Analog Interface) to interface with the Analog System. CoreAI allows for simple control of the analog peripherals within Fusion. Control can be implemented with an internal or external microprocessor or microcontroller, or with user-created custom logic within the FPGA fabric. The AMBA APB slave interface is used as the primary control mechanism within CoreAI, as shown in Figure 9-2. CoreAI instantiates the Analog Block (AB) macro, which includes the Analog Configuration MUX (ACM) interface, Analog Quads, and Real-Time Counter (RTC). Several aspects of CoreAI can be configured using top-level parameters (Verilog) or generics (VHDL). For a detailed description of the parameters/generics, refer to the *CoreAI Handbook*.

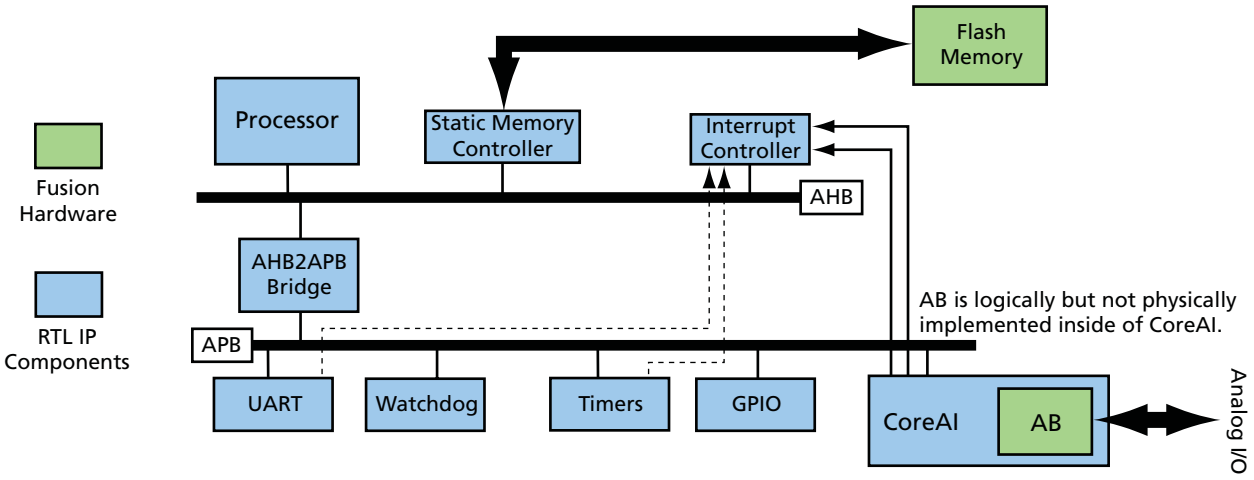


Figure 9-2 • **Processor System Using CoreAI**

Tools Overview

Actel offers FPGA development tools for microprocessor and microcontrollers, and a complete development and debug environment for Actel's microprocessor solutions (Figure 9-3). With Actel solutions, users can shorten development time using CoreConsole IP Deployment Platform (IDP), which includes a graphical user interface and a block stitcher to simplify the assembly of IP cores for embedded applications in FPGAs. This tool integrates with Actel Libero IDE, which includes Actel Designer software for place-and-route. To enable Cortex-M1 and CoreMP7 users to debug the programs they write for their processors, there is optional hardware within the core that implements JTAG debug features, such as breakpoints. Various third-party tools, like the ARM RealView® Developer Kit and Actel's own free SoftConsole, provide tools for building, debugging, and managing software development projects that run on the processor. The toolkit, available from Actel, contains an optimized C compiler, debugger, assembler, and instruction set simulator. For an overview of the processor design flow, refer to the *Actel Processor Design Flow webcast*. With a processor built into the Fusion FPGA fabric, the Fusion embedded flash memory can be used for program storage, which can in turn be executed out of internal or external memory. Implement the appropriate IP in CoreConsole for an internal or external RAM interface. Use the Data Storage Client from the Flash Memory System Builder in Libero IDE to create a partition in the flash memory and FlashPoint to load program code during device programming.

For CoreABC-based designs, users can choose either a soft or hard implementation of the core. In the soft implementation, CoreABC operates on assembly instruction code residing in flash memory and executed either directly out of flash or from local SRAM. In the hard implementation, CoreABC becomes a VHDL- or Verilog-coded state machine derived from assembly code. For either implementation, Actel recommends that users configure and generate the core using CoreConsole. With CoreConsole, users can easily connect the required peripherals and build the subsystem including CoreAI on the APB. Once the CoreConsole component including CoreABC and CoreAI is generated, users should follow a standard HDL FPGA design flow. Note that if CoreABC soft mode is used, users must create an initialization client for the RAM from the Embedded Flash Memory, using the Flash Memory System Builder in Libero IDE. The initialization client can be created by loading the memory contents file that is automatically created by CoreConsole during generation of CoreABC. For more information on building a design with CoreABC, refer to the *Fusion Starter Kit User's Guide and Tutorial*.

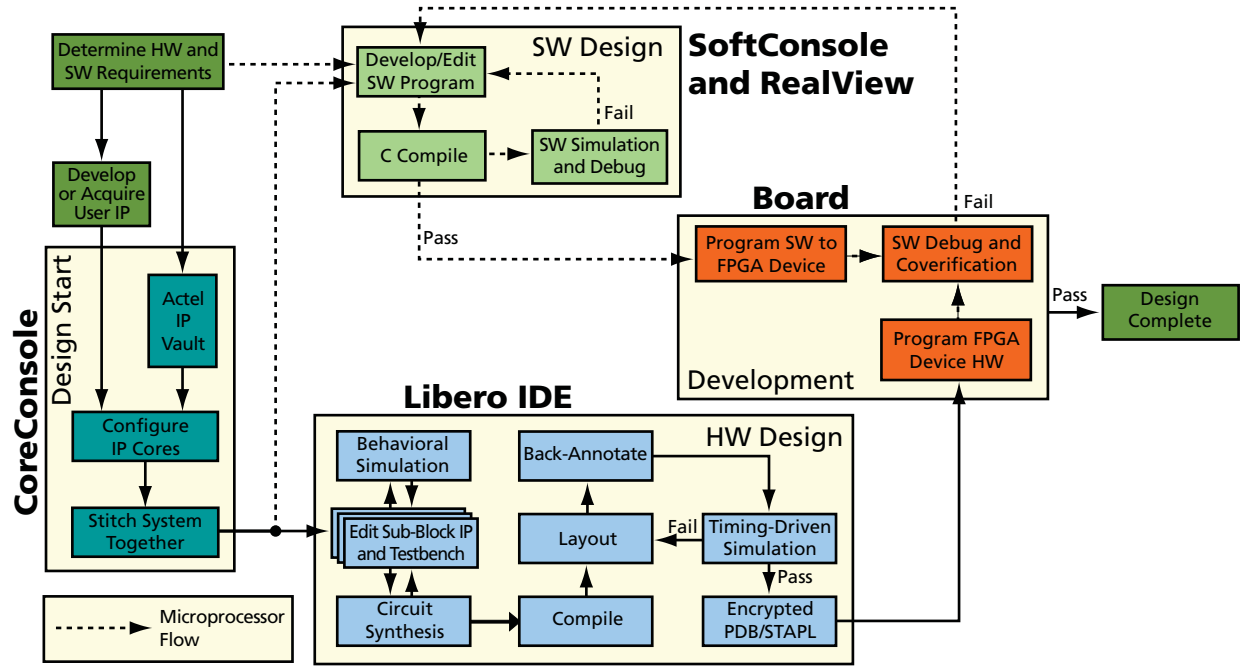


Figure 9-3 • **Fusion Design Flow**

Part Number and Revision Date

Part Number 51700092-004-0
Revised November 2007

10 – Interfacing with the Fusion Analog System: Processor/Microcontroller Interface

Objective

This chapter describes the applications in which a microprocessor or microcontroller is the core of the design that controls the Fusion Analog Block (AB). The design's microprocessor/microcontroller interacts with CoreAI (Analog Interface) as the Analog Block interface and does not access the AB macro directly.

The design in this chapter uses CoreAI within Actel CoreConsole IP Deployment Platform (IDP). However, the contents and usage of this chapter are not limited to CoreConsole users.

CoreAI

Introduction

CoreAI is Actel's Analog Interface core designed to facilitate access to the Actel Fusion® Analog Block (AB) by a microprocessor/microcontroller. CoreAI provides register/address space that can be written to or read from by a microprocessor/microcontroller to configure, control, and interact with the Analog Block. For more information on CoreAI specifications and usage, refer to the *CoreAI Handbook*. This section describes how the microprocessor/microcontroller accesses and configures CoreAI to implement voltage, current, and temperature monitoring, as well as gate-driving applications.

CoreAI Settings in CoreConsole

CoreAI can be configured by writing the desired values into all required CoreAI address spaces. CoreAI's parameters and generics, used by the core's source code, can be set within CoreConsole. Refer to the CoreAI Parameter/Generic Descriptions table in the *CoreAI Handbook* for a complete list of parameters.

Analog Configuration MUX (ACM) Clocking, Interrupt, and Internal Temperature Monitor Configuration

The first step of CoreAI configuration in CoreConsole is the ACM clock divider setting, shown in Figure 10-1. The ACM clock (ACMCLK) maximum frequency is limited to 10 MHz, so the user must select a setting that will ensure that ACMCLK is not greater than 10 MHz.

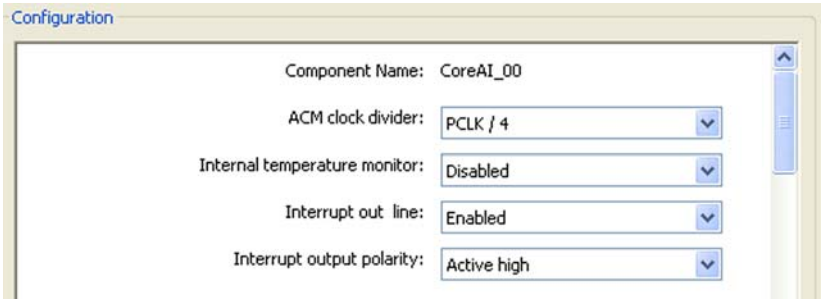


***Figure 10-1* • ACM Clock Configuration in CoreConsole**

In the ACM clock divider setting, select the dividing factor to be used based on EQ 10-1:

$$F_{ACMCLK} = F_{PCLK} / n$$

EQ 10-1

where n = 2, 4, 8, or 16; F_{ACMCLK} is the frequency of the ACMCLK; and F_{PCLK} is the frequency of PCLK, the peripheral bus clock usually connected to the design main system clock.

The PCLK frequency is essentially the speed of the Advanced Peripheral Bus (APB) and the clock speed of the design's microprocessor/microcontroller. For example, if the system clock speed is designed to be more than 20 MHz, the ACM clock divider factor cannot be set to two, since it correlates to an ACM clock frequency of more than 10 MHz.

Analog Quad Configuration

The main part of CoreAI configuration in CoreConsole is dedicated to the Analog Quad settings (Figure 10-2). Each quad consists of four settings—AV, AC, AT, and AG—and represents the Fusion device architecture.

Figure 10-2 • **Analog Quad Configuration in CoreConsole**

The CoreConsole GUI settings facilitate the configuration of the core's internal register space (initialization). The settings in CoreConsole set the parameters and generics of the CoreAI source code and create information to be used in the Analog Block's initialization. CoreConsole will export all files used for initialization to the software export folder. *Acm_defines.h* contains definitions for the values used to configure the ACM (defined per user settings in CoreConsole) in the Analog Block of a Fusion device.

Note: For CoreABC (AMBA [Advanced Microcontroller Bus Architecture] Bus Controller) users, if the APBWRT ACM command is enabled, a Verilog or VHDL file will be generated. This file contains a lookup table that will configure the ACM with user-specified settings. Initialization files and their usage are discussed in further detail in "Analog Configuration MUX Initialization" on page 10-6. The user can also configure the ACM bus without the CoreConsole-generated initialization files.

The following describes the Analog Quad software GUI settings and their effects on CoreAI parameters/generics:

- AVn input: The AV configuration drop-down menu lists all supported analog input voltage ranges and polarities for ACM initialization purposes. The main categories for AV configuration are as follows:
  - **Analog voltage input enabled/disabled**: If disabled, the specified analog AV input of the Analog Block will be tied LOW internally and will not be listed as an accessible CoreAI port in the code.
  - **AV input used as digital input**: If used as a digital input, the corresponding DAVOUT output port will be listed as a top-level port of the core, to be connected to the digital input of your design.
- ACn input: The AC configuration drop-down menu supports all analog input voltage ranges and polarities. The basic settings are as follows:
  - **AC pin disabled**: If disabled, the specified analog AC input of the Analog Block will be tied LOW internally and will not be listed as an accessible CoreAI port in the code.
  - **AC pin used as current monitor**: If the AC pin is set to be used as a current monitor or voltage monitor, the CFG_ACx bits will be set as described in the *CoreAI Handbook*. In current monitoring applications, sampling the current from an Analog Quad (configured

as current monitor) is controlled by the corresponding CMSTB input. CoreConsole gives the option to configure the CMSTB input of a current monitoring quad to be either register-driven and controlled by the software, or hardware-driven. If configured to be hardware-driven, an HD_CMSTB port is added to the CoreAI input pins and should be controlled by the user logic in the FPGA fabric.

  - **AC pin used as voltage monitor**: If the AC pin is set to be used as a voltage monitor, the CFG_ACx bits will be set as described in the *CoreAI Handbook*.
  - **AC input used as digital input**: If used as a digital input, the corresponding DACOUT output port will be listed as a top-level port of the core, to be connected to the digital input of your design.

- ATn input: In the AT configuration drop-down menu, the following parameters can be set:
  - **AT input enabled as temperature monitor or completely disabled**: If disabled, the specified analog AT input of the Analog Block will be tied LOW internally and will not be listed as an accessible CoreAI port in the code.
  - **AT input used as digital input**: If used as a digital input, the corresponding DATOUT output port will be listed as a top-level port of CoreAI, to be connected the digital input of your design.

  In temperature monitoring applications, sampling temperature from the AT input pin (configured as temperature monitor) is controlled by the corresponding TMSTB input. CoreConsole gives the option to configure the TMSTB input of a temperature monitoring quad to be either a software-driven or a hardware-driven register space. If configured to be hardware-driven, an HD_TMSTB port is added to the CoreAI input pins. The HD_TMSTB port must be controlled by the user's logic in the FPGA fabric. A similar implementation is used in the CoreAI module for internal temperature monitoring.
- AGn output: In the AG configuration drop-down menu, the following parameters can be set:
  - **AG output disabled**: If disabled, the specified AG output will be unused and omitted from the top-level CoreAI ports. The corresponding GDON input of the Analog Block will be tied LOW internal to the core.
  - **AG output enabled and driven by software-controlled register (software-driven)**: If the AG output is enabled and driven by software, the AG output pin will turn on or off (consequently turning the external PMOS or NMOS on and off) when 1 or 0 is written to the desired location of the analog-to-digital converter (ADC) control registers.
  - **AG output enabled and driven by FPGA core logic (hardware-driven)**: If the AG output is enabled and driven by hardware, the corresponding HD_GDONx input of CoreAI will be added to the top-level ports of the core, and the AG output pin will follow the logic that drives that HD_GDON input of CoreAI. In this case, the appropriate GDON bit in ADC control register 5 will be set to 1 (enabled) during initialization of the CoreAI.

ADC Settings

The main ADC settings can be configured in CoreConsole:

- Mode
- TVC
- STC (Sample Time Control—used to define sampling time of ADC: {2 to 257} × ADC clock period)
- ADCRESET
- ADCSTART
- Power Down
- VAREF Selection
- ADC Channel Number Control

The purpose and specific usage of each setting are described in the *Analog-to-Digital Converter* section of *Designing the Fusion Analog System* and in the *CoreAI Handbook*. These settings can be

controlled in two manners: software- or hardware-driven. When software-driven, these settings are controlled by an APB register/address space in CoreAI according to the address mapping described in the *CoreAI Handbook*. When hardware driven, these settings are controlled by direct inputs (e.g., HD_TVC) to the core and need to be managed accordingly by the user design in the FPGA fabric or tied to specific values.

When the Analog Block (ADCSTART) and ADC (ADCRESET) are configured as software-driven, the ADCSTART and ADCRESET registers are self-clearing, and ADCSTART or ADCRESET can be initiated by writing 1 to the specific register address. These registers will clear themselves and be ready for the next ADCSTART or ADCRESET request with no need for the user to set these registers back to 0 prior to the next request. The self-clearing functionality does not exist when ADCSTART or ADCRESET are hardware-driven, and these inputs must be reset to 0 before issuing the next request.

Clocking Scheme

There are four main clock domains in a basic mixed-signal Fusion design: the system clock (SYSCLK), ACMCLK, ADCCLK, and the initialization clock.

Any basic, functional mixed-signal Fusion design that uses Analog Quads needs to interface with the ACM and ADC. Besides the frequency requirements of ACMCLK and ADCCLK, there will be areas in which data is transferred from one clock domain to another between SYSCLK and ACMCLK or ADCCLK.

SYSCLK

The system clock (SYSCLK) is the microprocessor clock. When CoreABC (or any other processor/controller) and CoreAI are used together, SYSCLK and PCLK (the APB interface clock) are normally connected together as the overall system clock.

ACMCLK

ACMCLK is the clock input to the ACM used for configuration/initialization of the Analog Quads. The ACMCLK frequency is limited to 10 MHz. CoreAI uses an internal clock divider that divides the input SYSCLK by a user-defined factor and drives it to the ACMCLK input of the Analog Block. When CoreAI is used as an APB peripheral to the microprocessor, ACMCLK will be seamless if the PCLK / n factor is set appropriately in the CoreAI settings to limit the ACMCLK frequency to below 10 MHz.

ADCCLK

ADCCLK is the clock input to the Analog Block used by the ADC. ADCCLK is used for internal ADC operations and serves as a reference for determining the conversion time of the ADC (through the STC setting). The ADCCLK maximum frequency is 10 MHz. An internal clock divider inside the Analog Block is used to divide the input system clock (SYSCLK) and generate the ADCCLK input to the ADC. The internal divider value is configurable through the 8-bit TVC register, where TVC can be set from 0 to 255 (EQ 10-2):

$$ADCCLK = SYSCLK / (4 \times (1 + TVC))$$

EQ 10-2

Setting TVC to 0 sets the ADCCLK frequency to the SYSCLK frequency divided by four, and the ADCCLK-to-SYSCLK division ratio increases in steps of four as the TVC value increases. This is important if the design requires ADCCLK to run at specific speed to maintain a certain sampling rate. For example, if the ADC is required to run at 10 MHz, the SYSCLK input to the Analog Block should be 40 MHz (TVC = 0), 80 MHz (TVC = 1), 120 MHz (TVC = 2), etc. If a required SYSCLK frequency does not result in the determined ADCCLK frequency (governed by EQ 10-2), SYSCLK can be fed to the Fusion CCC to generate an auxiliary SYSCLK signal with the desired frequency. This auxiliary SYSCLK signal can then drive the Analog Block. Use EQ 10-2 and appropriate TVC settings

to determine the appropriate frequency of the auxiliary SYSCLK that will generate the desired ADCCLK frequency.

Initialization Clock

Typically in microprocessor-based applications, the processor's program code is stored in a nonvolatile memory, used during power-up boots. If the microprocessor program code is stored in the Fusion embedded flash memory, clocking the embedded flash memory is an important part of the design's clocking scheme. If the embedded flash memory is used to store the processor's program code, there are two general options for the processor to access the code:

- Accessing the embedded flash memory directly. The performance of SYSCLK is limited by the speed of the embedded flash memory. In this scheme the embedded flash memory clock and SYSCLK are driven by the same signal.
- Initializing embedded SRAM blocks with the contents of the embedded flash memory and running the processor's program from SRAM. The contents of the SRAM should be initialized by the embedded flash memory (power-up initialization). The memory initialization clients can be created using Actel Libero® Integrated Design Environment (IDE). The dual-port SRAM has one port connected to the embedded flash memory and one port connected to the microprocessor. The embedded flash memory (the initialization part of the design) is clocked separately from the operational part of the design. The initialization clock (INIT_CLK) input of the initialization client is limited to 10 MHz maximum frequency. Therefore, if the SYSCLK frequency is more than 10 MHz, INIT_CLK should be driven separately. The Fusion CCC can be used to generate INIT_CLK with a frequency less than 10 MHz.

  When CoreABC is the system's microcontroller, the instruction program SRAMs are instantiated within the CoreABC module exported from CoreConsole, and the clocking scheme, shown in Figure 10-3, can be used to ensure that the initialization of the SRAM from flash memory is performed by a clock of less than 10 MHz. Once the initialization is completed, INIT_DONE will assert (active high), and the design runs from the high-speed system clock.

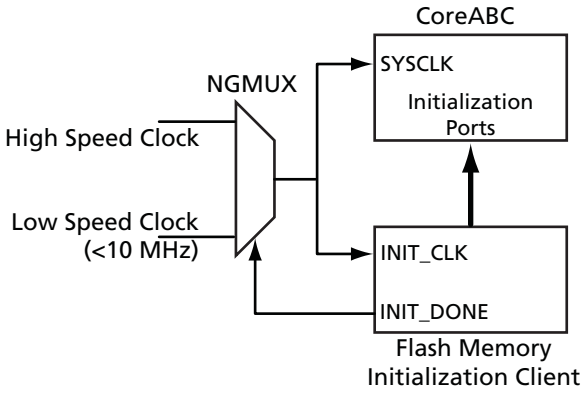


***Figure 10-3* • Clocking Scheme when Initializing SRAM with CoreABC Instructions**

The clocking architecture shown in Figure 10-3 uses an NGMUX block as the multiplexer to switch between the high- and low-speed clocks. The usage of the NGMUX macro is to prevent glitches on the clock output of the MUX when switching between the two clock inputs. Refer to the *No-Glitch Multiplexer (NGMUX)* section of *Fusion Clock Resources* for more information about the NGMUX.

Analog Configuration MUX Initialization

The ACM is a register space in the Analog Block used to configure the Analog Quads and the Real-Time Counter (RTC) architecture with the user's specification. Since the configuration scheme is stored in registers, the ACM needs to be initialized after each power-up. When using CoreAI as the interface to the Analog Block, the initialization of the ACM should be done in two steps:

1. Reset the ACM register space to put unused Analog Quads into a known mode.
2. Configure used Analog Quads (and RTC if used in the design) to desired specification.

Note: In addition to ACM initialization, the ADC needs to be calibrated after power-up for correct functionality; see "ADC Configuration and Calibration" on page 10-9.

ACM Reset

The ACM registers can be reset by activating the active-high ACMRESET bit of the CoreAI register space.[1] When configuring CoreAI, this input is controlled by bit 0 of the ACM control/status register. The ACM can be reset by writing 1 to this bit. Note that bit 0 of the ACM control/status register is self-clearing: when written with 1, it will clear itself; it does not need to be written with 0 to clear it. The ACM control/status register is designed to be located at address 0x00 of the CoreAI internal register address map.

ACM Initialization

Before a design enters the operational phase, the ACM must be initialized with the desired configuration for the Analog Quads and/or RTC registers. When CoreConsole generates all the necessary files for a microprocessor-based design, it also generates the files to be used for ACM initialization.

There are two files, found in the *SoftwareExport* folder of the CoreConsole directory, that can be used for ACM initialization: *acm_defines.h* and *quads_acm_cfg.h*. These two files contain information on initialization values for different ACM address spaces per user entries in the CoreAI settings within CoreConsole. These files can be referenced by the general microprocessor program design to be used for initialization prior to entry into operational sections.

The use of the above-mentioned files is optional. The user can manually write to the desired ACM address space (from a microprocessor/microcontroller through the APB into CoreAI) with values that will initialize the targeted Analog Quads and/or RTC registers to the desired configuration.

Note: When writing to ACM registers, the design should check the ACM status register to ensure that the previous ACM actions (write, read, or reset) are completed and that the ACM is ready to be accessed again.

1. *Note that the ACMRESET input to the Analog Block is active low. To reset the ACM in the CoreAI register space, the ACM RESET bit should be set to 1.*

ACM Initialization Specific to CoreABC

In addition to the initialization files and methodology described above, if the APBWRT ACM instruction is activated in the CoreABC settings in CoreConsole, CoreConsole will export an HDL file named *acmtable.vhd*. This file can be found in the *CoreABC/RTL* folder within the Libero IDE project. Below is an example of an *amctable.vhd* file generated by CoreConsole:

```
-- ********************************************************************/
-- Copyright 2007 Actel Corporation. All rights reserved.
-- IP Solutions Group
--
-- ANY USE OR REDISTRIBUTION IN PART OR IN WHOLE MUST BE HANDLED IN
-- ACCORDANCE WITH THE ACTEL LICENSE AGREEMENT AND MUST BE APPROVED
-- IN ADVANCE IN WRITING.
--
-- File: INSTRUCTIONS.vhd
--
-- Description: Simple APB Bus Controller
--       ACM Lookup table
--
-- Rev: 2.3  01Mar07 IPB : Production Release
--
-- Notes:
--
-- ********************************************************************/

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;

use work.support.all;

entity ACMTABLE is
  generic ( ID   : integer range 0 to 9;
            TM   : integer range 0 to 99
          );
  port   ( ACMADDR : in std_logic_vector(7 downto 0);
             ACMDATA : out std_logic_vector(7 downto 0);
             ACMDO  : out std_logic
           );
end ACMTABLE;

architecture RTL of ACMTABLE is

begin

-- This is dummy data used for testing

process(ACMADDR)
variable ADDRINT : integer range 0 to 255;
begin
  ADDRINT := conv_integer(ACMADDR);
  ACMDO <= '1';

  if TM>0 then
    case ADDRINT is
      when 0  to 99 => ACMDATA <= not ACMADDR;
      when 101 to 255 => ACMDATA <= not ACMADDR;
      when others   => ACMDATA <= (others =>'-'); ACMDO <= '0';
    end case;
  end if;

  if TM=0 then
  -- CCDirective Insert code
```

```
  --ACM lookup table for CoreABC_00(ID=0) with CoreAI_00
    if ID=0 then
      case ADDRINT is
        when 1 => ACMDATA <= conv_std_logic_vector(16#83#, 8);
        when 2 => ACMDATA <= conv_std_logic_vector(16#82#, 8);
        when 5 => ACMDATA <= conv_std_logic_vector(16#82#, 8);
        when 7 => ACMDATA <= conv_std_logic_vector(16#80#, 8);
        when 9 => ACMDATA <= conv_std_logic_vector(16#83#, 8);
        when 11 => ACMDATA <= conv_std_logic_vector(16#80#, 8);
        when 13 => ACMDATA <= conv_std_logic_vector(16#82#, 8);
        when 15 => ACMDATA <= conv_std_logic_vector(16#80#, 8);
        when 17 => ACMDATA <= conv_std_logic_vector(16#82#, 8);
        when 19 => ACMDATA <= conv_std_logic_vector(16#80#, 8);
        when others => ACMDATA <= (others => '-'); ACMDO <= '0';
      end case;
    end if;
  end if;

end process;

end RTL;
```

The last portion of the *acmtable.vhd* file defines the ACM address spaces and the corresponding values used to initialize certain Analog Quads to user specifications.

If the user decides to perform all the initialization within the CoreABC program code without utilizing any logic tiles from the FPGA fabric, this last portion of the *acmtable.vhd* file can be used to determine the constant values in the ACM write commands. The following example shows a sample of a CoreABC program in which location 1 of the ACM address space is initialized with 83h, as defined in the above *acmtable.vhd*:

```
// The following example assumes that CoreAI is in slot 0 of the APB and CoreABC is the
// bus master. Refer to the table "CoreAI Internal Register Address Map" in the CoreAI
// Handbook for ACM address mapping. Write 1 (as ACM ADDRESS) to address 0x04 of the
// CoreAI register space.
APBWRT DAT 0 0x04 1
// Write 0x83 into ACM DATA of CoreAI register space. This will result in writing 0x83
// into address 1 of ACM address space.
APBWRT DAT 0 0x08 0x83
CALL $WAIT ACM_WRITE
$WAIT_ACM_WRITE
  // Read ACM STATUS register of CoreAI
  APBREAD 0 0x00
  // Check to see if bit 4 of ACM status is cleared (write busy)
  BITTST 4
  // Remain in the loop if bit 4 is not cleared yet
  JUMP IFNOT ZERO $WAIT_ACM_WRITE
  RETURN
```

In the above example, the WAIT_ACM_WRITE function ensures that the write into ACM register space is completed before proceeding to the next ACM write instruction.

The *acmtable.vhd* file represents a MUX architecture where the output value is determined by the corresponding ACM address, which acts as the select lines. If the user intends to use the APBWRT ACM command in the design, this MUX can be implemented in the FPGA fabric (using tiles) and can be controlled by the CoreABC program to initialize ACM registers.

The following example shows the usage of the APBWRT ACM instruction in CoreABC to initialize the ACM using ACM table files:

```
// The following example assumes that CoreAI is in slot 0 of the APB and CoreABC is the
// bus master. Refer to the table "CoreAI Internal Register Address Map" in the CoreAI
// Handbook for ACM address mapping.
// Only Analog Quads are being initialized in this example (no RTC). According to the
// "ACM Address Map for Configuring Analog Quads and RTC" table in the CoreAI
// Handbook, the maximum ACM size for configuration is assumed to be 28h.
```

```
$WaitRegProg
CALL $WaitACMReady
// Write accumulator value in the ACM Address register of CoreAI
APBWRT ACC 0 0x04
// Write the value from acmtable file into ACM Data register and
// start an ACM write
APBWRT ACM 0 0x08
// Increment accumulator
INC
// Compare accumulator to 0x28
CMP 0x28
JUMP IFNOT ZERO $WaitRegProg

$WaitACMReady
  PUSH
$WaitACMReady1
  APBREAD 0 0x00
  AND 0x001C
  JUMP IFNOT ZERO $WaitACMReady1
  POP
  RETURN
```

In the above example, the CoreABC accumulator sweeps from address 0x00 to 0x28 of the ACM address space, and for each of these addresses, the APBWRT ACM command writes the appropriate values to the ACM data register space for initialization of all Analog Quads.

ADC Configuration and Calibration

ACM initialization is only used to configure the Analog Quads and/or RTC block. The ADC configuration (Mode, TVC, STC, etc.) is done through several inputs to the Analog Block. When configuring CoreAI in CoreConsole, the ADC settings can be set to be hardwired or register-controlled. When an ADC setting is selected to be hardwired, there is no need to initialize that setting before the operational phase of the design. However, when the ADC setting is selected to be software register-controlled, the settings are driven by a set of registers generally labeled as ADC control registers in CoreAI.

The most important configuration settings are in ADC control registers 1 and 2. The only configuration setting in ADC control register 2 is the STC value. Since this register also controls the ADCSTART and CHNUMBER inputs to the ADC, it will be written to during the operational phase. Therefore, there is no need for an initialization operation on STC (when selected to be register-controlled) before entering the operational phase of the design. The desired settings need to be written to ADC control register 1 to configure (TVC, PWRDOWN, VAREFSEL, and MODE). This only needs to happen once after each power-up, similar to ACM initialization.

ADC will self-calibrate after device power-up and after deactivation of ADCRESET, if ADCRESET is applied.

When the ADC is in calibration, bit 15 of the ADC status register will be set to 1 to flag that the ADC is busy calibrating itself. When the calibration is completed, this bit will be reset to 0.

Note: Both ADC calibration and ACM initialization (see "ACM Initialization" on page 10-6) should be completed after power-up before the Analog Block can properly function in the user's design.

It is common design practice to issue an ADCRESET request and check the status of bit 15 of ADC status register before entering the operation phase of the design. When this bit is cleared, the design enters the operation phase.

In the following example, CoreABC issues an ADCRESET request and then checks the status of the ADC for completion of calibration:

```
// In this example, ADC_STATUS and ADC_CTRL1 represent the address of the ADC status
// register and ADC control register 1, respectively, in the CoreAI address mapping.
// It is also assumed that CoreAI is in slot 0 of the APB.
// Write 0x0040 into ADC_CTRL1 register space of CoreAI
APBWRT 0 ADC_CTRL1 0x0040
$WaitCalibrate
    // Read ADC_STATUS register space of CoreAI
    APBREAD 0 ADC_STATUS
    AND 0x8000
    JUMP IFNOT ZERO $WaitCalibrate
```

Implementing Voltage Monitoring Applications

After completion of both ACM initialization and ADC calibration, the design enters the functional stage. This section describes how to implement a voltage monitoring application and provides design techniques to enhance sampling rate. The voltage monitoring operations are as follows:

- Selecting the analog voltage input pin to be monitored
- Sampling the voltage on the selected pin
- Translating the ADC output results into application-specific data
- Implementing a digital low-pass filter (or averaging) for voltage monitoring (optional)
- Implementing a sampling sequence

Channel Selection, ADC Sample, and Conversion Request

Voltage monitoring channel selection, requesting the ADC to start a sample, and conversion are all implemented through ADC control register 2 in the CoreAI register space. This register also controls the STC value input to the ADC. If, during the configuration of CoreAI in CoreConsole, the STC input to the ADC is set to be hardwired, bits 7–0 of ADC control register 2 will be considered as "don't care." A request to the ADC to sample and convert a specified channel is performed by activating ADCSTART. Since ADCSTART and CHNUMBER are both controlled by ADC control register 2, these two signals will be fed to ADC at the same time. Note that the ADCSTART bit is self-clearing and will be cleared to 0 after the user writes 1 to it to request an ADC start. The following example shows a CoreABC instruction in which the ADC is requested to sample and convert channel 2, with STC set to 4:

```
// In this example, ADC_CTRL2 represents the address of ADC control register 2 in the
// CoreAI address mapping. It is also assumed that CoreAI is in slot 0 of the APB.
APBWRT 0 ADC_CTRL2 0x2204
```

Obtaining Results from the ADC Output

The ADC status register in CoreAI contains status bits for the ADC and the ADC output results. Immediately after issuing an ADCSTART request, the ADC starts sampling the selected channel. During this period, the SAMPLE bit (bit 14) of the ADC status register will be set HIGH. When sampling is completed, the ADC enters conversion mode. In this mode, the SAMPLE bit will be cleared and the BUSY bit (bit 13) will be set HIGH. When the conversion is completed, the BUSY bit will be cleared, the ADC output will be placed on the RESULTS bits (bits 11 to 0), and the DATAVALID bit (bit 12) will be set HIGH. In a typical voltage monitoring application, the only status bit that needs to be monitored after issuing an ADCSTART request is the DATAVALID bit. Once DATAVALID is set HIGH, the RESULTS bits are ready to be used by the microprocessor/microcontroller. Depending on whether the ADC is configured to operate in 12-, 10-, or 8-bit mode, the ADC output will be stored in RESULTS[11:0], RESULTS[11:2], or RESULTS[11:4], respectively.

The following example shows a program routine in CoreABC instructions that continuously checks the status of the DATAVALID pin after issuing an ADCSTART request. Once DATAVALID is HIGH, it will clear all bits of the read value except the ADC RESULTS bits.

```
// In this example, ADC_STATUS represents the address of ADC status register in the
// CoreAI address mapping. It is also assumed that CoreAI is in slot 0 of the APB and
// that ADC is configured in 8-bit mode.
$ADC_Wait
  APBREAD 0 ADC_STATUS
  // Check to see of bit 12 of ADC_STATUS register is set or not
  BITTST 12
  // Remain in the loop if bit 12 is not set to 1
  JUMP IF ZERO $ADC_Wait
  AND 0x0FF0
  RETURN
```

Instead of continuous reading of the ADC_STATUS register to check DATAVALID, ADC_STATUS can be fed to the microprocessor as an interrupt and set HIGH. Then the microprocessor can read from the ADC_STATUS register to obtain RESULTS. For more information, refer to the "ADC Configuration and Calibration" section on page 10-9.

Sample Sequencing

When the microprocessor receives the latest results from the ADC status register, the microprocessor can issue another ADCSTART request on the same channel (to achieve more sampling on a particular channel) or a different channel. The desired sampling sequence can be achieved by writing the appropriate value to the CHNUMBER bit of ADC control register 2 when issuing an ADCSTART request. Refer to the *Sample Sequencing Overview* and *Sample Rate and Sample Sequence Calculation* sections of the *Designing the Fusion Analog System* chapter for background information on sampling theory.

Sample Averaging

In applications where the measured voltage line is expected to be noisy or where voltage variations are too fast for the application to respond to, it is recommended to take multiple samples from a voltage and use the computed average value of the measured samples as representative of the sampled voltage. The user can choose the filtering factor by deciding how many samples need to be taken and averaged to acquire a single data point for processing. The higher the filtering factor, the lower the effective sampling rate will be.

The following example shows a CoreABC program in which a low-pass filter on the voltage measured from channel 2 is implemented with a filtering factor of four. In other words, four samples are taken from the single channel (v1, v2, v3, and v4) and averaged: (v1 + v2 + v3 + v4) / 4. The ultimate result to be used in the rest of the program is stored in address 0 of the internal memory.

```
// In this example, it is assumed that CoreAI is in slot 0 of the APB. Also,
// ADC_CTRL2 represents the ADC control register 2 address in the CoreAI register
// space. Also, this example calls the $ADC_WAIT function as described in
// "Obtaining Results from the ADC Output" on page 10-11. In the following example,
// the 8-bit output of the ADC is averaged using four samples, and the final average
// value is stored in internal RAM address 0 when the function is completed.
// Load Z register (used as loop counter) with value 4
LOADZ DAT 4
// Write 0 to internal RAM address 0
RAMWRT 0 DAT 0
$SAMPLE_FILTER
  // Issue ADC sample and Conversion Request
  APBWRT 0 ADC_CTRL2 0x2204
  CALL $ADC_WAIT
  // Divide by 4 and add to previous values
  SHR0
  SHR0
  // Add content of internal RAM address 0 to accumulator
  ADD RAM 0
  // Write accumulator value to internal RAM address 0
  RAMWRT 0 ACC
  // Decrement the Z register value (loop counter)
  DECZ
  // Check to see of Z register (loop counter) is 0 or not
  JUMP IFNOT ZZERO $SAMPLE_FILTER
$DONE
HALT
```

Once the four samples are averaged, the sampled values are discarded. Depending on the application and sampling sequence, the user can only discard the oldest sampled value and keep the most recent values to be used for averaging with the next sampled data.

Techniques to Enhance Design Performance/Throughput

After each ADCSTART request to the Analog Block, there is a period of time (duration of ADC sample and conversion) during which the design's microprocessor continuously checks the ADC status register to indicate when the ADC is done and data is ready to be fetched from RESULTS bits. The processing throughput of the design can improve significantly if the microprocessor/microcontroller can perform other necessary tasks while waiting for the ADC to complete its cycle. There are two general methods to do this, depending on the application's requirements:

- After activating the ADCSTART input to the Analog Block and requesting an ADC sample and conversion to start, the design's microprocessor can go on performing other tasks that do not need the results of the ADC conversion. When these tasks are completed (or periodically), the microprocessor can return and check the ADC status register for DATAVALID assertion. The drawback of this method is that the ADC maximum sampling rate

capability may be compromised. In other words, the ADC might complete conversion and sit idle prior to the microprocessor's checking the ADC status register.

- The required flag in the ADC status register (DATAVALID) can be used as an interrupt input to the microprocessor. In this case, it can be assured that the microprocessor will attend to the ADC, read the data output, issue another ADCSTART request, and continue with the rest of the tasks until the next interrupt from the DATAVALID bit. The time the ADC is idle is reduced to a minimal level, enhancing the sampling rate of the ADC at the given operating frequency.

The maximum frequency of ADCCLK is 10 MHz, and the relationship between ADCCLK and SYSCLK (the system clock) is governed by the TVC setting. Therefore, maximum ADCCLK frequency is achieved at certain frequencies. For example, with a SYSCLK frequency of 40 MHz and TVC set to 0, the user can achieve 10 MHz on the ADCCLK. On the other hand, if SYSCLK is at 50 MHz, the maximum achievable ADCCLK frequency is 6.25 MHz (TVC = 1). Higher SYSCLK frequencies do not necessary translate into higher ADCCLK frequencies. This is due to the fact that ADCCLK is bounded by a 10 MHz limit and is also governed by the TVC value.

Implementing Current Monitor Applications

Regardless of the type of the signal being measured (i.e., voltage, current, or temperature), the ACM needs to be initialized at power-up to configure the Analog Quads. The ADC may need to be configured and calibrated on device power-up, a global reset event, or at the user's discretion. "Analog Configuration MUX Initialization" on page 10-6 and "ADC Configuration and Calibration" on page 10-9 still apply when a current monitoring application is being implemented.

The current monitoring procedure is very similar to the voltage monitoring steps described in "Implementing Voltage Monitoring Applications" on page 10-10. The only additional step required to perform sampling on a current monitor channel is the accurate stimulation of the CMSTB inputs to the Analog Block. For current sampling on a desired channel, CMSTB should be kept LOW for more than the TMPWC value (refer to the values in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet) to discharge previous measurements, and then HIGH for at least the TMPWC value prior to the ADCSTART request. Figure 10-4 illustrates the assertion/deassertion of the CMSTB input of a specific channel prior to starting sampling.

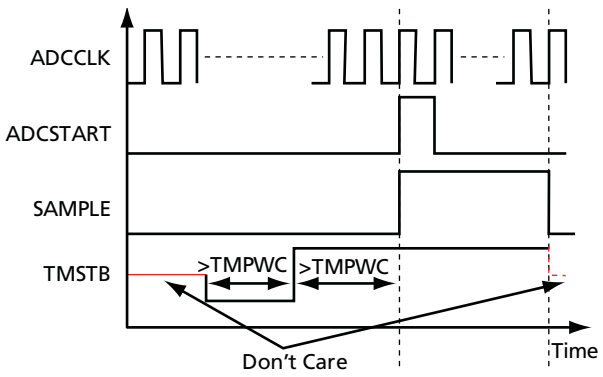


Figure 10-4 • **Assertion/Deassertion of CMSTB Input**

The user should not assert another ADCSTART prior to the completion of current ADC conversion. When sampling a current, ADCSTART cannot be asserted for at least for 2 × TMPWC after the previous ADCSTART. If the next ADCSTART is also a current monitor (or temperature monitor), there should be at least another TMPWC waiting period during which CMSTB of the desired channel is kept LOW before assertion of ADCSTART. The selected channel's CMSTB should be deasserted after the ADC has completed sampling the channel (as shown in Figure 10-4). CMSTB can be kept HIGH (after assertion of ADCSTART) until DATAVALID is asserted.

If CoreAI is configured in CoreConsole to have a hardwired CMSTB input for the desired channels (HD_CMSTBn input), the user will need to stimulate CMSTB to fulfill the requirements discussed in this section.

In summary, the following are the necessary operations to implement current monitoring:

- Selecting the analog current input pin to be monitored
- Ensuring the corresponding CMSTB input pin is LOW for more than the required TMPWC value
- Asserting CMSTB HIGH for longer than TMPWC
- Issuing an ADCSTART request on the desired CHNUMBER
- Ensuring that CMSTB remains HIGH until sampling is completed or until assertion of DATAVALID
- Translating the ADC output results into application-specific data once DATAVALID is asserted
- Implementing a digital low-pass filter for voltage monitoring (optional)
- Implementing a sampling sequence (selecting the next channel)

Translating the ADC output results to the actual measured current is slightly different from the same operation in voltage monitoring. As shown in Figure 2-56 of the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet, when configured as current monitor, a fixed 10× prescaler is used to buffer the differential voltage measured across the external resistor into the ADC. Therefore, the maximum differential voltage that can be measured with the ADC (before overflow) is limited by the V_{REF} value (EQ 10-3):

$$\text{Max. Differential Voltage across Resistor} = I(\max) \times R = V_{REF} / 10 \quad \text{EQ 10-3}$$

where R is the external resistor value in ohms. Therefore, the measured current value, based on the ADC mode, can be calculated as shown in Table 10-1.

Table 10-1 • **ADC Output Translation in Current Monitoring Applications**

LSB for 8-Bit ADC (mA)	LSB for 10-Bit ADC (mA)	LSB for 12-Bit ADC (mA)
V_{REF} / (10 × R × 0.255)	V_{REF} / (10 × R × 1.023)	V_{REF} / (10 × R × 4.095)

Implementing Temperature Monitor Applications

Regardless of the type of signal being measured (i.e., voltage, current, or temperature), the ACM needs to be initialized at power-up to configure the Analog Quads. The ADC may need to be configured and calibrated as well at device power-up, a global reset event, or the user's discretion. Therefore, the "Analog Configuration MUX Initialization" section on page 10-6 and the "ADC Configuration and Calibration" section on page 10-9 still apply when a current monitoring application is being implemented.

The temperature monitoring procedure is very similar to the current monitoring steps described in the "Implementing Current Monitor Applications" section on page 10-13. The only difference is the stimulation of the TMSTB input for the temperature monitoring channels instead of CMSTB in current monitoring. For accurate temperature sampling on a desired channel, TMSTB should be kept LOW for longer than the TMPWT value to discharge previous measurements, and then HIGH for longer than the TMPWT value prior to the ADCSTART request. The TMSTB input of the selected temperature monitoring input should remain HIGH at least until the completion of sampling. Figure 10-5 on page 10-15 illustrates the assertion/deassertion of the TMSTB input of a specific channel prior to starting sampling. TMSTB can be kept HIGH (after assertion of ADCSTART) until DATAVALID is asserted.

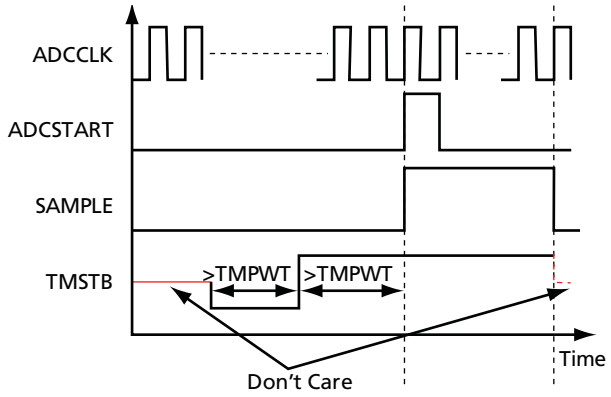


Figure 10-5 • **Assertion/Deassertion of TMSTB Input**

Since, in CoreAI, the TMSTB inputs and ADCSTART are in two different register spaces, they cannot be asserted at the same time (see Figure 10-5). Assert TMSTB HIGH first and then issue ADCSTART. If CoreAI is configured in CoreConsole to have a hardwired TMSTB input for desired channels (HD_TMSTBn input), stimulate TMSTB accordingly.

In summary, the following are the necessary operations to implement temperature monitoring:

- Selecting the analog temperature input pin to be monitored
- Ensuring the corresponding TMSTB input pin is LOW for more than the minimum value of TMPWT
- Asserting TMSTB HIGH for more than the minimum value of TMPWT
- Issuing an ADCSTART request on the desired CHNUMBER
- Ensuring that TMSTB remains HIGH until completion of sampling or assertion of DATAVALID
- Translating the ADC output results into application-specific data once DATAVALID is asserted
- Implementing a digital low-pass filter for voltage monitoring (optional)
- Implementing a sampling sequence (selecting the next channel)

Translating the ADC output voltage in temperature monitoring applications depends on the V_{REF} value of the ADC, the ADC mode, and the characteristics of the external (temperature sensor) diode. The relationship between the measured voltage and the external temperature is described in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet.

Implementing Gate Driver Applications

Implementing gate driver applications is different from implementing voltage, temperature, and current monitoring because gate drivers are outputs, and driving them does not require interaction with the ADC. However, the ACM initialization procedure, described in the "Analog Configuration MUX Initialization" section on page 10-6, is still necessary to configure the selected Analog Quads as gate drivers with specific parameters, such as drive strength and polarity. Gate driver applications can be implemented as software-/register-driven or hardware-driven.

Software-Controlled Gate Drivers

When a specific gate driver is configured in CoreAI to be software-controlled, the corresponding AG output pad (analog gate driver) follows the contents of the corresponding bit in CoreAI ADC control register 5. In this approach, the microprocessor can turn the external MOSFET (connected to the AG output pad) off or on by writing 1 or 0 to the corresponding bit of ADC control register 5. A typical use model for a software-driven gate is in power sequencing applications, where the system

management processor turns the voltage rails on/off in the desired sequence. The following example shows a sample CoreABC program in which three gate drivers are turned on sequentially.

```
// In this example, it is assumed that CoreAI is in slot 0 of the APB. Also, ADC_CTRL5
// represents the ADC control register 5 address in the CoreAI register space.
APBWRT DAT 0 ADC_CTRL5 0x0001
APBWRT DAT 0 ADC_CTRL5 0x0002
APBWRT DAT 0 ADC_CTRL5 0x0003
```

Hardware-Controlled Gate Drivers

When a specific gate driver is configured in CoreAI to be hardware-controlled, the corresponding AG output pad (analog gate driver) follows the corresponding HDGDON input of CoreAI. When designing with CoreConsole, the selected HDGDON inputs can be added to the top-level ports to be driven by the FPGA fabric.

In a typical use model for hardware-controlled gate drivers, the AG pad drives the external MOSFET gate with a pulse width modulation (PWM) signal.

Design Example

This section includes a practical example of a design using CoreABC and CoreAI to implement the following applications:

- Voltage, current, and temperature monitors
- Gate drivers

Functionality

Figure 10-6 illustrates the top-level functionality of the example design.

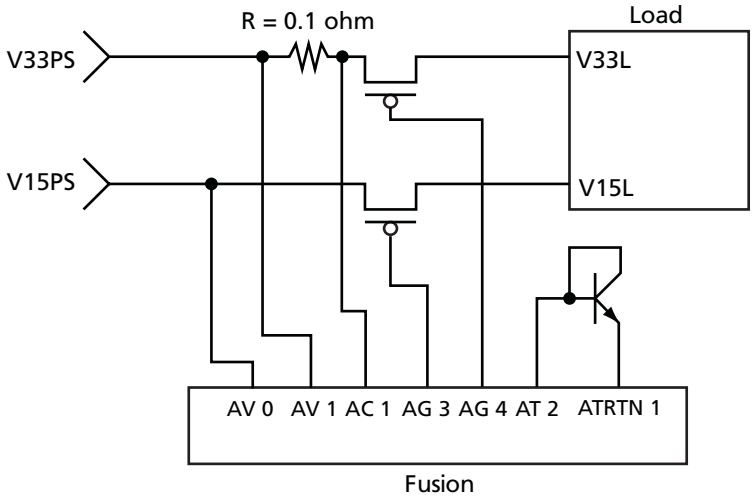


***Figure 10-6* • Top-Level Functionality of the Example Design**

Figure 10-6 shows two voltage supplies in the system: V15PS supplying 1.5 V and V33PS supplying 3.3 V. Two voltage rails (V15L and V33L) drive the power inputs of a load. These voltage rails are powered up through two MOSFETs that are controlled by the AG3 and AG4 gate drivers of the Fusion device. The current on the 3.3 V supply rail is measured across a 0.1 Ω resistor.

Figure 10-7 shows the flow chart of the example design, implemented in Fusion using CoreABC and CoreAI.

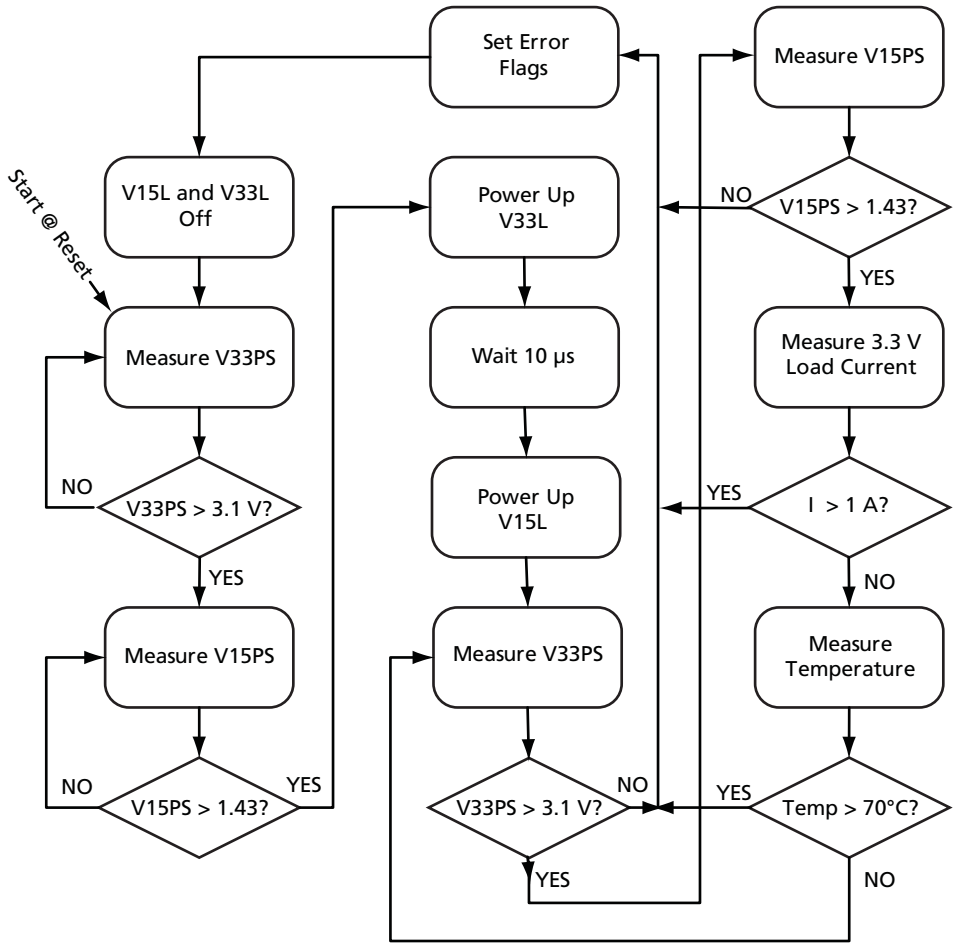


Figure 10-7 • **Flow Chart of Example Design**

The operational phase of the design starts with measuring the power supplies and ensuring that they satisfy the minimum requirements of the load voltage rails. Once the power supplies are up and running, the V33L and V15L rails are powered up sequentially (3.3 V first and 1.5 V second after 10 µs). After powering the rails, the design continuously monitors the 3.3 V supply, 1.5 V supply, 3.3 V rail current, and operating temperature (defined sampling sequence). If no abnormal condition occurs, the design remains in this loop monitoring the operating conditions until one of the supplies is turned off externally. In case of an abnormal situation (e.g., current being more than 1 A), the design sets error flags, turns off both V33L and V15L, and checks for supply lines, and if they are still up and running, it attempts to power up the load again.

Implementation in a Fusion Device

This design example is implemented using CoreABC as the core microcontroller and CoreAI as the interface to the Fusion Analog Block. The steps below describe the CoreABC and CoreAI settings and connections and the CoreABC program that implements the flow chart illustrated in Figure 10-7 on page 10-17.

Building the Example Design System in CoreConsole

CoreABC and CoreAI in this example are connected together using an APB, as shown in Figure 10-8.

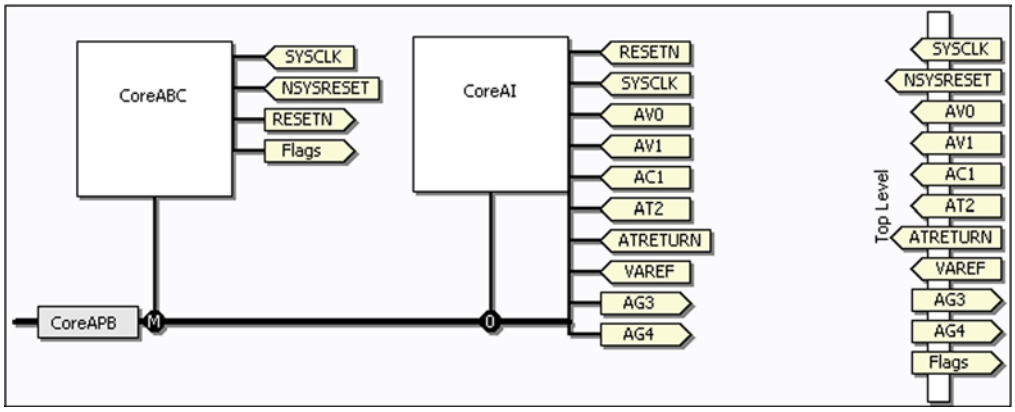


Figure 10-8 • **Implementation of Example Design in CoreConsole**

The following are the major settings of the CoreABC and CoreAI configuration in CoreConsole:

CoreABC Settings:

- APB address bus width: 8
- APB data bus width: 16
- Number of CoreABC outputs: 1
- Instruction store: soft

CoreAI Settings:

- ACM clock divider: 4
- AV0: 0–2 V analog input
- AV1: 0–4 V analog input
- AC1: current monitor
- AT2: temperature monitor
- AG3: software-driven
- AG4: software-driven
- VAREF: internal 2.56 V
- ADC mode: fixed 12-bit mode
- TVC: fixed to 0
- STC: register-controlled
- APB interface width: 16 bits

The above settings result in the memory address map shown in Table 10-2 on page 10-19.

The memory of the design in CoreConsole can be obtained from the following location:

<CoreConsole Project Directory>/SoftwareExport/<projectname>/memorymap.html

Table 10-2 • **Memory Map of the Example Design**

Address	Type	Width	Reset Value	Name	Description
base address + 0x00	Read/write	16	0x0	ACM_CTRL_STATUS	ACM Control Status Register
base address + 0x04	Read/write	16	0x0	ACM_ADDR	ACM Address Register
base address + 0x08	Read/write	8	0x0	ACM_DATA	ACM Data Register
base address + 0x0C	Read/write	16	0x0	ADC_CTRL_1	ACM Control Register 1
base address + 0x10	Read/write	16	0x0	ADC_CTRL_2	ACM Control Register 2
base address + 0x14	Read/write	16	0x0	ADC_CTRL_3	ACM Control Register 3
base address + 0x18	Read/write	16	0x0	ADC_CTRL_4	ACM Control Register 4
base address + 0x1C	Read/write	16	0x0	ADC_CTRL_5	ACM Control Register 5
base address + 0x20	Read-only	16	0x0	ADC_STATUS	ADC Status Register
base address + 0x24	Read-only	16	0x0	READ_FIFO_DATA_OUTPUT	Read FIFO Data Output
base address + 0x28	Read-only	8	0x0	READ_FIFO_STATUS	Read FIFO Status
base address + 0x2C	Read/write	16	0x0	IRQ_ENABLE	Interrupt Enable Register
base address + 0x30	Read-only	16	0x0	IRQ_STATUS	Interrupt Status Register

The initialization values for the ACM register space used to configure the Analog Quads can be calculated as discussed in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet or obtained from the *acmtable.vhd* (or *acmtable.v*) file. In this example, the ACM initialization is performed by using the APBWRT instruction to write desired values to the corresponding ACM register space. As discussed in "ACM Initialization Specific to CoreABC" on page 10-7, designers can take advantage of the APBWRT ACM command and write to the ACM registers with the values defined in the *acmtable.vhd* (or *.v*) file.

CoreABC Instruction Program

The CoreABC program in the Fusion Handbook design files executes the functionality depicted in Figure 10-7 on page 10-17. The CoreConsole project of this example can also be found in the Fusion Handbook design files. $Wait_10us is called whenever needed to ensure that the elapsed timing for TMSTB and CMSTB activation/deactivation is more than the minimum requirement. Inside the function, a loop counter is loaded with a count value and then decremented until it reaches zero. The count value should be chosen such that the elapsed time after exiting the function is 10 µs. Each of the instructions used in the $Wait_10us routine takes three clock cycles to execute. Assuming SYSCLK runs at 40 MHz, the count value can be obtained from EQ 10-4:

$$\text{Delay} = (3 + 3 + \text{count_value} \times (3 + 3) + 3) / f$$

EQ 10-4

where f = 40 MHz.

Instructions in this example are based on CoreABC v2.1. Some instructions may vary (e.g., loop instructions used in the $Wait_10us routine) if a different version of CoreABC is used.

In a real application design, the $Wait_10us routine can be replaced by a set of instructions that perform part of the design functionality during the required TMPWC or TMPWT period. This increases the processing performance of the design.

HDL Implementation

After building the core of the design in CoreConsole, the design has be to be completed in HDL where the microprocessor and the CoreAI system (along with any other peripherals in the system) are imported into Libero IDE in HDL format and connected to the rest of the design, to go through

the rest of the FPGA design flow. The HDL code in the Fusion Handbook design files is the top-level wrapper used in the example design.

In this example, the CoreABC instructions are stored in soft mode. Therefore, the instructions can be stored in the Fusion embedded flash memory or in external memory. The instructions are loaded into SRAM from the flash memory at power-up (initialization), and the microprocessor runs off the SRAM; CoreABC offers a feature to run directly off the flash memory as well. In the HDL code, Init_Block represents the initialization client of the Fusion embedded flash memory. An embedded CCC is used to generate two clock signals: a 10 MHz clock to drive the embedded flash memory and the initialization process to load the SRAM with CoreABC instructions, and a 40 MHz clock used as the system clock for the operational phase of the design.

Designing with the RTC

CoreAI enables you to interface with the Fusion RTC. When configuring CoreAI in CoreConsole, you need to specify the RTC and its parameters. CoreAI provides the necessary I/Os, as described in the *CoreAI Handbook*. ACM address space 0x40 to 0x58 is used to read or write specific RTC parameters, such as count or match values. Reading from and writing to these registers is no different from other ACM read/writes described in the "Analog Configuration MUX Initialization" section on page 10-6. The only difference is the targeted address of the ACM register space.

Part Number and Revision Date

Part Number 5100092-006-1

Revised December 2007

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.0)	Page
51700092-006-0	Corrected the operations given in the "Implementing Temperature Monitor Applications" section.	10-14

Rate This Document

11 – Interfacing with the Fusion Analog System: IP Interface

Fusion Analog System Soft IP Design

The Analog System Soft IP Design Flow is an IP-based design method for HDL designs that establishes a backbone to interconnect the Actel Fusion® FPGA fabric, the Analog System, the embedded flash memory block, and other peripherals. The Analog System soft IP includes an Analog-to-Digital Converter Sample Sequence Controller (ASSC) that sets up the ADC sample sequence, a System Monitor Evaluation Phase State Machine (SMEV) that compares the ADC results to user-defined threshold values, and a System Monitor Transition Phase State Machine (SMTR) that asserts threshold flags accordingly. Using the IP configuration catalog in Actel Libero® Integrated Design Environment (IDE), the user creates and configures the VHDL or Verilog Analog System soft IP along with the Flash Memory Analog System Client. The user can configure over/under threshold flags, acquisition time, filtering factor, and assert/deassert samples for analog inputs. More details are addressed in the "Basic Analog Block Settings" section. Once the IP is configured, the user instantiates the Analog System into HDL or builds the system in a SmartDesign project. Standard HDL design flow is used to complete the design, as described in *Fusion Design Solutions and Methodologies*.

When creating the Analog System in Libero IDE using the Analog System Builder (ASB), a configuration file is generated, and its data is stored in the spare pages within the embedded flash memory during FPGA programming. The Flash Memory Analog System Client is used to create the memory partitions to store this configuration data.

The Analog System uses the embedded flash memory to hold the nonvolatile configuration data for the analog subsystem. After power-up and during the initialization process, the flash memory is read and the data is stored in the Analog System's volatile register or RAM blocks within the analog subsystem. More information about the embedded flash memory system and clients is available in *Fusion Embedded Flash Memory Blocks*.

Note: Any time the Analog System is regenerated, the Analog System Client must also be regenerated from the Flash Memory System Builder in Libero IDE.

The Analog System Soft IP Design Flow offers a number of advantages to users. All the necessary soft IP cores are free. Sample sequence control, averaging/filtering, and threshold response functions are built-in and specified by an intuitive GUI in Libero IDE. IP configuration and connectivity are tightly integrated into SmartDesign and Libero IDE, enabling users to rapidly and seamlessly implement the complete analog and peripheral interface. Users do not have to write up their own code to control the analog and flash memory systems.

For processor- or microcontroller-based designs, a more efficient implementation can be realized through CoreAI (Analog Interface) and the methods discussed in *Interfacing with the Fusion Analog System: Processor/Microcontroller Interface*.

System Overview – Interface Components

Figure 11-1 gives an overview of the interface between the Analog System soft IP, the ADC, the clock circuitry, the device RAM, and the embedded flash memory. As shown in Figure 11-1, there are three Analog Interface soft IP blocks and several blocks that interact directly or indirectly with the Analog Interface soft IP blocks. The Analog Interface soft IP components are listed in Table 11-1 on page 11-3, and the components that interact with these soft IP components are listed in Table 11-2 on page 11-3. Detailed descriptions for each of the components listed in Table 11-1 on page 11-3 are included in the "SmartGen Soft IP Blocks" section on page 11-5.

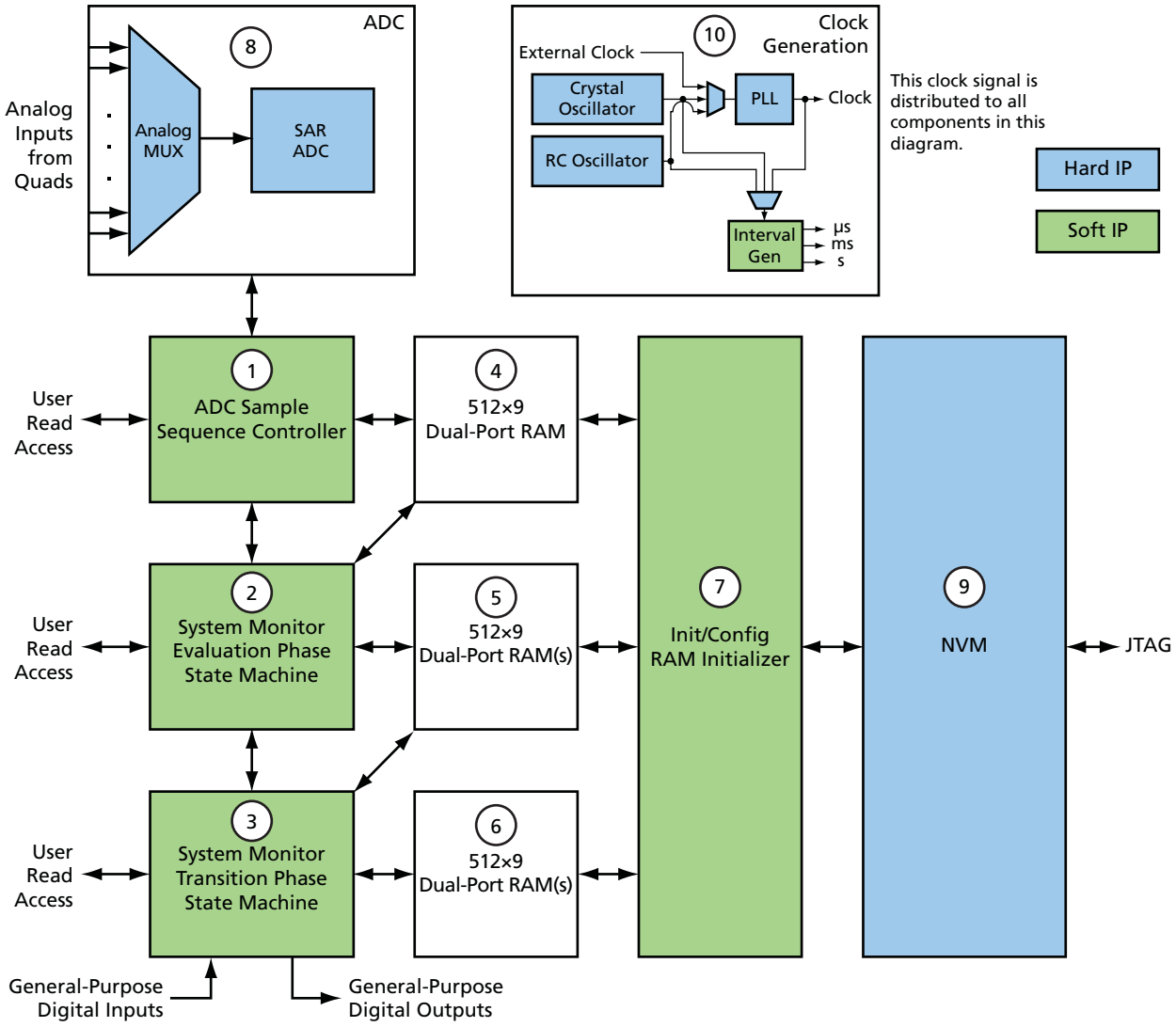


***Figure 11-1* • Fusion Interface Components (relative to Analog Interface soft IP)**

Table 11-1 • **Fusion Analog Interface Soft IP Components**

Number	Description
1	Analog-to-Digital Converter Sample Sequence Controller: The ASSC is a configurable sequencer that sets up the order of samples from the ADC, controls various measurement parameters of the ADC samples, and sends control commands to the ADC.
2	System Monitor Evaluation Phase State Machine: The SMEV reads ADC samples from the result locations in the ASSC RAM and compares the results to user-defined threshold values. Comparison results are stored in SMEV RAM.
3	System Monitor Transition Phase State Machine: The SMTR reads from SMEV RAM and, for each enabled channel, checks comparison results (previously calculated by the SMEV block) and generates the threshold flags defined by the user in the Analog System Builder.

Table 11-2 • **Fusion Components that Interact with Analog Interface Soft IP Components**

Number	Description
4, 5, 6	512×9 dual-port RAM blocks: These RAM blocks are used to store "program" sequences for the SMTR and SMEV. They are also used to store data samples calculated by the ADC and data values that control the operation of the ASSC. They are initialized by the Init/Config soft IP block, which transfers data from nonvolatile memory (NVM) after a system reset. Note that these RAM blocks can also be modified while the system is live to allow the user to perform real-time system debugging before committing resources to programming the NVM. This debugging feature is addressed by Synplicity® Identify software. For more information, refer to the Identify user's guide.
7	Init/Config RAM Initializer: The sole purpose of this soft IP block is to initialize the system RAM blocks after a system reset, by reading data from the NVM.
8	ADC: This analog-to-digital converter hard IP component is the main interface between the external analog voltage, current, and temperature sources, and the internal digital FPGA user logic (soft IP). It is selectable for 8-, 10-, or 12-bit operation.
9	NVM: The nonvolatile memory (flash) will be used, at minimum, to store program sequences for the ASSC, SMEV, and SMTR.
10	Clock Generation: Internal or external clock generation for digital system time base reference. The internal PLL, internal RC oscillator circuit, or external crystal oscillator circuit can be used in this process.

System Operation

The Analog Block (AB) System contains the Analog Block hard IP and the Analog Interface soft IP, which includes ASSC, SMEV, SMTR, and their corresponding SRAM blocks. The Flash Memory System contains the embedded flash memory hard IP block and the interface soft IP, or the Init/Config IP block. Figure 11-2 shows the generic connections between the Analog Block System and the Flash Memory System in the SmartGen soft IP design flow.

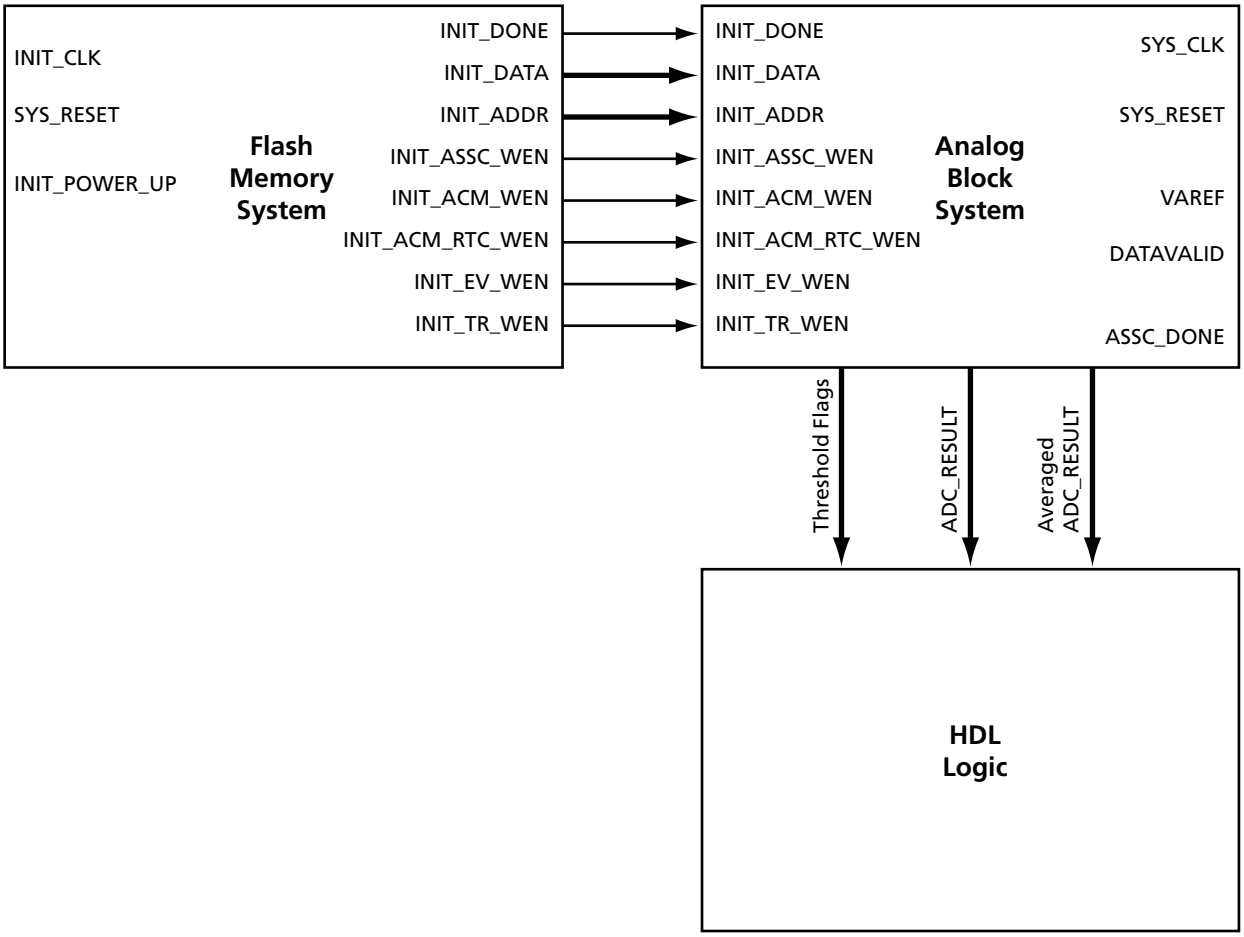


***Figure 11-2* • SmartGen Soft IP Design Flow – Generic Connections**

Initialization

The Init/Config soft IP is used to accomplish the initialization of the Fusion Analog Block. All user-defined Analog Block parameters are preprogrammed into the embedded flash memory and are loaded into the corresponding Analog System soft IP RAM blocks and Analog Configuration MUX (ACM) registers by the Init/Config IP during device power-up. For more information about Analog Client Initialization, refer to *Fusion Embedded Flash Memory Blocks*.

Sample and Convert

Once the initialization and calibration is done, or INIT_DONE and calibrate_o are asserted HIGH, the Analog System soft IP starts functioning. The ASSC IP controls the sample sequence, the SMEV applies a moving average to the ADC conversion output and compares the outputs with the preset threshold values, and the SMTR checks the comparison results and acts on them based on

predefined behavior. The user can also probe and process the ADC output directly and set up corresponding reactions. The two modes are described below.

Threshold Flags Operation Mode

The threshold values are preset during Analog System soft IP configuration in Libero IDE. The values are programmed into the embedded flash through the Analog System Client, then loaded into the SRAM during initialization. SMEV soft IP does the comparison between the average ADC results and the threshold values, and saves the comparison results to the SMEV SRAM. The SMTR soft IP reads the results from the SMEV SRAM and asserts or deasserts user-defined threshold flags, which are general-purpose outputs (GPOs) of the Analog System Block. The GPOs can be used by internal logic in the FPGA array, or they can trigger external I/Os directly.

ADC Result Direct Access Mode

Instead of reacting on the threshold value comparison result from the SMEV, users can directly access the ADC results and convert them to meaningful voltage, current, or temperature values and process these values for different purposes. To accomplish this, the user first needs to expose the necessary ports from the ASSC or SMEV IP through the advanced options in the Analog System Builder. The corresponding ports are listed in the ASSC and SMEV sections. Second, the user needs to build an interface to read out the valid ADC results from the ASSC or SMEV RAM. The basic logic of the interface is discussed in the CoreAI section of the *Interfacing with the Fusion Analog System: Processor/Microcontroller Interface* chapter. Sample code for fetching the ADC result is located in the "Sample Code" section on page 11-19. Once the ADC result has been fetched, it can be translated back to a voltage, current, or temperature value. Sample code for this is also provided in the "Sample Code" section on page 11-19.

SmartGen Soft IP Blocks

ADC Sample Sequence Controller (ASSC)

Function

The ASSC is a configurable sequencer that sets up the order of samples from the ADC, controls various measurement parameters of the ADC samples, and sends control commands to the ADC. It also contains multiplexer logic for reading from and writing to a 512×9 dual-port RAM block. In addition to controlling sequencing of ADC samples, the ASSC block performs digital post-scaling of the ADC samples, and ADC saturation detection, functions that were previously handled with additional soft IP blocks.

Note: The ASSC does not control the various prescaler and other signals within each Analog Quad (except for the current monitor and temperature monitor strobe connections). These are defined during soft IP configuration in Libero IDE and initialized via the Init/Config soft IP block into the ACM.

Interfaces

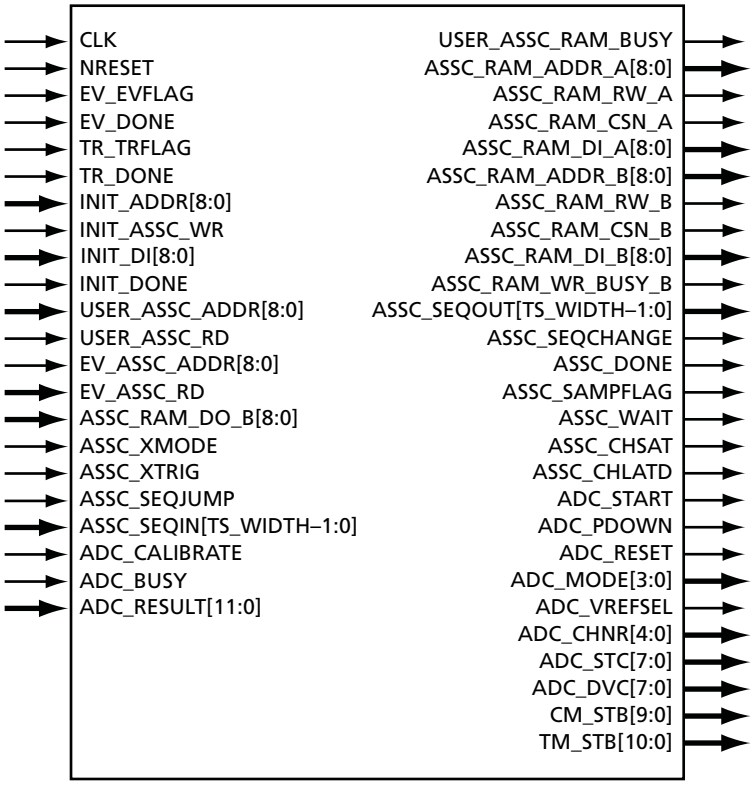


Figure 11-3 • **ASSC I/O Signal Diagram**

Note: All signals are active high (logic 1) unless otherwise noted. All port width specifications are in Verilog notation. Any alphabetic text within port width brackets indicates that the specified port width is controlled via a generic (VHDL) or parameter (Verilog) described in Table 11-3. All I/O signals are synchronous to the rising edge of the CLK signal unless otherwise noted.

The signals in Table 11-3 are the typical signals users should monitor in the simulation for the regular ADC conversion process.

Table 11-3 • **ASSC I/O Signal Descriptions**

Name	Type	Description
CLK	Input	System Clock: Reference clock for all internal logic (100 MHz maximum). This signal is connected to the top level as SYS_CLK.
NRESET	Input	Active-low asynchronous reset. This signal is connected to the top level as SYS_RESET.
INIT_ADDR[8:0]	Input	Init/Config RAM Address: These address signals come from the Init/Config soft IP block for writing to the 512×9 ASSC RAM.
INIT_DONE	Input	Init/Config Done: This static signal indicates that the Init/Config soft IP block has completed loading all of its clients (including the ASSC RAM block) from data stored in the internal Fusion NVM block(s).

Table 11-3 • **ASSC I/O Signal Descriptions (continued)**

Name	Type	Description
ASSC_DONE	Output	ASSC Done: This output indicates that the ASSC block has completed the current function and is either waiting for further action (e.g., the power-down or stop function) or is about to transition to the next sequence (e.g., the sample or calibration function). If the ASSC_DONE signal is active at the same time that the ASSC_SAMPFLAG signal is active, this indicates that a sample function has just completed and will cause the SMEV block to commence with evaluation sequences followed by transition sequences within the SMTR block; otherwise, if the ASSC_DONE signal is active and the ASSC_SAMPFLAG signal is inactive (logic 0), the SMEV block will not have evaluation sequences for the current sequence (timeslot). This output is connected to the SMEV block and may optionally be used external to the Analog Interface soft IP blocks by the user.
ADC_CALIBRATE	Input	ADC Calibration: This signal from the ADC indicates that internal calibration is currently in effect. This input connects to the calibrate_o output from the ADC.
ADC_RESULT[11:0]	Input	ADC Result: These signals comprise the conversion result from the ADC. In 12-bit mode, the ADC uses all bits; in 10-bit mode, it uses bits 11:2; and in 8-bit mode, it uses bits 11:4. All unused ADC bits are set to logic 0 when in 10-bit or 8-bit mode. These inputs connect to the result_o[11:0] outputs from the ADC.
The following signals are used for jump sequence control.		
ASSC_XMODE	Input	External Trigger Mode: If this input is logic 1, the ASSC uses the ASSC_XTRIG signal to transition to and complete the current sequence timeslot. If this input is logic 0 (default operation for automated sequencing), the internal timeslot counter is used to automatically advance to the next sequence number. This input can come from the SMTR (from one of the GPO signals) or user logic external to the Analog Interface soft IP blocks, or can be statically tied off to logic 0 or logic 1.
ASSC_XTRIG	Input	External Trigger: If the ASSC_XMODE input is logic 1 and this input is held at logic 1 for exactly one clock cycle, the ASSC block will transition to and complete the current sequence. If the ASSC_XMODE input is logic 0 (default operation for automated sequencing), this input is ignored. This input can come from the SMTR (from one of the GPO signals) or user logic external to the Analog Interface soft IP blocks, or can be statically tied off to logic 0. If this signal is used to control external triggering, the user should monitor the ASSC_DONE signal to know after which point the ASSC_XTRIG will again have effect.
ASSC_SEQJUMP	Input	Sequence Jump Enable: Setting this signal to logic 1 jumps to the sequence number indicated on the ASSC_SEQIN[TS_WIDTH–1:0] input pins after the current sequence timeslot has completed. This input can come from the SMTR (from one of the GPO signals) or user logic external to the Analog Interface soft IP blocks, or can be statically tied off to logic 0.

Table 11-3 • **ASSC I/O Signal Descriptions (continued)**

Name	Type	Description
ASSC_SEQIN[TS_WIDTH–1:0]	Input	Sequence Number In: These inputs are used in conjunction with the ASSC_SEQJUMP signal to jump to a particular sequence number from the current sequence after the current sequence timeslot has completed. The SMTR sets these signals. These inputs can come from the SMTR (from several of the GPO signals) or user logic external to the Analog Interface soft IP blocks, or can be statically tied off to any combination of logic 0 and logic 1 values.
ASSC_SEQOUT[TS_WIDTH–1:0]	Output	Sequence Number Out: These outputs denote the current sequence timeslot. The SMEV block uses these signals. These outputs are connected to the SMEV block and may optionally be used external to the Analog Interface soft IP blocks by the user.
ASSC_SEQCHANGE	Output	Sequence Change: This output indicates that the outputs ASSC_SEQOUT[TS_WIDTH–1:0] will change after the very next rising edge of CLK. This output is connected to the SMEV block and may optionally be used external to the Analog Interface soft IP blocks by the user.
Users should monitor the following signals if they want to access the ADC conversion results from the ASSC RAM.		
USER_ASSC_RAM_BUSY	Output	ASSC RAM Busy: This output signal indicates that either the Init/Config soft IP block or the SMEV soft IP block is busy accessing the A-port of the ASSC RAM. This signal can optionally be used by user logic external to the Analog Interface soft IP blocks, or can be left unconnected if unused.
USER_ASSC_ADDR[8:0]	Input	User RAM Address: These address signals can be controlled by the user to allow read access from the A-port of the 512×9 ASSC RAM. If unused, these signals should be tied off to logic 0 or logic 1.
USER_ASSC_RD	Input	User RAM Read Enable: This control signal can be controlled by the user to allow read access from the A-port of the 512×9 ASSC dual-port RAM (the user will need to connect to the ASSC_RAM_DO_A[8:0] port for read data). If unused, this signal should be tied off to logic 0. **The user must ensure that the ASSC_RAM_BUSY signal is inactive at logic 0 while this signal is activated; otherwise, the data read from the A-port of the ASSC RAM will not be from the USER_ASSC_ADDR[8:0] address.**
ASSC_RAM_DI_A[8:0]	Output	ASSC RAM Write Data: These signals are connected to the A-port data inputs (write data) of the 512×9 ASSC RAM.

System Monitor Evaluation Phase State Machine (SMEV)

Function

The SMEV reads ADC samples from the result[11:0] locations in the ASSC RAM after each channel sequence has been processed by the ASSC block, and performs evaluation processing on the ADC samples. The SMEV block performs digital low-pass filtering of these samples, compares the low-pass-filtered samples against compare thresholds, and writes the results back into one or more 512×9 dual-port RAM tiles (the number of 512×9 RAM tiles required will depend upon how many program sequences are required for each application). Although the SMEV block deals with samples from the ADC, there is no direct link between it and the ADC; the ASSC block writes all raw ADC samples into the ASSC 512×9 dual-port RAM, which the SMEV reads during the evaluation phase.

Interfaces

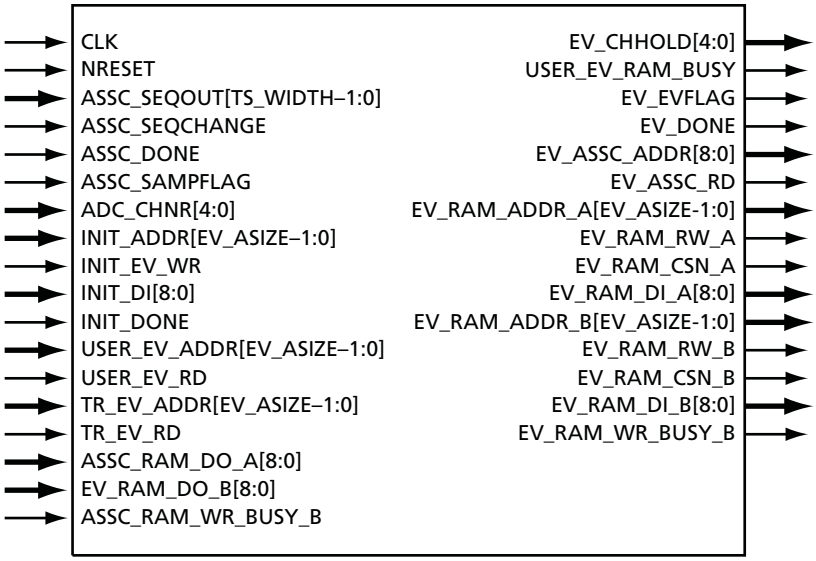


Figure 11-4 • **SMEV I/O Signal Diagram**

Note: All signals are active high (logic 1) unless otherwise noted. All port width specifications are in Verilog notation. Any alphabetic text within port width brackets indicates that the specified port width is controlled via a generic (VHDL) or parameter (Verilog) described in Table 11-4 on page 11-10.

To access the ADC conversion results from the SMEV RAM, users should monitor the signals in Table 11-4 on page 11-10.

Table 11-4 • **SMEV I/O Signal Descriptions**

Name	Type	Description
USER_EV_ADDR[EV_ASIZE–1:0]	Input	User RAM Address: These address signals can be controlled by the user to allow read access from the 512×9 SMEV RAM(s). If unused, these signals should be tied off to logic 0 or logic 1.
USER_EV_RD	Input	User RAM Read Enable: This control signal can be controlled by the user to allow read access from the A-port of the 512×9 SMEV dual-port RAM(s) (the user will need to connect to the EV_RAM_DO_A[8:0] port for read data). If unused, this signal should be tied off to logic 0. **The user must ensure that the USER_EV_RAM_BUSY signal is inactive at logic 0 while this signal is activated; otherwise, the data read from the A-port of the SMEV RAM(s) will not be from the USER_EV_ADDR[EV_ASIZE–1:0] address.**
USER_EV_RAM_BUSY	Output	SMEV RAM Busy: This output signal indicates that either the Init/Config Soft IP block or the SMTR block is busy accessing the A-port of the SMEV RAM(s). This signal can optionally be used by user logic external to the Analog Interface soft IP blocks or can be left unconnected if unused.
ASSC_RAM_WR_BUSY_B	Input	ASSC Busy Writing: This active-high signal indicates that the ASSC block is busy writing to the B-port of its dual-port RAM (this input is only used for non-Fusion/-ProASIC®3 technology implementation, such as ProASIC PLUS®, which has no true dual-port RAM).
EV_RAM_WR_BUSY_B	Output	SMEV Busy Writing: This active-high signal is for user status monitoring and indicates that the SMEV block is busy writing to the B-port of its dual-port RAM. It must be connected to the SMTR block for non-Fusion/-ProASIC3 technology implementation, such as ProASIC PLUS; otherwise, it can be left unconnected.
EV_RAM_DI_A[8:0]	Output	SMEV RAM Write Data: These signals are connected to the A-port data inputs (write data) of the 512×9 SMEV RAM(s).

System Monitor Transition Phase State Machine (SMTR)

Function

For each enabled channel, the SMTR checks comparison results previously calculated by the SMEV block and stored in the SMEV RAM, and asserts or deasserts different user-defined threshold flags, which are general purpose outputs (GPOs) of the Analog System Block. The GPOs can be used by internal logic in the FPGA array, or they can trigger external I/Os directly.

Interfaces

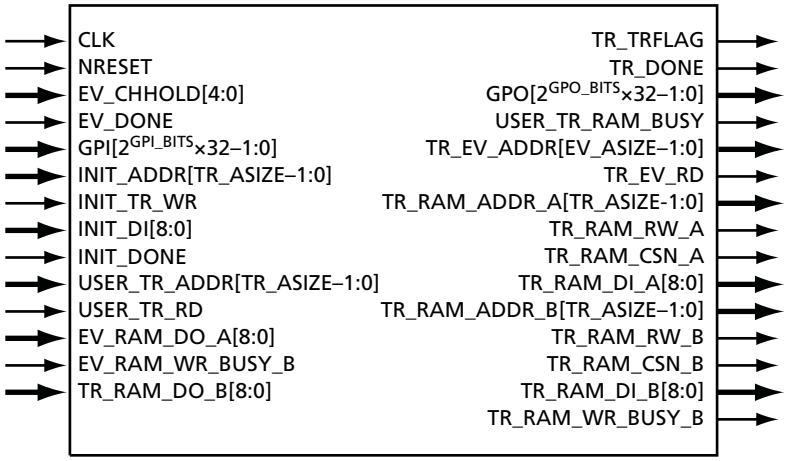


Figure 11-5 • **SMTR I/O Signal Diagram**

Note: All signals are active high (logic 1) unless otherwise noted.

Basic Analog Block Settings

In the Analog System Builder main window, the user can enter system clock frequency and ADC resolution. The system clock is used to drive the ASSC, SMEV, and SMTR soft IP blocks. Also, ADC_CLK is derived from the system clock, and has to be equal to or less than 10 MHz. The ADC block has a built-in divider (4×, minimum divider = 4) to automatically divide the system clock into the appropriate ADC_CLK range. Users can achieve maximum rate for ADC_CLK (10 MHz) by selecting a system clock frequency of 40 MHz or 80 MHz. For more information about the ADC sample rate and accuracy, refer to the *Analog-to-Digital Converter Background* section in the *Designing the Fusion Analog System* chapter.

Refer to the *Fusion Starter Kit User's Guide and Tutorial* for a sample design implementing the following settings.

AV Parameter Settings

To configure a voltage monitor, the user can choose to use either direct analog input or prescaled input.

Direct analog input limits the input voltage to less than the VAREF voltage. Usually, it is 2.56 V if the internal reference voltage is chosen, or it can be 0 to 3.3 V if an external reference voltage is used.

For example, if internal VAREF (2.56 V) is selected, choosing a direct analog input to sample a signal that swings between 0 to 2.56 V can avoid the gain and offset error that could be introduced by the prescaler.

However, if the input voltage is too small (0 V – 0.2 V) compared to 2.56 V, and if direct input is used, resolution will be degraded. At this time, a prescaler should be used to amplify the signal for better resolution.

If the input signal is greater than VAREF, the prescaler must be used to scale down the input range before the ADC can sample and convert it.

Once the ADC finishes converting the analog signal to a digital value, it filters (averages) the resulting digital output. Digital filtering is a single-pole low-pass filter built in soft gates, that can be used to improve the signal-to-noise ratio. If the ADC input data is very erratic, the filtering will smooth out the input and reduce the noise.

The filtered value is calculated using EQ 11-1:

$$\text{Filtering_result}_n = \text{filtering_result}_{n-1} + (\text{ADC_Result}_n / \text{filtering_factor}) - (\text{filtering_result}_{n-1} / \text{filtering_factor})$$

EQ 11-1

If the digital filtering factor is set to 1, it is ignored.

In some cases where the inputs have very low frequency and the electrical environment is not very noisy, it may be possible to proceed without any special filtering of input analog signals. However, in most applications it is desirable to at least implement a simple post-conversion digital filter inside the FPGA by oversampling and averaging several results to reduce the effects of random noise in the conversion signal path and improve overall accuracy. This simple averaging is automatically handled in the software by setting the digital filtering factor in the Analog System Builder to specify how many samples are averaged (when the factor = *N*, 2^N samples are averaged together).

For situations where greater accuracy is required, an external analog filter may be needed to eliminate non-random and out-of-band noise sources. If an analog filter is not used to restrict the input signal content to the band of interest, any out-of-band signals or noise will be aliased into the conversion result as random in-band noise.

Some applications—for example, those that require frequency detection—may need both external analog filtering to limit out-of-band effects, and more sophisticated digital processing such as a multi-tap Finite Impulse Response (FIR) filter. A wide variety of digital filtering methods are available through the FPGA gates available in a Fusion device.

Once a digital filter factor is selected, the Initial Value option is activated. This initial value is used for simulation purposes. The user can preset an initial value to imitate a real situation. For example, let the input signal be a 3.3 V power supply that fluctuates around 3.3 V with a range of 50 mV. The user can set 3.3 V as the initial value for simulation mode.

Acquisition time defines how much time the user gives the ADC to conduct the sampling and conversion. If the acquisition time is too short, the input signal may not even be settled yet, and the ADC will just sample some invalid signals. The recommendation is at least 0.2 µs for direct analog input, and 10 µs for the prescaled input. Refer to the *Analog-to-Digital Converter Background* section in the *Designing the Fusion Analog System* chapter for more info on acquisition time.

Maximum voltage defines the expected maximum input voltage on this particular channel.

Users can set threshold flags for SMEV IP to compare against the input voltage, and SMTR IP will trigger the corresponding flags based on the SMEV comparison results.

The Assert Samples and Deassert Samples parameters define after how many consecutive events a flag should be asserted or deasserted.

AC Parameter Settings

Besides all the parameters discussed in the voltage monitor configuration, a current monitor acquisition time should be at least 5 µs. Users should also define the signal polarity and make sure that the potential on the adjacent AV pad MUST be greater than the AC pad.

A realistic sense resistor value (0.005 – 100 Ω) should be entered in the AC peripheral configuration window. The adjacent AV channel in the same Analog Quad can still be used as a voltage monitor.

AT Parameter Settings

Similar to the current monitor, a 5 µs acquisition time and digital filter factor value of greater than 512 are recommended for better conversion accuracy.

Sample Sequence Setting

Users can choose to sample some or all of the analog channels. To manually adjust the sample order, select **Allow manual modification of operating sequence** in the sample sequence configuration window.

The last operation should always jump back to the main procedure or jump to another procedure.

Package Pin Assignment

Users can assign the package pin number for the AV/AC/AT or gate driver peripherals in the Analog System Builder window, and this assignment will be honored in the Designer software.

Soft IP Implementation Options

Default Implementation (ASSC, SMEV, and SMTR)

The default implementation for the Analog System soft IP includes the ASSC, SMEV, and SMTR IP blocks for the Fusion Analog System and the Init/Config IP block for the Fusion flash memory system. This section focuses on the Analog Block soft IP. The Init/Config IP is discussed in the *Using the Embedded Flash Memory for Initialization* section in *Fusion Embedded Flash Memory Blocks*.

Figure 11-6 is a view of the IP and the RAM blocks in the Analog System. The datapath of the system is shown in the figure (MUXes are not shown for clarity).

Dual-port RAMs are used in the soft IP. All IP blocks access their corresponding RAMs through PORTB. All IP blocks access each other's RAM through PORTA. All user access is through PORTA.

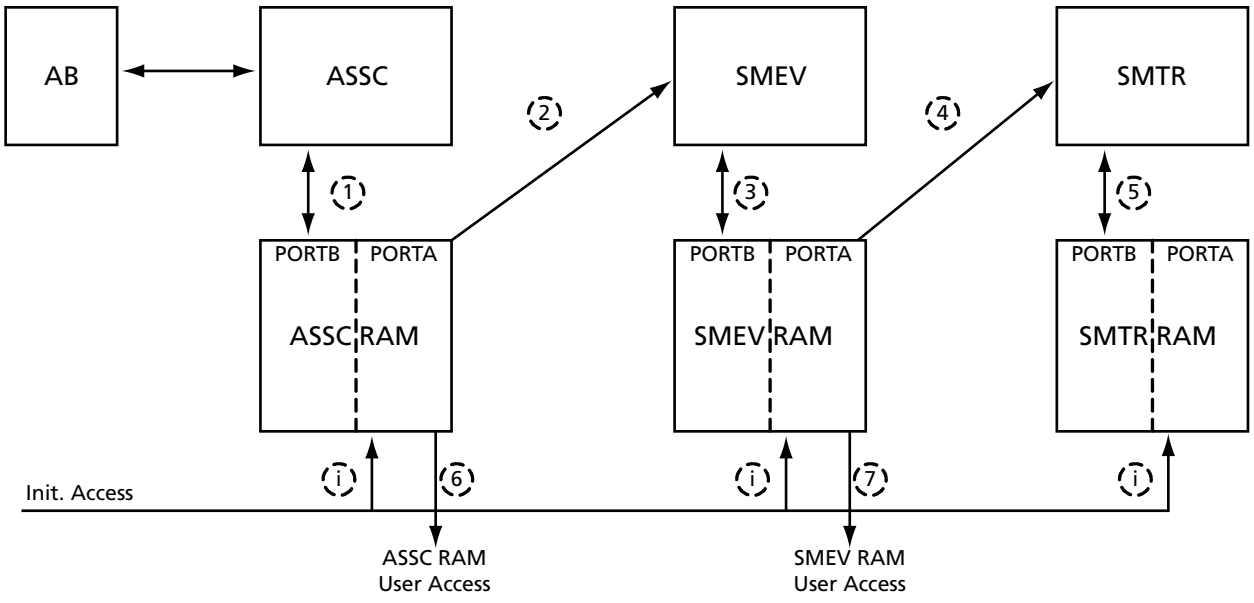


***Figure 11-6* • Analog System – IP and RAM Blocks**

The sequence of events for processing ADC data is as follows:

1. ASSC reads its opcode from the ASSC RAM for slot N processing.

2. ASSC processes ADC for slot N.
3. ASSC completes slot N processing and writes ADC result value to ASSC RAM.
4. ASSC signals DONE.
5. SMEV wakes up and begins reading SMEV RAM for opcodes.
6. ASSC reads its opcode from the ASSC RAM for slot N + 1 processing.
7. SMEV reads ASSC RAM for ADC result value of slot N processing.
8. The SMEV and SMTR state machines do not execute in parallel. The SMEV state machine finishes its processing and then signals the SMTR to begin.

Figure 11-7 is the simulation result illustrating the soft IP events:

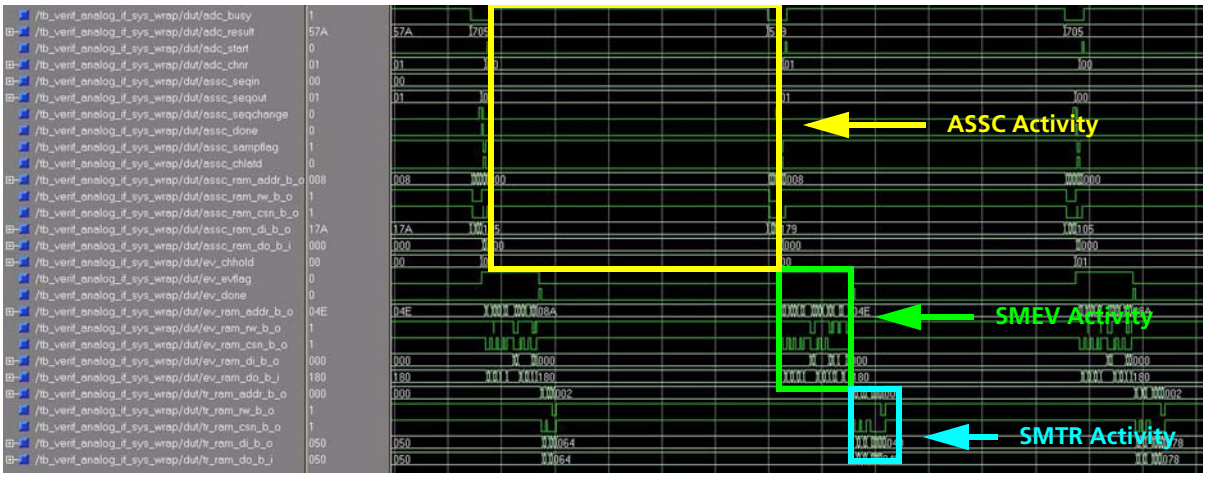


Figure 11-7 • **Simulation Result Illustrating Soft IP Events**

Figure 11-8 shows the soft IP process for multiple channels in a pipeline mode:

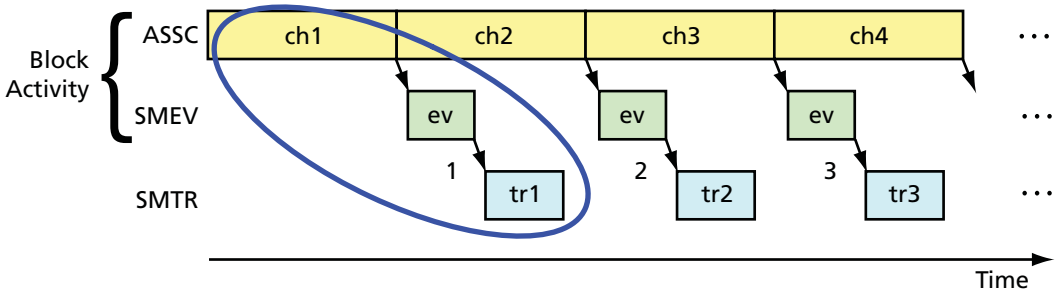


Figure 11-8 • **Soft IP Process for Multiple Channels in Pipeline Mode**

Use Default Implementation and Expose ADC Result (ASSC I/Os, SMEV I/Os, and ACM I/Os)

When users want to directly access the raw ADC results from the ADC or from the ASSC RAM, or access the averaged ADC result from the SMEV RAM, they can select the corresponding ports in the Advanced Options window. They will need to develop corresponding HDL code to access those interfaces. Users can also export and access the ACM bus, as described in the *Analog Configuration MUX (ACM)* section in the *Interfacing with the Fusion Analog System: Processor/Microcontroller Interface* chapter. For sample HDL code, refer to the *Fusion Starter Kit User's Guide and Tutorial*.

User Accessing ADC_RESULT Directly

To read ADC_RESULT from the ADC directly, the following signals should be monitored closely:

ASSC_DONE (active-high)

DATAVALID (active-high)

ADC_CHNUMBER

ADC_RESULT

Figure 11-9 shows the timing relationship among the four signals listed above.

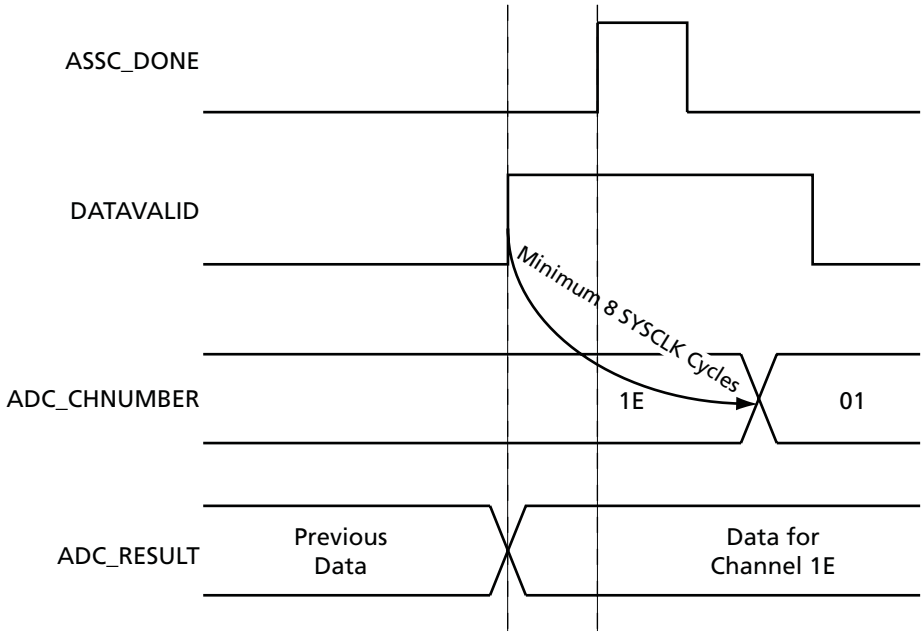


***Figure 11-9* • How to Read Valid ADC_RESULT**

ASSC_DONE assertion determines which channel data is available on the ADC_RESULT bus.

DATAVALID assertion indicates that the new ADC_RESULT is ready.

When both ASSC_DONE and DATAVALID are asserted, the user should record which channel number is active and then read ADC_RESULT for that channel.

The time elapsed from the rising edge of DATAVALID to the next channel number change is at least eight SYSCLK cycles. If there is concern that ADC_RESULT may not be read out and processed as fast as the channel changes, ADC_RESULT and ADC_CHNUMBER should be latched—this is one of the functions of the ASSC RAM.

ASSC and User Accessing ASSC RAM

A signal is exported to indicate that the ASSC is reading or writing the SRAM (USER_ASSC_RAM_BUSY). The user should not access this interface while that is occurring.

SMEV and User Accessing ASSC RAM

The USER_ASSC_RAM_BUSY signal indicates that the SMEV is accessing the RAM, and as stated in the documentation, the user should not access this interface while that is occurring.

SMEV and User Accessing SMEV RAM

A signal is exported to indicate that the SMEV is reading or writing the SRAM (USER_EV_RAM_BUSY). This indicates to the user to stop reading.

SMTR and User Accessing SMEV RAM

The USER_EV_RAM_BUSY signal indicates that the SMTR is accessing the RAM, and as stated in the documentation, the user should not access this interface while that is occurring.

In general, the USER_ASSC_RAM_BUSY or USER_EV_RAM_BUSY signal indicates that other IP is accessing the corresponding RAM, and the user should not access the interface while that is occurring. In other words, users can only access the RAM content while the BUSY signal is deasserted.

For a given channel, the ADC result is saved as two parts in two consecutive address locations inside the RAM. For example:

```
*******************************************************************************
            ASSC Memory Content Report
*******************************************************************************
Slot   Channel   Address  Bits       Value
-------------------------------------------------------------------------------
0      AV0
                  3|       [08:00]|   Raw ADC Result [08:00]
                  4|       [02:00]|   Raw ADC Result [11:09]
-------------------------------------------------------------------------------


*******************************************************************************
            SMEV Memory Content Report
*******************************************************************************
       Channel   Address  Bits       Value
-------------------------------------------------------------------------------
       AV0
                  75|      [08:00]|   Averaged ADC Result [08:00]
                  76|      [02:00]|   Averaged ADC Result [11:09]
-------------------------------------------------------------------------------
```

When the BUSY signal is deasserted, the user must execute two User Reads (RD1 and RD2) to read out the whole ADC result. If the BUSY signal is asserted after RD1 but before RD2, the user must re-execute RD1 to read the lower address data once the BUSY signal is deasserted again, followed by RD2 to read the higher address data. Meanwhile, the user must control the address increment appropriately. For sample HDL coding on this topic, refer to the "Sample Code" section on page 11-19. This code is also used in the Fusion Starter Kit tutorial design example.

The user interfaces to the ASSC RAM or the SMEV RAM are the same. The glue logic to the interface should keep monitoring the BUSY signal and sending the right RAM address at the appropriate time, then execute the READ action. For more details, refer to the timing diagrams in Figure 11-10 on page 11-17 and Figure 11-11 on page 11-17.

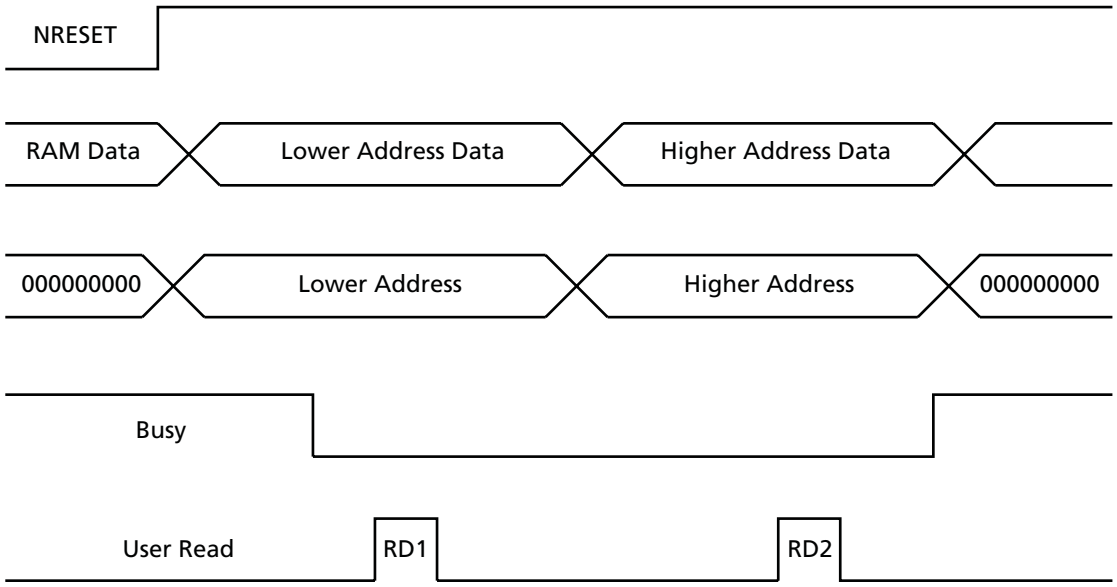


***Figure 11-10* • User Read Soft IP Block RAMs**

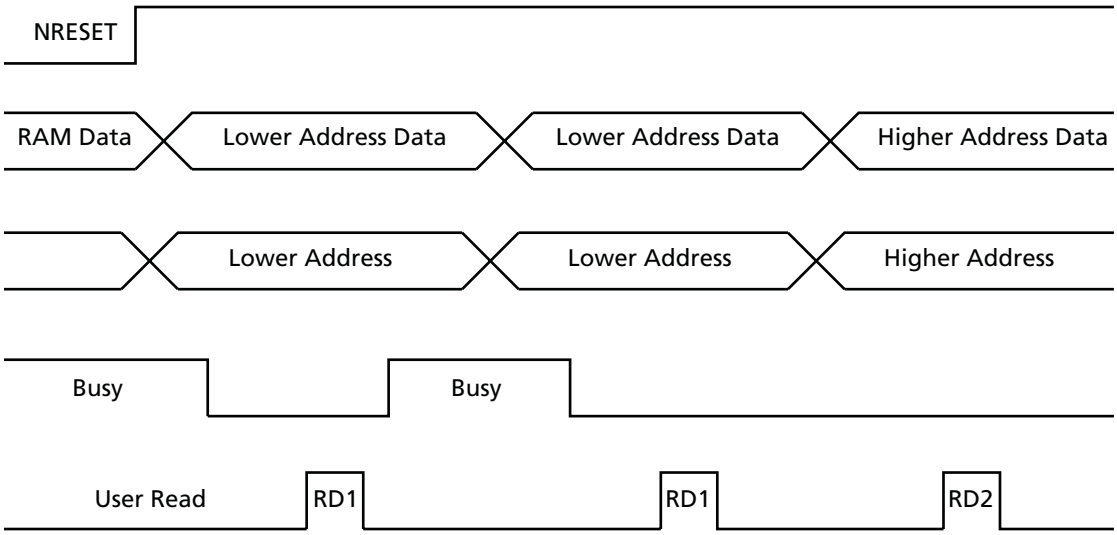


***Figure 11-11* • User Read Soft IP Block RAMs (continued)**

Use IP Cores for ADC Sequence Control Only

When users do not need the SMEV (averaging ADC results) and SMTR (triggering threshold flags) functions, but only need the ADC sequence control function (ASSC IP block), they can select the **IP cores for ADC sequence control only** option in the Advanced Options window. Additionally, if they want to directly access the raw ADC result from the ADC or from the ASSC RAM, or access the ACM bus, they can select the corresponding ports in the Advanced Options window.

VAREF Capacitor Value Selection

The Fusion device can be configured to generate a 2.56 V internal reference voltage (VAREF) that can be used by the ADC. When VAREF is internally generated by the Fusion device, a by-pass capacitor must be connected from this pin to ground. For more information, please refer to the "Pin Description" section of the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet.

In the Smartgen Analog System Builder, under Advanced Options, users can select the capacitor value based on the system level requirements. Depending on the capacitor value, a delay circuitry will be automatically added to ensure the Smartgen IP will not perform an ADC conversion until the voltage level on the VAREF is stable. Table 11-5 shows the corresponding settling time based on the selectable capacitor value in the software. If the capacitor value is different than the selectable values in the software, the user should pick the next higher capacitor value to ensure sufficient settling time is added. The additional delay will significantly increase the simulation time, but the user may consider changing the resolution of the simulator to speed up the simulation process.

Table 11-5 • **Settling Time for Using the Internal VAREF Reference Voltage**

Capacitor Value (µF)	Settling Time (ms)
3.3	116.45
10.0	324.73
22.0	750.24

Note: Users who are using the standalone AB macro and build their own control interface need to be aware that there is no hold time check in SmartTime for the following signals: VAREFSEL, TVC, STC, MODE, CHNUMBER. Users need to ensure these signals remain stable for at least one clock cycle after the assertion of ADCSTART (see Figure 11-12) and the simulation model provides a check to ensure this requirement is met.

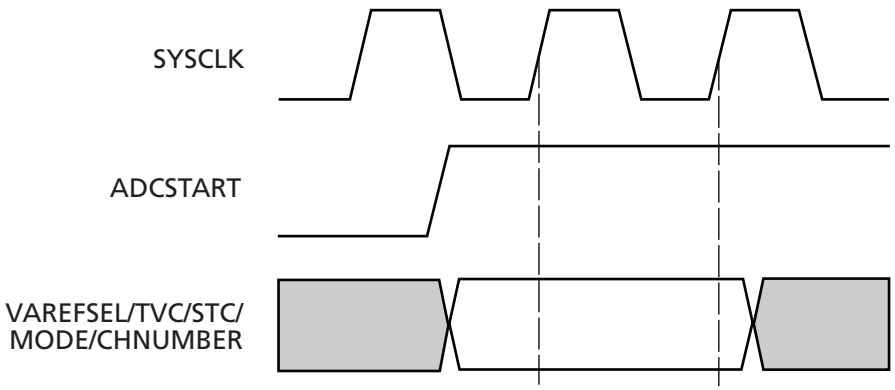


Figure 11-12 • **ADCSTART Timing Diagram**

Analog Configuration MUX (ACM)

The ACM is an interface between FPGA/JTAG test registers and Analog Quad configuration latches, and the Real-Time Counter (RTC). The Analog Block consists of four 8-bit latches per Analog Quad, which get initialized through the ACM. These latches act as configuration bits for Analog Quads. The Analog System soft IP generated from Libero IDE configures the ACM latches automatically. If the user does not use the soft IP, the ACM can be configured manually. For more information, refer to the *Analog Configuration MUX (ACM) Initialization* in the *Interfacing with the Fusion Analog System: Processor/Microcontroller Interface* chapter. The ACM block runs off the core voltage supply (1.5 V).

Sample Code

The following sample VHDL code is used to read out the ADC conversion results from either the ASSC RAM or the SMEV RAM.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity ram_reader is
  port (
    CLK         :  in std_logic; -- clk
    NRESET      :  in std_logic; -- active low reset

    ASSCDONE    :  in std_logic;
    EVDATA      :  in std_logic_vector( 8 downto 0 );
    EVBUSY      :  in std_logic;

    EVADDR      :  out std_logic_vector( 8 downto 0 );
    EVRD        :  out std_logic;

    AVGDATA     :  out std_logic_vector( 11 downto 0 )
  );
end ram_reader;

architecture ctrl of ram_reader is

-- constants for ACM addresses
constant EV_AVG_DATA1 : std_logic_vector( 8 downto 0 ) := "001001011"; --75d
constant EV_AVG_DATA2 : std_logic_vector( 8 downto 0 ) := "001001100"; --76d

-- state machine
type fsm_type is ( IDLE,
          RDWAIT,
          RD1,
          RD2
         );

signal state        : fsm_type;

-- internal registers to hold match values
signal avgdata_reg      : std_logic_vector( 11 downto 0 );




-- internal signals
signal evrd_i      : std_logic;
signal evaddr_i     : std_logic_vector( 8 downto 0 );
signal ev_d1en, ev_d2en : std_logic;

begin

-- toplevel port maps
EVADDR  <= evaddr_i;
EVRD   <= evrd_i;
AVGDATA <= avgdata_reg;

--------------------------------------------------------------------------------
-- EV AVG DATA REGISTER
--------------------------------------------------------------------------------
process(CLK, NRESET)
begin
  if NRESET = '0' then
    avgdata_reg <= (others=>'0') ;
```

```
  elsif rising_edge(CLK) then
    if ( ev_d1en = '1' ) then
      avgdata_reg( 8 downto 0 ) <= EVDATA( 8 downto 0 );
    else
      avgdata_reg( 8 downto 0 ) <= avgdata_reg( 8 downto 0);
    end if;

    if ( ev_d2en = '1' ) then
      avgdata_reg( 11 downto 9 ) <= EVDATA( 2 downto 0 );
    else
      avgdata_reg( 11 downto 9 ) <= avgdata_reg( 11 downto 9 );
    end if;

  end if;
end process;

-------------------------------------------------------------------------------
-- MAIN STATE MACHINE
-------------------------------------------------------------------------------
process(CLK, NRESET)
begin
  if NRESET = '0' then
    evaddr_i <= "000000000";
    evrd_i <= '0';
    ev_d1en <= '0';
    ev_d2en <= '0';
    state <= IDLE ;
  elsif rising_edge(CLK) then
    case state is
      when IDLE =>
        ev_d2en <= '0';
        if ( ASSCDONE = '1' ) then
          state <= RDWAIT;
        else
          state <= IDLE;
        end if;
      when RDWAIT =>
        if ( EVBUSY = '0' ) then
          evaddr_i <= EV_AVG_DATA1;
          evrd_i <= '1';
          state <= RD1;
         end if;
      when RD1 =>
        if ( EVBUSY = '1' ) then
          state <= RDWAIT;
        else
          evaddr_i <= EV_AVG_DATA2;
          ev_d1en <= '1';
          evrd_i <= '1';
          state <= RD2;
        end if;
      when RD2 =>
        if ( EVBUSY = '1' ) then
          state <= RD1;
        else
          evaddr_i <= "000000000";
          evrd_i <= '0';
          ev_d1en <= '0';
          ev_d2en <= '1';
          state <= IDLE;
        end if;
      when others =>
        state <= IDLE;
    end case;
  end if;
```

```
end process;


end ctrl;
```

The following VHDL code is used to translate the ADC result back to a voltage, current, or temperature value.

```
scale: process (reset_n, clock)
begin
if reset_n = '0' then
  scaled_input <= (others => '0');
elsif clock'event and clock = '1' then
  case format_select is
    when "VOLTAGE" =>
      -- 8V full scale, display volts
      -- need to multiply ADC counts x2
      scaled_input <= "000" & counts_in & '0';
    when "TEMPERATURE" =>
      -- drop two LSBs to read in deg K
      scaled_input <= "000000" & counts_in_int_int(11 downto 2);
    when others =>
      null;
  end case;
end if;

end process;
```

Note: For the current conversion, the ADC result is the differential voltage value across the current sense resistor. To get the final current value, divide the differential voltage value by the sense resistor value.

Part Number and Revision Date

Part Number 51700092-007-3

Revised November 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.2)	Page
v1.1 (September 2008)	EQ 11-1 was revised to correctly subscript portions of variable names.	11-12
51700092-007-1	Modified ADC_RESULT[11:0] description.	11-7
51700092-007-0	Added "VAREF Capacitor Value Selection" section.	11-18

12 – Temperature, Voltage, and Current Calibration in Fusion FPGAs

Introduction

Actel Fusion® mixed-signal FPGAs integrate configurable analog features, including I/Os, prescalers, low-pass filters, and an analog-to-digital converter (ADC), enabling customers to perform temperature, voltage, and current measurements in their applications. Analog components have a specific accuracy for a given set of conditions. The accuracy can have a broad range of definitions and is affected by many parameters in the system. For example, in a temperature measurement application, the accuracy of the measured temperature is influenced by the accuracy of on-chip elements (temperature sensor, op amps, and ADC), use model (sample rate, ADC resolution setting, post-processing, etc.), and board-level considerations. For the purpose of this document, accuracy can be defined as the difference/error between the actual value and the measured value. For example, in a temperature measurement application, an accuracy of ±2°C means that the measured value may be up to ±2°C different from the actual value.

If the difference between the measured value and the actual value is too great, you can use calibration to bring the measured value closer to the actual value. Calibration assumes a profile for the relationship between the actual value and the measured value. This profile depends on the characteristics of the components used in the measurement. There are two calibration profiles: one corrects for offset error only, and the second accounts for both offset and gain errors. Figure 12-1 illustrates these typical profiles that define the calibration implementation methodology.

To completely calibrate a system, users can calibrate individual components, or they can calibrate the entire system, taking into account the error of all the individual components working together. Many users may decide to perform both levels of calibration. This document provides a description of the factory calibration methodology for voltage inputs on Fusion devices, and also provides recommendations on system calibration methods for voltage, temperature, and current using Fusion. Using Actel's device calibration solution, the Fusion ADC sampling accuracy for voltage prescalar inputs can be improved to 1%, enabling Fusion to better meet customers' design requirements.

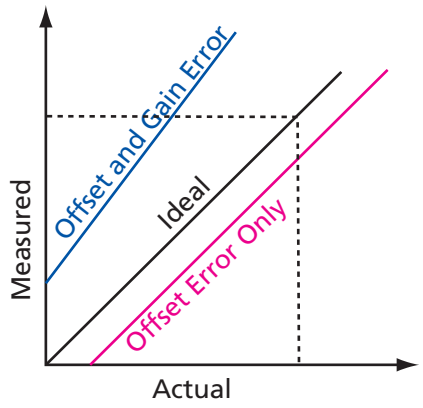


***Figure 12-1* • Example Profiles of Measured Value and Actual Value Variations**

General Calibration Concept

Calibration Methods

Based on the measured-versus-actual profile and the required accuracy, customers can define the most efficient method for translating the measured value into an actual value. In most analog components, including the Fusion FPGA, the relationship between measured and actual values follows the profiles illustrated in Figure 12-1 on page 12-1. This document describes only calibration methodologies associated with offset-only and offset-and-gain corrections.

Offset-Only Calibration

Offset-only calibration (sometimes known as one-point calibration) is based on the relationship between the measured and actual values given in EQ 12-1:

$$y = x + c$$

EQ 12-1

where

y = actual value
x = measured value
c = offset compensation between measured and actual values

As shown by EQ 12-1, offset-only calibration accounts for the offset between the actual and measured values.

Offset-and-Gain Calibration

If the correlation between the actual and measured values is defined primarily by an offset, as shown in the offset error line in Figure 12-1 on page 12-1, or if the actual measured value is naturally constrained to a specific region, offset-only calibration may be sufficient to achieve a high degree of accuracy.

However, in some cases, especially if the range of the measurement varies widely, the difference between the actual and measured values not only includes an offset but is also governed by a gain variation, as shown in Figure 12-1 on page 12-1. In such cases, offset-only calibration may not provide sufficient correction to achieve the accuracy required by the customer's application. In this case, offset-and-gain calibration (also known as two-point calibration) can be implemented to achieve a higher level of accuracy.

In two-point calibration, the relationship between the measured and actual values is governed by EQ 12-2:

$$y = mx + c$$

EQ 12-2

where

y = actual value
x = measured value
c = offset compensation between measured and actual values
m = gain compensation between measured and actual values

Choosing between one-point calibration and two-point calibration depends on many parameters, some of which include the following:

- Customer application's required accuracy
- Measurement gain and offset error of electrical components, such as the Fusion FPGA
- Application's operating range

Customer Application's Required Accuracy

Given the required accuracy of an application within its operating range, designers can use the specified gain and offset error of Fusion FPGAs to determine the suitable calibration method to achieve the desired accuracy. Refer to the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet for more information.

Calibration Measurements

To calculate m or c in EQ 12-1 and EQ 12-2 on page 12-2, measurements must be taken in a known environment so measured data can be compared against actual values. The number of required data points depends on the method of calibration. One data point would suffice for one-point calibration (determining offset), whereas for two-point calibration, two data points are needed to define gain and offset. In calibration measurements, a known actual value (temperature, current, or voltage) is supplied to the system, and the measured value is recorded.

Offset-Only Calibration Measurement

In offset-only (or one-point) calibration measurement, an actual value of P_{a1} (e.g., temperature) is applied to the system, and its value is measured as P_{m1} by the system. Then, the offset value c in Figure 12-1 on page 12-2 can be calculated as shown in EQ 12-3.

$$c = y1 - x1$$

EQ 12-3

Offset-and-Gain (two-point) Calibration Measurement

As shown in Figure 12-2, to calculate m and c in EQ 12-2 on page 12-2, two data points are needed for two-point calibration.

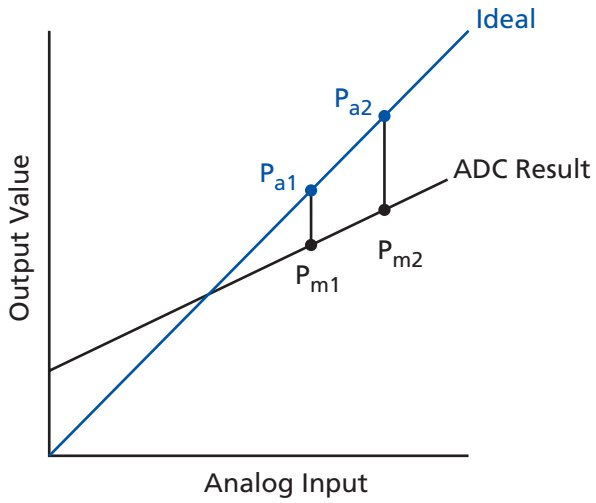


***Figure 12-2* • Two-Point Calibration for Offset-and-Gain Error**

Therefore, two known actual values (y_1 and y_2) must be applied to the system, and two measured points (x_1 and x_2) must be recorded. The m and c values in EQ 12-2 on page 12-2 are calculated as shown in EQ 12-4 and EQ 12-5.

$$m = (y_2 - y_1) / (x_2 - x_1)$$

EQ 12-4

$$c = (y_1 \times x_2 - y_2 \times x_1) / (x_2 - x_1)$$

EQ 12-5

Choosing Calibration Data Points

In one-point calibration, from a practical point of view, Actel recommends that the applied actual value be in the middle of the operating range as defined by the application. For example, if the system measures a voltage that operates from 0 V to 5 V, taking the calibration measurement at ~2.5 V will typically give the best results.

For two-point calibration measurement, Actel recommends that the two data points be taken at 20% and 80% of the operation range. For example, if a temperature measurement application is used in a system that operates from 0°C to 50°C, Actel recommends that the calibration measurements be taken at 10°C and 40°C.

In many voltage or current monitoring applications where the operating range includes 0 V or 0 A, customers tend to choose 0 V or 0 A as one of their calibration measurement data points. This is mainly driven by the simplicity of setup for measurements at the ground level. However, the ground level of the system is typically noisy due to the operation of the system and other noise factors. In such situations, the calibration measurement may not be sufficient to achieve the accuracy level required by the overall design. Therefore, Actel recommends that zero-level measurements be avoided for voltage and current calibration data collection.

Actel Fusion FPGAs offer up to 32 analog channels for temperature, current, or voltage measurements. Many applications, such as system management, use more than one analog channel in their design. Though all these channels use a single ADC inside Fusion, each prescaler circuit within the analog I/O structure has a unique set of op amp circuits. Therefore, it is necessary to calibrate each channel that requires the increased level of accuracy independently. In this case, each channel has its own calibration coefficient based on the method used for calibration (one-point or two-point).

Furthermore, in an analog (voltage/current/temperature) measurement, application designers exploit other components besides the Fusion FPGA to sense, transport, or amplify the measured parameters.

Customers can use two general approaches in calibrating these systems:

1. Calibrate each device used in the measurement individually.
2. Calibrate all the utilized devices when operating together in the system.

In the first approach, the customer calibrates each device individually in a controlled setting. In this case, the methods and recommendations explained in this document are applicable for the Fusion device. For other components, the customer should follow the recommendations and techniques provided by the vendor of each component.

In the second approach, all the system components in the application are used to take the calibration data points. In this case, the total measurement error can be adjusted by calibrating the measured values after the ADC. If this method is used, all the recommendations and techniques in this document can be applied.

Actel Calibration Solution

Actel's device-level calibration solution offers significant improvement in ADC accuracy for voltage monitor applications. There are two ways of exercising the Fusion ADC for voltage monitoring: sampling prescaled analog input or sampling direct analog input. If a customer design requires better accuracy than the default Fusion ADC performance, then calibration is needed. Since direct analog input sampling accuracy is well within 1%, the Actel calibration solution does not offer any additional benefit, so it is only available for prescaled inputs.

Temperature and current monitor calibration are not supported.

The Actel calibration solution is a two-point offset-and-gain calibration scheme. The implementation is performed through the following two steps:

1. During production test and screening flows, m and c compensation values are determined for each analog voltage channel and stored in the flash memory block of each Fusion device.
2. In Actel Libero® Integrated Design Environment (IDE) v8.2 SP1 and later, an RTL calibration IP block is built into the SmartGen Fusion Analog System Builder core. This calibration block reads the m and c values stored in the flash memory and uses them to calibrate data for each analog voltage channel.

Coefficient Measurement and Programming

During the Fusion production test flow, in a controlled environment, Actel measures the calibration coefficients, m and c, of every prescaler level of all 30 channels of each device. Measurements are done with the Fusion ADC VAREF set to 2.56 V. In other words, the coefficients do not apply to any customer designs that use a VAREF other than 2.56 V. Actel calibration implementation is disabled in software when VAREF is set to another value.

Then coefficients are programmed into the dedicated spare page of Fusion flash memory block (FB) 0 (AFS600 has blocks 0 and 1), from page 50 to 62.

Customers should avoid overwriting these spare pages. An old design generated prior to Libero IDE v8.2 SP1 utilizes these spare pages for Analog System configuration data. Programming an old design to a calibrated device could overwrite these spare pages and corrupt the coefficients. Calibration coefficients of that device would then no longer be available.

On the other hand, programming a design with a calibration block generated from Libero IDE v8.2 SP1 or newer to an uncalibrated device will result in erroneous data from the ADC. To avoid this, customers can pre-program the device with the POPULATION.stp file provided in the Libero IDE v8.2 SP1 release. This programming action populates the dedicated flash memory area for calibration with m = 1 and c = 0. Then customers can program the device with the design STAPL file.

Calibration IP Deployment

To implement Actel's calibration solution, customers must generate a new Analog System and Flash Memory System using Libero IDE v8.2 SP1 or newer. Actel's calibration IP solution is not available for processor systems that use the CoreAI to interface to the analog block.

Analog System Builder Update

Through the Analog System Builder, customers have an option to deploy a calibration IP block named "CalibIP" into Fusion designs. The CalibIP block is seamlessly inserted into the original Analog System macro, as shown in Figure 12-3 on page 12-6. Figure 12-3 on page 12-6 shows only

the insertion into a full Analog System; the same concept applies to the Sequence Only (without SMEV and SMTR stages) and ADC Only (without ASSC, SMEV, and SMTR stages) flows.

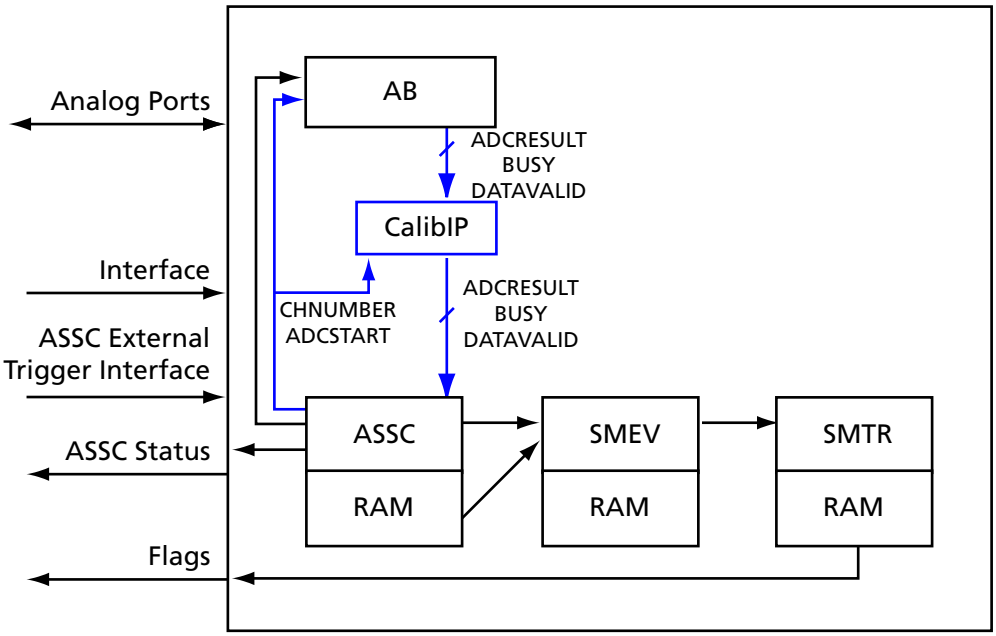


***Figure 12-3* • Full Analog System Macro with CalibIP**

During power-up, the initialization state machine, INIT/CONFIG IP, loads the coefficients from the flash memory block into a dedicated SRAM block for the CalibIP core. CalibIP reads the coefficients from the SRAM block and applies the m and c values to the raw ADCRESULT, following EQ 12-2 on page 12-2 to generate the calibrated ADCRESULT. The calibrated ADCRESULT then goes through the rest of the process as in the original processing flow.

There are two new ports created at the Analog Block top level to support calibration initialization from the flash memory block:

- INIT_CALIBROM_WEN – Write enable to ROM region, single-bit, active-high
- INIT_CALIBCOEFF_WEN – Write enable to coefficient region, single-bit, active-high

These are write enables for the INIT/CONFIG interface. Connect them to corresponding ports of the flash memory block top level, as shown in Figure 12-4 on page 12-7.

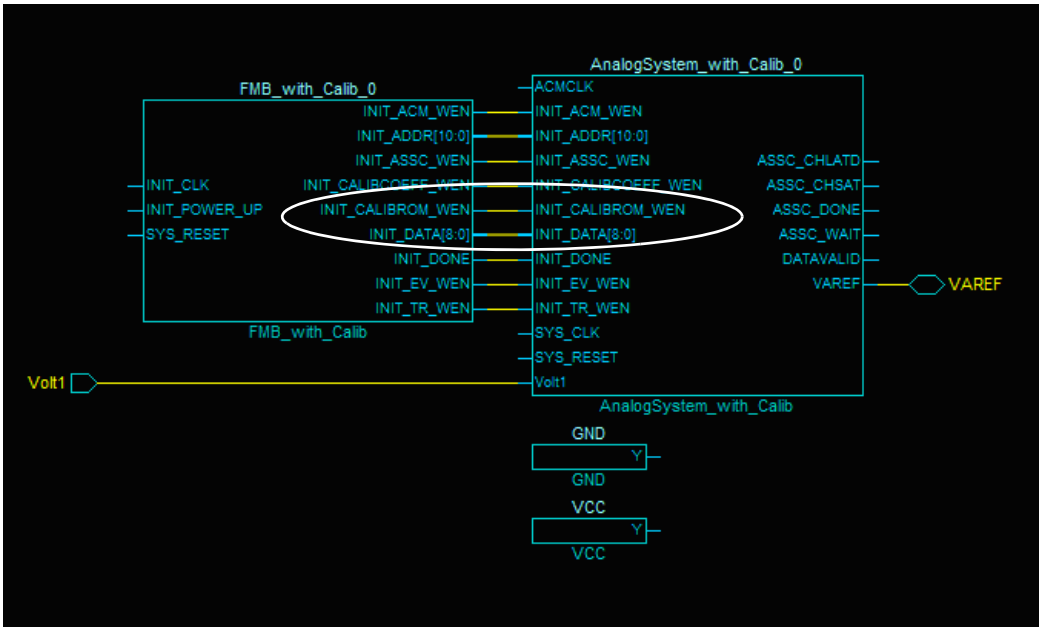


Figure 12-4 • **Connectivity between Analog System Block and Flash Memory Block**

Flash Memory System Builder Update

Inside the Flash Memory System Builder (FMSB), customers can generate an analog client to properly initialize the Analog System macro with CalibIP deployed. In addition to the regular Analog System configuration data partition, FMSB also creates two other partitions for the analog client: one for the coefficients' storage (CALIBCOEFFICIENT), from spare page 50 to 62, and one for a lookup table (CALIBROM) that records which channel and prescaler level need to be calibrated, from spare page 43 to 48.

Because of the new calibration coefficients' storage partition, the SMTR configuration data is assigned to flash memory block sector 63, from page 2,016 through 2,047 (or as addresses: 0x3F000 through 0x3FF80). SMTR uses up to 32 pages. The actual number of pages used in this sector depends on whether (and how many) flags are used in the Analog System design. The Flash Memory System Builder prevents customers from assigning any other clients to these pages.

The flash memory maps prior to and after the Libero IDE v8.2 SP1 release are shown in Figure 12-5 and Figure 12-6 on page 12-9.

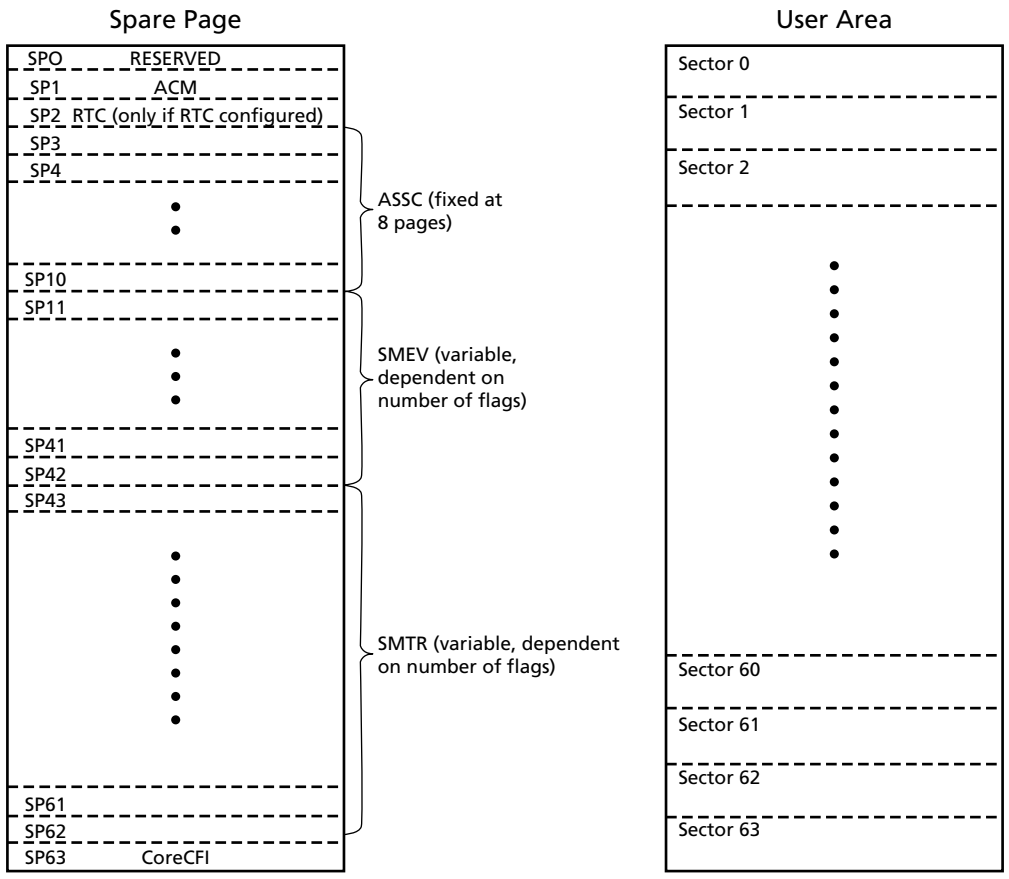


Figure 12-5 • **Flash Memory Map Prior to Libero IDE v8.2 SP1**

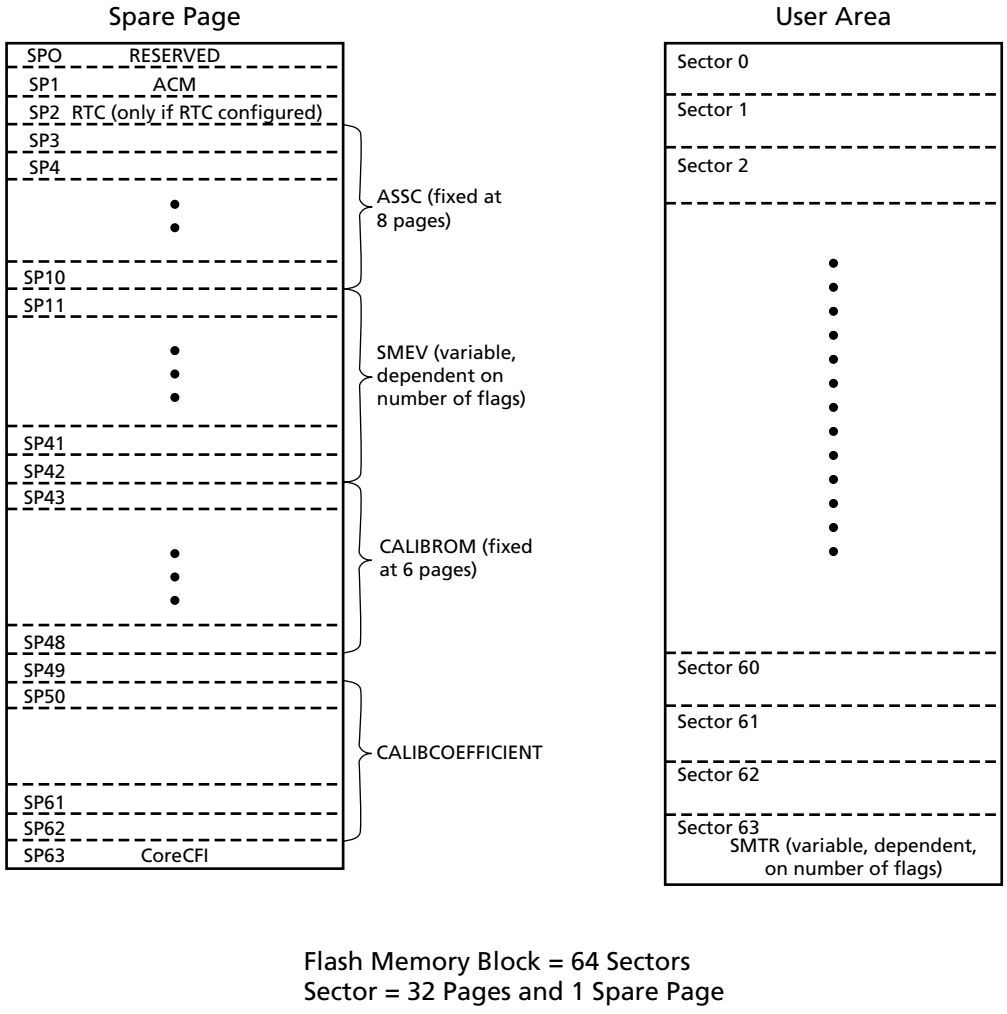


Figure 12-6 • **Flash Memory Map after Libero IDE v8.2 SP1**

There are two new ports created at the flash memory block top level, corresponding to those created for the Analog Block:

- INIT_CALIBROM_WEN
- INIT_CALIBCOEFF_WEN

Connect these ports to the Analog Block top level.

Design Flow and Tips

For calibrated Fusion devices, there are several implementation scenarios.

Scenario 1: Brand New Design

Follow the regular design flow to implement the Actel calibration solution in a new design:

By default, the calibration IP is enabled in the Analog System macro, and the flash memory block initializes the IP. The Designer software places the corresponding analog client in flash memory block 0.

The content of the memory file (*.mem) is different from that of the embedded flash configuration file (*.efc). The *.mem file produced by the Flash Memory Builder for simulation purposes is populated with m = 1.0 and c = 0 for all channels and all prescaler combinations. CalibIP can function appropriately during simulation. The *.efc file for programming file generation does not include the m and c content because the coefficients are pre-programmed into the device during production test.

To disable the calibration IP, uncheck **Include calibration IP** in the Advanced Options of the Analog System Builder, as shown in Figure 12-7.

For a calibration-enabled design, when **Bypass calibration on saturated ADC input** is selected, the saturated ADC result is passed to the next level of computation without calibration. If unchecked, the saturated ADC result is calibrated before it is passed to the next level.

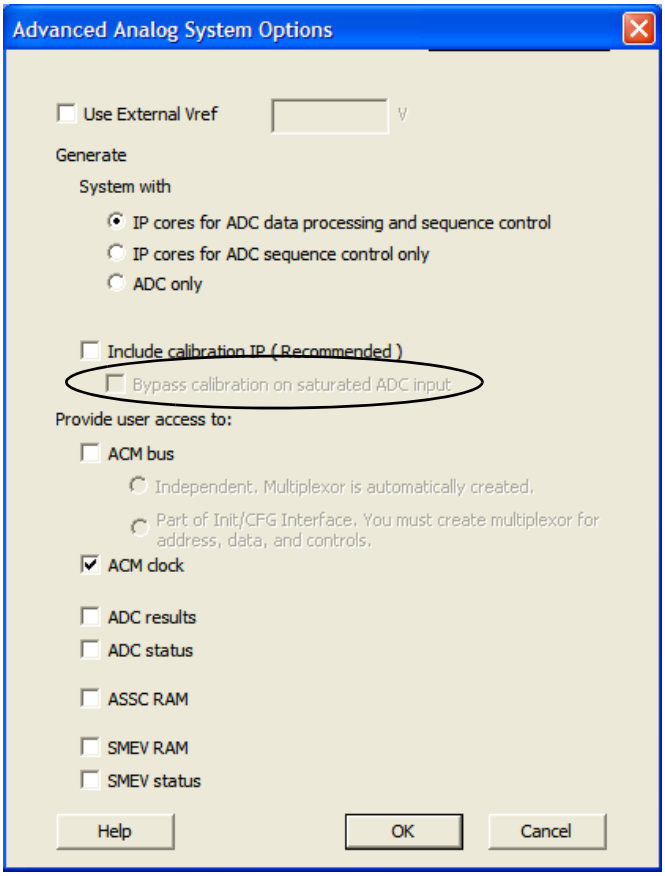


Figure 12-7 • **Disable CalibIP from Advanced Options (Libero IDE v8.2 SP1)**

Scenario 2: Update Existing Design to Implement Calibration Solution

To update existing Fusion designs and utilize the Actel calibration solution, take the following steps to regenerate the design:

1. Open the design in Libero IDE v8.2 SP1 or newer.
2. Regenerate the Analog System macro in Analog System Builder.

   Open **Advanced Options**, select the **Include calibration IP** option, then regenerate the macro.
3. Regenerate the flash memory block.
4. Make sure the additional ports (INIT_CALIBROM_WEN and INIT_CALIBCOEFF_WEN) are properly connected, either through SmartDesign (Figure 12-8) or HDL coding.

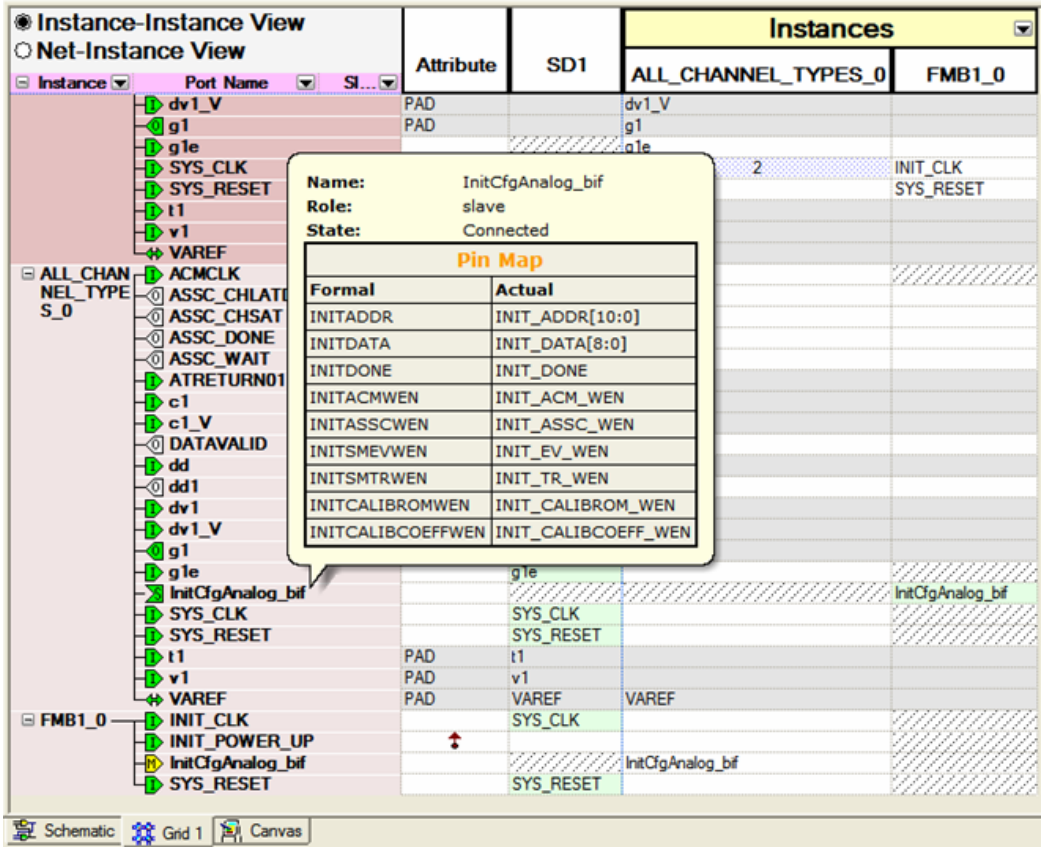


Figure 12-8 • **SmartDesign Connectivity Grid in Libero v8.2 SP1**

Go through the rest of the regular design flow (synthesis, compile, and layout with proper simulation and timing analysis).

Scenario 3: Existing Design for Firmware Image Update Only

To update the firmware image in the existing design without using the calibration IP, regenerate the Flash Memory Block. Then go through the rest of design flow (synthesis, compile, and layout with proper simulation and timing analysis).

Scenario 4: Maintain Existing Design in New Software Release without Using Calibration Solution

Actel recommends that all customers regenerate the Analog System Block and the flash memory block in Libero IDE v8.2 SP1 or newer software releases, unless the Analog System Block utilizes less than 20 channels and less than four flags per channel.

Programming a calibrated device with designs generated prior to Libero IDE v8.2 SP1 can overwrite and corrupt the pre-programmed coefficient data in the dedicated flash memory partition. The FlashPro software released with Libero IDE v8.2 SP1 detects whether there is a memory map overlap. If there is a memory overlap, FlashPro cancels the programming action and asks the user to regenerate the Analog System.

To program a design with calibration implemented to a targeted device that is uncalibrated, pre-program the device with the POPULATION.stp file provided by Actel. This programming action populates the dedicated flash memory area for calibration with m = 1 and c = 0. Then program the device with the design STAPL file.

Utilization and Performance

The total RAM block and core tile utilization to implement CalibIP and the corresponding initialization process is listed in Table 12-1. CalibIP and other IPs infer registers with enable. When these registers have a SET or RESET signal, assign the SET or RESET signal to a global resource to make sure that the register remains a one-tile implementation (NOT split into two tiles).

Table 12-1 • **Calibration Implementation Utilization Report**

RAM Block	Tile Count for CalibIP	
	Optimized for Area	Optimized for Speed
1	363	453

The CalibIP performance is listed in Table 12-2.

Table 12-2 • **CalibIP Performance Report**

Speed Grade	CalibIP Performance (MHz)	
	Optimized for Area	Optimized for Speed
Std.	45	57
–2	75	96

The performance of CalibIP is only limited by the latency introduced by the Compute Block. The Compute Block adds a latency of 14 clock cycles. For example, 14 clock cycles of a 40 MHz system clock is 0.35 microseconds. With calibration implemented, you can expect the ADC result 0.35 microseconds later than in a design without calibration implementation. The sampling rate is degraded by 2%.

Improvement from Actel Calibration Solution

Table 12-3 shows typical error using Actel's calibration solution.

Table 12-3 • **Fusion Analog System Typical Error with CalibIP Deployed**

Input Voltage (V)	Calibrated Typical Error per Positive Prescaler Setting[1] (%)							Direct ADC[2, 3] (%)
	16 V (AT)	16 V (12 V) (AV/AC)	8 V (AV/AC)	4 V (AT)	4 V (AV/AC)	2 V (AV/AC)	1 V (AV/AC)	VAREF = 2.56 V
15	1							
14	1							
12	1	1						
5	2	2	1					
3.3	2	2	1	1	1			
2.5	3	2	1	1	1			1
1.8	4	4	1	1	1	1		1
1.5	5	5	2	2	2	1		1
1.2	7	6	2	2	2	1		1
0.9	9	9	4	4	3	1	1	1

Notes:

1. *Requires enabling Analog Calibration in the Actel tool flow.*
2. *Direct ADC mode using an external VAREF of 2.56V±4.6mV, without Analog Calibration macro.*
3. *For input greater than 2.56 V, the ADC output will saturate. A higher V_{AREF} or prescaler usage is recommended.*

Microprocessor-Based Design Flow

In a microprocessor-based design flow, designers can use CoreAI (Analog Interface) to interface and control the analog peripherals within the Fusion device family. Designers using Core8051, CoreMP7, or Cortex™-M1 with CoreAI can take advantage of the CoreAI Driver (provided in C code) to support the calibration features.

The CoreAI driver provides a set of Application Program Interface (API) functions to support different calibration modes and automatically calculate the final calibrated value, based on the m and c coefficient stored in the spare page of the Fusion flash memory block.

Currently, the calibration scheme only supports voltage monitor applications (AVx pins) and is only needed for prescaled voltage inputs. If direct analog input sampling accuracy is well within 1%, the Actel calibration solution does not offer any additional benefits. CoreAI driver must be used with CoreAI version 2.1 (or higher) and CoreAhbNvm version 1.3.135 (or higher). Refer to the *CoreAI Driver User's Guide* for more information.

Performing System-Level Calibration Using Fusion

Previous sections of this document describe the general approach to calibration using the offset-only and offset-plus-gain approaches, and provided a detailed explanation of Actel's calibration solution for Fusion voltage input signals. In addition to this solution, users may desire calibration of temperature and current inputs, as well as calibration of the entire system working together. A recommended approach to accomplishing these tasks is provided below. This methodology may be added in the user's design on top of the device-level calibration solution.

The "Calibration Measurements" section on page 12-3 explains how the calibration coefficients, as described in EQ 12-1 and EQ 12-2 on page 12-2, are calculated. EQ 12-1 and EQ 12-2 on page 12-2 (depending on the calibration method used) are implemented using an adder (one-point calibration) or a combination of an adder and a multiplier (two-point calibration), as shown in Figure 12-9.

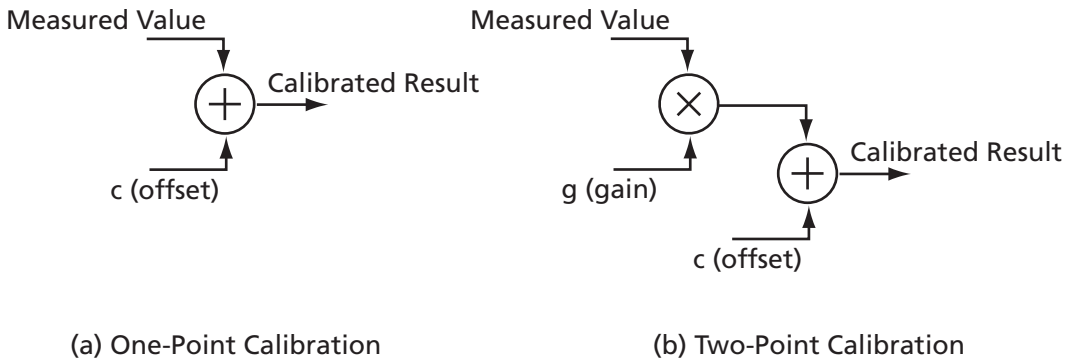


Figure 12-9 • **Implementation of One-Point and Two-Point Calibration**

The calibration coefficients (m and c in Figure 12-9) can be stored in the nonvolatile flash memory of each Fusion device. Therefore, inside the flash memory architecture, different memory addresses can contain the calibration coefficients for each channel in the design. Depending on which channel in the design is performing the measurement at the time, appropriate calibration coefficients can be fetched and fed into the adder and/or multiplier. For details on writing and reading to the Fusion flash memory block, refer to the Embedded Flash Memory chapter of the *Fusion Handbook*.

Where and how the adder and multiplier are implemented depends highly on the application and the user's design. Generally, there are three ways designers can implement the arithmetic functions in Figure 12-9 to produce the calibrated results of measurement. The following explains each of these implementations and their pros and cons:

Implementing a Dedicated Adder and Multiplier Using FPGA Core Gates

Pros

- High speed
- Efficient for one-point calibration

Cons

- Multiplier implementation consumes large number of gates

Using the Design's Microprocessor/Microcontroller ALU to Perform Calibration Calculations

Pros

- Saves FPGA gate resources compared to implementing dedicated multiplier

Cons

- May slow down microprocessor's performance depending on the overall sampling rate

Implementing the Numerical Calculations in Application Software

Pros

- Saves FPGA gates

Cons

- Depending on the application, may take up a lot of bandwidth from the host processor
- Only suitable for applications where there is software communicating with hardware

Conclusion

Designers use calibration to increase the accuracy achievable in applications involving analog components. Actel Fusion mixed-signal FPGAs offer the capability of measuring analog voltage, current, and temperature. If customers require more accuracy than is inherent in Fusion FPGAs, calibration techniques can be used to achieve these requirements. Fusion FPGAs offer the advantage of having the calibration design and coefficients programmed into the FPGA itself without a need for any external components. This document discusses the typical calibration techniques and the implementation of Actel calibration solutions for Fusion FPGAs. The result shows that with the Actel calibration implementation, customers can achieve 1% ADC sampling accuracy for all Fusion devices and all channels.

Related Documents

Datasheets

Fusion Family of Mixed-Signal Flash FPGAs

http://www.actel.com/documents/Fusion_DS.pdf

Handbooks

Fusion Handbook

http://www.actel.com/documents/Fusion_HB.pdf

Part Number and Revision Date

This document was previously published as an application note describing features and functions of the device, and as such has now been incorporated into the device handbook format. No technical changes have been made to the content.

Part Number 51700092-019-0

Revised October 2008

List of Changes

The following table lists critical changes that were made in the current version of the document.

Previous Version	Changes in Current Version (v1.0)	Page
51900161-3/6.08	The "Microprocessor-Based Design Flow" section was added.	12-13
51900161-1/2.08	This statement was added to the "Calibration IP Deployment" section: Actel's calibration IP solution is not available for processor systems that use the CoreAI to interface to the analog block.	12-5
	In the "Performing System-Level Calibration Using Fusion"section, a reference to the Fusion Handbook was added.	12-13
51900161-0/2.07	Please read the document very carefully. A lot of helpful and useful information was added to the document.	N/A
	The "Introduction" section was updated.	12-1
	The heading title "Calibration Measurements" section is new and all subsections were significantly updated. Please note the variables in all equations were changed. In addition, the variable g was changed to m throughout the document.	12-3
	The "Actel Calibration Solution" section and all subsections are new.	12-5
	The heading title "Implementing Calibration Design" was changed to "Performing System-Level Calibration Using Fusion" section. In Figure 12-9 •Implementation of One-Point and Two-Point Calibration, the m was changed to g.	12-13

I/O Descriptions and Usage

Actel
POWER MATTERS

13 – I/O Software Control in Low-Power Flash Devices

Actel Fusion,® IGLOO,® and ProASIC®3 I/Os provide more design flexibility, allowing the user to control specific features by enabling certain I/O standards. Some features are selectable only for certain I/O standards, whereas others are available for all I/O standards. For example, slew control is not supported by differential I/O standards. Conversely, I/O register combining is supported by all I/O standards. For detailed information about which I/O standards and features are available on each device and each I/O type, refer to the I/O Structures section of the handbook for the device you are using.

Figure 13-1 shows the various points in the software design flow where a user can provide input or control of the I/O selection and parameters. A detailed description is provided throughout this document.

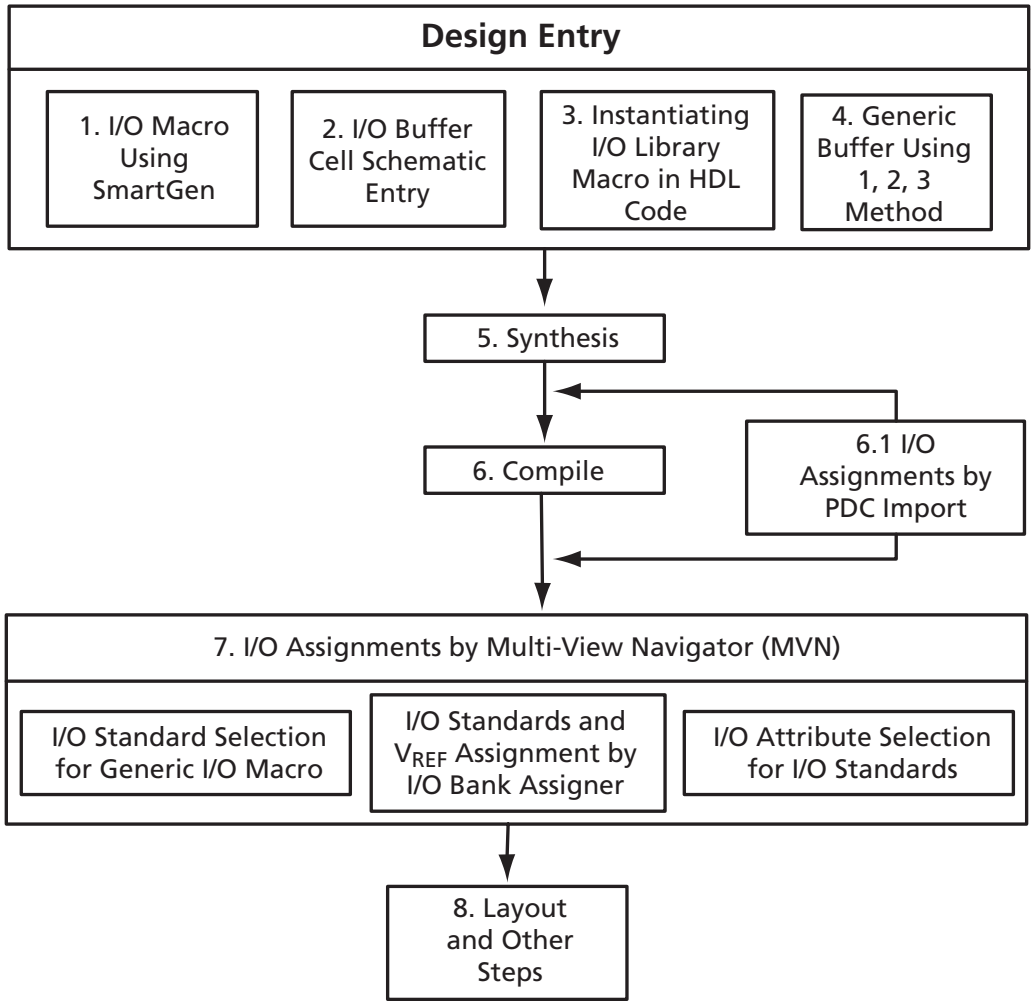


Figure 13-1 • **User I/O Assignment Flow Chart**

Flash FPGAs I/O Support

The flash FPGAs listed in Table 13-1 support I/Os and the functions described in this document.

Table 13-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 13-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 13-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Software-Controlled I/O Attributes

Users may modify these programmable I/O attributes using the I/O Attribute Editor. Modifying an I/O attribute may result in a change of state in Designer. Table 13-2 details which steps have to be re-run as a function of modified I/O attribute.

Table 13-2 • **Designer State (resulting from I/O attribute modification)**

	Designer States				
I/O Attribute	**Compile**	**Layout**	**Fuse**	**Timing**	**Power**
Slew Control	No	No	Yes	Yes	Yes
Output Drive (mA)	No	No	Yes	Yes	Yes
Skew Control	No	No	Yes	Yes	Yes
Resistor Pull	No	No	Yes	Yes	Yes
Input Delay	No	No	Yes	Yes	Yes
Schmitt Trigger	No	No	Yes	Yes	Yes
OUT_LOAD	No	No	No	Yes	Yes
COMBINE_REGISTER	Yes	Yes	N/A	N/A	N/A

Notes:

1. *No = Remains the same, Yes = Re-run the step, N/A = Not applicable*
2. *Skew control and input delay do not apply to IGLOO nano, IGLOO PLUS, and ProASIC3 nano devices.*

Implementing I/Os in Actel Software

Actel Libero® Integrated Design Environment (IDE) is integrated with design entry tools such as the SmartGen macro builder, the ViewDraw schematic entry tool, and an HDL editor. It is also integrated with the synthesis and Designer tools. In this section, all necessary steps to implement the I/Os are discussed.

Design Entry

There are three ways to implement I/Os in a design:

1. Use the SmartGen macro builder to configure I/Os by generating specific I/O library macros and then instantiating them in top-level code. This is especially useful when creating I/O bus structures.
2. Use an I/O buffer cell in a schematic design.
3. Manually instantiate specific I/O macros in the top-level code.

If technology-specific macros, such as INBUF_LVCMOS33 and OUTBUF_PCI, are used in the HDL code or schematic, the user will not be able to change the I/O standard later on in Designer. If generic I/O macros are used, such as INBUF, OUTBUF, TRIBUF, CLKBUF, and BIBUF, the user can change the I/O standard using the Designer I/O Attribute Editor tool.

Using SmartGen for I/O Configuration

The SmartGen tool in Libero IDE provides a GUI-based method of configuring the I/O attributes. The user can select certain I/O attributes while configuring the I/O macro in SmartGen. The steps to configure an I/O macro with specific I/O attributes are as follows:

1. Open Libero IDE.
2. On the left-hand side of the Catalog View, select **I/O**, as shown in Figure 13-2.

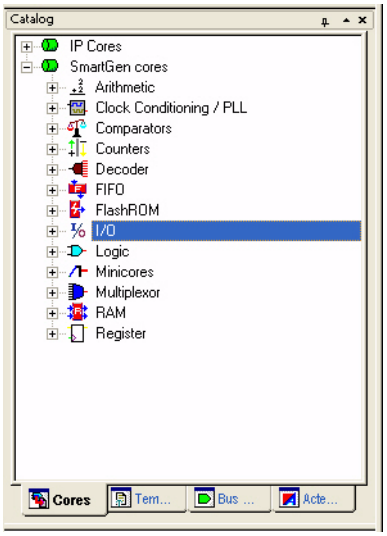


Figure 13-2 • **SmartGen Catalog**

3. Expand the I/O section and double-click one of the options (Figure 13-3).

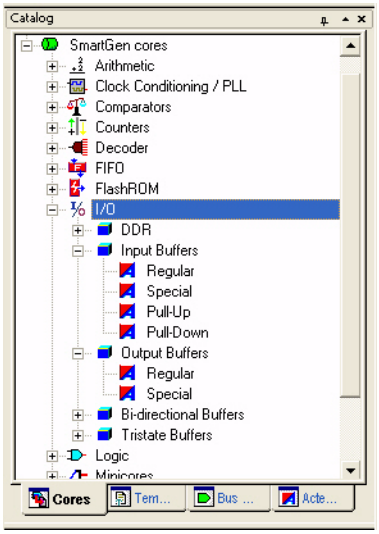


Figure 13-3 • **Expanded I/O Section**

4. Double-click any of the varieties. The I/O Create Core window opens (Figure 13-4).

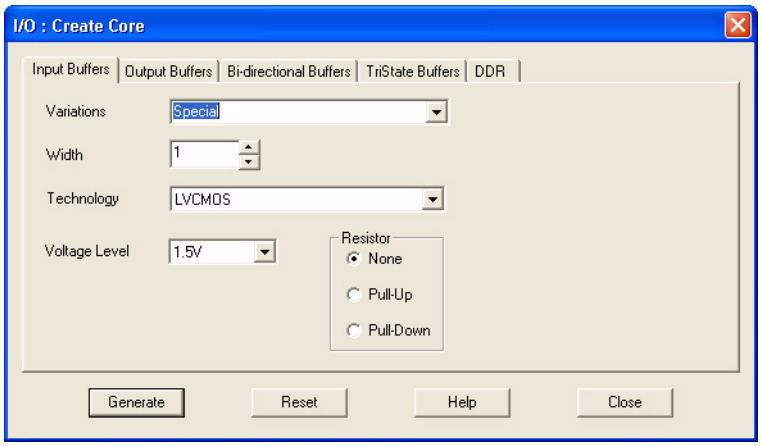


Figure 13-4 • **I/O Create Core Window**

As seen in Figure 13-4, there are five tabs to configure the I/O macro: Input Buffers, Output Buffers, Bidirectional Buffers, Tristate Buffers, and DDR.

Input Buffers

There are two variations: Regular and Special.

If the **Regular** variation is selected, only the Width (1 to 128) needs to be entered. The default value for Width is 1.

The **Special** variation has Width, Technology, Voltage Level, and Resistor Pull-Up/-Down options (see Figure 13-4). All the I/O standards and supply voltages (V_{CCI}) supported for the device family are available for selection.

Output Buffers

There are two variations: Regular and Special.

If the **Regular** variation is selected, only the Width (1 to 128) needs to be entered. The default value for Width is 1.

The **Special** variation has Width, Technology, Output Drive, and Slew Rate options.

Bidirectional Buffers

There are two variations: Regular and Special.

The **Regular** variation has Enable Polarity (Active High, Active Low) in addition to the Width option.

The **Special** variation has Width, Technology, Output Drive, Slew Rate, and Resistor Pull-Up/-Down options.

Tristate Buffers

Same as Bidirectional Buffers.

DDR

There are eight variations: DDR with Regular Input Buffers, Special Input Buffers, Regular Output Buffers, Special Output Buffers, Regular Tristate Buffers, Special Tristate Buffers, Regular Bidirectional Buffers, and Special Bidirectional Buffers.

These variations resemble the options of the previous I/O macro. For example, the Special Input Buffers variation has Width, Technology, Voltage Level, and Resistor Pull-Up/-Down options. DDR is not available on IGLOO PLUS devices.

5. Once the desired configuration is selected, click the **Generate** button. The Generate Core window opens (Figure 13-5).
6. Enter a name for the macro. Click **OK**. The core will be generated and saved to the appropriate location within the project files (Figure 13-6 on page 13-7).

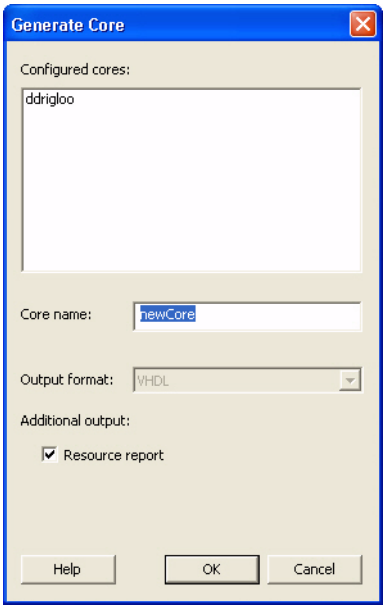


Figure 13-5 • **Generate Core Window**

7. Instantiate the I/O macro in the top-level code.

   The user must instantiate the DDR_REG or DDR_OUT macro in the design. Use SmartGen to generate both these macros and then instantiate them in your top level. To combine the DDR macros with the I/O, the following rules must be met:

Rules for the DDR I/O Function

- The fanout between an I/O pin (D or Y) and a DDR (DDR_REG or DDR_OUT) macro must be equal to one for the combining to happen on that pin.
- If a DDR_REG macro and a DDR_OUT macro are combined on the same bidirectional I/O, they must share the same clear signal.
- Registers will not be combined in an I/O in the presence of DDR combining on the same I/O.

Using the I/O Buffer Schematic Cell

Libero IDE includes the ViewDraw schematic entry tool. Using ViewDraw, the user can insert any supported I/O buffer cell in the top-level schematic. Figure 13-6 shows a top-level schematic with different I/O buffer cells. When synthesized, the netlist will contain the same I/O macro.

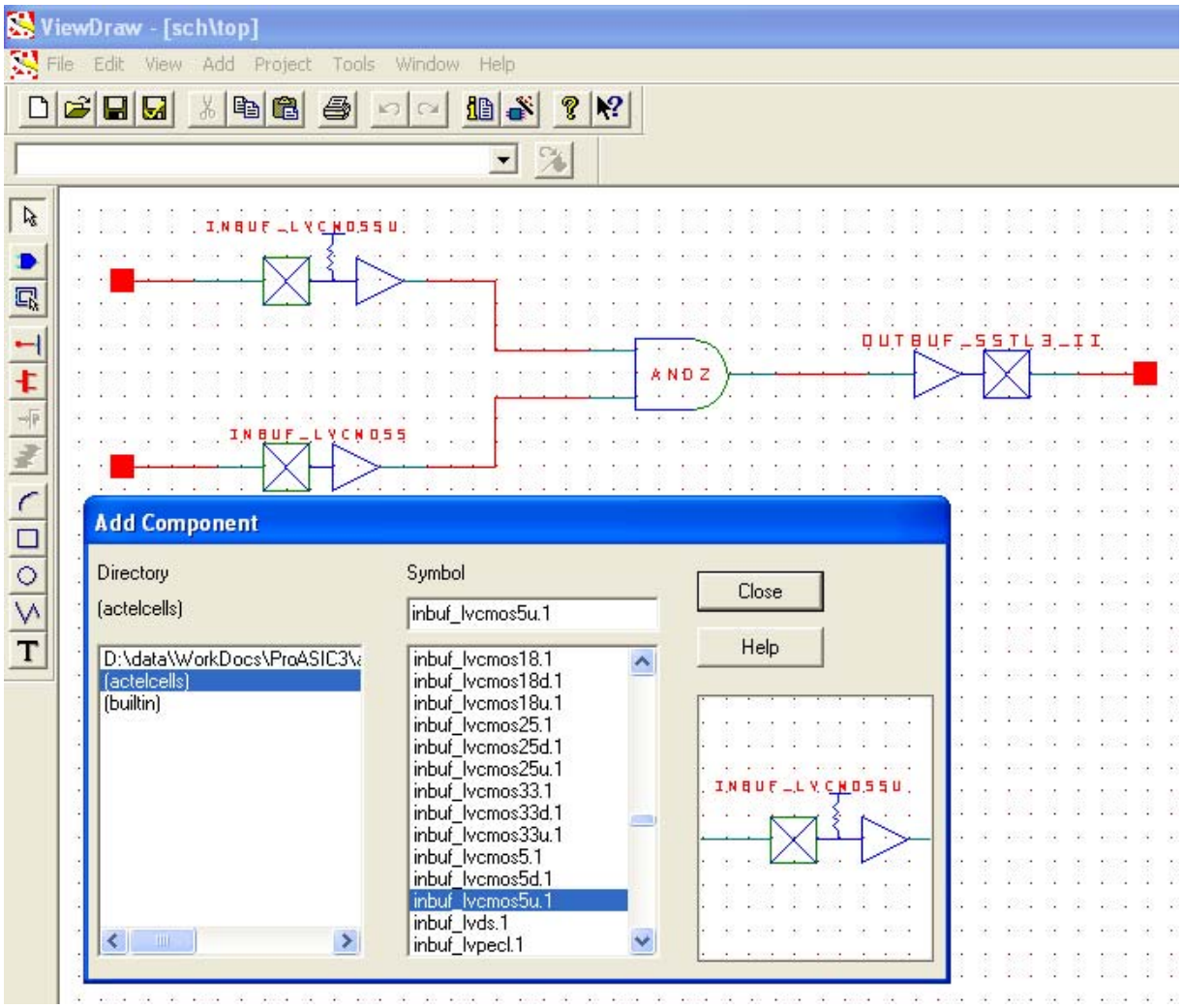


***Figure 13-6* • I/O Buffer Schematic Cell Usage**

Instantiating in HDL code

All the supported I/O macros can be instantiated in the top-level HDL code (refer to the *IGLOO, Fusion, and ProASIC3 Macro Library Guide* for a detailed list of all I/O macros). The following is an example:

```
library ieee;
use ieee.std_logic_1164.all;
library proasic3e;

entity TOP is
  port(IN2, IN1 : in std_logic; OUT1 : out std_logic);
end TOP;

architecture DEF_ARCH of TOP is

  component INBUF_LVCMOS5U
    port(PAD : in std_logic := 'U'; Y : out std_logic);
  end component;

  component INBUF_LVCMOS5
    port(PAD : in std_logic := 'U'; Y : out std_logic);
  end component;

  component OUTBUF_SSTL3_II
    port(D : in std_logic := 'U'; PAD : out std_logic);
  end component;

  Other component …..

signal x, y, z…….other signals : std_logic;

begin

  I1 : INBUF_LVCMOS5U
    port map(PAD => IN1, Y =>x);
  I2 : INBUF_LVCMOS5
    port map(PAD => IN2, Y => y);
  I3 : OUTBUF_SSTL3_II
    port map(D => z, PAD => OUT1);

  other port mapping…

end DEF_ARCH;
```

Synthesizing the Design

Libero IDE integrates with the Synplify® synthesis tool. Other synthesis tools can also be used with Libero IDE. Refer to the *Actel Libero IDE User's Guide* or Libero IDE online help for details on how to set up the Libero IDE tool profile with synthesis tools from other vendors.

During synthesis, the following rules apply:

- Generic macros:
  - Users can instantiate generic INBUF, OUTBUF, TRIBUF, and BIBUF macros.
  - Synthesis will automatically infer generic I/O macros.
  - The default I/O technology for these macros is LVTTL.
  - Users will need to use the I/O Attribute Editor in Designer to change the default I/O standard if needed (see Figure 13-7 on page 13-9).
- Technology-specific I/O macros:
  - Technology-specific I/O macros, such as INBUF_LVCMO25 and OUTBUF_GTL25, can be instantiated in the design. Synthesis will infer these I/O macros in the netlist.

– The I/O standard of technology-specific I/O macros cannot be changed in the I/O Attribute Editor (see Figure 13-7).
– The user MUST instantiate differential I/O macros (LVDS/LVPECL) in the design. This is the only way to use these standards in the design.
– To implement the DDR I/O function, the user must instantiate a DDR_REG or DDR_OUT macro. This is the only way to use a DDR macro in the design.

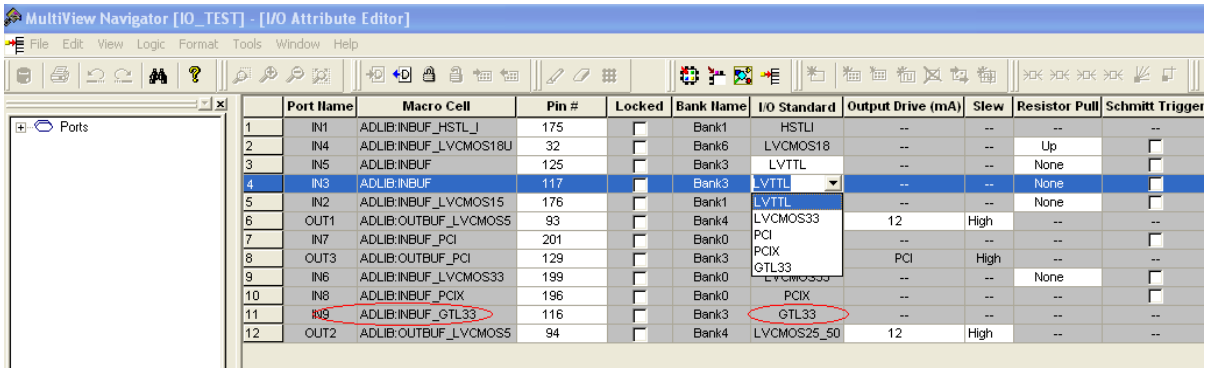


Figure 13-7 • **Assigning a Different I/O Standard to the Generic I/O Macro**

Performing Place-and-Route on the Design

The netlist created by the synthesis tool should now be imported into Designer and compiled. During Compile, the user can specify the I/O placement and attributes by importing the PDC file. The user can also specify the I/O placement and attributes using ChipPlanner and the I/O Attribute Editor under MVN.

Defining I/O Assignments in the PDC File

A PDC file is a Tcl script file specifying physical constraints. This file can be imported to and exported from Designer.

Table 13-3 shows I/O assignment constraints supported in the PDC file.

Table 13-3 • **PDC I/O Constraints**

Command	Action	Example	Comment
I/O Banks Setting Constraints			

set_iobank	Sets the I/O supply voltage, V_{CCI}, and the input reference voltage, V_{REF}, for the specified I/O bank.	`set_iobank bankname [-vcci vcci_voltage] [-vref vref_voltage]`
		
		`set_iobank Bank7 -vcci 1.50 -vref 0.75`
			Must use in case of mixed I/O voltage (V_{CCI}) design

set_vref	Assigns a V_{REF} pin to a bank.	`set_vref -bank [bankname] [pinnum]`
		
		`set_vref -bank Bank0 685 704 723 742 761`
			Must use if voltage-referenced I/Os are used

Note: Refer to the Actel Libero IDE User's Guide for detailed rules on PDC naming and syntax conventions.

Table 13-3 • **PDC I/O Constraints (continued)**

Command	Action	Example	Comment

set_vref_defaults	Sets the default V_{REF} pins for the specified bank. This command is ignored if the bank does not need a V_{REF} pin.	`set_vref_defaults bankname`
		
		`set_vref_defaults bank2`	

I/O Attribute Constraint			

set_io	Sets the attributes of an I/O	`set_io portname`
		`[-pinname value]`
		`[-fixed value]`
		`[-iostd value]`
		`[-out_drive value]`
		`[-slew value]`
		`[-res_pull value]`
		`[-schmitt_trigger value]`
		`[-in_delay value]`
		`[-skew value]`
		`[-out_load value]`
		`[-register value]`
		
		`set_io IN2 -pinname 28`
		`-fixed yes -iostd LVCMOS15`
		`-out_drive 12 -slew high`
		`-RES_PULL None`
		`-SCHMITT_TRIGGER Off`
		`-IN_DELAY Off –skew off`
		`-REGISTER No`
			If the I/O macro is generic (e.g., INBUF) or technology-specific (INBUF_LVCMOS25), then all I/O attributes can be assigned using this constraint.
			If the netlist has an I/O macro that specifies one of its attributes, that attribute cannot be changed using this constraint, though other attributes can be changed.
			Example: OUTBUF_S_24 (low slew, output drive 24 mA)
			Slew and output drive cannot be changed.

I/O Region Placement Constraints			

define_region	Defines either a rectangular region or a rectilinear region	`define_region`
		`-name [region_name]`
		`-type [region_type] x1 y1 x2 y2`
		
		`define_region -name test`
		`-type inclusive 0 15 2 29`
			If any number of I/Os must be assigned to a particular I/O region, such a region can be created with this constraint.

assign_region	Assigns a set of macros to a specified region	`assign_region [region name]`
		`[macro_name...]`
		
		`assign_region test U12`
			This constraint assigns I/O macros to the I/O regions. When assigning an I/O macro, PDC naming conventions must be followed if the macro name contains special characters; e.g., if the macro name is \\$1I19\\, the correct use of escape characters is \\\\\$1I19\\\\.

Note: Refer to the Actel Libero IDE User's Guide for detailed rules on PDC naming and syntax conventions.

Compiling the Design

During Compile, a PDC I/O constraint file can be imported along with the netlist file. If only the netlist file is compiled, certain I/O assignments need to be completed before proceeding to Layout. All constraints that can be entered in PDC can also be entered using ChipPlanner, I/O Attribute Editor, and PinEditor.

There are certain rules that must be followed in implementing I/O register combining and the I/O DDR macro (refer to the I/O Registers section of the handbook for the device that you are using and the "DDR" section on page 13-6 for details). Provided these rules are met, the user can enable or disable I/O register combining by using the PDC command `set_io portname –register yes|no` in the I/O Attribute Editor or selecting a check box in the Compile Options dialog box (see Figure 13-8). The Compile Options dialog box appears when the design is compiled for the first time. It can also be accessed by choosing **Options** > **Compile** during successive runs. I/O register combining is off by default. The PDC command overrides the setting in the Compile Options dialog box.

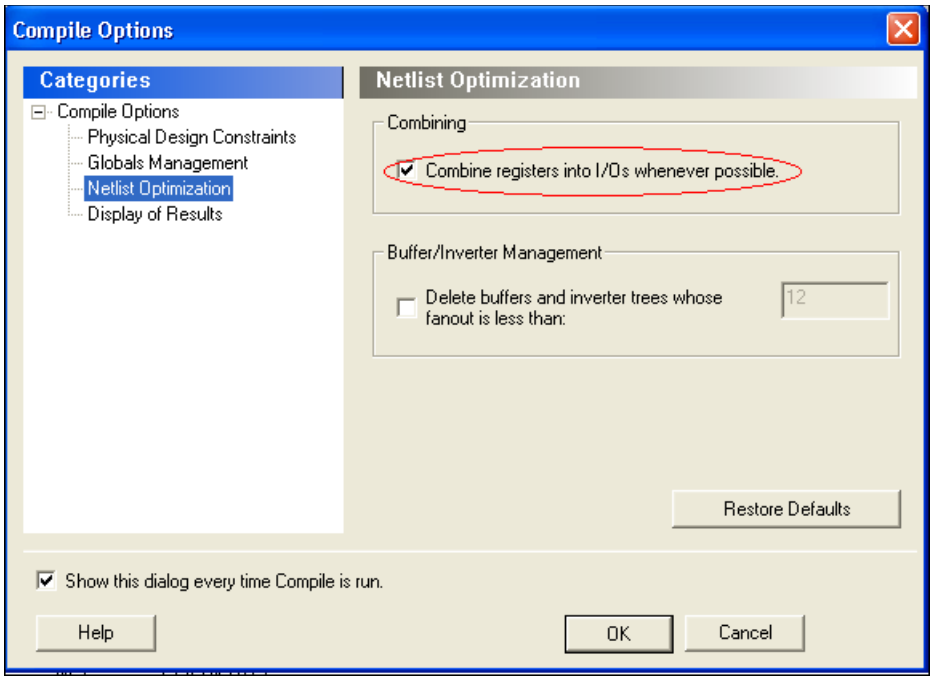


Figure 13-8 • **Setting Register Combining During Compile**

Understanding the Compile Report

The I/O bank report is generated during Compile and displayed in the log window. This report lists the I/O assignments necessary before Layout can proceed.

When Designer is started, the I/O Bank Assigner tool is run automatically if the Layout command is executed. The I/O Bank Assigner takes care of the necessary I/O assignments. However, these assignments can also be made manually with MVN or by importing the PDC file. Refer to the "Assigning Technologies and V_{REF} to I/O Banks" section on page 13-14 for further description.

The I/O bank report can also be extracted from Designer by choosing **Tools** > **Report** and setting the Report Type to **IOBank**.

This report has the following tables: I/O Function, I/O Technology, I/O Bank Resource Usage, and I/O Voltage Usage. This report is useful if the user wants to do I/O assignments manually.

I/O Function

Figure 13-9 shows an example of the I/O Function table included in the I/O bank report:

I/O Function:

Type	w/o register	w/ register	w/ DDR register
Input I/O	7	0	1
Output I/O	1	1	0
Bidirectional I/O	0	0	0
Differential Input I/O Pairs	0	0	0
Differential Output I/O Pairs	0	0	1

Figure 13-9 • **I/O Function Table**

This table lists the number of input I/Os, output I/Os, bidirectional I/Os, and differential input and output I/O pairs that use I/O and DDR registers.

Certain rules must be met to implement registered and DDR I/O functions (refer to the I/O Structures section of the handbook for the device you are using and the "DDR" section on page 13-6).

I/O Technology

The I/O Technology table (shown in Figure 13-10) gives the values of V_{CCI} and V_{REF} (reference voltage) for all the I/O standards used in the design. The user should assign these voltages appropriately.

I/O Technology:

	Voltages		I/Os		
I/O Standard(s)	Vcci	Vref	Input	Output	Bidirectional
LVTTL	3.30v	N/A	1	1	0
LVCMOS33	3.30v	N/A	1	0	0
LVCMOS25_50	2.50v	N/A	1	1	0
LVCMOS18	1.80v	N/A	1	0	0
LVCMOS15	1.50v	N/A	1	0	0
PCIX	3.30v	N/A	1	0	0
LVDS	2.50v	N/A	0	2	0
SSTL3I (Input/Bidirectional)	3.30v	1.50v	1	0	0
GTLP33 (Input/Bidirectional)	3.30v	1.00v	1	0	0

Figure 13-10 • **I/O Technology Table**

I/O Bank Resource Usage

This is an important portion of the report. The user must meet the requirements stated in this table. Figure 13-11 shows the I/O Bank Resource Usage table included in the I/O bank report:

```
I/O Bank Resource Usage:

          |   Voltages    | Single I/Os  | Diff I/O Pairs |        Vref I/Os
          |---------------|--------------|----------------|-------------------
          | Vcci  | Vref  | Used | Total | Used  | Total  | Used | Total | Vref Pins
    ------|-------|-------|------|-------|-------|--------|------|-------|----------
    Bank0 | N/A   | N/A   | 0    | 25    | 0     | 12     | N/A  | N/A   | N/A
    Bank1 | N/A   | N/A   | 0    | 15    | 0     | 7      | N/A  | N/A   | N/A
    Bank2 | N/A   | N/A   | 0    | 17    | 0     | 6      | N/A  | N/A   | N/A
    Bank3 | N/A   | N/A   | 0    | 16    | 0     | 7      | N/A  | N/A   | N/A
    Bank4 | N/A   | N/A   | 0    | 15    | 0     | 7      | N/A  | N/A   | N/A
    Bank5 | N/A   | N/A   | 0    | 22    | 0     | 10     | N/A  | N/A   | N/A
    Bank6 | N/A   | N/A   | 0    | 19    | 0     | 9      | N/A  | N/A   | N/A
    Bank7 | N/A   | N/A   | 0    | 18    | 0     | 7      | N/A  | N/A   | N/A

Warning: IOPRL1: 8 I/O Bank(s) have not been assigned any voltages.
         The I/O modules located in these banks cannot be assigned any I/O macro.
```

Figure 13-11 • **I/O Bank Resource Usage Table**

The example in Figure 13-11 shows that none of the I/O macros is assigned to the bank because more than one V_{CCI} is detected.

I/O Voltage Usage

The I/O Voltage Usage table provides the number of V_{REF} (E devices only) and V_{CCI} assignments required in the design. If the user decides to make I/O assignments manually (PDC or MVN), the issues listed in this table must be resolved before proceeding to Layout. As stated earlier, V_{REF} assignments must be made if there are any voltage-referenced I/Os.

Figure 13-12 shows the I/O Voltage Usage table included in the I/O bank report.

```
I/O Voltage Usage:

      Voltages    |     I/Os
    --------------|--------------
     Vcci  | Vref  | Used | Total
    -------|-------|------|-------
     1.50v | N/A   | 1*   | 0
     1.80v | N/A   | 1*   | 0
     2.50v | N/A   | 4*   | 0
     3.30v | N/A   | 6*   | 0
     3.30v | 1.00v | 1*   | 0
     3.30v | 1.50v | 1*   | 0

Warning: IOPRL3: This design has infeasible I/O voltage requirement(s),
         which are indicated with a '*' in the I/O Voltage Usage table.
         Please consider importing a Physical Design Constraint (PDC) file or
         use the MultiView Navigator (MVN) to resolve the design's voltage requirements
         before running Layout.
```

Figure 13-12 • **I/O Voltage Usage Table**

The table in Figure 13-12 indicates that there are two voltage-referenced I/Os used in the design. Even though both of the voltage-referenced I/O technologies have the same V_{CCI} voltage, their V_{REF} voltages are different. As a result, two I/O banks are needed to assign the V_{CCI} and V_{REF} voltages.

In addition, there are six single-ended I/Os used that have the same V_{CCI} voltage. Since two banks are already assigned with the same V_{CCI} voltage and there are enough unused bonded I/Os in those banks, the user does not need to assign the same V_{CCI} voltage to another bank. The user needs to assign the other three V_{CCI} voltages to three more banks.

Assigning Technologies and V_{REF} to I/O Banks

Low-power flash devices offer a wide variety of I/O standards, including voltage-referenced standards. Before proceeding to Layout, each bank must have the required V_{CCI} voltage assigned for the corresponding I/O technologies used for that bank. The voltage-referenced standards require the use of a reference voltage (V_{REF}). This assignment can be done manually or automatically. The following sections describe this in detail.

Manually Assigning Technologies to I/O Banks

The user can import the PDC at this point and resolve this requirement. The PDC command is

```
set_iobank [bank name] –vcci [vcci value]
```

Another method is to use the I/O Bank Settings dialog box (**MVN** > **Edit** > **I/O Bank Settings**) to set up the V_{CCI} voltage for the bank (Figure 13-13).

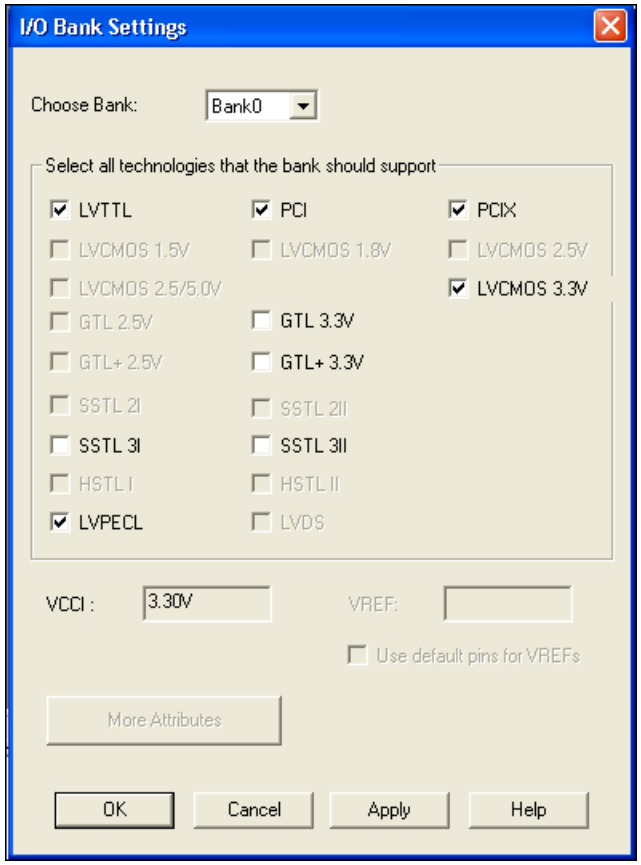


Figure 13-13 • **Setting V_{CCI} for a Bank**

The procedure is as follows:

1. Select the bank to which you want V_{CCI} to be assigned from the **Choose Bank** list.
2. Select the I/O standards for that bank. If you select any standard, the tool will automatically show all compatible standards that have a common V_{CCI} voltage requirement.
3. Click **Apply**.
4. Repeat steps 1–3 to assign V_{CCI} voltages to other banks. Refer to Figure 13-12 on page 13-13 to find out how many I/O banks are needed for V_{CCI} bank assignment.

Manually Assigning V_{REF} Pins

Voltage-referenced inputs require an input reference voltage (V_{REF}). The user must assign V_{REF} pins before running Layout. Before assigning a V_{REF} pin, the user must set a V_{REF} technology for the bank to which the pin belongs.

V_{REF} Rules for the Implementation of Voltage-Referenced I/O Standards

The V_{REF} rules are as follows:

1. Any I/O (except JTAG I/Os) can be used as a V_{REF} pin.
2. One V_{REF} pin can support up to 15 I/Os. It is recommended, but not required, that eight of them be on one side and seven on the other side (in other words, all 15 can still be on one side of V_{REF}).
3. SSTL3 (I) and (II): Up to 40 I/Os per north or south bank in any position
4. LVPECL / GTL+ 3.3 V / GTL 3.3 V: Up to 48 I/Os per north or south bank in any position
5. SSTL2 (I) and (II) / GTL+ 2.5 V / GTL 2.5 V: Up to 72 I/Os per north or south bank in any position
6. V_{REF} minibanks partition rule: Each I/O bank is physically partitioned into V_{REF} minibanks. The V_{REF} pins within a V_{REF} minibank are interconnected internally, and consequently, only one V_{REF} voltage can be used within each V_{REF} minibank. If a bank does not require a V_{REF} signal, the V_{REF} pins of that bank are available as user I/Os.
7. The first V_{REF} minibank includes all I/Os starting from one end of the bank to the first power triple and eight more I/Os after the power triple. Therefore, the first V_{REF} minibank may contain (0 + 8), (2 + 8), (4 + 8), (6 + 8), or (8 + 8) I/Os.

   The second V_{REF} minibank is adjacent to the first V_{REF} minibank and contains eight I/Os, a power triple, and eight more I/Os after the triple. An analogous rule applies to all other V_{REF} minibanks but the last.

   The last V_{REF} minibank is adjacent to the previous one but contains eight I/Os, a power triple, and all I/Os left at the end of the bank. This bank may also contain (8 + 0), (8 + 2), (8 + 4), (8 + 6), or (8 + 8) available I/Os.

Example:

4 I/Os → Triple → 8 I/Os, 8 I/Os → Triple → 8 I/Os, 8 I/Os → Triple → 2 I/Os

That is, minibank A = (4 + 8) I/Os, minibank B = (8 + 8) I/Os, minibank C = (8 + 2) I/Os.

Assigning the V_{REF} Voltage to a Bank

When importing the PDC file, the V_{REF} voltage can be assigned to the I/O bank. The PDC command is as follows:

```
set_iobank –vref [value]
```

Another method for assigning V_{REF} is by using **MVN** > **Edit** > **I/O Bank Settings** (Figure 13-14 on page 13-16).

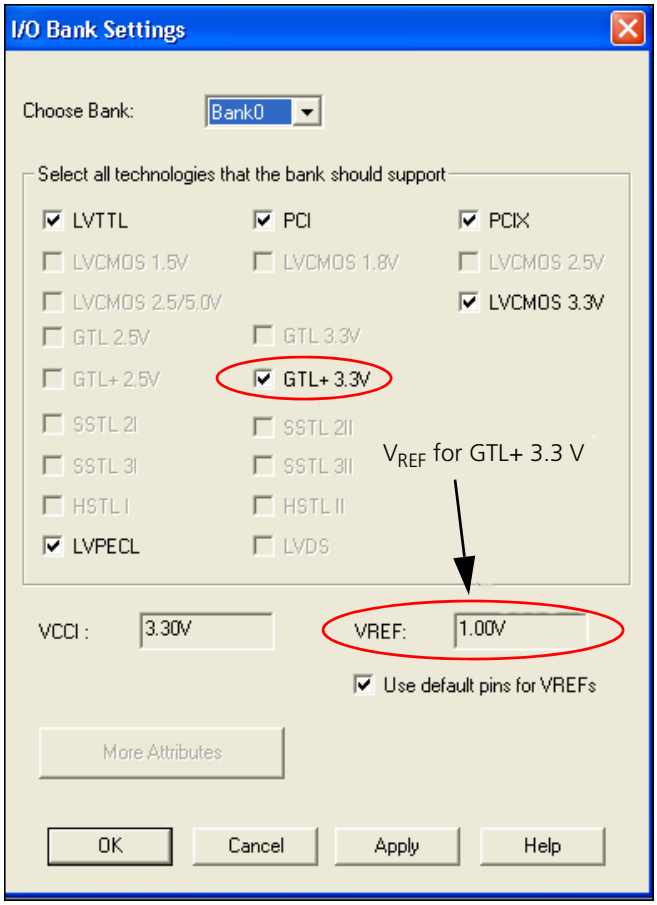


Figure 13-14 • **Selecting V_{REF} Voltage for the I/O Bank**

Assigning V_{REF} Pins for a Bank

The user can use default pins for V_{REF}. In this case, select the **Use default pins for VREFs** check box (Figure 13-14). This option guarantees full V_{REF} coverage of the bank. The equivalent PDC command is as follows:

```
set_vref_default [bank name]
```

To be able to choose V_{REF} pins, adequate V_{REF} pins must be created to allow legal placement of the compatible voltage-referenced I/Os.

To assign V_{REF} pins manually, the PDC command is as follows:

```
set_vref –bank [bank name] [package pin numbers]
```

For ChipPlanner/PinEditor to show the range of a V_{REF} pin, perform the following steps:

1. Assign V_{CCI} to a bank using **MVN > Edit > I/O Bank Settings**.
2. Open **ChipPlanner**. Zoom in on an I/O package pin in that bank.
3. Highlight the pin and then right-click. Choose **Use Pin for VREF**.

4. Right-click and then choose **Highlight VREF range**. All the pins covered by that V_{REF} pin will be highlighted (Figure 13-15).

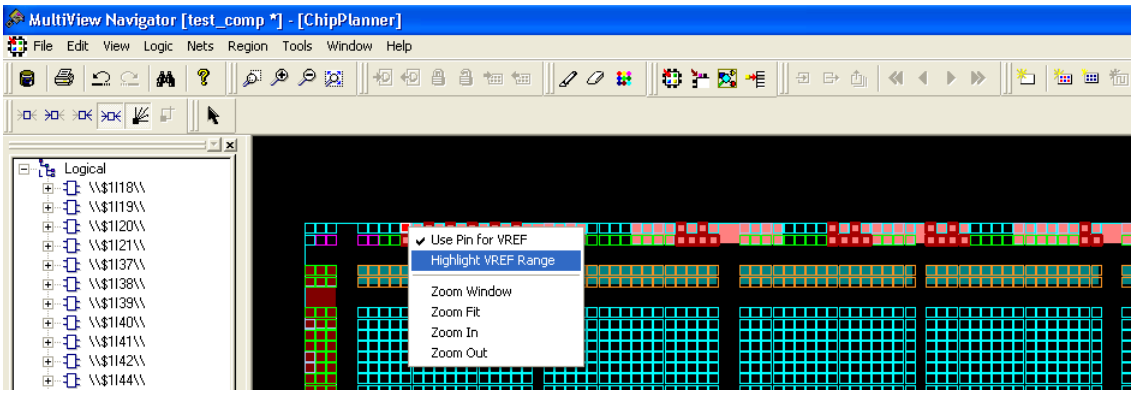


Figure 13-15 • V_{REF} Range

Using PinEditor or ChipPlanner, V_{REF} pins can also be assigned (Figure 13-16).

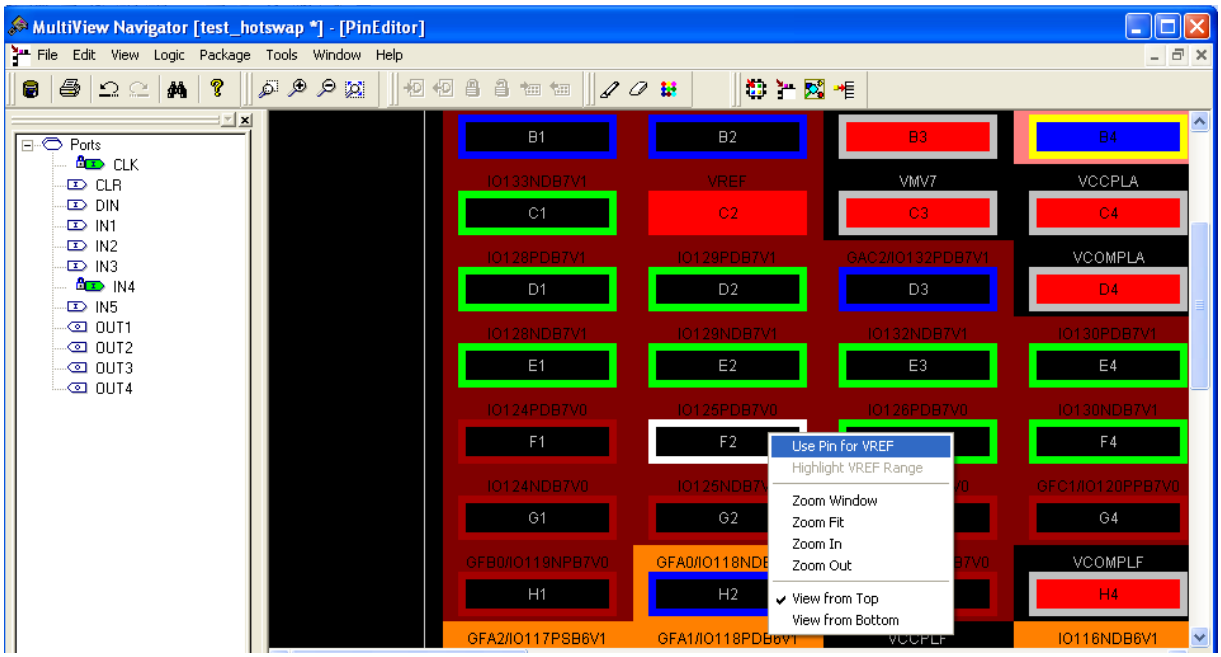


Figure 13-16 • Assigning V_{REF} from PinEditor

To unassign a V_{REF} pin:

1. Select the pin to unassign.
2. Right-click and choose **Use Pin for VREF.** The check mark next to the command disappears. The V_{REF} pin is now a regular pin.

Resetting the pin may result in unassigning I/O cores, even if they are locked. In this case, a warning message appears so you can cancel the operation.

After you assign the V_{REF} pins, right-click a V_{REF} pin and choose **Highlight VREF Range** to see how many I/Os are covered by that pin. To unhighlight the range, choose **Unhighlight All** from the **Edit** menu.

Automatically Assigning Technologies to I/O Banks

The I/O Bank Assigner (IOBA) tool runs automatically when you run Layout. You can also use this tool from within the MultiView Navigator (Figure 13-18). The IOBA tool automatically assigns technologies and V_{REF} pins (if required) to every I/O bank that does not currently have any technologies assigned to it. This tool is available when at least one I/O bank is unassigned.

To automatically assign technologies to I/O banks, choose **I/O Bank Assigner** from the **Tools** menu (or click the I/O Bank Assigner's toolbar button, shown in Figure 13-17).

***Figure 13-17* • I/O Bank Assigner's Toolbar Button**

Messages will appear in the Output window informing you when the automatic I/O bank assignment begins and ends. If the assignment is successful, the message "I/O Bank Assigner completed successfully" appears in the Output window, as shown in Figure 13-18.

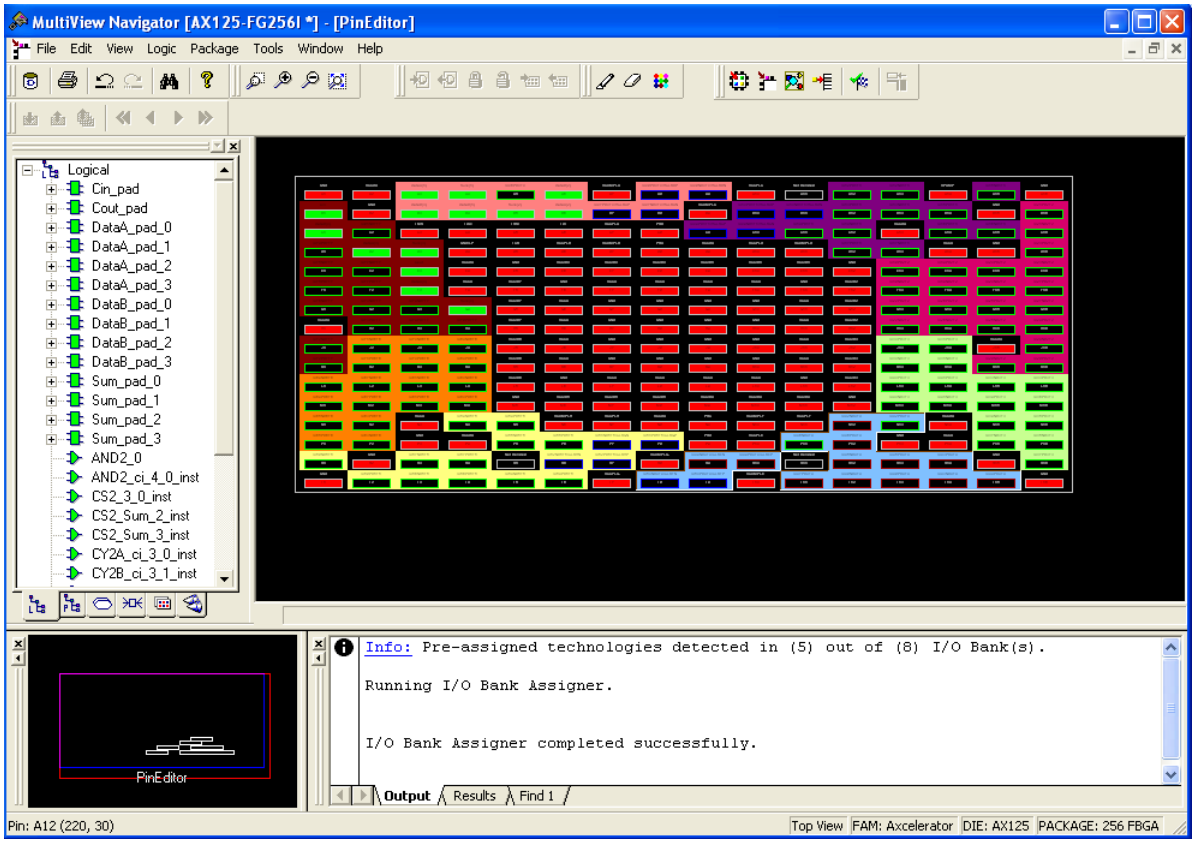


***Figure 13-18* • I/O Bank Assigner Displays Messages in Output Window**

If the assignment is not successful, an error message appears in the Output window.

To undo the I/O bank assignments, choose **Undo** from the **Edit** menu. Undo removes the I/O technologies assigned by the IOBA. It does not remove the I/O technologies previously assigned.

To redo the changes undone by the Undo command, choose **Redo** from the **Edit** menu.

To clear I/O bank assignments made before using the Undo command, manually unassign or reassign I/O technologies to banks. To do so, choose **I/O Bank Settings** from the **Edit** menu to display the I/O Bank Settings dialog box.

Conclusion

Actel Fusion, IGLOO, and ProASIC3 support for multiple I/O standards minimizes board-level components and makes possible a wide variety of applications. The Actel Designer software, integrated with Actel Libero IDE, presents a clear visual display of I/O assignments, allowing users to verify I/O and board-level design requirements before programming the device. The device I/O features and functionalities ensure board designers can produce low-cost and low-power FPGA applications fulfilling the complexities of contemporary design needs.

Related Documents

Handbook Documents

DDR for Actel's Low-Power Flash Devices
http://www.actel.com/documents/LPD_DDR_HBs.pdf
*Flash*Freeze Technology and Low-Power Modes in IGLOO and ProASIC3L Devices*
http://www.actel.com/documents/LPD_FlashFreeze_HBs.pdf
Global Resources in Actel Low-Power Flash Devices
http://www.actel.com/documents/LPD_Global_HBs.pdf
I/O Structures in IGLOO and ProASIC3 Devices
http://www.actel.com/documents/IGLOO_PA3_IO_HBs.pdf
I/O Structures in IGLOO PLUS Devices
http://www.actel.com/documents/IGLOOPLUS_IO_HBs.pdf
I/O Structures in IGLOOe and ProASIC3E Devices
http://www.actel.com/documents/IGLOOe_PA3E_IO_HBs.pdf
Pin Descriptions
http://www.actel.com/documents/LPD_PinDescriptions_HBs.pdf
Power-Up/-Down Behavior of Low-Power Flash Devices
http://www.actel.com/documents/LPD_PowerUp_HBs.pdf
ProASIC3/E SSO and Pin Placement and Guidelines
http://www.actel.com/documents/PA3_E_SSO_HBs.pdf

User's Guides

Actel Libero IDE User's Guide
http://www.actel.com/documents/libero_ug.pdf
IGLOO, Fusion, and ProASIC3 Macro Library Guide
http://www.actel.com/documents/pa3_libguide_ug.pdf
SmartGen Core Reference Guide
http://www.actel.com/documents/genguide_ug.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-026-3

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the document.

Previous Version	Changes in Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 13-1 · Flash-Based FPGAs.	13-2
	The notes for Table 13-2 · Designer State (resulting from I/O attribute modification) were revised to indicate that skew control and input delay do not apply to nano devices.	13-3
v1.2 (June 2008)	The "Flash FPGAs I/O Support" section was revised to include new families and make the information more concise.	13-2
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 13-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	13-2
v1.0 (January 2008)	This document was previously part of the *I/O Structures in IGLOO and ProASIC3 Devices* document. The content was separated and made into a new document.	N/A
	Table 13-2 · Designer State (resulting from I/O attribute modification) was updated to include note 2 for IGLOO PLUS.	13-3

14 – DDR for Actel's Low-Power Flash Devices

Introduction

The I/Os in Fusion, IGLOO,® and ProASIC®3 devices support Double Data Rate (DDR) mode. In this mode, new data is present on every transition (or clock edge) of the clock signal. This mode doubles the data transfer rate compared with Single Data Rate (SDR) mode, where new data is present on one transition (or clock edge) of the clock signal. Low-power flash devices have DDR circuitry built into the I/O tiles. I/Os are configured to be DDR receivers or transmitters by instantiating the appropriate special macros (examples shown in Figure 14-4 on page 14-6 and Figure 14-5 on page 14-7) and buffers (DDR_OUT or DDR_REG) in the RTL design. This document discusses the options the user can choose to configure the I/Os in this mode and how to instantiate them in the design.

Double Data Rate (DDR) Architecture

Low-power flash devices support 350 MHz DDR inputs and outputs. In DDR mode, new data is present on every transition of the clock signal. Clock and data lines have identical bandwidths and signal integrity requirements, making them very efficient for implementing very high-speed systems. High-speed DDR interfaces can be implemented using LVDS. In IGLOOe, ProASIC3E, AFS600, and AFS1500 devices, DDR interfaces can also be implemented using the HSTL, SSTL, and LVPECL I/O standards. The DDR feature is primarily implemented in the FPGA core periphery and is not tied to a specific I/O technology or limited to any I/O standard.

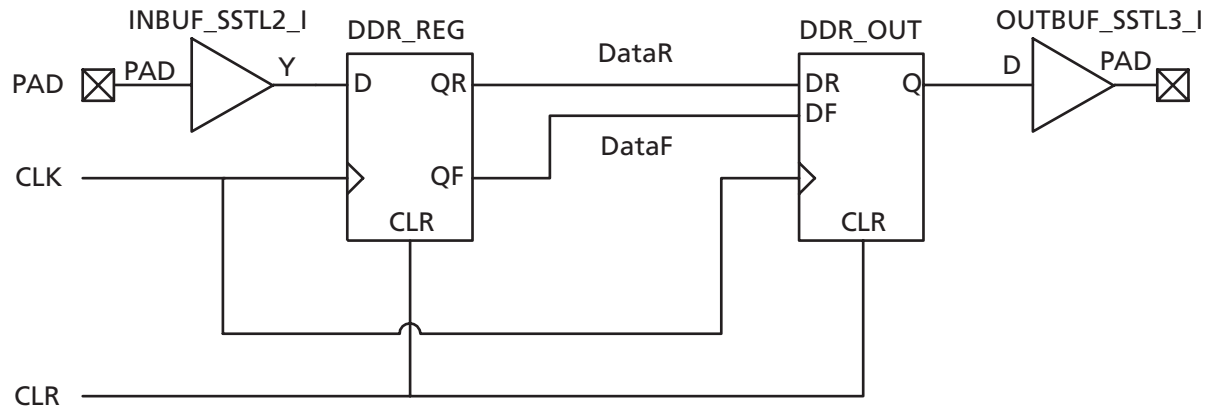


***Figure 14-1* • DDR Support in Low-Power Flash Devices**

DDR Support in Flash-Based Devices

The flash FPGAs listed in Table 14-1 support the DDR feature and the functions described in this document.

Table 14-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 14-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 14-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

I/O Cell Architecture

Low-power flash devices support DDR in the I/O cells in four different modes: Input, Output, Tristate, and Bidirectional pins. For each mode, different I/O standards are supported, with most I/O standards having special sub-options. For the ProASIC3 nano and IGLOO nano devices, DDR is supported only in the 60 k, 125 k, and 250 k logic densities. Refer to Table 14-2 for a sample of the available I/O options. Additional I/O options can be found in the relevant family datasheet.

Table 14-2 • **DDR I/O Options**

DDR Register Type	I/O Type	I/O Standard	Sub-Options	Comments
Receive Register	Input	Normal	None	3.3 V TTL (default)
		LVCMOS	Voltage	1.5 V, 1.8 V, 2.5 V, 5 V (1.5 V default)
			Pull-Up	None (default)
		PCI/PCI-X	None	
		GTL/GTL+	Voltage	2.5 V, 3.3 V (3.3 V default)
		HSTL	Class	I / II (I default)
		SSTL2/SSTL3	Class	I / II (I default)
		LVPECL	None	
		LVDS	None	
Transmit Register	Output	Normal	None	3.3 V TTL (default)
		LVTTL	Output Drive	2, 4, 6, 8, 12, 16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
		LVCMOS	Voltage	1.5 V, 1.8 V, 2.5 V, 5 V (1.5 V default)
		PCI/PCI-X	None	
		GTL/GTL+	Voltage	1.8 V, 2.5 V, 3.3 V (3.3 V default)
		HSTL	Class	I / II (I default)
		SSTL2/SSTL3	Class	I / II (I default)
		LVPECL	None	
		LVDS	None	

Table 14-2 • **DDR I/O Options (continued)**

DDR Register Type	I/O Type	I/O Standard	Sub-Options	Comments
Transmit Register (continued)	Tristate Buffer	Normal	Enable Polarity	Low/high (low default)
		LVTTL	Output Drive	2, 4, 6, 8, 12,16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
			Enable Polarity	Low/high (low default)
			Pull-Up/-Down	None (default)
		LVCMOS	Voltage	1.5 V, 1.8 V, 2.5 V, 5 V (1.5 V default)
			Output Drive	2, 4, 6, 8, 12, 16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
			Enable Polarity	Low/high (low default)
			Pull-Up/-Down	None (default)
		PCI/PCI-X	Enable Polarity	Low/high (low default)
		GTL/GTL+	Voltage	1.8 V, 2.5 V, 3.3 V (3.3 V default)
			Enable Polarity	Low/high (low default)
		HSTL	Class	I / II (I default)
			Enable Polarity	Low/high (low default)
		SSTL2/SSTL3	Class	I / II (I default)
			Enable Polarity	Low/high (low default)
	Bidirectional Buffer	Normal	Enable Polarity	Low/high (low default)
		LVTTL	Output Drive	2, 4, 6, 8, 12, 16, 24, 36 mA (8 mA default)
			Slew Rate	Low/high (high default)
			Enable Polarity	Low/high (low default)
			Pull-Up/-Down	None (default)
		LVCMOS	Voltage	1.5 V, 1.8 V, 2.5 V, 5 V (1.5 V default)
			Enable Polarity	Low/high (low default)
			Pull-Up	None (default)
		PCI/PCI-X	None	
			Enable Polarity	Low/high (low default)
		GTL/GTL+	Voltage	1.8 V, 2.5 V, 3.3 V (3.3 V default)
			Enable Polarity	Low/high (low default)
		HSTL	Class	I / II (I default)
			Enable Polarity	Low/high (low default)
		SSTL2/SSTL3	Class	I / II (I default)
			Enable Polarity	Low/high (low default)

Input Support for DDR

The basic structure to support a DDR input is shown in Figure 14-2. Three input registers are used to capture incoming data, which is presented to the core on each rising edge of the I/O register clock. Each I/O tile supports DDR inputs.

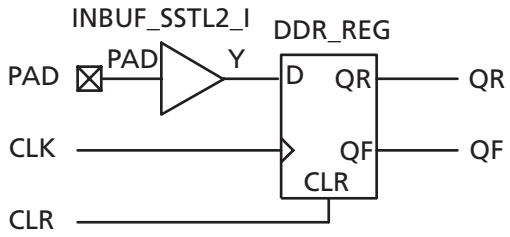


***Figure 14-2* • DDR Input Register Support in Low-Power Flash Devices**

Output Support for DDR

The basic DDR output structure is shown in Figure 14-1 on page 14-1. New data is presented to the output every half clock cycle.

Note: DDR macros and I/O registers do not require additional routing. The combiner automatically recognizes the DDR macro and pushes its registers to the I/O register area at the edge of the chip. The routing delay from the I/O registers to the I/O buffers is already taken into account in the DDR macro.

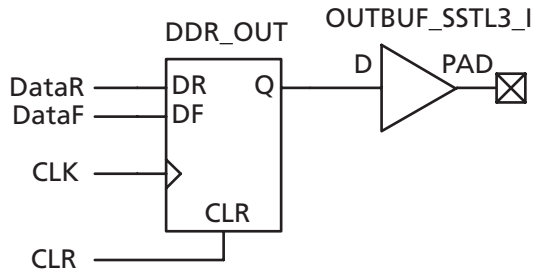


***Figure 14-3* • DDR Output Register (SSTL3 Class I)**

Instantiating DDR Registers

Using SmartGen is the simplest way to generate the appropriate RTL files for use in the design. Figure 14-4 shows an example of using SmartGen to generate a DDR SSTL2 Class I input register. SmartGen provides the capability to generate all of the DDR I/O cells as described. The user, through the graphical user interface, can select from among the many supported I/O standards. The output formats supported are Verilog, VHDL, and EDIF.

Figure 14-5 on page 14-7 through Figure 14-8 on page 14-10 show the I/O cell configured for DDR using SSTL2 Class I technology. For each I/O standard, the I/O pad is buffered by a special primitive that indicates the I/O standard type.

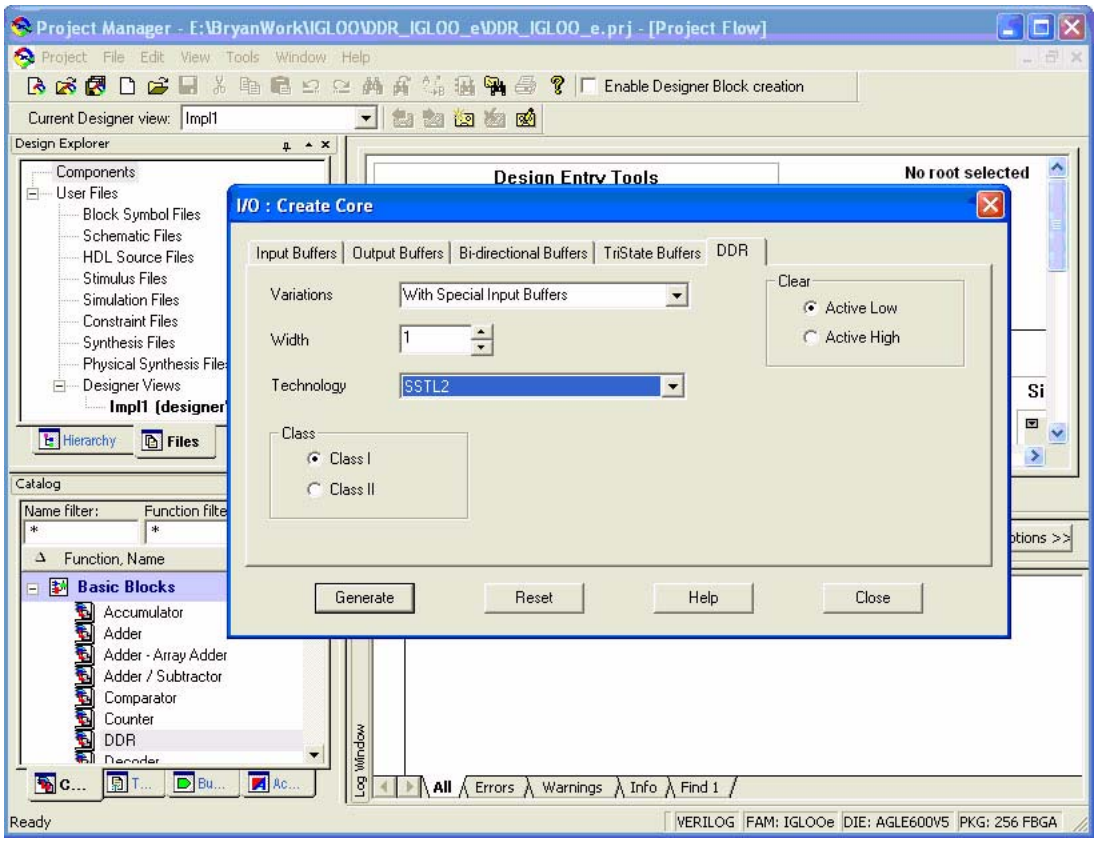


***Figure 14-4* • Example of Using SmartGen to Generate a DDR SSTL2 Class I Input Register**

DDR Input Register

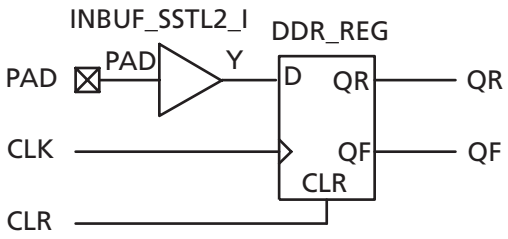


Figure 14-5 • **DDR Input Register (SSTL2 Class I)**

The corresponding structural representations, as generated by SmartGen, are shown below:

Verilog

```
module DDR_InBuf_SSTL2_I(PAD,CLR,CLK,QR,QF);

input   PAD, CLR, CLK;
output  QR, QF;

wire Y;

  INBUF_SSTL2_I INBUF_SSTL2_I_0_inst(.PAD(PAD),.Y(Y));
  DDR_REG DDR_REG_0_inst(.D(Y),.CLK(CLK),.CLR(CLR),.QR(QR),.QF(QF));

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
--The correct library will be inserted automatically by SmartGen
library proasic3; use proasic3.all;
--library fusion; use fusion.all;
--library igloo; use igloo.all;

entity DDR_InBuf_SSTL2_I is
  port(PAD, CLR, CLK : in std_logic;  QR, QF : out std_logic) ;
end DDR_InBuf_SSTL2_I;

architecture DEF_ARCH of  DDR_InBuf_SSTL2_I is

  component INBUF_SSTL2_I
    port(PAD : in std_logic := 'U'; Y : out std_logic) ;
  end component;

  component DDR_REG
    port(D, CLK, CLR : in std_logic := 'U'; QR, QF : out std_logic) ;
  end component;

signal Y : std_logic ;

begin

  INBUF_SSTL2_I_0_inst : INBUF_SSTL2_I
  port map(PAD => PAD, Y => Y);
  DDR_REG_0_inst : DDR_REG
  port map(D => Y, CLK => CLK, CLR => CLR, QR => QR, QF => QF);

end DEF_ARCH;
```

DDR Output Register

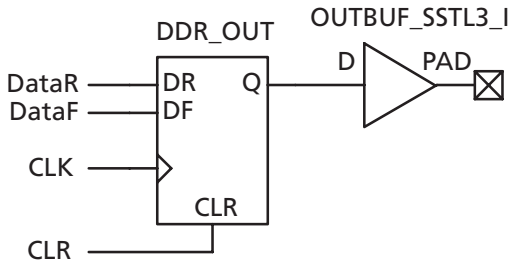


Figure 14-6 • **DDR Output Register (SSTL3 Class I)**

Verilog

```
module DDR_OutBuf_SSTL3_I(DataR,DataF,CLR,CLK,PAD);

input   DataR, DataF, CLR, CLK;
output  PAD;

wire Q, VCC;

  VCC VCC_1_net(.Y(VCC));
  DDR_OUT DDR_OUT_0_inst(.DR(DataR),.DF(DataF),.CLK(CLK),.CLR(CLR),.Q(Q));
  OUTBUF_SSTL3_I OUTBUF_SSTL3_I_0_inst(.D(Q),.PAD(PAD));

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
library proasic3; use proasic3.all;

entity DDR_OutBuf_SSTL3_I is
  port(DataR, DataF, CLR, CLK : in std_logic;  PAD : out std_logic) ;
end DDR_OutBuf_SSTL3_I;

architecture DEF_ARCH of  DDR_OutBuf_SSTL3_I is

  component DDR_OUT
    port(DR, DF, CLK, CLR : in std_logic := 'U'; Q : out std_logic) ;
  end component;

  component OUTBUF_SSTL3_I
    port(D : in std_logic := 'U'; PAD : out std_logic) ;
  end component;

  component VCC
    port( Y : out std_logic);
  end component;

signal Q, VCC_1_net : std_logic ;

begin

  VCC_2_net : VCC port map(Y => VCC_1_net);
  DDR_OUT_0_inst : DDR_OUT
  port map(DR => DataR, DF => DataF, CLK => CLK, CLR => CLR, Q => Q);
  OUTBUF_SSTL3_I_0_inst : OUTBUF_SSTL3_I
  port map(D => Q, PAD => PAD);

end DEF_ARCH;
```

DDR Tristate Output Register

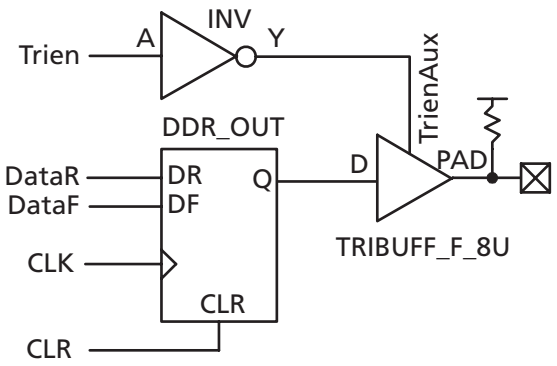


Figure 14-7 • **DDR Tristate Output Register, LOW Enable, 8 mA, Pull-Up (LVTTL)**

Verilog

```
module DDR_TriStateBuf_LVTTL_8mA_HighSlew_LowEnb_PullUp(DataR, DataF, CLR, CLK, Trien,
  PAD);

input   DataR, DataF, CLR, CLK, Trien;
output  PAD;

wire TrienAux, Q;

  INV Inv_Tri(.A(Trien),.Y(TrienAux));
  DDR_OUT DDR_OUT_0_inst(.DR(DataR),.DF(DataF),.CLK(CLK),.CLR(CLR),.Q(Q));
  TRIBUFF_F_8U TRIBUFF_F_8U_0_inst(.D(Q),.E(TrienAux),.PAD(PAD));

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
library proasic3; use proasic3.all;

entity DDR_TriStateBuf_LVTTL_8mA_HighSlew_LowEnb_PullUp is
  port(DataR, DataF, CLR, CLK, Trien : in std_logic;  PAD : out std_logic) ;
end DDR_TriStateBuf_LVTTL_8mA_HighSlew_LowEnb_PullUp;

architecture DEF_ARCH of DDR_TriStateBuf_LVTTL_8mA_HighSlew_LowEnb_PullUp is

  component INV
    port(A : in std_logic := 'U'; Y : out std_logic) ;
  end component;

  component DDR_OUT
    port(DR, DF, CLK, CLR : in std_logic := 'U'; Q : out std_logic) ;
  end component;

  component TRIBUFF_F_8U
    port(D, E : in std_logic := 'U'; PAD : out std_logic) ;
  end component;

signal TrienAux, Q : std_logic ;

begin

  Inv_Tri : INV
```

```
  port map(A => Trien, Y => TrienAux);
  DDR_OUT_0_inst : DDR_OUT
  port map(DR => DataR, DF => DataF, CLK => CLK, CLR => CLR, Q => Q);
  TRIBUFF_F_8U_0_inst : TRIBUFF_F_8U
  port map(D => Q, E => TrienAux, PAD => PAD);

end DEF_ARCH;
```

DDR Bidirectional Buffer

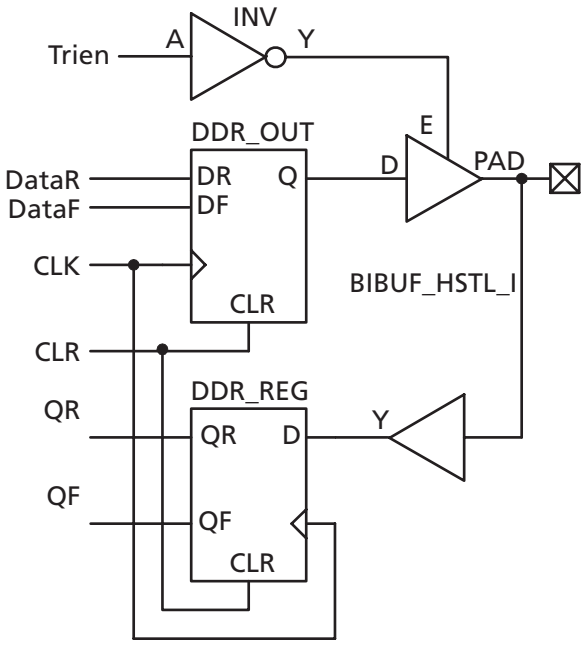


Figure 14-8 • **DDR Bidirectional Buffer, LOW Output Enable (HSTL Class II)**

Verilog

```
module DDR_BiDir_HSTL_I_LowEnb(DataR,DataF,CLR,CLK,Trien,QR,QF,PAD);

input   DataR, DataF, CLR, CLK, Trien;
output  QR, QF;
inout   PAD;

wire TrienAux, D, Q;

  INV Inv_Tri(.A(Trien), .Y(TrienAux));
  DDR_OUT DDR_OUT_0_inst(.DR(DataR),.DF(DataF),.CLK(CLK),.CLR(CLR),.Q(Q));
  DDR_REG DDR_REG_0_inst(.D(D),.CLK(CLK),.CLR(CLR),.QR(QR),.QF(QF));
  BIBUF_HSTL_I BIBUF_HSTL_I_0_inst(.PAD(PAD),.D(Q),.E(TrienAux),.Y(D));

endmodule
```

VHDL

```
library ieee;
use ieee.std_logic_1164.all;
library proasic3; use proasic3.all;

entity DDR_BiDir_HSTL_I_LowEnb is
  port(DataR, DataF, CLR, CLK, Trien : in std_logic; QR, QF : out std_logic;
    PAD : inout std_logic) ;
end DDR_BiDir_HSTL_I_LowEnb;

architecture DEF_ARCH of  DDR_BiDir_HSTL_I_LowEnb is

  component INV
    port(A : in std_logic := 'U'; Y : out std_logic) ;
  end component;

  component DDR_OUT
    port(DR, DF, CLK, CLR : in std_logic := 'U'; Q : out std_logic) ;
  end component;

  component DDR_REG
    port(D, CLK, CLR : in std_logic := 'U'; QR, QF : out std_logic) ;
  end component;

  component BIBUF_HSTL_I
    port(PAD : inout std_logic := 'U'; D, E : in std_logic := 'U'; Y : out std_logic) ;
  end component;

signal TrienAux, D, Q : std_logic ;

begin

  Inv_Tri : INV
  port map(A => Trien, Y => TrienAux);
  DDR_OUT_0_inst : DDR_OUT
  port map(DR => DataR, DF => DataF, CLK => CLK, CLR => CLR, Q => Q);
  DDR_REG_0_inst : DDR_REG
  port map(D => D, CLK => CLK, CLR => CLR, QR => QR, QF => QF);
  BIBUF_HSTL_I_0_inst : BIBUF_HSTL_I
  port map(PAD => PAD, D => Q, E => TrienAux, Y => D);

end DEF_ARCH;
```

Design Example

Figure 14-9 shows a simple example of a design using both DDR input and DDR output registers. The user can copy the HDL code in Actel Libero® Integrated Design Environment (IDE) and go through the design flow. Figure 14-10 and Figure 14-11 on page 14-13 show the netlist and ChipPlanner views of the ddr_test design. Diagrams may vary slightly for different families.

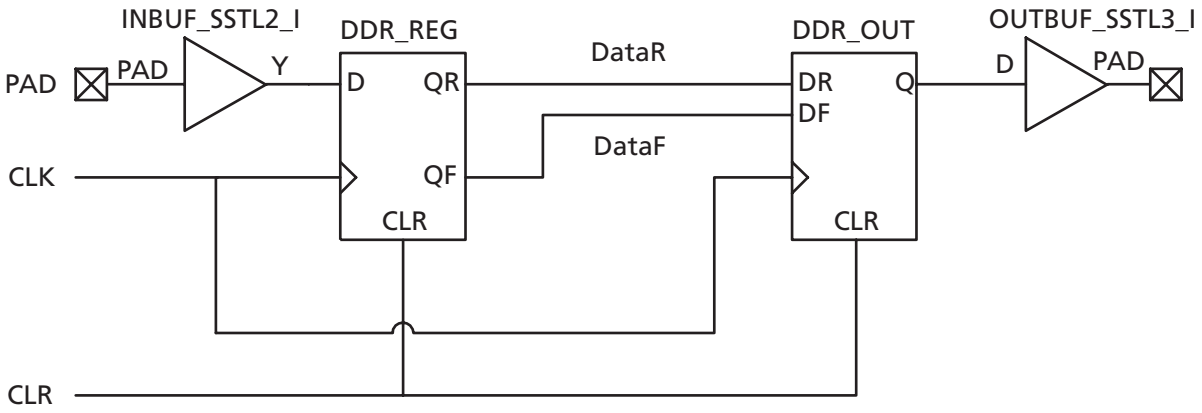


Figure 14-9 • **Design Example**

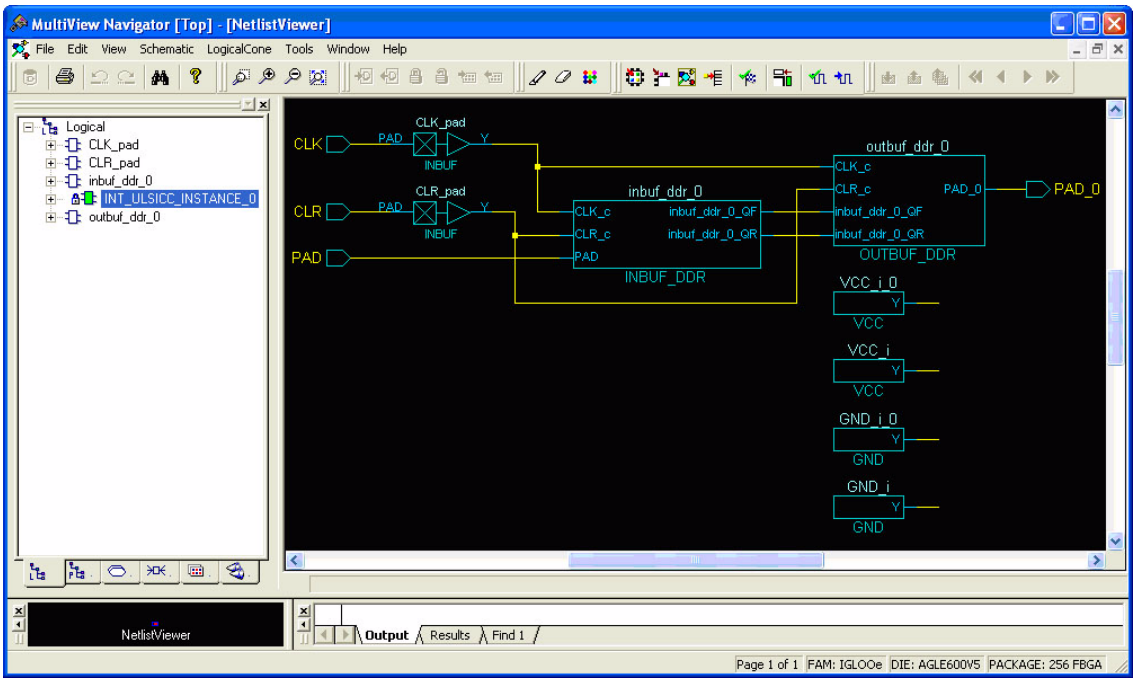


Figure 14-10 • **DDR Test Design as Seen by NetlistViewer for IGLOO/e Devices**

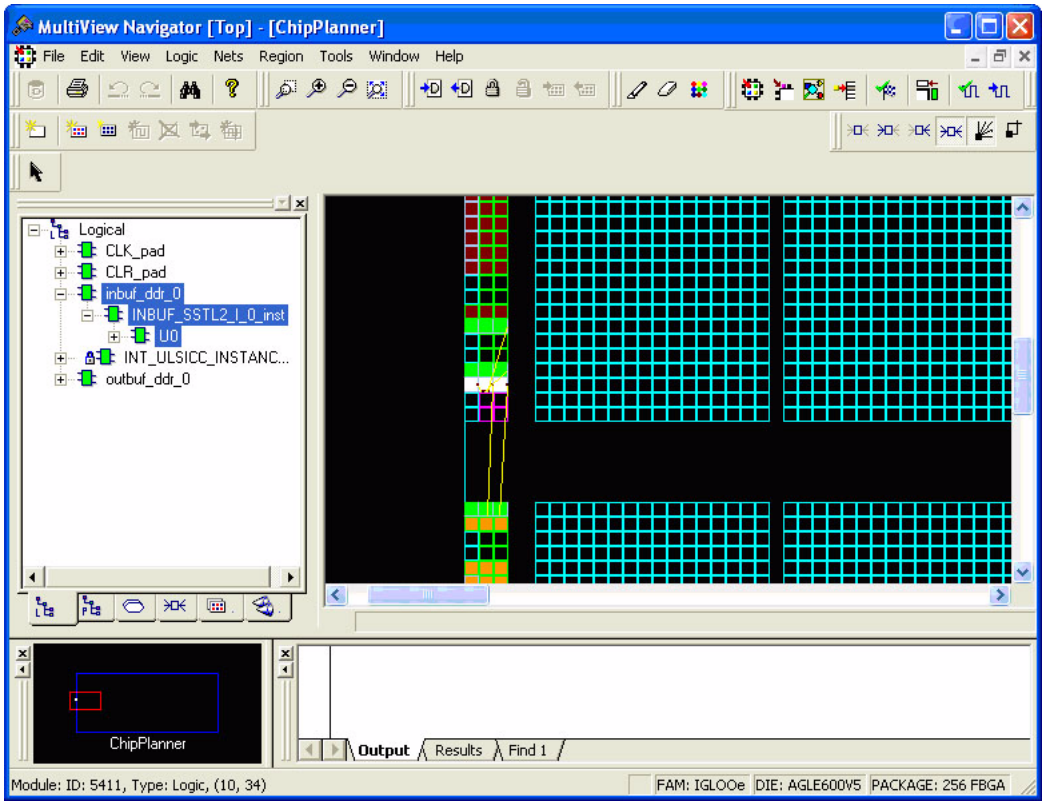


Figure 14-11 • **DDR Input/Output Cells as Seen by ChipPlanner for IGLOO/e Devices**

Verilog

```
module Inbuf_ddr(PAD,CLR,CLK,QR,QF);

input PAD, CLR, CLK;
output  QR, QF;

wire Y;

  DDR_REG DDR_REG_0_inst(.D(Y), .CLK(CLK), .CLR(CLR), .QR(QR), .QF(QF));
  INBUF INBUF_0_inst(.PAD(PAD), .Y(Y));

endmodule

module Outbuf_ddr(DataR,DataF,CLR,CLK,PAD);

input DataR, DataF, CLR, CLK;
output  PAD;

wire Q, VCC;

  VCC VCC_1_net(.Y(VCC));
  DDR_OUT DDR_OUT_0_inst(.DR(DataR), .DF(DataF), .CLK(CLK), .CLR(CLR), .Q(Q));
  OUTBUF OUTBUF_0_inst(.D(Q), .PAD(PAD));

endmodule
```

```
module ddr_test(DIN, CLK, CLR, DOUT);

input  DIN, CLK, CLR;
output DOUT;

  Inbuf_ddr Inbuf_ddr (.PAD(DIN), .CLR(clr), .CLK(clk), .QR(qr), .QF(qf));
  Outbuf_ddr Outbuf_ddr (.DataR(qr),.DataF(qf), .CLR(clr), .CLK(clk),.PAD(DOUT));

  INBUF INBUF_CLR (.PAD(CLR), .Y(clr));
  INBUF INBUF_CLK (.PAD(CLK), .Y(clk));

endmodule
```

Simulation Consideration

Actel DDR simulation models use inertial delay modeling by default (versus transport delay modeling). As such, pulses that are shorter than the actual gate delays should be avoided, as they will not be seen by the simulator and may be an issue in post-routed simulations. The user must be aware of the default delay modeling and must set the correct delay model in the simulator as needed.

Conclusion

Fusion, IGLOO, and ProASIC3 devices support a wide range of DDR applications with different I/O standards and include built-in DDR macros. The powerful capabilities provided by SmartGen and its GUI can simplify the process of including DDR macros in designs and minimize design errors. Additional considerations should be taken into account by the designer in design floorplanning and placement of I/O flip-flops to minimize datapath skew and to help improve system timing margins. Other system-related issues to consider include PLL and clock partitioning.

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-010-4

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in the Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 14-1 · Flash-Based FPGAs.	14-2
	The "I/O Cell Architecture" section was updated with information applicable to nano devices.	14-3
	The output buffer (OUTBUF_SSTL3_I) input was changed to D, instead of Q, in Figure 14-1 · DDR Support in Low-Power Flash Devices, Figure 14-3 · DDR Output Register (SSTL3 Class I), Figure 14-6 · DDR Output Register (SSTL3 Class I), Figure 14-7 · DDR Tristate Output Register, LOW Enable, 8 mA, Pull-Up (LVTTL), and the output from the DDR_OUT macro was connected to the input of the TRIBUFF macro in Figure 14-7 · DDR Tristate Output Register, LOW Enable, 8 mA, Pull-Up (LVTTL).	14-1, 14-5, 14-8, 14-9
v1.2 (June 2008)	The "Double Data Rate (DDR) Architecture" section was updated to include mention of the AFS600 and AFS1500 devices.	14-1
	The "DDR Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	14-2
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 14-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	14-2
v1.0 (January 2008)	The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	14-2

Design Migration

Actel
POWER MATTERS

15 – Prototyping With AFS600 for Smaller Devices

This document is designed as an aid for customers who may ultimately wish to use one of the smaller members of the Fusion family (AFS090 and/or AFS250). The first device available in the Fusion family is the AFS600, which can be used as a development or prototyping platform for the smaller devices. This document will help highlight differences between the AFS600 and these smaller devices in order to ease transition to the targeted device when it becomes available.

The Actel Fusion® family, based on the highly successful ProASIC®3E and ProASIC3 Flash FPGA architecture, has been designed as a high-performance, programmable, mixed-signal platform. By combining an advanced flash FPGA core with embedded flash memory and analog peripherals, Fusion devices dramatically simplify system design, and save overall system cost and board space as a result. Figure 15-1 shows the Fusion device architecture overview.

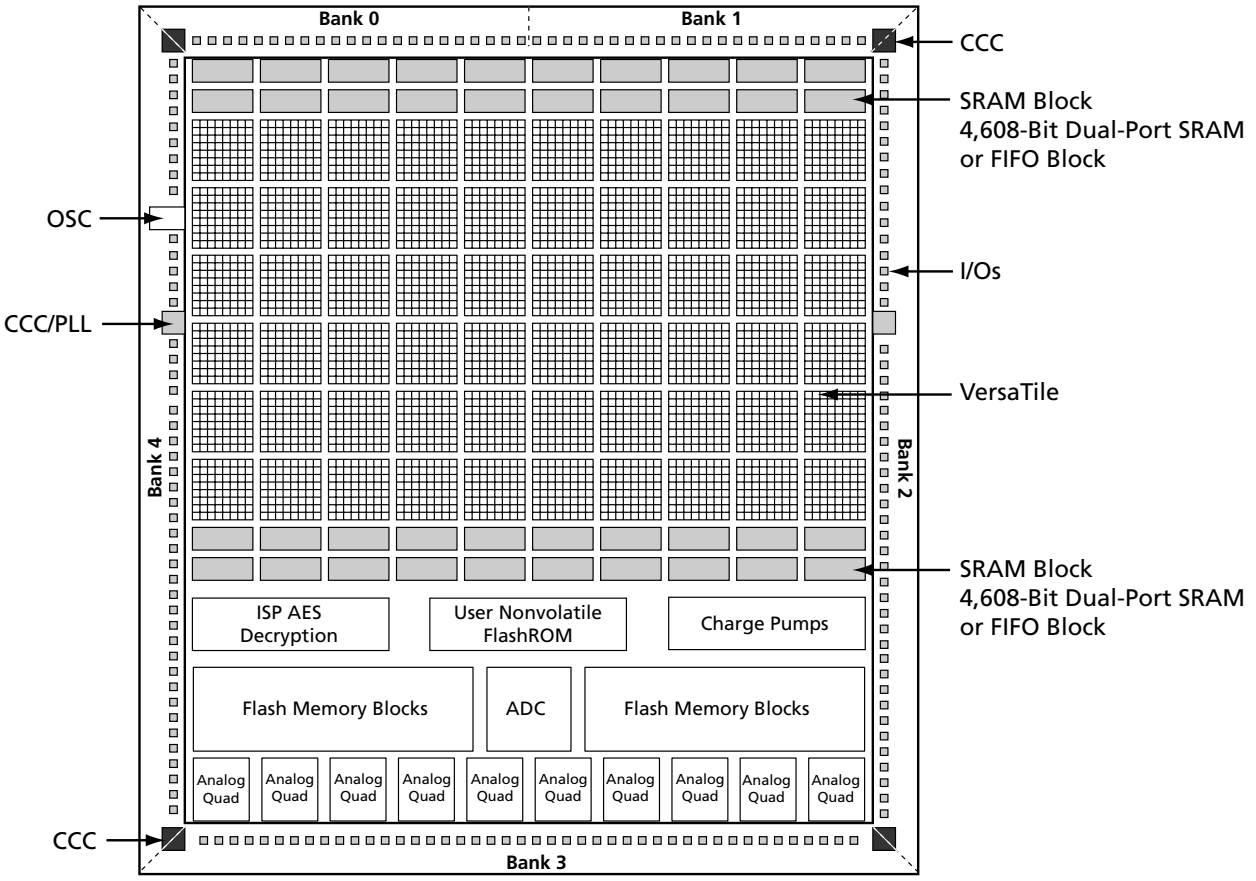


***Figure 15-1* • Fusion Device Architecture Overview**

The state-of-the-art embedded flash memory technology offers high-density integrated flash memory arrays, enabling savings in cost, power, and board area relative to external flash solutions, while providing increased flexibility and performance.

Fusion devices offer a robust and flexible analog mixed-signal addition to the high-performance flash FPGA fabric and embedded flash memory. The many built-in analog peripherals include a configurable 32:1 input analog multiplexer, up to 10 independent metal-oxide semiconductor field-effect transistor (MOSFET) gate driver outputs, and a configurable analog-to-digital converter (ADC). The Analog Quad is an I/O structure that contains three adjacent analog inputs and a gate driver output.

The addition of the real-time counter (RTC) system enables Fusion devices to support both standby and sleep modes of operation, greatly reducing power consumption in many applications.

Prototype Guideline

AFS090, AFS250, AFS600, and AFS1500 Device Configuration

AFS600 is a medium size device in the Fusion family. It supports all of the Fusion features, as shown in Table 15-1. The smaller devices (AFS090 and AFS250) in the Fusion family have lower gate counts and fewer memory blocks, I/Os, and PLLs.

Table 15-1 • **AFS090, AFS250, and AFS600 Device Summary**

Part Number		AFS090	AFS250	AFS600	AFS1500
General Information	System Gates	90,000	250,000	600,000	1,500,000
	Tiles (D-Flip-Flop)	2,304	6,144	13,824	38,400
	Secure (AES) ISP	Yes	Yes	Yes	Yes
	PLLs	1	1	2	2
	Globals	18	18	18	18
Memory	Flash Memory Blocks (256 kbytes)	1	1	2	4
	Flash Memory (kbytes)	256	256	512	1,024
	FlashROM Bits	1 k	1 k	1 k	1 k
	RAM Blocks (4,608 bits)	6	8	24	60
	RAM kbits	27	36	108	270
Analog	Analog Quads	5	6	10	10
	Analog Input Channels	15	18	30	30
	Gate Driver Outputs	5	6	10	10
I/O	I/O Types	Analog/ LVDS / Std+	Analog/ LVDS / Std+	Analog/ LVDS / Pro	Analog/ LVDS / Pro
	I/O Banks (+ JTAG)	4	4	5	5
	Maximum Digital I/Os	73	114	172	278
	Analog I/Os	20	24	40	40
I/O: Digital/Analog	QN108	36/14		–	–
	QN180	48/20	62/24	–	–
	PQ208	–	93/24	95/40	–
	FG256	73/20	114/24	119/40	119/40
	FG484	–	–	172/40	228/40
	FG676	–	–	–	278/40

Table 15-2 shows compatible devices for each package. The FG256 package is designed to support migration across all family members.

Table 15-2 • **Package Compatibility Table**

Package Types	PQ208	PQ208	FG256	FG484	FG676	QN108	QN180
Compatible Devices	AFS90	AFS600	AFS090	AFS600	AFS1500	AFS90	AFS090
	AFS250	AFS1500	AFS250	AFS1500			AFS250
			AFS600				
			AFS1500				

List of the Guidelines for Prototyping

Actel recommends AFS600-FG256 as the platform for prototyping smaller devices within the same compatible package type. AFS600-FG256 is also the first available Fusion silicon in the rollout roadmap and is used in the Fusion Starter Kit, which can serve as a prototype board to demonstrate the majority of Fusion features.

Memory Blocks

The AFS250 and AFS090 have a single 256-kbyte block of embedded flash memory, whereas the AFS600 has two 256-kbyte blocks (512 kbytes total). Therefore, the user must keep the usage less than 256 kbytes while doing prototyping with the AFS600.

The AFS250 has 8 RAM blocks, while the AFS600 has 24 RAM blocks. A SmartGen analog system generated soft IP uses 3 to 9 blocks of RAM. The user needs to keep track of the RAM block usage, especially if the design contains a RAM initialization application, data storage application or other applications that utilize extra RAM blocks. Usage must be no more than 8 blocks. Likewise, if the user is prototyping for AFS090, then the RAM block usage in AFS600 should not exceed 6 blocks.

PLLs

The AFS250 and AFS090 have one PLL on the west side of the device, whereas AFS600 has two PLLs—one for each side of the device. During prototyping using AFS600, the user should only implement the PLL on the west side and use the corresponding PLL input pin, so that the delays from the PLL input through the PLL to the global network have a minimum variation between AFS090/AFS250 and AFS600.

I/Os

All special function I/Os (V_{CC}, GND, JTAG, Programming Control, etc.) of the AFS250 and AFS090 devices are in exactly the same locations as in the AFS600 device, with one exception for the AFS090 as listed below. The AFS250 device has 6 Analog Quads, whereas the AFS600 device has 10 Analog Quads. The user should only use Analog Quads 0–5 while doing prototyping in AFS600, in order to have exactly the same analog pin map. To prototype AFS090, the user should only use Analog Quads 0–4, since AFS090 has 5 Analog Quads.

While the AFS250 differential I/Os have the same locations as the AFS600, the AFS090 differential I/O locations are slightly different from those of the AFS600. More details on the pin list are available in the *Fusion Datasheet*. The user should be aware that if the design has differential I/Os, the pinout needs to be changed from the AFS600 prototype design to an AFS090 production design.

Prototype Consideration in Software

After validating the design in the AFS600, the user needs to create a new Actel Libero® Integrated Design Environment (IDE) project for the AFS090 or AFS250, then recreate the SmartGen cores by using the same parameters used for the AFS600. All other source code used in the AFS600 project can be directly imported into the AFS090 or AFS250 project. The same validation process (simulation, static timing analysis, and functional test on silicon) should be performed for the AFS090 or AFS250 design as the user has done for AFS600.

Summary

AFS600-FG256 is the recommended Fusion prototyping vehicle for smaller devices in the same compatible package. It is also used on the Fusion Starter Kit board, which can demonstrate most of the Fusion family features.

Part Number and Revision Date

This document was previously published as an application note describing features and functions of the device, and as such has now been incorporated into the device handbook format. No technical changes have been made to the content.

Part Number 51700092-020-0

Revised October 2008

Programming and Security

Actel®
POWER MATTERS

16 – Programming Flash Devices

Introduction

This document provides an overview of the various programming options available for the Actel flash families. The electronic version of this document includes active links to all programming resources, which are available at http://www.actel.com/products/hardware/default.aspx. For Actel antifuse devices, refer to the *Programming Antifuse Devices* document.

Summary of Programming Support

FlashPro3 is a high-performance in-system programming (ISP) tool targeted at the latest generation of low-power flash devices offered by Actel: Fusion, IGLOO,® and ProASIC®3, including ARM®-enabled devices. FlashPro3 offers extremely high performance through the use of USB 2.0, is high-speed compliant for full use of the 480 Mbps bandwidth, and can program ProASIC3 devices in under 30 seconds. Powered exclusively via USB, FlashPro3 provides a V_{PUMP} voltage of 3.3 V for programming these devices.

Silicon Sculptor 3 is an easy-to-use, single-site programming tool for Actel FPGAs that delivers high data throughput and promotes ease of use while lowering the overall cost of ownership. Silicon Sculptor 3 includes a high-speed USB 2.0 interface that allows a customer to connect as many as 12 programmers to a single PC. Furthermore, Silicon Sculptor 3 is compatible with adapter modules from Silicon Sculptor II, thereby preserving a customer's investment and enabling a seamless upgrade to this latest generation of the tool.

For details of programmer support for each device, refer to Table 16-6 on page 16-10.

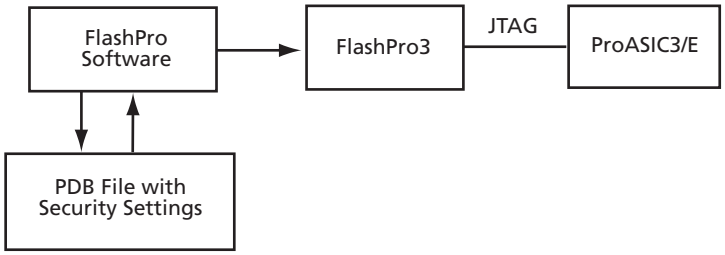


Figure 16-1 • **FlashPro Programming Setup**

Programming Support in Flash Devices

The flash FPGAs listed in Table 16-1 support flash in-system programming and the functions described in this document.

Table 16-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device
ProASIC	ProASIC	First generation ProASIC devices
	ProASIC^{PLUS}	Second generation ProASIC devices

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 16-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 16-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

General Flash Programming Information

Programming Basics

When choosing a programming solution, there are a number of options available. This section provides a brief overview of those options. The next sections provide more detail on those options as they apply to Actel FPGAs.

Reprogrammable or One-Time-Programmable (OTP)

Depending on the technology chosen, devices may be reprogrammable or one-time-programmable. As the name implies, a reprogrammable device can be programmed many times. Generally, the contents of such a device will be completely overwritten when it is reprogrammed. All Actel flash devices are reprogrammable.

An OTP device is programmable one time only. Once programmed, no more changes can be made to the contents. Actel flash devices provide the option of disabling the reprogrammability for security purposes. This combines the convenience of reprogrammability during design verification with the security of an OTP technology for highly sensitive designs.

Device Programmer or In-System Programming

There are two fundamental ways to program an FPGA: using a device programmer or, if the technology permits, using in-system programming. A device programmer is a piece of equipment in a lab or on the production floor that is used for programming devices. The devices are placed into a socket mounted in a programming adapter module, and the appropriate electrical interface is applied. The device can then be placed on the board. A typical programmer, used during development, programs a single device at a time and is referred to as a single-site engineering programmer.

With ISP, the device is already mounted onto the board when programming occurs, most typically via the JTAG pins. The JTAG pins can be controlled either by an on-board resource, such as a microprocessor, or by an off-board programmer through a header connection. Once mounted, it can be programmed repeatedly. If the application requires it, the system can be designed to reprogram itself using a microprocessor, without the use of any external programmer.

For production, high-volume multi-site production programmers handle designs that require device programmers. In addition, Actel can preprogram devices for production, negating the need for further programming. This service is referred to as in-house programming (IHP).

Live at Power-Up (LAPU) or Boot PROM

Utilizing the technology of the FPGA significantly impacts board-level power-up considerations. Some technologies are nonvolatile and are considered functional, or "live," as soon as power reaches the operational level. All Actel FPGA technologies are live at power-up. By contrast, SRAM technology is volatile, and devices built using SRAM cells lose their contents when power cycling occurs. These devices must be reprogrammed every time power is applied. Such a design must include nonvolatile storage for the contents as well as the means to reprogram. There is a delay before SRAM devices are functional; other parts of the board must come alive first to reprogram these types of FPGAs. Therefore, such devices can never be part of critical boot circuits.

Design Security

Design security is a growing concern for systems designers. The choice of programming methodology and technology affects system security. Use of Actel programming technology is the most secure option available, providing much better protection than SRAM-based devices and ASICs. Actel provides a number of ways to ensure designs are protected. General information on design security can be found on the Actel website:

http://www.actel.com/products/solutions/security/default.aspx

Programming Features for Actel Devices

Actel provides two types of FPGAs: flash and antifuse (Table 16-2). Some programming methods are common to both and some are exclusive to flash. This document describes only the programming solutions supported for flash devices.

Table 16-2 • **Programming Features for Actel Devices**

Feature	Flash	Antifuse
Reprogrammable	Yes	No
In-system programmable	Yes	No
One-time programmable	Yes (option)	Yes
Live at power-up	Yes	Yes
Secure	Yes	Yes
Single-site programmer support	Yes	Yes
Multi-site programmer support	Yes	Yes
In-house programming support	Yes	Yes

Flash Devices

The flash devices supplied by Actel are reprogrammable by either a generic device programmer or ISP. Actel supports ISP using JTAG, which is supported by the FlashPro3, FlashPro Lite, FlashPro, Silicon Sculptor 3, and Silicon Sculptor II programmers.

Levels of ISP support vary depending on the device chosen:

- All Fusion, IGLOO, and ProASIC3 devices support ISP.
- IGLOO, IGLOOe, IGLOO nano V5, and IGLOO PLUS devices can be programmed in-system when the device is using a 1.5 V supply voltage to the FPGA core.
- All IGLOO V2 and ProASIC3L devices can be operated at any voltage between 1.2 V and 1.5 V; the Designer software allows 50 mV increments in the voltage. Although the device can operate at 1.2 V core voltage, the device can only be reprogrammed when the core voltage is 1.5 V. Voltage switching is required in-system to switch from a 1.2 V supply (V_{CC}, V_{CCI}, and V_{JTAG}) to 1.5 V for programming.

Since flash devices are nonvolatile, they are live at power-up. This is different from an SRAM-based device, which loads its programming information when it is powered up. SRAM devices require a time on the order of hundreds of milliseconds before the system is active.

There are multiple levels of security available in flash devices. Use of a security key will lock the device. The device can then only be reprogrammed by first unlocking the device with the appropriate security key. It can also be locked permanently, which means there is no key that can access the device. The command to secure the device is embedded within the programming file, optionally enabled by the programming software. This is also referred to as the OTP version of flash, allowing for only a single programming instance. This is discussed in more detail in the *Implementation of Security in Actel's ProASIC and ProASIC^{PLUS} Flash-Based FPGAs* application note, and in the *Security in Low-Power Flash Devices* handbook section.

Flash devices can also be programmed using single-site or multi-site programmers as well as volume programming services from Actel or other vendors.

Types of Programming for Flash Devices

The number of devices to be programmed will influence the optimal programming methodology. Those available are listed below:

- In-system programming
  - Using a programmer
  - Using a microprocessor or microcontroller
- Device programmers
  - Single-site programmers
  - Multi-site programmers, batch programmers, or gang programmers
  - Automated production (robotic) programmers
- Volume programming services
  - Actel in-house programming
  - Programming centers

In-System Programming

Device Type Supported: Flash

ISP refers to programming the FPGA after it has been mounted on the system board. The FPGA may be preprogrammed and later reprogrammed using ISP.

The advantage of using ISP is the ability to update the FPGA design many times without any changes to the board. This eliminates the requirement of using a socket for the FPGA, saving cost and improving reliability. It also reduces programming hardware expenses, as the ISP methodology is die-/package-independent.

There are two methods of in-system programming: external and internal.

- Programmer ISP—Refer to *In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3* for more information.

  Using an external programmer and a cable, the device can be programmed through a header on the system board. In Actel documentation, this is referred to as external ISP. Actel provides FlashPro3, FlashPro Lite, FlashPro, or Silicon Sculptor 3 to perform external ISP. Note that Silicon Sculptor II and Silicon Sculptor 3 can only provide ISP for ProASIC and ProASIC^{PLUS}® families, not for Fusion, IGLOO, or ProASIC3.

  - Advantages: Allows local control of programming and data files for maximum security. The programming algorithms and hardware are available from Actel. The only hardware required on the board is a programming header.
  - Limitations: A negligible board space requirement for the programming header and JTAG signal routing
- Microprocessor ISP—Refer to *Microprocessor Programming of Actel's Low-Power Flash Devices* for more information.

  Using a microprocessor and an external or internal memory, you can store the program in memory and use the microprocessor to perform the programming. In Actel documentation, this is referred to as internal ISP. Both the code for the programming algorithm and the FPGA programming file must be stored in memory on the board. Programming voltages must also be generated on the board.

  - Advantages: The programming code is stored in the system memory. An external programmer is not required during programming.
  - Limitations: This is the approach that requires the most design work, since some way of getting and/or storing the data is needed; a system interface to the device must be designed; and the low-level API to the programming firmware must be written and linked into the code provided by Actel. While there are benefits to this methodology, serious thought and planning should go into the decision.

Device Programmers

Device Type Supported: Flash and Antifuse

Device programmers are used to program a device before it is mounted on the system board.

The advantage of using device programmers is that no programming hardware is required on the system board. Therefore, no additional components or board space are required.

If devices are to be reprogrammed multiple times, or if the quantity of devices to be programmed is relatively low, a single-site device programmer is the simplest solution. For applications in which design security is paramount (often the case in military or space designs), the use of on-site programming maintains design security at all times.

Adapter modules are purchased with the programmers to support the FPGA packages used. The FPGA is placed in the adapter module and the programming software is run from a PC. Actel supplies the programming software for all of the Actel programmers. The software allows for the selection of the correct die/package and programming files. It will then program and verify the device.

- Single-site programmers

  A single-site programmer programs one device at a time. Actel offers Silicon Sculptor 3 as a single-site programmer.

  – Advantages: Lower cost than multi-site programmers. No additional overhead for programming on the system board. Allows local control of programming and data files for maximum security. Allows on-demand programming on-site.

  – Limitations: Only programs one device at a time.

- Multi-site programmers

  Often referred to as batch or gang programmers, multi-site programmers can program multiple devices at the same time using the same programming file. This is often used for large volume programming and by programming houses. The sites often have independent processors and memory enabling the sites to operate concurrently, meaning each site may start programming the same file independently. This enables the operator to change one device while the other sites continue programming, which increases throughput. Multiple adapter modules for the same package are required when using a multi-site programmer. Silicon Sculptor I, II, and 3 programmers can be cascaded to program multiple devices in a chain. Multi-site programmers can also be purchased from BP Microsystems.

  – Advantages: Provides the capability of programming multiple devices at the same time. No additional overhead for programming on the system board. Allows local control of programming and data files for maximum security.

  – Limitations: More expensive than a single-site programmer

- Automated production (robotic) programmers

  Automated production programmers are based on multi-site programmers. They consist of a large input tray holding multiple parts and a robotic arm to select and place parts into appropriate programming sockets automatically. When the programming of the parts is complete, the parts are removed and placed in a finished tray. The automated programmers are often used in volume programming houses to program parts for which the programming time is small.

Volume Programming Services

Device Type Supported: Flash and Antifuse

Once the design is stable for applications with large production volumes, preprogrammed devices can be purchased. Table 16-3 describes the volume programming services.

Table 16-3 • **Volume Programming Services**

Programmer	Vendor	Availability
In-House Programming	Actel	Contact Actel Sales
Distributor Programming Centers	Memec Unique	Contact Distribution
Independent Programming Centers	Various	Contact Vendor

Advantages: As programming is outsourced, this solution is easier to implement than creating a substantial in-house programming capability. As programming houses specialize in large-volume programming, this is often the most cost-effective solution.

Limitations: There are some logistical issues with the use of a programming service provider, such as the transfer of programming files and the approval of First Articles. By definition, the programming file must be released to a third-party programming house. Nondisclosure agreements (NDAs) can be signed to help ensure data protection; however, for extremely security-conscious designs, this may not be an option.

- Actel In-House Programming

  When purchasing Actel devices in volume, IHP can be requested as part of the purchase. If this option is chosen, there is a small cost adder for each device programmed. Each device is marked with a special mark to distinguish it from blank parts. Programming files for the design will be sent to Actel. Sample parts with the design programmed, First Articles, will be returned for customer approval. Once approval of First Articles has been received, Actel will proceed with programming the remainder of the order. To request Actel IHP, contact your local Actel representative.

- Distributor Programming Centers

  If purchases are made through a distributor, many distributors will provide programming for their customers. Consult with your preferred distributor about this option.

- Independent Programming Centers

  There are many programming centers that specialize only in programming but are not directly affiliated with Actel or our distributors. These programming centers must follow the guidelines for programming Actel devices and use certified programmers to program Actel devices. Actel does not have recommendations for external programming centers.

Programming Solutions

Details for the available programmers can be found in the programmer user's guides listed in the "Related Documents" section on page 16-15. Refer to Table 16-6 on page 16-10 for more information concerning programming solutions.

All of the programmers except FlashPro3, FlashPro Lite, and FlashPro require adapter modules, which are designed to support device packages. The modules are all listed on the Actel website at http://www.actel.com/products/hardware/program_debug/ss/modules.aspx. They are not listed in this document, since this list is updated frequently with new package options and any upgrades required to improve programming yield or support new families.

Table 16-4 • **Programming Solutions**

Programmer	Vendor	ISP	Single Device	Multi-Device	Availability
FlashPro3	Actel	Only	Yes	Yes[1]	Available
FlashPro Lite	Actel	Only	Yes	Yes[1]	Available
FlashPro	Actel	Only	Yes	Yes[1]	Available
Silicon Sculptor 3	Actel	Yes[2]	Yes	Cascade option (up to two)	Available
Silicon Sculptor II	Actel	Yes[2]	Yes	Cascade option (up to two)	Available
Silicon Sculptor	Actel	Yes	Yes	Cascade option (up to four)	Discontinued
Sculptor 6X	Actel	No	Yes	Yes	Discontinued
BP MicroProgrammers	BP Microsystems	No	Yes	Yes	Contact BP Microsystems at www.bpmicro.com

Notes:

1. *Multiple devices can be connected in the same JTAG chain for programming.*
2. *Silicon Sculptor II and Silicon Sculptor 3 can only provide ISP for ProASIC and ProASIC^{PLUS} families, not for Fusion, IGLOO, or ProASIC3 devices.*

Programmer Ordering Codes

The products shown in Table 16-5 can be ordered through Actel sales and will be shipped directly from Actel. Products can also be ordered from Actel distributors, but will still be shipped directly from Actel. Table 16-5 includes ordering codes for the full kit, as well as codes for replacement items and any related hardware. Some additional products can be purchased from external suppliers for use with the programmers. Ordering codes for adapter modules used with Silicon Sculptor are available on the Actel website at http://www.actel.com/products/hardware/program_debug/ss/modules.aspx.

Table 16-5 • **Programming Ordering Codes**

Description	Vendor	Ordering Code	Comment
FlashPro3 ISP programmer	Actel	FLASHPRO 3	Uses a 2×5, RA male header connector
FlashPro Lite ISP programmer	Actel	FLASHPRO LITE	Supports small programming header or large header through header converter (not included)
FlashPro ISP programmer	Actel	FLASH PRO	Supports small programming header or large header through header converter (not included)
Silicon Sculptor 3	Actel	SILICON-SCULPTOR 3	USB 2.0 high-speed production programmer
Silicon Sculptor II	Actel	SILICON-SCULPTOR II	Requires add-on adapter modules to support devices
Silicon Sculptor ISP module	Actel	SMPA-ISP-ACTEL-3-KIT	Ships with both large and small header support

** A maximum of two Silicon Sculptor II programmers can be chained together using a standard IEEE 1284 parallel port cable.*

Table 16-5 • **Programming Ordering Codes (continued)**

Description	Vendor	Ordering Code	Comment
Concurrent programming cable	Actel	SS-EXPANDER	Used to cascade Silicon Sculptor I programmers together*
Software for Silicon Sculptor	Actel	SCULPTOR-SOFTWARE-CD	http://www.actel.com/download/program_debug/ss/
ISP cable for small header	Actel	ISP-CABLE-S	Supplied with SMPA-ISP-ACTEL-3-KIT
ISP cable for large header	Actel	PA-ISP-CABLE	Supplied with SMPA-ISP-ACTEL-3-KIT
Header converter	Actel	Header-Converter	Converts from small to large header
Small programming header	Samtec	FTSH-113-01-L-D-K	Supported by FlashPro, FlashPro Lite, and Silicon Sculptor In migrating to ProASIC3/E devices, an FP3-26PIN-ADAPTER is required.
10-pin 0.1" pitch cable header (right-angle PCB mount angle)	AMP	103310-1	Supported by FlashPro3
10-pin 0.1" pitch cable header (straight PCB mount angle)	3M	2510-6002UB	Supported by FlashPro3
Compact programming header (10-pin 0.05" pitch, 2 rows of 5 pins)	Samtec	FTSH-105-01-L-D-K	Supported by FlashPro3, FP3-10PIN-ADAPTER-KIT required. Used for boards where space is at a premium.
Migration and compact header adapter	Actel	FP3-10PIN-ADAPTER-KIT	Compact header and migration kit. Comes with Samtec FFSD-05-D-06.00-01-N (6-inch cable for connecting to compact 10-pin headers).
Large programming header, 0.062" board thickness	3M	3429-6502	Supported by Silicon Sculptor by default, FlashPro and FlashPro Lite with header converter
Large programming header, 0.094" to 0.125" board thickness	3M	3429-6503	Supported by Silicon Sculptor by default, FlashPro and FlashPro Lite with header converter
Plug-in header, small	Actel	SMPA-ISP-HEADER-S	Required for small header for ProASIC only; not used for ProASIC^{PLUS}
Plug-in header	Actel	SMPA-ISP-HEADER	Required for large header for ProASIC only; not used for ProASIC^{PLUS}
Vacuum pens for PQ, TQ, VQ; <208 pins	Actel	PENVAC	
Vacuum pens for PQ, TQ, VQ; ≥208 pins	Actel	PENVAC-HD	Heavy-duty, provides stronger vacuum

* *A maximum of two Silicon Sculptor II programmers can be chained together using a standard IEEE 1284 parallel port cable.*

Programmer Device Support

Refer to Table 16-6 to determine which general-purpose flash devices have programmer device support. To learn more about the different Actel families, refer to the Actel website: http://www.actel.com/products/devices.aspx.

Data in Table 16-6 also applies to ARM-enabled M7 device versions of Fusion, IGLOO, and ProASIC3 devices. Refer to the appropriate family datasheets for information on die/package combinations available as ARM-enabled versions.

Table 16-6 • **Programmer Device Support**

Actel Family	Device	ARM-Enabled	Silicon Sculptor	Silicon Sculptor 6X	Silicon Sculptor II	Silicon Sculptor 3	FlashPro	FlashPro Lite	FlashPro3
Fusion	AFS090		No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support
	AFS250								
	AFS600	✓							
	AFS1500	✓							
IGLOO	AGL015		No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support.
	AGL030								
	AGL060								
	AGL125								
	AGL250	✓							
	AGL400	✓							
	AGL600	✓							
	AGL1000	✓							
IGLOOe	AGLE600	✓	No	No	Yes. No ISP support	Yes. No ISP support.	No	No	Yes. ISP support
	AGLE3000	✓							
IGLOO PLUS	AGLP030		No	No	Yes. No ISP support	Yes. No ISP support.	No	No	Yes. ISP support
	AGLP060								
	AGLP125								
IGLOO nano	AGLN010		No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support
	AGLN015								
	AGLN020								
	AGLN060								
	AGLN125								
	AGLN250								
ProASIC3 nano	A3PN010		No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support.
	A3PN015								
	A3PN020								
	A3PN060								
	A3PN125								
	A3PN250								
ProASIC3L	A3P250L	✓	No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support.
	A3P600L	✓							
	A3P1000L	✓							
	A3PE3000L	✓							

** Refer to the "Certified Programming Solutions" section on page 16-12 for more information on programmer support.*

Table 16-6 • **Programmer Device Support (continued)**

Actel Family	Device	ARM-Enabled	Silicon Sculptor	Silicon Sculptor 6X	Silicon Sculptor II	Silicon Sculptor 3	FlashPro	FlashPro Lite	FlashPro3
ProASIC3	A3P015		No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support.
	A3P030								
	A3P060								
	A3P125								
	A3P250	✓							
	A3P400	✓							
	A3P600	✓							
	A3P1000	✓							
ProASIC3E	A3PE600	✓	No	No	Yes. No ISP support.	Yes. No ISP support.	No	No	Yes. ISP support.
	A3PE1500	✓							
	A3PE3000	✓							
ProASIC^{PLUS}	APA075		Yes. ISP support.	Yes. ISP support.	Yes. ISP support.	Yes. ISP support.	Yes. ISP support.	Yes. ISP support.	No
	APA150								
	APA300								
	APA450								
	APA600								
	APA750								
	APA1000								
ProASIC	A500K50		Yes	Yes	Yes	Yes	Yes	No	No
	A500K130								
	A500K180								
	A500K270								

* *Refer to the "Certified Programming Solutions" section on page 16-12 for more information on programmer support.*

Certified Programming Solutions

The Actel-certified programmers for flash devices are FlashPro3, FlashPro Lite, FlashPro, Silicon Sculptor I and II, and any programmer that is built by BP Microsystems. All other programmers are considered noncertified programmers.

- **FlashPro3, FlashPro Lite, FlashPro**

  The Actel family of FlashPro device programmers provides in-system programming in an easy-to-use, compact system that supports all flash families. Whether programming a board containing a single device or multiple devices connected in a chain, the Actel line of FlashPro programmers enables fast programming and reprogramming. Programming with the FlashPro series of programmers saves board space and money as it eliminates the need for sockets on the board. There are no built-in algorithms, so there is no delay between product release and programming support.

- **Silicon Sculptor II**

  Silicon Sculptor II is a robust, compact, single-device programmer with standalone software for the PC. It is designed to enable concurrent programming of multiple units from the same PC with speeds equivalent to or faster than previous Actel programmers. It replaces Silicon Sculptor I as the Actel programmer of choice.

- **Silicon Sculptor I and Silicon Sculptor 6X**

  Actel no longer offers Silicon Sculptor I or Silicon Sculptor 6X for sale. Both items have been discontinued. Actel does support Silicon Sculptor I and Silicon Sculptor 6X by continuing to release new software that enables improved programming of previously covered Actel devices; new Actel devices are only supported on Silicon Sculptor II. All software support for Silicon Sculptor I and Silicon Sculptor 6X programmers will be disconnected by the end of 2005; no support for these older programmers will be offered in 2006. Actel recommends that all customers upgrade to Silicon Sculptor II or a BP multi-site programmer.

- **Noncertified Programmers**

  Actel does not test programming solutions from other vendors, and CANNOT guarantee programming yield. Also, Actel will not perform any failure analysis on devices programmed by hardware from other vendors.

- **Programming Centers**

  Actel programming hardware policy also applies to programming centers. Actel expects all programming centers to use certified programmers to program Actel devices. If a programming center uses noncertified programmers to program Actel devices, the "Noncertified Programmers" policy applies.

Flash Programming Guidelines

Preprogramming Setup

Before programming, several steps are required to ensure an optimal programming yield.

Use Proper Handling and Electrostatic Discharge (ESD) Precautions

Actel FPGAs are sensitive electronic devices that are susceptible to damage from ESD and other types of mishandling. For more information about ESD, refer to the *Actel Quality and Reliability Guide,* beginning with page 41.

Use the Latest Version of the Designer Software to Generate Your Programming File (recommended)

The files used to program Actel flash devices (*.bit, *.stp) contain important information about the switches that will be programmed in the FPGA. Find the latest version and corresponding release notes at http://www.actel.com/download/software/designer/. Also, programming files must always be zipped during file transfer to avoid the possibility of file corruption.

Use the Latest Version of the Programming Software

The programming software is frequently updated to accommodate yield enhancements in FPGA manufacturing. These updates ensure maximum programming yield and minimum programming times. Before programming, always check the version of software being used to ensure it is the most recent. Depending on the programming software, refer to one of the following:

- FlashPro: http://www.actel.com/download/program_debug/flashpro/
- Silicon Sculptor: http://www.actel.com/download/program_debug/ss/

Use the Most Recent Adapter Module with Silicon Sculptor

Occasionally, Actel makes modifications to the adapter modules to improve programming yields and programming times. To identify the latest version of each module before programming, visit http://www.actel.com/products/hardware/program_debug/ss/modules.aspx.

Perform Routine Hardware Self-Diagnostic Test

- FlashPro

  The self-test is only applicable when programming with FlashPro and FlashPro3 programmers. It is not supported with FlashPro Lite. To run the self-diagnostic test, follow the instructions given in the "Performing a Self-Test" section of http://www.actel.com/documents/FlashPro_UG.pdf.

- Silicon Sculptor

  The self-diagnostic test verifies correct operation of the pin drivers, power supply, CPU, memory, and adapter module. This test should be performed before every programming session. At minimum, the test must be executed every week. To perform self-diagnostic testing using the Silicon Sculptor software, perform the following steps, depending on the operating system:

  – DOS: From anywhere in the software, type **ALT + D**.

  – Windows: Click **Device** > choose **Actel Diagnostic** > select the **Test** tab > click **OK**.

Programming Flash FPGAs

Programming a flash device is a one-step process, whether programming is conducted with a socket adapter module or via ISP. The Execute function will automatically erase the device, program the flash cells, and verify that it is programmed correctly. Actel recommends confirming the security status is correct before programming.

The following steps are required to program Actel flash FPGAs.

Programming with FlashPro

Setup

Properly connect the FlashPro ribbon cable with the programming header and turn on the switch. Actel recommends running the self-test before programming any devices; see the "Perform Routine Hardware Self-Diagnostic Test" section on page 16-13.

In the programming software, from the File menu, choose **Connect**. In the FlashPro Connect to Programmer dialog box that appears, select the port to which the FlashPro programmer is connected, and select the device family. Disable voltages from the programmer if they are available on the board.

Click **Connect**. A successful connect or any errors appear in the Log window.

Analyze Chain and Device Selection

From the File menu, choose **Analyze Chain**. Chain details appear in the Log window. If any failures appear, refer to the error and troubleshooting section of the *FlashPro User's Guide*. Select the device to be programmed from the **Device** list. If only one device is present in the chain, performing Analyze Chain selects that device automatically from the Device list.

Loading the STAPL file

FlashPro3, FlashPro Lite, and FlashPro programmers use a STAPL (*.stp) file to program the device. To load the STAPL file, from the File menu, choose **Open STAPL file**, or click the **Open File** button on the toolbar.

Selecting an Action

After loading the STAPL file, select an action from the Action list. See the "Programming File Actions" section in the *FlashPro User's Guide* for a definition of each action.

Programming the Device

To program the device, in the Action list, select **Program**. Make the required selections and click **Execute** to start programming. The progress of the programming action displays in the Log window. The message "Exit 0" indicates that the device has successfully been programmed.

Note: Do not interrupt the programming sequence; it may damage the device or programmer.

Verify Correct Programming

To verify the device is programmed with the correct STAPL file, load the STAPL file and in the Action list, click **Verify**. Click **Execute** to start the verification process. A successful verification results in "Exit 0."

Note: Verification is also performed in the previous "Programming the Device" step; clicking **Verify** is an additional standalone option.

Programming Failure Allowances

Flash FPGAs are reprogrammable, so Actel tests the programmability for 100% of the devices shipped.

Return Material Authorization (RMA) Policies

Actel consistently strives to exceed customer expectations by continuing to improve the quality of our products and our quality management system. Actel has RMA procedures in place to address programming fallout. Customers should be mindful of the following RMA policies.

All devices submitted for an RMA must be within the Actel warranty period of one year from date of shipment. Actel will reject RMAs for devices that are no longer under warranty.

RMAs will only be authorized for current Actel devices. Devices that have been discontinued will not receive RMAs. All functional failure analysis requests must be initiated by opening a case with Actel Technical Support. Devices returned for failure analysis against an RMA should be in their original packaging and must have an RMA number issued by Actel.

Contacting the Customer Support Group

Highly skilled engineers staff the Customer Applications Center from 7:00 A.M. to 6:00 P.M., Pacific time, Monday through Friday. You can contact the center by one of the following methods:

Electronic Mail

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. Actel monitors the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and contact information for efficient processing of your request. The technical support email address is tech@actel.com.

Telephone

Our Technical Support Hotline answers all calls. The center retrieves information, such as your name, company name, telephone number, and question. Once this is done, a case number is assigned. Then the center forwards the information to a queue where the first available applications engineer receives the data and returns your call. The phone hours are from 7:00 A.M. to 6:00 P.M., Pacific time, Monday through Friday.

The Customer Applications Center number is (800) 262-1060.

European customers can call +44 (0) 1256 305 600.

Related Documents

Below is a list of related documents, their location on the Actel website, and a brief summary of each document.

Application Notes

Programming Antifuse Devices

http://www.actel.com/documents/AntifuseProgram_AN.pdf

Implementation of Security in Actel's ProASIC and ProASIC^{PLUS} Flash-Based FPGAs

http://www.actel.com/documents/Flash_Security_AN.pdf

Handbook Documents

Security in Low-Power Flash Devices

http://www.actel.com/documents/LPD_Security_HBs.pdf

In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3

http://www.actel.com/documents/LPD_ISP_HBs.pdf

Microprocessor Programming of Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_Microprocessor_HBs.pdf

User's Guides

FlashPro Programmers

FlashPro3, FlashPro Lite, and FlashPro

http://www.actel.com/products/hardware/program_debug/flashpro/default.aspx

FlashPro User's Guide

http://www.actel.com/documents/FlashPro_UG.pdf

The FlashPro User's Guide includes hardware and software setup, self-test instructions, use instructions, and a troubleshooting / error message guide.

Silicon Sculptor 3 and Silicon Sculptor II

http://www.actel.com/products/hardware/program_debug/ss/default.aspx

Other Documents

http://www.actel.com/products/solutions/security/default.aspx#flashlock

The security resource center describes security in Actel Flash FPGAs.

Actel Quality and Reliability Guide

http://www.actel.com/documents/RelGuide.pdf

Part Number and Revision Date

This document was previously published as an application note describing features and functions of the device, and as such has now been incorporated into the device handbook format. No technical changes have been made to the content.

Part Number 51700094-013-3

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.3)	Page
v1.2 (October 2008)	The "Programming Support in Flash Devices" section was updated to include IGLOO nano and ProASIC3 nano devices.	16-2
	The "Flash Devices" section was updated to include information for IGLOO nano devices. The following sentence was added: IGLOO PLUS devices can also be operated at any voltage between 1.2 V and 1.5 V; the Designer software allows 50 mV increments in the voltage.	16-4
	Table 16-5 · Programming Ordering Codes was updated to replace FP3-26PIN-ADAPTER with FP3-10PIN-ADAPTER-KIT.	16-8
	Table 16-6 · Programmer Device Support was updated to add IGLOO nano and ProASIC3 nano devices. AGL400 was added to the IGLOO portion of the table.	16-10
v1.1 (March 2008)	The "Programming Support in Flash Devices" section was revised to include new families and make the information more concise.	16-2
	Figure 16-1 · FlashPro Programming Setup and the "Programming Support in Flash Devices" section are new.	16-1, 16-2
	Table 16-6 · Programmer Device Support was updated to include A3PE600L with the other ProASIC3L devices, and the RT ProASIC3 family was added.	16-10
v1.0 (January 2008)	The "Flash Devices" section was updated to include the IGLOO PLUS family. The text, "Voltage switching is required in-system to switch from a 1.2 V core to 1.5 V core for programming," was revised to state, "Although the device can operate at 1.2 V core voltage, the device can only be reprogrammed when the core voltage is 1.5 V. Voltage switching is required in-system to switch from a 1.2 V supply (V_{CC}, V_{CCI}, and V_{JTAG}) to 1.5 V for programming."	16-4
	The ProASIC3L family was added to Table 16-6 · Programmer Device Support as a separate set of rows rather than combined with ProASIC3 and ProASIC3E devices. The IGLOO PLUS family was included, and AGL015 and A3P015 were added.	16-10

17 – Security in Low-Power Flash Devices

Security in Programmable Logic

The need for security on FPGA programmable logic devices (PLDs) has never been greater than today. If the contents of the FPGA can be read by an external source, the intellectual property (IP) of the system is vulnerable to unauthorized copying. Actel Fusion,® IGLOO,® and ProASIC®3 devices contain state-of-the-art circuitry to make the flash-based devices secure during and after programming. Low-power flash devices have a built-in 128-bit Advanced Encryption Standard (AES) decryption core (except for 30 k gate devices and smaller). The decryption core facilitates secure in-system programming (ISP) of the FPGA core array fabric, the FlashROM, and the Flash Memory Blocks (FBs) in Fusion devices. The FlashROM, Flash Blocks, and FPGA core fabric can be programmed independently of each other, allowing the FlashROM or Flash Blocks to be updated without the need for change to the FPGA core fabric.

Actel has incorporated the AES decryption core into the low-power flash devices and has also included the Actel flash-based lock technology, FlashLock.® Together, they provide leading-edge security in a programmable logic device. Configuration data loaded into a device can be decrypted prior to being written to the FPGA core using the AES 128-bit block cipher standard. The AES encryption key is stored in on-chip, nonvolatile flash memory.

This document outlines the security features offered in low-power flash devices, some applications and uses, as well as the different software settings for each application.

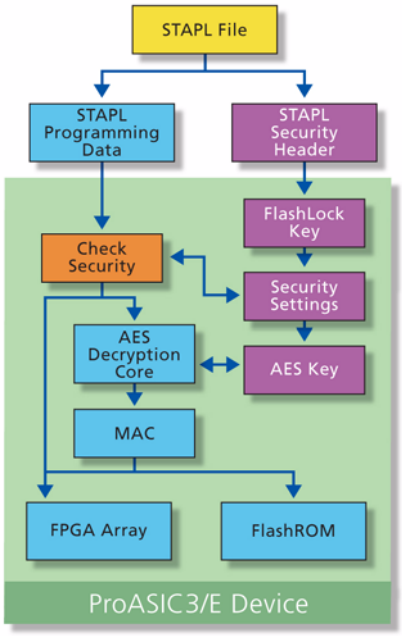


Figure 17-1 • **Overview on Security**

Security Support in Flash-Based Devices

The flash FPGAs listed in Table 17-1 support the security feature and the functions described in this document.

Table 17-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 17-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

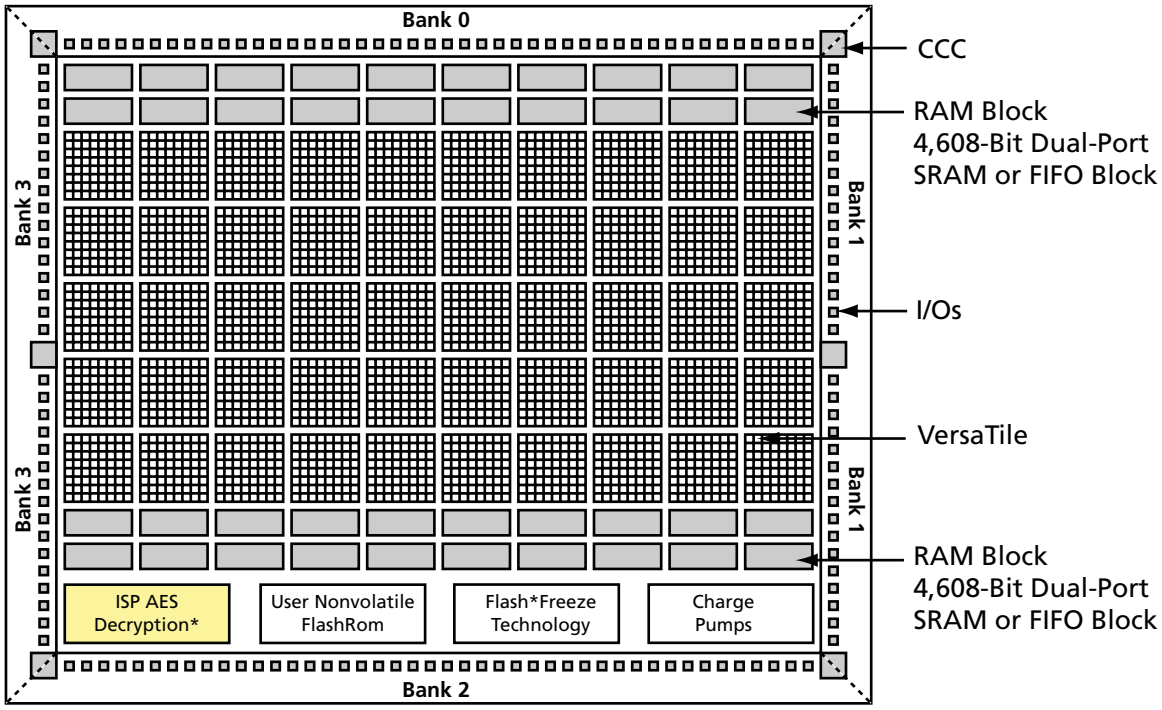
ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 17-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Security Architecture

Fusion, IGLOO, and ProASIC3 devices have been designed with the most comprehensive programming logic design security in the industry. In the architecture of these devices, security has been designed into the very fabric. The flash cells are located beneath seven metal layers, and the use of many device design and layout techniques makes invasive attacks difficult. Since device layers cannot be removed without disturbing the charge on the programmed (or erased) flash gates, devices cannot be easily deconstructed to decode the design. Low-power flash devices are unique in being reprogrammable and having inherent resistance to both invasive and noninvasive attacks on valuable IP. Secure, remote ISP is now possible with AES encryption capability for the programming file during electronic transfer. Figure 17-2 shows a view of the AES decryption core inside an IGLOO device; Figure 17-3 on page 17-4 shows the AES decryption core inside a Fusion device. The AES core is used to decrypt the encrypted programming file when programming.



*Note: *ISP AES Decryption is not supported by 30 k gate devices and smaller. For details of other architecture features by device, refer to the appropriate family datasheet.*

***Figure 17-2* • Block Representation of the AES Decryption Core in IGLOO and ProASIC3 Devices**

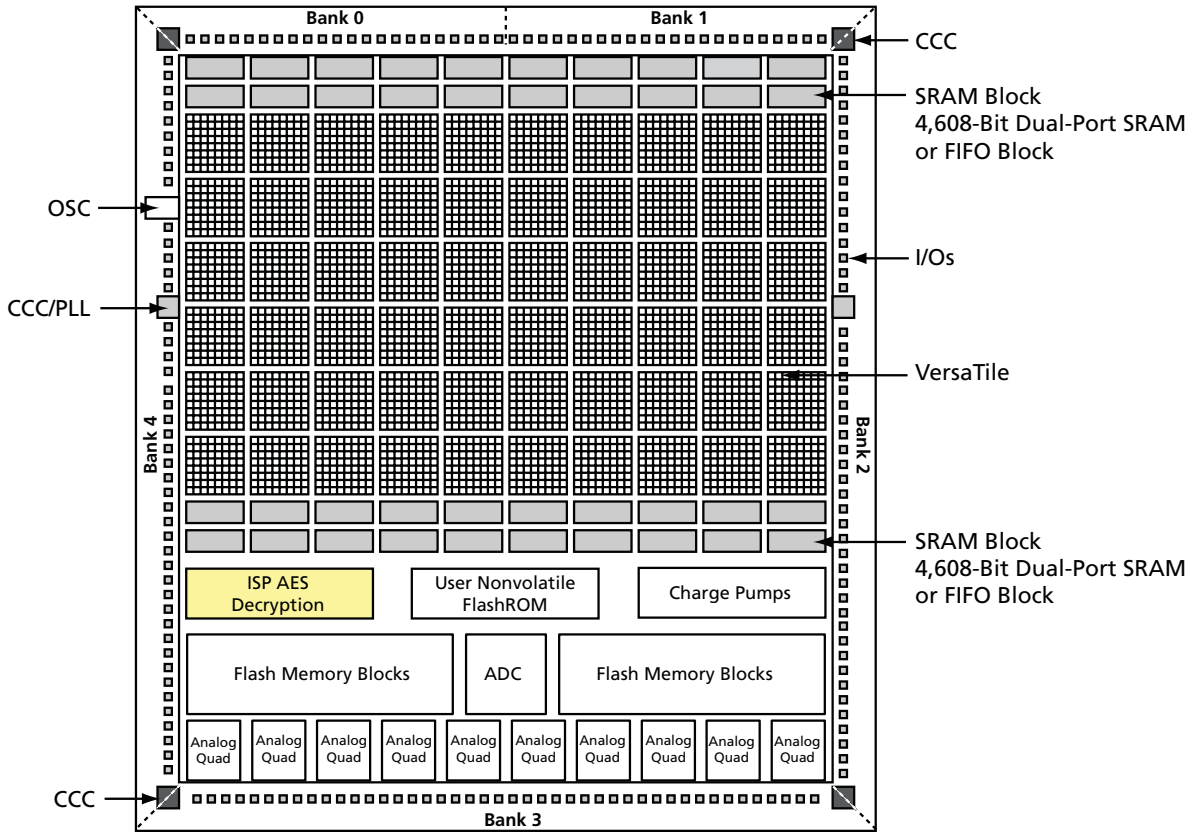


Figure 17-3 • **Block Representation of the AES Decryption Core in a Fusion AFS600 FPGA**

Security Features

IGLOO and ProASIC3 devices have two entities inside: FlashROM and the FPGA core fabric. Fusion devices contain three entities: FlashROM, FBs, and the FPGA core fabric. The parts can be programmed or updated independently with a STAPL programming file. The programming files can be AES-encrypted or plaintext. This allows maximum flexibility in providing security to the entire device. Refer to *FlashROM in Actel's Low-Power Flash Devices* for information on the FlashROM structure.

Unlike SRAM-based FPGA devices, which require a separate boot PROM to store programming data, low-power flash devices are nonvolatile, and the secured configuration data is stored in on-chip flash cells that are part of the FPGA fabric. Once programmed, this data is an inherent part of the FPGA array and does not need to be loaded at system power-up. SRAM-based FPGAs load the configuration bitstream upon power-up; therefore, the configuration is exposed and can be read easily.

The built-in FPGA core, FBs, and FlashROM support programming files encrypted with the 128-bit AES (FIPS-192) block ciphers. The AES key is stored in dedicated, on-chip flash memory and can be programmed before the device is shipped to other parties (allowing secure remote field updates).

Security in ARM-Enabled Low-Power Flash Devices

There are slight differences between the regular flash devices and the ARM®-enabled flash devices, which have the M1 and M7 prefix.

The AES key is used by Actel and preprogrammed into the device to protect the ARM IP. As a result, the design is encrypted along with the ARM IP, according to the details below.

Cortex-M1 Device Security

Cortex-M1–enabled devices are shipped with the following security features:

- FPGA array enabled for AES-encrypted programming and verification
- FlashROM enabled for AES-encrypted Write and Verify
- Fusion Embedded Flash Memory enabled for AES-encrypted Write

AES Encryption of Programming Files

Low-power flash devices employ AES as part of the security mechanism that prevents invasive and noninvasive attacks. The mechanism entails encrypting the programming file with AES encryption and then passing the programming file through the AES decryption core, which is embedded in the device. The file is decrypted there, and the device is successfully programmed. The AES master key is stored in on-chip nonvolatile memory (flash). The AES master key can be preloaded into parts in a secure programming environment (such as the Actel In-House Programming center), and then "blank" parts can be shipped to an untrusted programming or manufacturing center for final personalization with an AES-encrypted bitstream. Late-stage product changes or personalization can be implemented easily and securely by simply sending a STAPL file with AES-encrypted data. Secure remote field updates over public networks (such as the Internet) are possible by sending and programming a STAPL file with AES-encrypted data.

The AES key protects the programming data for file transfer into the device with 128-bit AES encryption. If AES encryption is used, the AES key is stored or preprogrammed into the device. To program, you must use an AES-encrypted file, and the encryption used on the file must match the encryption key already in the device.

The AES key is protected by a FlashLock security Pass Key that is also implemented in each device. The AES key is always protected by the FlashLock Key, and the AES-encrypted file does NOT contain the FlashLock Key. This FlashLock Pass Key technology is exclusive to the Actel flash-based device families. FlashLock Pass Key technology can also be implemented without the AES encryption option, providing a choice of different security levels.

In essence, security features can be categorized into the following three options:

- AES encryption with FlashLock Pass Key protection
- FlashLock protection only (no AES encryption)
- No protection

Each of the above options is explained in more detail in the following sections with application examples and software implementation options.

Advanced Encryption Standard

The 128-bit AES standard (FIPS-192) block cipher is the NIST (National Institute of Standards and Technology) replacement for DES (Data Encryption Standard FIPS46-2). AES has been designed to protect sensitive government information well into the 21st century. It replaces the aging DES, which NIST adopted in 1977 as a Federal Information Processing Standard used by federal agencies to protect sensitive, unclassified information. The 128-bit AES standard has 3.4×10^{38} possible 128-bit key variants, and it has been estimated that it would take 1,000 trillion years to crack 128-bit AES cipher text using exhaustive techniques. Keys are stored (securely) in low-power flash devices in nonvolatile flash memory. All programming files sent to the device can be authenticated by the part prior to programming to ensure that bad programming data is not loaded into the part that may possibly damage it. All programming verification is performed on-chip, ensuring that the contents of low-power flash devices remain secure.

Actel has implemented the 128-bit AES (Rijndael) algorithm in low-power flash devices. With this key size, there are approximately 3.4×10^{38} possible 128-bit keys. DES has a 56-bit key size, which provides approximately 7.2×10^{16} possible keys. In their AES fact sheet, the National Institute of Standards and Technology uses the following hypothetical example to illustrate the theoretical security provided by AES. If one were to assume that a computing system existed that could recover a DES key in a second, it would take that same machine approximately 149 trillion years to crack a 128-bit AES key. NIST continues to make their point by stating the universe is believed to be less than 20 billion years old.[1]

The AES key is securely stored on-chip in dedicated low-power flash device flash memory and cannot be read out. In the first step, the AES key is generated and programmed into the device (for example, at a secure or trusted programming site). The Actel Designer software tool provides AES key generation capability. After the key has been programmed into the device, the device will only correctly decrypt programming files that have been encrypted with the same key. If the individual programming file content is incorrect, a Message Authentication Control (MAC) mechanism inside the device will fail in authenticating the programming file. In other words, when an encrypted programming file is being loaded into a device that has a different programmed AES key, the MAC will prevent this incorrect data from being loaded, preventing possible device damage. See Figure 17-3 on page 17-4 and Figure 17-4 on page 17-7 for graphical representations of this process.

It is important to note that the user decides what level of protection will be implemented for the device. When AES protection is desired, the FlashLock Pass Key must be set. The AES key is a content protection mechanism, whereas the FlashLock Pass Key is a device protection mechanism. When the AES key is programmed into the device, the device still needs the Pass Key to protect the FPGA and FlashROM contents and the security settings, including the AES key. Using the FlashLock Pass Key prevents modification of the design contents by means of simply programming the device with a different AES key.

AES Decryption and MAC Authentication

Low-power flash devices have a built-in 128-bit AES decryption core, which decrypts the encrypted programming file and performs a MAC check that authenticates the file prior to programming.

MAC authenticates the entire programming data stream. After AES decryption, the MAC checks the data to make sure it is valid programming data for the device. This can be done while the device is still operating. If the MAC validates the file, the device will be erased and programmed. If the MAC fails to validate, then the device will continue to operate uninterrupted.

This will ensure the following:

- Correct decryption of the encrypted programming file
- Prevention of erroneous or corrupted data being programmed during the programming file transfer
- Correct bitstream passed to the device for decryption

1. *National Institute of Standards and Technology, "ADVANCED ENCRYPTION STANDARD (AES) Questions and Answers," 28 January 2002 (10 January 2005). See http://csrc.nist.gov/archive/aes/index1.html for more information.*

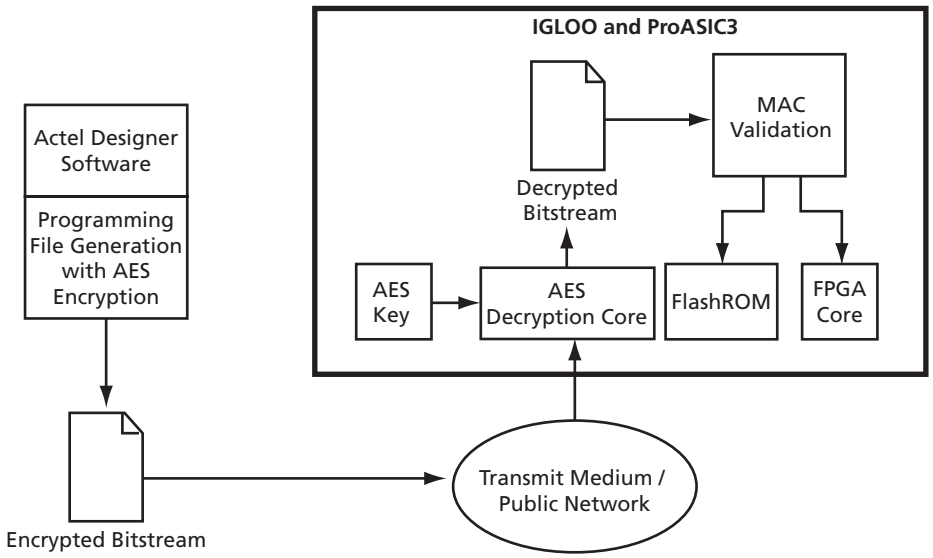


Figure 17-4 • **Example Application Scenario Using AES in IGLOO and ProASIC3 Devices**

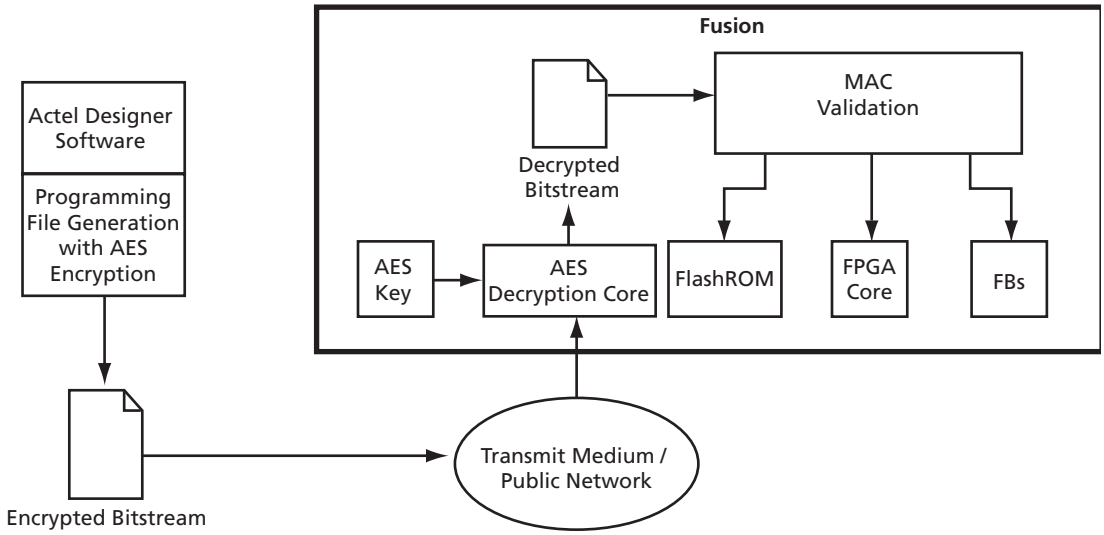


Figure 17-5 • **Example Application Scenario Using AES in Fusion Devices**

FlashLock

Additional Options for IGLOO and ProASIC3 Devices

The user also has the option of prohibiting Write operations to the FPGA array but allowing Verify operations on the FPGA array and/or Read operations on the FlashROM without the use of the FlashLock Pass Key. This option provides the user the freedom of verifying the FPGA array and/or reading the FlashROM contents after the device is programmed, without having to provide the FlashLock Pass Key. The user can incorporate AES encryption on the programming files to better enhance the level of security used.

Permanent Security Setting Options

In applications where a permanent lock is not desired, yet the security settings should not be modifiable, IGLOO and ProASIC3 devices can accommodate this requirement.

This application is particularly useful in cases where a device is located at a remote location and must be reprogrammed with a design or data update. Refer to the "Application 3: Nontrusted Environment—Field Updates/Upgrades" section on page 17-10 for further discussion and examples of how this can be achieved.

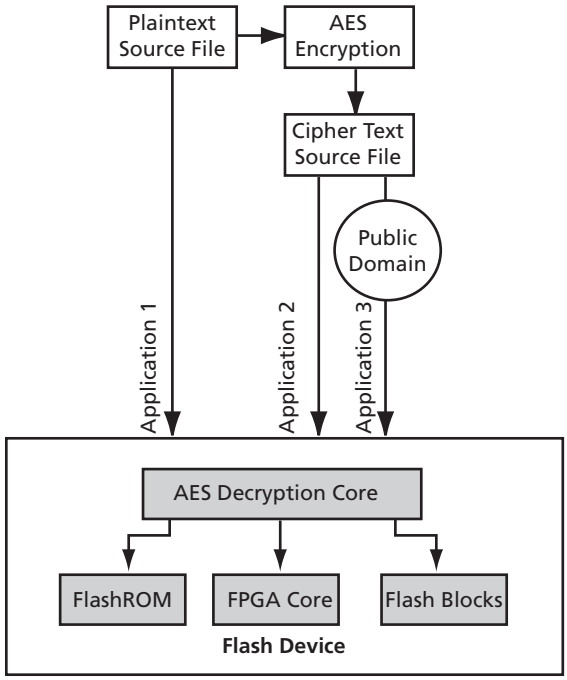
The user must be careful when considering the Permanent FlashLock or Permanent Security Settings option. Once the design is programmed with the permanent settings, it is not possible to reconfigure the security settings already employed on the device. Therefore, exercise careful consideration before programming permanent settings.

Permanent FlashLock

The purpose of the permanent lock feature is to provide the benefits of the highest level of security to IGLOO and ProASIC3 devices. If selected, the permanent FlashLock feature will create a permanent barrier, preventing any access to the contents of the device. This is achieved by permanently disabling Write and Verify access to the array, and Write and Read access to the FlashROM. After permanently locking the device, it has been effectively rendered one-time-programmable. This feature is useful if the intended applications do not require design or system updates to the device.

Security in Action

This section illustrates some applications of the security advantages of Actel's devices (Figure 17-6).



Note: Flash blocks are only used in Fusion devices.

Figure 17-6 • **Security Options**

Application 1: Trusted Environment

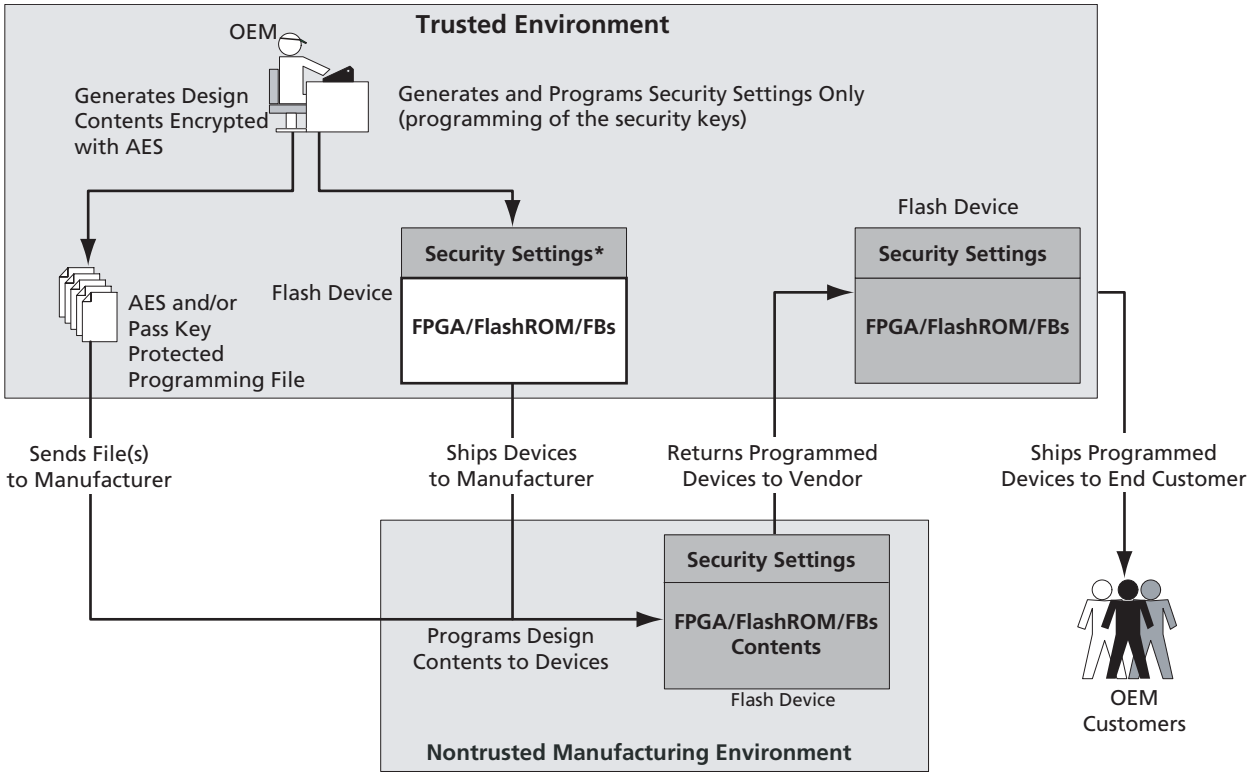
As illustrated in Figure 17-7 on page 17-10, this application allows the programming of devices at design locations where research and development take place. Therefore, encryption is not necessary and is optional to the user. This is often a secure way to protect the design, since the design program files are not sent elsewhere. In situations where production programming is not available at the design location, programming centers (such as Actel In-House Programming) provide a way of programming designs at an alternative, secure, and trusted location. In this scenario, the user generates a STAPL programming file from the Designer software in plaintext format, containing information on the entire design or the portion of the design to be programmed. The user can choose to employ the FlashLock Pass Key feature with the design. Once the design is programmed to unprogrammed devices, the design is protected by this FlashLock Pass Key. If no future programming is needed, the user can consider permanently securing the IGLOO and ProASIC3 device, as discussed in the "Permanent FlashLock" section on page 17-8.

Application 2: Nontrusted Environment—Unsecured Location

Often, programming of devices is not performed in the same location as actual design implementation, to reduce manufacturing cost. Overseas programming centers and contract manufacturers are examples of this scenario.

To achieve security in this case, the AES key and the FlashLock Pass Key can be initially programmed in-house (trusted environment). This is done by generating a programming file with only the security settings and no design contents. The design FPGA core, FlashROM, and (for Fusion) FB contents are generated in a separate programming file. This programming file must be set with the same AES key that was used to program to the device previously so the device will correctly decrypt this encrypted programming file. As a result, the encrypted design content programming file can

be safely sent off-site to nontrusted programming locations for design programming. Figure 17-7 shows a more detailed flow for this application.



Notes:

1. *Programmed portion indicated with dark gray.*
2. *Programming of FBs applies to Fusion only.*

Figure 17-7 • **Application 2: Device Programming in a Nontrusted Environment**

Application 3: Nontrusted Environment—Field Updates/Upgrades

Programming or reprogramming of devices may occur at remote locations. Reconfiguration of devices in consumer products/equipment through public networks is one example. Typically, the remote system is already programmed with particular design contents. When design update (FPGA array contents update) and/or data upgrade (FlashROM and/or FB contents upgrade) is necessary, an updated programming file with AES encryption can be generated, sent across public networks, and transmitted to the remote system. Reprogramming can then be done using this AES-encrypted programming file, providing easy and secure field upgrades. Low-power flash devices support this secure ISP using AES. The detailed flow for this application is shown in Figure 17-8 on page 17-11. Refer to *Microprocessor Programming of Actel's Low-Power Flash Devices* for more information.

To prepare devices for this scenario, the user can initially generate a programming file with the available security setting options. This programming file is programmed into the devices before shipment. During the programming file generation step, the user has the option of making the security settings permanent or not. In situations where no changes to the security settings are necessary, the user can select this feature in the software to generate the programming file with permanent security settings. Actel recommends that the programming file use encryption with an AES key, especially when ISP is done via public domain.

For example, if the designer wants to use an AES key for the FPGA array and the FlashROM, **Permanent** needs to be chosen for this setting. At first, the user chooses the options to use an AES key for the FPGA array and the FlashROM, and then chooses **Permanently lock the security settings**. A unique AES key is chosen. Once this programming file is generated and programmed to

the devices, the AES key is permanently stored in the on-chip memory, where it is secured safely. The devices are sent to distant locations for the intended application. When an update is needed, a new programming file must be generated. The programming file must use the same AES key for encryption; otherwise, the authentication will fail and the file will not be programmed in the device.

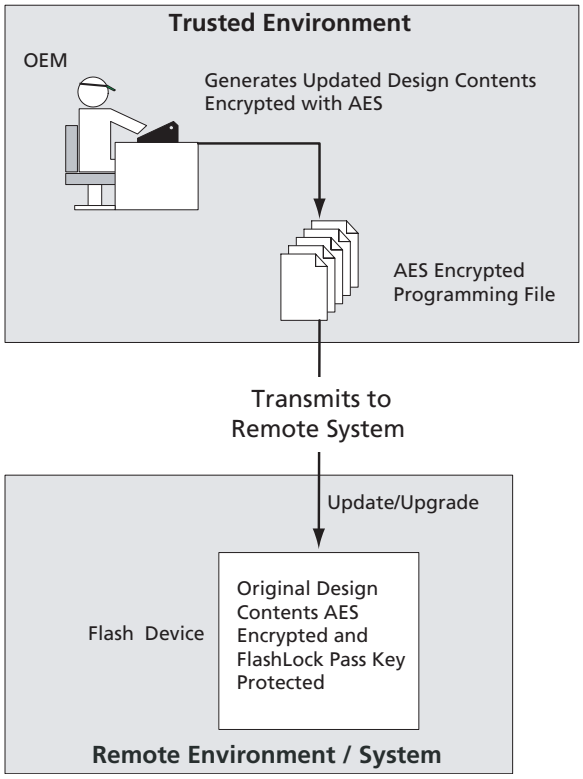


***Figure 17-8* • Application 3: Nontrusted Environment—Field Updates/Upgrades**

FlashROM Security Use Models

Each of the subsequent sections describes in detail the available selections in Actel Designer as an aid to understanding security applications and generating appropriate programming files for those applications. Before proceeding, it is helpful to review Figure 17-7 on page 17-10, which gives a general overview of the programming file generation flow within the Designer software as well as what occurs during the device programming stage. Specific settings are discussed in the following sections.

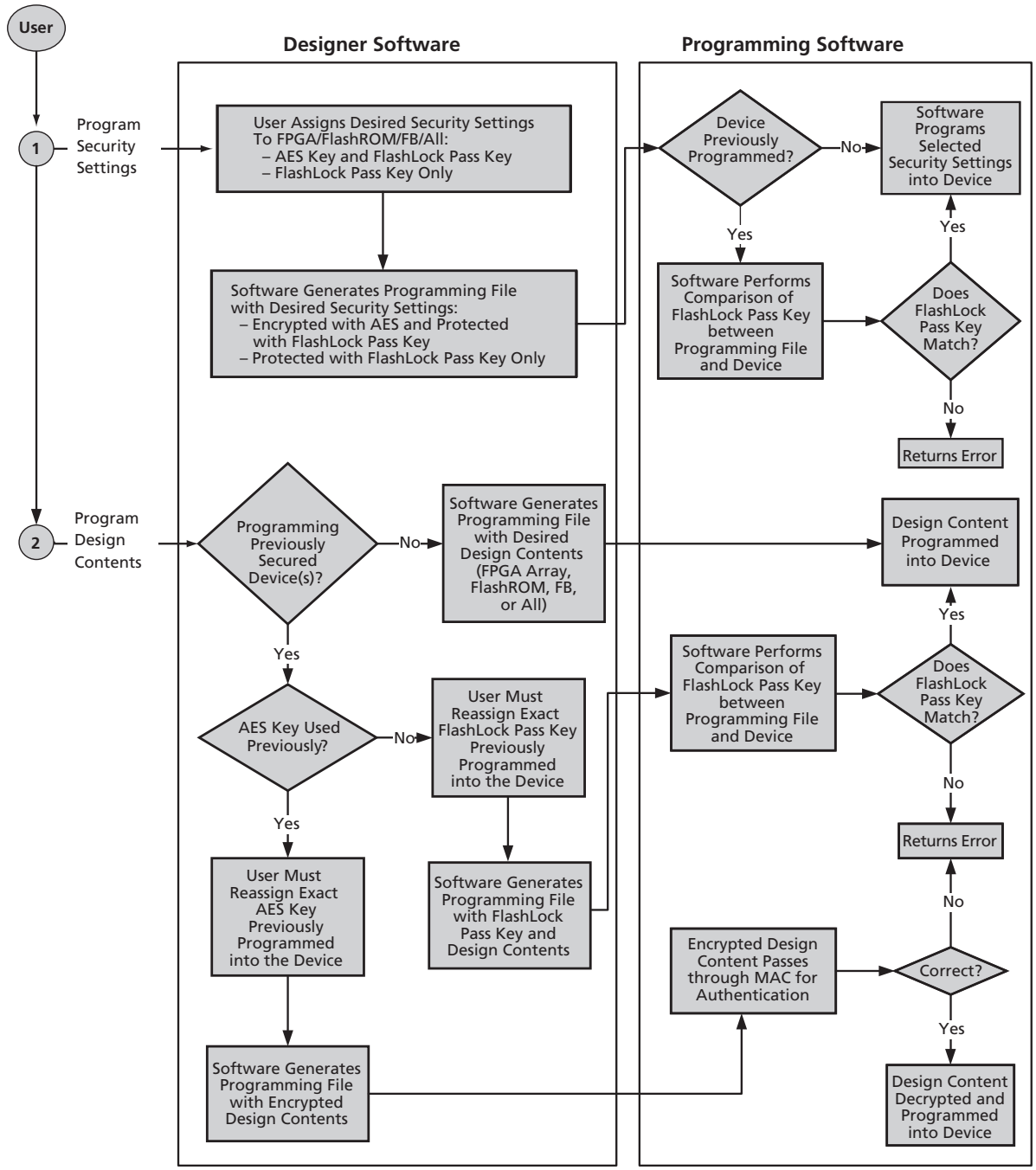
In Figure 17-7 on page 17-10, the flow consists of two sub-flows. Sub-flow 1 describes programming security settings to the device only, and sub-flow 2 describes programming the design contents only.

In Application 1, described in the "Application 1: Trusted Environment" section on page 17-9, the user does not need to generate separate files but can generate one programming file containing both security settings and design contents. Then programming of the security settings and design contents is done in one step. Both sub-flow 1 and sub-flow 2 are used.

In Application 2, described in the "Application 2: Nontrusted Environment—Unsecured Location" section on page 17-9, the trusted site should follow sub-flows 1 and 2 separately to generate two separate programming files. The programming file from sub-flow 1 will be used at the trusted site to program the device(s) first. The programming file from sub-flow 2 will be sent off-site for production programming.

In Application 3, described in the "Application 3: Nontrusted Environment—Field Updates/Upgrades" section on page 17-10, typically only sub-flow 2 will be used, because only updates to the design content portion are needed and no security settings need to be changed.

In the event that update of the security settings is necessary, see the "Reprogramming Devices" section on page 17-21 for details. For more information on programming low-power flash devices, refer to *In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3*.



Note: If programming the Security Header only, just perform sub-flow 1.
If programming design content only, just perform sub-flow 2.

Figure 17-9 • **Security Programming Flows**

Generating Programming Files

Generation of the Programming File in a Trusted Environment—Application 1

As discussed in the "Application 1: Trusted Environment" section on page 17-9, in a trusted environment, the user can choose to program the device with plaintext bitstream content. It is possible to use plaintext for programming even when the FlashLock Pass Key option has been selected. In this application, it is not necessary to employ AES encryption protection. For AES encryption settings, refer to the next sections.

The generated programming file will include the security setting (if selected) and the plaintext programming file content for the FPGA array, FlashROM, and/or FBs. These options are indicated in Table 17-2 and Table 17-3.

Table 17-2 • **IGLOO and ProASIC3 Plaintext Security Options, No AES**

Security Protection	FlashROM Only	FPGA Core Only	Both FlashROM and FPGA
No AES / no FlashLock	✓	✓	✓
FlashLock only	✓	✓	✓
AES and FlashLock	–	–	–

Table 17-3 • **Fusion Plaintext Security Options**

Security Protection	FlashROM Only	FPGA Core Only	FB Core Only	All
No AES / no FlashLock	✓	✓	✓	✓
FlashLock	✓	✓	✓	✓
AES and FlashLock	–	–	–	–

Note: For all instructions, the programming of Flash Blocks refers to Fusion only.

For this scenario, generate the programming file as follows:

1. Select the **Silicon features to be programmed** (Security Settings, FPGA Array, FlashROM, Flash Memory Blocks), as shown in Figure 17-10 on page 17-14 and Figure 17-11 on page 17-14. Click **Next**.

   If **Security Settings** is selected (i.e., the FlashLock security Pass Key feature), an additional dialog will be displayed to prompt you to select the security level setting. If no security setting is selected, you will be directed to Step 3.

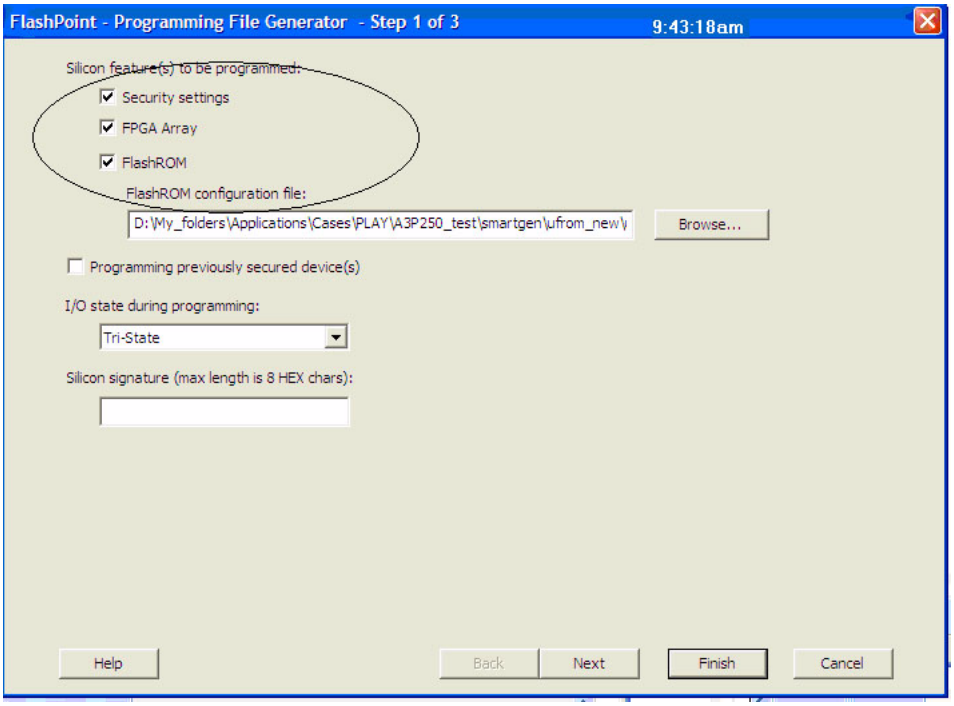


Figure 17-10 • **All Silicon Features Selected for IGLOO and ProASIC3 Devices**

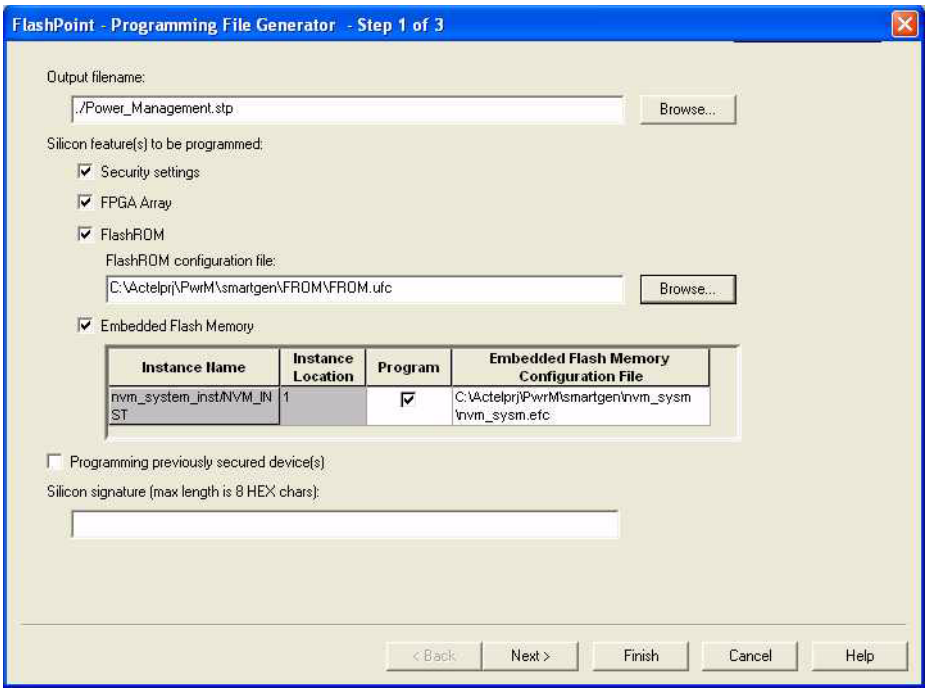


Figure 17-11 • **All Silicon Features Selected for Fusion**

2. Choose the appropriate security level setting and enter a FlashLock Pass Key. The default is the **Medium** security level (Figure 17-12). Click **Next**.

   If you want to select different options for the FPGA and/or FlashROM, this can be set by clicking **Custom Level**. Refer to the "Advanced Options" section on page 17-22 for different custom security level options and descriptions of each.

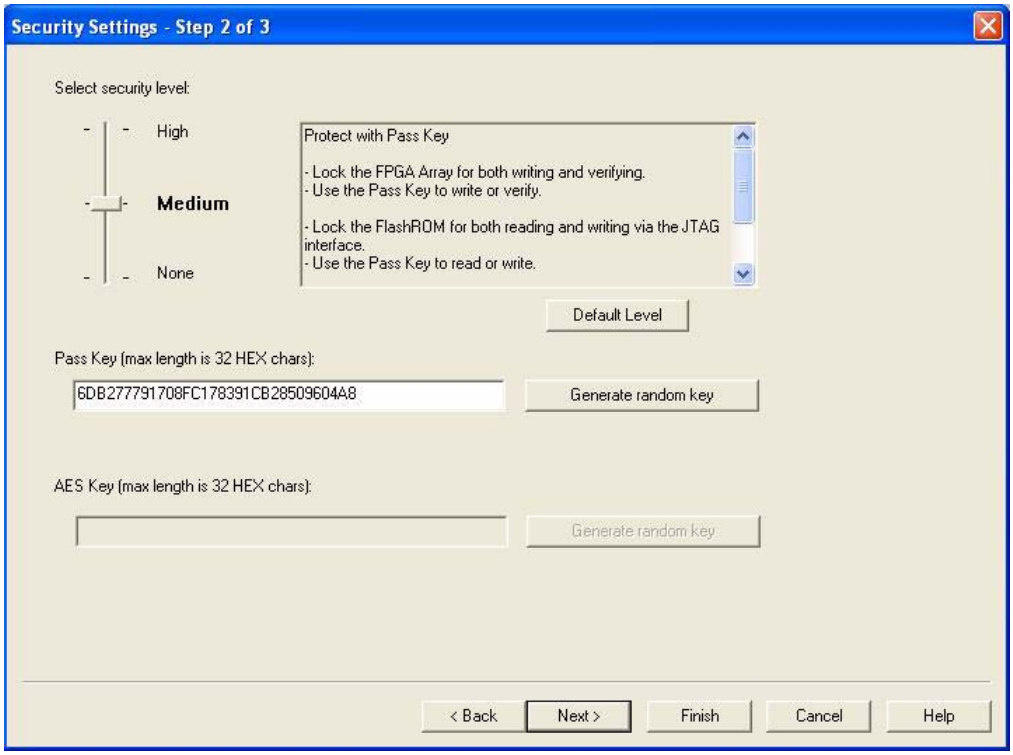


Figure 17-12 • **Medium Security Level Selected for Low-Power Flash Devices**

3. Choose the desired settings for the FlashROM configurations to be programmed (Figure 17-13). Click **Finish** to generate the STAPL programming file for the design.

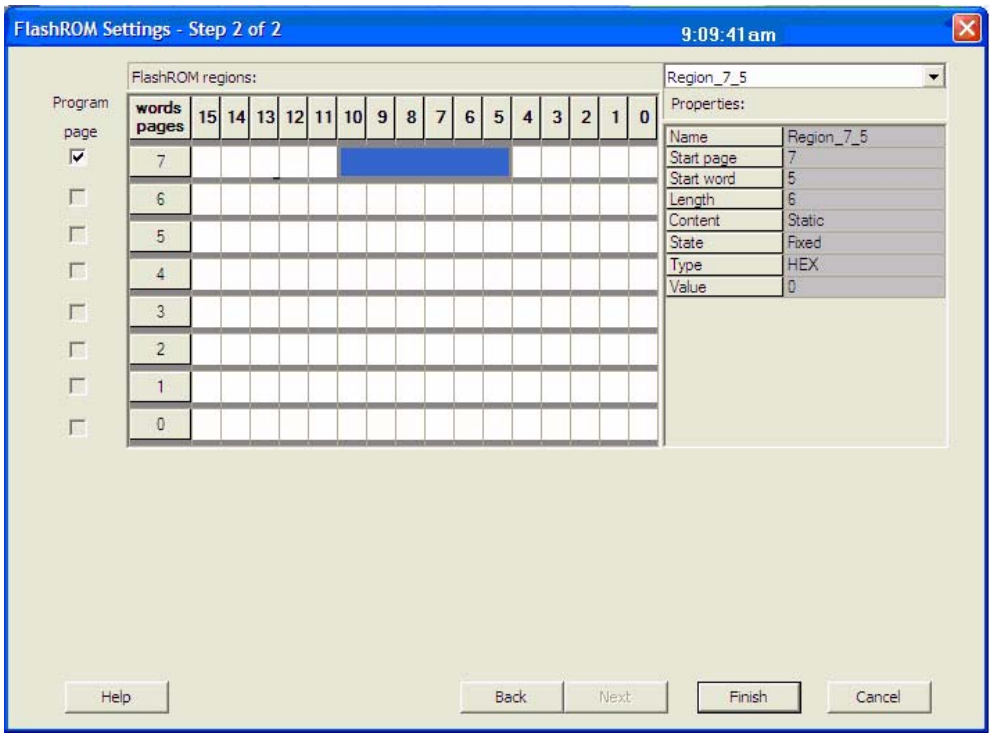


Figure 17-13 • **FlashROM Configuration Settings for Low-Power Flash Devices**

Generation of Security Header Programming File Only—Application 2

As mentioned in the "Application 2: Nontrusted Environment—Unsecured Location" section on page 17-9, the designer may employ FlashLock Pass Key protection or FlashLock Pass Key with AES encryption on the device before sending it to a nontrusted or unsecured location for device programming. To achieve this, the user needs to generate a programming file containing only the security settings desired (Security Header programming file).

Note: If AES encryption is configured, FlashLock Pass Key protection must also be configured.

The available security options are indicated in Table 17-4 and Table 17-5 on page 17-17.

Table 17-4 • **FlashLock Security Options for IGLOO and ProASIC3**

Security Option	FlashROM Only	FPGA Core Only	Both FlashROM and FPGA
No AES / no FlashLock	–	–	–
FlashLock only	✓	✓	✓
AES and FlashLock	✓	✓	✓

Table 17-5 • **FlashLock Security Options for Fusion**

Security Option	FlashROM Only	FPGA Core Only	FB Core Only	All
No AES / no FlashLock	–	–	–	–
FlashLock	✓	✓	✓	✓
AES and FlashLock	✓	✓	✓	✓

For this scenario, generate the programming file as follows:

1. Select only the **Security settings** option, as indicated in Figure 17-14 and Figure 17-15 on page 17-18. Click **Next**.

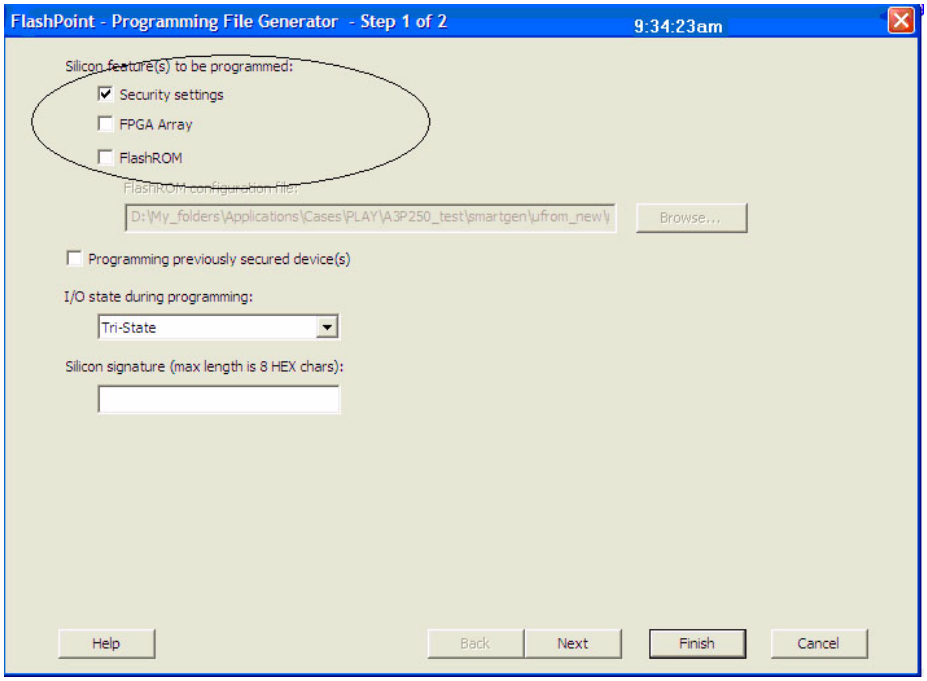


Figure 17-14 • **Programming IGLOO and ProASIC3 Security Settings Only**

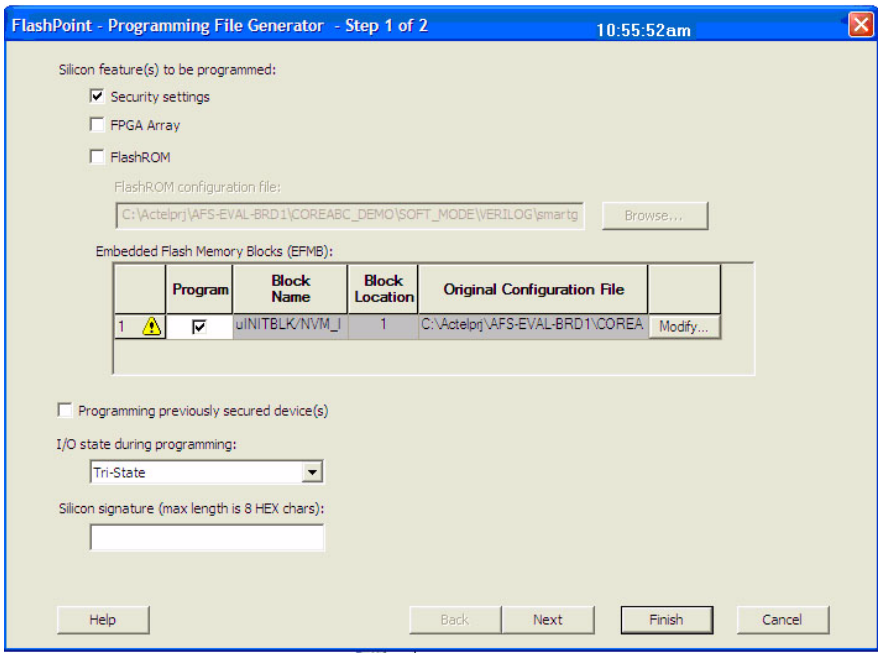


Figure 17-15 • **Programming Fusion Security Settings Only**

2. Choose the desired security level setting and enter the key(s).
   – The **High** security level employs FlashLock Pass Key with AES Key protection.
   – The **Medium** security level employs FlashLock Pass Key protection only.

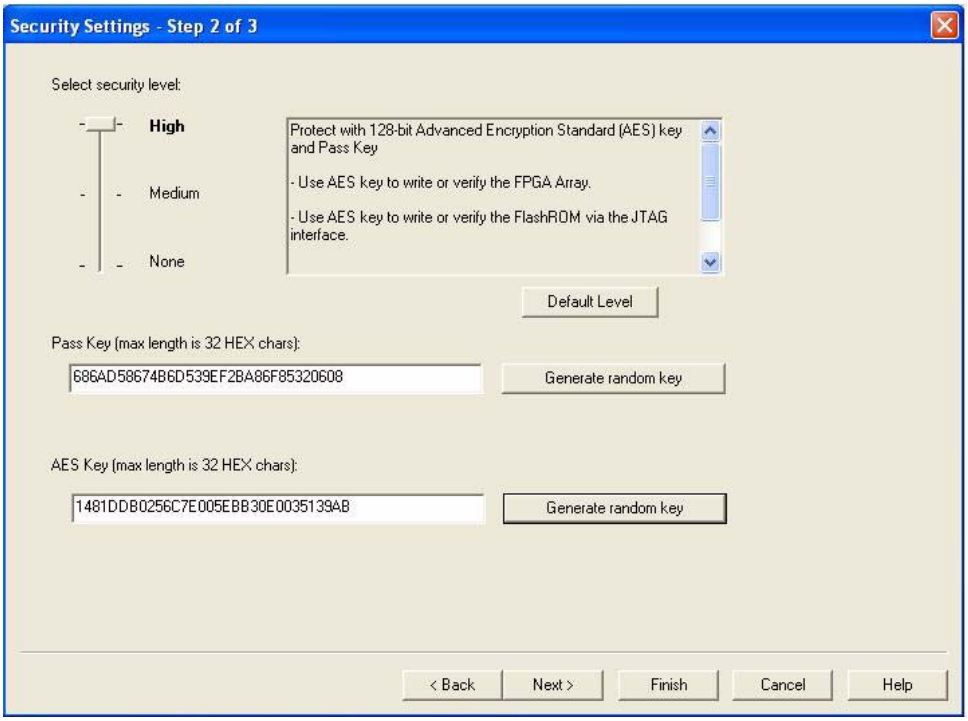


Figure 17-16 • **High Security Level to Implement FlashLock Pass Key and AES Key Protection**

Table 17-6 and Table 17-7 show all available options. If you want to implement custom levels, refer to the "Advanced Options" section on page 17-22 for information on each option and how to set it.

3. When done, click **Finish** to generate the Security Header programming file.

Table 17-6 • **All IGLOO and ProASIC3 Header File Security Options**

Security Option	FlashROM Only	FPGA Core Only	Both FlashROM and FPGA
No AES / no FlashLock	✓	✓	✓
FlashLock only	✓	✓	✓
AES and FlashLock	✓	✓	✓

Note: ✓ = options that may be used

Table 17-7 • **All Fusion Header File Security Options**

Security Option	FlashROM Only	FPGA Core Only	FB Core Only	All
No AES / No FlashLock	✓	✓	✓	✓
FlashLock	✓	✓	✓	✓
AES and FlashLock	✓	✓	✓	✓

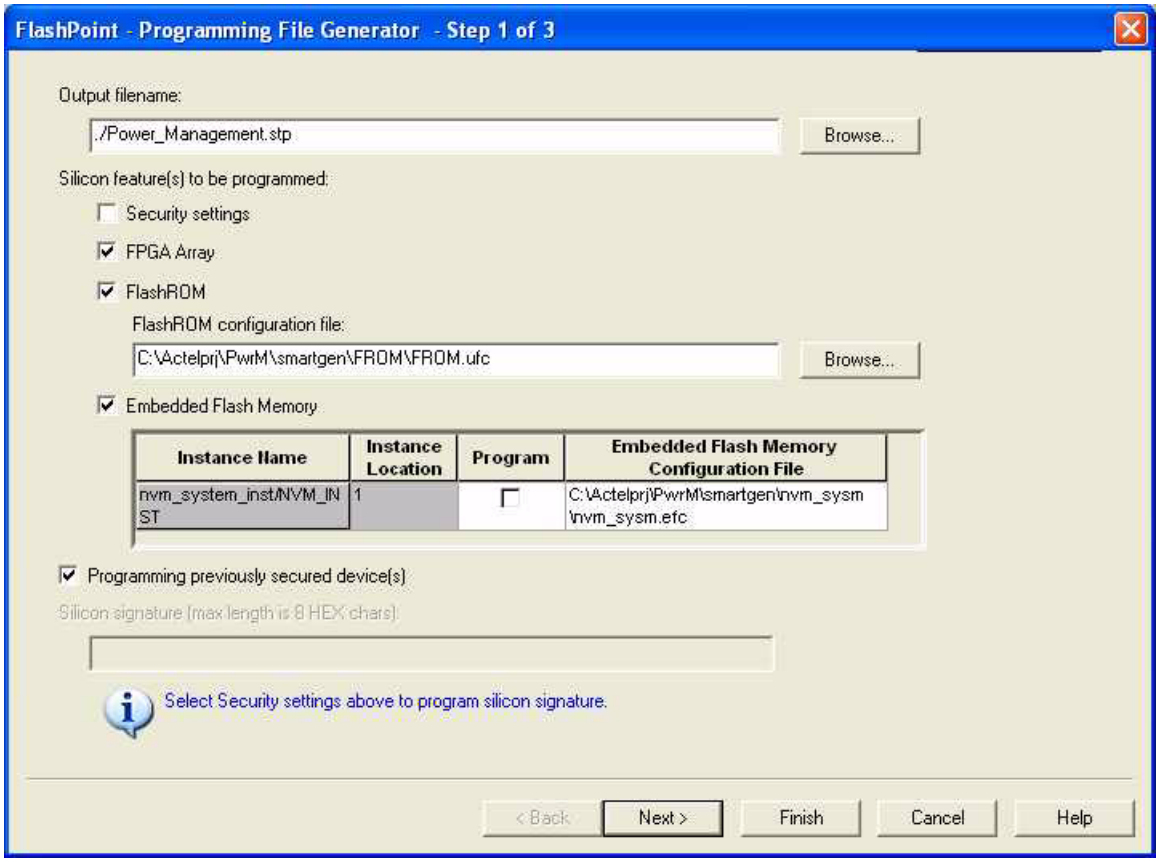
Generation of Programming Files with AES Encryption—Application 3

This section discusses how to generate design content programming files needed specifically at unsecured or remote locations to program devices with a Security Header (FlashLock Pass Key and AES key) already programmed ("Application 2: Nontrusted Environment—Unsecured Location" section on page 17-9 and "Application 3: Nontrusted Environment—Field Updates/Upgrades" section on page 17-10). In this case, the encrypted programming file must correspond to the AES key already programmed into the device. If AES encryption was previously selected to encrypt the FlashROM, FBs, and FPGA array, AES encryption must be set when generating the programming file for them. AES encryption can be applied to the FlashROM only, the FBs only, the FPGA array only, or all. The user must ensure both the FlashLock Pass Key and the AES key match those already programmed to the device(s), and all security settings must match what was previously programmed. Otherwise, the encryption and/or device unlocking will not be recognized when attempting to program the device with the programming file.

The generated programming file will be AES-encrypted.

In this scenario, generate the programming file as follows:

1. Deselect **Security settings** and select the portion of the device to be programmed (Figure 17-17 on page 17-20). Select **Programming previously secured device(s)**. Click **Next**.



Note: The settings in this figure are used to show the generation of an AES-encrypted programming file for the FPGA array, FlashROM, and FB contents. One or all locations may be selected for encryption.

Figure 17-17 • **Settings to Program a Device Secured with FlashLock and using AES Encryption**

Choose the **High** security level to reprogram devices using both the FlashLock Pass Key and AES key protection (Figure 17-18 on page 17-21). Enter the AES key and click **Next**.

A device that has already been secured with FlashLock and has an AES key loaded must recognize the AES key to program the device and generate a valid bitstream in authentication. The FlashLock Key is only required to unlock the device and change the security settings.

This is what makes it possible to program in an untrusted environment. The AES key is protected inside the device by the FlashLock Key, so you can only program if you have the correct AES key. In fact, the AES key is not in the programming file either. It is the key used to encrypt the data in the file. The same key previously programmed with the FlashLock Key matches to decrypt the file.

An AES-encrypted file programmed to a device without FlashLock would not be secure, since without FlashLock to protect the AES key, someone could simply reprogram the AES key first, then program with any AES key desired or no AES key at all. This option is therefore not available in the software.

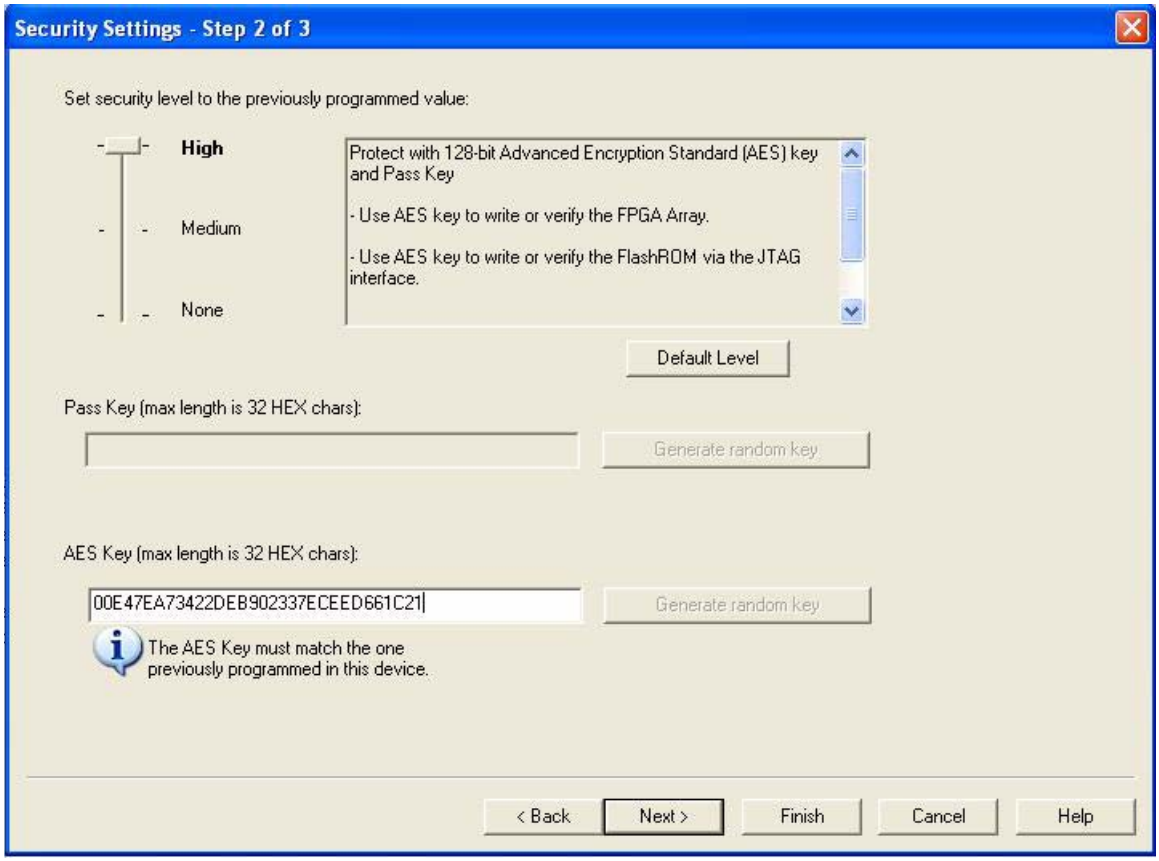


Figure 17-18 • **Security Level Set High to Reprogram Device with AES Key**

Programming with this file is intended for an unsecured environment. The AES key encrypts the programming file with the same AES key already used in the device and utilizes it to program the device.

Reprogramming Devices

Previously programmed devices can be reprogrammed using the steps in the "Generation of the Programming File in a Trusted Environment—Application 1" section on page 17-13 and "Generation of Security Header Programming File Only—Application 2" section on page 17-16. In the case where a FlashLock Pass Key has been programmed previously, the user must generate the new programming file with a FlashLock Pass Key that matches the one previously programmed into the device. The software will check the FlashLock Pass Key in the programming file against the FlashLock Pass Key in the device. The keys must match before the device can be unlocked to perform further programming with the new programming file.

Figure 17-10 on page 17-14 and Figure 17-11 on page 17-14 show the option **Programming previously secured device(s)**, which the user should select before proceeding. Upon going to the next step, the user will be notified that the same FlashLock Pass Key needs to be entered, as shown in Figure 17-19 on page 17-22.

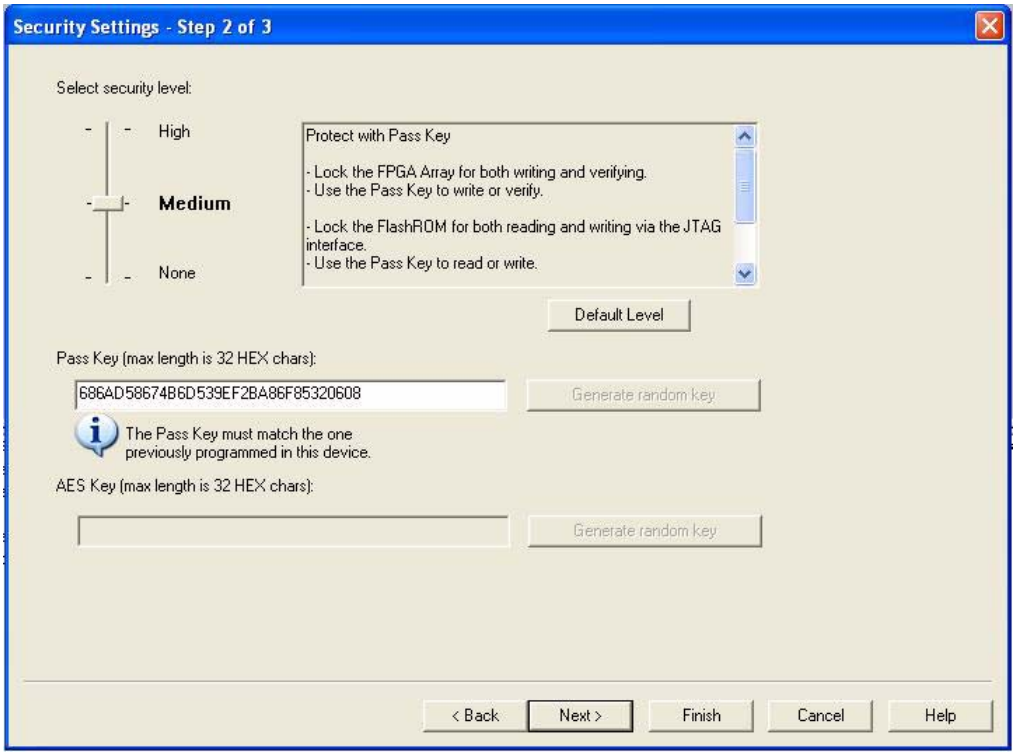


Figure 17-19 • **FlashLock Pass Key, Previously Programmed Devices**

It is important to note that when the security settings need to be updated, the user also needs to select the **Security settings** check box in Step 1, as shown in Figure 17-10 on page 17-14 and Figure 17-11 on page 17-14, to modify the security settings. The user must consider the following:

- If only a new AES key is necessary, the user must re-enter the same Pass Key previously programmed into the device in Designer and then generate a programming file with the same Pass Key and a different AES key. This ensures the programming file can be used to access and program the device and the new AES key.
- If a new Pass Key is necessary, the user can generate a new programming file with a new Pass Key (with the same or a new AES key if desired). However, for programming, the user must first load the original programming file with the Pass Key that was previously used to unlock the device. Then the new programming file can be used to program the new security settings.

Advanced Options

As mentioned, there may be applications where more complicated security settings are required. The "Custom Security Levels" section in the *FlashPro User's Guide* describes different advanced options available to aid the user in obtaining the best available security settings.

Programming File Header Definition

In each STAPL programming file generated, there will be information about how the AES key and FlashLock Pass Key are configured. Table 17-8 shows the header definitions in STAPL programming files for different security levels.

Table 17-8 • **STAPL Programming File Header Definitions by Security Level**

Security Level	STAPL File Header Definition
No security (no FlashLock Pass Key or AES key)	`NOTE "SECURITY" "Disable";`
FlashLock Pass Key with no AES key	`NOTE "SECURITY" "KEYED ";`
FlashLock Pass Key with AES key	`NOTE "SECURITY" "KEYED ENCRYPT ";`
Permanent Security Settings option enabled	`NOTE "SECURITY" "PERMLOCK ENCRYPT ";`
AES-encrypted FPGA array (for programming updates)	`NOTE "SECURITY" "ENCRYPT CORE ";`
AES-encrypted FlashROM (for programming updates)	`NOTE "SECURITY" "ENCRYPT FROM ";`
AES-encrypted FPGA array and FlashROM (for programming updates)	`NOTE "SECURITY" "ENCRYPT FROM CORE ";`

Example File Headers

STAPL Files Generated with FlashLock Key and AES Key Containing Key Information

- FlashLock Key / AES key indicated in STAPL file header definition
- Intended ONLY for secured/trusted environment programming applications

```
================================================
NOTE "CREATOR" "Designer Version: 6.1.1.108";
NOTE "DEVICE" "A3PE600";
NOTE "PACKAGE" "208 PQFP";
NOTE "DATE" "2005/04/08";
NOTE "STAPL_VERSION" "JESD71";
NOTE "IDCODE" "$123261CF";
NOTE "DESIGN" "counter32";
NOTE "CHECKSUM" "$EDB9";
NOTE "SAVE_DATA" "FRomStream";
NOTE "SECURITY" "KEYED ENCRYPT ";
NOTE "ALG_VERSION" "1";
NOTE "MAX_FREQ" "20000000";
NOTE "SILSIG" "$00000000";
NOTE "PASS_KEY" "$00123456789012345678901234567890";
NOTE "AES_KEY" "$ABCDEFABCDEFABCDEFABCDEFABCDEFAB";
================================================
```

STAPL File with AES Encryption

- Does not contain AES key / FlashLock Key information
- Intended for transmission through web or service to unsecured locations for programming

```
============================================
NOTE "CREATOR" "Designer Version: 6.1.1.108";
NOTE "DEVICE" "A3PE600";
NOTE "PACKAGE" "208 PQFP";
NOTE "DATE" "2005/04/08";
NOTE "STAPL_VERSION" "JESD71";
NOTE "IDCODE" "$123261CF";
NOTE "DESIGN" "counter32";
NOTE "CHECKSUM" "$EF57";
NOTE "SAVE_DATA" "FRomStream";
NOTE "SECURITY" "ENCRYPT FROM CORE ";
NOTE "ALG_VERSION" "1";
NOTE "MAX_FREQ" "20000000";
NOTE "SILSIG" "$00000000";
```

Conclusion

The new and enhanced security features offered in Actel Fusion, IGLOO, and ProASIC3 devices provide state-of-the-art security to designs programmed into these flash-based devices. Actel low-power flash devices employ the encryption standard used by NIST and the U.S. government—AES using the 128-bit Rijndael algorithm.

The combination of an on-chip AES decryption engine and Actel FlashLock technology provides the highest level of security against invasive attacks and design theft, implementing the most robust and secure ISP solution. These security features protect IP within the FPGA and protect the system from cloning, wholesale “black box” copying of a design, invasive attacks, and explicit IP or data theft.

Glossary

Term	Explanation
Security Header programming file	Programming file used to program the FlashLock Pass Key and/or AES key into the device to secure the FPGA, FlashROM, and/or FBs.
AES (encryption) key	128-bit key defined by the user when the AES encryption option is set in the Actel Designer software when generating the programming file.
FlashLock Pass Key	128-bit key defined by the user when the FlashLock option is set in the Actel Designer software when generating the programming file. The FlashLock Key protects the security settings programmed to the device. Once a device is programmed with FlashLock, whatever settings were chosen at that time are secure.
FlashLock	The combined security features that protect the device content from attacks. These features are the following: • Flash technology that does not require an external bitstream to program the device • FlashLock Pass Key that secures device content by locking the security settings and preventing access to the device as defined by the user • AES key that allows secure, encrypted device reprogrammability

References

National Institute of Standards and Technology. "ADVANCED ENCRYPTION STANDARD (AES) Questions and Answers." 28 January 2002 (10 January 2005).
See http://csrc.nist.gov/archive/aes/index1.html for more information.

Related Documents

Handbook Documents

FlashROM in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_FlashROM_HBs.pdf

Programming ProASIC3/E Using a Microprocessor

http://www.actel.com/documents/LPD_Microprocessor_HBs.pdf

In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3

http://www.actel.com/documents/LPD_ISP_HBs.pdf

Fusion Embedded Flash Memory Blocks

http://www.actel.com/documents/Fusion_Flash_HBs.pdf

User's Guides

FlashPro User's Guide

http://www.actel.com/documents/flashpro_ug.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information.

Part Number 51700094-014-5

Revised August 2009

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.5)	Page
v1.4 (December 2008)	The "CoreMP7 Device Security" section was removed from "Security in ARM-Enabled Low-Power Flash Devices", since M7-enabled devices are no longer supported.	17-4
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 17-1 · Flash-Based FPGAs.	17-2
v1.2 (June 2008)	The "Security Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	17-2
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 17-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	17-2
v1.0 (January 2008)	The chapter was updated to include the IGLOO PLUS family and information regarding 15 k gate devices.	N/A
	The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	17-2

18 – In-System Programming (ISP) of Actel's Low-Power Flash Devices Using FlashPro3

Introduction

Actel's low-power flash devices are all in-system programmable. This document describes the general requirements for programming a device and specific requirements for the FlashPro3 programmer.

Fusion, IGLOO,® and ProASIC®3 devices offer a low-power, single-chip, live-at-power-up solution with the ASIC advantages of security and low unit cost through nonvolatile flash technology. Each device contains 1 kbit of on-chip, user-accessible, nonvolatile FlashROM. The FlashROM can be used in diverse system applications such as Internet Protocol (IP) addressing, user system preference storage, device serialization, or subscription-based business models. Fusion, IGLOO, and ProASIC3 devices offer the best in-system programming (ISP) solution, FlashLock® security features, and AES-decryption-based ISP.

ISP Architecture

Low-power flash devices support ISP via JTAG and require a single V_{PUMP} voltage of 3.3 V during programming. In addition, programming via a microcontroller in a target system is also supported.

Refer to *Microprocessor Programming of Actel's Low-Power Flash Devices*.

Family-specific support:

- Fusion, ProASIC3, and ProASIC3E devices support ISP.
- ProASIC3L devices operate using a 1.2 V core voltage and support ISP at 1.5 V only. Voltage switching is required in-system to switch from a 1.2 V core to 1.5 V core for programming.
- IGLOO and IGLOOe V5 devices can be programmed in-system when the device is using a 1.5 V supply voltage to the FPGA core.
- All IGLOO V2 and ProASIC3L devices can be operated at any voltage between 1.2 V and1.5 V. Designer software allows 50 mV increments in the voltage. Although devices can operate at 1.2 V core voltage, a device can only be reprogrammed when the core voltage is 1.5 V. Voltage switching is required in-system to switch from a 1.2 V supply (V_{CC},V_{CCI}, and V_{JTAG}) to 1.5 V for programming.

IGLOO devices cannot be programmed in-system when the device is in Flash*Freeze mode. The device should exit Flash*Freeze mode and be in normal operation for programming to start. Programming operations in IGLOO devices can be achieved when the device is in normal operating mode and a 1.5 V core voltage is used.

JTAG 1532

Fusion, IGLOO, and ProASIC3 devices support the JTAG-based IEEE 1532 standard for ISP. To start JTAG operations, the IGLOO device should exit Flash*Freeze mode and be in normal operation before starting to send JTAG commands to the device. As part of this support, when a device is in an unprogrammed state, all user I/O pins are disabled. This is achieved by keeping the global IO_EN signal deactivated, which also has the effect of disabling the input buffers. The SAMPLE/PRELOAD instruction captures the status of pads in parallel and shifts them out as new data is shifted in for loading into the Boundary Scan Register (BSR). When the device is in an unprogrammed state, the SAMPLE/PRELOAD instruction has no effect on I/O status; however, it will continue to shift in new data to be loaded into the BSR. Therefore, when SAMPLE/PRELOAD is used on an unprogrammed device, the BSR will be loaded with undefined data. For JTAG timing information on setup, hold, and fall times, refer to the *FlashPro User's Guide*.

ISP Support in Flash-Based Devices

The flash FPGAs listed in Table 18-1 support the ISP feature and the functions described in this document.

Table 18-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 18-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 18-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Programming Voltage (V_{PUMP}) and V_{JTAG}

Low-power flash devices support on-chip charge pumps, and therefore require only a single 3.3 V programming voltage for the V_{PUMP} pin during programming. When the device is not being programmed, the V_{PUMP} pin can be left floating or can be tied (pulled up) to any voltage between 0 V and 3.6 V. During programming, the target board or the FlashPro3 programmer can provide V_{PUMP}. FlashPro3 is capable of supplying V_{PUMP} to a single device. If more than one device is to be programmed using FlashPro3 on a given board, FlashPro3 should not be relied on to supply the V_{PUMP} voltage.

Low-power flash device I/Os support a bank-based, voltage-supply architecture that simultaneously supports multiple I/O voltage standards (Table 18-2 on page 18-3). By isolating the JTAG power supply in a separate bank from the user I/Os, low-power flash devices provide greater flexibility with supply selection and simplify power supply and printed circuit board (PCB) design. The JTAG pins can be run at any voltage from 1.5 V to 3.3 V (nominal). Actel recommends that TCK be tied to GND or V_{JTAG} when not used. This prevents a possible totempole current on the input buffer stage. For TDI, TMS, and TRST pins, the devices provide an internal nominal 10 kΩ pull-up resistor. During programming, all I/O pins, except for JTAG interface pins, are tristated and weakly pulled up to V_{CCI}. This isolates the part and prevents the signals from floating. The JTAG interface pins are driven by the FlashPro3 during programming, including the TRST pin, which is driven HIGH.

Table 18-2 • **Power Supplies**

Power Supply	Programming Mode	Current during Programming
V_{CC}	1.5 V	< 70 mA
V_{CCI}	1.5 V / 1.8 V / 2.5 V / 3.3 V (bank-selectable)	I/Os are weakly pulled up.
V_{JTAG}	1.5 V / 1.8 V / 2.5 V / 3.3 V	< 20 mA
V_{PUMP}	3.0 V to 3.6 V	< 80 mA

Note: All supply voltages should be at 1.5 V or higher, regardless of the setting during normal operation.

IEEE 1532 (JTAG) Interface

The supported industry-standard IEEE 1532 programming interface builds on the IEEE 1149.1 (JTAG) standard. IEEE 1532 defines the standardized process and methodology for ISP. Both silicon and software issues are addressed in IEEE 1532 to create a simplified ISP environment. Any IEEE 1532–compliant programmer can be used to program low-power flash devices. However, only limited security and FlashROM features are supported when using the IEEE 1532 standard. The Actel FlashPro3 programmer was developed exclusively for these devices and will support all the security and device serialization features. Refer to the standard for detailed information about IEEE 1532.

Security

Unlike SRAM-based FPGAs that require loading at power-up from an external source such as a microcontroller or boot PROM, Actel nonvolatile devices are live at power-up, and there is no bitstream required to load the device when power is applied. The unique flash-based architecture prevents reverse engineering of the programmed code on the device, because the programmed data is stored in nonvolatile memory cells. Each nonvolatile memory cell is made up of small capacitors and any physical deconstruction of the device will disrupt stored electrical charges.

Each low-power flash device has a built-in 128-bit Advanced Encryption Standard (AES) decryption core, except for the 30 k gate devices and smaller. Any FPGA core or FlashROM content loaded into the device can optionally be sent as encrypted bitstream and decrypted as it is loaded. This is

particularly suitable for applications where device updates must be transmitted over an unsecured network such as the Internet. The embedded AES decryption core can prevent sensitive data from being intercepted (Figure 18-1 on page 18-4). A single 128-bit AES Key (32 hex characters) is used to encrypt FPGA core programming data and/or FlashROM programming data in the Actel tools. The low-power flash devices also decrypt with a single 128-bit AES Key. In addition, low-power flash devices support a Message Authentication Code (MAC) for authentication of the encrypted bitstream on-chip. This allows the encrypted bitstream to be authenticated and prevents erroneous data from being programmed into the device. The FPGA core, FlashROM, and Flash Memory Blocks (FBs), in Fusion only, can be updated independently using a programming file that is AES-encrypted (cipher text) or uses plain text.

Security in ARM-Enabled Low-Power Flash Devices

There are slight differences between the regular flash device and the ARM®-enabled flash devices, which have the M1 and M7 prefix.

The AES key is used by Actel and preprogrammed into the device to protect the ARM IP. As a result, the design will be encrypted along with the ARM IP, according to the details below.

Cortex-M1 Device Security

Cortex-M1–enabled devices are shipped with the following security features:

- FPGA array enabled for AES-encrypted programming and verification
- FlashROM enabled for AES-encrypted write and verify
- Fusion Embedded Flash Memory enabled for AES encrypted write

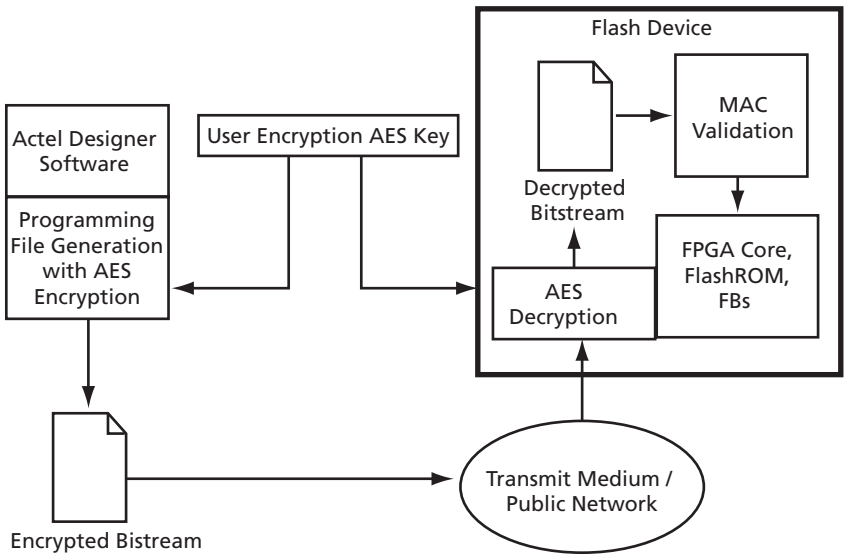


***Figure 18-1* • AES-128 Security Features**

Figure 18-2 shows different applications for ISP programming.

1. In a trusted programming environment, you can program the device using the unencrypted (plaintext) programming file.
2. You can program the AES Key in a trusted programming environment and finish the final programming in an untrusted environment using the AES-encrypted (cipher text) programming file.
3. For the remote ISP updating/reprogramming, the AES Key stored in the device enables the encrypted programming bitstream to be transmitted through the untrusted network connection.

Actel low-power flash devices also provide the unique Actel FlashLock feature, which protects the Pass Key and AES Key. Unless the original FlashLock Pass Key is used to unlock the device, security settings cannot be modified. Low-power flash devices do not support read-back of FPGA core-programmed data; however, the FlashROM contents can selectively be read back (or disabled) via the JTAG port based on the security settings established by the Actel Designer software. Refer to *Security in Low-Power Flash Devices* for more information.

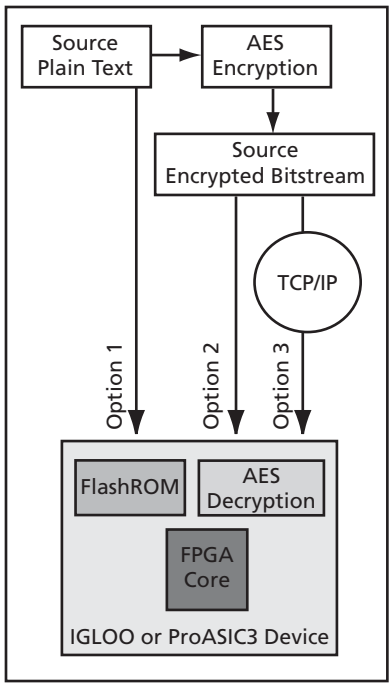


Figure 18-2 • **Different ISP Use Models**

FlashROM and Programming Files

Each low-power flash device has 1 kbit of on-chip, nonvolatile flash memory that can be accessed from the FPGA core. This nonvolatile FlashROM is arranged in eight pages of 128 bits (Figure 18-3). Each page can be programmed independently, with or without the 128-bit AES encryption. The FlashROM can only be programmed via the IEEE 1532 JTAG port and cannot be programmed from the FPGA core. In addition, during programming of the FlashROM, the FPGA core is powered down automatically by the on-chip programming control logic.

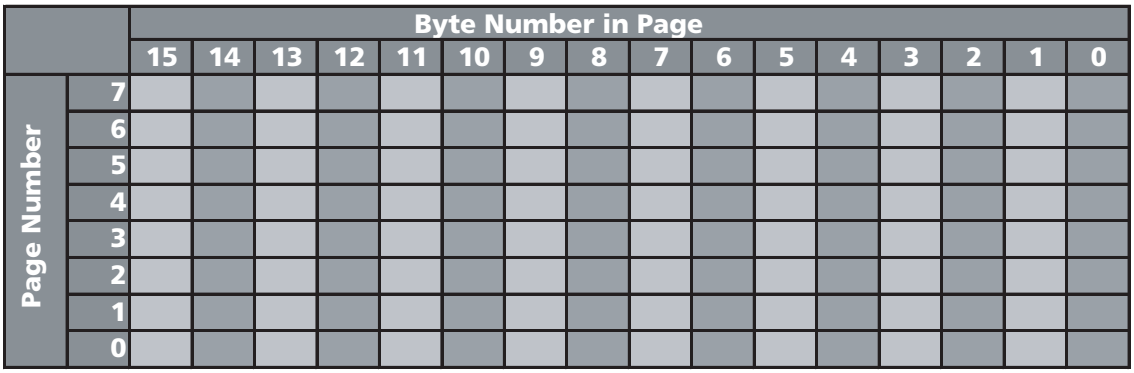


***Figure 18-3* • FlashROM Architecture**

Using FlashROM combined with AES, many subscription-based applications or device serialization applications are possible. SmartGen supports easy management of the FlashROM contents even over large numbers of devices. SmartGen can support FlashROM contents that contain the following:

- Static values
- Random numbers
- Values read from a file
- Independent updates of each page

In addition, auto-incrementing of fields is possible. In applications where the FlashROM content is different for each device, you have the option to generate a single STAPL file for all the devices or individual serialization files for each device. For more information on how to generate the FlashROM content for device serialization, refer to *FlashROM in Actel's Low-Power Flash Devices*.

Actel Libero® Integrated Designed Environment (IDE) includes a unique tool to support the generation and management of FlashROM and FPGA programming files. This tool is called FlashPoint.

Depending on the applications, designers can use the FlashPoint software to generate a STAPL file with different contents. In each case, optional AES encryption and/or different security settings can be set.

In Designer, when you click the Programming File icon, FlashPoint launches, and you can generate STAPL file(s) with four different cases (Figure 18-4 on page 18-7). When the serialization feature is used during the configuration of FlashROM in SmartGen, you can generate a single STAPL file that will program all the devices or an individual STAPL file for each device.

The following cases present the FPGA core and FlashROM programming file combinations that can be used for different applications. In each case, you can set the optional security settings (FlashLock Pass Key and/or AES Key) depending on the application.

1. A single STAPL file or multiple STAPL files with multiple FlashROM contents and the FPGA core content. A single STAPL file will be generated if the device serialization feature is not used. You can program the whole FlashROM or selectively program individual pages.
2. A single STAPL file for the FPGA core content

3. A single STAPL file or multiple STAPL files with multiple FlashROM contents. A single STAPL file will be generated if the device serialization feature is not used. You can program the whole FlashROM or selectively program individual pages.
4. A single STAPL file to configure the security settings for the device, such as the AES Key and/or Pass Key.

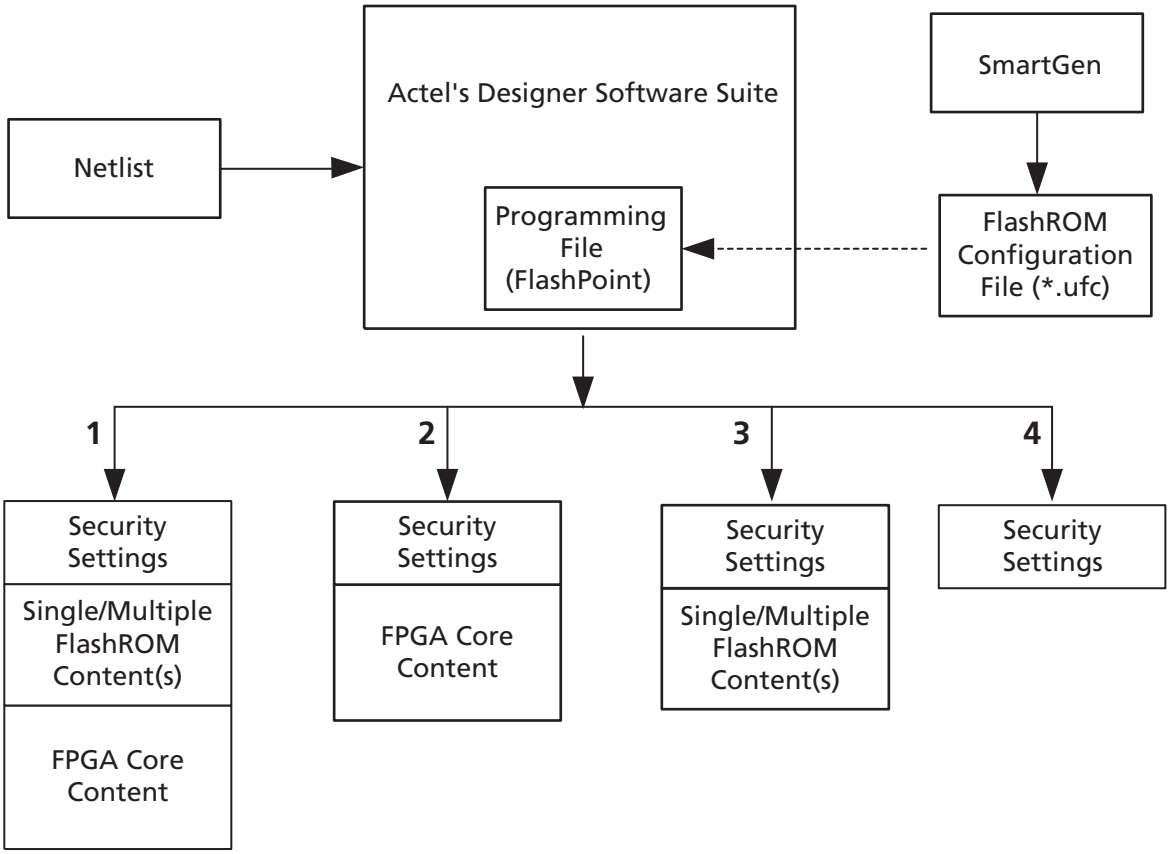


Figure 18-4 • **Flexible Programming File Generation for Different Applications**

Programming Solution

For device programming, any IEEE 1532–compliant programmer can be used; however, the FlashPro3 programmer must be used to control the low-power flash device's rich security features and FlashROM programming options. The FlashPro3 programmer is a low-cost portable programmer for the Actel flash families. It can also be used with a powered USB hub for parallel programming. General specifications for the FlashPro3 programmer are as follows:

- Programming clock – TCK is used with a maximum frequency of 20 MHz, and the default frequency is 4 MHz.
- Programming file – STAPL
- Daisy chain – Supported. You can use the ChainBuilder software to build the programming file for the chain.
- Parallel programming – Supported. Multiple FlashPro3 programmers can be connected together using a powered USB hub or through the multiple USB ports on the PC.
- Power supply – The target board must provide V_{CC}, V_{CCI}, V_{PUMP}, and V_{JTAG} during programming. However, if there is only one device on the target board, the FlashPro3 programmer can generate the required V_{PUMP} voltage from the USB port.

ISP Programming Header Information

The FlashPro3 programming cable connector can be connected with a 10-pin, 0.1"-pitch programming header. The recommended programming headers are manufactured by AMP (103310-1) and 3M (2510-6002UB). If you have limited board space, you can use a compact programming header manufactured by Samtec (FTSH-105-01-L-D-K). Using this compact programming header, you are required to order an additional header adapter manufactured by Actel (FP3-26PIN-ADAPTER).

Existing ProASIC^{PLUS} family customers who are using the Samtec Small Programming Header (FTSH-113-01-L-D-K) and are planning to migrate to IGLOO or ProASIC3 devices can order a separate adapter kit from Actel (FP3-10PIN-ADAPTER-KIT), which contains a compact 10-pin adapter kit as well as 26-pin migration capability.

Table 18-3 • **Programming Header Ordering Codes**

Manufacturer	Part Number	Description
AMP	103310-1	10-pin, 0.1"-pitch cable header (right-angle PCB mount angle)
3M	2510-6002UB	10-pin, 0.1"-pitch cable header (straight PCB mount angle)
Samtec	FTSH-113-01-L-D-K	Small programming header supported by FlashPro and Silicon Sculptor
Samtec	FTSH-105-01-L-D-K	Compact programming header
Samtec	FFSD-05-D-06.00-01-N	10-pin cable with 50 mil pitch sockets; included in FP3-10PIN-ADAPTER-KIT.
Actel	FP3-10PIN-ADAPTER-KIT	Compact header and migration kit

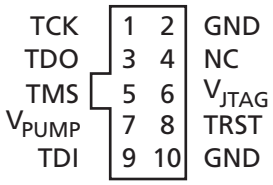


Figure 18-5 • **Programming Header (top view)**

Table 18-4 • **Programming Header Pin Numbers and Description**

Pin	Signal	Source	Description
1	TCK	Programmer	JTAG Clock
2	GND[1]	–	Signal Reference
3	TDO	Target Board	Test Data Output
4	NC	–	No Connect
5	TMS	Programmer	Test Mode Select
6	V_{JTAG}	Target Board	JTAG Supply Voltage
7	V_{PUMP}[2]	Programmer/Target Board	Programming Supply Voltage
8	nTRST	Programmer	JTAG Test Reset (Hi-Z with 10 kΩ pull-down, HIGH, LOW, or toggling)
9	TDI	Programmer	Test Data Input
10	GND[1]	–	Signal Reference

Notes:

1. *Both GND pins must be connected.*
2. *FlashPro3 can provide V_{PUMP} if there is only one device on the target board.*

Board-Level Considerations

A bypass capacitor is required from V_{PUMP} to GND for all low-power flash devices during programming. This bypass capacitor protects the devices from voltage spikes that may occur on the V_{PUMP} supplies during the erase and programming cycles. Refer to *Pin Descriptions* for specific recommendations. For proper programming, 0.01 µF and 0.33 µF capacitors (both rated at 16 V) are to be connected in parallel across V_{PUMP} and GND, and positioned as close to the FPGA pins as possible. The bypass capacitor must be placed within 2.5 cm of the device pins.

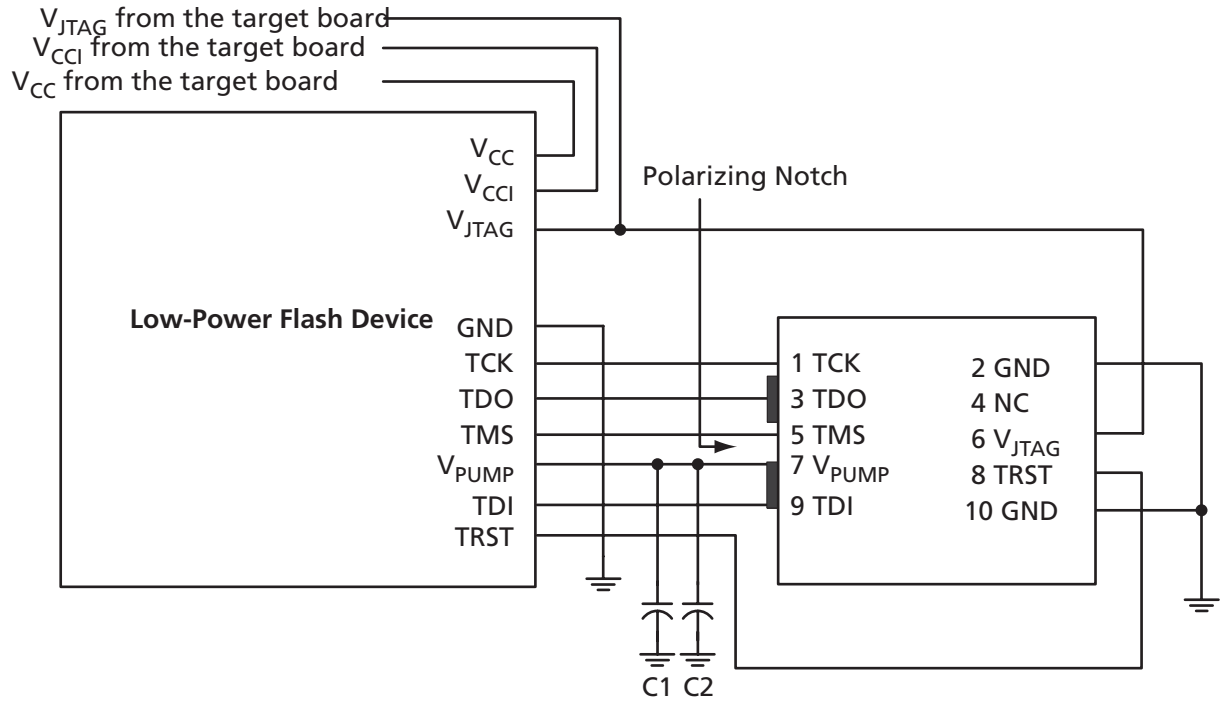


***Figure 18-6* • Board Layout and Programming Header Top View**

Troubleshooting Signal Integrity

Symptoms of a Signal Integrity Problem

A signal integrity problem can manifest itself in many ways. The problem may show up as extra or dropped bits during serial communication, changing the meaning of the communication. There is a normal variation of threshold voltage and frequency response between parts even from the same lot. Because of this, the effects of signal integrity may not always affect different devices on the same board in the same way. Sometimes, replacing a device appears to make signal integrity problems go away, but this is just masking the problem. Different parts on identical boards will exhibit the same problem sooner or later. It is important to fix signal integrity problems early. Unless the signal integrity problems are severe enough to completely block all communication between the device and the programmer, they may show up as subtle problems. Some of the FlashPro3 exit codes that are caused by signal integrity problems are listed below. Signal integrity problems are not the only possible cause of these errors, but this list is intended to show where problems can occur. FlashPro3 allows TCK to be lowered from 24 MHz down to 1 MHz to allow you to address some signal integrity problems that may occur with impedance mismatching at higher frequencies.

Chain Integrity Test Error or Analyze Chain Failure

Normally, the FlashPro3 Analyze Chain command expects to see 0x2 on the TDO pin. If the command reports reading 0x0 or 0x3, it is seeing the TDO pin stuck at 0 or 1. The only time the TDO

pin comes out of tristate is when the JTAG TAP state machine is in the Shift-IR or Shift-DR state. If noise or reflections on the TCK or TMS lines have disrupted the correct state transitions, the device's TAP state controller might not be in one of these two states when the programmer tries to read the device. When this happens, the output is floating when it is read and does not match the expected data value. This can also be caused by a broken TDO net. Only a small amount of data is read from the device during the Analyze Chain command, so marginal problems may not always show up during this command.

Exit 11

This error occurs during the verify stage of programming a device. After programming the design into the device, the device is verified to ensure it is programmed correctly. The verification is done by shifting the programming data into the device. An internal comparison is performed within the device to verify that all switches are programmed correctly. Noise induced by poor signal integrity can disrupt the writes and reads or the verification process and produce a verification error. While technically a verification error, the root cause is often related to signal integrity.

Refer to the *FlashPro User's Guide* for other error messages and solutions. For the most up-to-date known issues and solutions, refer to http://www.actel.com/support.

Conclusion

Fusion, IGLOO, and ProASIC3 devices offer a low-cost, single-chip solution that is live at power-up through nonvolatile flash technology. The FlashLock Pass Key and 128-bit AES Key security features enable secure ISP in an untrusted environment. On-chip FlashROM enables a host of new applications, including device serialization, subscription-based applications, and IP addressing. Additionally, as the FlashROM is nonvolatile, all of these services can be provided without battery backup.

Related Documents

Handbook Documents

Microprocessor Programming of Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_Microprocessor_HBs.pdf

Security in Low-Power Flash Devices

http://www.actel.com/LPD_Security_HBs.pdf

FlashROM in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_FlashROM_HBs.pdf

Pin Descriptions

http://www.actel.com/documents/LPD_PinDescriptions_HBs.pdf

User's Guides

FlashPro User's Guide

http://www.actel.com/documents/flashpro_ug.pdf

Part Number and Revision Date

This document was previously published as an application note describing features and functions of the device, and as such has now been incorporated into the device handbook format. No technical changes have been made to the content.

Part Number 51700094-015-5

Revised August 2009

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.5)	Page
v1.4 (December 2008)	The "CoreMP7 Device Security" section was removed from "Security in ARM-Enabled Low-Power Flash Devices", since M7-enabled devices are no longer supported.	18-4
v1.3 (October 2008)	The "ISP Architecture" section was revised to include information about core voltage for IGLOO V2 and ProASIC3L devices, as well as 50 mV increments allowable in Designer software.	18-1
	IGLOO nano and ProASIC3 nano devices were added to Table 18-1 · Flash-Based FPGAs.	18-2
	A second capacitor was added to Figure 18-6 · Board Layout and Programming Header Top View.	18-9
v1.2 (June 2008)	The "ISP Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	18-2
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 18-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	18-2
v1.0 (January 2008)	The "ISP Architecture" section was updated to included the IGLOO PLUS family in the discussion of family-specific support. The text, "When 1.2 V is used, the device can be reprogrammed in-system at 1.5 V only," was revised to state, "Although the device can operate at 1.2 V core voltage, the device can only be reprogrammed when all supplies (V_{CC}, V_{CCI}, and V_{JTAG}) are at 1.5 V."	18-1
	The "ISP Support in Flash-Based Devices" section and Table 18-1 · Flash-Based FPGAs were updated to include the IGLOO PLUS family. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	18-2
	The "Security" section was updated to mention that 15 k gate devices do not have a built-in 128-bit decryption core.	18-3
	Table 18-2 · Power Supplies was revised to remove the Normal Operation column and add a table note stating, "All supply voltages should be at 1.5 V or higher, regardless of the setting during normal operation."	18-3
	The "ISP Programming Header Information" section was revised to change FP3-26PIN-ADAPTER to FP3-10PIN-ADAPTER-KIT. Table 18-3 · Programming Header Ordering Codes was updated with the same change, as well as adding the part number FFSD-05-D-06.00-01-N, a 10-pin cable with 50-mil-pitch sockets.	18-8
	The "Board-Level Considerations" section was updated to describe connecting two capacitors in parallel across V_{PUMP} and GND for proper programming.	18-9

Previous Version	Changes in Current Version (v1.5)	Page
51900055-2/7.06	Information was added to the "Programming Voltage (VPUMP) and VJTAG" section about the JTAG interface pin.	18-3
51900055-1/1.05	ACTgen was changed to SmartGen.	N/A
	In Figure 18-6 · Board Layout and Programming Header Top View, the order of the text was changed to: V_{JTAG} from the target board V_{CCI} from the target board V_{CC} from the target board	18-9

19 – Microprocessor Programming of Actel's Low-Power Flash Devices

Introduction

The Fusion, IGLOO,® and ProASIC®3 families of flash FPGAs support in-system programming (ISP) with the use of a microprocessor. Flash-based FPGAs store their configuration information in the actual cells within the FPGA fabric. SRAM-based devices need an external configuration memory, and hybrid nonvolatile devices store the configuration in a flash memory inside the same package as the SRAM FPGA. Since the programming of a true flash FPGA is simpler, requiring only one stage, it makes sense that programming with a microprocessor in-system should be simpler than with other SRAM FPGAs. This reduces bill-of-materials costs and printed circuit board (PCB) area, and increases system reliability.

Nonvolatile flash technology also gives the low-power flash devices the advantage of a secure, low-power, live-at-power-up, and single-chip solution. Low-power flash devices are reprogrammable and offer time-to-market benefits at an ASIC-level unit cost. These features enable engineers to create high-density systems using existing ASIC or FPGA design flows and tools.

This document is an introduction to microprocessor programming only. To explain the difference between the options available, user's guides for DirectC and STAPL provide more detail on implementing each style.

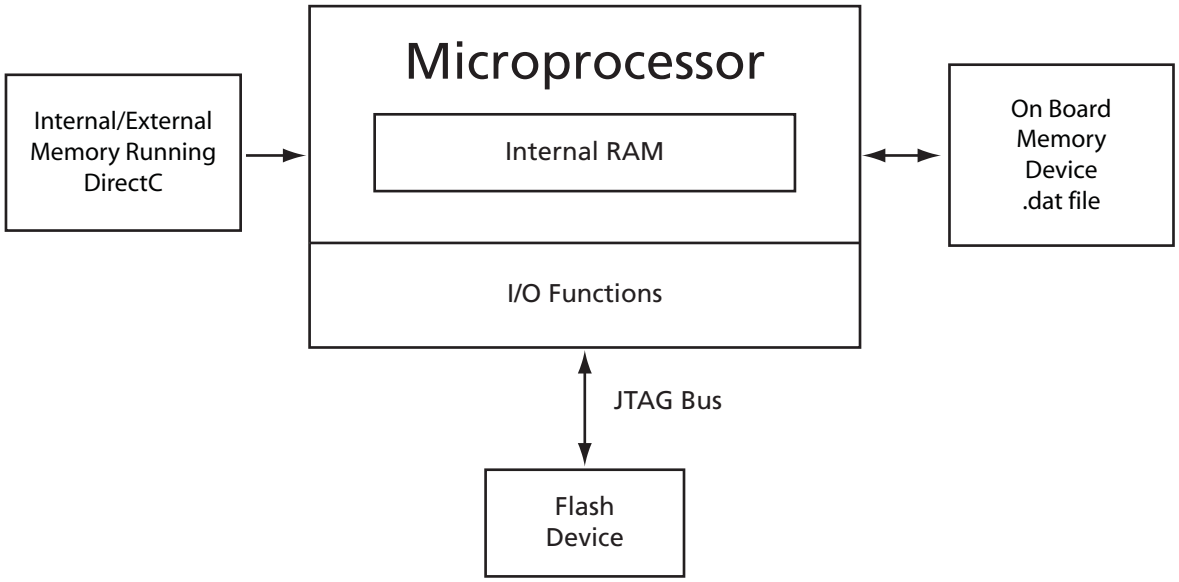


***Figure 19-1* • ISP Using Microprocessor**

Microprocessor Programming Support in Flash Devices

The flash-based FPGAs listed in Table 19-1 support programming with a microprocessor and the functions described in this document.

Table 19-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 19-1. Where the information applies to only one device or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 19-1. Where the information applies to only one device or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Programming Algorithm

JTAG Interface

The low-power flash families are fully compliant with the IEEE 1149.1 (JTAG) standard. They support all the mandatory boundary scan instructions (EXTEST, SAMPLE/PRELOAD, and BYPASS) as well as six optional public instructions (USERCODE, IDCODE, HIGHZ, and CLAMP).

IEEE 1532

The low-power flash families are also fully compliant with the IEEE 1532 programming standard. The IEEE 1532 standard adds programming instructions and associated data registers to devices that comply with the IEEE 1149.1 standard (JTAG). These instructions and registers extend the capabilities of the IEEE 1149.1 standard such that the Test Access Port (TAP) can be used for configuration activities. The IEEE 1532 standard greatly simplifies the programming algorithm, reducing the amount of time needed to implement microprocessor ISP.

Implementation Overview

To implement device programming with a microprocessor, the user should first download the C-based STAPL player or DirectC code from the Actel website. (See the Actel website for future updates regarding the STAPL player and DirectC code). Using the easy-to-follow Actel user's guide, create the low-level application programming interface (API) to provide the necessary basic functions. These API functions act as the interface between the programming software and the actual hardware (Figure 19-2).

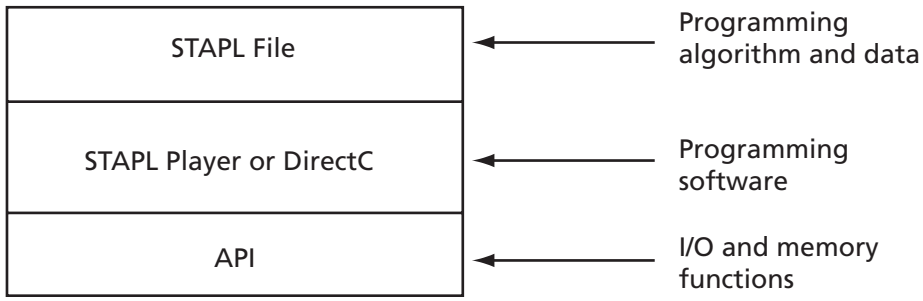


***Figure 19-2* • Device Programming Code Relationship**

The API is then linked with the STAPL player or DirectC and compiled using the microprocessor's compiler. Once the entire code is compiled, the user must download the resulting binary into the MCU system's program memory (such as ROM, EEPROM, or flash). The system is now ready for programming.

To program a design into the FPGA, the user creates a bitstream or STAPL file using the Actel Designer software, downloads it into the MCU system's volatile memory, and activates the stored programming binary file (Figure 19-3 on page 19-4). Once the programming is completed, the bitstream or STAPL file can be removed from the system, as the configuration profile is stored in the flash FPGA fabric and does not need to be reloaded at every system power-on.

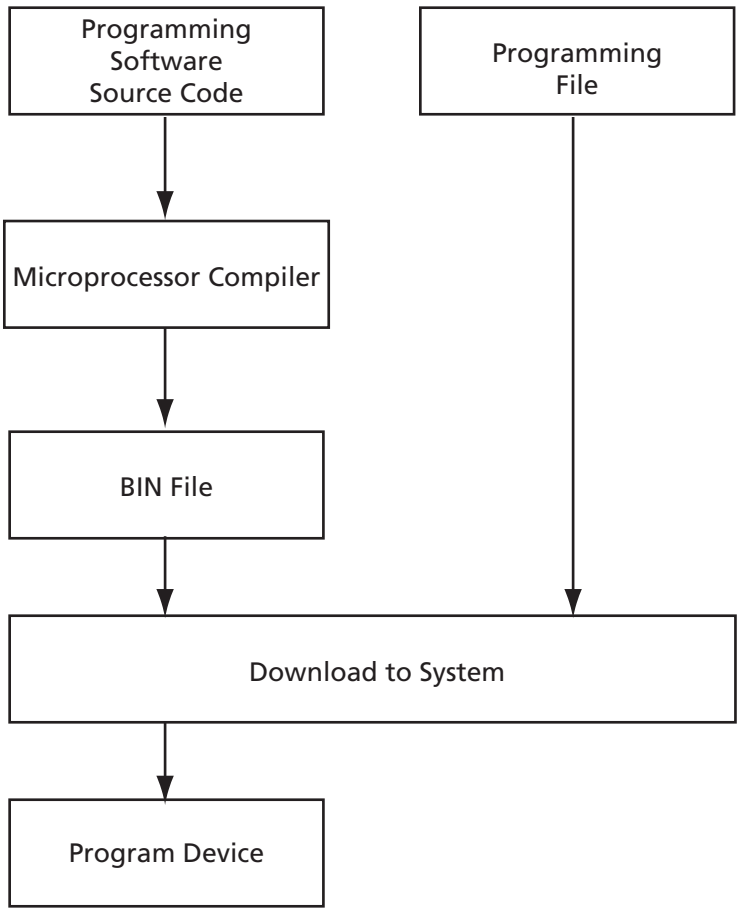


***Figure 19-3* • MCU FPGA Programming Model**

FlashROM

Actel low-power flash devices have 1 kbit of user-accessible, nonvolatile, FlashROM on-chip. This nonvolatile FlashROM can be programmed along with the core or on its own using the standard IEEE 1532 JTAG programming interface.

The FlashROM is architected as eight pages of 128 bits. Each page can be individually programmed (erased and written). Additionally, on-chip AES security decryption can be used selectively to load data securely into the FlashROM (e.g., over public or private networks, such as the Internet). Refer to *FlashROM in Actel's Low-Power Flash Devices*.

STAPL vs. DirectC

Programming the low-power flash devices is performed using DirectC or the STAPL player. Both tools use the STAPL file as an input. DirectC is a compiled language, whereas STAPL is an interpreted language. Microprocessors will be able to load the FPGA using DirectC much more quickly than STAPL. This speed advantage becomes more apparent when lower clock speeds of 8- or 16-bit microprocessors are used. DirectC also requires less memory than STAPL, since the programming algorithm is directly implemented. STAPL does have one advantage over DirectC—the ability to upgrade. When a new programming algorithm is required, the STAPL user simply needs to regenerate a STAPL file using the latest version of the Designer software and download it to the system. The DirectC user must download the latest version of DirectC from Actel, compile everything, and download the result into the system (Figure 19-4).

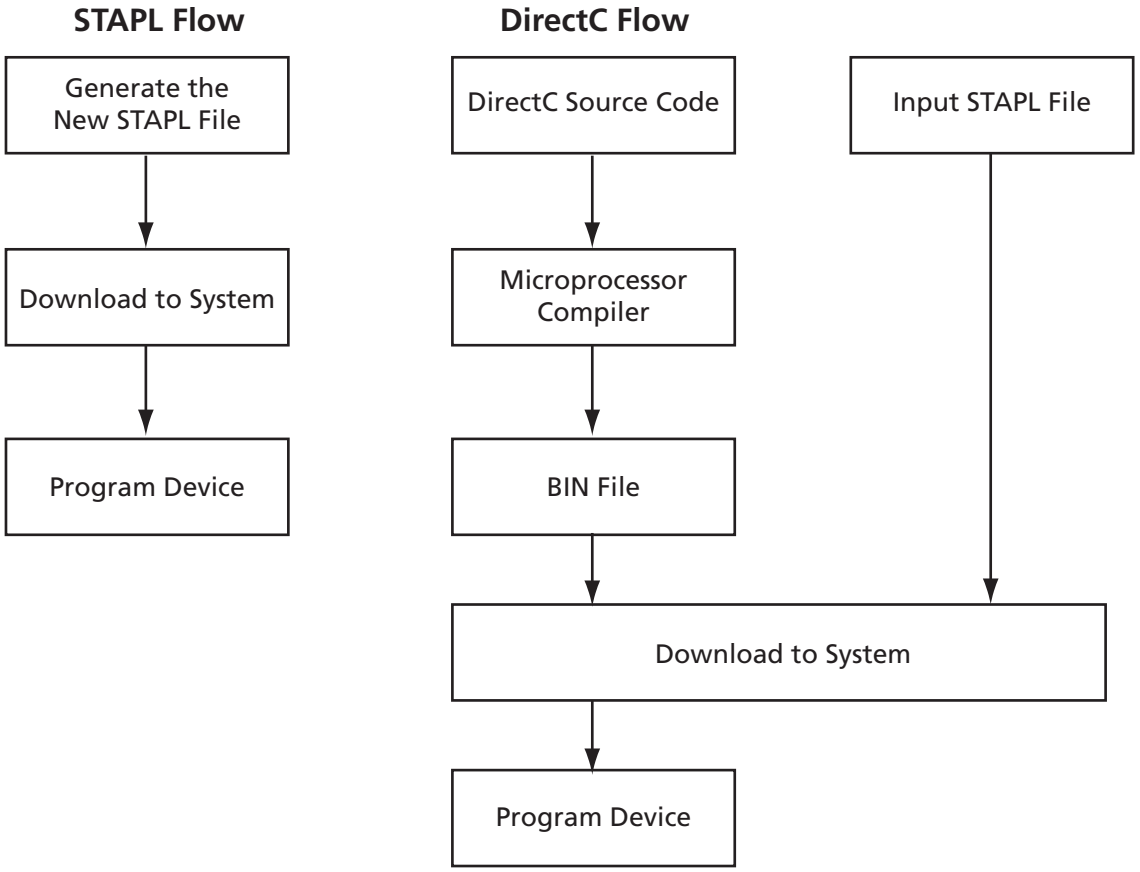


***Figure 19-4* • STAPL vs. DirectC**

Remote Upgrade via TCP/IP

Transmission Control Protocol (TCP) provides a reliable bitstream transfer service between two endpoints on a network. TCP depends on Internet Protocol (IP) to move packets around the network on its behalf. TCP protects against data loss, data corruption, packet reordering, and data duplication by adding checksums and sequence numbers to transmitted data and, on the receiving side, sending back packets and acknowledging the receipt of data.

The system containing the low-power flash device can be assigned an IP address when deployed in the field. When the device requires an update (core or FlashROM), the programming instructions along with the new programming data (AES-encrypted cipher text) can be sent over the Internet to the target system via the TCP/IP protocol. Once the MCU receives the instruction and data, it can proceed with the FPGA update. Low-power flash devices support Message Authentication Code (MAC), which can be used to validate data for the target device. More details are given in the "Message Authentication Code (MAC) Validation/Authentication" section on page 19-6.

Hardware Requirement

To facilitate the programming of the low-power flash families, the system must have a microprocessor (with access to the device JTAG pins) to process the programming algorithm, memory to store the programming algorithm, programming data, and the necessary programming voltage. Refer to the relevant datasheet for programming voltages.

Security

Read-Back Prevention

The low-power flash devices are designed with security in mind. Even without any security measures (such as FlashLock with AES), it is not possible to read back the programming data from a programmed device. Upon programming completion, the programming algorithm will reload the programming data into the device. The device will then use built-in circuitry to determine if it was programmed correctly.

As an additional security measure, the devices are equipped with AES decryption. AES works in two steps. The first step is to program a key into the devices in a secure or trusted programming center (such as Actel In-House Programming (IHP) center). The second step is to encrypt any programming files with the same encryption key. The encrypted programming file will only work with the devices that have the same key. The AES used in the low-power flash families is the 128-bit AES decryption engine (Rijndael algorithm).

Message Authentication Code (MAC) Validation/Authentication

As part of the AES decryption flow, the devices are equipped with a MAC validation/authentication system. MAC is an authentication tag, also called a checksum, derived by applying an on-chip authentication scheme to a STAPL file as it is loaded into the FPGA. MACs are computed and verified with the same key so they can only be verified by the intended recipient. When the MCU system receives the AES-encrypted programming data (cipher text), it can validate the data by loading it into the FPGA and performing a MAC verification prior to loading the data, via a second programming pass, into the FPGA core cells. This prevents erroneous or corrupt data from getting into the FPGA.

Low-power flash devices with AES and MAC are superior to devices with only DES or 3DES encryption. Because the MAC verifies the correctness of the data, the FPGA is protected from erroneous loading of invalid programming data that could damage a device (Figure 19-5 on page 19-7).

The AES with MAC enables field updates over public networks without fear of having the design stolen. An encrypted programming file can only work on devices with the correct key, rendering

any stolen files useless to the thief. To learn more about the low-power flash devices' security features, refer to *Security in Low-Power Flash Devices*.

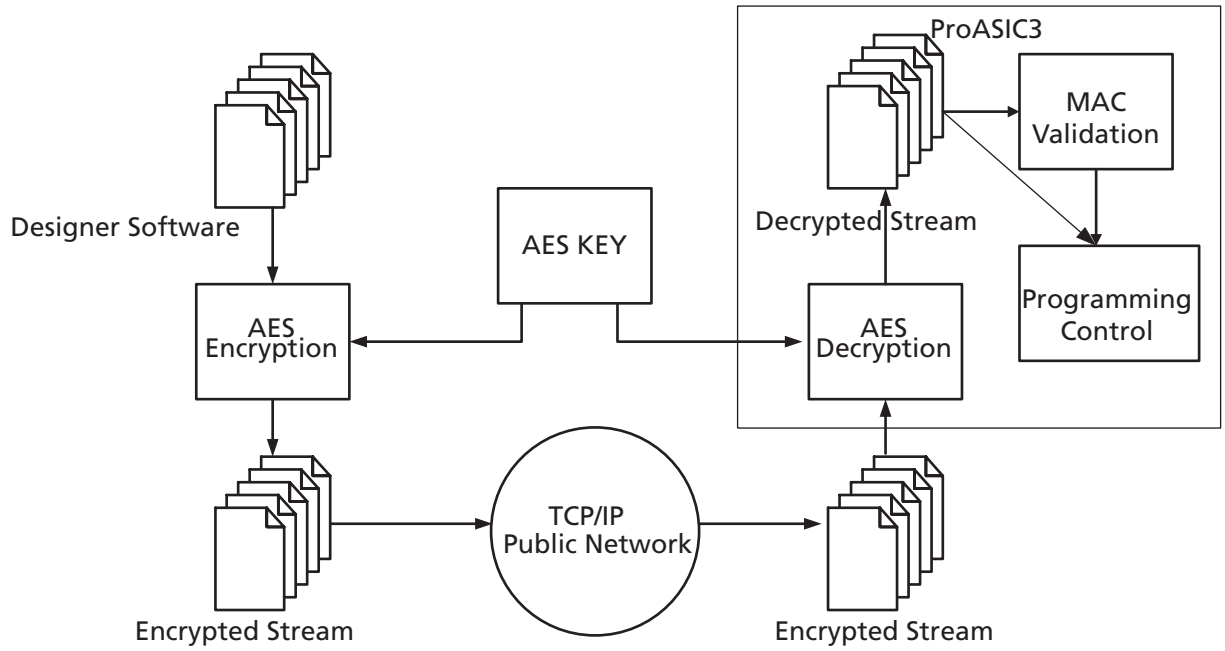


Figure 19-5 • **ProASIC3 Device Encryption Flow**

Conclusion

The Actel Fusion, IGLOO, and ProASIC3 FPGAs are ideal for applications that require field upgrades. The single-chip devices save board space by eliminating the need for EEPROM. The built-in AES with MAC enables transmission of programming data over any network without fear of design theft. Fusion, IGLOO, and ProASIC3 FPGAs are IEEE 1532–compliant and support STAPL, making the target programming software easy to implement.

Related Documents

Handbook Documents

FlashROM in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_FlashROM_HBs.pdf

Security in Low-Power Flash Devices

http://www.actel.com/documents/LPD_Security_HBs.pdf

Part Number and Revision Date

This document was previously published as an application note describing features and functions of the device, and as such has now been incorporated into the device handbook format. No technical changes have been made to the content.

Part Number 51700094-016-4

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in the Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 19-1 · Flash-Based FPGAs.	19-2
v1.2 (June 2008)	The "Microprocessor Programming Support in Flash Devices" section was revised to include new families and make the information more concise.	19-2
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 19-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	19-2
v1.0 (January 2008)	The "Microprocessor Programming Support in Flash Devices" section was updated to include information on the IGLOO PLUS family. The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	19-2

Boundary Scan and UJTAG

Actel
POWER MATTERS

20 – Boundary Scan in Low-Power Flash Devices

Boundary Scan

Low-power flash devices are compatible with IEEE Standard 1149.1, which defines a hardware architecture and the set of mechanisms for boundary scan testing. JTAG operations are used during boundary scan testing.

The basic boundary scan logic circuit is composed of the TAP controller, test data registers, and instruction register (Figure 20-2 on page 20-4).

Low-power flash devices support three types of test data registers: bypass, device identification, and boundary scan. The bypass register is selected when no other register needs to be accessed in a device. This speeds up test data transfer to other devices in a test data path. The 32-bit device identification register is a shift register with four fields (LSB, ID number, part number, and version). The boundary scan register observes and controls the state of each I/O pin. Each I/O cell has three boundary scan register cells, each with serial-in, serial-out, parallel-in, and parallel-out pins.

TAP Controller State Machine

The TAP controller is a 4-bit state machine (16 states) that operates as shown in Figure 20-1.

The 1s and 0s represent the values that must be present on TMS at a rising edge of TCK for the given state transition to occur. IR and DR indicate that the instruction register or the data register is operating in that state.

The TAP controller receives two control inputs (TMS and TCK) and generates control and clock signals for the rest of the test logic architecture. On power-up, the TAP controller enters the Test-Logic-Reset state. To guarantee a reset of the controller from any of the possible states, TMS must remain HIGH for five TCK cycles. The TRST pin can also be used to asynchronously place the TAP controller in the Test-Logic-Reset state.

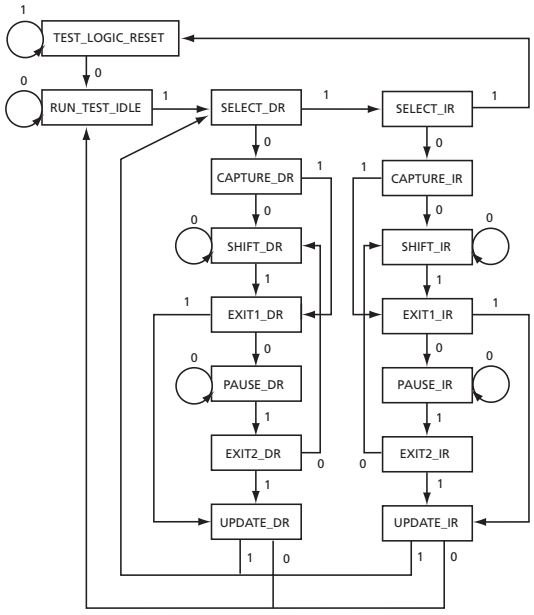


***Figure 20-1* • TAP Controller State Machine**

Actel's Flash Devices Support the JTAG Feature

The flash-based FPGAs listed in Table 20-1 support the JTAG feature and the functions described in this document.

Table 20-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO®	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC®3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 20-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 20-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

Boundary Scan Support in Low-Power Devices

The information in this document applies to all Fusion, IGLOO, and ProASIC3 devices. For IGLOO, IGLOO PLUS, and ProASIC3L devices, the Flash*Freeze pin must be deasserted for successful boundary scan operations. Devices cannot enter JTAG mode directly from Flash*Freeze mode.

Boundary Scan Opcodes

Low-power flash devices support all mandatory IEEE 1149.1 instructions (EXTEST, SAMPLE/PRELOAD, and BYPASS) and the optional IDCODE instruction (Table 20-2).

Table 20-2 • **Boundary Scan Opcodes**

	Hex Opcode
EXTEST	00
HIGHZ	07
USERCODE	0E
SAMPLE/PRELOAD	01
IDCODE	0F
CLAMP	05
BYPASS	FF

Boundary Scan Chain

The serial pins are used to serially connect all the boundary scan register cells in a device into a boundary scan register chain (Figure 20-2 on page 20-4), which starts at the TDI pin and ends at the TDO pin. The parallel ports are connected to the internal core logic I/O tile and the input, output, and control ports of an I/O buffer to capture and load data into the register to control or observe the logic state of each I/O.

Each test section is accessed through the TAP, which has five associated pins: TCK (test clock input), TDI, TDO (test data input and output), TMS (test mode selector), and TRST (test reset input). TMS, TDI, and TRST are equipped with pull-up resistors to ensure proper operation when no input data is supplied to them. These pins are dedicated for boundary scan test usage. Refer to the "JTAG Pins" description in *Pin Descriptions* for pull-up/-down recommendations for TDO and TCK pins.

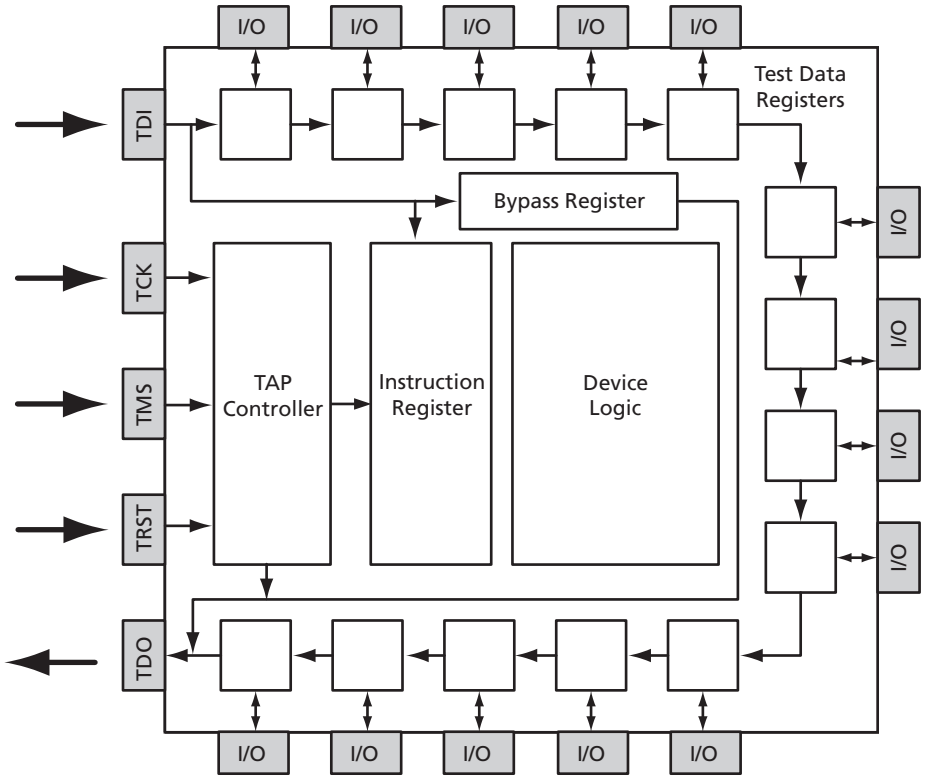


Figure 20-2 • **Boundary Scan Chain**

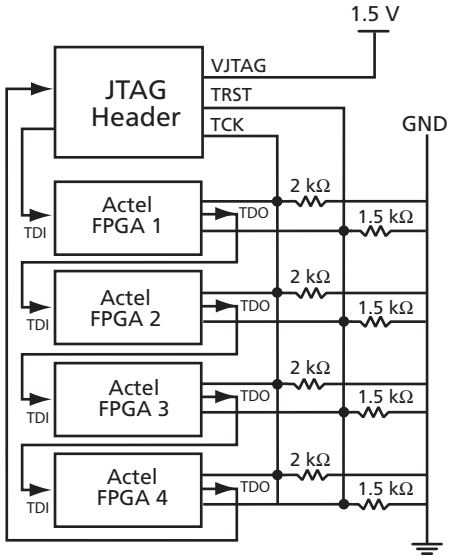
Board-Level Recommendations

Table 20-3 gives pull-down recommendations for the TRST and TCK pins.

Table 20-3 • **TRST and TCK Pull-Down Recommendations**

V_{JTAG}	Tie-Off Resistance*
V_{JTAG} at 3.3 V	200 Ω to 1 kΩ
V_{JTAG} at 2.5 V	200 Ω to 1 kΩ
V_{JTAG} at 1.8 V	500 Ω to 1 kΩ
V_{JTAG} at 1.5 V	500 Ω to 1 kΩ

* *Equivalent parallel resistance if more than one device is on JTAG chain (Figure 20-3)*



Note: TCK is correctly wired with an equivalent tie-off resistance of 500 Ω, which satisfies the table for V_{JTAG} of 1.5 V. The resistor values for TRST are not appropriate in this case, as the tie-off resistance of 375 Ω is below the recommended minimum for V_{JTAG} = 1.5 V, but would be appropriate for a V_{JTAG} setting of 2.5 V or 3.3 V.

Figure 20-3 • **Parallel Resistance on JTAG Chain of Devices**

Related Documents

Handbook Documents

Pin Descriptions

http://www.actel.com/documents/LPD_PinDescriptions_HBs.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet. To improve usability for customers, the device architecture information has now been split into handbook sections, which also include usage information. No technical changes were made to the content unless explicitly listed.

Part Number 51700094-019-3

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 20-1 · Flash-Based FPGAs.	20-2
v1.2 (June 2008)	The "Boundary Scan Support in Low-Power Devices" section was revised to include new families and make the information more concise.	20-3
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 20-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	20-2
v1.0 (January 2008)	The chapter was updated to include the IGLOO PLUS family and information regarding 15 k gate devices.	N/A
	The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	20-2

21 – UJTAG Applications in Actel's Low-Power Flash Devices

Introduction

In Fusion, IGLOO,® and ProASIC®3 devices, there is bidirectional access from the JTAG port to the core VersaTiles during normal operation of the device (Figure 21-1). User JTAG (UJTAG) is the ability for the design to use the JTAG ports for access to the device for updates, etc. While regular JTAG is used, the UJTAG tiles, located at the southeast area of the die, are directly connected to the JTAG Test Access Port (TAP) Controller in normal operating mode. As a result, all the functional blocks of the device, such as Clock Conditioning Circuits (CCCs) with PLLs, SRAM blocks, embedded FlashROM, flash memory blocks, and I/O tiles, can be reached via the JTAG ports. The UJTAG functionality is available by instantiating the UJTAG macro directly in the source code of a design. Access to the FPGA core VersaTiles from the JTAG ports enables users to implement different applications using the TAP Controller (JTAG port). This document introduces the UJTAG tile functionality and discusses a few application examples. However, the possible applications are not limited to what is presented in this document. UJTAG can serve different purposes in many designs as an elementary or auxiliary part of the design. For detailed usage information, refer to *Boundary Scan in Low-Power Flash Devices*.

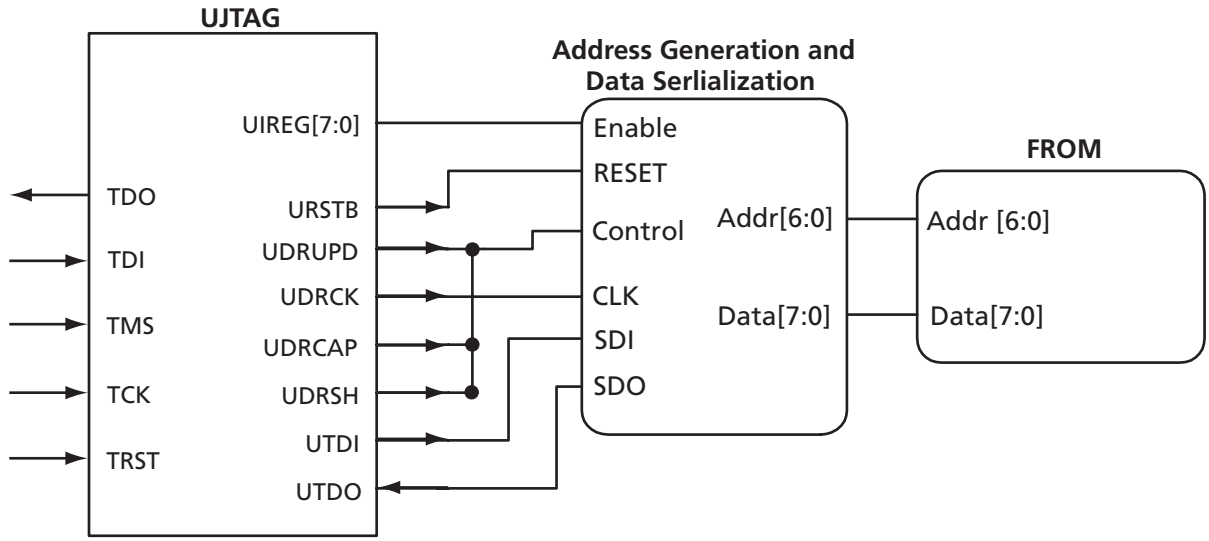


***Figure 21-1* • Block Diagram of Using UJTAG to Read FlashROM Contents**

UJTAG Support in Flash-Based Devices

The flash-based FPGAs listed in Table 21-1 support the UJTAG feature and the functions described in this document.

Table 21-1 • **Flash-Based FPGAs**

Series	Family*	Description
IGLOO	IGLOO	Ultra-low-power 1.2 V to 1.5 V FPGAs with Flash*Freeze technology
	IGLOOe	Higher density IGLOO FPGAs with six PLLs and additional I/O standards
	IGLOO nano	The industry's lowest-power, smallest-size solution
	IGLOO PLUS	IGLOO FPGAs with enhanced I/O capabilities
ProASIC3	ProASIC3	Low-power, high-performance 1.5 V FPGAs
	ProASIC3E	Higher density ProASIC3 FPGAs with six PLLs and additional I/O standards
	ProASIC3 nano	Lowest-cost solution with enhanced I/O capabilities
	ProASIC3L	ProASIC3 FPGAs supporting 1.2 V to 1.5 V with Flash*Freeze technology
	RT ProASIC3	Radiation-tolerant RT3PE600L and RT3PE3000L
	Military ProASIC3/EL	Military temperature A3PE600L, A3P1000, and A3PE3000L
	Automotive ProASIC3	ProASIC3 FPGAs qualified for automotive applications
Fusion	Fusion	Mixed-signal FPGA integrating ProASIC3 FPGA fabric, programmable analog block, support for ARM® Cortex™-M1 soft processors, and flash memory into a monolithic device

*Note: *The device names link to the appropriate datasheet, including product brief, DC and switching characteristics, and packaging information.*

IGLOO Terminology

In documentation, the terms IGLOO series and IGLOO devices refer to all of the IGLOO devices as listed in Table 21-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

ProASIC3 Terminology

In documentation, the terms ProASIC3 series and ProASIC3 devices refer to all of the ProASIC3 devices as listed in Table 21-1. Where the information applies to only one product line or limited devices, these exclusions will be explicitly stated.

To further understand the differences between the IGLOO and ProASIC3 devices, refer to the *Industry's Lowest Power FPGAs Portfolio*.

UJTAG Macro

The UJTAG tiles can be instantiated in a design using the UJTAG macro from the Fusion, IGLOO, or ProASIC3 macro library. Note that "UJTAG" is a reserved name and cannot be used for any other user-defined blocks. A block symbol of the UJTAG tile macro is presented in Figure 21-2. In this figure, the ports on the left side of the block are connected to the JTAG TAP Controller, and the right-side ports are accessible by the FPGA core VersaTiles. The TDI, TMS, TDO, TCK, and TRST ports of UJTAG are only provided for design simulation purposes and should be treated as external signals in the design netlist. However, these ports must NOT be connected to any I/O buffer in the netlist. Figure 21-3 on page 21-4 illustrates the correct connection of the UJTAG macro to the user design netlist. Actel Designer software will automatically connect these ports to the TAP during place-and-route. Table 21-2 gives the port descriptions for the rest of the UJTAG ports:

Table 21-2 • **UJTAG Port Descriptions**

Port	Description
UIREG [7:0]	This 8-bit bus carries the contents of the JTAG Instruction Register of each device. Instruction Register values 16 to 127 are not reserved and can be employed as user-defined instructions.
URSTB	URSTB is an active-low signal and will be asserted when the TAP Controller is in Test-Logic-Reset mode. URSTB is asserted at power-up, and a power-on reset signal resets the TAP Controller. URSTB will stay asserted until an external TAP access changes the TAP Controller state.
UTDI	This port is directly connected to the TAP's TDI signal.
UTDO	This port is the user TDO output. Inputs to the UTDO port are sent to the TAP TDO output MUX when the IR address is in user range.
UDRSH	Active-high signal enabled in the ShiftDR TAP state
UDRCAP	Active-high signal enabled in the CaptureDR TAP state
UDRCK	This port is directly connected to the TAP's TCK signal.
UDRUPD	Active-high signal enabled in the UpdateDR TAP state

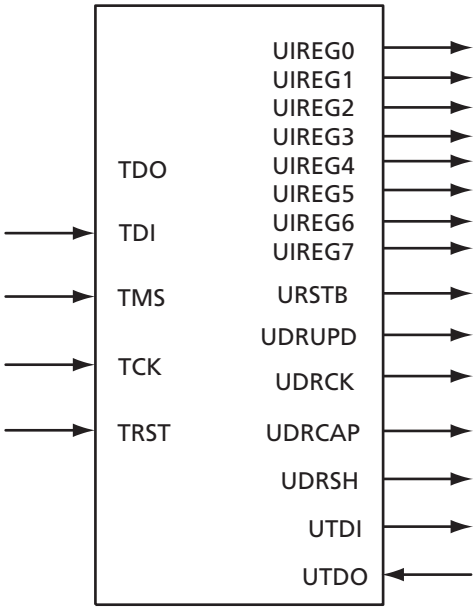
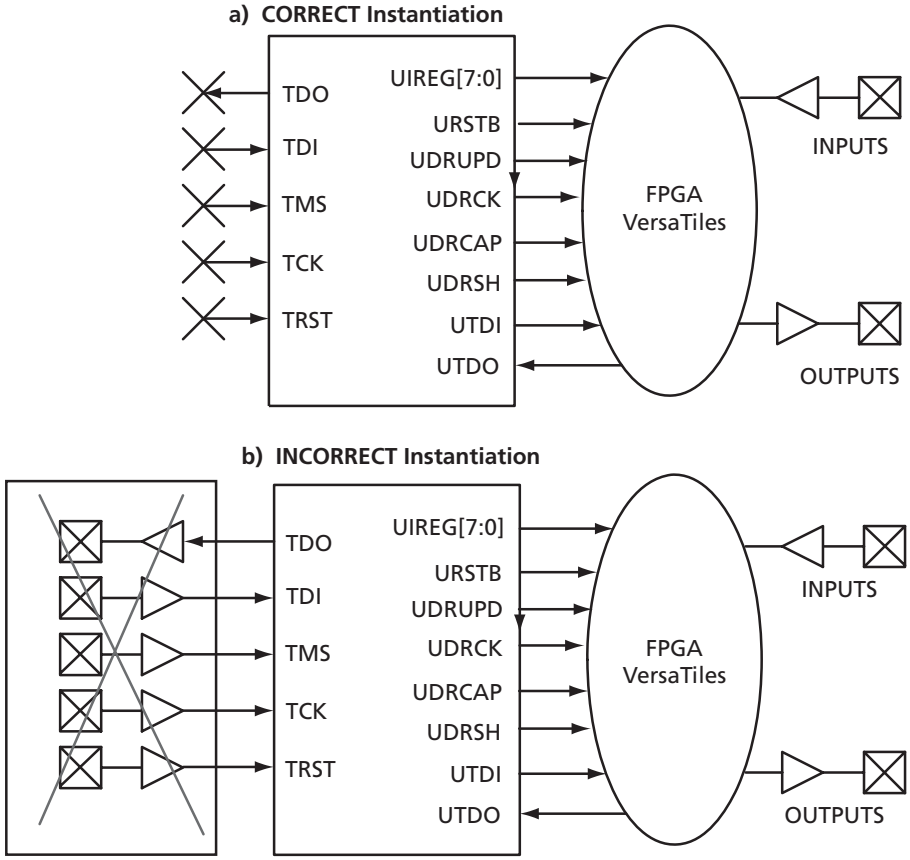


Figure 21-2 • **UJTAG Tile Block Symbol**



Note: Do not connect JTAG pins (TDO, TDI, TMS, TCK, or TRST) to I/Os in the design.

Figure 21-3 • **Connectivity Method of UJTAG Macro**

UJTAG Operation

There are a few basic functions of the UJTAG macro that users must understand before designing with it. The most important fundamental concept of the UJTAG design is its connection with the TAP Controller state machine.

TAP Controller State Machine

The 16 states of the TAP Controller state machine are shown in Figure 21-4 on page 21-5. The 1s and 0s, shown adjacent to the state transitions, represent the TMS values that must be present at the time of a rising TCK edge for a state transition to occur. In the states that include the letters "IR," the instruction register operates; in the states that contain the letters "DR," the test data register operates. The TAP Controller receives two control inputs, TMS and TCK, and generates control and clock signals for the rest of the test logic.

On power-up (or the assertion of TRST), the TAP Controller enters the Test-Logic-Reset state. To reset the controller from any other state, TMS must be held HIGH for at least five TCK cycles. After reset, the TAP state changes at the rising edge of TCK, based on the value of TMS.

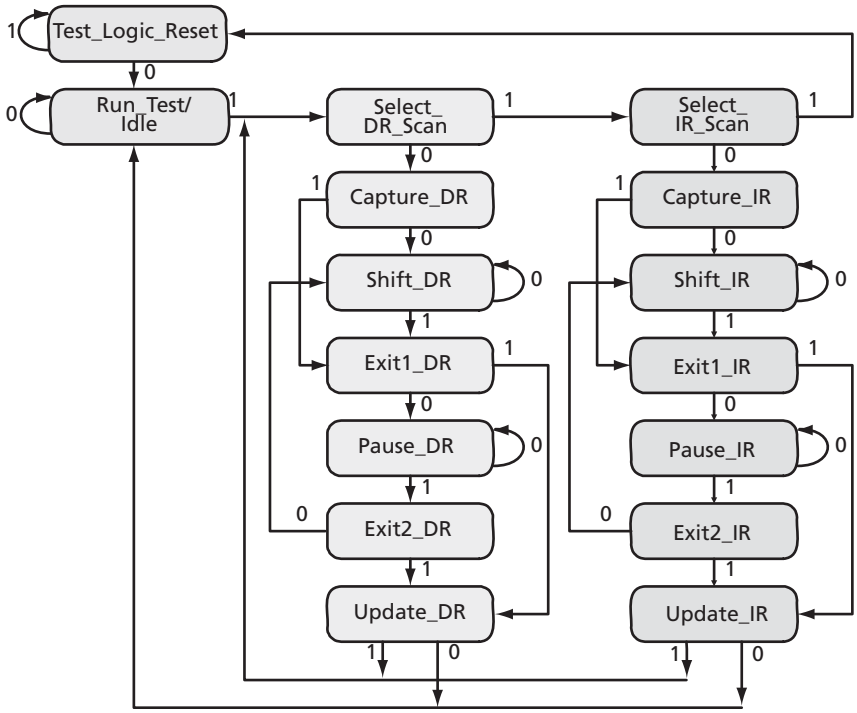


Figure 21-4 • **TAP Controller State Diagram**

UJTAG Port Usage

UIREG[7:0] hold the contents of the JTAG instruction register. The UIREG vector value is updated when the TAP Controller state machine enters the Update_IR state. Instructions 16 to 127 are user-defined and can be employed to encode multiple applications and commands within an application. Loading new instructions into the UIREG vector requires users to send appropriate logic to TMS to put the TAP Controller in a full IR cycle starting from the Select IR_Scan state and ending with the Update_IR state.

UTDI, UTDO, and UDRCK are directly connected to the JTAG TDI, TDO, and TCK ports, respectively. The TDI input can be used to provide either data (TAP Controller in the Shift_DR state) or the new contents of the instruction register (TAP Controller in the Shift_IR state).

UDRSH, UDRUPD, and UDRCAP are HIGH when the TAP Controller state machine is in the Shift_DR, Update_DR, and Capture_DR states, respectively. Therefore, they act as flags to indicate the stages of the data shift process. These flags are useful for applications in which blocks of data are shifted into the design from JTAG pins. For example, an active UDRSH can indicate that UTDI contains the data bitstream, and UDRUPD is a candidate for the end-of-data-stream flag.

As mentioned earlier, users should not connect the TDI, TDO, TCK, TMS, and TRST ports of the UJTAG macro to any port or net of the design netlist. The Designer software will automatically handle the port connection.

Typical UJTAG Applications

Bidirectional access to the JTAG port from VersaTiles—without putting the device into test mode—creates flexibility to implement many different applications. This section describes a few of these. All are based on importing/exporting data through the UJTAG tiles.

Clock Conditioning Circuitry—Dynamic Reconfiguration

In low-power flash devices, CCCs, which include PLLs, can be configured dynamically through either an 81-bit embedded shift register or static flash programming switches. These 81 bits control all the characteristics of the CCC: routing MUX architectures, delay values, divider values, etc. Table 21-3 lists the 81 configuration bits in the CCC.

Table 21-3 • **Configuration Bits of Fusion, IGLOO, and ProASIC3 CCC Blocks**

Bit Number(s)	Control Function
80	RESET ENABLE
79	DYNCSEL
78	DYNBSEL
77	DYNASEL
<76:74>	VCOSEL [2:0]
73	STATCSEL
72	STATBSEL
71	STATASEL
<70:66>	DLYC [4:0]
<65:61>	DLYB {4:0]
<60:56>	DLYGLC [4:0]
<55:51>	DLYGLB [4:0]
<50:46>	DLYGLA [4:0]
45	XDLYSEL
<44:40>	FBDLY [4:0]
<39:38>	FBSEL
<37:35>	OCMUX [2:0]
<34:32>	OBMUX [2:0]
<31:29>	OAMUX [2:0]
<28:24>	OCDIV [4:0]
<23:19>	OBDIV [4:0]
<18:14>	OADIV [4:0]
<13:7>	FBDIV [6:0]
<6:0>	FINDIV [6:0]

The embedded 81-bit shift register (for the dynamic configuration of the CCC) is accessible to the VersaTiles, which, in turn, have access to the UJTAG tiles. Therefore, the CCC configuration shift register can receive and load the new configuration data stream from JTAG.

Dynamic reconfiguration eliminates the need to reprogram the device when reconfiguration of the CCC functional blocks is needed. The CCC configuration can be modified while the device continues to operate. Employing the UJTAG core requires the user to design a module to provide the configuration data and control the CCC configuration shift register. In essence, this is a user-designed TAP Controller requiring chip resources.

Similar reconfiguration capability exists in the Actel ProASIC[PLUS]® family. The only difference is the number of shift register bits controlling the CCC (27 in ProASIC[PLUS] and 81 in IGLOO, ProASIC3, and Fusion).

Fine Tuning

In some applications, design constants or parameters need to be modified after programming the original design. The tuning process can be done using the UJTAG tile without reprogramming the device with new values. If the parameters or constants of a design are stored in distributed registers or embedded SRAM blocks, the new values can be shifted onto the JTAG TAP Controller pins, replacing the old values. The UJTAG tile is used as the "bridge" for data transfer between the JTAG pins and the FPGA VersaTiles or SRAM logic. Figure 21-5 shows a flow chart example for fine-tuning application steps using the UJTAG tile.

In Figure 21-5, the TMS signal sets the TAP Controller state machine to the appropriate states. The flow mainly consists of two steps: a) shifting the defined instruction and b) shifting the new data. If the target parameter is constantly used in the design, the new data can be shifted into a temporary shift register from UTDI. The UDRSH output of UJTAG can be used as a shift-enable signal, and UDRCK is the shift clock to the shift register. Once the shift process is completed and the TAP Controller state is moved to the Update_DR state, the UDRUPD output of the UJTAG can latch the new parameter value from the temporary register into a permanent location. This avoids any interruption or malfunctioning during the serial shift of the new value.

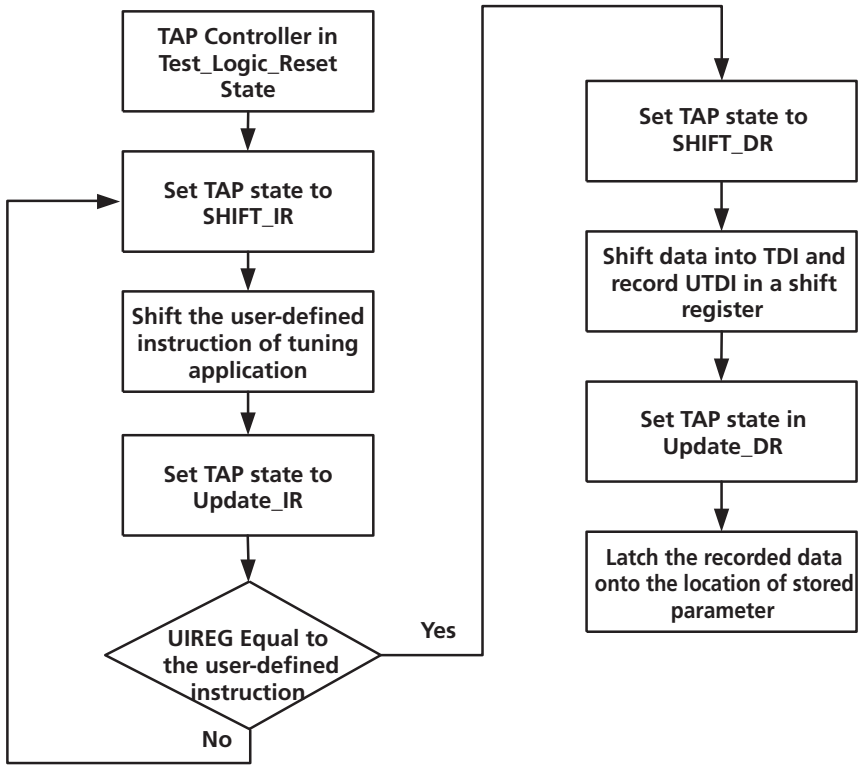


Figure 21-5 • **Flow Chart Example of Fine-Tuning an Application Using UJTAG**

Silicon Testing and Debugging

In many applications, the design needs to be tested, debugged, and verified on real silicon or in the final embedded application. To debug and test the functionality of designs, users may need to monitor some internal logic (or nets) during device operation. The approach of adding design test pins to monitor the critical internal signals has many disadvantages, such as limiting the number of user I/Os. Furthermore, adding external I/Os for test purposes may require additional or dedicated board area for testing and debugging.

The UJTAG tiles of low-power flash devices offer a flexible and cost-effective solution for silicon test and debug applications. In this solution, the signals under test are shifted out to the TDO pin of the TAP Controller. The main advantage is that all the test signals are monitored from the TDO pin; no pins or additional board-level resources are required. Figure 21-6 illustrates this technique. Multiple test nets are brought into an internal MUX architecture. The selection of the MUX is done using the contents of the TAP Controller instruction register, where individual instructions (values from 16 to 127) correspond to different signals under test. The selected test signal can be synchronized with the rising or falling edge of TCK (optional) and sent out to UTDO to drive the TDO output of JTAG.

The test and debug procedure is not limited to the example in Figure 21-5 on page 21-7. Users can customize the debug and test interface to make it appropriate for their applications. For example, multiple test signals can be registered and then sent out through UTDO, each at a different edge of TCK. In other words, *n* signals are sampled with an F_{TCK} / *n* sampling rate. The bandwidth of the information sent out to TDO is always proportional to the frequency of TCK.

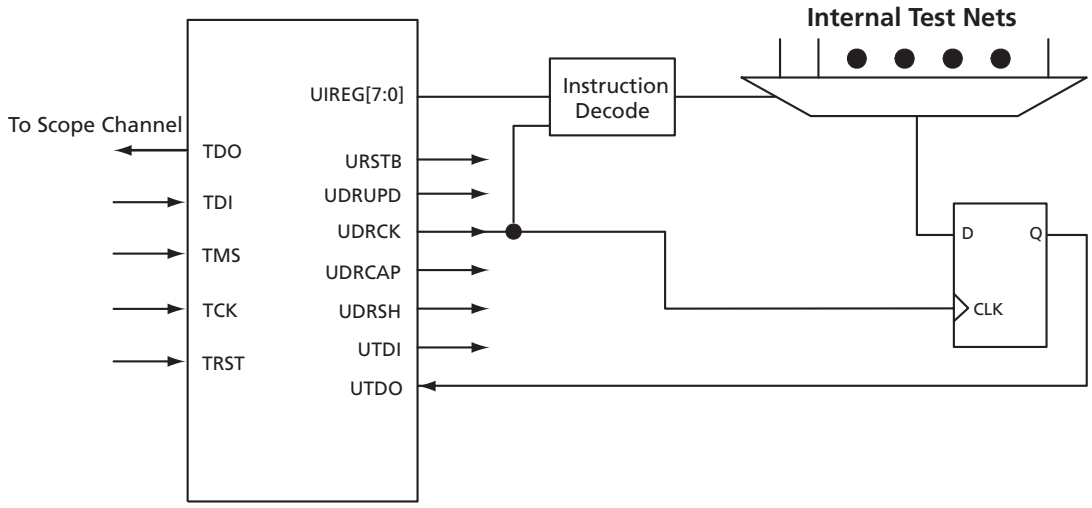


Figure 21-6 • **UJTAG Usage Example in Test and Debug Applications**

SRAM Initialization

Users can also initialize embedded SRAMs of the low-power flash devices. The initialization of the embedded SRAM blocks of the design can be done using UJTAG tiles, where the initialization data is imported using the TAP Controller. Similar functionality is available in ProASIC^{PLUS} devices using JTAG. The guidelines for implementation and design examples are given in the *RAM Initialization and ROM Emulation in ProASIC^{PLUS} Devices* application note.

SRAMs are volatile by nature; data is lost in the absence of power. Therefore, the initialization process should be done at each power-up if necessary.

FlashROM Read-Back Using JTAG

The low-power flash architecture contains a dedicated nonvolatile FlashROM block, which is formatted into eight 128-bit pages. For more information on FlashROM, refer to *FlashROM in Actel's Low-Power Flash Devices*. The contents of FlashROM are available to the VersaTiles during normal operation through a read operation. As a result, the UJTAG macro can be used to provide the FlashROM contents to the JTAG port during normal operation. Figure 21-7 illustrates a simple block diagram of using UJTAG to read the contents of FlashROM during normal operation.

The FlashROM read address can be provided from outside the FPGA through the TDI input or can be generated internally using the core logic. In either case, data serialization logic is required (Figure 21-7) and should be designed using the VersaTile core logic. FlashROM contents are read asynchronously in parallel from the flash memory and shifted out in a synchronous serial format to TDO. Shifting the serial data out of the serialization block should be performed while the TAP is in UDRSH mode. The coordination between TCK and the data shift procedure can be done using the TAP state machine by monitoring UDRSH, UDRCAP, and UDRUPD.

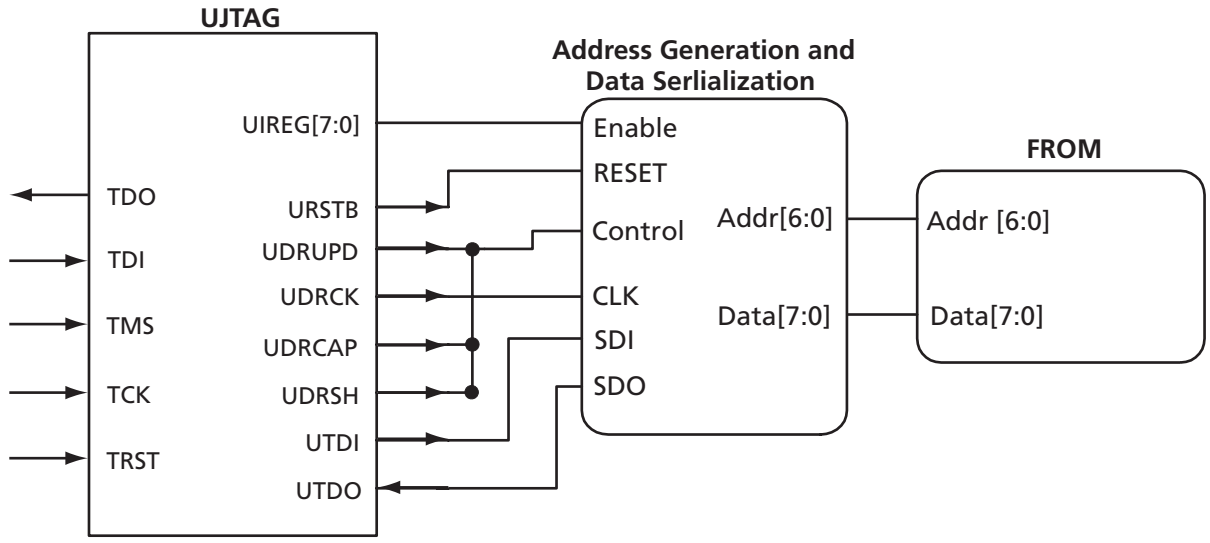


***Figure 21-7* • Block Diagram of Using UJTAG to Read FlashROM Contents**

Conclusion

Actel low-power flash FPGAs offer many unique advantages, such as security, nonvolatility, reprogrammablity, and low power—all in a single chip. In addition, Fusion, IGLOO, and ProASIC3 devices provide access to the JTAG port from core VersaTiles while the device is in normal operating mode. A wide range of available user-defined JTAG opcodes allows users to implement various types of applications, exploiting this feature of these devices. The connection between the JTAG port and core tiles is implemented through an embedded and hardwired UJTAG tile. A UJTAG tile can be instantiated in designs using the UJTAG library cell. This document presents multiple examples of UJTAG applications, such as dynamic reconfiguration, silicon test and debug, fine-tuning of the design, and RAM initialization. Each of these applications offers many useful advantages.

Related Documents

Application Notes

RAM Initialization and ROM Emulation in ProASIC^{PLUS} Devices

http://www.actel.com/documents/APA_RAM_Initd_AN.pdf

Handbook Documents

Boundary Scan in Low-Power Flash Devices

http://www.actel.com/documents/LPD_BoundaryScan_HBs.pdf

FlashROM in Actel's Low-Power Flash Devices

http://www.actel.com/documents/LPD_FlashROM_HBs.pdf

Part Number and Revision Date

This document contains content extracted from the Device Architecture section of the datasheet, combined with content previously published as an application note describing features and functions of the device. To improve usability for customers, the device architecture information has now been combined with usage information, to reduce duplication and possible inconsistencies in published information. No technical changes were made to the datasheet content unless explicitly listed. Changes to the application note content were made only to be consistent with existing datasheet information

Part Number 51700094-020-4

Revised December 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.4)	Page
v1.3 (October 2008)	IGLOO nano and ProASIC3 nano devices were added to Table 21-1 · Flash-Based FPGAs.	21-2
v1.2 (June 2008)	The "UJTAG Support in Flash-Based Devices" section was revised to include new families and make the information more concise.	21-2
	The title of Table 21-3 · Configuration Bits of Fusion, IGLOO, and ProASIC3 CCC Blocks was revised to include Fusion.	21-6
v1.1 (March 2008)	The following changes were made to the family descriptions in Table 21-1 · Flash-Based FPGAs: • ProASIC3L was updated to include 1.5 V. • The number of PLLs for ProASIC3E was changed from five to six.	21-2
v1.0 (January 2008)	The chapter was updated to include the IGLOO PLUS family and information regarding 15 k gate devices.	N/A
	The "IGLOO Terminology" section and "ProASIC3 Terminology" section are new.	21-2

Board-Level Requirements

Actel
POWER MATTERS

22 – Fusion Board-Level Design Guidelines

Objective

The successful design of Printed Circuit Boards (PCBs) incorporating the Actel Fusion® mixed-signal FPGA requires good understanding of the mixed-signal nature of the Fusion chips. Good board design practices are required to achieve the expected performance from the PCB and Fusion device and are essential to achieve high quality and reliable results, such as minimal noise levels, and adequate isolation between digital and analog domain.

This chapter presents guidelines for board-level design specific to applications using Fusion mixed-signal FPGAs. Note that these guidelines should be treated as a supplement to standard board-level design practices. This document assumes readers are experienced in digital and analog board layout and knowledgeable in the electrical characteristics of mixed-signal systems. Background information on the key theories and concepts of mixed-signal board-level design is available in *High Speed Digital Design: A Handbook of Black Magic*[1], as well as in many reference text books and literature.

Analog and Digital Plane Isolation

Since Fusion is a mixed-signal product in which both analog and digital components exist, it requires both analog and digital supply and ground planes. In addition, there are several voltage supply and ground pins on the device to power different components on the die. This section discusses the layout of the different analog or digital planes and recommends schemes to efficiently isolate different digital and analog domains from each other. This section also describes all ground and supply pins of the Fusion device, required to operate the chip, and explains how to connect them to the existing digital or analog supply or ground planes.

Placement of Fusion Device and Isolation of Ground Planes

In applications using Fusion devices, two separate grounds to the device should be provided: GND (digital ground) and GNDA (analog ground). The ground pins of the device are to be connected to one of the aforementioned ground planes appropriately, as discussed in "Isolation of Ground Planes" on page 22-3. GND is the digital ground plane that connects to all GND pins of a Fusion device. GNDA is the analog ground plane that connects to all GNDA pins of a Fusion device.

To avoid noise propagation from one plane to another (e.g. from digital to analog ground), the ground planes should be well isolated from each other. Correct layout of the ground planes on the board for current and return paths in the board will prevent the noise in one plane to affect others. For example if the return path of a digital signal trace on the board passes through the ground analog plane, the analog ground will be vulnerable to noise induced by the digital signal. Therefore, it is critical for digital traces and components on the board to be routed and placed only in the area of their corresponding layer that is covered by digital ground in the ground plane. Similar regulation should be applied to analog traces and components with respect to the analog ground as well. Figure 22-1 on page 22-2 illustrates the aforementioned regulation.

In Figure 22-1 on page 22-2, digital component C and the traces that connect to it overlap with the analog ground layout in the ground plane. This may cause some of the digital signaling current and return paths to pass through the analog domain and induce noise in this noise-sensitive domain.

Figure 22-1 on page 22-2 brings up a critical point: How a mixed-signal device, such as a Fusion device, should be placed on the board.

1. *Johnson, Howard, and Martin Graham, High Speed Digital Design: A Handbook of Black Magic. Prentice Hall PTR, 1993. ISBN-10 0133957241 or ISBN-13: 978-0133957242*

Placement of Fusion Device on Board

Fusion devices contain both analog and digital components and can interface with other digital and analog components on the board.

A Fusion device should be placed on boards such that analog signaling of the system falls within the boundaries of the analog ground and supply domain. Similarly, digital signaling of the system should fall within the boundaries of the digital domain. Figure 22-2 shows a simple illustration of the placement of a Fusion device on the board.

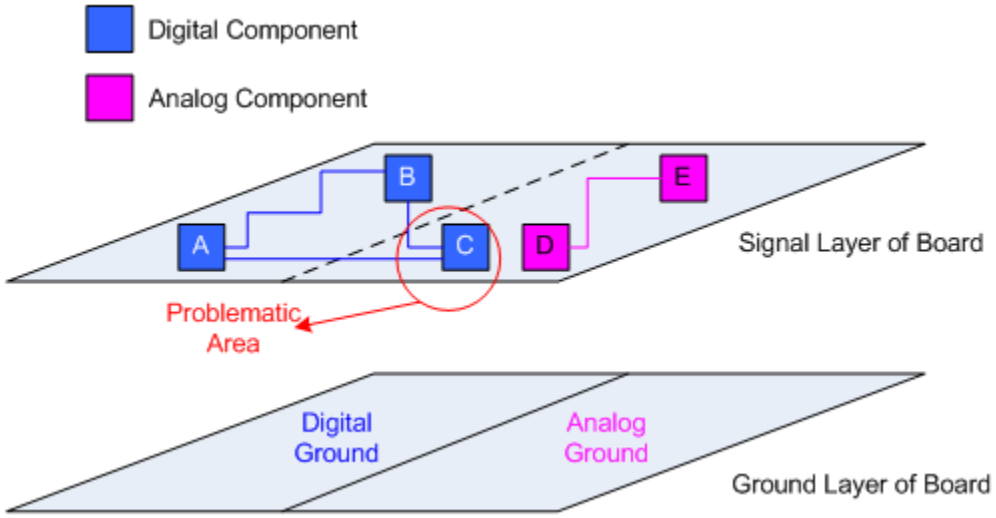


Figure 22-1 • **Illustration of Analog and Digital Components Placement on Boards**

As shown in Figure 22-2, the Fusion chip is placed on the boundary of analog and digital domains, so that the analog pins of the Fusion device are within the analog ground domain and the digital portion of the chip is placed within the digital ground domain.

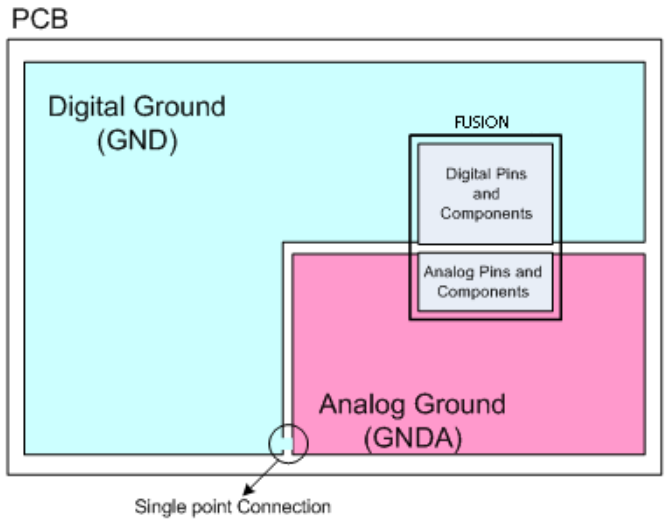


Figure 22-2 • **Simple Illustration of Fusion Device Placement on Boards**

In complicated system designs and more complicated device packages, the placement of a Fusion device may not be as straight forward as shown in the simplified diagram of Figure 22-2. However, in any board layout, it is critical to keep digital signals and their return paths well isolated from the

analog domain. The "Isolation of Ground Planes" section discusses an example of Fusion placement and ground plane layout in a real-world mixed-signal system design.

Figure 22-2 on page 22-2 also shows that the analog and digital grounds are to be connected to each other at a single point. The layout of the ground planes, as well as the power supply planes, plays a key role in reducing the noise and hence enhancing the performance and accuracy of the system.

Isolation of Ground Planes

As mentioned in "Placement of Fusion Device and Isolation of Ground Planes" on page 22-1, the ground and supply planes should be divided in two main domains: digital (e.g. GND) and analog (e.g. GNDA). Though there is no technical limitation in implementing more ground and supply domains for other necessary ground and supply pins of a Fusion device, the rest of the ground and supply pins can be connected to one of the aforementioned domains.

In order to isolate the ground of different domains from each other, the ground plane (or planes) of the board should be split into two domains: GND and GNDA, as an example. The components and signaling in each domain should remain within the boundaries of each ground as discussed in "Placement of Fusion Device and Isolation of Ground Planes" on page 22-1 and "Placement of Fusion Device on Board" on page 22-2. However, since data and control signals usually exchange between different domains, a common connection between analog and digital grounds is needed to ensure the two planes are at the same potential. Connection between two grounds should be made only through a single point as shown Figure 22-2 on page 22-2. More than a single connection point between two grounds can result in inter-domain current paths that can induce noise from one domain to another. Furthermore, the single point connection should be as far as possible from the Fusion device.

Figure 22-3 shows a real-world example of a ground plane layout and the relative placement of the Fusion chip. Refer to the "Analog and Digital Plane Isolation" section on page 22-1 for board layout recommendations.

Note: Blue = GND, Yellow = VCCI, and Green = GNDA

Figure 22-3 • **Example of Ground Plane Layout and Fusion Device Placement**

Other ground pins of the Fusion device can connect to one of the two grounds using traces on the board if necessary. However, the length of the traces should be kept as short as possible to reduce the trace inductance between ground pins and the ground plane. Table 22-1 lists all the ground pins of a Fusion device and the ground plane that they connect to.

Table 22-1 • **Ground Pin Connections to Ground Plate on Board**

Ground Pin Name	Ground Domain
GND	Digital(GND)
GNDQ	Digital(GND)
ADCGNDREF	Analog(GNDA)
GNDA	Analog(GNDA)
GNDAQ	Analog(GNDA)
GNDNVM	Digital(GND)
GNDOSC	Digital(GND)
VCOMPPLA/B*	Digital(GND)

*Note: *Older revisions of datasheets referred to a signal called VCOMPLF. This is now called VCOMPLA/B.*

Analog and Digital Voltage Supply Isolation

Digital and analog voltage supplies should be isolated from each other similar to the grounds as discussed in "Placement of Fusion Device and Isolation of Ground Planes" on page 22-1. There are three main power supplies to Fusion devices: VCC33A (3.3 V analog supply), VCC (1.5 V digital core supply), and VCCI (digital I/O supply). There may be multiple VCCI levels (for digital I/Os) since Fusion devices offer multiple I/O banks. Regardless of the number of power supply voltage levels, the layout of the board's power plans should conform to the same specifications as recommended for the ground plane in "Placement of Fusion Device and Isolation of Ground Planes" on page 22-1. None of the digital power domains should overlap with the analog power supply domain (VCC33A). This ensures that digital signaling and its return paths are well isolated form the analog power supply, minimizing noise in the analog domain. Figure 22-4 on page 22-5 shows a simple illustration of mixed-signal board layers and relative layout of the digital and analog domains.

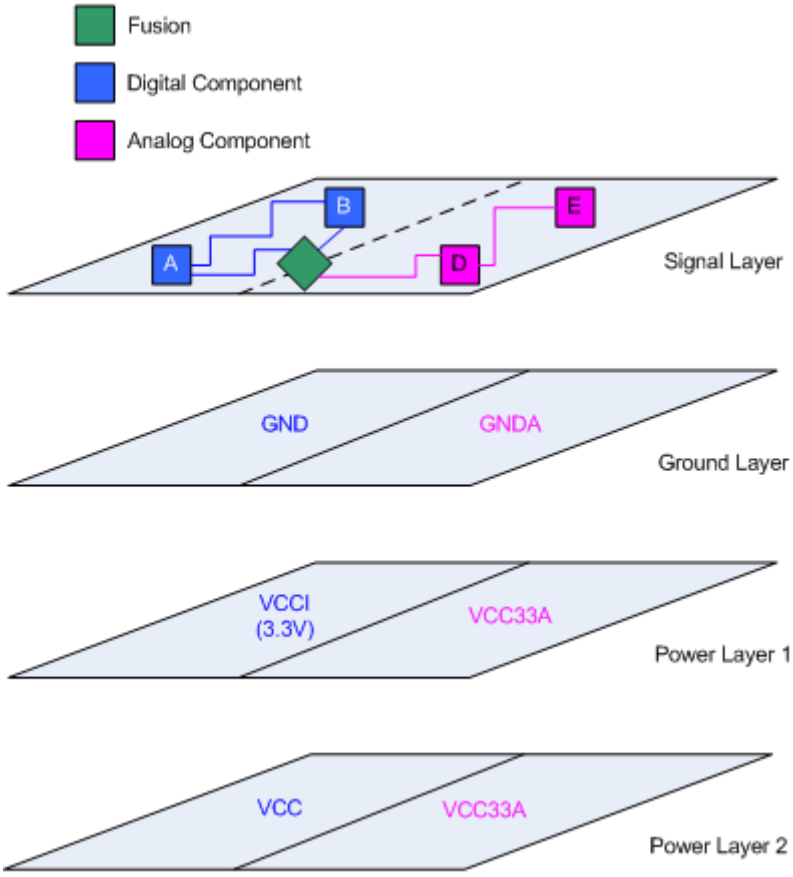


Figure 22-4 • **Simplified Illustration of a Mixed-Signal Board Stack Up**

As shown in Figure 22-4, no digital grounds or voltage supplies overlap with the analog domain. The ground plane is divided into two domains of analog and digital ground. The power planes in the Figure 22-4 board stack up follow the same layout as the ground plane. The Fusion device is placed on the boundary of the digital and analog domains as recommended in "Placement of Fusion Device and Isolation of Ground Planes" on page 22-1. Digital planes may be split if needed to accommodate additional supplies. For example, the VCCI plane can be split into 3.3 V and 2.5 V planes. The addition of another plane just to support the additional supply is typically not needed.

Additionally, Figure 22-4 emphasizes the layout of the signal traces in the signal layers of the board stack up. The digital signal traces are laid out within the digital domain and the analog traces are contained within the analog area of the layer.

Other power pins of the Fusion device can connect to one of the two domains using traces on the board. However, the length of the traces should be kept as short as possible to reduce the trace inductance between power pins and the power plane, induced by board traces, to a minimum. Figure 22-2 on page 22-2 shows that the analog and digital grounds are to be connected to each other at a single point. The same technique should be applied to digital and analog 3.3 V supplies.

Table 22-2 lists all the power pins of a Fusion device and the power plane that they connect to.

Table 22-2 • **Power Pin Connections to Power Plane on Board**

Supply Pin Name	Supply Domain
VCC	Digital (VCC)
VCC15A	Digital (VCC)
VCC33A	Analog (VCC33A)
VCC33PMP	Analog (VCC33A)
VCCNVM	Digital (VCC)
VCCOSC	Digital (3.3V)[1]
VCCIBx	Digital (VCCI)[2]
VCCPLA/B	Digital (VCC through recommended capacitors)[3]

Notes:

1. *Can be tied to any digital 3.3 V rail available in the application board (e.g. VCCIBx if the bank requires a 3.3 V supply)*
2. *If multiple banks are powered with different supply levels, different VCCI planes are needed for each voltage level*
3. *Capacitor recommendations for VCCPLA/B pins are similar to those for the ProASIC3 family device and can be found at: http://www.actel.com/documents/LPD_CCC_HBs.pdf*

Similar to any other board-level designs, decoupling/bypass capacitors or other power supply filtering techniques should be used between power supply pins and ground to reduce any potential fluctuation on the supply lines. Actel's *Board-Level Considerations* application note *(http://www.actel.com/documents/BoardLevelCons_AN.pdf*) provides additional recommendations on using decoupling capacitors. There are numerous other industry publications and guidelines available on the subject.

Other Special Function Pins

In addition to the general power and ground pins discussed in "Analog and Digital Plane Isolation" on page 22-1, there are a few other special pins that require special board considerations to ensure proper functionality of the Fusion device. This section of the document lists these pins and describes their connectivity in the board-level design.

VAREF

This pin is the voltage reference for Fusion's analog to digital converter (ADC). The Fusion device can provide a 2.56 V internal reference voltage. While using the internal reference, the reference voltage is output on VAREF for use as a system reference. If a different reference voltage is required, it can be externally supplied to VAREF and used by the ADC.

Since VAREF is the reference voltage for the ADC, it is critical for VAREF (either internal or external) to be very clean. Noise on VAREF affects the accuracy of the ADC and may cause the analog system to operate outside the specification listed in the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet. For internal VAREF use model, Actel recommends an external capacitor to be placed between VAREF and the analog ground, as close as possible to the VAREF pin. Actel recommends the capacitor value to be between 3.3 µF to 22 µF. High capacitor values (up to 22 µF) result in better noise filtering and higher ADC accuracy. However, the larger the capacitor value, the longer the rise time of VAREF at power-up. Longer time of VAREF will in turn delay the power-up time to functional of the analog block. Smaller capacitor values cause faster power-up time for VAREF, but noise filtering will be relatively less. The choice of capacitor values also depends on the total amount of noise existing on the user's board. Boards with relatively higher noise levels may need to

have capacitor values close to 22 µF and a VAREF pin, and may not perform to expectation if the capacitor values are close to 3.3 µF.

When VAREF is provided by an external source, the source must be clean to ensure the highest accuracy of the ADC. Refer to the *Fusion Family of Mixed-Signal Flash FPGAs* datasheet and the *Interfacing with the Fusion Analog System: Processor/Microcontroller Interface* chapter for more information on selecting the right capacitor value for internally generated VAREF.

VCC33N, PCAP, and NCAP

These three pins are associated with the -3.3 V charge pump. This charge pump uses two external capacitors in order to generate the -3.3 V supply. One capacitor is connected between the NCAP and PCAP pins, while the other is connected between VCC33N and the analog ground. The impulse charging of the capacitors, while the charge pump is in operation, is a source of electromagnetic interference (EMI). To reduce EMI, each of these capacitors consist of a 0.1 µF ceramic capacitor in parallel with a tantalum capacitor. The ceramic capacitors should be mounted as close as possible to the pins, using capacitors of small physical size. For the BGA package, these capacitors are to be mounted on the bottom layer, directly underneath the respective pins. The tantalum capacitors can be mounted a little further off, but users should try to minimize the distance. Ceramic capacitors are also available in higher values such as 2.2 µF. If such a capacitor is used, the 0.1 µF capacitor might not be needed.

XTAL1 and XTAL2

These pins are input from external oscillators to Fusion devices. Very slow rise and fall times of typical oscillator output (input to XTAL1 and XTAL2 pins) are much more prone to any noise induced by the system, and can result in the oscillator frequency to be misinterpreted by the Fusion device. Typical crystal oscillators generating low frequency signals, such as 32.768 KHz, typically have very slow rise and fall times (sinusoidal signal). Any small noise on the generated sine wave can result in misinterpretation of the frequency of the sine wave for the Fusion device and affect the correct functionality of the design. Therefore, for very low frequency signals, such as 32.768 KHz input to XTAL inputs of the Fusion device, Actel recommends users utilize a digital oscillator (fast rise and fall times) or to use Schmitt trigger buffers to shorten the rise and fall time of typical oscillators. For signals at 1 MHz and above, since the rise and fall times are inherently fast, a typical analog crystal oscillator can be used with no specific precaution.

For the layout and connection of the external crystal and the associated capacitors, keep stray capacitance and inductance to a minimum. It is important to keep any noise from coupling to the on-chip oscillator by way of the power supply, the crystal, the two load capacitors, or the copper traces used to connect these components. It is also important to prevent noise from coupling from the oscillator into the analog power supply, affecting the performance and accuracy of other analog circuitry. The following guidelines help achieve these objectives:

- The oscillator power supply pins should be decoupled by a 0.1 µF capacitor connected as close as possible between the VCCOSC and GNDOSC pins of the Fusion device. For a BGA package, this capacitor can be placed on the bottom layer; and if it is size 402, it fits between the pins.
- The crystal should be placed as close as possible to the XTAL1 and XTAL2 pins.
- The spacing between traces connecting crystal to XTAL1 and XTAL2 pins and nearby traces should be increased beyond the minimum spacing dictated by the PCB design rules to prevent any noise from coupling into these traces. In addition, copper traces carrying high speed digital signals should not be routed in parallel to the copper traces connected to the XTAL1 and XTAL2 pins, either on the same layer or on the other layers.
- To reduce electromagnetic emissions and provide good mechanical stability to the crystal, a copper pad slightly larger than the crystal and grounded to GNDOSC should be placed on the top layer of the PCB. The metal package of the crystal should be grounded to this pad with a suitable clip. Copper traces connected to this grounded pad and extending around the copper traces leading from the crystal to XTAL1 and XTAL2 pins shield these pins and further increase noise immunity of the oscillator. The shields add a very small amount of stray capacitance and this can be accounted for in the selection of the load capacitors.

Application-Specific Recommendations

This section of the document discusses some recommendations that are specific to temperature, voltage, or current monitoring applications. These recommendations are merely for improving the accuracy of the applications.

Temperature Monitor

The temperature monitor generates a voltage of about 2.5 mV/K (per degree Kelvin), as seen by the ADC. However, the voltage change that appears across the external discreet bipolar transistor may be much smaller. Such low levels mean that precautions should be taken to not couple noisy signals to the conductors connecting the transistor to the temperature monitor pins.

If the temperature sensing the diode/transistor is connected to an Actel Fusion device through cables, Actel recommends using a twin lead shielded cable to carry the AT and ATRTN traces with the shield of the cable grounded at the board.

If the connections are made by copper traces on the PCB, AT and ATRTN traces should be routed in such a way that traces carrying digital signal are not parallel to them above, below, or on the sides. To achieve this, lay the AT and ATRTN traces on the top layer, so that the next adjacent layer in the PCB stack is the ground layer. This provides for shielding against digital signals that can couple to the signals on the copper traces connected to the AT and ATRTN pins.

If digital signal carrying traces can not be avoided in the vicinity of the traces connecting to the transistor, sufficient distance is to be created between the offending trace and the AT or ATRTN traces.

It is important to minimize the resistance of the conductors connecting the external discrete bipolar transistor to the AT and ATRTN pins of the Fusion chip. If PCB copper traces are used as the interconnecting conductors, they should be of such a width that, taking into account their length, they contribute only a negligible voltage drop compared to 200 µV. The current through the bipolar transistor used for sensing the temperature changes by 90 µA during the measurement process. This current, multiplied by the total resistance of the copper trace from the AT pin to the transistor and from the transistor back to the ATRTN pin, should be negligible compared to 200 µV. If a shielded cable is used, the wire gauge of its conductors should be appropriately selected. If the system using the Actel Fusion devices is to be operated at other than room temperature, the effect of temperature on the resistance of the wire or copper traces should also be taken into account.

Voltage and Current Monitor

If any of the AV channels are used in the direct mode that is directly connecting to the ADC without prescalers, it is recommend that a ceramic capacitor of the NPO or COG variety, or better yet, a polyester capacitor of 2200 pF be placed from the corresponding AV channel pin to the analog ground, and as close as possible to the AV pin.

A resistor of 100 Ω should then be connected between the AV pin and whatever point is being monitored by the particular AV channel. If the accuracy requirements are not stringent, one may be able to get by without using the above mentioned resistor/capacitor combination. However, it is good practice to at least make provision for these components on the prototype PCB. The ADC is a switched capacitor design and needs to be driven from a low impedance. It draws a charging current every time a channel is sampled, and the capacitor helps to maintain the voltage steady at the particular AV pin during such intervals. All copper traces connecting to the AV or the AC pins should stay within the area covered by the analog ground plane. The power for the ADC, voltage and current monitors, and the internal voltage reference is provided from the same pins. These pins are to be adequately decoupled with 0.1 µF ceramic X7R dielectric capacitors in parallel with a tantalum capacitor of 22 µF capacity.

In applications using current monitor, it is important to route the AV and AC signals of each channel in parallel and keep the two traces matched as much as possible. Large differences in the nets bringing AV and AC signals to the device may cause significant inaccuracy in differential voltage across the AV and AC pin.

In current monitor applications, the current sense resistor should be chosen carefully so that optimal accuracy and resolution can be achieved. The *Fusion Family of Mixed-Signal Flash FPGAs* datasheet describes the recommended resistor values for various current ranges.

Connection to PLL

Table 22-1 on page 22-4 and Table 22-2 on page 22-6 describe the connection of the VCCPLA/B and VCOMPLA/B pins of the Fusion device to the power and ground planes. This section of the document discusses how these pins and the dedicated clock pins of the Fusion device connect to the PLLs on the chip. Connecting external signals into PLL and powering them up should be done considering that AFS090 and AFS250 devices contain only one PLL, while AFS600 and AFS1500 devices contain two PLL blocks. In AFS090 and AFS250 devices, the PLL is located on the west side of the die. In devices with two PLLs, the second PLL is placed on east side of the die[2].

Table 22-3 shows the corresponding power and ground pins for each PLL block

Table 22-3 • **Power and Ground Pin Names For Fusion Device PLLs**

PLL/Device	AFS090	AFS250	AFS600	AFS1500
West PLL	VCCPLA/VCOMPLA	VCCPLA/VCOMPLA	VCCPLA/VCOMPLA	VCCPLA/VCOMPLA
East PLL	–	–	VCCPLB/VCOMPLB	VCCPLB/VCOMPLB

In addition to hardwire clock pins, Fusion device PLLs can be driven by any internal net or external I/O pins. Although the hardwire I/Os can be used as any user I/O, if designers are required to minimize the propagation from external clock to the PLL, hardwire clock pins of the PLL provide the shortest paths from board to PLL clock input. Table 22-4 lists the hardwire clock pins for each PLL on the device.

Table 22-4 • **Hardwire Clock Pin Connections to PLL**

PLL/Device	AFS090	AFS250	AFS600	AFS1500
West PLL	GFA0/GFA1/GFA2*	GFA0/GFA1/GFA2*	GFA0/GFA1/GFA2	GFA0/GFA1/GFA2
East PLL	–	–	GCA0/GCA1/GCA2	GCA0/GCA1/GCA2

*Note: * Depending on the selected package, not all three hardwire clock I/Os may be available.*

2. *Refer to Fusion Family of Mixed-Signal Flash FPGAs datasheet for more information on Clock Conditioning Circuits and location of PLLs.*

Part Number and Revision Date

Part Number 51700092-012-1
Revised October 2008

List of Changes

The following table lists critical changes that were made in the current version of the chapter.

Previous Version	Changes in Current Version (v1.1)	Page
v1.0 (August 2008)	The "Temperature Monitor" section was revised to remove information about capacitors.	22-8

Software Design Tutorials and Examples

Actel®
POWER MATTERS

23 – Fusion Solutions, Design Examples, and Reference Designs

The unprecedented level of integration of Actel Fusion® enables a wide variety of functionality, such as power and thermal management, remote communications, and system clocking, in a single mixed-signal FPGA.

Actel, the world leader of mixed-signal FPGAs, now offers the only single-chip system management solution. The Actel Fusion mixed-signal FPGA integrates configurable analog, large flash memory blocks, comprehensive clock generation and management circuitry, and high-performance programmable logic in a monolithic device. Actel has developed turn-key solutions, including a development kit and a software GUI, for system management. This level of integration, configurability, and support establishes Fusion as the definitive system management solution.

This chapter captures the design examples and reference design information for Fusion applications in various aspects.

System Management Applications

System management continues to gain importance in the design of all electronic systems. Smaller process geometries drive more multivolt devices and are more susceptible to voltage and temperature fluctuations. Whereas system management designs can run into hundreds of discrete components, the Actel Fusion mixed-signal FPGA solution can integrate these system management functions and provide programmable flexibility and system-level integration—all in a single chip. Unprecedented integration in Fusion devices can offer cost and space savings of 50% or greater relative to current implementations.

White paper*: System Management Using a Mixed-Signal FPGA*

Power Management

- Control up to 10 power supplies
  - Voltage monitoring
  - Current monitoring
- Power-on detection and reset
- Custom power mode topology support for total system power reduction
- Power-up sequencing

Power Supply Monitoring

Application note: *Multi-Channel Analog Voltage Comparator in Fusion FPGAs*

- VHDL Design Files
- Verilog Design Files

Power Sequencing

Application note: *Fusion Power Sequencing and Ramp-Rate Control*

- Design Files

Motor Control

An obstacle to wide-scale use of electronic motor control has been the high cost of the computer and power electronics. This obstacle is diminishing due to tremendous technology improvement in semiconductor processes and integration. The Actel Fusion mixed-signal FPGA offers unprecedented integration by combining mixed-signal analog, flash memory, and FPGA fabric in a monolithic PSC. This means that for the first time, engineers can combine the motor control analog front end, high-speed flash lookup tables, and deterministic algorithm processing capabilities of programmable logic in a single-chip solution.

Reference board: Motor Control Reference Board

Application note: *PID Control*

MicroTCA

Actel provides highly integrated solutions for MicroTCA system management. Actel supports Fusion-based solutions with reference designs, semiconductor intellectual property (IP), software, and customization services that enable quicker time to market for Actel customers with reduced risk, lower costs, and improved availability over existing solutions. The high integration of Actel MicroTCA solutions also provides increased functionality in a fixed form factor.

Nearly every electronic system needs system management, especially those with telecommunications-driven standards like MicroTCA, ATCA, and AMC, as well as those with server-driven standards like IPMI. Fusion and ProASIC®3 are adept at supporting both proprietary and standards-based implementations.

Application note: *Actel Fusion FPGAs Supporting Intelligent Peripheral Management Interface (IPMI) Applications*

Application note: *MicroTCA*

Reference design: MicroTCA Power Module Reference Design

Other Applications

Single chip, live at power-up, embedded flash memory, and low power make Fusion suitable for many other applications.

Application note: *Context Save and Reload*

- Design Files

Application note: *Real-Time Calendar Applications in Actel Fusion Devices*

- Design Files

Technical brief: *Using Fusion FIFO for Generating Periodic Waveforms*

- Fusion Sine Table

Application note: *Using Fusion RAM as Multipliers*

Application note: *Configuring CorePWM Using RTL Blocks*

- Verilog Design Files
- VHDL Design Files

Application brief: *Smart Battery Management Applications*

Development System

Fusion Starter Kit

The Fusion Starter Kit is an all-inclusive, low-cost evaluation kit for the Actel Fusion family. Users can utilize this starter kit to explore the voltage, current, and temperature monitor, real-time counter for low power use model, and embedded flash for context-saving applications.

User's guide: *Fusion Starter Kit User's Guide & Tutorial*

- Design Files for Fusion Starter Kit Tutorial

System Management Development Kit

The Actel System Management Development Kit provides an excellent platform for developing system management applications and applications with a microprocessor. The kit includes an ARM-enabled Fusion device, a system management GUI, and a platform for systems that performs these functions:

- Power-up detection
- Power sequencing
- Thermal management
- Sleep modes
- System diagnostics
- Remote communications
- Clock generation and management
- Data logging

User's guide: *System Management Board User's Guide*

Part Number and Revision Date

Part Number 51700092-009-0
Revised November 2007

24 – Fusion Glossary

AB • Analog Block

ACM • Analog Configuration Multiplexer. Stores configuration data for the Analog Quads.

ACMCLK • The clock input to the ACM used for configuration/initialization of the Analog Quads: 10 MHz max.

ADC • Analog-to-digital converter

ADCCLK • The clock input to the AB used by the ADC: 10 MHz max.

ADCSTART • Request to the ADC to begin sampling and convert a defined channel

AHB • Advanced High-Performance Bus. The AHB sits above the APB in the AMBA architecture and implements the features required for high-performance, high-clock-frequency systems, including burst transfers, split transactions, single-cycle bus master handover, single-clock-edge operation, non-tristate implementation, and wider data bus configurations (64/128 bits).

AMBA • Advanced Microcontroller Bus Architecture. The AMBA protocol is an open-standard, on-chip bus specification that details a strategy for the interconnection and management of functional blocks.

APB • Advanced Peripheral Bus. As part of the AMBA architecture, the APB is optimized for reduced interface complexity and is used to interface with low-bandwidth peripherals.

ASB • Analog System Builder

ASSC • ADC Sample Sequence Controller

CCC • Clock Conditioning Circuit, which may include a PLL

CoreABC • AMBA Bus Controller soft IP

CoreAI • Analog Interface IP for microprocessors/-controllers

CoreConsole IP Deployment Platform (IDP) • IDP enabling users to construct a processor subsystem and assemble IP blocks within a design

CTRL_STAT • 8-bit register located within the ACM that defines the operation of the RTC

Designer • Actel's place-and-route tool, which allows users to assign I/Os, evaluate timing, and generate programming files

FPGAGOOD • Signal from VRPSM indicating the internal voltage regulator is on

INITCLK • Initialization clock used in the initialization client: 10 MHz max.

Libero® Integrated Design Environment (IDE) • Actel's FPGA design environment, integrating synthesis, simulation, place-and-route, design entry, and IP generation tools

MATCH • Signal indicating that RTC count register (COUNTER) matches MATCHREG

MATCHREG • 40-bit register containing a user-defined value to be compared with RTC count register

PCLK • APB interface clock

PUB • Power Up (Bar). FPGA input with weak internal pull-up used to force power-up of internal voltage regulator

PUCORE • Inverse of PUB

RTC • Real-Time Counter

RTCPSMMATCH • Match signal from RTC that is passed to VRPSM to power up voltage regulator

SmartTime • Actel's static timing analysis tool, integrated into Designer

SMEV • System Monitor Evaluation Phase State Machine

SMTR • System Monitor Transition Phase State Machine

STC • Sample Time Control setting for the acquisition time in ASB

TVC • 8-bit user-configurable register that determines the division value required to bring the system clock to less that 10 MHz for the ADC

VRINITSTATE • Defines the voltage regulator state upon power-up

VRPSM • Voltage Regulator Power Supply Monitor

VRPU • Voltage Regulator Power-Up. Internal, user-accessible signal to turn off internal voltage regulator from FPGA core

Part Number and Revision Date

Part Number 51700092-011-0

Revised November 2007

Actel, IGLOO, Actel Fusion, ProASIC, Libero, Pigeon Point and the associated logos are trademarks or registered trademarks of Actel Corporation. All other trademarks and service marks are the property of their respective owners.

Actel is the leader in low-power and mixed-signal FPGAs and offers the most comprehensive portfolio of system and power management solutions. Power Matters. Learn more at www.actel.com.

Actel Corporation
2061 Stierlin Court
Mountain View, CA
94043-4655 USA
Phone 650.318.4200
Fax 650.318.4600

Actel Europe Ltd.
River Court,Meadows Business Park
Station Approach, Blackwater
Camberley Surrey GU17 9AB
United Kingdom
Phone +44 (0) 1276 609 300
Fax +44 (0) 1276 607 540

Actel Japan
EXOS Ebisu Buillding 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150 Japan
Phone +81.03.3445.7671
Fax +81.03.3445.7668
http://jp.actel.com

Actel Hong Kong
Room 2107, China Resources Building
26 Harbour Road
Wanchai, Hong Kong
Phone +852 2185 6460
Fax +852 2185 6488
www.actel.com.cn

51700092-017-4/8.09