Stratix Device Handbook, Volume 1

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com

S5V1-3.4

Copyright © 2006 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

I.S. EN ISO 9001

Contents

Chapter Revision Dates vii

About This Handbook ix
How to Find Information ix
How to Contact Altera ix
Typographic Conventions x

Section I. Stratix Device Family Data Sheet
Revision History ..... Part I–1

Chapter 1. Introduction
Introduction 1–1
Features 1–2

Chapter 2. Stratix Architecture
Functional Description 2–1
Logic Array Blocks 2–3
LAB Interconnects 2–4
LAB Control Signals 2–5
Logic Elements 2–6
LUT Chain & Register Chain 2–8
addnsub Signal 2–8
LE Operating Modes 2–8
Clear & Preset Logic Control 2–13
MultiTrack Interconnect 2–14
TriMatrix Memory 2–21
Memory Modes 2–22
Clear Signals 2–24
Parity Bit Support 2–24
Shift Register Support 2–25
Memory Block Size 2–26
Independent Clock Mode 2–44
Input/Output Clock Mode 2–46
Read/Write Clock Mode 2–49
Single-Port Mode 2–51
Multiplier Block 2–57
Adder/Output Blocks 2–61
Modes of Operation 2–64

DSP Block Interface 2–70
PLLs & Clock Networks 2–73
Global & Hierarchical Clocking 2–73
Enhanced & Fast PLLs 2–81
Enhanced PLLs 2–87
Fast PLLs 2–100
I/O Structure 2–104
Double-Data Rate I/O Pins 2–111
External RAM Interfacing 2–115
Programmable Drive Strength 2–119
Open-Drain Output 2–120
Slew-Rate Control 2–120
Bus Hold 2–121
Programmable Pull-Up Resistor 2–122
Advanced I/O Standard Support 2–122
Differential On-Chip Termination 2–127
MultiVolt I/O Interface 2–129
High-Speed Differential I/O Support 2–130
Dedicated Circuitry 2–137
Byte Alignment 2–140
Power Sequencing & Hot Socketing 2–140

Chapter 3. Configuration & Testing

IEEE Std. 1149.1 (JTAG) Boundary-Scan Support 3–1
SignalTap II Embedded Logic Analyzer 3–5
Configuration 3–5
Operating Modes 3–5
Configuring Stratix FPGAs with JRunner 3–7
Configuration Schemes 3–7
Partial Reconfiguration 3–7
Remote Update Configuration Modes 3–8
Stratix Automated Single Event Upset (SEU) Detection 3–12
Custom-Built Circuitry 3–13
Software Interface 3–13
Temperature Sensing Diode 3–13

Chapter 4. DC & Switching Characteristics

Operating Conditions 4–1
Power Consumption 4–17
Timing Model 4–19
Preliminary & Final Timing 4–19
Performance 4–20
Internal Timing Parameters 4–22
External Timing Parameters 4–33
Stratix External I/O Timing 4–36
I/O Timing Measurement Methodology 4–60
External I/O Delay Parameters 4–66

Maximum Input & Output Clock Rates 4–76
High-Speed I/O Specification 4–87
PLL Specifications 4–94
DLL Specifications 4–102

Chapter 5. Reference & Ordering Information

Software 5–1
Device Pin-Outs 5–1
Ordering Information 5–1

Index

Chapter Revision Dates

The chapters in this book, *Stratix Device Handbook, Volume 1*, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1. Introduction
Revised: *July 2005*
Part number: *S51001-3.2*

Chapter 2. Stratix Architecture
Revised: *July 2005*
Part number: *S51002-3.2*

Chapter 3. Configuration & Testing
Revised: *July 2005*
Part number: *S51003-1.3*

Chapter 4. DC & Switching Characteristics
Revised: *January 2006*
Part number: *S51004-3.4*

Chapter 5. Reference & Ordering Information
Revised: *September 2004*
Part number: *S51005-2.1*

About This Handbook

This handbook provides comprehensive information about the Altera® Stratix family of devices.

How to Find Information

You can find more information in the following ways:

- The Adobe Acrobat Find feature, which searches the text of a PDF document. Click the binoculars toolbar icon to open the Find dialog box.
- Acrobat bookmarks, which serve as an additional table of contents in PDF documents.
- Thumbnail icons, which provide miniature previews of each page, provide a link to the pages.
- Numerous links, shown in green text, which allow you to jump to related information.

How to Contact Altera

For the most up-to-date information about Altera products, go to the Altera world-wide web site at www.altera.com. For technical support on this product, go to www.altera.com/mysupport. For additional information about Altera products, consult the sources shown below.

Information Type	USA & Canada	All Other Locations
Technical support	www.altera.com/mysupport/	www.altera.com/mysupport/
	(800) 800-EPLD (3753) (7:00 a.m. to 5:00 p.m. Pacific Time)	+1 408-544-8767 7:00 a.m. to 5:00 p.m. (GMT -8:00) Pacific Time
Product literature	www.altera.com	www.altera.com
Altera literature services	literature@altera.com	literature@altera.com
Non-technical customer service	(800) 767-3753	+ 1 408-544-7000 7:00 a.m. to 5:00 p.m. (GMT -8:00) Pacific Time
FTP site	ftp.altera.com	ftp.altera.com

Typographic Conventions

This document uses the typographic conventions shown below.

Visual Cue	Meaning
Bold Type with Initial Capital Letters	Command names, dialog box titles, checkbox options, and dialog box options are shown in bold, initial capital letters. Example: **Save As** dialog box.
bold type	External timing parameters, directory names, project names, disk drive names, filenames, filename extensions, and software utility names are shown in bold type. Examples: $\mathbf{f_{MAX}}$, **\qdesigns** directory, **d:** drive, **chiptrip.gdf** file.
Italic Type with Initial Capital Letters	Document titles are shown in italic type with initial capital letters. Example: *AN 75: High-Speed Board Designs.*
Italic type	Internal timing parameters and variables are shown in italic type. Examples: t_{PIA}, $n + 1$. Variable names are enclosed in angle brackets (< >) and shown in italic type. Example: *<file name>*, *<project name>***.pof** file.
Initial Capital Letters	Keyboard keys and menu names are shown with initial capital letters. Examples: Delete key, the Options menu.
“Subheading Title”	References to sections within a document and titles of on-line help topics are shown in quotation marks. Example: “Typographic Conventions.”
`Courier type`	Signal and port names are shown in lowercase Courier type. Examples: `data1`, `tdi`, `input.` Active-low signals are denoted by suffix `n`, e.g., `resetn`. Anything that must be typed exactly as it appears is shown in Courier type. For example: `c:\qdesigns\tutorial\chiptrip.gdf`. Also, sections of an actual file, such as a Report File, references to parts of files (e.g., the AHDL keyword `SUBDESIGN`), as well as logic function names (e.g., `TRI`) are shown in Courier.
1., 2., 3., and a., b., c., etc.	Numbered steps are used in a list of items when the sequence of the items is important, such as the steps listed in a procedure.
■ ● •	Bullets are used in a list of items when the sequence of the items is not important.
v	The checkmark indicates a procedure that consists of one step only.
1	The hand points to information that requires special attention.
r	The angled arrow indicates you should press the Enter key.
f	The feet direct you to more information on a particular topic.

Section I. Stratix Device Family Data Sheet

This section provides the data sheet specifications for Stratix® devices. They contain feature definitions of the internal architecture, configuration and JTAG boundary-scan testing information, DC operating conditions, AC timing parameters, a reference to power consumption, and ordering information for Stratix devices.

This section contains the following chapters:

- Chapter 1, Introduction
- Chapter 2, Stratix Architecture
- Chapter 3, Configuration & Testing
- Chapter 4, DC & Switching Characteristics
- Chapter 5, Reference & Ordering Information

Revision History

The table below shows the revision history for Chapters 1 through 5.

Chapter	Date/Version	Changes Made
1	July 2005, v3.2	• Minor content changes.
	September 2004, v3.1	• Updated Table 1–6 on page 1–5.
	April 2004, v3.0	• Main section page numbers changed on first page. • Changed PCI-X to PCI-X 1.0 in "Features" on page 1–2. • Global change from SignalTap to SignalTap II. • The DSP blocks in "Features" on page 1–2 provide dedicated implementation of multipliers that are now "faster than 300 MHz."
	January 2004, v2.2	• Updated -5 speed grade device information in Table 1-6.
	October 2003, v2.1	• Add -8 speed grade device information.
	July 2003, v2.0	• Format changes throughout chapter.

Chapter	Date/Version	Changes Made
2	July 2005 v3.2	• Added "Clear Signals" section. • Updated "Power Sequencing & Hot Socketing" section. • Format changes.
	September 2004, v3.1	• Updated fast regional clock networks description on page 2–73. • Deleted the word preliminary from the "specification for the maximum time to relock is 100 µs" on page 2–90. • Added information about differential SSTL and HSTL outputs in "External Clock Outputs" on page 2–92. • Updated notes in Figure 2–55 on page 2–93. • Added information about *m* counter to "Clock Multiplication & Division" on page 2–101. • Updated Note 1 in Table 2–58 on page 2–101. • Updated description of "Clock Multiplication & Division" on page 2–88. • Updated Table 2–22 on page 2–102. • Added references to AN 349 and AN 329 to "External RAM Interfacing" on page 2–115. • Table 2–25 on page 2–116: updated the table, updated Notes 3 and 4. Notes 4, 5, and 6, are now Notes 5, 6, and 7, respectively. • Updated Table 2–26 on page 2–117. • Added information about PCI Compliance to page 2–120. • Table 2–32 on page 2–126: updated the table and deleted Note 1. • Updated reference to device pin-outs now being available on the web on page 2–130. • Added Notes 4 and 5 to Table 2–36 on page 2–130. • Updated Note 3 in Table 2–37 on page 2–131. • Updated Note 5 in Table 2–41 on page 2–135.
	April 2004, v3.0	• Added note 3 to rows 11 and 12 in Table 2–18. • Deleted "Stratix and Stratix GX Device PLL Availability" table. • Added I/O standards row in Table 2–28 that support max and min strength. • Row `clk [1,3,8,10]` was removed from Table 2–30. • Added checkmarks in Enhanced column for LVPECL, 3.3-V PCML, LVDS, and HyperTransport technology rows in Table 2–32. • Removed the Left and Right I/O Banks row in Table 2–34. • Changed `RCLK` values in Figures 2–50 and 2–51. • External RAM Interfacing section replaced.
	November 2003, v2.2	• Added 672-pin BGA package information in Table 2–37. • Removed support for series and parallel on-chip termination. • Termination Technology renamed differential on-chip termination. • Updated the number of channels per PLL in Tables 2-38 through 2-42. • Updated Figures 2–65 and 2–67.
	October 2003, v2.1	• Updated DDR I information. • Updated Table 2–22. • Added Tables 2–25, 2–29, 2–30, and 2–72. • Updated Figures 2–59, 2–65, and 2–67. • Updated the Lock Detect section.

Chapter	Date/Version	Changes Made
2	July 2003, v2.0	• Added reference on page 2-73 to Figures 2-50 and 2-51 for RCLK connections. • Updated ranges for EPLL post-scale and pre-scale dividers on page 2-85. • Updated PLL Reconfiguration frequency from 25 to 22 MHz on page 2-87. • New requirement to assert are set signal each PLL when it has to re-acquire lock on either a new clock after loss of lock (page 2-96). • Updated max input frequency for CLK[1,3,8,10] from 462 to 500, Table 2-24. • Renamed impedance matching to series termination throughout. • Updated naming convention for DQS pins on page 2-112 to match pin tables. • Added DDR SDRAM Performance Specification on page 2-117. • Added external reference resistor values for terminator technology (page 2-136). • Added Terminator Technology Specification on pages 2-137 and 2-138. • Updated Tables 2-45 to 2-49 to reflect PLL cross-bank support for high speed differential channels at full speed. • Wire bond package performance specification for “high” speed channels was increased to 624 Mbps from 462 Mbps throughout chapter.
3	July 2005, v1.3	• Updated “Operating Modes” section. • Updated “Temperature Sensing Diode” section. • Updated “IEEE Std. 1149.1 (JTAG) Boundary-Scan Support” section. • Updated “Configuration” section.
	January 2005, v1.2	• Updated limits for JTAG chain of devices.
	September 2004, v1.1	• Added new section, “Stratix Automated Single Event Upset (SEU) Detection” on page 3–12. • Updated description of “Custom-Built Circuitry” on page 3–13.
	April 2003, v1.0	• No new changes in *Stratix Device Handbook* v2.0.
4	January 2006, v3.4	• Added Table 4–135.
	July 2005, v3.3	• Updated Tables 4–6 and 4–30. • Updated Tables 4–103 through 4–108. • Updated Tables 4–114 through 4–124. • Updated Table 4–129. • Added Table 4–130.

Chapter	Date/Version	Changes Made
4	January 2005, 3.2	• Updated rise and fall input values.
	September 2004, v3.1	• Updated Note 3 in Table 4–8 on page 4–4. • Updated Table 4–10 on page 4–6. • Updated Table 4–20 on page 4–12 through Table 4–23 on page 4–13. Added rows $V_{IL(AC)}$ and $V_{IH(AC)}$ to each table. • Updated Table 4–26 on page 4–14 through Table 4–29 on page 4–15. • Updated Table 4–31 on page 4–16. • Updated description of "External Timing Parameters" on page 4–33. • Updated Table 4–36 on page 4–20. • Added signals t_{OUTCO}, T_{XZ}, and T_{ZX} to Figure 4–4 on page 4–33. • Added rows $t_{M512CLKENSU}$ and $t_{M512CLKENH}$ to Table 4–40 on page 4–24. • Added rows $t_{M4CLKENSU}$ and $t_{M4CLKENH}$ to Table 4–41 on page 4–24. • Updated Note 2 in Table 4–54 on page 4–35. • Added rows $t_{MRAMCLKENSU}$ and $t_{MRAMCLKENH}$ to Table 4–42 on page 4–25. • Updated Table 4–46 on page 4–29. • Updated Table 4–47 on page 4–29.

Chapter	Date/Version	Changes Made
4		• Table 4–48 on page 4–30: added rows $t_{M512CLKSENSU}$ and $t_{M512CLKENH}$, and updated symbol names. • Updated power-up current (ICCINT) required to power a Stratix device on page 4–17. • Updated Table 4–37 on page 4–22 through Table 4–43 on page 4–27. • Table 4–49 on page 4–31: added rows $t_{M4KCLKENSU}$, $t_{M4KCLKENH}$, $t_{M4KBESU}$, and $t_{M4KBEH,}$ deleted rows $t_{M4KRADDRASU}$ and $t_{M4KRADDRH}$, and updated symbol names. • Table 4–50 on page 4–31: added rows $t_{MRAMCLKENSU}$, $t_{MRAMCLKENH}$, $t_{MRAMBESU}$, and $t_{MRAMBEH}$, deleted rows $t_{MRAMADDRASU}$ and $t_{MRAMRADDRH}$, and updated symbol names. • Table 4–52 on page 4–34: updated table, deleted “Conditions” column, and added rows t_{XZ} and t_{ZX}. • Table 4–52 on page 4–34: updated table, deleted “Conditions” column, and added rows t_{XZ} and t_{ZX}. • Table 4–53 on page 4–34: updated table and added rows t_{XZPLL} and t_{ZXPLL}. • Updated Note 2 in Table 4–53 on page 4–34. • Table 4–54 on page 4–35: updated table, deleted “Conditions” column, and added rows t_{XZPLL} and t_{ZXPLL}. • Updated Note 2 in Table 4–54 on page 4–35. • Deleted Note 2 from Table 4–55 on page 4–36 through Table 4–66 on page 4–41. • Updated Table 4–55 on page 4–36 through Table 4–96 on page 4–56. Added rows T_{XZ}, T_{ZX}, T_{XZPLL}, and $T_{ZXPLL.}$ • Added Note 4 to Table 4–101 on page 4–62. • Deleted Note 1 from Table 4–67 on page 4–42 through Table 4–84 on page 4–50. • Added new section “I/O Timing Measurement Methodology” on page 4–60. • Deleted Note 1 from Table 4–67 on page 4–42 through Table 4–84 on page 4–50. • Deleted Note 2 from Table 4–85 on page 4–51 through Table 4–96 on page 4–56. • Added Note 4 to Table 4–101 on page 4–62. • Table 4–102 on page 4–64: updated table and added Note 4. • Updated description of “External I/O Delay Parameters” on page 4–66. • Added Note 1 to Table 4–109 on page 4–73 and Table 4–110 on page 4–74. • Updated Table 4–103 on page 4–66 through Table 4–110 on page 4–74. • Deleted Note 2 from Table 4–103 on page 4–66 through Table 4–106 on page 4–69. • Added new paragraph about output adder delays on page 4–68. • Updated Table 4–110 on page 4–74. • Added Note 1 to Table 4–111 through Table 4–113 on page 4–75.

Chapter	Date/Version	Changes Made
4		• Updated Table 4–123 on page 4–85 through Table 4–126 on page 4–92. • Updated Note 3 in Table 4–123 on page 4–85. • Table 4–125 on page 4–88: moved to correct order in chapter, and updated table. • Updated Table 4–126 on page 4–92. • Updated Table 4–127 on page 4–94. • Updated Table 4–128 on page 4–95.
	April 2004, v3.0	• Table 4–129 on page 4–96: updated table and added Note 10. • Updated Table 4–131 and Table 4–132 on page 4–100. • Updated Table 4–110 on page 4–74. • Updated Table 4–123 on page 4–85. • Updated Table 4–124 on page 4–87. through Table 4–126 on page 4–92. • Added Note 10 to Table 4–129 on page 4–96. • Moved Table 4–127 on page 4–94 to correct order in the chapter. • Updated Table 4–131 on page 4–100 through Table 4–132 on page 4–100. • Deleted t_{XZ} and t_{ZX} from Figure 4–4. • Waveform was added to Figure 4–6. • The minimum and maximum duty cycle values in Note 3 of Table 4–8 were moved to a new Table 4–9. • Changes were made to values in SSTL-3 Class I and II rows in Table 4–17. • Note 1 was added to Table 4–34. • Added t_{SU_R} and t_{SU_C} rows in Table 4–38. • Changed Table 4–55 title from “EP1S10 Column Pin Fast Regional Clock External I/O Timing Parameters” to “EP1S10 External I/O Timing on Column Pins Using Fast Regional Clock Networks.” • Changed values in Tables 4–46, 4–48 to 4–51, 4–128, and 4–131. • Added t_{ARESET} row in Tables 4–127 to 4–132. • Deleted -5 Speed Grade column in Tables 4–117 to 4–119 and 4–122 to 4–123. • Fixed differential waveform in Figure 4–1. • Added “Definition of I/O Skew” section. • Added t_{SU} and t_{CO_C} rows and made changes to values in t_{PRE} and t_{CLKHL} rows in Table 4–46. • Values changed in the t_{SU} and t_H rows in Table 4–47. • Values changed in the $t_{M4KCLKHL}$ row in Table 4–49. • Values changed in the $t_{MRAMCLKHL}$ row in Table 4–50. • Added Table 4–51 to “Internal Timing Parameters” section. • The timing information is preliminary in Tables 4–55 through 4–96. • Table 4–111 was separated into 3 tables: Tables 4–111 to 4–113.
	November 2003, v2.2	• Updated Tables 4–127 through 4–129.

Chapter	Date/Version	Changes Made
4	October 2003, v2.1	• Added -8 speed grade information. • Updated performance information in Table 4–36. • Updated timing information in Tables 4–55 through 4–96. • Updated delay information in Tables 4–103 through 4–108. • Updated programmable delay information in Tables 4–100 and 4–103.
	July 2003, v2.0	• Updated clock rates in Tables 4–114 through 4–123. • Updated speed grade information in the introduction on page 4-1. • Corrected figures 4-1 & 4-2 and Table 4-9 to reflect how VID and VOD are specified. • Added note 6 to Table 4-32. • Updated Stratix Performance Table 4-35. • Updated EP1S60 and EP1S80 timing parameters in Tables 4-82 to 4-93. The Stratix timing models are final for all devices. • Updated Stratix IOE programmable delay chains in Tables 4-100 to 4-101. • Added single-ended I/O standard output pin delay adders for loading in Table 4-102. • Added spec for FPLL[10..7]CLK pins in Tables 4-104 and 4-107. • Updated high-speed I/O specification for J=2 in Tables 4-114 and 4-115. • Updated EPLL specification and fast PLL specification in Tables 4-116 to 4-120.
5	September 2004, v2.1	• Updated reference to device pin-outs on page 5–1 to indicate that device pin-outs are no longer included in this manual and are now available on the Altera web site.
	April 2003, v1.0	• No new changes in Stratix Device Handbook v2.0.

1. Introduction

S51001-3.2

Introduction

The Stratix® family of FPGAs is based on a 1.5-V, 0.13-µm, all-layer copper SRAM process, with densities of up to 79,040 logic elements (LEs) and up to 7.5 Mbits of RAM. Stratix devices offer up to 22 digital signal processing (DSP) blocks with up to 176 (9-bit × 9-bit) embedded multipliers, optimized for DSP applications that enable efficient implementation of high-performance filters and multipliers. Stratix devices support various I/O standards and also offer a complete clock management solution with its hierarchical clock structure with up to 420-MHz performance and up to 12 phase-locked loops (PLLs).

The following shows the main sections in the Stratix Device Family Data Sheet:

Section	Page
Features	1–2
Functional Description	2–1
Logic Array Blocks	2–3
Logic Elements	2–6
MultiTrack Interconnect	2–14
TriMatrix Memory	2–21
Digital Signal Processing Block	2–52
PLLs & Clock Networks	2–73
I/O Structure	2–104
High-Speed Differential I/O Support	2–130
Power Sequencing & Hot Socketing	2–140
IEEE Std. 1149.1 (JTAG) Boundary-Scan Support	3–1
SignalTap II Embedded Logic Analyzer	3–5
Configuration	3–5
Temperature Sensing Diode	3–13
Operating Conditions	4–1
Power Consumption	4–17
Timing Model	4–19
Software	5–1
Device Pin-Outs	5–1
Ordering Information	5–1

Features

The Stratix family offers the following features:

- 10,570 to 79,040 LEs; see Table 1–1
- Up to 7,427,520 RAM bits (928,440 bytes) available without reducing logic resources
- TriMatrix™ memory consisting of three RAM block sizes to implement true dual-port memory and first-in first-out (FIFO) buffers
- High-speed DSP blocks provide dedicated implementation of multipliers (faster than 300 MHz), multiply-accumulate functions, and finite impulse response (FIR) filters
- Up to 16 global clocks with 22 clocking resources per device region
- Up to 12 PLLs (four enhanced PLLs and eight fast PLLs) per device provide spread spectrum, programmable bandwidth, clock switch-over, real-time PLL reconfiguration, and advanced multiplication and phase shifting
- Support for numerous single-ended and differential I/O standards
- High-speed differential I/O support on up to 116 channels with up to 80 channels optimized for 840 megabits per second (Mbps)
- Support for high-speed networking and communications bus standards including RapidIO, UTOPIA IV, CSIX, HyperTransport™ technology, 10G Ethernet XSBI, SPI-4 Phase 2 (POS-PHY Level 4), and SFI-4
- Differential on-chip termination support for LVDS
- Support for high-speed external memory, including zero bus turnaround (ZBT) SRAM, quad data rate (QDR and QDRII) SRAM, double data rate (DDR) SDRAM, DDR fast cycle RAM (FCRAM), and single data rate (SDR) SDRAM
- Support for 66-MHz PCI (64 and 32 bit) in -6 and faster speed-grade devices, support for 33-MHz PCI (64 and 32 bit) in -8 and faster speed-grade devices
- Support for 133-MHz PCI-X 1.0 in -5 speed-grade devices
- Support for 100-MHz PCI-X 1.0 in -6 and faster speed-grade devices
- Support for 66-MHz PCI-X 1.0 in -7 speed-grade devices
- Support for multiple intellectual property megafunctions from Altera MegaCore® functions and Altera Megafunction Partners Program (AMPP℠) megafunctions
- Support for remote configuration updates

Table 1–1. Stratix Device Features — EP1S10, EP1S20, EP1S25, EP1S30

Feature	EP1S10	EP1S20	EP1S25	EP1S30
LEs	10,570	18,460	25,660	32,470
M512 RAM blocks (32 × 18 bits)	94	194	224	295
M4K RAM blocks (128 × 36 bits)	60	82	138	171
M-RAM blocks (4K × 144 bits)	1	2	2	4
Total RAM bits	920,448	1,669,248	1,944,576	3,317,184
DSP blocks	6	10	10	12
Embedded multipliers *(1)*	48	80	80	96
PLLs	6	6	6	10
Maximum user I/O pins	426	586	706	726

Table 1–2. Stratix Device Features — EP1S40, EP1S60, EP1S80

Feature	EP1S40	EP1S60	EP1S80
LEs	41,250	57,120	79,040
M512 RAM blocks (32 × 18 bits)	384	574	767
M4K RAM blocks (128 × 36 bits)	183	292	364
M-RAM blocks (4K × 144 bits)	4	6	9
Total RAM bits	3,423,744	5,215,104	7,427,520
DSP blocks	14	18	22
Embedded multipliers *(1)*	112	144	176
PLLs	12	12	12
Maximum user I/O pins	822	1,022	1,238

Note to Tables 1–1 and 1–2:

(1) This parameter lists the total number of 9 × 9-bit multipliers for each device. For the total number of 18 × 18-bit multipliers per device, divide the total number of 9 × 9-bit multipliers by 2. For the total number of 36 × 36-bit multipliers per device, divide the total number of 9 × 9-bit multipliers by 8.

Stratix devices are available in space-saving FineLine BGA® and ball-grid array (BGA) packages (see Tables 1–3 through 1–5). All Stratix devices support vertical migration within the same package (for example, you can migrate between the EP1S10, EP1S20, and EP1S25 devices in the 672-pin BGA package). Vertical migration means that you can migrate to devices whose dedicated pins, configuration pins, and power pins are the same for a given package across device densities. For I/O pin migration across densities, you must cross-reference the available I/O pins using the device pin-outs for all planned densities of a given package type to identify which I/O pins are migrational. The Quartus® II software can automatically cross reference and place all pins except differential pins for migration when given a device migration list. You must use the pin-outs for each device to verify the differential placement migration. A future version of the Quartus II software will support differential pin migration.

Table 1–3. Stratix Package Options & I/O Pin Counts

Device	672-Pin BGA	956-Pin BGA	484-Pin FineLine BGA	672-Pin FineLine BGA	780-Pin FineLine BGA	1,020-Pin FineLine BGA	1,508-Pin FineLine BGA
EP1S10	345		335	345	426		
EP1S20	426		361	426	586		
EP1S25	473			473	597	706	
EP1S30		683			597	726	
EP1S40		683			615	773	822
EP1S60		683				773	1,022
EP1S80		683				773	1,203

Note to Table 1–3:

(1) All I/O pin counts include 20 dedicated clock input pins (`clk[15..0]p`, `clk0n`, `clk2n`, `clk9n`, and `clk11n`) that can be used for data inputs.

Table 1–4. Stratix BGA Package Sizes

Dimension	672 Pin	956 Pin
Pitch (mm)	1.27	1.27
Area (mm^2)	1,225	1,600
Length × width (mm × mm)	35 × 35	40 × 40

Table 1–5. Stratix FineLine BGA Package Sizes

Dimension	484 Pin	672 Pin	780 Pin	1,020 Pin	1,508 Pin
Pitch (mm)	1.00	1.00	1.00	1.00	1.00
Area (mm^2)	529	729	841	1,089	1,600
Length × width (mm × mm)	23 × 23	27 × 27	29 × 29	33 × 33	40 × 40

Stratix devices are available in up to four speed grades, -5, -6, -7, and -8, with -5 being the fastest. Table 1–6 shows Stratix device speed-grade offerings.

Table 1–6. Stratix Device Speed Grades

Device	672-Pin BGA	956-Pin BGA	484-Pin FineLine BGA	672-Pin FineLine BGA	780-Pin FineLine BGA	1,020-Pin FineLine BGA	1,508-Pin FineLine BGA
EP1S10	-6, -7		-5, -6, -7	-6, -7	-5, -6, -7		
EP1S20	-6, -7		-5, -6, -7	-6, -7	-5, -6, -7		
EP1S25	-6, -7			-6, -7, -8	-5, -6, -7	-5, -6, -7	
EP1S30		-5, -6, -7			-5, -6, -7, -8	-5, -6, -7	
EP1S40		-5, -6, -7			-5, -6, -7, -8	-5, -6, -7	-5, -6, -7
EP1S60		-6, -7				-5, -6, -7	-6, -7
EP1S80		-6, -7				-5, -6, -7	-5, -6, -7

2. Stratix Architecture

S51002-3.2

Functional Description

Stratix® devices contain a two-dimensional row- and column-based architecture to implement custom logic. A series of column and row interconnects of varying length and speed provide signal interconnects between logic array blocks (LABs), memory block structures, and DSP blocks.

The logic array consists of LABs, with 10 logic elements (LEs) in each LAB. An LE is a small unit of logic providing efficient implementation of user logic functions. LABs are grouped into rows and columns across the device.

M512 RAM blocks are simple dual-port memory blocks with 512 bits plus parity (576 bits). These blocks provide dedicated simple dual-port or single-port memory up to 18-bits wide at up to 318 MHz. M512 blocks are grouped into columns across the device in between certain LABs.

M4K RAM blocks are true dual-port memory blocks with 4K bits plus parity (4,608 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 36-bits wide at up to 291 MHz. These blocks are grouped into columns across the device in between certain LABs.

M-RAM blocks are true dual-port memory blocks with 512K bits plus parity (589,824 bits). These blocks provide dedicated true dual-port, simple dual-port, or single-port memory up to 144-bits wide at up to 269 MHz. Several M-RAM blocks are located individually or in pairs within the device's logic array.

Digital signal processing (DSP) blocks can implement up to either eight full-precision 9 × 9-bit multipliers, four full-precision 18 × 18-bit multipliers, or one full-precision 36 × 36-bit multiplier with add or subtract features. These blocks also contain 18-bit input shift registers for digital signal processing applications, including FIR and infinite impulse response (IIR) filters. DSP blocks are grouped into two columns in each device.

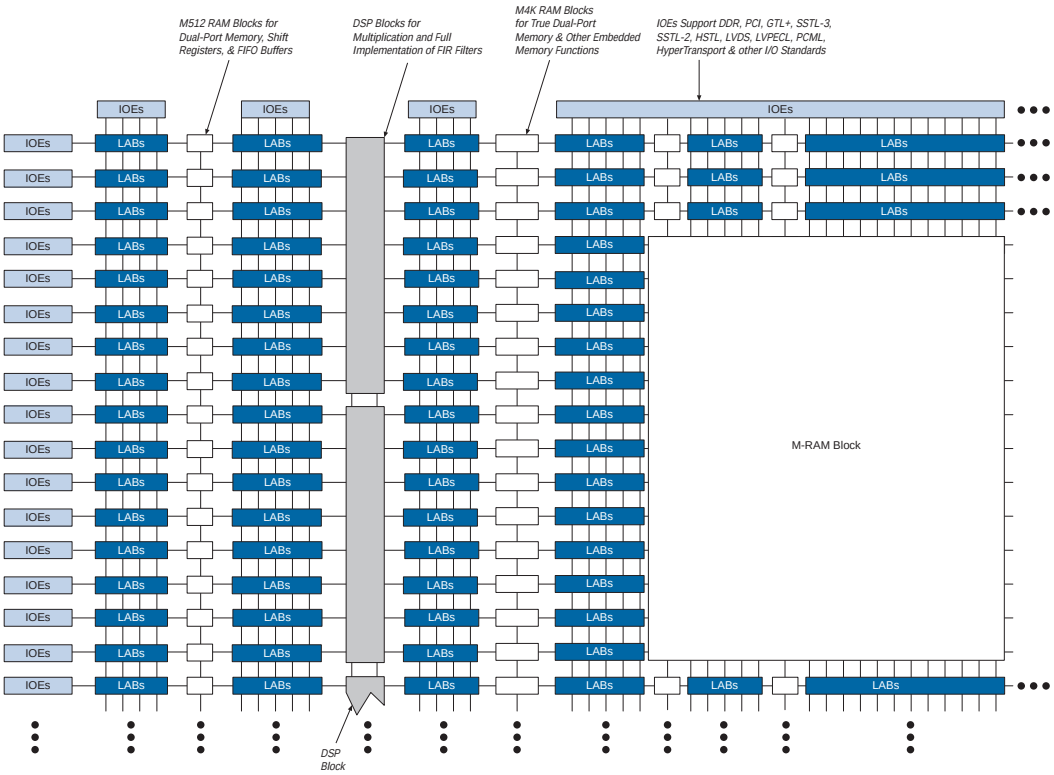
Each Stratix device I/O pin is fed by an I/O element (IOE) located at the end of LAB rows and columns around the periphery of the device. I/O pins support numerous single-ended and differential I/O standards. Each IOE contains a bidirectional I/O buffer and six registers for registering input, output, and output-enable signals. When used with

dedicated clocks, these registers provide exceptional performance and interface support with external memory devices such as DDR SDRAM, FCRAM, ZBT, and QDR SRAM devices.

High-speed serial interface channels support transfers at up to 840 Mbps using LVDS, LVPECL, 3.3-V PCML, or HyperTransport technology I/O standards.

Figure 2–1 shows an overview of the Stratix device.

Figure 2–1. Stratix Block Diagram



The number of M512 RAM, M4K RAM, and DSP blocks varies by device along with row and column numbers and M-RAM blocks. Table 2–1 lists the resources available in Stratix devices.

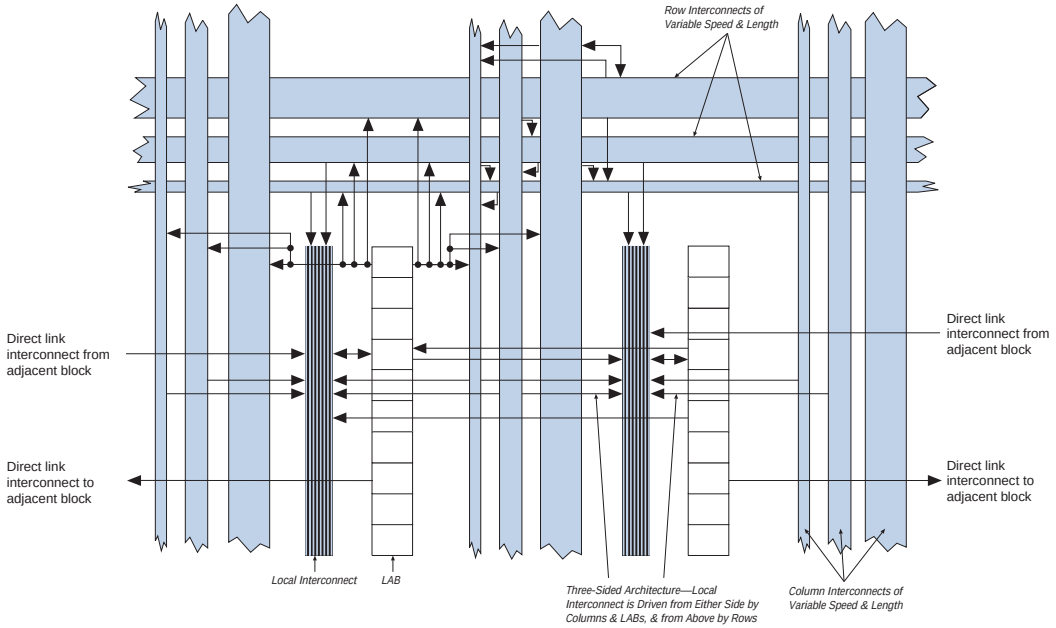
Table 2–1. Stratix Device Resources

Device	M512 RAM Columns/Blocks	M4K RAM Columns/Blocks	M-RAM Blocks	DSP Block Columns/Blocks	LAB Columns	LAB Rows
EP1S10	4 / 94	2 / 60	1	2 / 6	40	30
EP1S20	6 / 194	2 / 82	2	2 / 10	52	41
EP1S25	6 / 224	3 / 138	2	2 / 10	62	46
EP1S30	7 / 295	3 / 171	4	2 / 12	67	57
EP1S40	8 / 384	3 / 183	4	2 / 14	77	61
EP1S60	10 / 574	4 / 292	6	2 / 18	90	73
EP1S80	11 / 767	4 / 364	9	2 / 22	101	91

Logic Array Blocks

Each LAB consists of 10 LEs, LE carry chains, LAB control signals, local interconnect, LUT chain, and register chain connection lines. The local interconnect transfers signals between LEs in the same LAB. LUT chain connections transfer the output of one LE's LUT to the adjacent LE for fast sequential LUT connections within the same LAB. Register chain connections transfer the output of one LE's register to the adjacent LE's register within an LAB. The Quartus® II Compiler places associated logic within an LAB or adjacent LABs, allowing the use of local, LUT chain, and register chain connections for performance and area efficiency. Figure 2–2 shows the Stratix LAB.

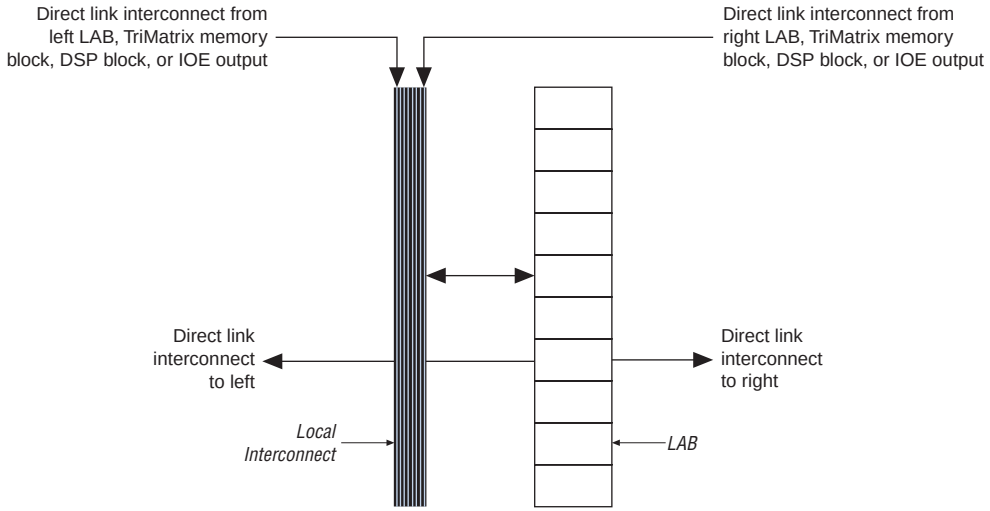
Figure 2–2. Stratix LAB Structure



LAB Interconnects

The LAB local interconnect can drive LEs within the same LAB. The LAB local interconnect is driven by column and row interconnects and LE outputs within the same LAB. Neighboring LABs, M512 RAM blocks, M4K RAM blocks, or DSP blocks from the left and right can also drive an LAB's local interconnect through the direct link connection. The direct link connection feature minimizes the use of row and column interconnects, providing higher performance and flexibility. Each LE can drive 30 other LEs through fast local and direct link interconnects. Figure 2–3 shows the direct link connection.

Figure 2–3. Direct Link Connection



LAB Control Signals

Each LAB contains dedicated logic for driving control signals to its LEs. The control signals include two clocks, two clock enables, two asynchronous clears, synchronous clear, asynchronous preset/load, synchronous load, and add/subtract control signals. This gives a maximum of 10 control signals at a time. Although synchronous load and clear signals are generally used when implementing counters, they can also be used with other functions.

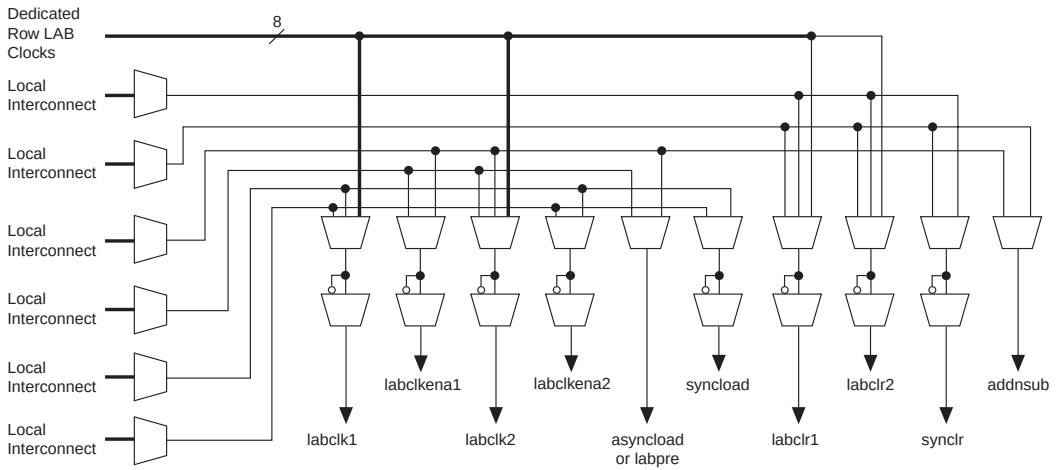
Each LAB can use two clocks and two clock enable signals. Each LAB's clock and clock enable signals are linked. For example, any LE in a particular LAB using the `labclk1` signal will also use `labclkena1`. If the LAB uses both the rising and falling edges of a clock, it also uses both LAB-wide clock signals. De-asserting the clock enable signal will turn off the LAB-wide clock.

Each LAB can use two asynchronous clear signals and an asynchronous load/preset signal. The asynchronous load acts as a preset when the asynchronous load data input is tied high.

With the LAB-wide addnsub control signal, a single LE can implement a one-bit adder and subtractor. This saves LE resources and improves performance for logic functions such as DSP correlators and signed multipliers that alternate between addition and subtraction depending on data.

The LAB row clocks [7..0] and LAB local interconnect generate the LAB-wide control signals. The MultiTrack™ interconnect's inherent low skew allows clock and control signal distribution in addition to data. Figure 2–4 shows the LAB control signal generation circuit.

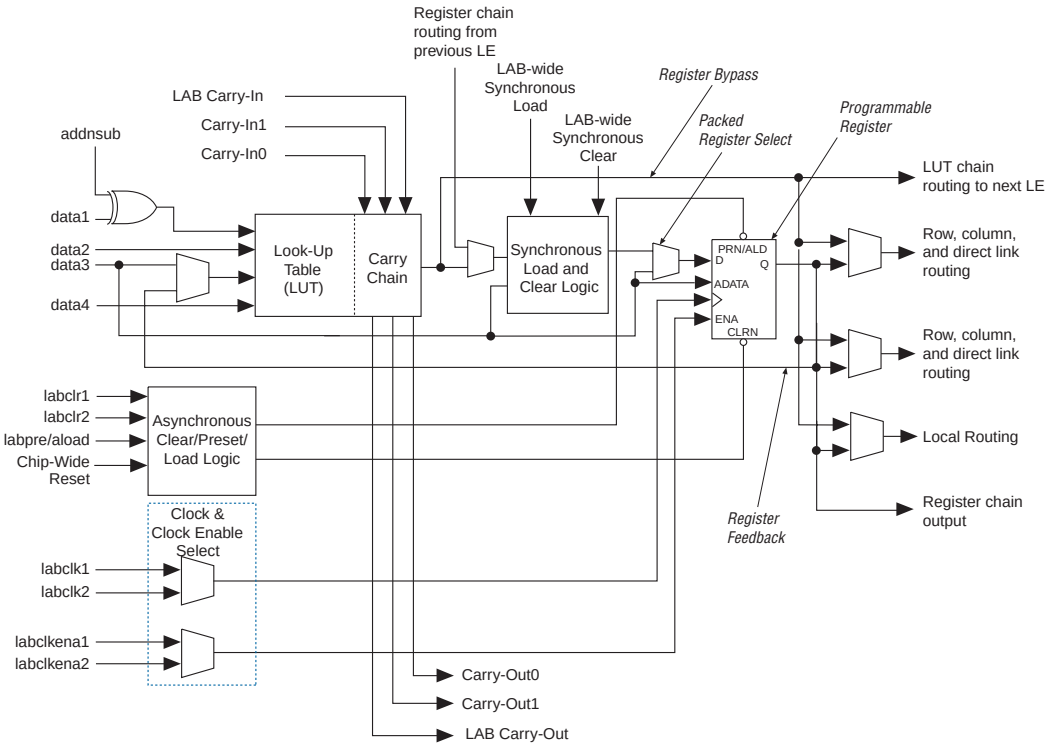
Figure 2–4. LAB-Wide Control Signals



Logic Elements

The smallest unit of logic in the Stratix architecture, the LE, is compact and provides advanced features with efficient logic utilization. Each LE contains a four-input LUT, which is a function generator that can implement any function of four variables. In addition, each LE contains a programmable register and carry chain with carry select capability. A single LE also supports dynamic single bit addition or subtraction mode selectable by an LAB-wide control signal. Each LE drives all types of interconnects: local, row, column, LUT chain, register chain, and direct link interconnects. See Figure 2–5.

Figure 2–5. Stratix LE



Each LE's programmable register can be configured for D, T, JK, or SR operation. Each register has data, true asynchronous load data, clock, clock enable, clear, and asynchronous load / preset inputs. Global signals, general-purpose I / O pins, or any internal logic can drive the register's clock and clear control signals. Either general-purpose I / O pins or internal logic can drive the clock enable, preset, asynchronous load, and asynchronous data. The asynchronous load data input comes from the `data3` input of the LE. For combinatorial functions, the register is bypassed and the output of the LUT drives directly to the outputs of the LE.

Each LE has three outputs that drive the local, row, and column routing resources. The LUT or register output can drive these three outputs independently. Two LE outputs drive column or row and direct link routing connections and one drives local interconnect resources. This allows the LUT to drive one output while the register drives another output. This feature, called register packing, improves device utilization because the device can use the register and the LUT for unrelated

functions. Another special packing mode allows the register output to feed back into the LUT of the same LE so that the register is packed with its own fan-out LUT. This provides another mechanism for improved fitting. The LE can also drive out registered and unregistered versions of the LUT output.

LUT Chain & Register Chain

In addition to the three general routing outputs, the LEs within an LAB have LUT chain and register chain outputs. LUT chain connections allow LUTs within the same LAB to cascade together for wide input functions. Register chain outputs allow registers within the same LAB to cascade together. The register chain output allows an LAB to use LUTs for a single combinatorial function and the registers to be used for an unrelated shift register implementation. These resources speed up connections between LABs while saving local interconnect resources. See "MultiTrack Interconnect" on page 2–14 for more information on LUT chain and register chain connections.

addnsub Signal

The LE's dynamic adder / subtractor feature saves logic resources by using one set of LEs to implement both an adder and a subtractor. This feature is controlled by the LAB-wide control signal `addnsub`. The `addnsub` signal sets the LAB to perform either A + B or A – B. The LUT computes addition, and subtraction is computed by adding the two's complement of the intended subtractor. The LAB-wide signal converts to two's complement by inverting the B bits within the LAB and setting carry-in = 1 to add one to the least significant bit (LSB). The LSB of an adder / subtractor must be placed in the first LE of the LAB, where the LAB-wide `addnsub` signal automatically sets the carry-in to 1. The Quartus II Compiler automatically places and uses the adder / subtractor feature when using adder / subtractor parameterized functions.

LE Operating Modes

The Stratix LE can operate in one of the following modes:

- Normal mode
- Dynamic arithmetic mode

Each mode uses LE resources differently. In each mode, eight available inputs to the LE—the four data inputs from the LAB local interconnect; `carry-in0` and `carry-in1` from the previous LE; the LAB carry-in from the previous carry-chain LAB; and the register chain connection—are directed to different destinations to implement the desired logic function. LAB-wide signals provide clock, asynchronous clear,

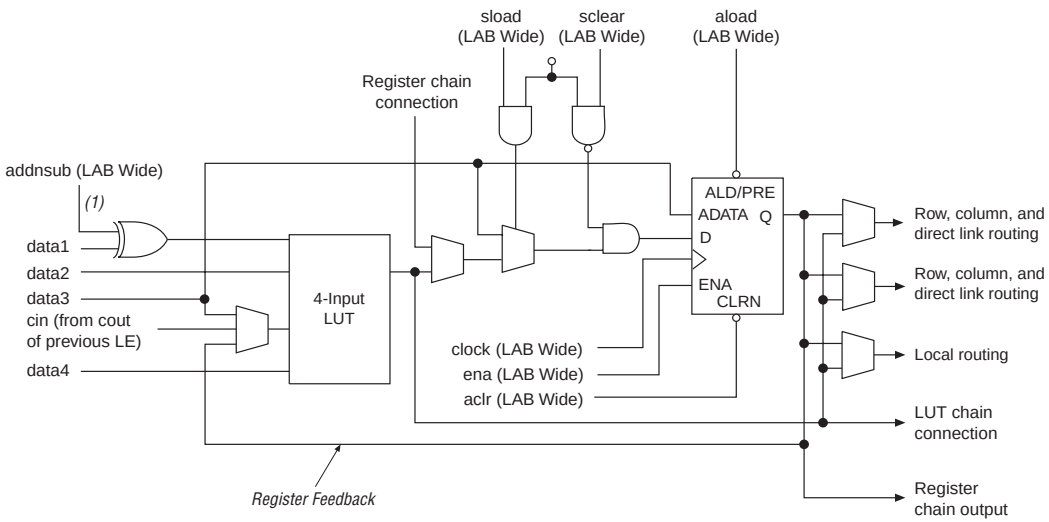
asynchronous preset load, synchronous clear, synchronous load, and clock enable control for the register. These LAB-wide signals are available in all LE modes. The `addnsub` control signal is allowed in arithmetic mode.

The Quartus II software, in conjunction with parameterized functions such as library of parameterized modules (LPM) functions, automatically chooses the appropriate mode for common functions such as counters, adders, subtractors, and arithmetic functions. If required, you can also create special-purpose functions that specify which LE operating mode to use for optimal performance.

Normal Mode

The normal mode is suitable for general logic applications and combinatorial functions. In normal mode, four data inputs from the LAB local interconnect are inputs to a four-input LUT (see Figure 2–6). The Quartus II Compiler automatically selects the carry-in or the `data3` signal as one of the inputs to the LUT. Each LE can use LUT chain connections to drive its combinatorial output directly to the next LE in the LAB. Asynchronous load data for the register comes from the `data3` input of the LE. LEs in normal mode support packed registers.

Figure 2–6. LE in Normal Mode



Note to Figure 2–6:

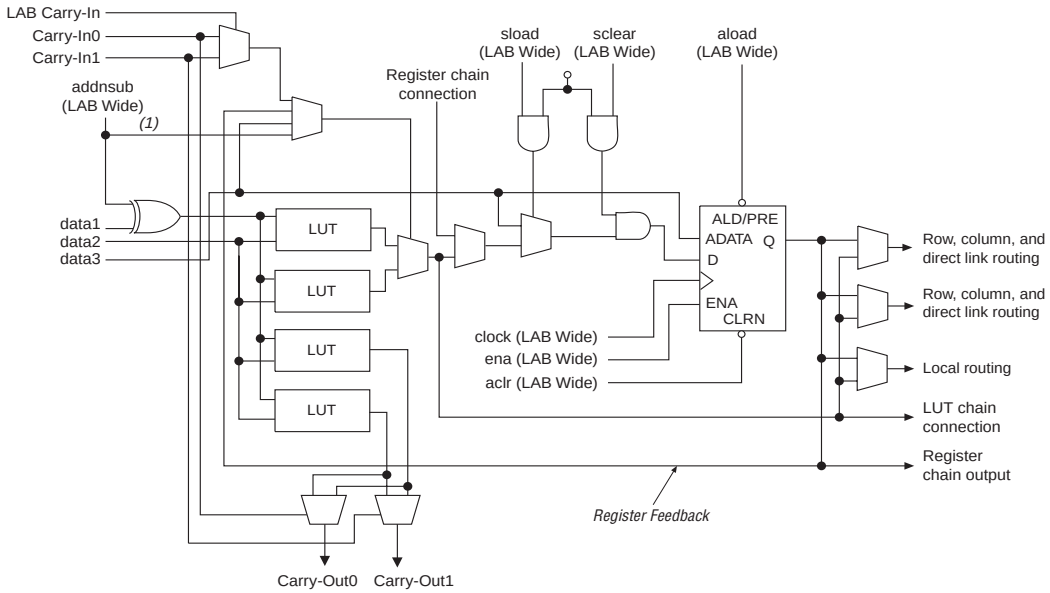
(1) This signal is only allowed in normal mode if the LE is at the end of an adder / subtractor chain.

Dynamic Arithmetic Mode

The dynamic arithmetic mode is ideal for implementing adders, counters, accumulators, wide parity functions, and comparators. An LE in dynamic arithmetic mode uses four 2-input LUTs configurable as a dynamic adder / subtractor. The first two 2-input LUTs compute two summations based on a possible carry-in of 1 or 0; the other two LUTs generate carry outputs for the two chains of the carry select circuitry. As shown in Figure 2–7, the LAB carry-in signal selects either the `carry-in0` or `carry-in1` chain. The selected chain's logic level in turn determines which parallel sum is generated as a combinatorial or registered output. For example, when implementing an adder, the sum output is the selection of two possible calculated sums: `data1` + `data2` + `carry-in0` or `data1` + `data2` + `carry-in1`. The other two LUTs use the `data1` and `data2` signals to generate two possible carry-out signals—one for a carry of 1 and the other for a carry of 0. The `carry-in0` signal acts as the carry select for the `carry-out0` output and `carry-in1` acts as the carry select for the `carry-out1` output. LEs in arithmetic mode can drive out registered and unregistered versions of the LUT output.

The dynamic arithmetic mode also offers clock enable, counter enable, synchronous up / down control, synchronous clear, synchronous load, and dynamic adder / subtractor options. The LAB local interconnect data inputs generate the counter enable and synchronous up / down control signals. The synchronous clear and synchronous load options are LAB-wide signals that affect all registers in the LAB. The Quartus II software automatically places any registers that are not used by the counter into other LABs. The `addnsub` LAB-wide signal controls whether the LE acts as an adder or subtractor.

Figure 2–7. LE in Dynamic Arithmetic Mode



Note to Figure 2–7:

(1) The addnsub signal is tied to the carry input for the first LE of a carry chain only.

Carry-Select Chain

The carry-select chain provides a very fast carry-select function between LEs in arithmetic mode. The carry-select chain uses the redundant carry calculation to increase the speed of carry functions. The LE is configured to calculate outputs for a possible carry-in of 1 and carry-in of 0 in parallel. The `carry-in0` and `carry-in1` signals from a lower-order bit feed forward into the higher-order bit via the parallel carry chain and feed into both the LUT and the next portion of the carry chain. Carry-select chains can begin in any LE within an LAB.

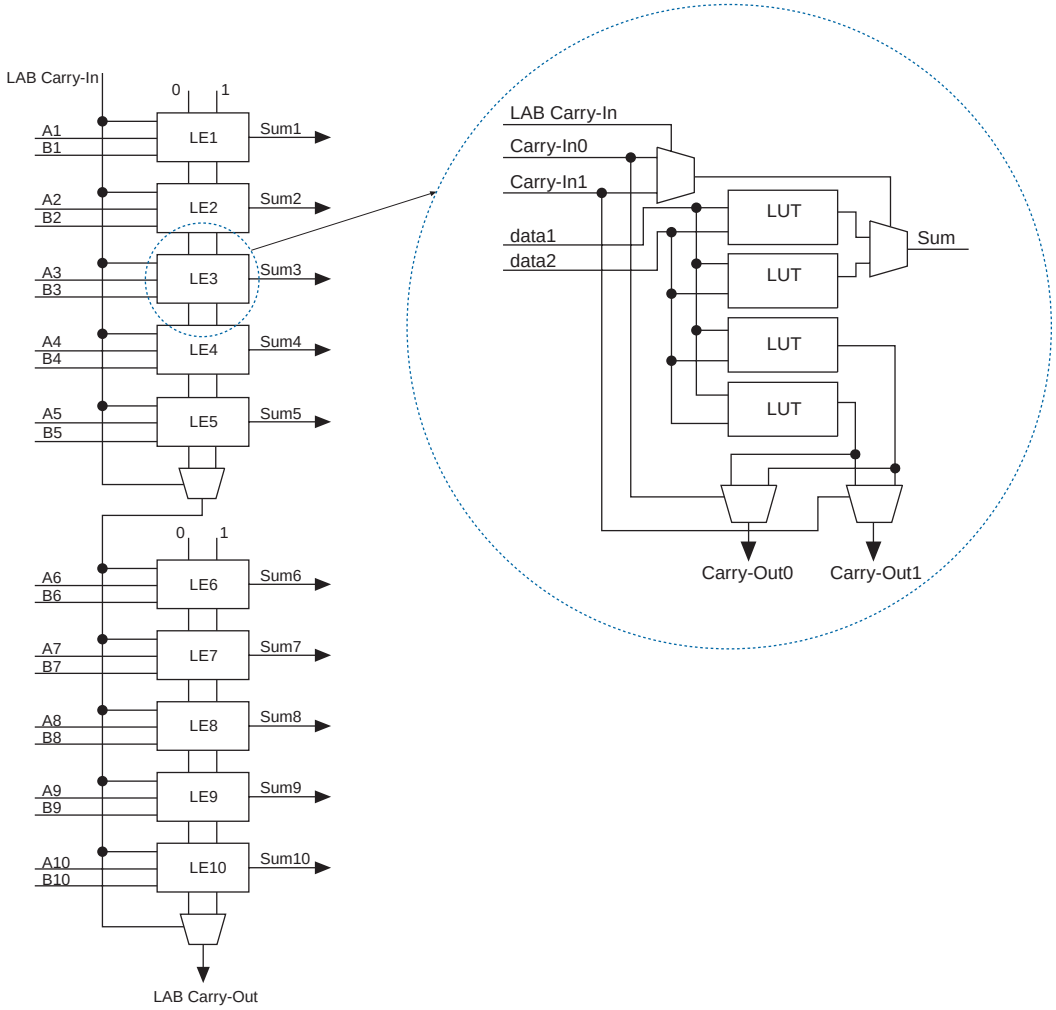
The speed advantage of the carry-select chain is in the parallel pre-computation of carry chains. Since the LAB carry-in selects the precomputed carry chain, not every LE is in the critical path. Only the propagation delay between LAB carry-in generation (LE 5 and LE 10) are now part of the critical path. This feature allows the Stratix architecture to implement high-speed counters, adders, multipliers, parity functions, and comparators of arbitrary width.

Figure 2–8 shows the carry-select circuitry in an LAB for a 10-bit full adder. One portion of the LUT generates the sum of two bits using the input signals and the appropriate carry-in bit; the sum is routed to the output of the LE. The register can be bypassed for simple adders or used for accumulator functions. Another portion of the LUT generates carry-out bits. An LAB-wide carry in bit selects which chain is used for the addition of given inputs. The carry-in signal for each chain, `carry-in0` or `carry-in1`, selects the carry-out to carry forward to the carry-in signal of the next-higher-order bit. The final carry-out signal is routed to an LE, where it is fed to local, row, or column interconnects.

The Quartus II Compiler automatically creates carry chain logic during design processing, or you can create it manually during design entry. Parameterized functions such as LPM functions automatically take advantage of carry chains for the appropriate functions.

The Quartus II Compiler creates carry chains longer than 10 LEs by linking LABs together automatically. For enhanced fitting, a long carry chain runs vertically allowing fast horizontal connections to TriMatrix™ memory and DSP blocks. A carry chain can continue as far as a full column.

Figure 2–8. Carry Select Chain



Clear & Preset Logic Control

LAB-wide signals control the logic for the register's clear and preset signals. The LE directly supports an asynchronous clear and preset function. The register preset is achieved through the asynchronous load of a logic high. The direct asynchronous preset does not require a NOT-gate push-back technique. Stratix devices support simultaneous preset/

asynchronous load, and clear signals. An asynchronous clear signal takes precedence if both signals are asserted simultaneously. Each LAB supports up to two clears and one preset signal.

In addition to the clear and preset ports, Stratix devices provide a chip-wide reset pin (DEV_CLRn) that resets all registers in the device. An option set before compilation in the Quartus II software controls this pin. This chip-wide reset overrides all other control signals.

MultiTrack Interconnect

In the Stratix architecture, connections between LEs, TriMatrix memory, DSP blocks, and device I/O pins are provided by the MultiTrack interconnect structure with DirectDrive™ technology. The MultiTrack interconnect consists of continuous, performance-optimized routing lines of different lengths and speeds used for inter- and intra-design block connectivity. The Quartus II Compiler automatically places critical design paths on faster interconnects to improve design performance.

DirectDrive technology is a deterministic routing technology that ensures identical routing resource usage for any function regardless of placement within the device. The MultiTrack interconnect and DirectDrive technology simplify the integration stage of block-based designing by eliminating the re-optimization cycles that typically follow design changes and additions.

The MultiTrack interconnect consists of row and column interconnects that span fixed distances. A routing structure with fixed length resources for all devices allows predictable and repeatable performance when migrating through different device densities. Dedicated row interconnects route signals to and from LABs, DSP blocks, and TriMatrix memory within the same row. These row resources include:

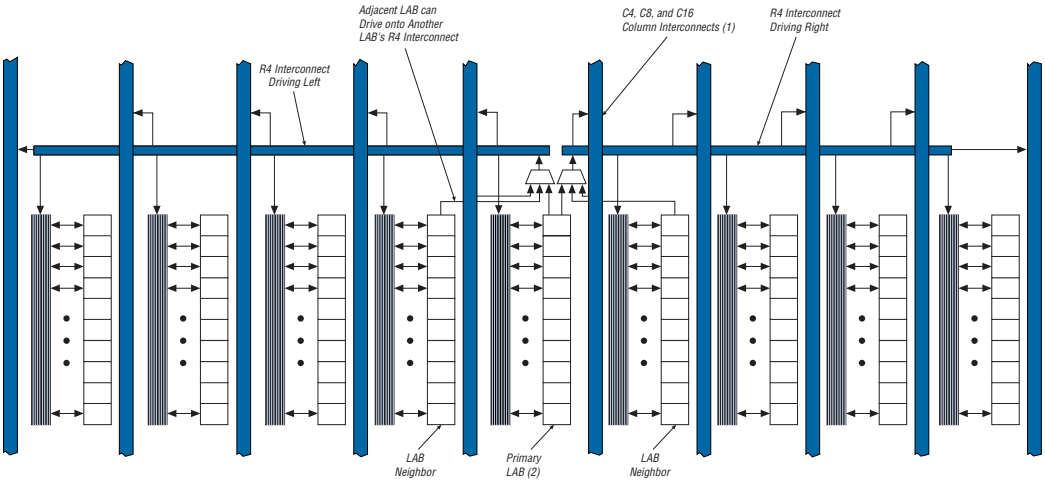
- Direct link interconnects between LABs and adjacent blocks.
- R4 interconnects traversing four blocks to the right or left.
- R8 interconnects traversing eight blocks to the right or left.
- R24 row interconnects for high-speed access across the length of the device.

The direct link interconnect allows an LAB, DSP block, or TriMatrix memory block to drive into the local interconnect of its left and right neighbors and then back into itself. Only one side of a M-RAM block interfaces with direct link and row interconnects. This provides fast communication between adjacent LABs and/or blocks without using row interconnect resources.

The R4 interconnects span four LABs, three LABs and one M512 RAM block, two LABs and one M4K RAM block, or two LABs and one DSP block to the right or left of a source LAB. These resources are used for fast

row connections in a four-LAB region. Every LAB has its own set of R4 interconnects to drive either left or right. Figure 2–9 shows R4 interconnect connections from an LAB. R4 interconnects can drive and be driven by DSP blocks and RAM blocks and horizontal IOEs. For LAB interfacing, a primary LAB or LAB neighbor can drive a given R4 interconnect. For R4 interconnects that drive to the right, the primary LAB and right neighbor can drive on to the interconnect. For R4 interconnects that drive to the left, the primary LAB and its left neighbor can drive on to the interconnect. R4 interconnects can drive other R4 interconnects to extend the range of LABs they can drive. R4 interconnects can also drive C4 and C16 interconnects for connections from one row to another. Additionally, R4 interconnects can drive R24 interconnects.

Figure 2–9. R4 Interconnect Connections



Notes to Figure 2–9:

(1) C4 interconnects can drive R4 interconnects.

(2) This pattern is repeated for every LAB in the LAB row.

The R8 interconnects span eight LABs, M512 or M4K RAM blocks, or DSP blocks to the right or left from a source LAB. These resources are used for fast row connections in an eight-LAB region. Every LAB has its own set of R8 interconnects to drive either left or right. R8 interconnect connections between LABs in a row are similar to the R4 connections shown in Figure 2–9, with the exception that they connect to eight LABs to the right or left, not four. Like R4 interconnects, R8 interconnects can drive and be driven by all types of architecture blocks. R8 interconnects

can drive other R8 interconnects to extend their range as well as C8 interconnects for row-to-row connections. One R8 interconnect is faster than two R4 interconnects connected together.

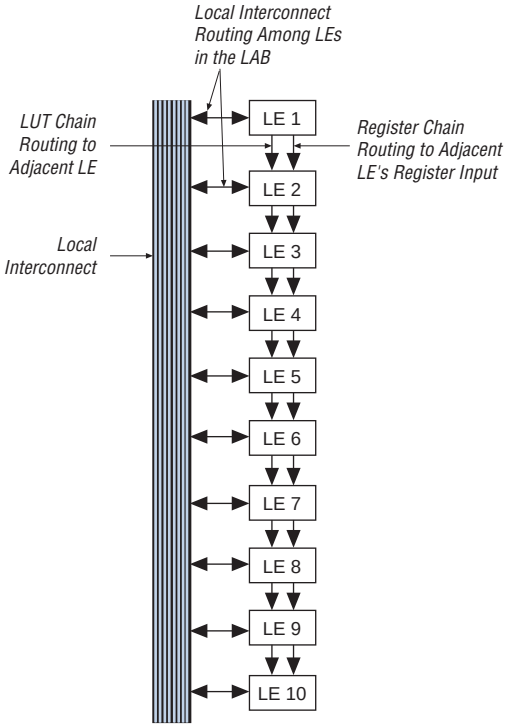
R24 row interconnects span 24 LABs and provide the fastest resource for long row connections between LABs, TriMatrix memory, DSP blocks, and IOEs. The R24 row interconnects can cross M-RAM blocks. R24 row interconnects drive to other row or column interconnects at every fourth LAB and do not drive directly to LAB local interconnects. R24 row interconnects drive LAB local interconnects via R4 and C4 interconnects. R24 interconnects can drive R24, R4, C16, and C4 interconnects.

The column interconnect operates similarly to the row interconnect and vertically routes signals to and from LABs, TriMatrix memory, DSP blocks, and IOEs. Each column of LABs is served by a dedicated column interconnect, which vertically routes signals to and from LABs, TriMatrix memory and DSP blocks, and horizontal IOEs. These column resources include:

- LUT chain interconnects within an LAB
- Register chain interconnects within an LAB
- C4 interconnects traversing a distance of four blocks in up and down direction
- C8 interconnects traversing a distance of eight blocks in up and down direction
- C16 column interconnects for high-speed vertical routing through the device

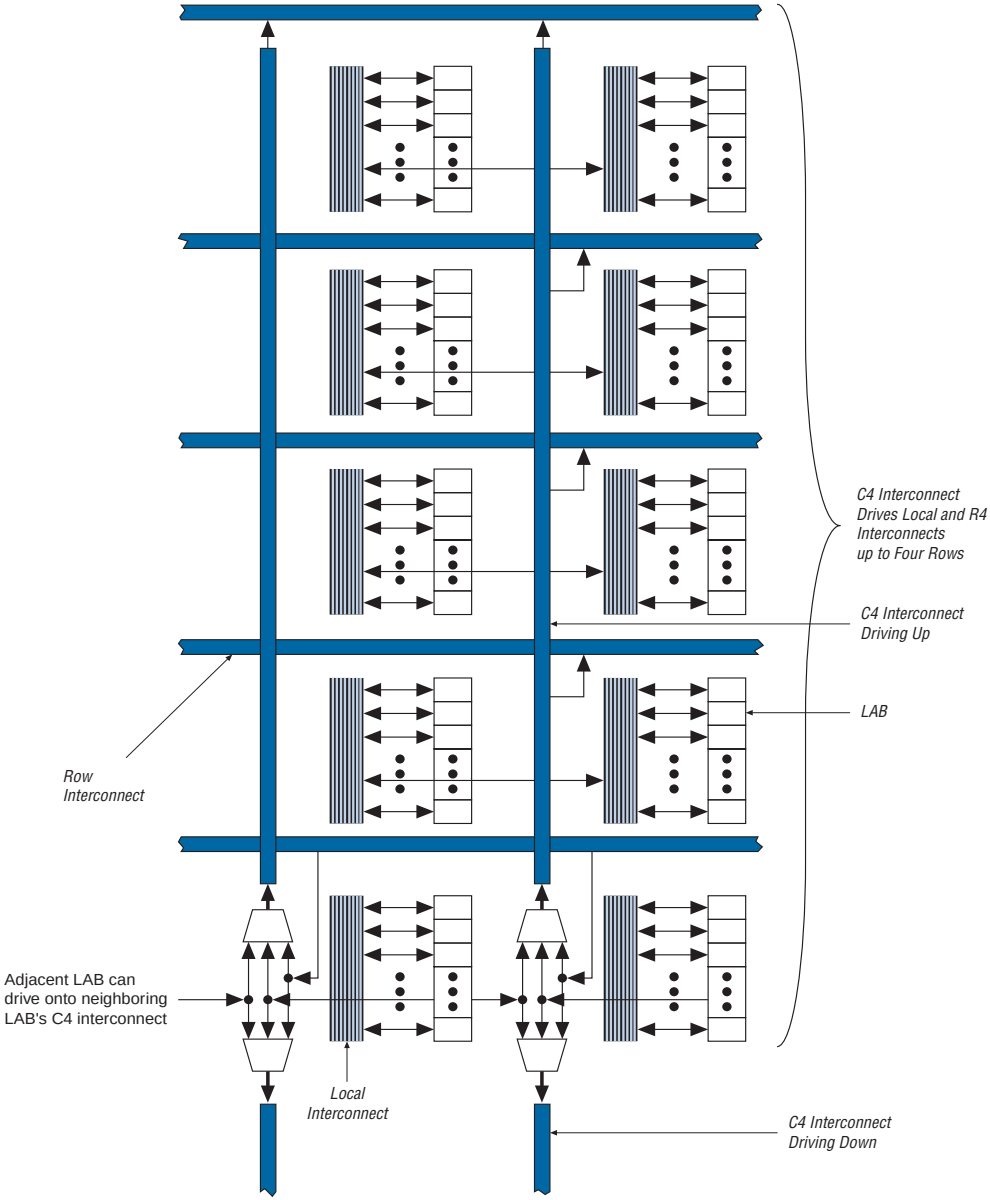
Stratix devices include an enhanced interconnect structure within LABs for routing LE output to LE input connections faster using LUT chain connections and register chain connections. The LUT chain connection allows the combinatorial output of an LE to directly drive the fast input of the LE right below it, bypassing the local interconnect. These resources can be used as a high-speed connection for wide fan-in functions from LE 1 to LE 10 in the same LAB. The register chain connection allows the register output of one LE to connect directly to the register input of the next LE in the LAB for fast shift registers. The Quartus II Compiler automatically takes advantage of these resources to improve utilization and performance. Figure 2–10 shows the LUT chain and register chain interconnects.

Figure 2–10. LUT Chain & Register Chain Interconnects



The C4 interconnects span four LABs, M512, or M4K blocks up or down from a source LAB. Every LAB has its own set of C4 interconnects to drive either up or down. Figure 2–11 shows the C4 interconnect connections from an LAB in a column. The C4 interconnects can drive and be driven by all types of architecture blocks, including DSP blocks, TriMatrix memory blocks, and vertical IOEs. For LAB interconnection, a primary LAB or its LAB neighbor can drive a given C4 interconnect.
C4 interconnects can drive each other to extend their range as well as drive row interconnects for column-to-column connections.

Figure 2–11. C4 Interconnect Connections *Note (1)*



Note to Figure 2–11:

(1) Each C4 interconnect can drive either up or down four rows.

C8 interconnects span eight LABs, M512, or M4K blocks up or down from a source LAB. Every LAB has its own set of C8 interconnects to drive either up or down. C8 interconnect connections between the LABs in a column are similar to the C4 connections shown in Figure 2–11 with the exception that they connect to eight LABs above and below. The C8 interconnects can drive and be driven by all types of architecture blocks similar to C4 interconnects. C8 interconnects can drive each other to extend their range as well as R8 interconnects for column-to-column connections. C8 interconnects are faster than two C4 interconnects.

C16 column interconnects span a length of 16 LABs and provide the fastest resource for long column connections between LABs, TriMatrix memory blocks, DSP blocks, and IOEs. C16 interconnects can cross M-RAM blocks and also drive to row and column interconnects at every fourth LAB. C16 interconnects drive LAB local interconnects via C4 and R4 interconnects and do not drive LAB local interconnects directly.

All embedded blocks communicate with the logic array similar to LAB-to-LAB interfaces. Each block (i.e., TriMatrix memory and DSP blocks) connects to row and column interconnects and has local interconnect regions driven by row and column interconnects. These blocks also have direct link interconnects for fast connections to and from a neighboring LAB. All blocks are fed by the row LAB clocks, `labclk[7..0]`.

Table 2–2 shows the Stratix device's routing scheme.

Table 2–2. Stratix Device Routing Scheme

Source	Destination																
	LUT Chain	Register Chain	Local Interconnect	Direct Link Interconnect	R4 Interconnect	R8 Interconnect	R24 Interconnect	C4 Interconnect	C8 Interconnect	C16 Interconnect	LE	M512 RAM Block	M4K RAM Block	M-RAM Block	DSP Blocks	Column IOE	Row IOE
LUT Chain											v						
Register Chain											v						
Local Interconnect											v	v	v	v	v	v	v
Direct Link Interconnect			v														
R4 Interconnect			v		v		v	v		v							
R8 Interconnect			v			v			v								
R24 Interconnect					v		v	v		v							
C4 Interconnect			v		v			v									
C8 Interconnect			v			v			v								
C16 Interconnect					v		v	v		v							
LE	v	v	v	v	v	v		v	v								
M512 RAM Block			v	v	v	v		v	v								
M4K RAM Block			v	v	v	v		v	v								
M-RAM Block								v	v								
DSP Blocks			v	v	v	v		v	v								
Column IOE				v				v	v	v							
Row IOE				v		v	v	v	v	v							

TriMatrix Memory

TriMatrix memory consists of three types of RAM blocks: M512, M4K, and M-RAM blocks. Although these memory blocks are different, they can all implement various types of memory with or without parity, including true dual-port, simple dual-port, and single-port RAM, ROM, and FIFO buffers. Table 2–3 shows the size and features of the different RAM blocks.

Table 2–3. TriMatrix Memory Features (Part 1 of 2)

Memory Feature	M512 RAM Block (32 × 18 Bits)	M4K RAM Block (128 × 36 Bits)	M-RAM Block (4K × 144 Bits)
Maximum performance	*(1)*	*(1)*	*(1)*
True dual-port memory		v	v
Simple dual-port memory	v	v	v
Single-port memory	v	v	v
Shift register	v	v	
ROM	v	v	*(2)*
FIFO buffer	v	v	v
Byte enable		v	v
Parity bits	v	v	v
Mixed clock mode	v	v	v
Memory initialization	v	v	
Simple dual-port memory mixed width support	v	v	v
True dual-port memory mixed width support		v	v
Power-up conditions	Outputs cleared	Outputs cleared	Outputs unknown
Register clears	Input and output registers	Input and output registers	Output registers
Mixed-port read-during-write	Unknown output/old data	Unknown output/old data	Unknown output

Table 2–3. TriMatrix Memory Features (Part 2 of 2)

Memory Feature	M512 RAM Block (32 × 18 Bits)	M4K RAM Block (128 × 36 Bits)	M-RAM Block (4K × 144 Bits)
Configurations	512 × 1 256 × 2 128 × 4 64 × 8 64 × 9 32 × 16 32 × 18	4K × 1 2K × 2 1K × 4 512 × 8 512 × 9 256 × 16 256 × 18 128 × 32 128 × 36	64K × 8 64K × 9 32K × 16 32K × 18 16K × 32 16K × 36 8K × 64 8K × 72 4K × 128 4K × 144

Notes to Table 2–3:

(1) See Table 4–36 for maximum performance information.

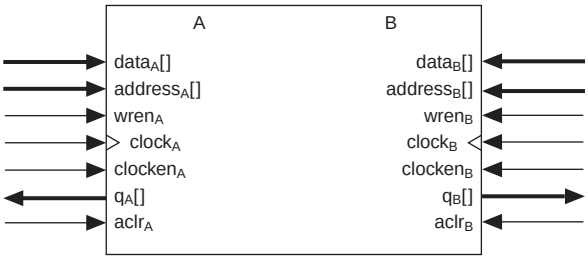
(2) The M-RAM block does not support memory initializations. However, the M-RAM block can emulate a ROM function using a dual-port RAM bock. The Stratix device must write to the dual-port memory once and then disable the write-enable ports afterwards.

1 Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Memory Modes

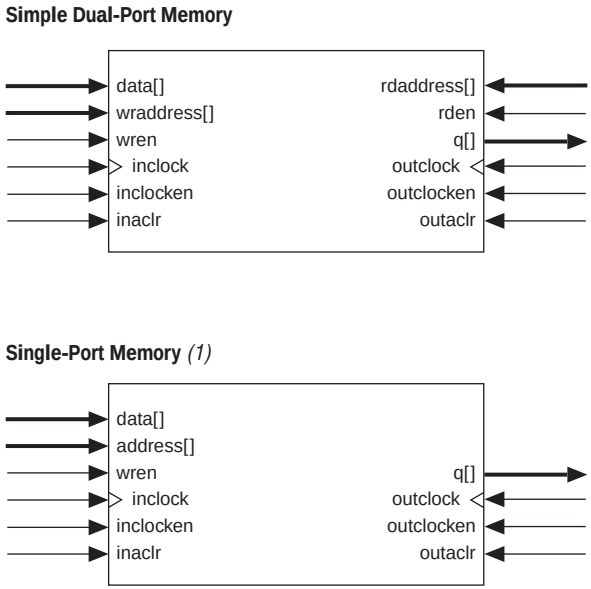
TriMatrix memory blocks include input registers that synchronize writes and output registers to pipeline designs and improve system performance. M4K and M-RAM memory blocks offer a true dual-port mode to support any combination of two-port operations: two reads, two writes, or one read and one write at two different clock frequencies. Figure 2–12 shows true dual-port memory.

Figure 2–12. True Dual-Port Memory Configuration



In addition to true dual-port memory, the memory blocks support simple dual-port and single-port RAM. Simple dual-port memory supports a simultaneous read and write and can either read old data before the write occurs or just read the don't care bits. Single-port memory supports non-simultaneous reads and writes, but the `q[]` port will output the data once it has been written to the memory (if the outputs are not registered) or after the next rising edge of the clock (if the outputs are registered). For more information, see Chapter 2, TriMatrix Embedded Memory Blocks in Stratix & Stratix GX Devices of the *Stratix Device Handbook, Volume 2*. Figure 2–13 shows these different RAM memory port configurations for TriMatrix memory.

Figure 2–13. Simple Dual-Port & Single-Port Memory Configurations



Note to Figure 2–13:

(1) Two single-port memory blocks can be implemented in a single M4K block as long as each of the two independent block sizes is equal to or less than half of the M4K block size.

The memory blocks also enable mixed-width data ports for reading and writing to the RAM ports in dual-port RAM configuration. For example, the memory block can be written in ×1 mode at port A and read out in ×16 mode from port B.

TriMatrix memory architecture can implement pipelined RAM by registering both the input and output signals to the RAM block. All TriMatrix memory block inputs are registered providing synchronous write cycles. In synchronous operation, the memory block generates its own self-timed strobe write enable (`WREN`) signal derived from the global or regional clock. In contrast, a circuit using asynchronous RAM must generate the RAM `WREN` signal while ensuring its data and address signals meet setup and hold time specifications relative to the `WREN` signal. The output registers can be bypassed. Flow-through reading is possible in the simple dual-port mode of M512 and M4K RAM blocks by clocking the read enable and read address registers on the negative clock edge and bypassing the output registers.

Two single-port memory blocks can be implemented in a single M4K block as long as each of the two independent block sizes is equal to or less than half of the M4K block size.

The Quartus II software automatically implements larger memory by combining multiple TriMatrix memory blocks. For example, two 256 × 16-bit RAM blocks can be combined to form a 256 × 32-bit RAM block. Memory performance does not degrade for memory blocks using the maximum number of words available in one memory block. Logical memory blocks using less than the maximum number of words use physical blocks in parallel, eliminating any external control logic that would increase delays. To create a larger high-speed memory block, the Quartus II software automatically combines memory blocks with LE control logic.

Clear Signals

When applied to input registers, the asynchronous clear signal for the TriMatrix embedded memory immediately clears the input registers. However, the output of the memory block does not show the effects until the next clock edge. When applied to output registers, the asynchronous clear signal clears the output registers and the effects are seen immediately.

Parity Bit Support

The memory blocks support a parity bit for each byte. The parity bit, along with internal LE logic, can implement parity checking for error detection to ensure data integrity. You can also use parity-size data words to store user-specified control bits. In the M4K and M-RAM blocks, byte enables are also available for data input masking during write operations.

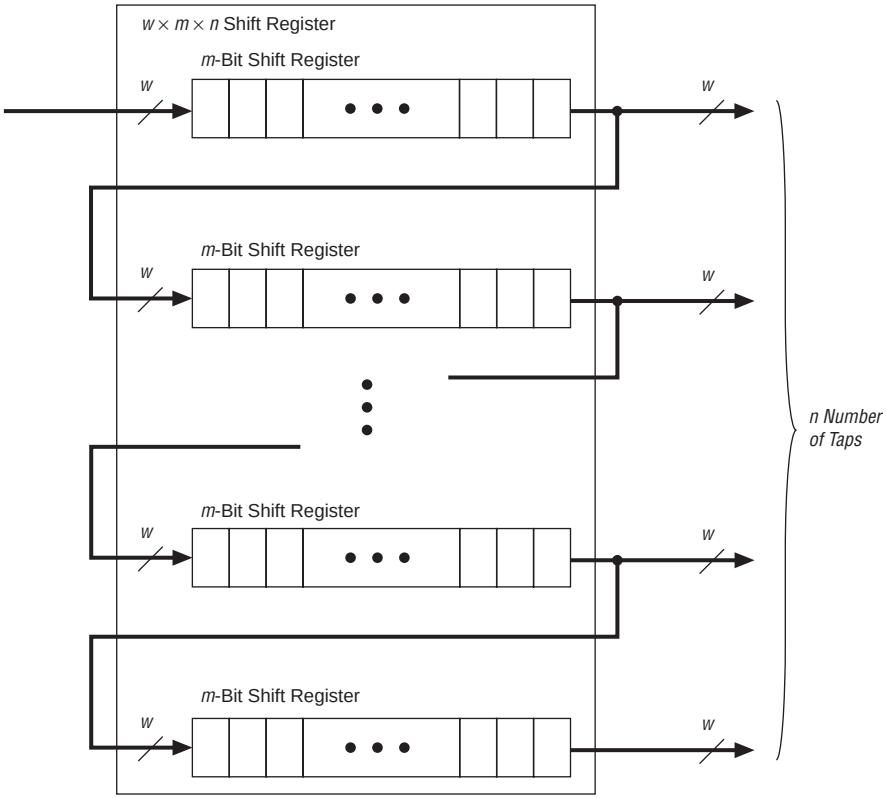
Shift Register Support

You can configure embedded memory blocks to implement shift registers for DSP applications such as pseudo-random number generators, multi-channel filtering, auto-correlation, and cross-correlation functions. These and other DSP applications require local data storage, traditionally implemented with standard flip-flops, which can quickly consume many logic cells and routing resources for large shift registers. A more efficient alternative is to use embedded memory as a shift register block, which saves logic cell and routing resources and provides a more efficient implementation with the dedicated circuitry.

The size of a $w \times m \times n$ shift register is determined by the input data width (w), the length of the taps (m), and the number of taps (n). The size of a $w \times m \times n$ shift register must be less than or equal to the maximum number of memory bits in the respective block: 576 bits for the M512 RAM block and 4,608 bits for the M4K RAM block. The total number of shift register outputs (number of taps $n \times$ width w) must be less than the maximum data width of the RAM block (18 for M512 blocks, 36 for M4K blocks). To create larger shift registers, the memory blocks are cascaded together.

Data is written into each address location at the falling edge of the clock and read from the address at the rising edge of the clock. The shift register mode logic automatically controls the positive and negative edge clocking to shift the data in one clock cycle. Figure 2–14 shows the TriMatrix memory block in the shift register mode.

Figure 2–14. Shift Register Memory Configuration



Memory Block Size

TriMatrix memory provides three different memory sizes for efficient application support. The large number of M512 blocks are ideal for designs with many shallow first-in first-out (FIFO) buffers. M4K blocks provide additional resources for channelized functions that do not require large amounts of storage. The M-RAM blocks provide a large single block of RAM ideal for data packet storage. The different-sized blocks allow Stratix devices to efficiently support variable-sized memory in designs.

The Quartus II software automatically partitions the user-defined memory into the embedded memory blocks using the most efficient size combinations. You can also manually assign the memory to a specific block size or a mixture of block sizes.

M512 RAM Block

The M512 RAM block is a simple dual-port memory block and is useful for implementing small FIFO buffers, DSP, and clock domain transfer applications. Each block contains 576 RAM bits (including parity bits). M512 RAM blocks can be configured in the following modes:

- Simple dual-port RAM
- Single-port RAM
- FIFO
- ROM
- Shift register

When configured as RAM or ROM, you can use an initialization file to pre-load the memory contents.

The memory address depths and output widths can be configured as 512 × 1, 256 × 2, 128 × 4, 64 × 8 (64 × 9 bits with parity), and 32 × 16 (32 × 18 bits with parity). Mixed-width configurations are also possible, allowing different read and write widths. Table 2–4 summarizes the possible M512 RAM block configurations.

Table 2–4. M512 RAM Block Configurations (Simple Dual-Port RAM)

Read Port	Write Port						
	512 × 1	256 × 2	128 × 4	64 × 8	32 × 16	64 × 9	32 × 18
512 × 1	v	v	v	v	v		
256 × 2	v	v	v	v	v		
128 × 4	v	v	v		v		
64 × 8	v	v		v			
32 × 16	v	v	v		v		
64 × 9						v	
32 × 18							v

When the M512 RAM block is configured as a shift register block, a shift register of size up to 576 bits is possible.

The M512 RAM block can also be configured to support serializer and deserializer applications. By using the mixed-width support in combination with DDR I/O standards, the block can function as a SERDES to support low-speed serial I/O standards using global or regional clocks. See "I/O Structure" on page 2–104 for details on dedicated SERDES in Stratix devices.

M512 RAM blocks can have different clocks on its inputs and outputs. The `wren`, `datain`, and write address registers are all clocked together from one of the two clocks feeding the block. The read address, `rden`, and output registers can be clocked by either of the two clocks driving the block. This allows the RAM block to operate in read / write or input / output clock modes. Only the output register can be bypassed. The eight `labclk` signals or local interconnect can drive the `inclock`, `outclock`, `wren`, `rden`, `inclr`, and `outclr` signals. Because of the advanced interconnect between the LAB and M512 RAM blocks, LEs can also control the `wren` and `rden` signals and the RAM clock, clock enable, and asynchronous clear signals. Figure 2–15 shows the M512 RAM block control signal generation logic.

The RAM blocks within Stratix devices have local interconnects to allow LEs and interconnects to drive into RAM blocks. The M512 RAM block local interconnect is driven by the R4, R8, C4, C8, and direct link interconnects from adjacent LABs. The M512 RAM blocks can communicate with LABs on either the left or right side through these row interconnects or with LAB columns on the left or right side with the column interconnects. Up to 10 direct link input connections to the M512 RAM block are possible from the left adjacent LABs and another 10 possible from the right adjacent LAB. M512 RAM outputs can also connect to left and right LABs through 10 direct link interconnects. The M512 RAM block has equal opportunity for access and performance to and from LABs on either its left or right side. Figure 2–16 shows the M512 RAM block to logic array interface.

Figure 2–15. M512 RAM Block Control Signals

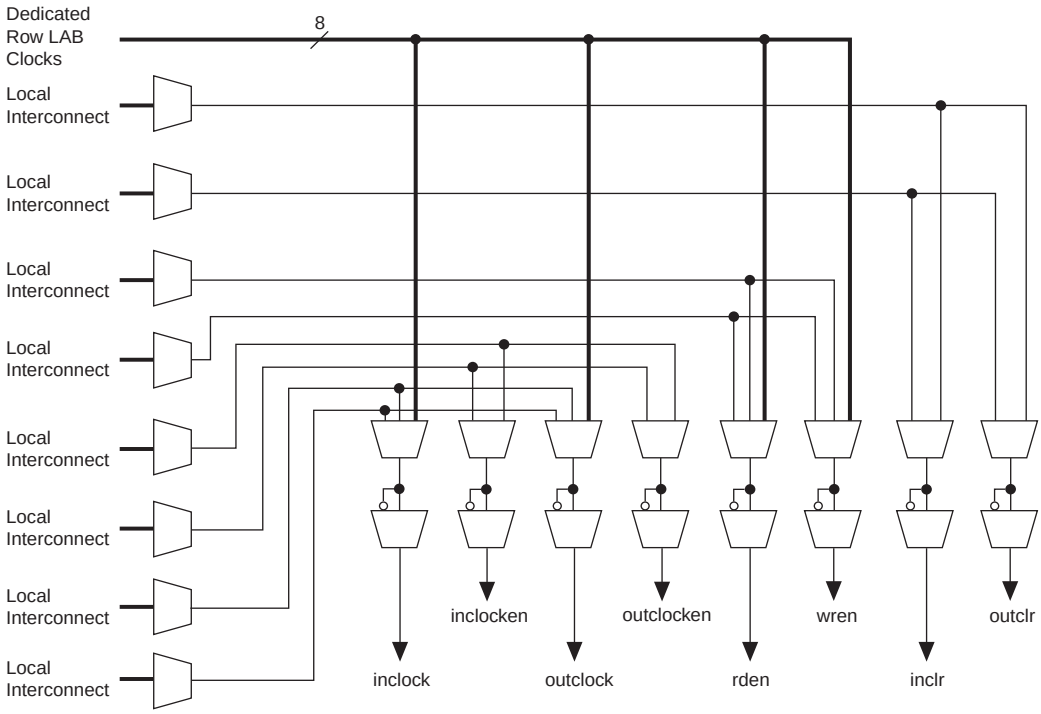
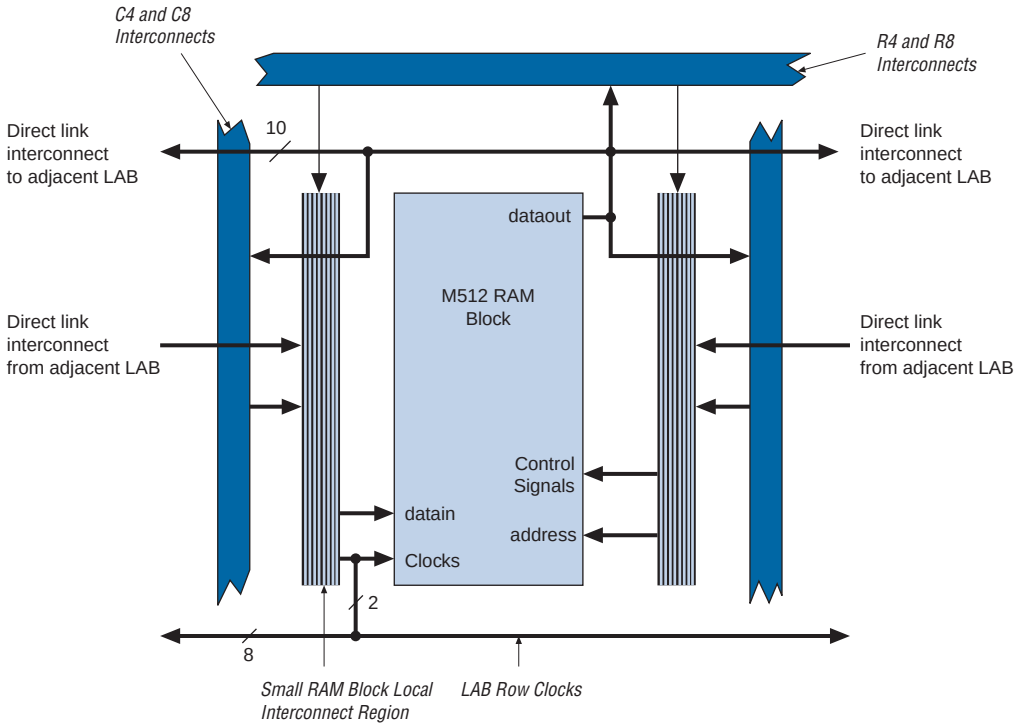


Figure 2–16. M512 RAM Block LAB Row Interface



M4K RAM Blocks

The M4K RAM block includes support for true dual-port RAM. The M4K RAM block is used to implement buffers for a wide variety of applications such as storing processor code, implementing lookup schemes, and implementing larger memory applications. Each block contains 4,608 RAM bits (including parity bits). M4K RAM blocks can be configured in the following modes:

- True dual-port RAM
- Simple dual-port RAM
- Single-port RAM
- FIFO
- ROM
- Shift register

When configured as RAM or ROM, you can use an initialization file to pre-load the memory contents.

The memory address depths and output widths can be configured as 4,096 × 1, 2,048 × 2, 1,024 × 4, 512 × 8 (or 512 × 9 bits), 256 × 16 (or 256 × 18 bits), and 128 × 32 (or 128 × 36 bits). The 128 × 32- or 36-bit configuration is not available in the true dual-port mode. Mixed-width configurations are also possible, allowing different read and write widths. Tables 2–5 and 2–6 summarize the possible M4K RAM block configurations.

Table 2–5. M4K RAM Block Configurations (Simple Dual-Port)

Read Port	Write Port								
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	128 × 32	512 × 9	256 × 18	128 × 36
4K × 1	v	v	v	v	v	v			
2K × 2	v	v	v	v	v	v			
1K × 4	v	v	v	v	v	v			
512 × 8	v	v	v	v	v	v			
256 × 16	v	v	v	v	v	v			
128 × 32	v	v	v	v	v	v			
512 × 9							v	v	v
256 × 18							v	v	v
128 × 36							v	v	v

Table 2–6. M4K RAM Block Configurations (True Dual-Port)

Port A	Port B						
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	512 × 9	256 × 18
4K × 1	v	v	v	v	v		
2K × 2	v	v	v	v	v		
1K × 4	v	v	v	v	v		
512 × 8	v	v	v	v	v		
256 × 16	v	v	v	v	v		
512 × 9						v	v
256 × 18						v	v

When the M4K RAM block is configured as a shift register block, you can create a shift register up to 4,608 bits ($w \times m \times n$).

M4K RAM blocks support byte writes when the write port has a data width of 16, 18, 32, or 36 bits. The byte enables allow the input data to be masked so the device can write to specific bytes. The unwritten bytes retain the previous written value. Table 2–7 summarizes the byte selection.

Table 2–7. Byte Enable for M4K Blocks *Notes (1), (2)*

byteena[3..0]	datain ×18	datain ×36
[0] = 1	[8..0]	[8..0]
[1] = 1	[17..9]	[17..9]
[2] = 1	–	[26..18]
[3] = 1	–	[35..27]

Notes to Table 2–7:

(1) Any combination of byte enables is possible.

(2) Byte enables can be used in the same manner with 8-bit words, i.e., in ×16 and ×32 modes.

The M4K RAM blocks allow for different clocks on their inputs and outputs. Either of the two clocks feeding the block can clock M4K RAM block registers (`renwe`, address, byte enable, `datain`, and output registers). Only the output register can be bypassed. The eight `labclk` signals or local interconnects can drive the control signals for the A and B ports of the M4K RAM block. LEs can also control the `clock_a`, `clock_b`, `renwe_a`, `renwe_b`, `clr_a`, `clr_b`, `clocken_a`, and `clocken_b` signals, as shown in Figure 2–17.

The R4, R8, C4, C8, and direct link interconnects from adjacent LABs drive the M4K RAM block local interconnect. The M4K RAM blocks can communicate with LABs on either the left or right side through these row resources or with LAB columns on either the right or left with the column resources. Up to 10 direct link input connections to the M4K RAM Block are possible from the left adjacent LABs and another 10 possible from the right adjacent LAB. M4K RAM block outputs can also connect to left and right LABs through 10 direct link interconnects each. Figure 2–18 shows the M4K RAM block to logic array interface.

Figure 2–17. M4K RAM Block Control Signals

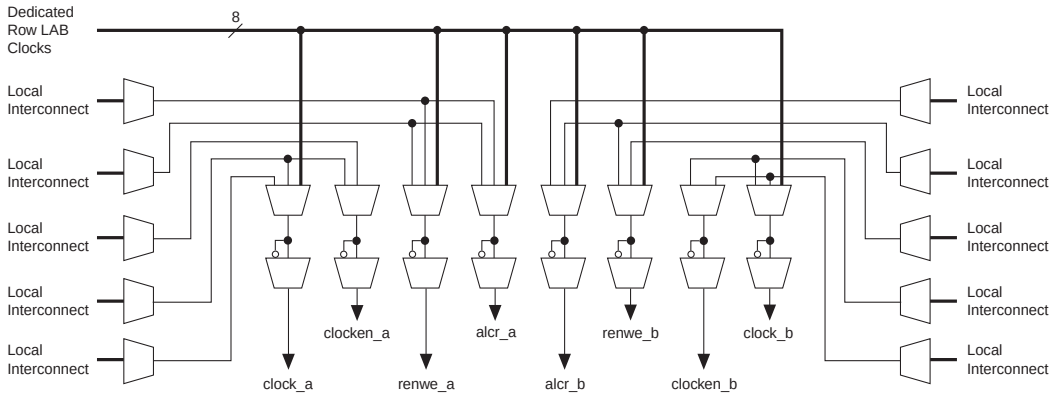
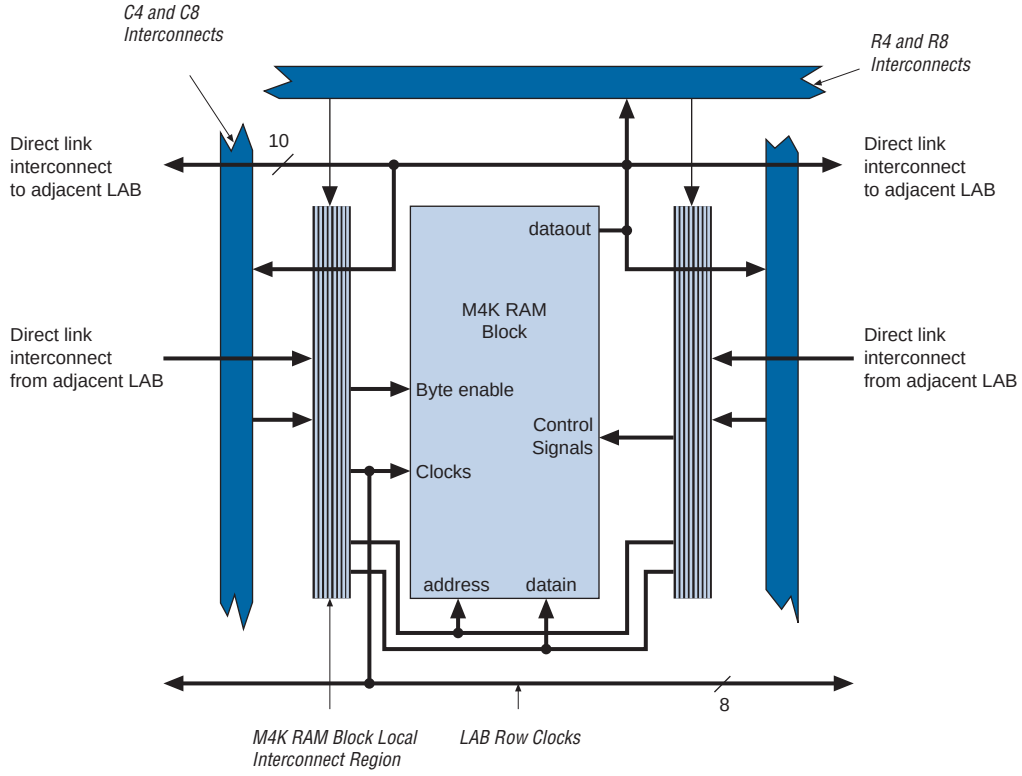


Figure 2–18. M4K RAM Block LAB Row Interface



M-RAM Block

The largest TriMatrix memory block, the M-RAM block, is useful for applications where a large volume of data must be stored on-chip. Each block contains 589,824 RAM bits (including parity bits). The M-RAM block can be configured in the following modes:

- True dual-port RAM
- Simple dual-port RAM
- Single-port RAM
- FIFO RAM

You cannot use an initialization file to initialize the contents of a M-RAM block. All M-RAM block contents power up to an undefined value. Only synchronous operation is supported in the M-RAM block, so all inputs are registered. Output registers can be bypassed. The memory address and output width can be configured as 64K × 8 (or 64K × 9 bits), 32K × 16 (or 32K × 18 bits), 16K × 32 (or 16K × 36 bits), 8K × 64 (or 8K × 72 bits), and 4K × 128 (or 4K × 144 bits). The 4K × 128 configuration is unavailable in true dual-port mode because there are a total of 144 data output drivers in the block. Mixed-width configurations are also possible, allowing different read and write widths. Tables 2–8 and 2–9 summarize the possible M-RAM block configurations:

Table 2–8. M-RAM Block Configurations (Simple Dual-Port)

Read Port	Write Port				
	64K × 9	**32K × 18**	**16K × 36**	**8K × 72**	**4K × 144**
64K × 9	v	v	v	v	
32K × 18	v	v	v	v	
16K × 36	v	v	v	v	
8K × 72	v	v	v	v	
4K × 144					v

Table 2–9. M-RAM Block Configurations (True Dual-Port)

Port A	Port B			
	64K × 9	32K × 18	16K × 36	8K × 72
64K × 9	v	v	v	v
32K × 18	v	v	v	v
16K × 36	v	v	v	v
8K × 72	v	v	v	v

The read and write operation of the memory is controlled by the WREN signal, which sets the ports into either read or write modes. There is no separate read enable (RE) signal.

Writing into RAM is controlled by both the WREN and byte enable (byteena) signals for each port. The default value for the byteena signal is high, in which case writing is controlled only by the WREN signal. The byte enables are available for the ×18, ×36, and ×72 modes. In the ×144 simple dual-port mode, the two sets of byteena signals (byteena_a and byteena_b) are combined to form the necessary 16 byte enables. Tables 2–10 and 2–11 summarize the byte selection.

Table 2–10. Byte Enable for M-RAM Blocks Notes (1), (2)

byteena[3..0]	datain ×18	datain ×36	datain ×72
[0] = 1	[8..0]	[8..0]	[8..0]
[1] = 1	[17..9]	[17..9]	[17..9]
[2] = 1	–	[26..18]	[26..18]
[3] = 1	–	[35..27]	[35..27]
[4] = 1	–	–	[44..36]
[5] = 1	–	–	[53..45]
[6] = 1	–	–	[62..54]
[7] = 1	–	–	[71..63]

Table 2–11. M-RAM Combined Byte Selection for ×144 Mode Notes (1), (2)	
byteena[15..0]	**datain ×144**
[0] = 1	[8..0]
[1] = 1	[17..9]
[2] = 1	[26..18]
[3] = 1	[35..27]
[4] = 1	[44..36]
[5] = 1	[53..45]
[6] = 1	[62..54]
[7] = 1	[71..63]
[8] = 1	[80..72]
[9] = 1	[89..81]
[10] = 1	[98..90]
[11] = 1	[107..99]
[12] = 1	[116..108]
[13] = 1	[125..117]
[14] = 1	[134..126]
[15] = 1	[143..135]

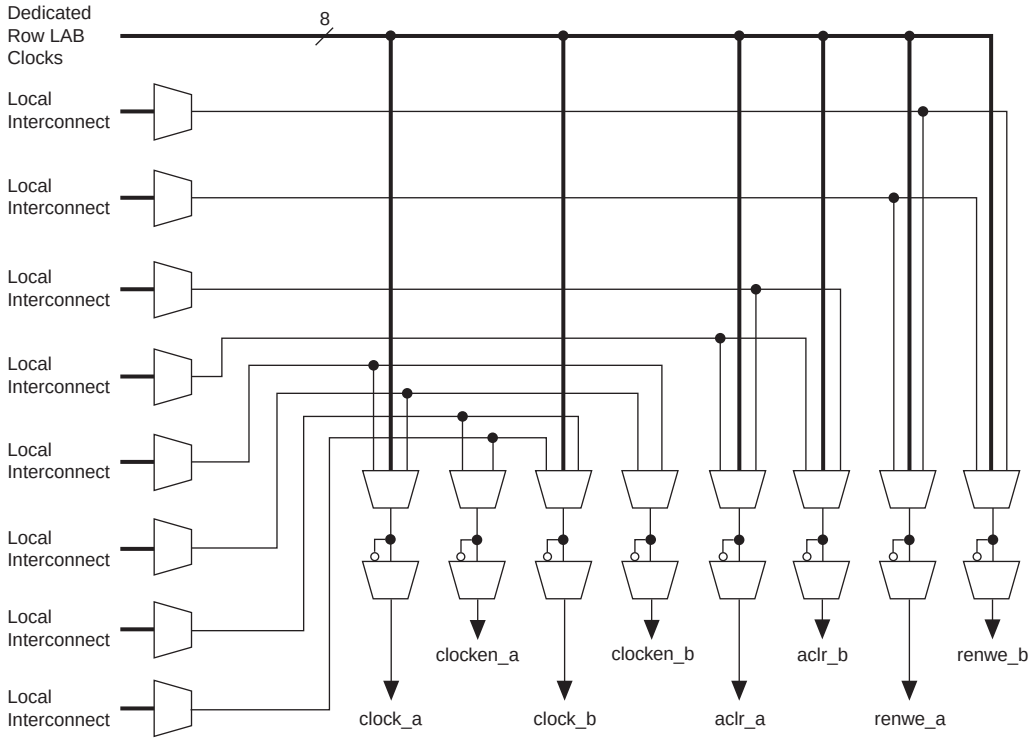
Notes to Tables 2–10 and 2–11:

(1) Any combination of byte enables is possible.

(2) Byte enables can be used in the same manner with 8-bit words, i.e., in ×16, ×32, ×64, and ×128 modes.

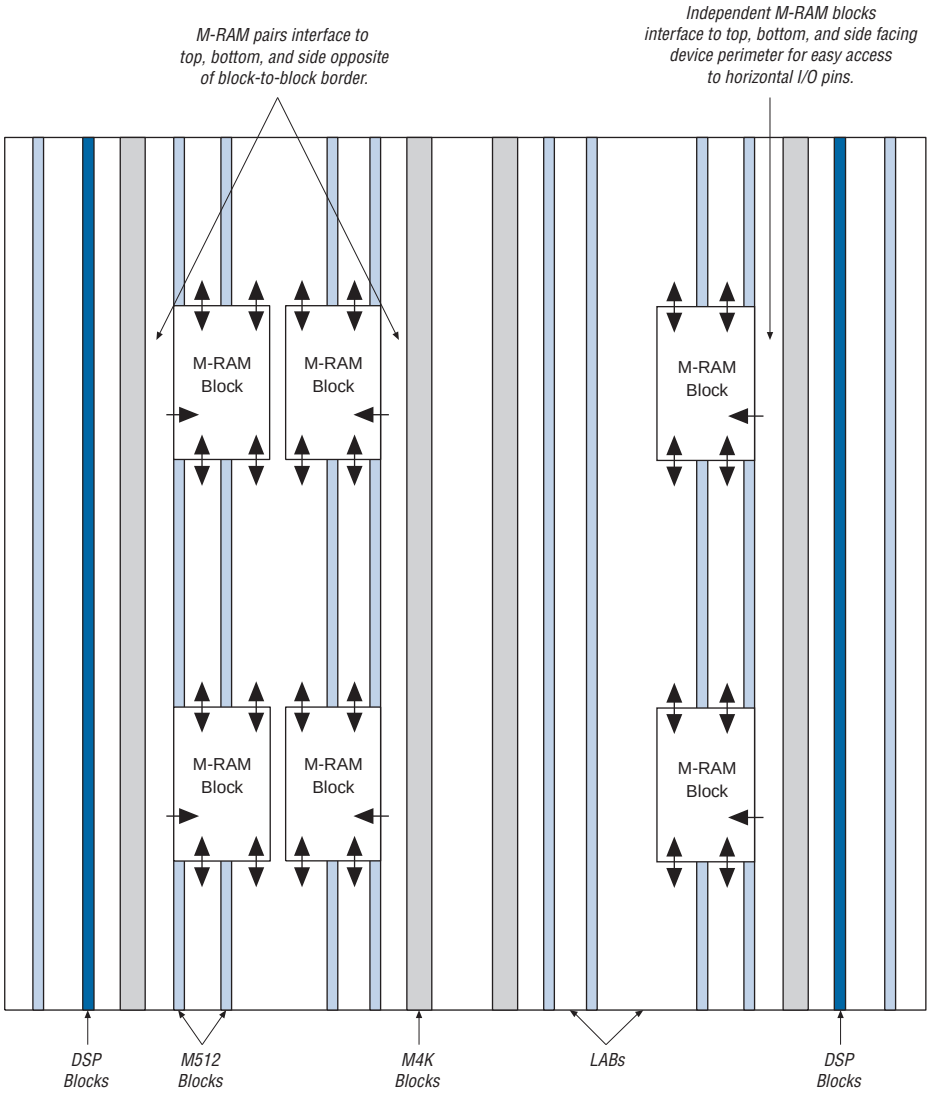
Similar to all RAM blocks, M-RAM blocks can have different clocks on their inputs and outputs. All input registers—`renwe`, `datain`, address, and byte enable registers—are clocked together from either of the two clocks feeding the block. The output register can be bypassed. The eight `labclk` signals or local interconnect can drive the control signals for the A and B ports of the M-RAM block. LEs can also control the `clock_a`, `clock_b`, `renwe_a`, `renwe_b`, `clr_a`, clr_b, `clocken_a`, and `clocken_b` signals as shown in Figure 2–19.

Figure 2–19. M-RAM Block Control Signals



One of the M-RAM block's horizontal sides drive the address and control signal (clock, renwe, byteena, etc.) inputs. Typically, the horizontal side closest to the device perimeter contains the interfaces. The one exception is when two M-RAM blocks are paired next to each other. In this case, the side of the M-RAM block opposite the common side of the two blocks contains the input interface. The top and bottom sides of any M-RAM block contain data input and output interfaces to the logic array. The top side has 72 data inputs and 72 data outputs for port B, and the bottom side has another 72 data inputs and 72 data outputs for port A. Figure 2–20 shows an example floorplan for the EP1S60 device and the location of the M-RAM interfaces.

Figure 2–20. EP1S60 Device with M-RAM Interface Locations *Note (1)*



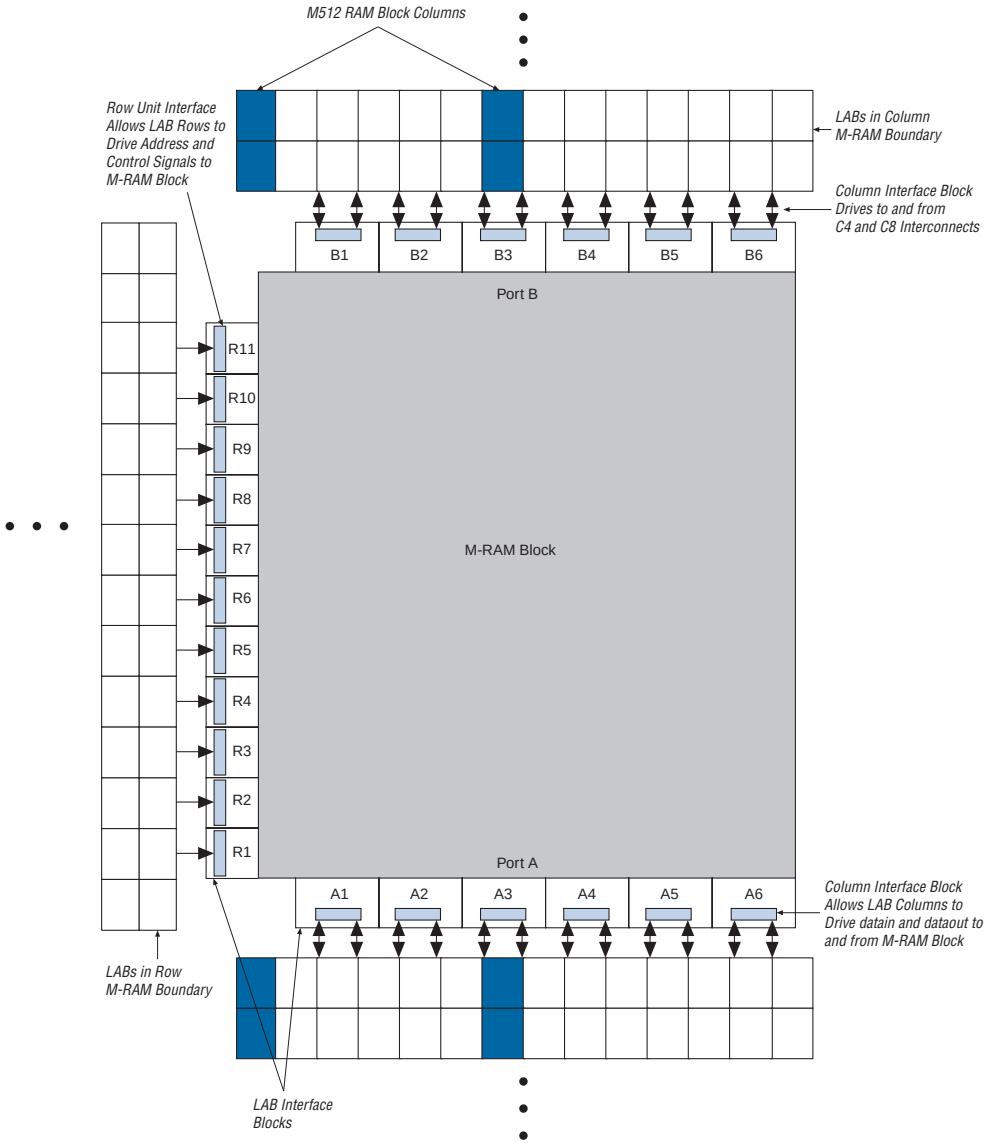
Note to Figure 2–20:

(1) Device shown is an EP1S60 device. The number and position of M-RAM blocks varies in other devices.

The M-RAM block local interconnect is driven by the R4, R8, C4, C8, and direct link interconnects from adjacent LABs. For independent M-RAM blocks, up to 10 direct link address and control signal input connections to the M-RAM block are possible from the left adjacent LABs for M-RAM

blocks facing to the left, and another 10 possible from the right adjacent LABs for M-RAM blocks facing to the right. For column interfacing, every M-RAM column unit connects to the right and left column lines, allowing each M-RAM column unit to communicate directly with three columns of LABs. Figures 2–21 through 2–23 show the interface between the M-RAM block and the logic array.

Figure 2–21. Left-Facing M-RAM to Interconnect Interface *Notes (1), (2)*



Notes to Figure 2–21:

(1) Only R24 and C16 interconnects cross the M-RAM block boundaries.

(2) The right-facing M-RAM block has interface blocks on the right side, but none on the left. B1 to B6 and A1 to A6 orientation is clipped across the vertical axis for right-facing M-RAM blocks.

Figure 2–22. M-RAM Row Unit Interface to Interconnect

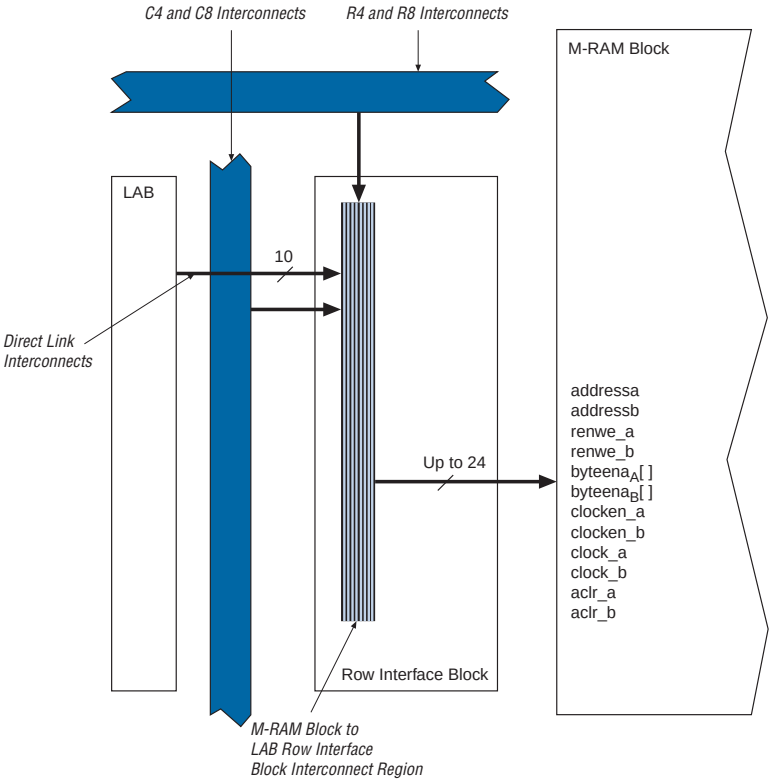


Figure 2–23. M-RAM Column Unit Interface to Interconnect

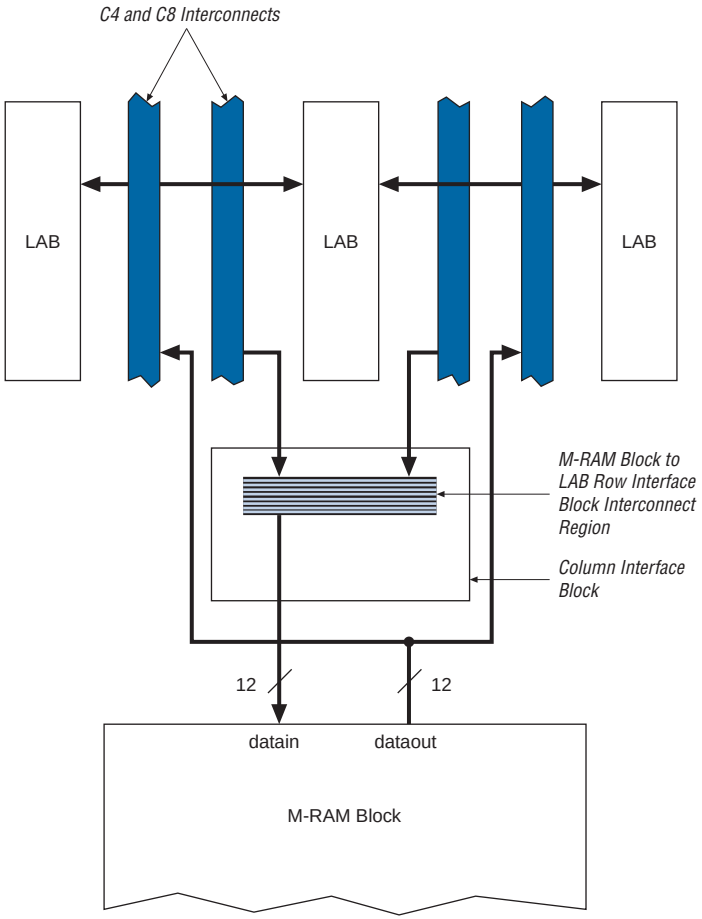


Table 2–12 shows the input and output data signal connections for the column units (B1 to B6 and A1 to A6). It also shows the address and control signal input connections to the row units (R1 to R11).

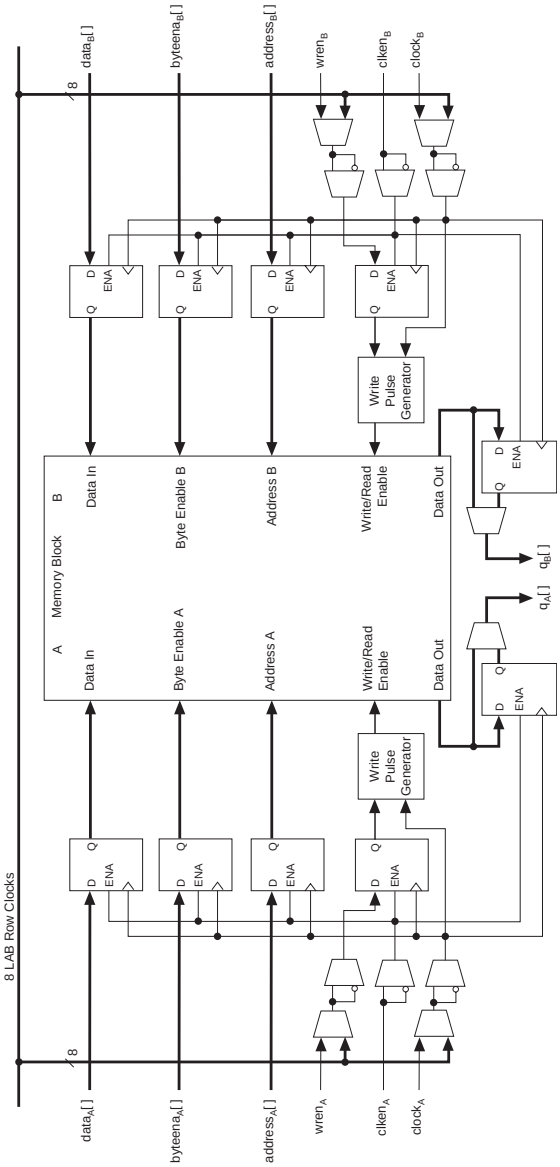
Table 2–12. M-RAM Row & Column Interface Unit Signals

Unit Interface Block	Input SIgnals	Output Signals
R1	addressa[7..0]	
R2	addressa[15..8]	
R3	byte_enable_a[7..0] renwe_a	
R4	-	
R5	-	
R6	clock_a clocken_a clock_b clocken_b	
R7	-	
R8	-	
R9	byte_enable_b[7..0] renwe_b	
R10	addressb[15..8]	
R11	addressb[7..0]	
B1	datain_b[71..60]	dataout_b[71..60]
B2	datain_b[59..48]	dataout_b[59..48]
B3	datain_b[47..36]	dataout_b[47..36]
B4	datain_b[35..24]	dataout_b[35..24]
B5	datain_b[23..12]	dataout_b[23..12]
B6	datain_b[11..0]	dataout_b[11..0]
A1	datain_a[71..60]	dataout_a[71..60]
A2	datain_a[59..48]	dataout_a[59..48]
A3	datain_a[47..36]	dataout_a[47..36]
A4	datain_a[35..24]	dataout_a[35..24]
A5	datain_a[23..12]	dataout_a[23..12]
A6	datain_a[11..0]	dataout_a[11..0]

Independent Clock Mode

The memory blocks implement independent clock mode for true dual-port memory. In this mode, a separate clock is available for each port (ports A and B). Clock A controls all registers on the port A side, while clock B controls all registers on the port B side. Each port, A and B, also supports independent clock enables and asynchronous clear signals for port A and B registers. Figure 2–24 shows a TriMatrix memory block in independent clock mode.

Figure 2–24. Independent Clock Mode *Notes (1), (2)*



Notes to Figure 2–24

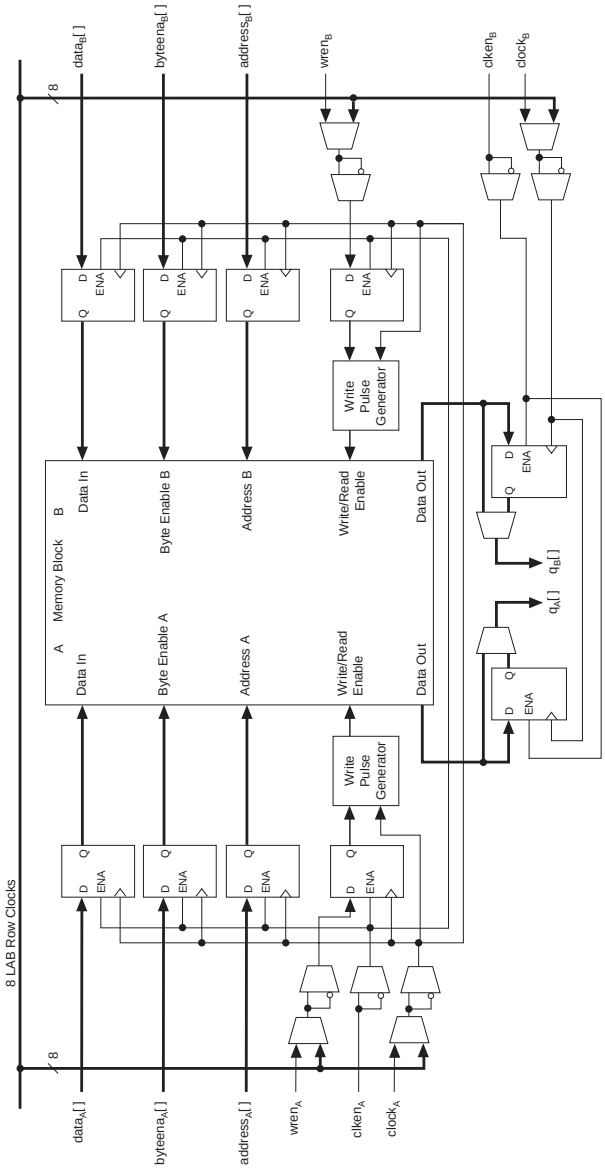
(1) All registers shown have asynchronous clear ports.

(2) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Input/Output Clock Mode

Input/output clock mode can be implemented for both the true and simple dual-port memory modes. On each of the two ports, A or B, one clock controls all registers for inputs into the memory block: data input, `wren`, and address. The other clock controls the block's data output registers. Each memory block port, A or B, also supports independent clock enables and asynchronous clear signals for input and output registers. Figures 2–25 and 2–26 show the memory block in input/output clock mode.

Figure 2–25. Input/Output Clock Mode in True Dual-Port Mode *Notes (1), (2)*

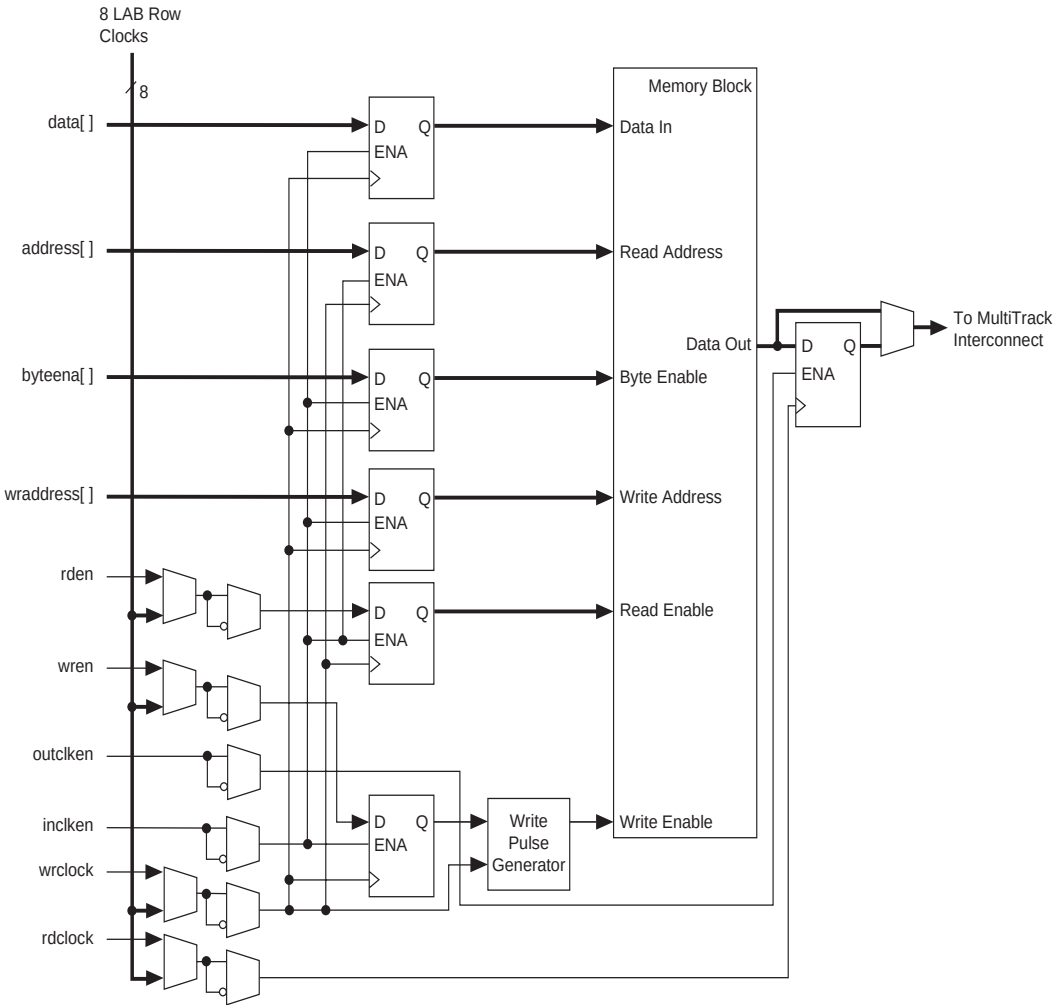


Notes to Figure 2–25:

(1) All registers shown have asynchronous clear ports.

(2) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Figure 2–26. Input/Output Clock Mode in Simple Dual-Port Mode *Notes (1), (2)*



Notes to Figure 2–26:

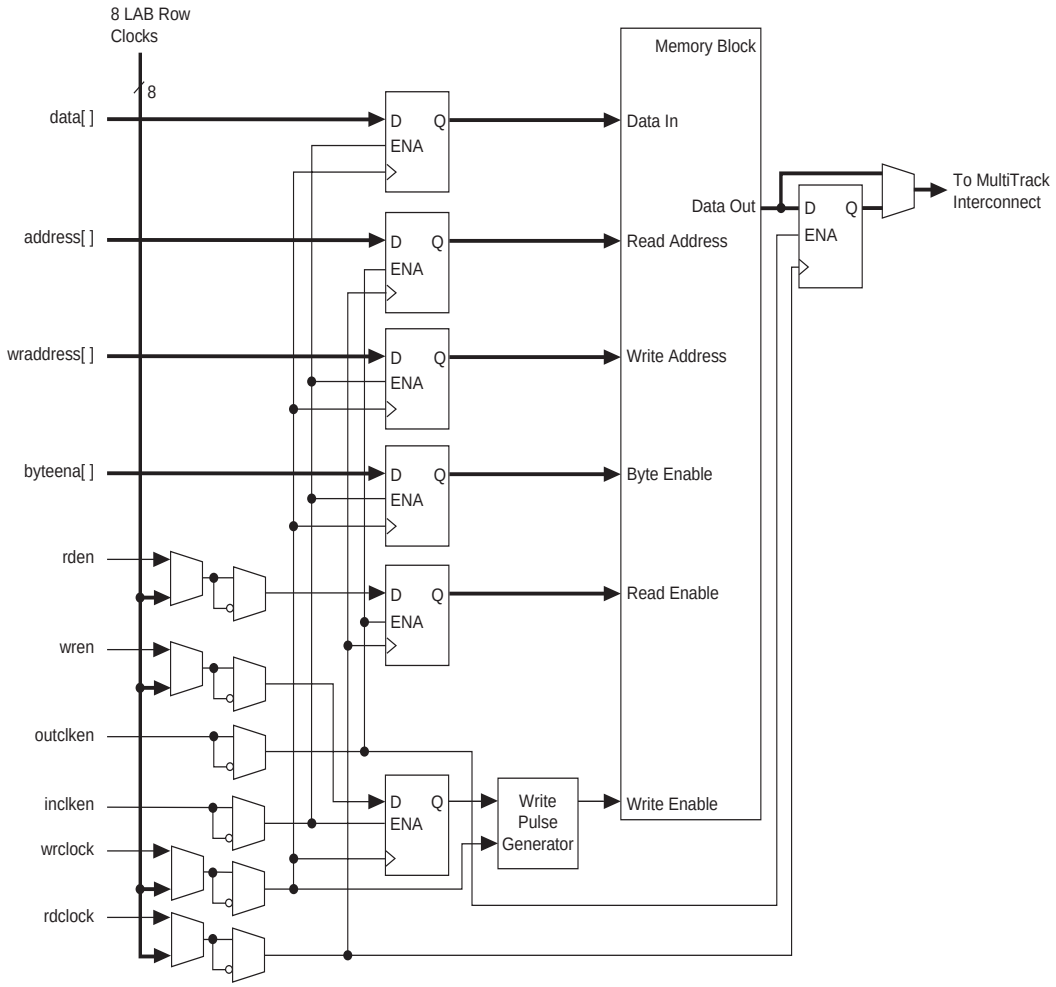
(1) All registers shown except the `rden` register have asynchronous clear ports.

(2) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Read/Write Clock Mode

The memory blocks implement read / write clock mode for simple dual-port memory. You can use up to two clocks in this mode. The write clock controls the block's data inputs, `wraddress`, and `wren`. The read clock controls the data output, `rdaddress`, and `rden`. The memory blocks support independent clock enables for each clock and asynchronous clear signals for the read- and write-side registers. Figure 2–27 shows a memory block in read / write clock mode.

Figure 2–27. Read/Write Clock Mode in Simple Dual-Port Mode *Notes (1), (2)*



Notes to Figure 2–27:

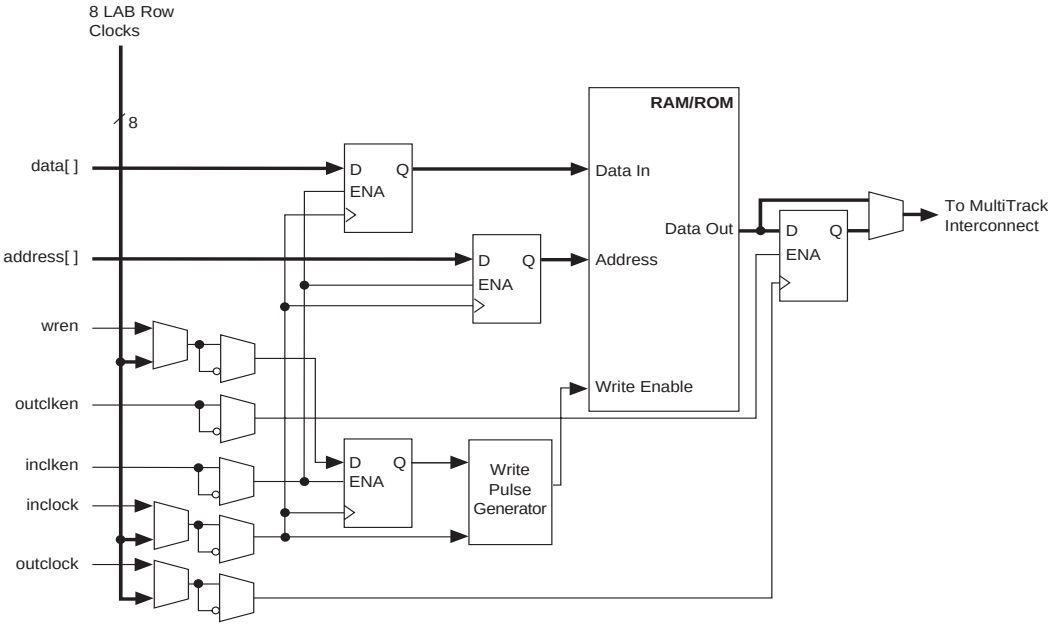
(1) All registers shown except the `rden` register have asynchronous clear ports.

(2) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Single-Port Mode

The memory blocks also support single-port mode, used when simultaneous reads and writes are not required. See Figure 2–28. A single block in a memory block can support up to two single-port mode RAM blocks in the M4K RAM blocks if each RAM block is less than or equal to 2K bits in size.

Figure 2–28. Single-Port Mode *Note (1)*



Note to Figure 2–28:

(1) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Digital Signal Processing Block

The most commonly used DSP functions are finite impulse response (FIR) filters, complex FIR filters, infinite impulse response (IIR) filters, fast Fourier transform (FFT) functions, direct cosine transform (DCT) functions, and correlators. All of these blocks have the same fundamental building block: the multiplier. Additionally, some applications need specialized operations such as multiply-add and multiply-accumulate operations. Stratix devices provide DSP blocks to meet the arithmetic requirements of these functions.

Each Stratix device has two columns of DSP blocks to efficiently implement DSP functions faster than LE-based implementations. Larger Stratix devices have more DSP blocks per column (see Table 2–13). Each DSP block can be configured to support up to:

- Eight 9 × 9-bit multipliers
- Four 18 × 18-bit multipliers
- One 36 × 36-bit multiplier

As indicated, the Stratix DSP block can support one 36 × 36-bit multiplier in a single DSP block. This is true for any matched sign multiplications (either unsigned by unsigned or signed by signed), but the capabilities for dynamic and mixed sign multiplications are handled differently. The following list provides the largest functions that can fit into a single DSP block.

- 36 × 36-bit unsigned by unsigned multiplication
- 36 × 36-bit signed by signed multiplication
- 35 × 36-bit unsigned by signed multiplication
- 36 × 35-bit signed by unsigned multiplication
- 36 × 35-bit signed by dynamic sign multiplication
- 35 × 36-bit dynamic sign by signed multiplication
- 35 × 36-bit unsigned by dynamic sign multiplication
- 36 × 35-bit dynamic sign by unsigned multiplication
- 35 × 35-bit dynamic sign multiplication when the sign controls for each operand are different
- 36 × 36-bit dynamic sign multiplication when the same sign control is used for both operands

1 This list only shows functions that can fit into a single DSP block. Multiple DSP blocks can support larger multiplication functions.

Figure 2–29 shows one of the columns with surrounding LAB rows.

Figure 2–29. DSP Blocks Arranged in Columns

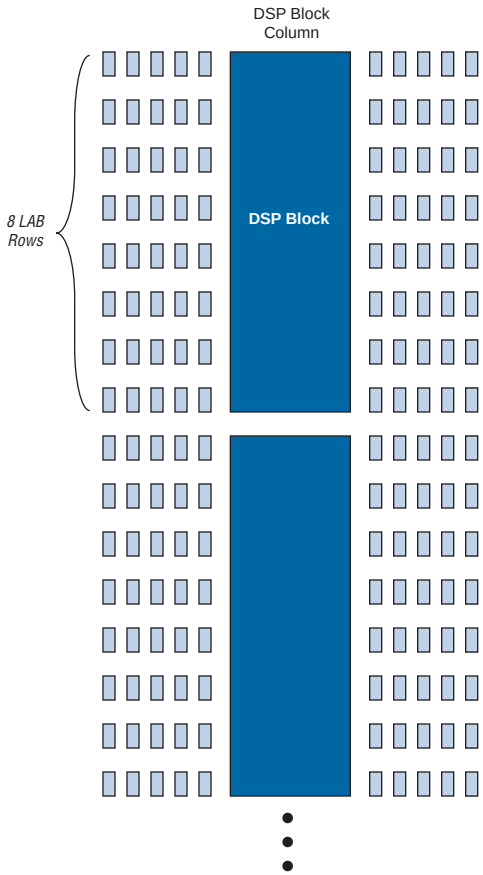


Table 2–13 shows the number of DSP blocks in each Stratix device.

Table 2–13. DSP Blocks in Stratix Devices Notes (1), (2)

Device	DSP Blocks	Total 9 × 9 Multipliers	Total 18 × 18 Multipliers	Total 36 × 36 Multipliers
EP1S10	6	48	24	6
EP1S20	10	80	40	10
EP1S25	10	80	40	10
EP1S30	12	96	48	12
EP1S40	14	112	56	14
EP1S60	18	144	72	18
EP1S80	22	176	88	22

Notes to Table 2–13:

(1) Each device has either the number of 9 × 9-, 18 × 18-, or 36 × 36-bit multipliers shown. The total number of multipliers for each device is not the sum of all the multipliers.

(2) The number of supported multiply functions shown is based on signed / signed or unsigned / unsigned implementations.

DSP block multipliers can optionally feed an adder / subtractor or accumulator within the block depending on the configuration. This makes routing to LEs easier, saves LE routing resources, and increases performance, because all connections and blocks are within the DSP block. Additionally, the DSP block input registers can efficiently implement shift registers for FIR filter applications.

Figure 2–30 shows the top-level diagram of the DSP block configured for 18 × 18-bit multiplier mode. Figure 2–31 shows the 9 × 9-bit multiplier configuration of the DSP block.

Figure 2–30. DSP Block Diagram for 18 × 18-Bit Configuration

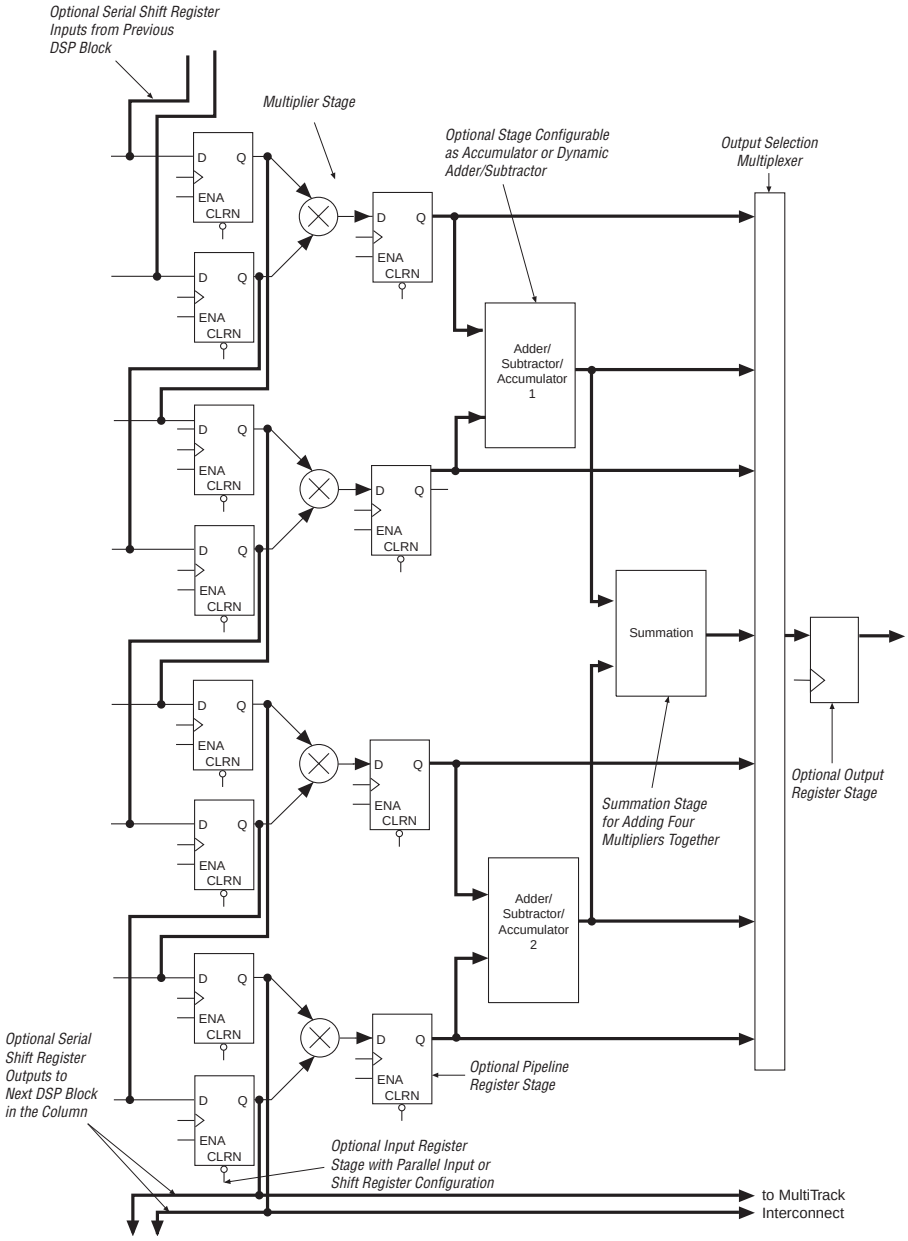
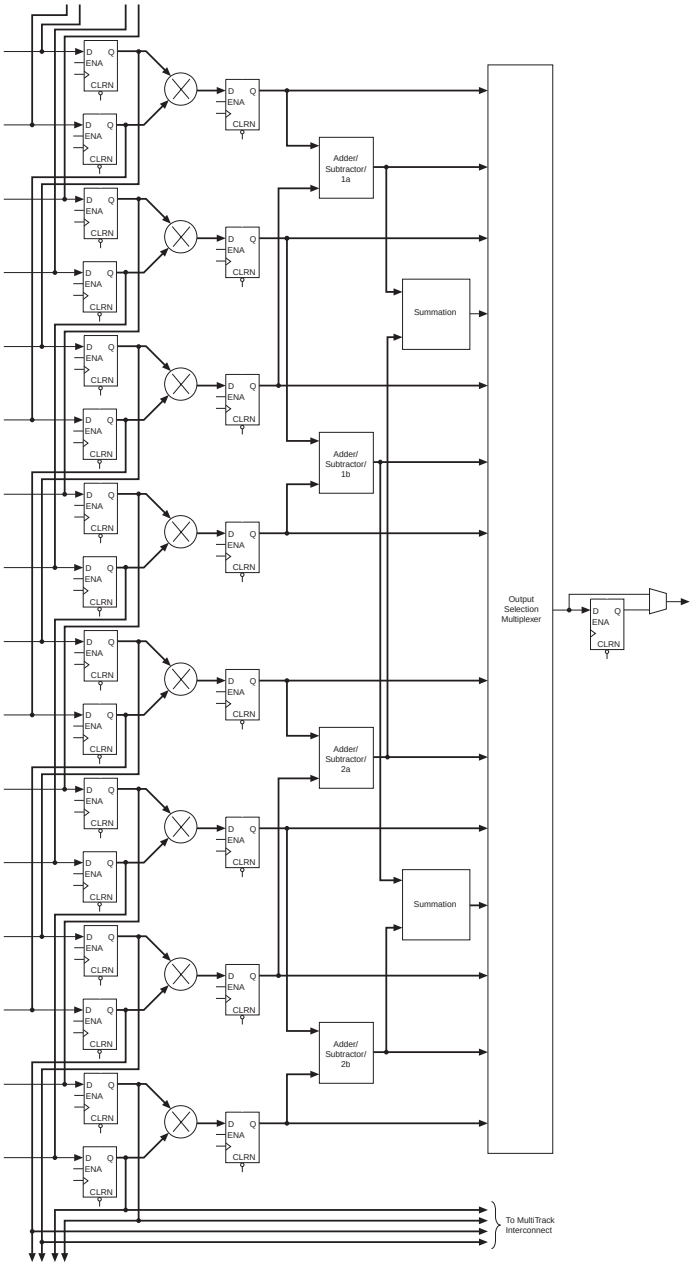


Figure 2–31. DSP Block Diagram for 9 × 9-Bit Configuration



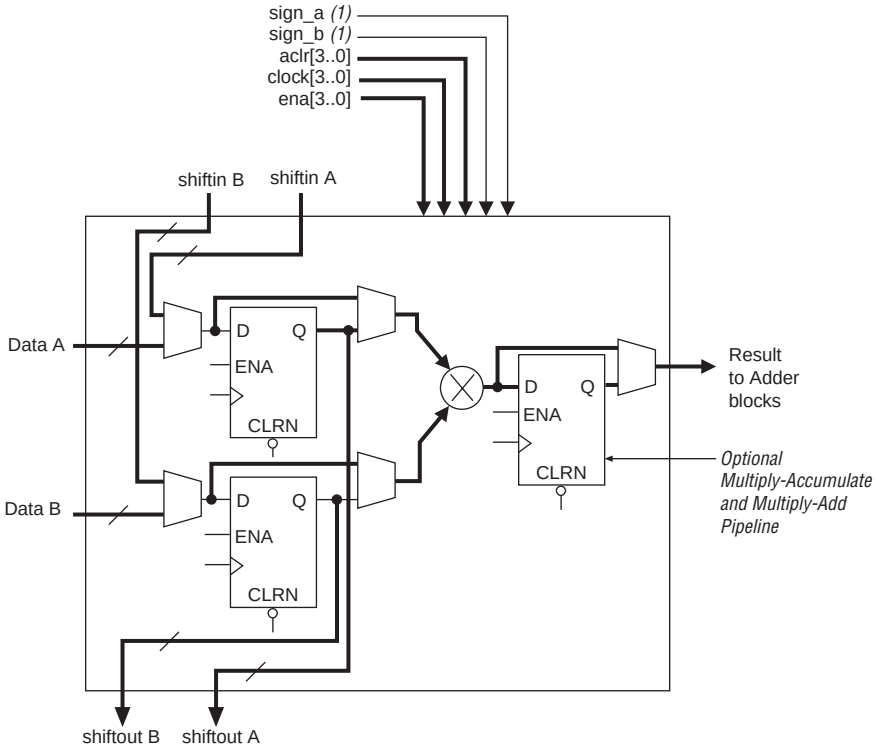
The DSP block consists of the following elements:

- Multiplier block
- Adder / output block

Multiplier Block

The DSP block multiplier block consists of the input registers, a multiplier, and pipeline register for pipelining multiply-accumulate and multiply-add / subtract functions as shown in Figure 2–32.

Figure 2–32. Multiplier Sub-Block within Stratix DSP Block



Note to Figure 2–32:

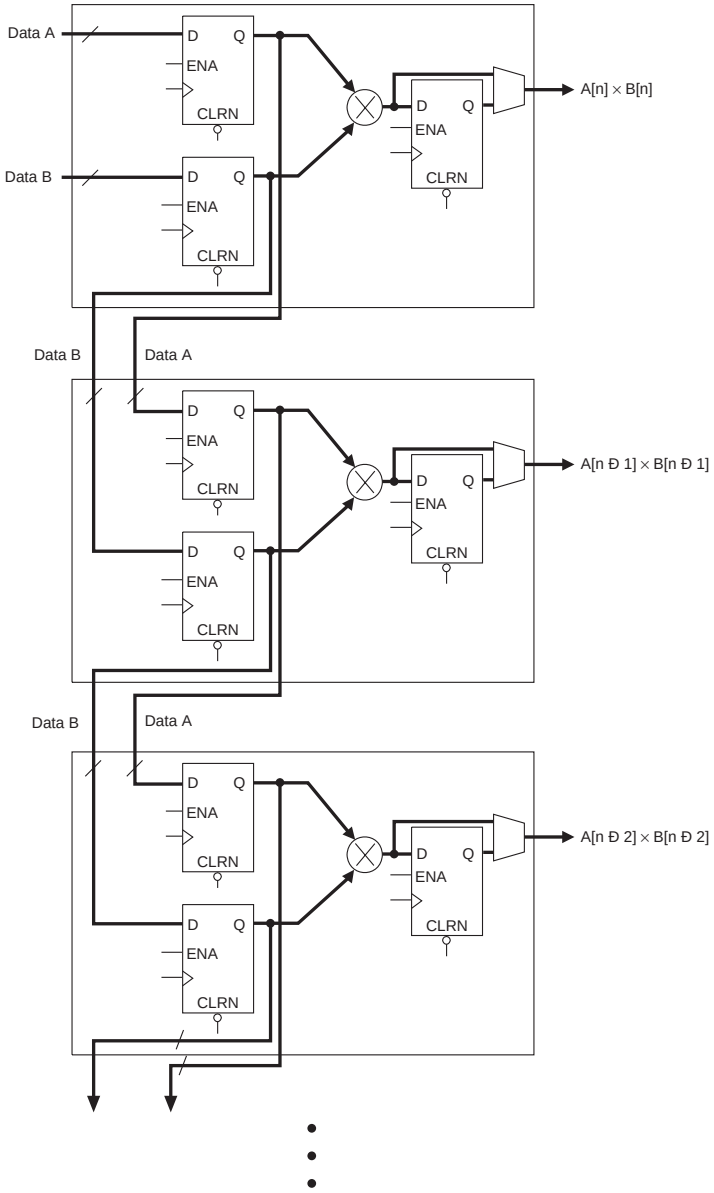
(1) These signals can be unregistered or registered once to match data path pipelines if required.

Input Registers

A bank of optional input registers is located at the input of each multiplier and multiplicand inputs to the multiplier. When these registers are configured for parallel data inputs, they are driven by regular routing resources. You can use a clock signal, asynchronous clear signal, and a clock enable signal to independently control each set of A and B inputs for each multiplier in the DSP block. You select these control signals from a set of four different `clock[3..0]`, `aclr[3..0]`, and `ena[3..0]` signals that drive the entire DSP block.

You can also configure the input registers for a shift register application. In this case, the input registers feed the multiplier and drive two dedicated shift output lines: `shiftoutA` and `shiftoutB`. The shift outputs of one multiplier block directly feed the adjacent multiplier block in the same DSP block (or the next DSP block) as shown in Figure 2–33, to form a shift register chain. This chain can terminate in any block, that is, you can create any length of shift register chain up to 224 registers. You can use the input shift registers for FIR filter applications. One set of shift inputs can provide data for a filter, and the other are coefficients that are optionally loaded in serial or parallel. When implementing 9 × 9- and 18 × 18-bit multipliers, you do not need to implement external shift registers in LAB LEs. You implement all the filter circuitry within the DSP block and its routing resources, saving LE and general routing resources for general logic. External registers are needed for shift register inputs when using 36 × 36-bit multipliers.

Figure 2–33. Multiplier Sub-Blocks Using Input Shift Register Connections
Note (1)



Note to Figure 2–33:

(1) Either Data A or Data B input can be set to a parallel input for constant coefficient multiplication.

Table 2–14 shows the summary of input register modes for the DSP block.

Table 2–14. Input Register Modes

Register Input Mode	9 × 9	18 × 18	36 × 36
Parallel input	v	v	v
Shift register input	v	v	

Multiplier

The multiplier supports 9 × 9-, 18 × 18-, or 36 × 36-bit multiplication. Each DSP block supports eight possible 9 × 9-bit or smaller multipliers. There are four multiplier blocks available for multipliers larger than 9 × 9 bits but smaller than 18 × 18 bits. There is one multiplier block available for multipliers larger than 18 × 18 bits but smaller than or equal to 36 × 36 bits. The ability to have several small multipliers is useful in applications such as video processing. Large multipliers greater than 18 × 18 bits are useful for applications such as the mantissa multiplication of a single-precision floating-point number.

The multiplier operands can be signed or unsigned numbers, where the result is signed if either input is signed as shown in Table 2–15. The `sign_a` and `sign_b` signals provide dynamic control of each operand's representation: a logic 1 indicates the operand is a signed number, a logic 0 indicates the operand is an unsigned number. These sign signals affect all multipliers and adders within a single DSP block and you can register them to match the data path pipeline. The multipliers are full precision (that is, 18 bits for the 18-bit multiply, 36-bits for the 36-bit multiply, and so on) regardless of whether `sign_a` or `sign_b` set the operands as signed or unsigned numbers.

Table 2–15. Multiplier Signed Representation

Data A	Data B	Result
Unsigned	Unsigned	Unsigned
Unsigned	Signed	Signed
Signed	Unsigned	Signed
Signed	Signed	Signed

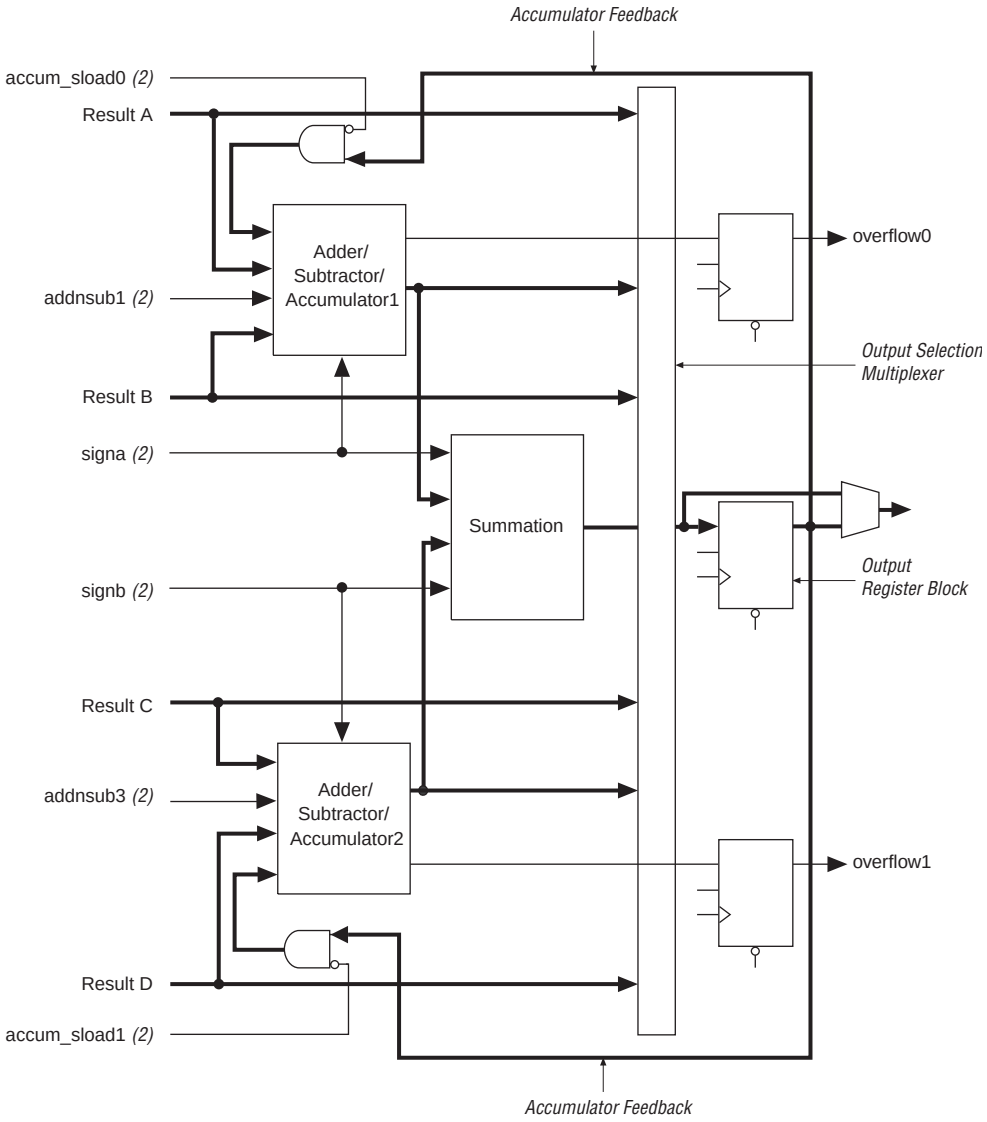
Pipeline/Post Multiply Register

The output of 9 × 9- or 18 × 18-bit multipliers can optionally feed a register to pipeline multiply-accumulate and multiply-add / subtract functions. For 36 × 36-bit multipliers, this register will pipeline the multiplier function.

Adder/Output Blocks

The result of the multiplier sub-blocks are sent to the adder / output block which consist of an adder / subtractor / accumulator unit, summation unit, output select multiplexer, and output registers. The results are used to configure the adder / output block as a pure output, accumulator, a simple two-multiplier adder, four-multiplier adder, or final stage of the 36-bit multiplier. You can configure the adder / output block to use output registers in any mode, and must use output registers for the accumulator. The system cannot use adder / output blocks independently of the multiplier. Figure 2–34 shows the adder and output stages.

Figure 2–34. Adder/Output Blocks *Note (1)*



Notes to Figure 2–34:

(1) Adder / output block shown in Figure 2–34 is in 18× 18-bit mode. In 9× 9-bit mode, there are four adder / subtractor blocks and two summation blocks.

(2) These signals are either not registered, registered once, or registered twice to match the data path pipeline.

Adder/Subtractor/Accumulator

The adder / subtractor / accumulator is the first level of the adder / output block and can be used as an accumulator or as an adder / subtractor.

Adder/Subtractor

Each adder / subtractor / accumulator block can perform addition or subtraction using the `addnsub` independent control signal for each first-level adder in 18 × 18-bit mode. There are two `addnsub[1..0]` signals available in a DSP block for any configuration. For 9 × 9-bit mode, one `addnsub[1..0]` signal controls the top two one-level adders and another `addnsub[1..0]` signal controls the bottom two one-level adders. A high `addnsub` signal indicates addition, and a low signal indicates subtraction. The `addnsub` control signal can be unregistered or registered once or twice when feeding the adder blocks to match data path pipelines.

The `signa` and `signb` signals serve the same function as the multiplier block `signa` and `signb` signals. The only difference is that these signals can be registered up to two times. These signals are tied to the same `signa` and `signb` signals from the multiplier and must be connected to the same clocks and control signals.

Accumulator

When configured for accumulation, the adder / output block output feeds back to the accumulator as shown in Figure 2–34. The `accum_sload[1..0]` signal synchronously loads the multiplier result to the accumulator output. This signal can be unregistered or registered once or twice. Additionally, the `overflow` signal indicates the accumulator has overflowed or underflowed in accumulation mode. This signal is always registered and must be externally latched in LEs if the design requires a latched `overflow` signal.

Summation

The output of the adder / subtractor / accumulator block feeds to an optional summation block. This block sums the outputs of the DSP block multipliers. In 9 × 9-bit mode, there are two summation blocks providing the sums of two sets of four 9 × 9-bit multipliers. In 18 × 18-bit mode, there is one summation providing the sum of one set of four 18 × 18-bit multipliers.

Output Selection Multiplexer

The outputs from the various elements of the adder / output block are routed through an output selection multiplexer. Based on the DSP block operational mode and user settings, the multiplexer selects whether the output from the multiplier, the adder / subtractor / accumulator, or summation block feeds to the output.

Output Registers

Optional output registers for the DSP block outputs are controlled by four sets of control signals: `clock[3..0]`, `aclr[3..0]`, and `ena[3..0]`. Output registers can be used in any mode.

Modes of Operation

The adder, subtractor, and accumulate functions of a DSP block have four modes of operation:

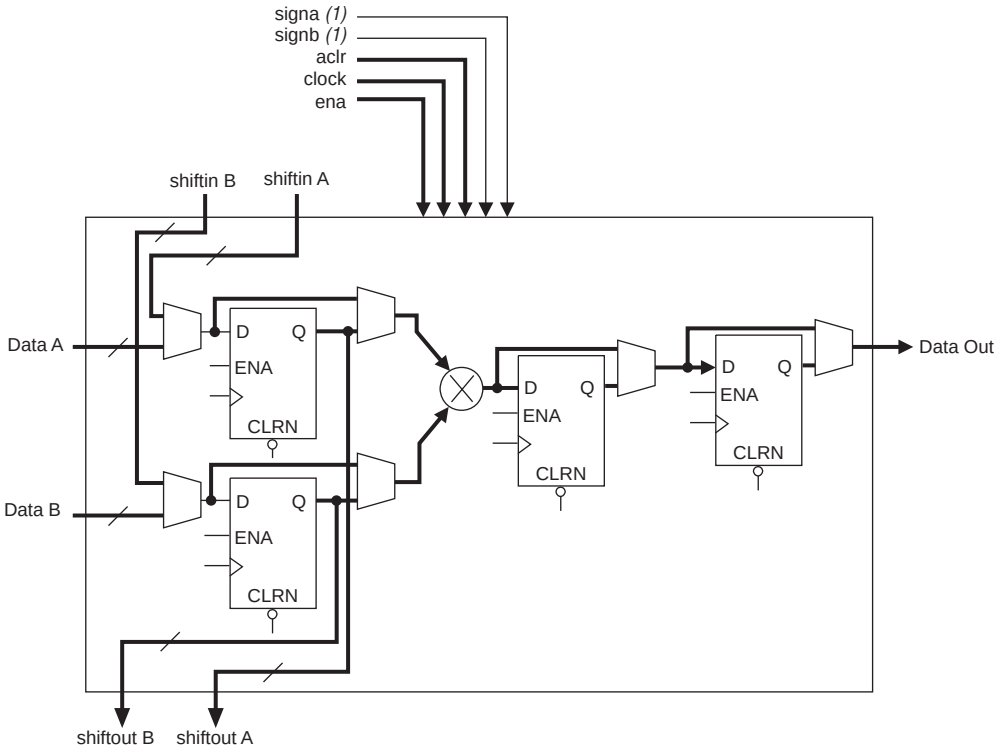
- Simple multiplier
- Multiply-accumulator
- Two-multipliers adder
- Four-multipliers adder

1 Each DSP block can only support one mode. Mixed modes in the same DSP block is not supported.

Simple Multiplier Mode

In simple multiplier mode, the DSP block drives the multiplier sub-block result directly to the output with or without an output register. Up to four 18 × 18-bit multipliers or eight 9 × 9-bit multipliers can drive their results directly out of one DSP block. See Figure 2–35.

Figure 2–35. Simple Multiplier Mode

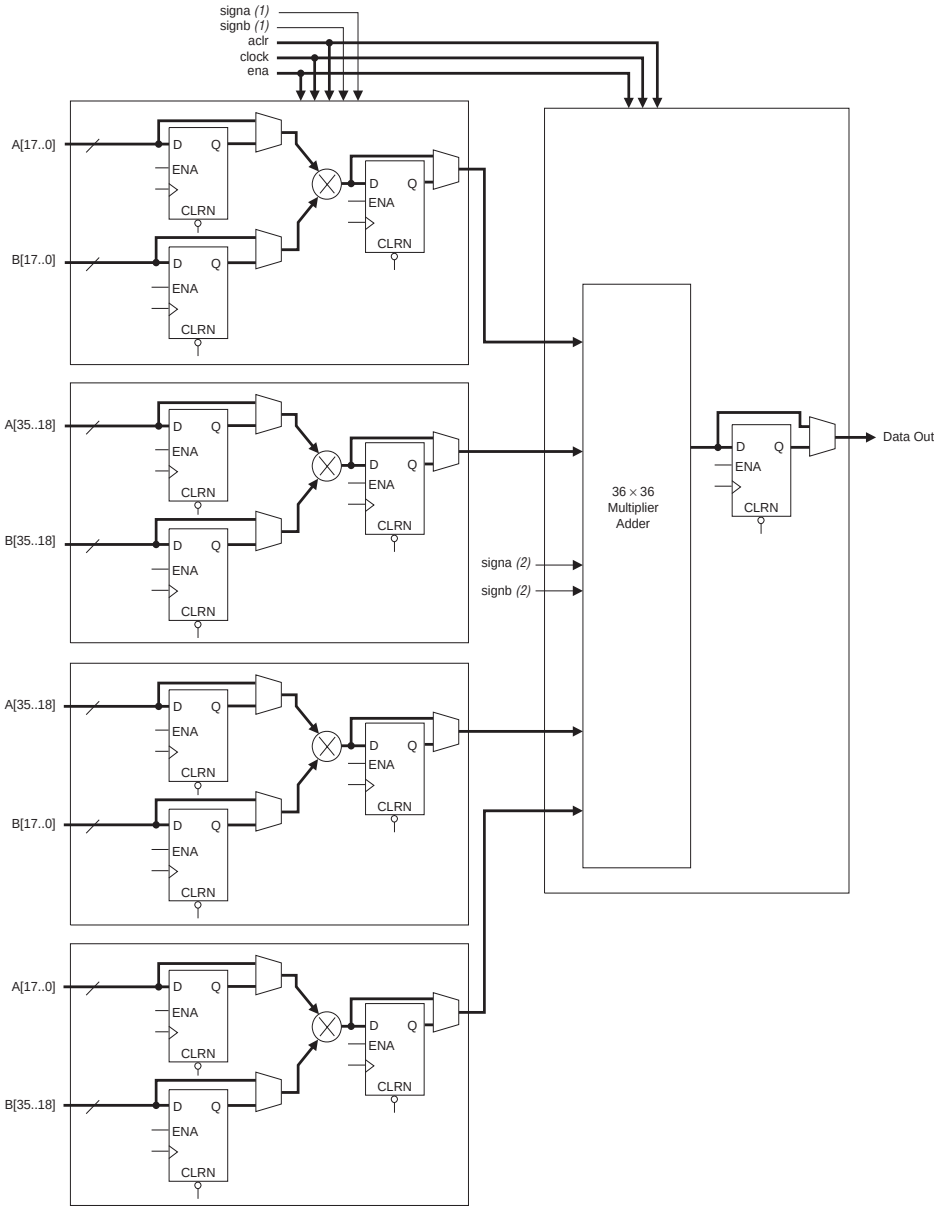


Note to Figure 2–35:

(1) These signals are not registered or registered once to match the data path pipeline.

DSP blocks can also implement one 36 × 36-bit multiplier in multiplier mode. DSP blocks use four 18 × 18-bit multipliers combined with dedicated adder and internal shift circuitry to achieve 36-bit multiplication. The input shift register feature is not available for the 36 × 36-bit multiplier. In 36 × 36-bit mode, the device can use the register that is normally a multiplier-result-output register as a pipeline stage for the 36 × 36-bit multiplier. Figure 2–36 shows the 36 × 36-bit multiply mode.

Figure 2–36. 36 × 36 Multiply Mode



Notes to Figure 2–36:

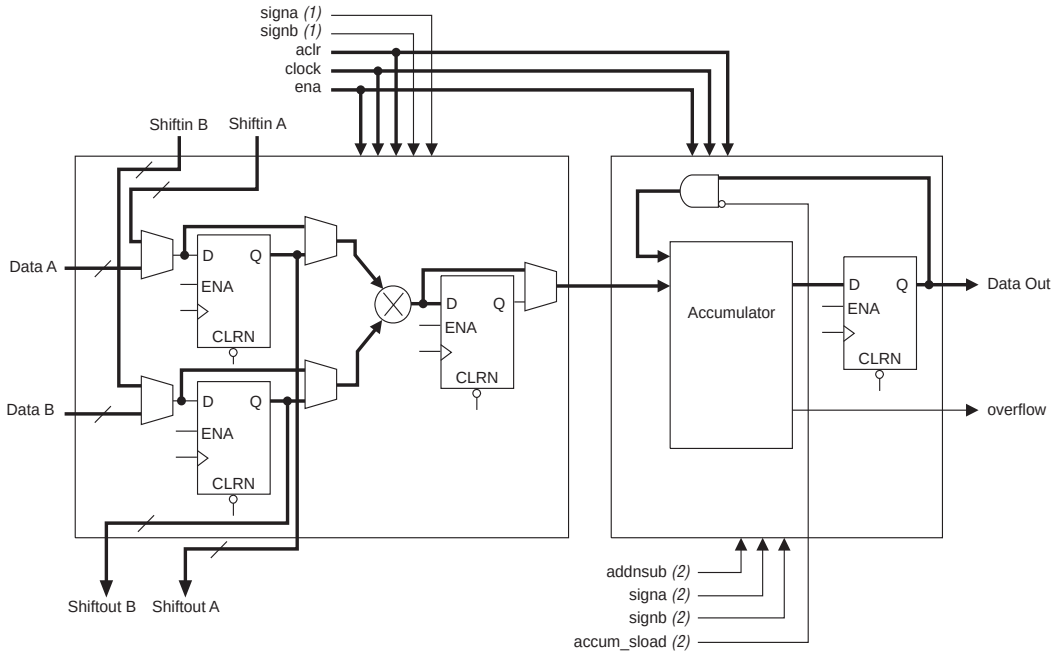
(1) These signals are not registered or registered once to match the pipeline.

(2) These signals are not registered, registered once, or registered twice for latency to match the pipeline.

Multiply-Accumulator Mode

In multiply-accumulator mode (see Figure 2–37), the DSP block drives multiplied results to the adder / subtractor / accumulator block configured as an accumulator. You can implement one or two multiply-accumulators up to 18 × 18 bits in one DSP block. The first and third multiplier sub-blocks are unused in this mode, because only one multiplier can feed one of two accumulators. The multiply-accumulator output can be up to 52 bits—a maximum of a 36-bit result with 16 bits of accumulation. The `accum_sload` and `overflow` signals are only available in this mode. The `addnsub` signal can set the accumulator for decimation and the `overflow` signal indicates underflow condition.

Figure 2–37. Multiply-Accumulate Mode



Notes to Figure 2–37:

(1) These signals are not registered or registered once to match the data path pipeline.

(2) These signals are not registered, registered once, or registered twice for latency to match the data path pipeline.

Two-Multipliers Adder Mode

The two-multipliers adder mode uses the adder / subtractor / accumulator block to add or subtract the outputs of the multiplier block, which is useful for applications such as FFT functions and complex FIR filters. A

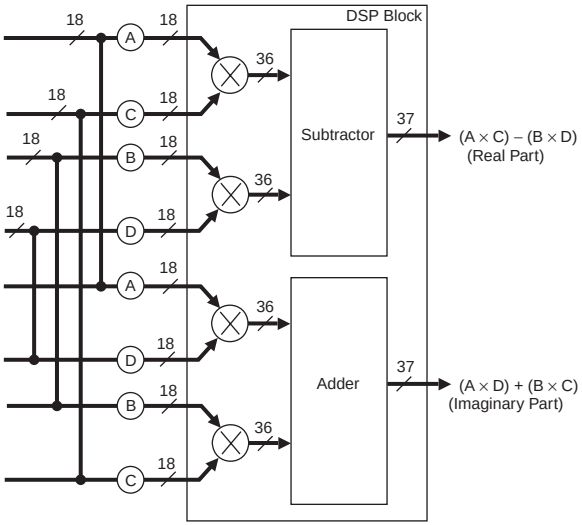
single DSP block can implement two sums or differences from two 18 × 18-bit multipliers each or four sums or differences from two 9 × 9-bit multipliers each.

You can use the two-multipliers adder mode for complex multiplications, which are written as:

$$(a + jb) \times (c + jd) = [(a \times c) - (b \times d)] + j \times [(a \times d) + (b \times c)]$$

The two-multipliers adder mode allows a single DSP block to calculate the real part [(a × c) – (b × d)] using one subtractor and the imaginary part [(a × d) + (b × c)] using one adder, for data widths up to 18 bits. Two complex multiplications are possible for data widths up to 9 bits using four adder / subtractor / accumulator blocks. Figure 2–38 shows an 18-bit two-multipliers adder.

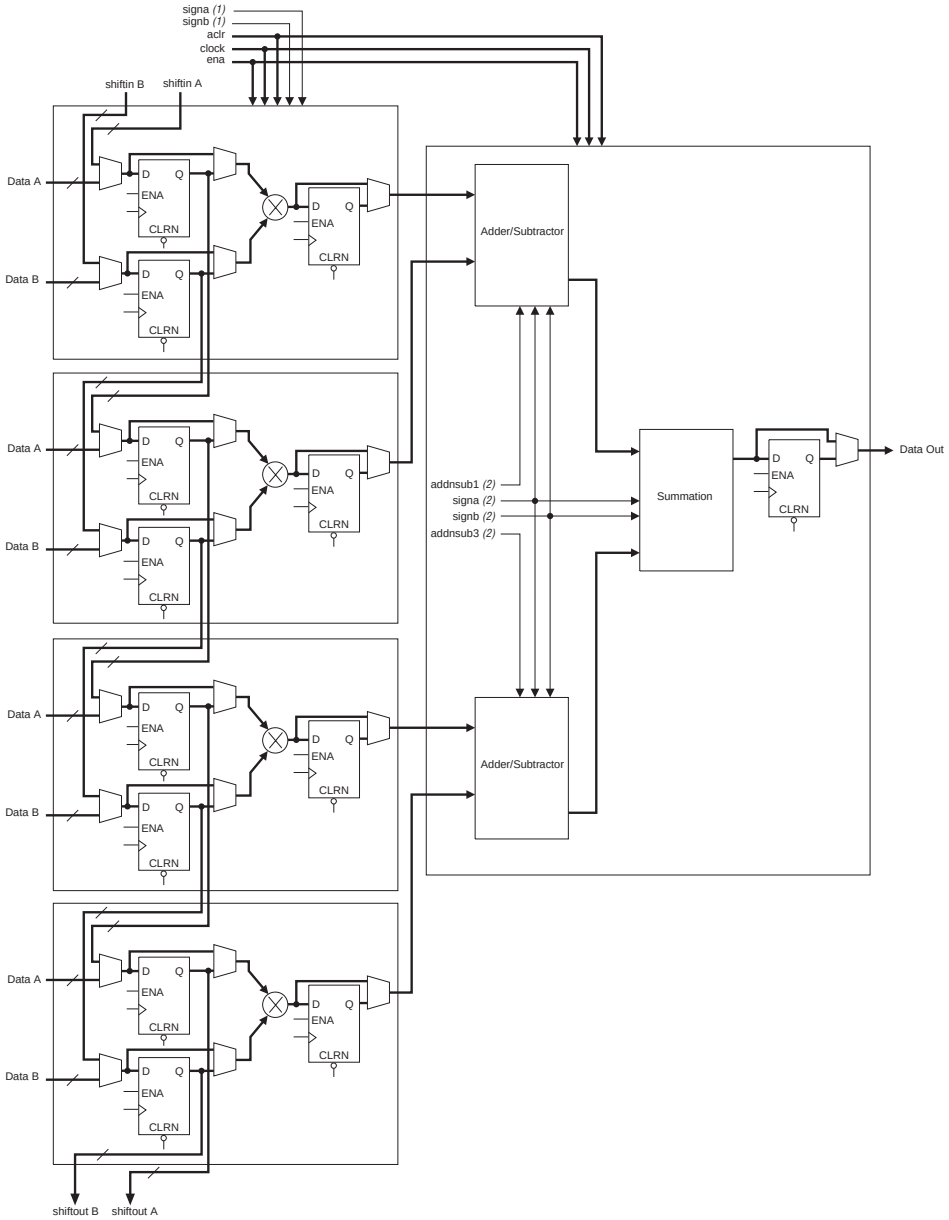
Figure 2–38. Two-Multipliers Adder Mode Implementing Complex Multiply



Four-Multipliers Adder Mode

In the four-multipliers adder mode, the DSP block adds the results of two first -stage adder / subtractor blocks. One sum of four 18 × 18-bit multipliers or two different sums of two sets of four 9 × 9-bit multipliers can be implemented in a single DSP block. The product width for each multiplier must be the same size. The four-multipliers adder mode is useful for FIR filter applications. Figure 2–39 shows the four multipliers adder mode.

Figure 2–39. Four-Multipliers Adder Mode



Notes to Figure 2–39:

(1) These signals are not registered or registered once to match the data path pipeline.

(2) These signals are not registered, registered once, or registered twice for latency to match the data path pipeline.

For FIR filters, the DSP block combines the four-multipliers adder mode with the shift register inputs. One set of shift inputs contains the filter data, while the other holds the coefficients loaded in serial or parallel. The input shift register eliminates the need for shift registers external to the DSP block (i.e., implemented in LEs). This architecture simplifies filter design since the DSP block implements all of the filter circuitry.

One DSP block can implement an entire 18-bit FIR filter with up to four taps. For FIR filters larger than four taps, DSP blocks can be cascaded with additional adder stages implemented in LEs.

Table 2–16 shows the different number of multipliers possible in each DSP block mode according to size. These modes allow the DSP blocks to implement numerous applications for DSP including FFTs, complex FIR, FIR, and 2D FIR filters, equalizers, IIR, correlators, matrix multiplication and many other functions.

Table 2–16. Multiplier Size & Configurations per DSP block

DSP Block Mode	9 × 9	18 × 18	36 × 36 *(1)*
Multiplier	Eight multipliers with eight product outputs	Four multipliers with four product outputs	One multiplier with one product output
Multiply-accumulator	Two multiply and accumulate (52 bits)	Two multiply and accumulate (52 bits)	–
Two-multipliers adder	Four sums of two multiplier products each	Two sums of two multiplier products each	–
Four-multipliers adder	Two sums of four multiplier products each	One sum of four multiplier products each	–

Note to Table 2–16:

(1) The number of supported multiply functions shown is based on signed / signed or unsigned / unsigned implementations.

DSP Block Interface

Stratix device DSP block outputs can cascade down within the same DSP block column. Dedicated connections between DSP blocks provide fast connections between the shift register inputs to cascade the shift register chains. You can cascade DSP blocks for 9 × 9- or 18 × 18-bit FIR filters larger than four taps, with additional adder stages implemented in LEs. If the DSP block is configured as 36 × 36 bits, the adder, subtractor, or accumulator stages are implemented in LEs. Each DSP block can route the shift register chain out of the block to cascade two full columns of DSP blocks.

The DSP block is divided into eight block units that interface with eight LAB rows on the left and right. Each block unit can be considered half of an 18 × 18-bit multiplier sub-block with 18 inputs and 18 outputs. A local interconnect region is associated with each DSP block. Like an LAB, this interconnect region can be fed with 10 direct link interconnects from the LAB to the left or right of the DSP block in the same row. All row and column routing resources can access the DSP block's local interconnect region. The outputs also work similarly to LAB outputs as well. Nine outputs from the DSP block can drive to the left LAB through direct link interconnects and nine can drive to the right LAB though direct link interconnects. All 18 outputs can drive to all types of row and column routing. Outputs can drive right- or left-column routing. Figures 2–40 and 2–41 show the DSP block interfaces to LAB rows.

Figure 2–40. DSP Block Interconnect Interface

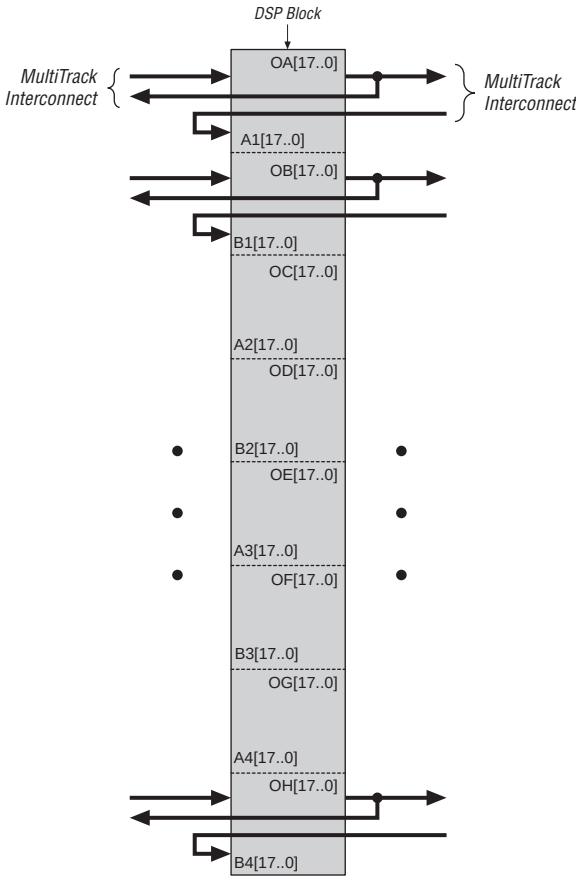
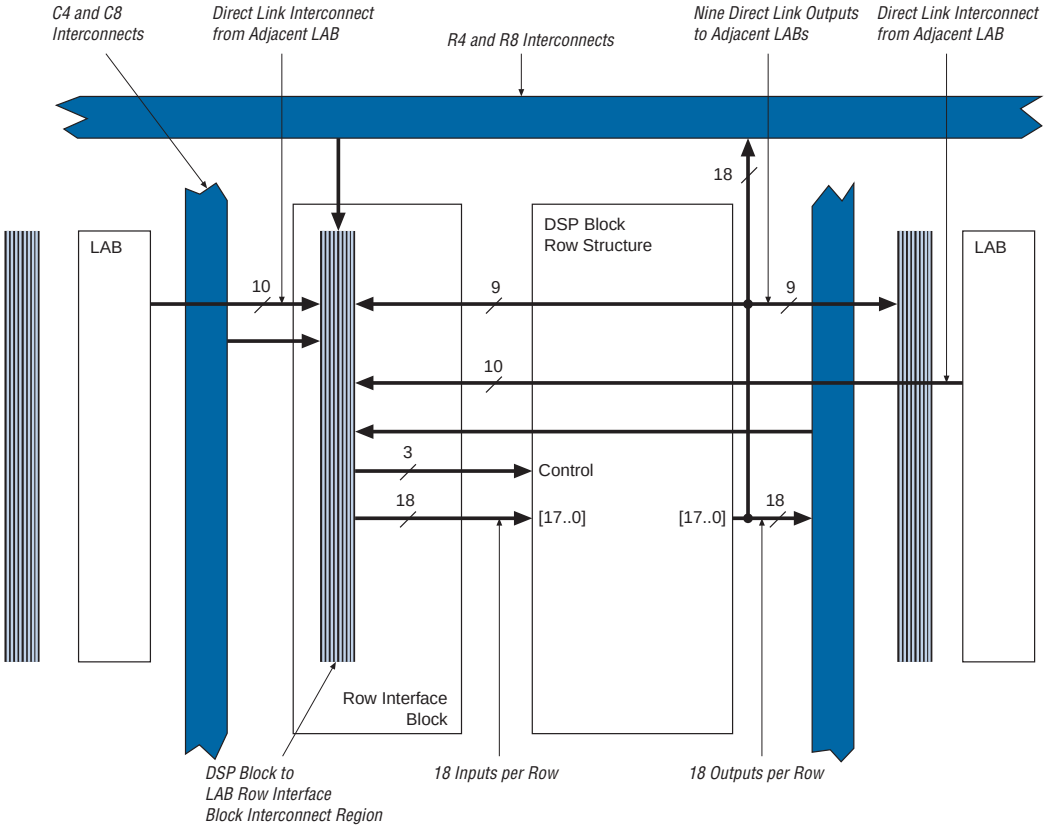


Figure 2–41. DSP Block Interface to Interconnect



A bus of 18 control signals feeds the entire DSP block. These signals include `clock[0..3]` clocks, `aclr[0..3]` asynchronous clears, `ena[1..4]` clock enables, `signa`, `signb` signed / unsigned control signals, `addnsub1` and `addnsub3` addition and subtraction control signals, and `accum_sload[0..1]` accumulator synchronous loads. The

clock signals are routed from LAB row clocks and are generated from specific LAB rows at the DSP block interface. The LAB row source for control signals, data inputs, and outputs is shown in Table 2–17.

Table 2–17. DSP Block Signal Sources & Destinations

LAB Row at Interface	Control Signals Generated	Data Inputs	Data Outputs
1	signa	A1[17..0]	OA[17..0]
2	aclr0 accum_sload0	B1[17..0]	OB[17..0]
3	addnsub1 clock0 ena0	A2[17..0]	OC[17..0]
4	aclr1 clock1 ena1	B2[17..0]	OD[17..0]
5	aclr2 clock2 ena2	A3[17..0]	OE[17..0]
6	sign_b clock3 ena3	B3[17..0]	OF[17..0]
7	clear3 accum_sload1	A4[17..0]	OG[17..0]
8	addnsub3	B4[17..0]	OH[17..0]

PLLs & Clock Networks

Stratix devices provide a hierarchical clock structure and multiple PLLs with advanced features. The large number of clocking resources in combination with the clock synthesis precision provided by enhanced and fast PLLs provides a complete clock management solution.

Global & Hierarchical Clocking

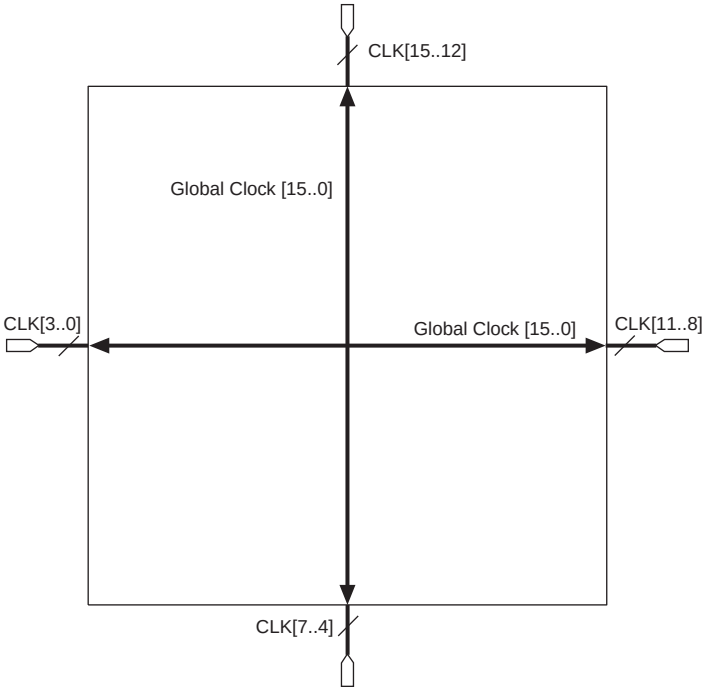
Stratix devices provide 16 dedicated global clock networks, 16 regional clock networks (four per device quadrant), and 8 dedicated fast regional clock networks (for EP1S10, EP1S20, and EP1S25 devices), and 16 dedicated fast regional clock networks (for EP1S30 EP1S40, and EP1S60, and EP1S80 devices). These clocks are organized into a hierarchical clock structure that allows for up to 22 clocks per device region with low skew and delay. This hierarchical clocking scheme provides up to 48 unique clock domains within Stratix devices.

There are 16 dedicated clock pins (`CLK[15..0]`) to drive either the global or regional clock networks. Four clock pins drive each side of the device, as shown in Figure 2–42. Enhanced and fast PLL outputs can also drive the global and regional clock networks.

Global Clock Network

These clocks drive throughout the entire device, feeding all device quadrants. The global clock networks can be used as clock sources for all resources within the device—IOEs, LEs, DSP blocks, and all memory blocks. These resources can also be used for control signals, such as clock enables and synchronous or asynchronous clears fed from the external pin. The global clock networks can also be driven by internal logic for internally generated global clocks and asynchronous clears, clock enables, or other control signals with large fanout. Figure 2–42 shows the 16 dedicated `CLK` pins driving global clock networks.

Figure 2–42. Global Clocking *Note (1)*



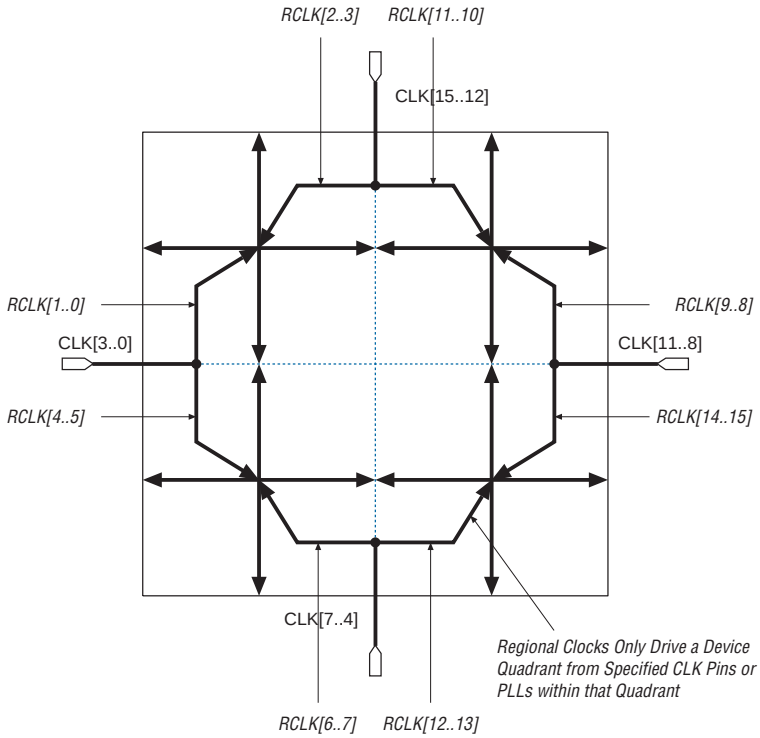
Note to Figure 2–42:

(1) The corner fast PLLs can also be driven through the global or regional clock networks. The global or regional clock input to the fast PLL can be driven by an output from another PLL, a pin-driven global or regional clock, or internally-generated global signals.

Regional Clock Network

There are four regional clock networks within each quadrant of the Stratix device that are driven by the same dedicated `CLK[15..0]` input pins or from PLL outputs. From a top view of the silicon, `RCLK[0..3]` are in the top left quadrant, `RCLK[8..11]` are in the top-right quadrant, `RCLK[4..7]` are in the bottom-left quadrant, and `RCLK[12..15]` are in the bottom-right quadrant. The regional clock networks only pertain to the quadrant they drive into. The regional clock networks provide the lowest clock delay and skew for logic contained within a single quadrant. `RCLK` cannot be driven by internal logic. The `CLK` clock pins symmetrically drive the `RCLK` networks within a particular quadrant, as shown in Figure 2–43. See Figures 2–50 and 2–51 for `RCLK` connections from PLLs and `CLK` pins.

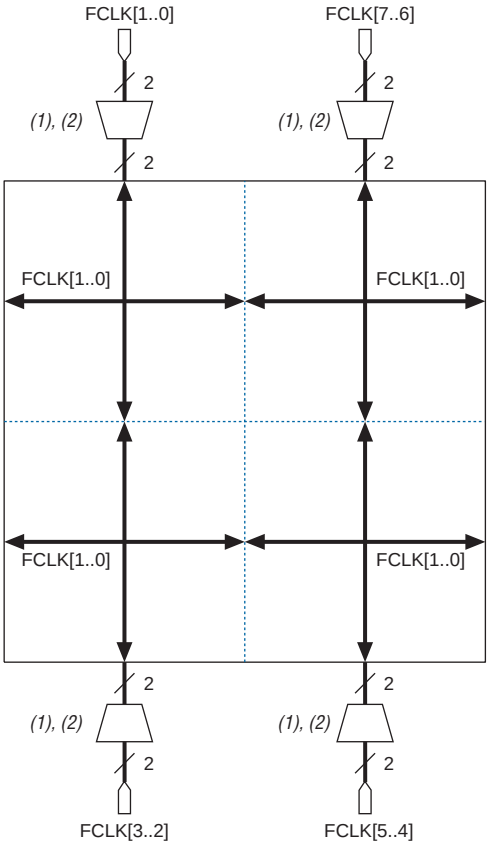
Figure 2–43. Regional Clocks



Fast Regional Clock Network

In EP1S25, EP1S20, and EP1S10 devices, there are two fast regional clock networks, `FCLK[1..0]`, within each quadrant, fed by input pins that can connect to fast regional clock networks (see Figure 2–44). In EP1S30 and larger devices, there are two fast regional clock networks within each half-quadrant (see Figure 2–45). Dual-purpose `FCLK` pins drive the fast clock networks. All devices have eight `FCLK` pins to drive fast regional clock networks. Any I/O pin can drive a clock or control signal onto any fast regional clock network with the addition of a delay. This signal is driven via the I/O interconnect. The fast regional clock networks can also be driven from internal logic elements.

Figure 2–44. EP1S25, EP1S20 & EP1S10 Device Fast Clock Pin Connections to Fast Regional Clocks

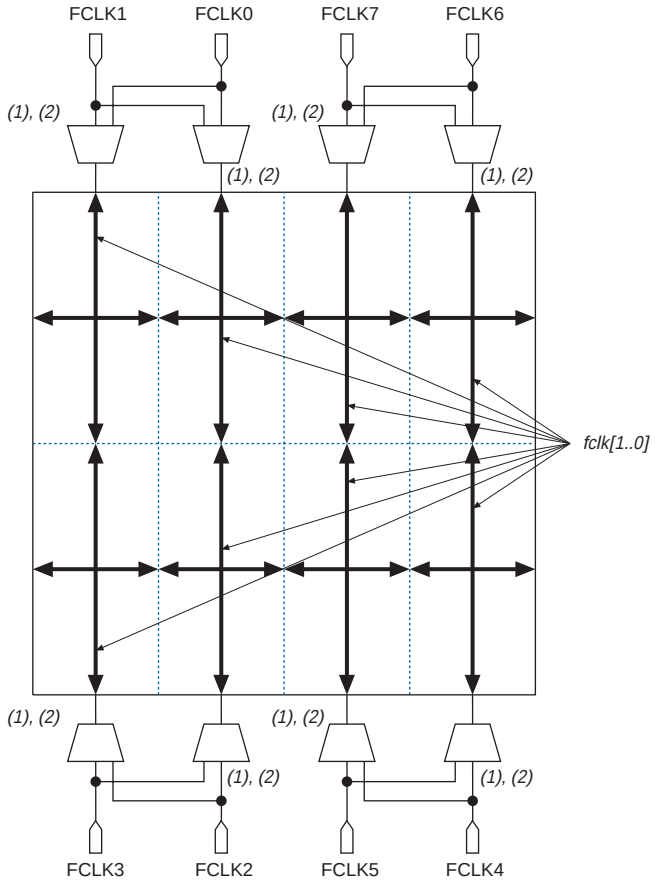


Notes to Figure 2–44:

(1) This is a set of two multiplexers.

(2) In addition to the FCLK pin inputs, there is also an input from the I/O interconnect.

Figure 2–45. EP1S30 Device Fast Regional Clock Pin Connections to Fast Regional Clocks



Notes to Figure 2–45:

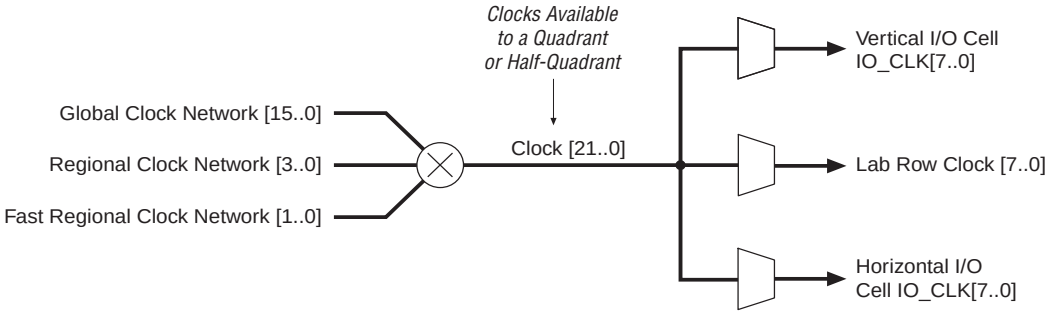
(1) This is a set of two multiplexers.

(2) In addition to the FCLK pin inputs, there is also an input from the I/O interconnect.

Combined Resources

Within each region, there are 22 distinct dedicated clocking resources consisting of 16 global clock lines, four regional clock lines, and two fast regional clock lines. Multiplexers are used with these clocks to form eight bit busses to drive LAB row clocks, column IOE clocks, or row IOE clocks. Another multiplexer is used at the LAB level to select two of the eight row clocks to feed the LE registers within the LAB. See Figure 2–46.

Figure 2–46. Regional Clock Bus



IOE clocks have horizontal and vertical block regions that are clocked by eight I/O clock signals chosen from the 22 quadrant or half-quadrant clock resources. Figures 2–47 and 2–48 show the quadrant and half-quadrant relationship to the I/O clock regions, respectively. The vertical regions (column pins) have less clock delay than the horizontal regions (row pins).

Figure 2–47. EP1S10, EP1S20 & EP1S25 Device I/O Clock Groups

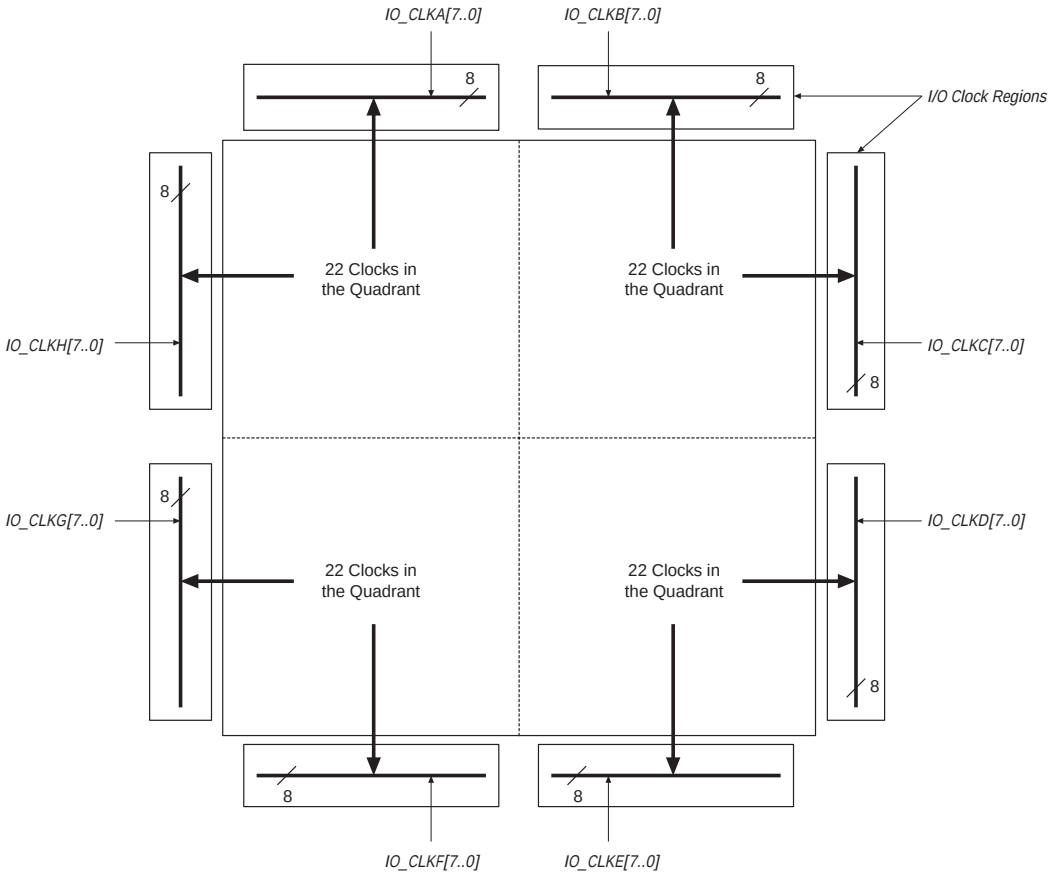
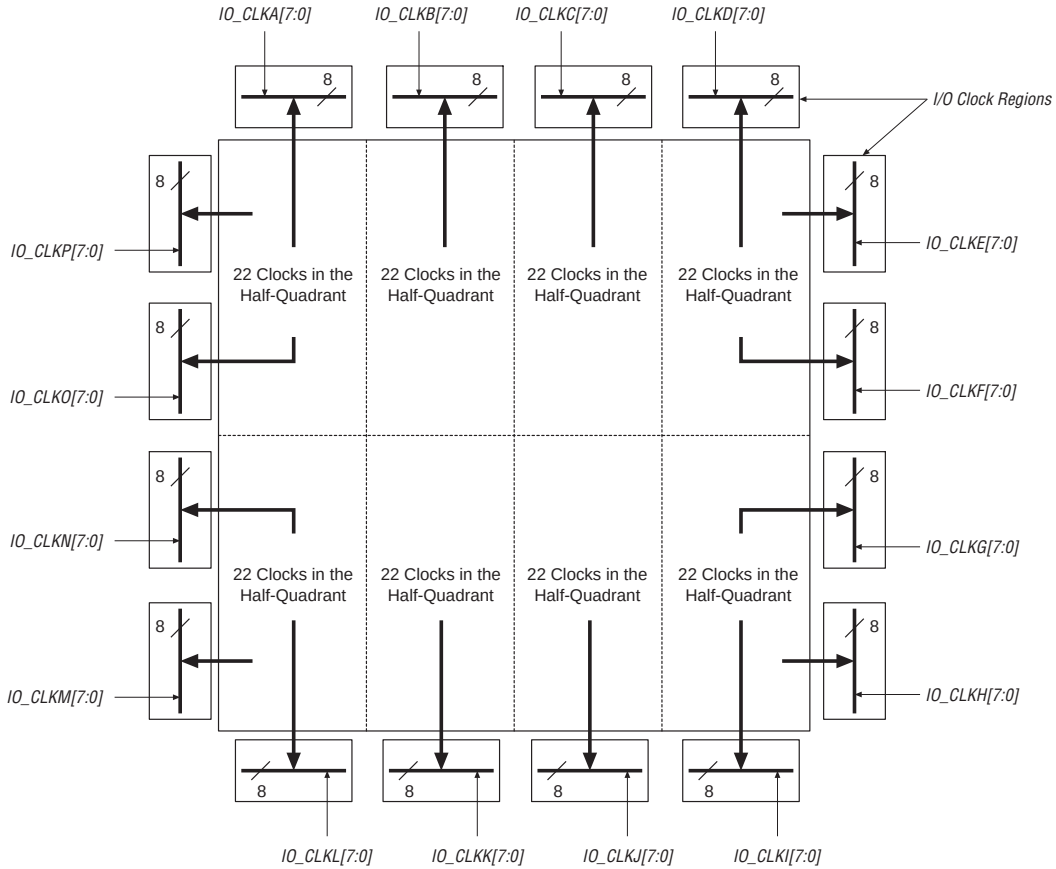


Figure 2–48. EP1S30, EP1S40, EP1S60, EP1S80 Device I/O Clock Groups



You can use the Quartus II software to control whether a clock input pin is either global, regional, or fast regional. The Quartus II software automatically selects the clocking resources if not specified.

Enhanced & Fast PLLs

Stratix devices provide robust clock management and synthesis using up to four enhanced PLLs and eight fast PLLs. These PLLs increase performance and provide advanced clock interfacing and clock-frequency synthesis. With features such as clock switchover, spread spectrum clocking, programmable bandwidth, phase and delay control, and PLL reconfiguration, the Stratix device's enhanced PLLs provide you with complete control of your clocks and system timing. The fast PLLs

provide general purpose clocking with multiplication and phase shifting as well as high-speed outputs for high-speed differential I/O support. Enhanced and fast PLLs work together with the Stratix high-speed I/O and advanced clock architecture to provide significant improvements in system performance and bandwidth.

The Quartus II software enables the PLLs and their features without requiring any external devices. Table 2–18 shows the PLLs available for each Stratix device.

Table 2–18. Stratix Device PLL Availability

Device	Fast PLLs								Enhanced PLLs			
	1	2	3	4	7	8	9	10	5 *(1)*	6 *(1)*	11 *(2)*	12 *(2)*
EP1S10	v	v	v	v					v	v		
EP1S20	v	v	v	v					v	v		
EP1S25	v	v	v	v					v	v		
EP1S30	v	v	v	v	v *(3)*	v *(3)*	v *(3)*	v *(3)*	v	v		
EP1S40	v	v	v	v	v *(3)*	v *(3)*	v *(3)*	v *(3)*	v	v	v *(3)*	v *(3)*
EP1S60	v	v	v	v	v	v	v	v	v	v	v	v
EP1S80	v	v	v	v	v	v	v	v	v	v	v	v

Notes to Table 2–18:

(1) PLLs 5 and 6 each have eight single-ended outputs or four differential outputs.

(2) PLLs 11 and 12 each have one single-ended output.

(3) EP1S30 and EP1S40 devices do not support these PLLs in the 780-pin FineLine BGA® package.

Table 2–19 shows the enhanced PLL and fast PLL features in Stratix devices.

Table 2–19. Stratix PLL Features

Feature	Enhanced PLL	Fast PLL
Clock multiplication and division	*m*/(*n* × post-scale counter) *(1)*	*m*/(post-scale counter) *(2)*
Phase shift	Down to 156.25-ps increments *(3)*, *(4)*	Down to 125-ps increments *(3)*, *(4)*
Delay shift	250-ps increments for ±3 ns	
Clock switchover	v	
PLL reconfiguration	v	
Programmable bandwidth	v	
Spread spectrum clocking	v	
Programmable duty cycle	v	v
Number of internal clock outputs	6	3 *(5)*
Number of external clock outputs	Four differential/eight singled-ended or one single-ended *(6)*	*(7)*
Number of feedback clock inputs	2 *(8)*	

Notes to Table 2–19:

(1) For enhanced PLLs, *m*, *n*, range from 1 to 512 and post-scale counters *g*, *l*, *e* range from 1 to 1024 with 50% duty cycle. With a non-50% duty cycle the post-scale counters *g*, *l*, *e* range from 1 to 512.

(2) For fast PLLs, *m* and post-scale counters range from 1 to 32.

(3) The smallest phase shift is determined by the voltage controlled oscillator (VCO) period divided by 8.

(4) For degree increments, Stratix devices can shift all output frequencies in increments of at least 45°. Smaller degree increments are possible depending on the frequency and divide parameters.

(5) PLLs 7, 8, 9, and 10 have two output ports per PLL. PLLs 1, 2, 3, and 4 have three output ports per PLL.

(6) Every Stratix device has two enhanced PLLs (PLLs 5 and 6) with either eight single-ended outputs or four differential outputs each. Two additional enhanced PLLs (PLLs 11 and 12) in EP1S80, EP1S60, and EP1S40 devices each have one single-ended output. Devices in the 780 pin FineLine BGA packages do not support PLLs 11 and 12.

(7) Fast PLLs can drive to any I/O pin as an external clock. For high-speed differential I/O pins, the device uses a data channel to generate `txclkout`.

(8) Every Stratix device has two enhanced PLLs with one single-ended or differential external feedback input per PLL.

Figure 2–49 shows a top-level diagram of the Stratix device and PLL floorplan.

Figure 2–49. PLL Locations

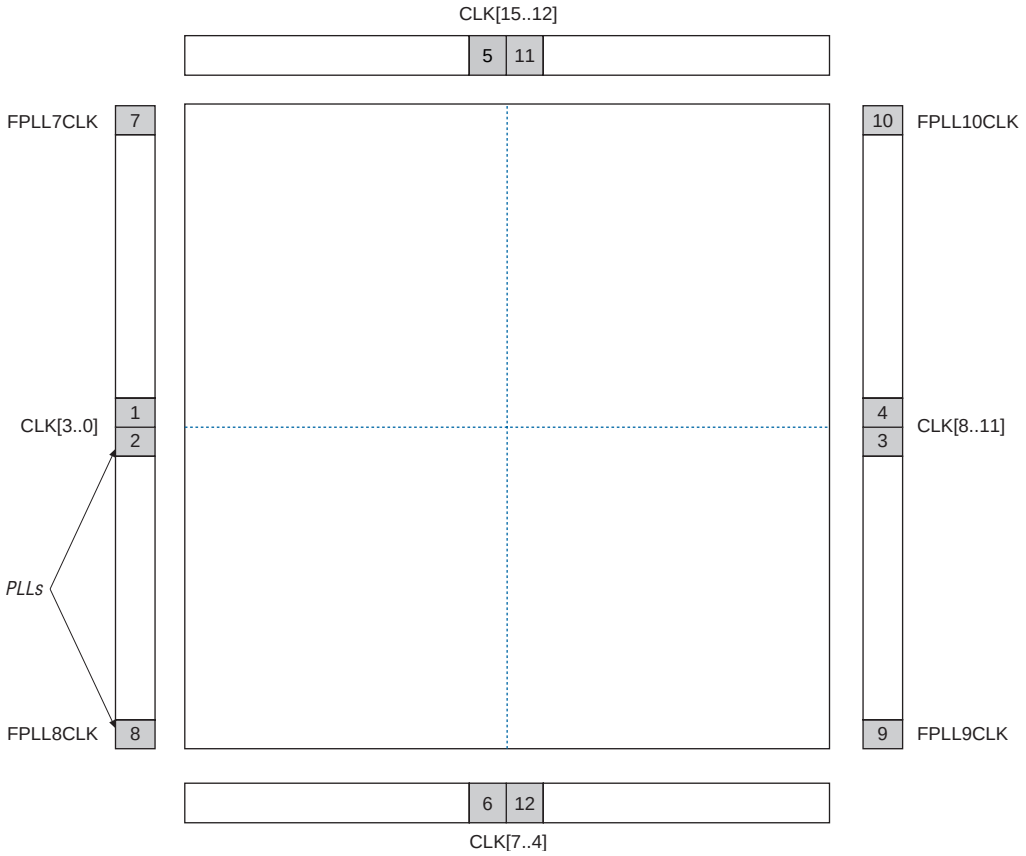
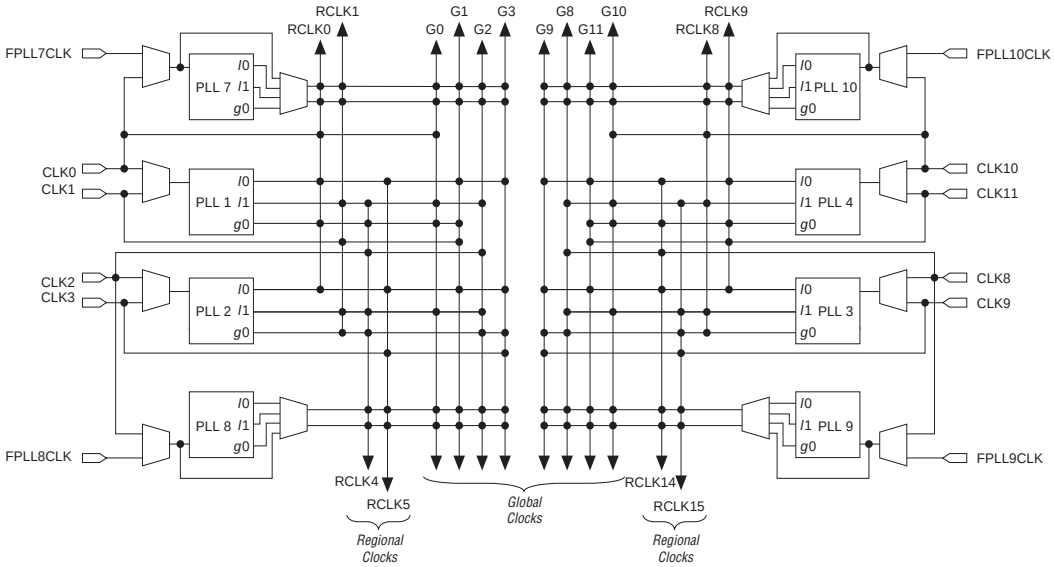


Figure 2–50 shows the global and regional clocking from the PLL outputs and the CLK pins.

Figure 2–50. Global & Regional Clock Connections from Side Pins & Fast PLL Outputs *Note (1), (2)*



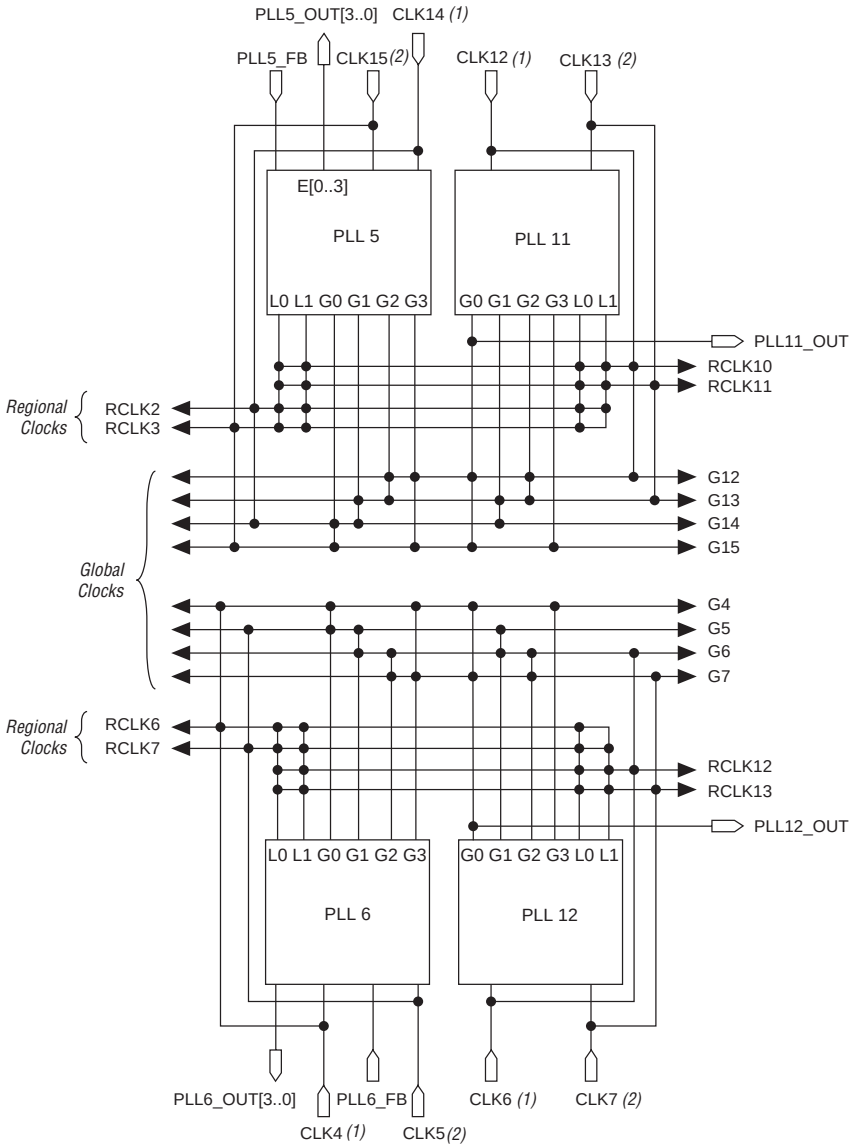
Notes to Figure 2–50:

(1) PLLs 1 to 4 and 7 to 10 are fast PLLs. PLLs 5, 6, 11, and 12 are enhanced PLLs.

(2) The global or regional clocks in a fast PLL's quadrant can drive the fast PLL input. A pin or other PLL must drive the global or regional source. The source cannot be driven by internally generated logic before driving the fast PLL.

Figure 2–51 shows the global and regional clocking from enhanced PLL outputs and top CLK pins.

Figure 2–51. Global & Regional Clock Connections from Top Clock Pins & Enhanced PLL Outputs Note (1)



Notes to Figure 2–51:

(1) PLLs 1 to 4 and 7 to 10 are fast PLLs. PLLs 5, 6, 11, and 12 are enhanced PLLs.

(2) `CLK4`, `CLK6`, `CLK12`, and `CLK14` feed the corresponding PLL's `inclk0` port.

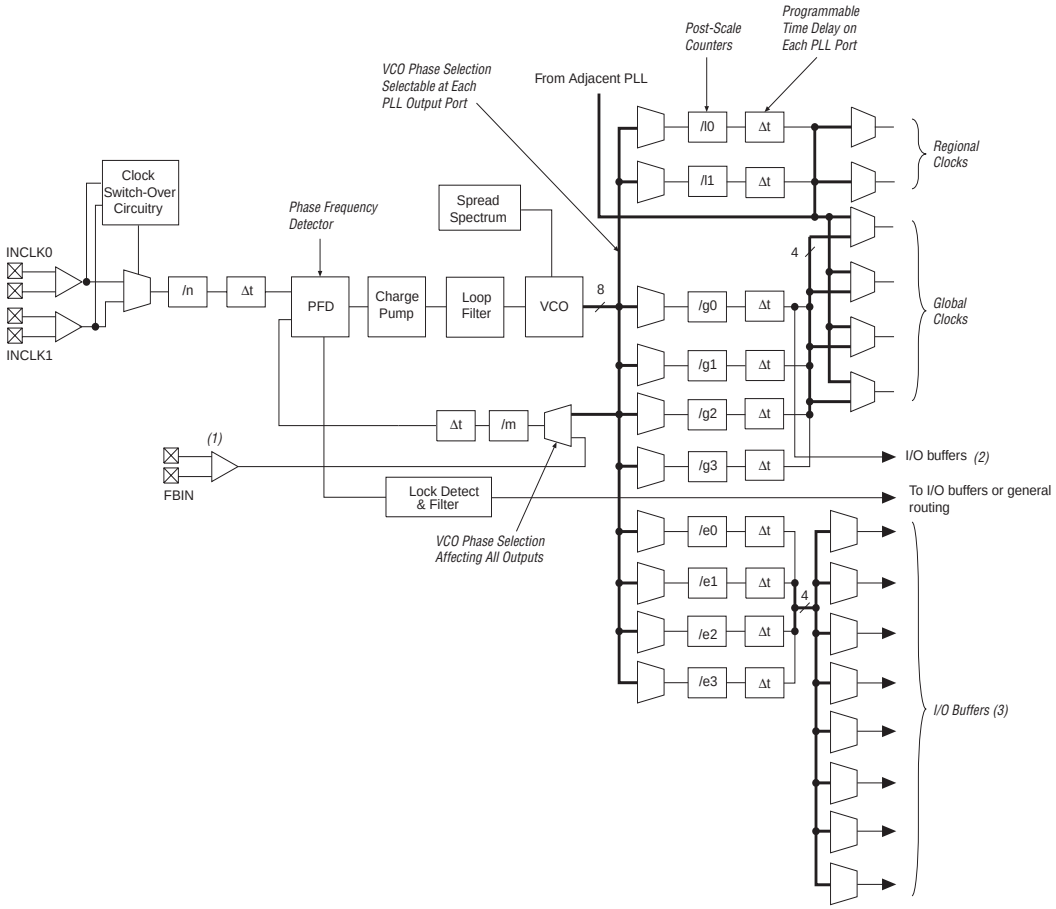
(3) `CLK5`, `CLK7`, `CLK13`, and `CLK15` feed the corresponding PLL's `inclk1` port.

(4) The EP1S40 device in the 780-pin FineLine BGA package does not support PLLs 11 and 12.

Enhanced PLLs

Stratix devices contain up to four enhanced PLLs with advanced clock management features. Figure 2–52 shows a diagram of the enhanced PLL.

Figure 2–52. Stratix Enhanced PLL



Notes to Figure 2–52:

(1) External feedback is available in PLLs 5 and 6.

(2) This single-ended external output is available from the *g*0 counter for PLLs 11 and 12.

(3) These four counters and external outputs are available in PLLs 5 and 6.

(4) This connection is only available on EP1S40 and larger Stratix devices. For example, PLLs 5 and 11 are adjacent and PLLs 6 and 12 are adjacent. The EP1S40 device in the 780-pin FineLine BGA package does not support PLLs 11 and 12.

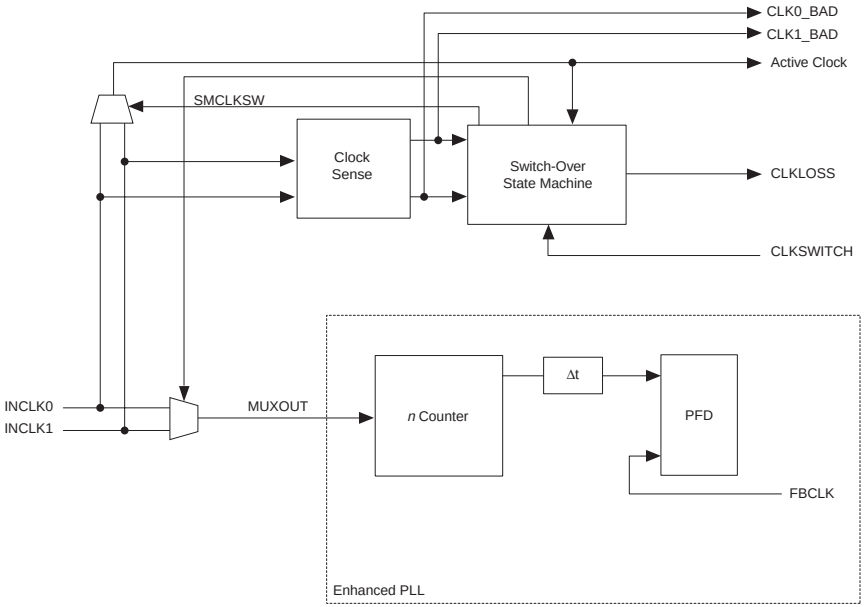
Clock Multiplication & Division

Each Stratix device enhanced PLL provides clock synthesis for PLL output ports using $m / (n \times$ post-scale counter) scaling factors. The input clock is divided by a pre-scale divider, *n*, and is then multiplied by the *m* feedback factor. The control loop drives the VCO to match $f_{IN} \times (m / n)$. Each output port has a unique post-scale counter that divides down the high-frequency VCO. For multiple PLL outputs with different frequencies, the VCO is set to the least common multiple of the output frequencies that meets its frequency specifications. Then, the post-scale dividers scale down the output frequency for each output port. For example, if output frequencies required from one PLL are 33 and 66 MHz, set the VCO to 330 MHz (the least common multiple in the VCO's range). There is one pre-scale counter, *n*, and one multiply counter, *m*, per PLL, with a range of 1 to 512 on each. There are two post-scale counters (*l*) for regional clock output ports, four counters (*g*) for global clock output ports, and up to four counters (*e*) for external clock outputs, all ranging from 1 to 1024 with a 50% duty cycle setting. The post-scale counters range from 1 to 512 with any non-50% duty cycle setting. The Quartus II software automatically chooses the appropriate scaling factors according to the input frequency, multiplication, and division values entered.

Clock Switchover

To effectively develop high-reliability network systems, clocking schemes must support multiple clocks to provide redundancy. For this reason, Stratix device enhanced PLLs support a flexible clock switchover capability. Figure 2–53 shows a block diagram of the switchover circuit.The switchover circuit is configurable, so you can define how to implement it. Clock-sense circuitry automatically switches from the primary to secondary clock for PLL reference when the primary clock signal is not present.

Figure 2–53. Clock Switchover Circuitry



There are two possible ways to use the clock switchover feature.

- Use automatic switchover circuitry for switching between inputs of the same frequency. For example, in applications that require a redundant clock with the same frequency as the primary clock, the switchover state machine generates a signal that controls the multiplexer select input on the bottom of Figure 2–53. In this case, the secondary clock becomes the reference clock for the PLL.
- Use the `clkswitch` input for user- or system-controlled switch conditions. This is possible for same-frequency switchover or to switch between inputs of different frequencies. For example, if `inclk0` is 66 MHz and `inclk1` is 100 MHz, you must control the switchover because the automatic clock-sense circuitry cannot monitor primary and secondary clock frequencies with a frequency difference of more than ±20%. This feature is useful when clock sources can originate from multiple cards on the backplane, requiring a system-controlled switchover between frequencies of operation. You can use `clkswitch` together with the lock signal to trigger the switch from a clock that is running but becomes unstable and cannot be locked onto.

During switchover, the PLL VCO continues to run and will either slow down or speed up, generating frequency drift on the PLL outputs. The clock switchover transitions without any glitches. After the switch, there is a finite resynchronization period to lock onto new clock as the VCO ramps up. The exact amount of time it takes for the PLL to relock relates to the PLL configuration and may be adjusted by using the programmable bandwidth feature of the PLL. The specification for the maximum time to relock is 100 µs.

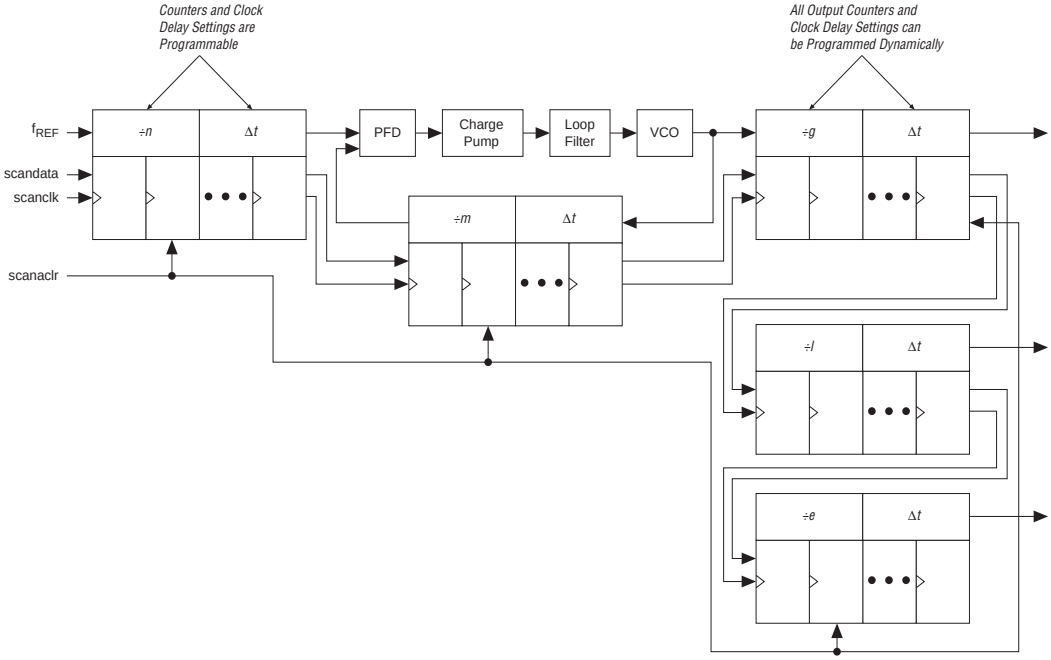
f For more information on clock switchover, see *AN 313, Implementing Clock Switchover in Stratix & Stratix GX Devices*.

PLL Reconfiguration

The PLL reconfiguration feature enables system logic to change Stratix device enhanced PLL counters and delay elements without reloading a Programmer Object File (**.pof**). This provides considerable flexibility for frequency synthesis, allowing real-time PLL frequency and output clock delay variation. You can sweep the PLL output frequencies and clock delay in prototype environments. The PLL reconfiguration feature can also dynamically or intelligently control system clock speeds or t_{CO} delays in end systems.

Clock delay elements at each PLL output port implement variable delay. Figure 2–54 shows a diagram of the overall dynamic PLL control feature for the counters and the clock delay elements. The configuration time is less than 20 µs for the enhanced PLL using a input shift clock rate of 22 MHz. The charge pump, loop filter components, and phase shifting using VCO phase taps cannot be dynamically adjusted.

Figure 2–54. Dynamically Programmable Counters & Delays in Stratix Device Enhanced PLLs



PLL reconfiguration data is shifted into serial registers from the logic array or external devices. The PLL input shift data uses a reference input shift clock. Once the last bit of the serial chain is clocked in, the register chain is synchronously loaded into the PLL configuration bits. The shift circuitry also provides an asynchronous clear for the serial registers.

f For more information on PLL reconfiguration, see *AN 282: Implementing PLL Reconfiguration in Stratix & Stratix GX Devices.*

Programmable Bandwidth

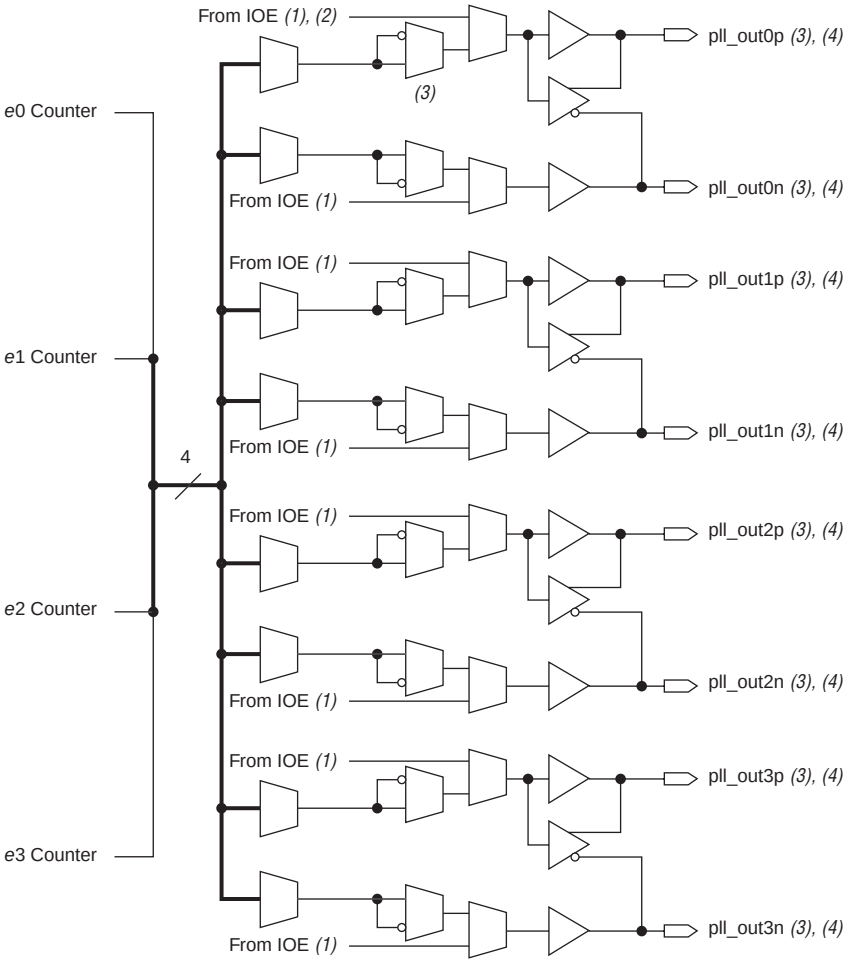
You have advanced control of the PLL bandwidth using the programmable control of the PLL loop characteristics, including loop filter and charge pump. The PLL's bandwidth is a measure of its ability to track the input clock and jitter. A high-bandwidth PLL can quickly lock onto a reference clock and react to any changes in the clock. It also will allow a wide band of input jitter spectrum to pass to the output. A low-bandwidth PLL will take longer to lock, but it will attenuate all high-frequency jitter components. The Quartus II software can adjust PLL characteristics to achieve the desired bandwidth. The programmable

bandwidth is tuned by varying the charge pump current, loop filter resistor value, high frequency capacitor value, and *m* counter value. You can manually adjust these values if desired. Bandwidth is programmable from 200 kHz to 1.5 MHz.

External Clock Outputs

Enhanced PLLs 5 and 6 each support up to eight single-ended clock outputs (or four differential pairs). Differential SSTL and HSTL outputs are implemented using 2 single-ended output buffers which are programmed to have opposite polarity. In Quartus II software, simply assign the appropriate differential I/O standard and the software will implement the inversion. See Figure 2–55.

Figure 2–55. External Clock Outputs for PLLs 5 & 6



Notes to Figure 2–55:

(1) The design can use each external clock output pin as a general-purpose output pin from the logic array. These pins are multiplexed with IOE outputs.

(2) Two single-ended outputs are possible per output counter—either two outputs of the same frequency and phase or one shifted 180°.

(3) EP1S10, EP1S20, and EP1S25 devices in 672-pin BGA and 484- and 672-pin FineLine BGA packages only have two pairs of external clocks (i.e., `pll_out0p`, `pll_out0n`, `pll_out1p`, and `pll_out1n`).

(4) Differential SSTL and HSTL outputs are implemented using two single-ended output buffers, which are programmed to have opposite polarity.

Any of the four external output counters can drive the single-ended or differential clock outputs for PLLs 5 and 6. This means one counter or frequency can drive all output pins available from PLL 5 or PLL 6. Each pair of output pins (four pins total) has dedicated VCC and GND pins to reduce the output clock's overall jitter by providing improved isolation from switching I/O pins.

For PLLs 5 and 6, each pin of a single-ended output pair can either be in phase or 180° out of phase. The clock output pin pairs support the same I/O standards as standard output pins (in the top and bottom banks) as well as LVDS, LVPECL, 3.3-V PCML, HyperTransport technology, differential HSTL, and differential SSTL. Table 2–20 shows which I/O standards the enhanced PLL clock pins support. When in single-ended or differential mode, the two outputs operate off the same power supply. Both outputs use the same standards in single-ended mode to maintain performance. You can also use the external clock output pins as user output pins if external enhanced PLL clocking is not needed.

Table 2–20. I/O Standards Supported for Enhanced PLL Pins (Part 1 of 2)

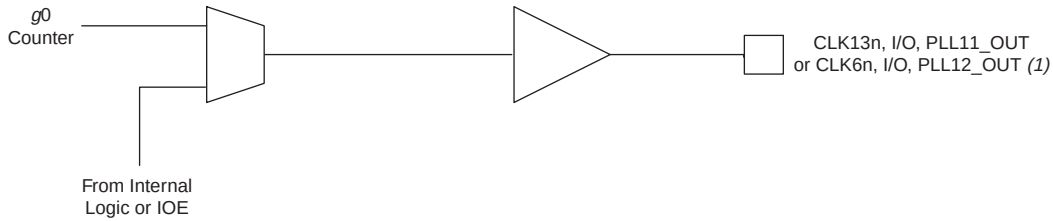
I/O Standard	Input			Output
	INCLK	FBIN	PLLENABLE	EXTCLK
LVTTL	v	v	v	v
LVCMOS	v	v	v	v
2.5 V	v	v		v
1.8 V	v	v		v
1.5 V	v	v		v
3.3-V PCI	v	v		v
3.3-V PCI-X 1.0	v	v		v
LVPECL	v	v		v
3.3-V PCML	v	v		v
LVDS	v	v		v
HyperTransport technology	v	v		v
Differential HSTL	v			v
Differential SSTL				v
3.3-V GTL	v	v		v
3.3-V GTL+	v	v		v
1.5-V HSTL Class I	v	v		v

Table 2–20. I/O Standards Supported for Enhanced PLL Pins (Part 2 of 2)

I/O Standard	Input			Output
	INCLK	FBIN	PLLENABLE	EXTCLK
1.5-V HSTL Class II	v	v		v
1.8-V HSTL Class I	v	v		v
1.8-V HSTL Class II	v	v		v
SSTL-18 Class I	v	v		v
SSTL-18 Class II	v	v		v
SSTL-2 Class I	v	v		v
SSTL-2 Class II	v	v		v
SSTL-3 Class I	v	v		v
SSTL-3 Class II	v	v		v
AGP (1× and 2×)	v	v		v
CTT	v	v		v

Enhanced PLLs 11 and 12 support one single-ended output each (see Figure 2–56). These outputs do not have their own VCC and GND signals. Therefore, to minimize jitter, do not place switching I/O pins next to this output pin.

Figure 2–56. External Clock Outputs for Enhanced PLLs 11 & 12



Note to Figure 2–56:

(1) For PLL 11, this pin is CLK13n; for PLL 12 this pin is CLK7n.

Stratix devices can drive any enhanced PLL driven through the global clock or regional clock network to any general I/O pin as an external output clock. The jitter on the output clock is not guaranteed for these cases.

Clock Feedback

The following four feedback modes in Stratix device enhanced PLLs allow multiplication and/or phase and delay shifting:

- Zero delay buffer: The external clock output pin is phase-aligned with the clock input pin for zero delay. Altera recommends using the same I/O standard on the input clock and the output clocks for optimum performance.

- External feedback: The external feedback input pin, `FBIN`, is phase-aligned with the clock input, `CLK`, pin. Aligning these clocks allows you to remove clock delay and skew between devices. This mode is only possible for PLLs 5 and 6. PLLs 5 and 6 each support feedback for one of the dedicated external outputs, either one single-ended or one differential pair. In this mode, one *e* counter feeds back to the PLL `FBIN` input, becoming part of the feedback loop. Altera recommends using the same I/O standard on the input clock, the `FBIN` pin, and the output clocks for optimum performance.

- Normal mode: If an internal clock is used in this mode, it is phase-aligned to the input clock pin. The external clock output pin will have a phase delay relative to the clock input pin if connected in this mode. You define which internal clock output from the PLL should be phase-aligned to the internal clock pin.

- No compensation: In this mode, the PLL will not compensate for any clock networks or external clock outputs.

Phase & Delay Shifting

Stratix device enhanced PLLs provide advanced programmable phase and clock delay shifting. These parameters are set in the Quartus II software.

Phase Delay

The Quartus II software automatically sets the phase taps and counter settings according to the phase shift entry. You enter a desired phase shift and the Quartus II software automatically sets the closest setting achievable. This type of phase shift is not reconfigurable during system operation. For phase shifting, enter a phase shift (in degrees or time units) for each PLL clock output port or for all outputs together in one shift. You can select phase-shifting values in time units with a resolution of 156.25 to 416.66 ps. This resolution is a function of frequency input and the multiplication and division factors (that is, it is a function of the VCO period), with the finest step being equal to an eighth (×0.125) of the VCO period. Each clock output counter can choose a different phase of the

VCO period from up to eight taps for individual fine step selection. Also, each clock output counter can use a unique initial count setting to achieve individual coarse shift selection in steps of one VCO period. The combination of coarse and fine shifts allows phase shifting for the entire input clock period.

The equation to determine the precision of the phase shifting in degrees is: 45° ÷ post-scale counter value. Therefore, the maximum step size is 45°, and smaller steps are possible depending on the multiplication and division ratio necessary on the output counter port.

This type of phase shift provides the highest precision since it is the least sensitive to process, supply, and temperature variation.

Clock Delay

In addition to the phase shift feature, the ability to fine tune the Δt clock delay provides advanced time delay shift control on each of the four PLL outputs. There are time delays for each post-scale counter (*e*, *g*, or *l*) from the PLL, the *n* counter, and *m* counter. Each of these can shift in 250-ps increments for a range of 3.0 ns. The *m* delay shifts all outputs earlier in time, while *n* delay shifts all outputs later in time. Individual delays on post-scale counters (*e*, *g*, and *l*) provide positive delay for each output. Table 2–21 shows the combined delay for each output for normal or zero delay buffer mode where Δt_e, Δt_g, or Δt_l is unique for each PLL output.

The t_{OUTPUT} for a single output can range from –3 ns to +6 ns. The total delay shift difference between any two PLL outputs, however, must be less than ±3 ns. For example, shifts on two outputs of –1 and +2 ns is allowed, but not –1 and +2.5 ns because these shifts would result in a difference of 3.5 ns. If the design uses external feedback, the Δt_e delay will remove delay from outputs, represented by a negative sign (see Table 2–21). This effect occurs because the Δt_e delay is then part of the feedback loop.

Table 2–21. Output Clock Delay for Enhanced PLLs

Normal or Zero Delay Buffer Mode	External Feedback Mode
$\Delta t_{eOUTPUT} = \Delta t_n - \Delta t_m + \Delta t_e$ $\Delta t_{gOUTPUT} = \Delta t_n - \Delta t_m + \Delta t_g$ $\Delta t_{lOUTPUT} = \Delta t_n - \Delta t_m + \Delta t_l$	$\Delta t_{eOUTPUT} = \Delta t_n - \Delta t_m - \Delta t_e$ *(1)* $\Delta t_{gOUTPUT} = \Delta t_n - \Delta t_m + \Delta t_g$ $\Delta t_{lOUTPUT} = \Delta t_n - \Delta t_m + \Delta t_l$

Note to Table 2–21:

(1) Δt_e removes delay from outputs in external feedback mode.

The variation due to process, voltage, and temperature is about ±15% on the delay settings. PLL reconfiguration can control the clock delay shift elements, but not the VCO phase shift multiplexers, during system operation.

Spread-Spectrum Clocking

Stratix device enhanced PLLs use spread-spectrum technology to reduce electromagnetic interference generation from a system by distributing the energy over a broader frequency range. The enhanced PLL typically provides 0.5% down spread modulation using a triangular profile. The modulation frequency is programmable. Enabling spread-spectrum for a PLL affects all of its outputs.

Lock Detect

The lock output indicates that there is a stable clock output signal in phase with the reference clock. Without any additional circuitry, the lock signal may toggle as the PLL begins tracking the reference clock. You may need to gate the lock signal for use as a system control. The lock signal from the locked port can drive the logic array or an output pin.

Whenever the PLL loses lock (for example, `inclk` jitter, clock switchover, PLL reconfiguration, power supply noise, and so on), the PLL must be reset with the `areset` signal to guarantee correct phase relationship between the PLL output clocks. If the phase relationship between the input clock versus output clock, and between different output clocks from the PLL is not important in the design, then the PLL need not be reset.

f See the *Stratix FPGA Errata Sheet* for more information on implementing the gated lock signal in a design.

Programmable Duty Cycle

The programmable duty cycle allows enhanced PLLs to generate clock outputs with a variable duty cycle. This feature is supported on each enhanced PLL post-scale counter (*g*0..*g*3, *l*0..*l*3, *e*0..*e*3). The duty cycle setting is achieved by a low and high time count setting for the post-scale dividers. The Quartus II software uses the frequency input and the required multiply or divide rate to determine the duty cycle choices.

Advanced Clear & Enable Control

There are several control signals for clearing and enabling PLLs and their outputs. You can use these signals to control PLL resynchronization and gate PLL output clocks for low-power applications.

The `pllenable` pin is a dedicated pin that enables / disables PLLs. When the `pllenable` pin is low, the clock output ports are driven by `GND` and all the PLLs go out of lock. When the `pllenable` pin goes high again, the PLLs relock and resynchronize to the input clocks. You can choose which PLLs are controlled by the `pllenable` signal by connecting the `pllenable` input port of the `altpll` megafunction to the common `pllenable` input pin.

The `areset` signals are reset / resynchronization inputs for each PLL. The `areset` signal should be asserted every time the PLL loses lock to guarantee correct phase relationship between the PLL output clocks. Users should include the `areset` signal in designs if any of the following conditions are true:

- PLL Reconfiguration or Clock switchover enables in the design.
- Phase relationships between output clocks need to be maintained after a loss of lock condition

The device input pins or logic elements (LEs) can drive these input signals. When driven high, the PLL counters will reset, clearing the PLL output and placing the PLL out of lock. The VCO will set back to its nominal setting (~700 MHz). When driven low again, the PLL will resynchronize to its input as it relocks. If the target VCO frequency is below this nominal frequency, then the output frequency will start at a higher value than desired as the PLL locks. If the system cannot tolerate this, the `clkena` signal can disable the output clocks until the PLL locks.

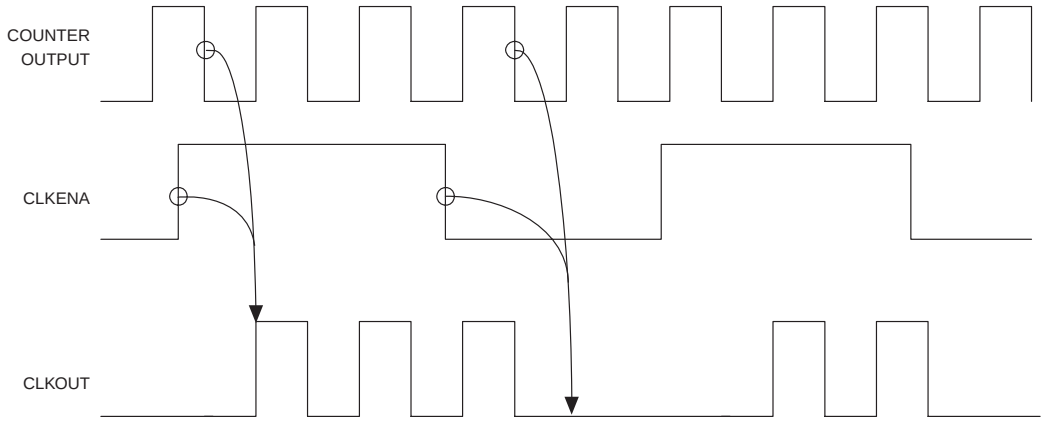
The `pfdena` signals control the phase frequency detector (PFD) output with a programmable gate. If you disable the PFD, the VCO operates at its last set value of control voltage and frequency with some long-term drift to a lower frequency. The system continues running when the PLL goes out of lock or the input clock is disabled. By maintaining the last locked frequency, the system has time to store its current settings before shutting down. You can either use your own control signal or a `clkloss` status signal to trigger `pfdena`.

The `clkena` signals control the enhanced PLL regional and global outputs. Each regional and global output port has its own `clkena` signal. The `clkena` signals synchronously disable or enable the clock at the PLL output port by gating the outputs of the *g* and *l* counters. The `clkena` signals are registered on the falling edge of the counter output clock to enable or disable the clock without glitches. Figure 2–57 shows the waveform example for a PLL clock port enable. The PLL can remain locked independent of the `clkena` signals since the loop-related counters are not affected. This feature is useful for applications that require a low power or sleep mode. Upon re-enabling, the PLL does not need a

resynchronization or relock period. The `clkena` signal can also disable clock outputs if the system is not tolerant to frequency overshoot during resynchronization.

The `extclkena` signals work in the same way as the `clkena` signals, but they control the external clock output counters (*e*0, *e*1, *e*2, and *e*3). Upon re-enabling, the PLL does not need a resynchronization or relock period unless the PLL is using external feedback mode. In order to lock in external feedback mode, the external output must drive the board trace back to the `FBIN` pin.

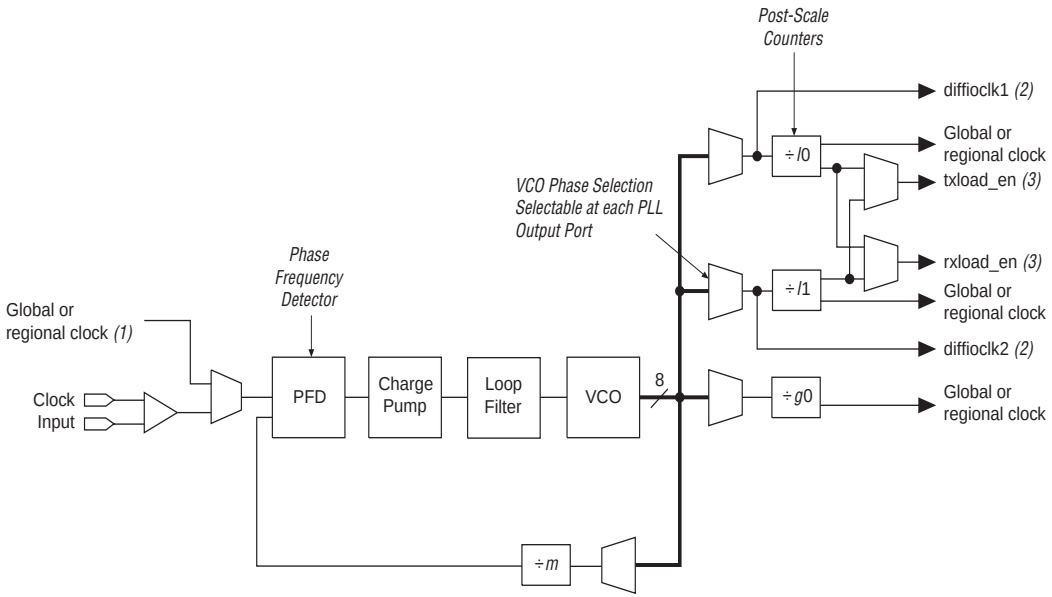
Figure 2–57. extclkena Signals



Fast PLLs

Stratix devices contain up to eight fast PLLs with high-speed serial interfacing ability, along with general-purpose features. Figure 2–58 shows a diagram of the fast PLL.

Figure 2–58. Stratix Device Fast PLL



Notes to Figure 2–58:

(1) The global or regional clock input can be driven by an output from another PLL or any dedicated CLK or FCLK pin. It cannot be driven by internally-generated global signals.

(2) In high-speed differential I/O support mode, this high-speed PLL clock feeds the SERDES. Stratix devices only support one rate of data transfer per fast PLL in high-speed differential I/O support mode.

(3) This signal is a high-speed differential I/O support SERDES control signal.

Clock Multiplication & Division

Stratix device fast PLLs provide clock synthesis for PLL output ports using *m* / (post scaler) scaling factors. The input clock is multiplied by the *m* feedback factor. Each output port has a unique post scale counter to divide down the high-frequency VCO. There is one multiply divider, *m*, per fast PLL with a range of 1 to 32. There are two post scale L dividers for regional and / or LVDS interface clocks, and *g*0 counter for global clock output port; all range from 1 to 32.

In the case of a high-speed differential interface, set the output counter to 1 to allow the high-speed VCO frequency to drive the SERDES. When used for clocking the SERDES, the *m* counter can range from 1 to 30. The VCO frequency is equal to $f_{IN} \times m$, where VCO frequency must be between 300 and 1000 MHz.

External Clock Inputs

Each fast PLL supports single-ended or differential inputs for source synchronous transmitters or for general-purpose use. Source-synchronous receivers support differential clock inputs. The fast PLL inputs are fed by `CLK[0..3]`, `CLK[8..11]`, and `FPLL[7..10]CLK` pins, as shown in Figure 2–50 on page 2–85.

Table 2–22 shows the I/O standards supported by fast PLL input pins.

Table 2–22. Fast PLL Port I/O Standards (Part 1 of 2)

I/O Standard	Input	
	INCLK	PLLENABLE
LVTTL	v	v
LVCMOS	v	v
2.5 V	v	
1.8 V	v	
1.5 V	v	
3.3-V PCI		
3.3-V PCI-X 1.0		
LVPECL	v	
3.3-V PCML	v	
LVDS	v	
HyperTransport technology	v	
Differential HSTL	v	
Differential SSTL		
3.3-V GTL		
3.3-V GTL+	v	
1.5-V HSTL Class I	v	
1.5-V HSTL Class II		
1.8-V HSTL Class I	v	
1.8-V HSTL Class II		
SSTL-18 Class I	v	
SSTL-18 Class II		
SSTL-2 Class I	v	

Table 2–22. Fast PLL Port I/O Standards (Part 2 of 2)		
I/O Standard	Input	
	INCLK	PLLENABLE
SSTL-2 Class II	v	
SSTL-3 Class I	v	
SSTL-3 Class II	v	
AGP (1× and 2×)		
CTT	v	

Table 2–23 shows the performance on each of the fast PLL clock inputs when using LVDS, LVPECL, 3.3-V PCML, or HyperTransport technology.

Table 2–23. LVDS Performance on Fast PLL Input	
Fast PLL Clock Input	Maximum Input Frequency (MHz)
CLK0, CLK2, CLK9, CLK11, FPLL7CLK, FPLL8CLK, FPLL9CLK, FPLL10CLK	717*(1)*
CLK1, CLK3, CLK8, CLK10	645

Note to Table 2–23:

(1) See the chapter *DC & Switching Characteristics* of the *Stratix Device Handbook, Volume 1* for more information.

External Clock Outputs

Each fast PLL supports differential or single-ended outputs for source-synchronous transmitters or for general-purpose external clocks. There are no dedicated external clock output pins. Any I/O pin can be driven by the fast PLL global or regional outputs as an external output pin. The I/O standards supported by any particular bank determines what standards are possible for an external clock output driven by the fast PLL in that bank.

Phase Shifting

Stratix device fast PLLs have advanced clock shift capability that enables programmable phase shifts. You can enter a phase shift (in degrees or time units) for each PLL clock output port or for all outputs together in one shift. You can perform phase shifting in time units with a resolution range of 125 to 416.66 ps. This resolution is a function of the VCO period, with the finest step being equal to an eighth (×0.125) of the VCO period.

Control Signals

The fast PLL has the same `lock` output, `pllenable` input, and `areset` input control signals as the enhanced PLL.

If the input clock stops and causes the PLL to lose lock, then the PLL must be reset for correct phase shift operation.

For more information on high-speed differential I/O support, see "High-Speed Differential I/O Support" on page 2–130.

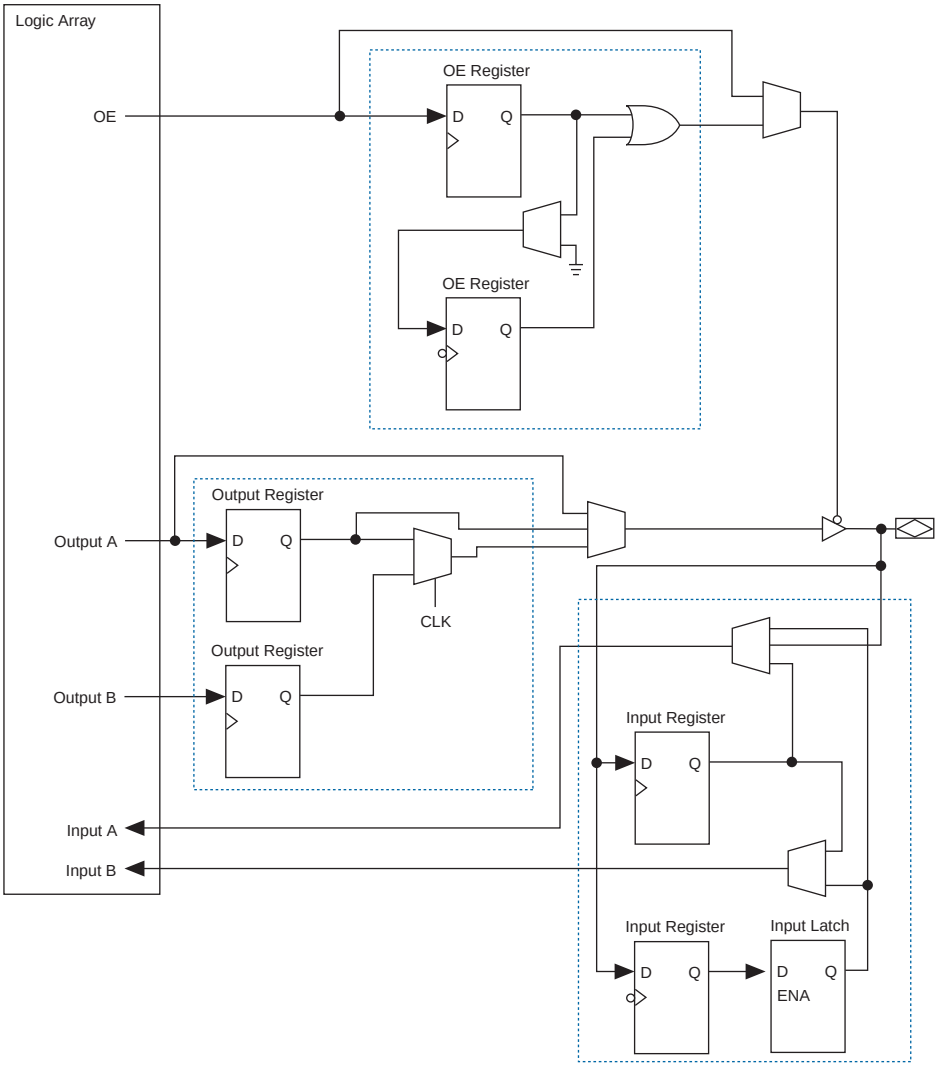
I/O Structure

IOEs provide many features, including:

- Dedicated differential and single-ended I/O buffers
- 3.3-V, 64-bit, 66-MHz PCI compliance
- 3.3-V, 64-bit, 133-MHz PCI-X 1.0 compliance
- Joint Test Action Group (JTAG) boundary-scan test (BST) support
- Differential on-chip termination for LVDS I/O standard
- Programmable pull-up during configuration
- Output drive strength control
- Slew-rate control
- Tri-state buffers
- Bus-hold circuitry
- Programmable pull-up resistors
- Programmable input and output delays
- Open-drain outputs
- DQ and DQS I/O pins
- Double-data rate (DDR) Registers

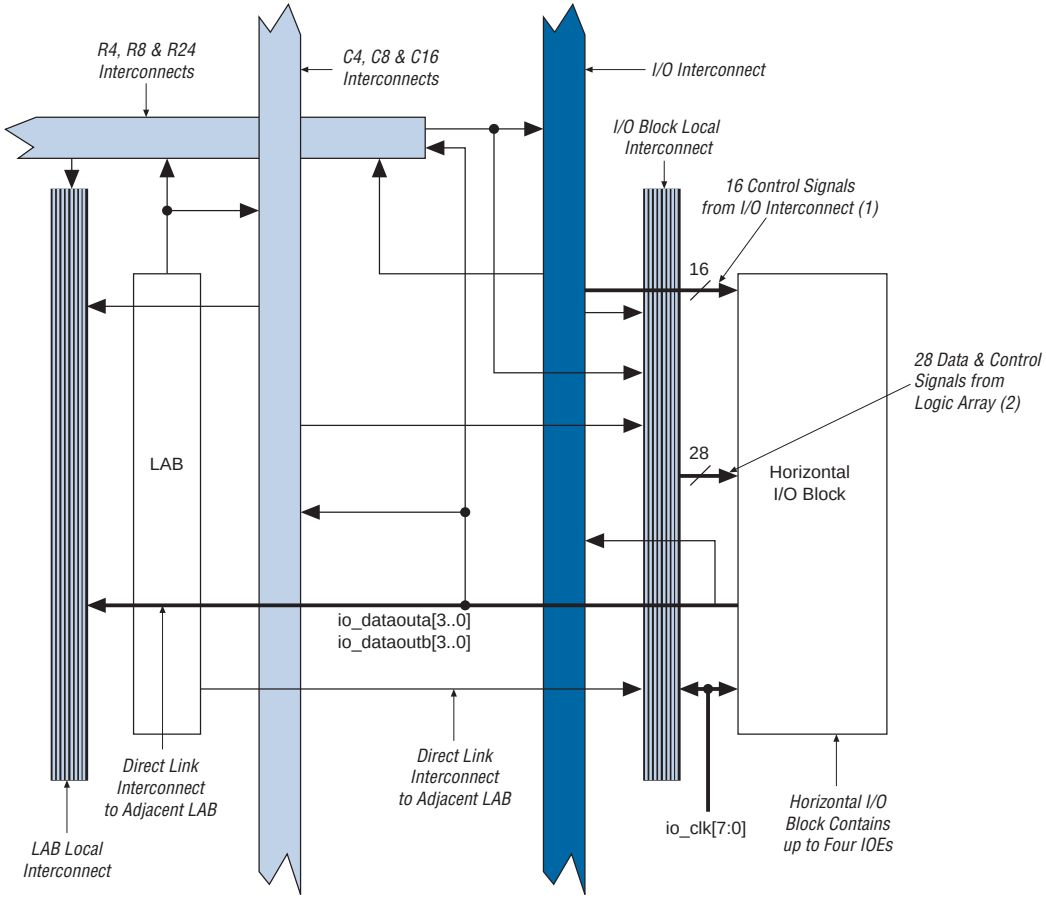
The IOE in Stratix devices contains a bidirectional I/O buffer, six registers, and a latch for a complete embedded bidirectional single data rate or DDR transfer. Figure 2–59 shows the Stratix IOE structure. The IOE contains two input registers (plus a latch), two output registers, and two output enable registers. The design can use both input registers and the latch to capture DDR input and both output registers to drive DDR outputs. Additionally, the design can use the output enable (OE) register for fast clock-to-output enable timing. The negative edge-clocked OE register is used for DDR SDRAM interfacing. The Quartus II software automatically duplicates a single OE register that controls multiple output or bidirectional pins.

Figure 2–59. Stratix IOE Structure



The IOEs are located in I/O blocks around the periphery of the Stratix device. There are up to four IOEs per row I/O block and six IOEs per column I/O block. The row I/O blocks drive row, column, or direct link interconnects. The column I/O blocks drive column interconnects. Figure 2–60 shows how a row I/O block connects to the logic array. Figure 2–61 shows how a column I/O block connects to the logic array.

Figure 2–60. Row I/O Block Connection to the Interconnect

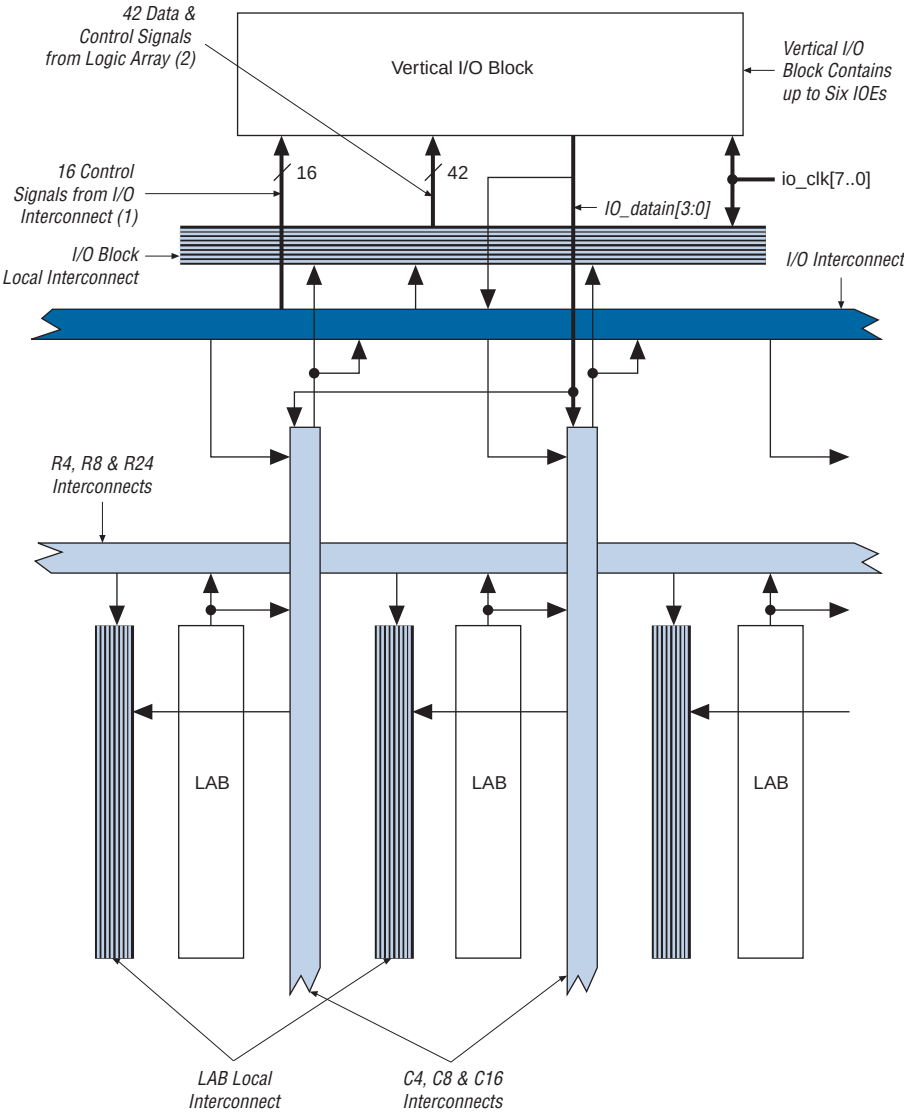


Notes to Figure 2–60:

(1) The 16 control signals are composed of four output enables `io_boe[3..0]`, four clock enables `io_bce[3..0]`, four clocks `io_clk[3..0]`, and four clear signals `io_bclr[3..0]`.

(2) The 28 data and control signals consist of eight data out lines: four lines each for DDR applications `io_dataouta[3..0]` and `io_dataoutb[3..0]`, four output enables `io_coe[3..0]`, four input clock enables `io_cce_in[3..0]`, four output clock enables `io_cce_out[3..0]`, four clocks `io_cclk[3..0]`, and four clear signals `io_cclr[3..0]`.

Figure 2–61. Column I/O Block Connection to the Interconnect



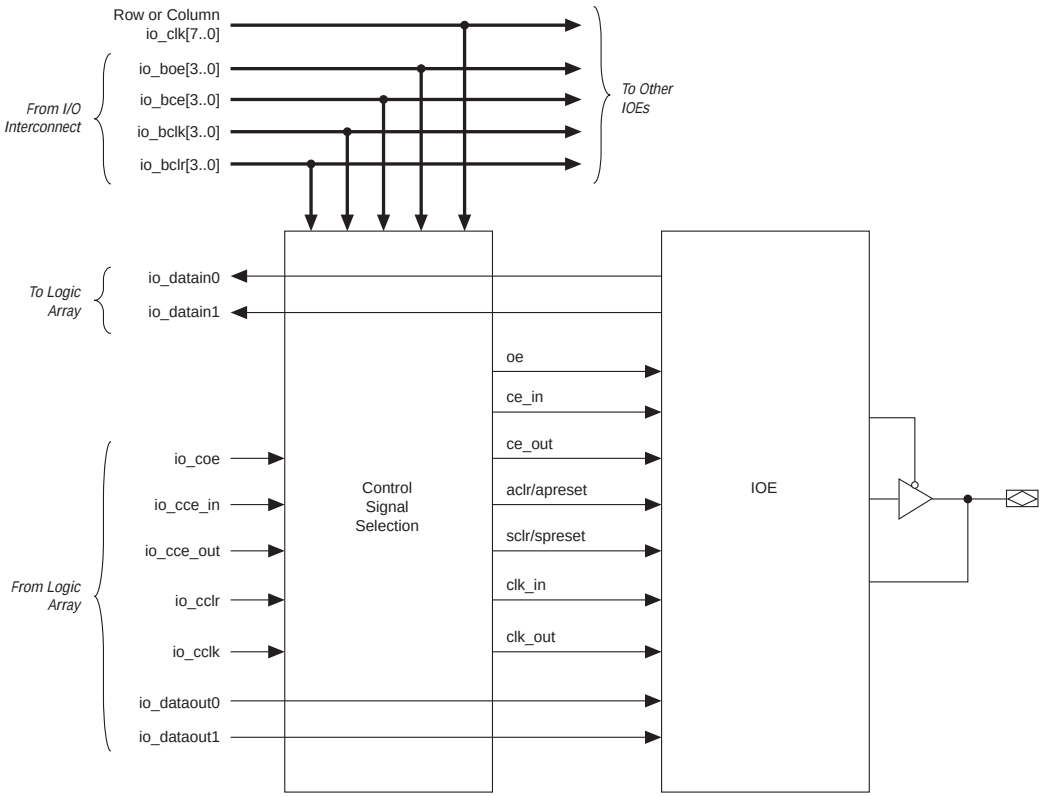
Notes to Figure 2–61:

(1) The 16 control signals are composed of four output enables `io_boe[3..0]`, four clock enables `io_bce[3..0]`, four clocks `io_bclk[3..0]`, and four clear signals `io_bclr[3..0]`.

(2) The 42 data and control signals consist of 12 data out lines; six lines each for DDR applications `io_dataouta[5..0]` and `io_dataoutb[5..0]`, six output enables `io_coe[5..0]`, six input clock enables `io_cce_in[5..0]`, six output clock enables `io_cce_out[5..0]`, six clocks `io_cclk[5..0]`, and six clear signals `io_cclr[5..0]`.

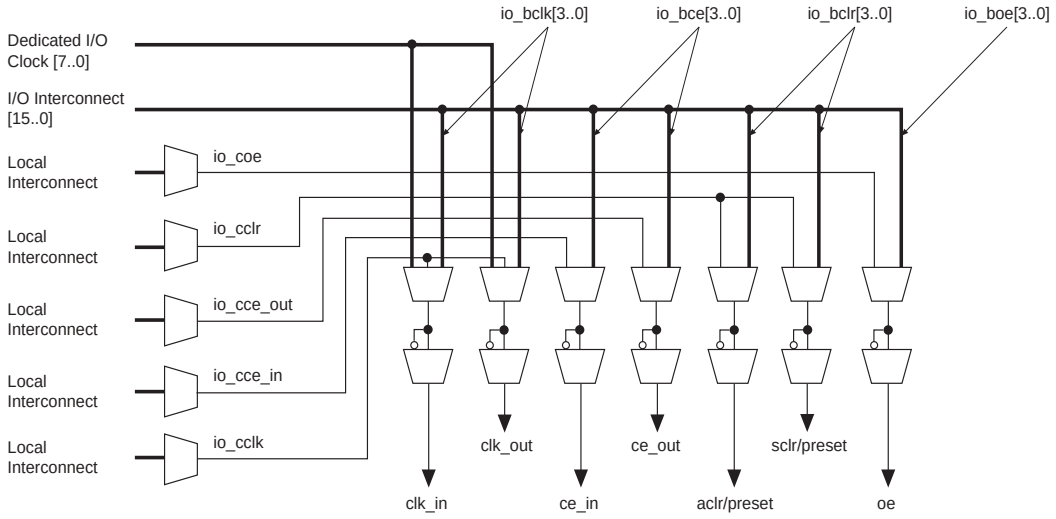
Stratix devices have an I/O interconnect similar to the R4 and C4 interconnect to drive high-fanout signals to and from the I/O blocks. There are 16 signals that drive into the I/O blocks composed of four output enables `io_boe[3..0]`, four clock enables `io_bce[3..0]`, four clocks `io_bclk[3..0]`, and four clear signals `io_bclr[3..0]`. The pin's `datain` signals can drive the IO interconnect, which in turn drives the logic array or other I/O blocks. In addition, the control and data signals can be driven from the logic array, providing a slower but more flexible routing resource. The row or column IOE clocks, `io_clk[7..0]`, provide a dedicated routing resource for low-skew, high-speed clocks. I/O clocks are generated from regional, global, or fast regional clocks (see "PLLs & Clock Networks" on page 2–73). Figure 2–62 illustrates the signal paths through the I/O block.

Figure 2–62. Signal Path through the I/O Block



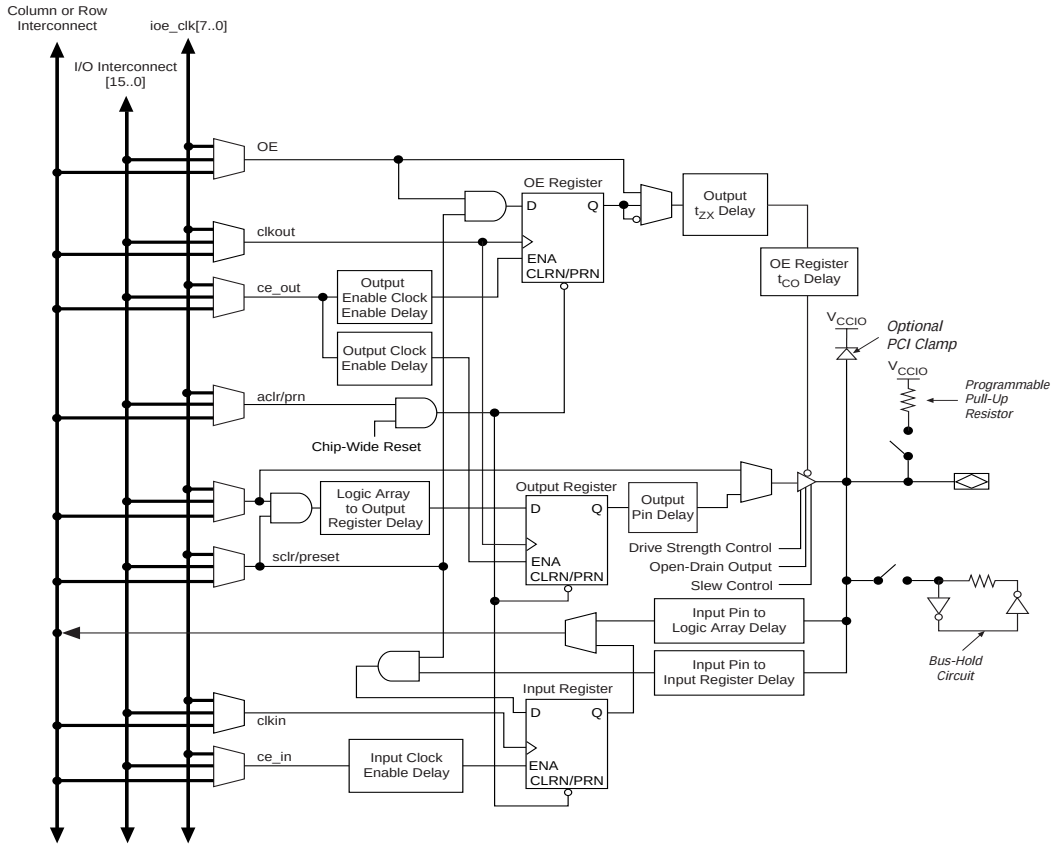
Each IOE contains its own control signal selection for the following control signals: `oe`, `ce_in`, `ce_out`, `aclr/preset`, `sclr/preset`, `clk_in`, and `clk_out`. Figure 2–63 illustrates the control signal selection.

Figure 2–63. Control Signal Selection per IOE



In normal bidirectional operation, the input register can be used for input data requiring fast setup times. The input register can have its own clock input and clock enable separate from the OE and output registers. The output register can be used for data requiring fast clock-to-output performance. The OE register can be used for fast clock-to-output enable timing. The OE and output register share the same clock source and the same clock enable source from local interconnect in the associated LAB, dedicated I/O clocks, and the column and row interconnects. Figure 2–64 shows the IOE in bidirectional configuration.

Figure 2–64. Stratix IOE in Bidirectional I/O Configuration Note (1)



Note to Figure 2–64:

(1) All input signals to the IOE can be inverted at the IOE.

The Stratix device IOE includes programmable delays that can be activated to ensure zero hold times, input IOE register-to-logic array register transfers, or logic array-to-output IOE register transfers.

A path in which a pin directly drives a register may require the delay to ensure zero hold time, whereas a path in which a pin drives a register through combinatorial logic may not require the delay. Programmable delays exist for decreasing input-pin-to-logic-array and IOE input register delays. The Quartus II Compiler can program these delays to automatically minimize setup time while providing a zero hold time. Programmable delays can increase the register-to-pin delays for output

and / or output enable registers. A programmable delay exists to increase the t_{ZX} delay to the output pin, which is required for ZBT interfaces. Table 2–24 shows the programmable delays for Stratix devices.

Table 2–24. Stratix Programmable Delay Chain

Programmable Delays	Quartus II Logic Option
Input pin to logic array delay	Decrease input delay to internal cells
Input pin to input register delay	Decrease input delay to input register
Output pin delay	Increase delay to output pin
Output enable register t_{CO} delay	Increase delay to output enable pin
Output t_{ZX} delay	Increase t_{ZX} delay to output pin
Output clock enable delay	Increase output clock enable delay
Input clock enable delay	Increase input clock enable delay
Logic array to output register delay	Decrease input delay to output register
Output enable clock enable delay	Increase output enable clock enable delay

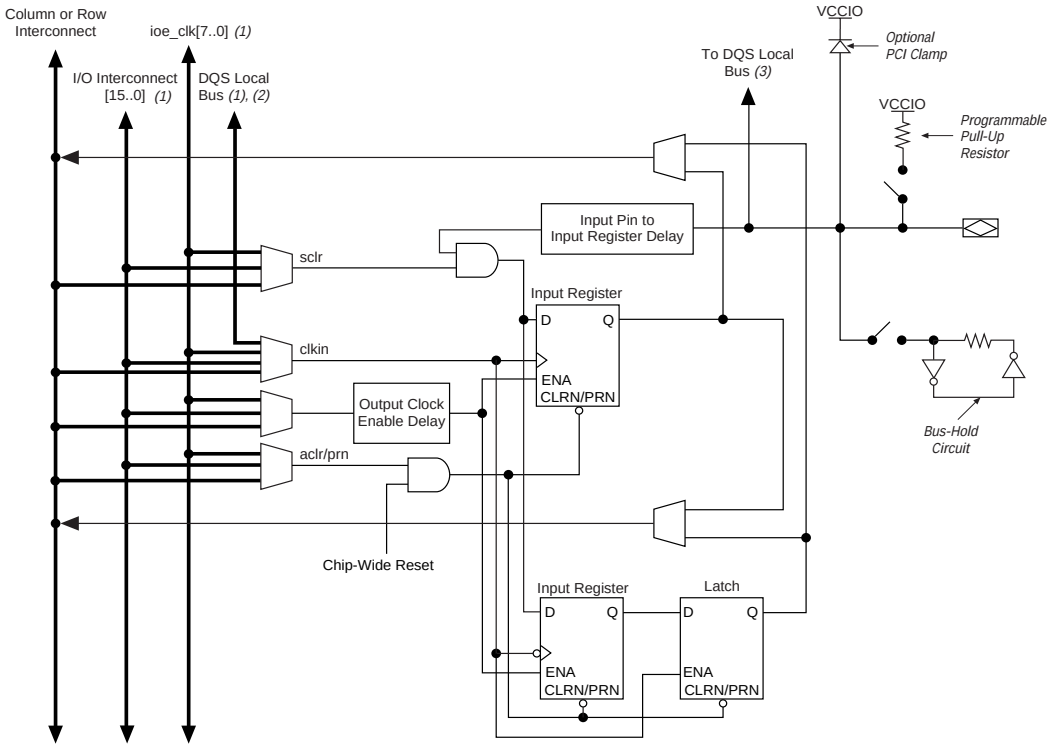
The IOE registers in Stratix devices share the same source for clear or preset. You can program preset or clear for each individual IOE. You can also program the registers to power up high or low after configuration is complete. If programmed to power up low, an asynchronous clear can control the registers. If programmed to power up high, an asynchronous preset can control the registers. This feature prevents the inadvertent activation of another device's active-low input upon power-up. If one register in an IOE uses a preset or clear signal then all registers in the IOE must use that same signal if they require preset or clear. Additionally a synchronous reset signal is available for the IOE registers.

Double-Data Rate I/O Pins

Stratix devices have six registers in the IOE, which support DDR interfacing by clocking data on both positive and negative clock edges. The IOEs in Stratix devices support DDR inputs, DDR outputs, and bidirectional DDR modes.

When using the IOE for DDR inputs, the two input registers clock double rate input data on alternating edges. An input latch is also used within the IOE for DDR input acquisition. The latch holds the data that is present during the clock high times. This allows both bits of data to be synchronous with the same clock edge (either rising or falling). Figure 2–65 shows an IOE configured for DDR input. Figure 2–66 shows the DDR input timing diagram.

Figure 2–65. Stratix IOE in DDR Input I/O Configuration Note (1)



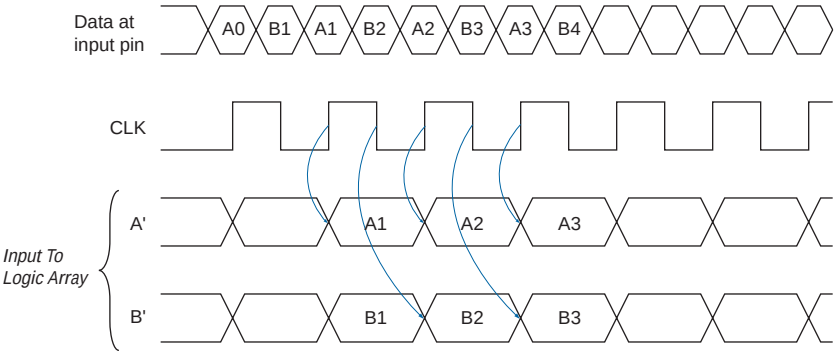
Notes to Figure 2–65:

(1) All input signals to the IOE can be inverted at the IOE.

(2) This signal connection is only allowed on dedicated DQ function pins.

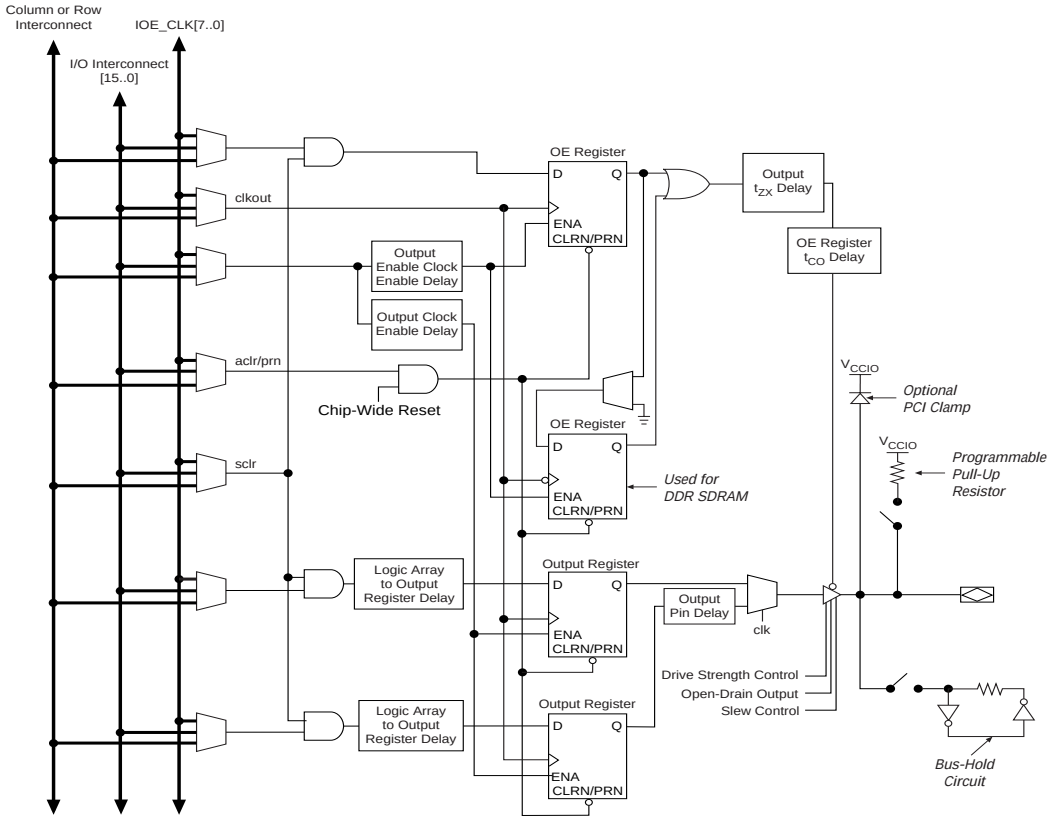
(3) This signal is for dedicated DQS function pins only.

Figure 2–66. Input Timing Diagram in DDR Mode



When using the IOE for DDR outputs, the two output registers are configured to clock two data paths from LEs on rising clock edges. These output registers are multiplexed by the clock to drive the output pin at a ×2 rate. One output register clocks the first bit out on the clock high time, while the other output register clocks the second bit out on the clock low time. Figure 2–67 shows the IOE configured for DDR output. Figure 2–68 shows the DDR output timing diagram.

Figure 2–67. Stratix IOE in DDR Output I/O Configuration *Notes (1), (2)*

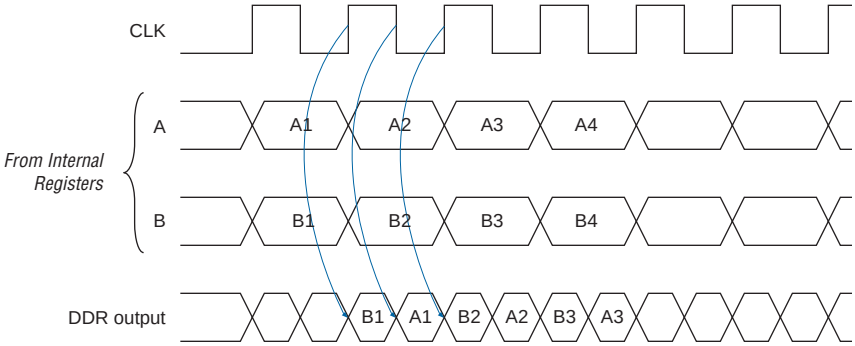


Notes to Figure 2–67:

(1) All input signals to the IOE can be inverted at the IOE.

(2) The tristate is by default active high. It can, however, be designed to be active low.

Figure 2–68. Output Timing Diagram in DDR Mode



The Stratix IOE operates in bidirectional DDR mode by combining the DDR input and DDR output configurations. Stratix device I/O pins transfer data on a DDR bidirectional bus to support DDR SDRAM. The negative-edge-clocked OE register holds the OE signal inactive until the falling edge of the clock. This is done to meet DDR SDRAM timing requirements.

External RAM Interfacing

Stratix devices support DDR SDRAM at up to 200 MHz (400-Mbps data rate) through dedicated phase-shift circuitry, QDR and QDRII SRAM interfaces up to 167 MHz, and ZBT SRAM interfaces up to 200 MHz. Stratix devices also provide preliminary support for reduced latency DRAM II (RLDRAM II) at rates up to 200 MHz through the dedicated phase-shift circuitry.

1 In addition to the required signals for external memory interfacing, Stratix devices offer the optional clock enable signal. By default the Quartus II software sets the clock enable signal high, which tells the output register to update with new values. The output registers hold their own values if the design sets the clock enable signal low. See Figure 2–64.

f To find out more about the DDR SDRAM specification, see the JEDEC web site (**www.jedec.org**). For information on memory controller megafunctions for Stratix devices, see the Altera web site (**www.altera.com**). See *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices* for more information on DDR SDRAM interface in Stratix. Also see *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices* and *AN 329: ZBT SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

Tables 2–25 and 2–26 show the performance specification for DDR SDRAM, RLDRAM II, QDR SRAM, QDRII SRAM, and ZBT SRAM interfaces in EP1S10 through EP1S40 devices and in EP1S60 and EP1S80 devices. The DDR SDRAM and QDR SRAM numbers in Table 2–25 have been verified with hardware characterization with third-party DDR SDRAM and QDR SRAM devices over temperature and voltage extremes.

Table 2–25. External RAM Support in EP1S10 through EP1S40 Devices

DDR Memory Type	I/O Standard	Maximum Clock Rate (MHz)						
		-5 Speed Grade	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade	
		Flip-Chip	Flip-Chip	Wire-Bond	Flip-Chip	Wire-Bond	Flip-Chip	Wire-Bond
DDR SDRAM *(1)*, *(2)*	SSTL-2	200	167	133	133	100	100	100
DDR SDRAM - side banks *(2)*, *(3)*, *(4)*	SSTL-2	150	133	110	133	100	100	100
RLDRAM II *(4)*	1.8-V HSTL	200	*(5)*	*(5)*	*(5)*	*(5)*	*(5)*	*(5)*
QDR SRAM *(6)*	1.5-V HSTL	167	167	133	133	100	100	100
QDRII SRAM *(6)*	1.5-V HSTL	200	167	133	133	100	100	100
ZBT SRAM *(7)*	LVTTL	200	200	200	167	167	133	133

Notes to Table 2–25:

(1) These maximum clock rates apply if the Stratix device uses DQS phase-shift circuitry to interface with DDR SDRAM. DQS phase-shift circuitry is only available in the top and bottom I/O banks (I/O banks 3, 4, 7, and 8).

(2) For more information on DDR SDRAM, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*.

(3) DDR SDRAM is supported on the Stratix device side I/O banks (I/O banks 1, 2, 5, and 6) without dedicated DQS phase-shift circuitry. The read DQS signal is ignored in this mode.

(4) These performance specifications are preliminary.

(5) This device does not support RLDRAM II.

(6) For more information on QDR or QDRII SRAM, see *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

(7) For more information on ZBT SRAM, see *AN 329: ZBT SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

Table 2–26. External RAM Support in EP1S60 & EP1S80 Devices

DDR Memory Type	I/O Standard	Maximum Clock Rate (MHz)		
		-5 Speed Grade	-6 Speed Grade	-7 Speed Grade
DDR SDRAM *(1)*, *(2)*	SSTL-2	167	167	133
DDR SDRAM - side banks *(2)*, *(3)*	SSTL-2	150	133	133
QDR SRAM *(4)*	1.5-V HSTL	133	133	133
QDRII SRAM *(4)*	1.5-V HSTL	167	167	133
ZBT SRAM *(5)*	LVTTL	200	200	167

Notes to Table 2–26:

(1) These maximum clock rates apply if the Stratix device uses DQS phase-shift circuitry to interface with DDR SDRAM. DQS phase-shift circuitry is only available in the top and bottom I/O banks (I/O banks 3, 4, 7, and 8).

(2) For more information on DDR SDRAM, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices.*

(3) DDR SDRAM is supported on the Stratix device side I/O banks (I/O banks 1, 2, 5, and 6) without dedicated DQS phase-shift circuitry. The read DQS signal is ignored in this mode. Numbers are preliminary.

(4) For more information on QDR or QDRII SRAM, see *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

(5) For more information on ZBT SRAM, see *AN 329: ZBT SRAM Controller Reference Design for Stratix & Stratix GX Devices.*

In addition to six I/O registers and one input latch in the IOE for interfacing to these high-speed memory interfaces, Stratix devices also have dedicated circuitry for interfacing with DDR SDRAM. In every Stratix device, the I/O banks at the top (I/O banks 3 and 4) and bottom (I/O banks 7 and 8) of the device support DDR SDRAM up to 200 MHz. These pins support DQS signals with DQ bus modes of ×8, ×16, or ×32.

Table 2–27 shows the number of DQ and DQS buses that are supported per device.

Table 2–27. DQS & DQ Bus Mode Support (Part 1 of 2) Note (1)

Device	Package	Number of ×8 Groups	Number of ×16 Groups	Number of ×32 Groups
EP1S10	672-pin BGA 672-pin FineLine BGA	12 *(2)*	0	0
	484-pin FineLine BGA 780-pin FineLine BGA	16 *(3)*	0	4
EP1S20	484-pin FineLine BGA	18*(4)*	7 *(5)*	4
	672-pin BGA 672-pin FineLine BGA	16*(3)*	7 *(5)*	4
	780-pin FineLine BGA	20	7 *(5)*	4

Table 2–27. DQS & DQ Bus Mode Support (Part 2 of 2) Note (1)

Device	Package	Number of ×8 Groups	Number of ×16 Groups	Number of ×32 Groups
EP1S25	672-pin BGA 672-pin FineLine BGA	16 *(3)*	8	4
	780-pin FineLine BGA 1,020-pin FineLine BGA	20	8	4
EP1S30	956-pin BGA 780-pin FineLine BGA 1,020-pin FineLine BGA	20	8	4
EP1S40	956-pin BGA 1,020-pin FineLine BGA 1,508-pin FineLine BGA	20	8	4
EP1S60	956-pin BGA 1,020-pin FineLine BGA 1,508-pin FineLine BGA	20	8	4
EP1S80	956-pin BGA 1,508-pin FineLine BGA 1,923-pin FineLine BGA	20	8	4

Notes to Table 2–27:

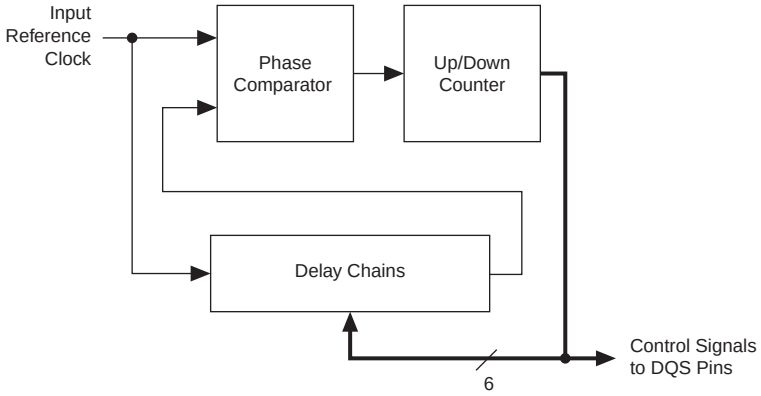
(1) See the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook, Volume 2* for V_{REF} guidelines.
(2) These packages have six groups in I/O banks 3 and 4 and six groups in I/O banks 7 and 8.
(3) These packages have eight groups in I/O banks 3 and 4 and eight groups in I/O banks 7 and 8.
(4) This package has nine groups in I/O banks 3 and 4 and nine groups in I/O banks 7 and 8.
(5) These packages have three groups in I/O banks 3 and 4 and four groups in I/O banks 7 and 8.

A compensated delay element on each DQS pin automatically aligns input DQS synchronization signals with the data window of their corresponding DQ data signals. The DQS signals drive a local DQS bus in the top and bottom I/O banks. This DQS bus is an additional resource to the I/O clocks and is used to clock DQ input registers with the DQS signal.

Two separate single phase-shifting reference circuits are located on the top and bottom of the Stratix device. Each circuit is driven by a system reference clock through the `CLK` pins that is the same frequency as the DQS signal. Clock pins `CLK[15..12]p` feed the phase-shift circuitry on the top of the device and clock pins `CLK[7..4]p` feed the phase-shift circuitry on the bottom of the device. The phase-shifting reference circuit on the top of the device controls the compensated delay elements for all 10 DQS pins located at the top of the device. The phase-shifting reference circuit on the bottom of the device controls the compensated delay elements for all 10 DQS pins located on the bottom of the device. All 10 delay elements (DQS signals) on either the top or bottom of the device

shift by the same degree amount. For example, all 10 DQS pins on the top of the device can be shifted by 90° and all 10 DQS pins on the bottom of the device can be shifted by 72°. The reference circuits require a maximum of 256 system reference clock cycles to set the correct phase on the DQS delay elements. Figure 2–69 illustrates the phase-shift reference circuit control of each DQS delay shift on the top of the device. This same circuit is duplicated on the bottom of the device.

Figure 2–69. Simplified Diagram of the DQS Phase-Shift Circuitry



See the *External Memory Interfaces* chapter in the *Stratix Device Handbook, Volume 2* for more information on external memory interfaces.

Programmable Drive Strength

The output buffer for each Stratix device I/O pin has a programmable drive strength control for certain I/O standards. The LVTTL and LVCMOS standard has several levels of drive strength that the user can control. SSTL-3 Class I and II, SSTL-2 Class I and II, HSTL Class I and II, and 3.3-V GTL+ support a minimum setting, the lowest drive strength that guarantees the I_{OH}/I_{OL} of the standard. Using minimum settings provides signal slew rate control to reduce system noise and signal overshoot.

Table 2–28 shows the possible settings for the I/O standards with drive strength control.

Table 2–28. Programmable Drive Strength

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA)
3.3-V LVTTL	24 *(1)*, 16, 12, 8, 4
3.3-V LVCMOS	24 *(2)*, 12 *(1)*, 8, 4, 2
2.5-V LVTTL/LVCMOS	16 *(1)*, 12, 8, 2
1.8-V LVTTL/LVCMOS	12 *(1)*, 8, 2
1.5-V LVCMOS	8 *(1)*, 4, 2
GTL/GTL+ 1.5-V HSTL Class I and II 1.8-V HSTL Class I and II SSTL-3 Class I and II SSTL-2 Class I and II SSTL-18 Class I and II	Support max and min strength

Notes to Table 2–28:

(1) This is the Quartus II software default current setting.

(2) I/O banks 1, 2, 5, and 6 do not support this setting.

Quartus II software version 4.2 and later will report current strength as "PCI Compliant" for 3.3-V PCI, 3.3-V PCI-X 1.0, and Compact PCI I/O standards.

Stratix devices support series on-chip termination (OCT) using programmable drive strength. For more information, contact your Altera Support Representative.

Open-Drain Output

Stratix devices provide an optional open-drain (equivalent to an open-collector) output for each I/O pin. This open-drain output enables the device to provide system-level control signals (e.g., interrupt and write-enable signals) that can be asserted by any of several devices.

Slew-Rate Control

The output buffer for each Stratix device I/O pin has a programmable output slew-rate control that can be configured for low-noise or high-speed performance. A faster slew rate provides high-speed transitions for high-performance systems. However, these fast transitions may introduce noise transients into the system. A slow slew rate reduces system noise, but adds a nominal delay to rising and falling edges. Each

I/O pin has an individual slew-rate control, allowing you to specify the slew rate on a pin-by-pin basis. The slew-rate control affects both the rising and falling edges.

Bus Hold

Each Stratix device I/O pin provides an optional bus-hold feature. The bus-hold circuitry can weakly hold the signal on an I/O pin at its last-driven state. Since the bus-hold feature holds the last-driven state of the pin until the next input signal is present, an external pull-up or pull-down resistor is not needed to hold a signal level when the bus is tri-stated.

Table 2–29 shows bus hold support for different pin types.

Table 2–29. Bus Hold Support

Pin Type	Bus Hold
I/O pins	v
CLK[15..0]	
CLK[0,1,2,3,8,9,10,11]	
FCLK	v
FPLL[7..10]CLK	

The bus-hold circuitry also pulls undriven pins away from the input threshold voltage where noise can cause unintended high-frequency switching. You can select this feature individually for each I/O pin. The bus-hold output drives no higher than V_{CCIO} to prevent overdriving signals. If the bus-hold feature is enabled, the programmable pull-up option cannot be used. Disable the bus-hold feature when using open-drain outputs with the GTL+ I/O standard or when the I/O pin has been configured for differential signals.

The bus-hold circuitry uses a resistor with a nominal resistance (R_{BH}) of approximately 7 kΩto weakly pull the signal level to the last-driven state. See the *DC & Switching Characteristics* chapter of the *Stratix Device Handbook, Volume 1* for the specific sustaining current driven through this resistor and overdrive current used to identify the next-driven input level. This information is provided for each V_{CCIO} voltage level.

The bus-hold circuitry is active only after configuration. When going into user mode, the bus-hold circuit captures the value on the pin present at the end of configuration.

Programmable Pull-Up Resistor

Each Stratix device I/O pin provides an optional programmable pull-up resistor during user mode. If this feature is enabled for an I/O pin, the pull-up resistor (typically 25 kΩ) weakly holds the output to the V_{CCIO} level of the output pin's bank. Table 2–30 shows which pin types support the weak pull-up resistor feature.

Table 2–30. Programmable Weak Pull-Up Resistor Support

Pin Type	Programmable Weak Pull-Up Resistor
I/O pins	v
`CLK[15..0]`	
`FCLK`	v
`FPLL[7..10]CLK`	
Configuration pins	
JTAG pins	v *(1)*

Note to Table 2–30:

(1) TDO pins do not support programmable weak pull-up resistors.

Advanced I/O Standard Support

Stratix device IOEs support the following I/O standards:

- LVTTL
- LVCMOS
- 1.5 V
- 1.8 V
- 2.5 V
- 3.3-V PCI
- 3.3-V PCI-X 1.0
- 3.3-V AGP (1× and 2×)
- LVDS
- LVPECL
- 3.3-V PCML
- HyperTransport
- Differential HSTL (on input/output clocks only)
- Differential SSTL (on output column clock pins only)
- GTL/GTL+
- 1.5-V HSTL Class I and II

- 1.8-V HSTL Class I and II
- SSTL-3 Class I and II
- SSTL-2 Class I and II
- SSTL-18 Class I and II
- CTT

Table 2–31 describes the I/O standards supported by Stratix devices.

Table 2–31. Stratix Supported I/O Standards

I/O Standard	Type	Input Reference Voltage (V_{REF}) (V)	Output Supply Voltage (V_{CCIO}) (V)	Board Termination Voltage (V_{TT}) (V)
LVTTL	Single-ended	N/A	3.3	N/A
LVCMOS	Single-ended	N/A	3.3	N/A
2.5 V	Single-ended	N/A	2.5	N/A
1.8 V	Single-ended	N/A	1.8	N/A
1.5 V	Single-ended	N/A	1.5	N/A
3.3-V PCI	Single-ended	N/A	3.3	N/A
3.3-V PCI-X 1.0	Single-ended	N/A	3.3	N/A
LVDS	Differential	N/A	3.3	N/A
LVPECL	Differential	N/A	3.3	N/A
3.3-V PCML	Differential	N/A	3.3	N/A
HyperTransport	Differential	N/A	2.5	N/A
Differential HSTL *(1)*	Differential	0.75	1.5	0.75
Differential SSTL *(2)*	Differential	1.25	2.5	1.25
GTL	Voltage-referenced	0.8	N/A	1.20
GTL+	Voltage-referenced	1.0	N/A	1.5
1.5-V HSTL Class I and II	Voltage-referenced	0.75	1.5	0.75
1.8-V HSTL Class I and II	Voltage-referenced	0.9	1.8	0.9
SSTL-18 Class I and II	Voltage-referenced	0.90	1.8	0.90
SSTL-2 Class I and II	Voltage-referenced	1.25	2.5	1.25
SSTL-3 Class I and II	Voltage-referenced	1.5	3.3	1.5
AGP (1× and 2°)	Voltage-referenced	1.32	3.3	N/A
CTT	Voltage-referenced	1.5	3.3	1.5

Notes to Table 2–31:

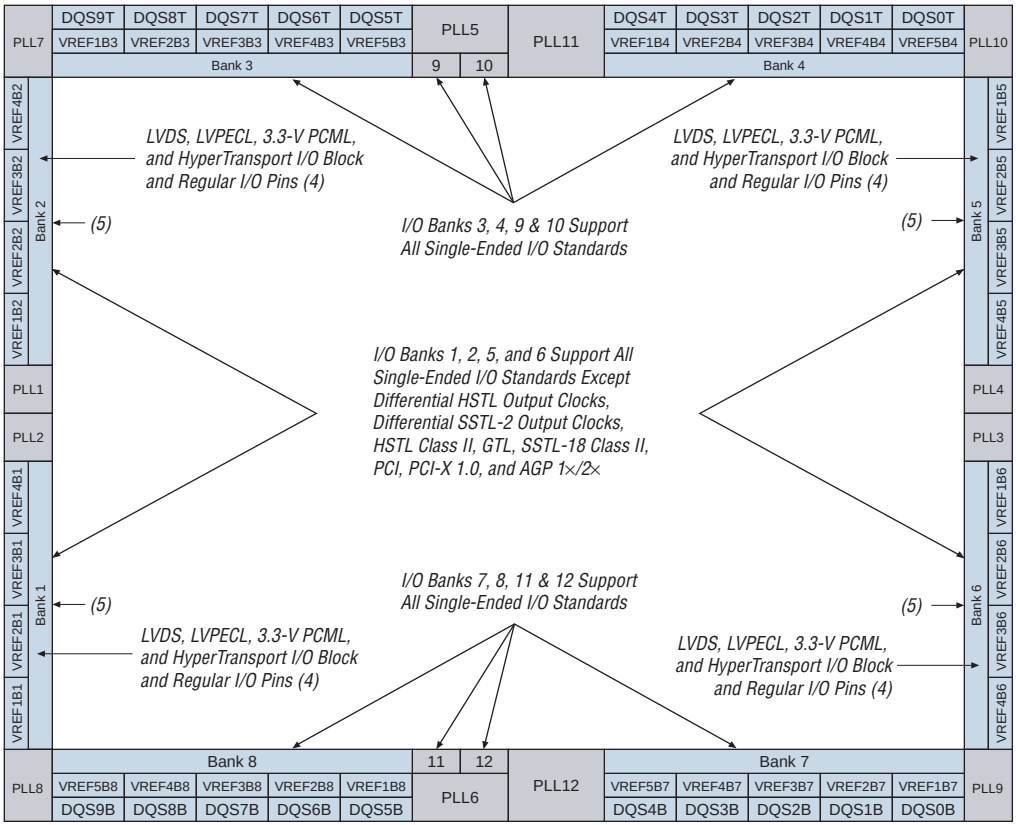
(1) This I/O standard is only available on input and output clock pins.

(2) This I/O standard is only available on output column clock pins.

f For more information on I/O standards supported by Stratix devices, see the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter of the *Stratix Device Handbook, Volume 2*.

Stratix devices contain eight I/O banks in addition to the four enhanced PLL external clock out banks, as shown in Figure 2–70. The four I/O banks on the right and left of the device contain circuitry to support high-speed differential I/O for LVDS, LVPECL, 3.3-V PCML, and HyperTransport inputs and outputs. These banks support all I/O standards listed in Table 2–31 except PCI I/O pins or PCI-X 1.0, GTL, SSTL-18 Class II, and HSTL Class II outputs. The top and bottom I/O banks support all single-ended I/O standards. Additionally, Stratix devices support four enhanced PLL external clock output banks, allowing clock output capabilities such as differential support for SSTL and HSTL. Table 2–32 shows I/O standard support for each I/O bank.

Figure 2–70. Stratix I/O Banks *Notes (1), (2), (3)*



Notes to Figure 2–70:

(1) Figure 2–70 is a top view of the silicon die. This will correspond to a top-down view for non-flip-chip packages, but will be a reverse view for flip-chip packages.

(2) Figure 2–70 is a graphic representation only. See the device pin-outs on the web (**www.altera.com**) and the Quartus II software for exact locations.

(3) Banks 9 through 12 are enhanced PLL external clock output banks.

(4) If the high-speed differential I/O pins are not used for high-speed differential signaling, they can support all of the I/O standards except HSTL Class I and II, GTL, SSTL-18 Class II, PCI, PCI-X 1.0, and AGP 1× /2×.

(5) For guidelines for placing single-ended I/O pads next to differential I/O pads, see the *Selectable I/O Standards in Stratix and Stratix GX Devices* chapter in the *Stratix Device Handbook, Volume 2*.

Table 2–32 shows I/O standard support for each I/O bank.

Table 2–32. I/O Support by Bank (Part 1 of 2)

I/O Standard	Top & Bottom Banks (3, 4, 7 & 8)	Left & Right Banks (1, 2, 5 & 6)	Enhanced PLL External Clock Output Banks (9, 10, 11 & 12)
LVTTL	v	v	v
LVCMOS	v	v	v
2.5 V	v	v	v
1.8 V	v	v	v
1.5 V	v	v	v
3.3-V PCI	v		v
3.3-V PCI-X 1.0	v		v
LVPECL		v	v
3.3-V PCML		v	v
LVDS		v	v
HyperTransport technology		v	v
Differential HSTL (clock inputs)	v	v	
Differential HSTL (clock outputs)			v
Differential SSTL (clock outputs)			v
3.3-V GTL	v		v
3.3-V GTL+	v	v	v
1.5-V HSTL Class I	v	v	v
1.5-V HSTL Class II	v		v
1.8-V HSTL Class I	v	v	v
1.8-V HSTL Class II	v		v
SSTL-18 Class I	v	v	v
SSTL-18 Class II	v		v
SSTL-2 Class I	v	v	v
SSTL-2 Class II	v	v	v
SSTL-3 Class I	v	v	v

Table 2–32. I/O Support by Bank (Part 2 of 2)

I/O Standard	Top & Bottom Banks (3, 4, 7 & 8)	Left & Right Banks (1, 2, 5 & 6)	Enhanced PLL External Clock Output Banks (9, 10, 11 & 12)
SSTL-3 Class II	v	v	v
AGP (1× and 2×)	v		v
CTT	v	v	v

Each I/O bank has its own VCCIO pins. A single device can support 1.5-, 1.8-, 2.5-, and 3.3-V interfaces; each bank can support a different standard independently. Each bank also has dedicated VREF pins to support any one of the voltage-referenced standards (such as SSTL-3) independently.

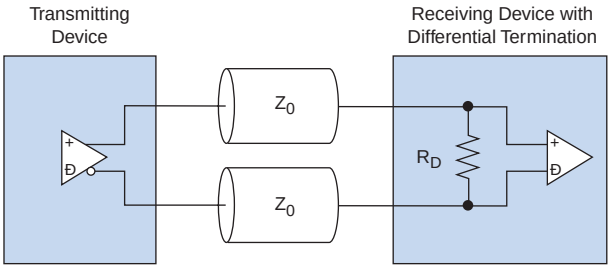
Each I/O bank can support multiple standards with the same V_{CCIO} for input and output pins. Each bank can support one voltage-referenced I/O standard. For example, when V_{CCIO} is 3.3 V, a bank can support LVTTL, LVCMOS, 3.3-V PCI, and SSTL-3 for inputs and outputs.

Differential On-Chip Termination

Stratix devices provide differential on-chip termination (LVDS I/O standard) to reduce reflections and maintain signal integrity. Differential on-chip termination simplifies board design by minimizing the number of external termination resistors required. Termination can be placed inside the package, eliminating small stubs that can still lead to reflections. The internal termination is designed using transistors in the linear region of operation.

Stratix devices support internal differential termination with a nominal resistance value of 137.5 Ω for LVDS input receiver buffers. LVPECL signals require an external termination resistor. Figure 2–71 shows the device with differential termination.

Figure 2–71. LVDS Input Differential On-Chip Termination



I/O banks on the left and right side of the device support LVDS receiver (far-end) differential termination.

Table 2–33 shows the Stratix device differential termination support.

Table 2–33. Differential Termination Supported by I/O Banks

Differential Termination Support	I/O Standard Support	Top & Bottom Banks (3, 4, 7 & 8)	Left & Right Banks (1, 2, 5 & 6)
Differential termination *(1)*, *(2)*	LVDS		v

Notes to Table 2–33:
(1) Clock pin `CLK0`, `CLK2`, `CLK9`, `CLK11`, and pins `FPLL[7..10]CLK` do not support differential termination.
(2) Differential termination is only supported for LVDS because of a 3.3-V V_{CCIO}.

Table 2–34 shows the termination support for different pin types.

Table 2–34. Differential Termination Support Across Pin Types

Pin Type	R_D
Top and bottom I/O banks (3, 4, 7, and 8)	
`DIFFIO_RX[]`	v
`CLK[0,2,9,11],CLK[4-7],CLK[12-15]`	
`CLK[1,3,8,10]`	v
`FCLK`	
`FPLL[7..10]CLK`	

The differential on-chip resistance at the receiver input buffer is 118 Ω ±20 %.

However, there is additional resistance present between the device ball and the input of the receiver buffer, as shown in Figure 2–72. This resistance is because of package trace resistance (which can be calculated as the resistance from the package ball to the pad) and the parasitic layout metal routing resistance (which is shown between the pad and the intersection of the on-chip termination and input buffer).

Figure 2–72. Differential Resistance of LVDS Differential Pin Pair (R_D)

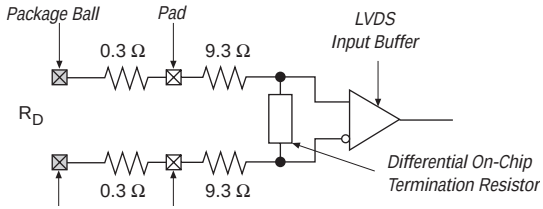


Table 2–35 defines the specification for internal termination resistance for commercial devices.

Table 2–35. Differential On-Chip Termination

Symbol	Description	Conditions	Resistance			Unit
			Min	Typ	Max	
R_D *(2)*	Internal differential termination for LVDS	Commercial *(1)*, *(3)*	110	135	165	W
		Industrial *(2)*, *(3)*	100	135	170	W

Notes to Table 2–35:

(1) Data measured over minimum conditions (T_j = 0 C, V_{CCIO} +5%) and maximum conditions (T_j = 85 C, V_{CCIO} = –5%).

(2) Data measured over minimum conditions (T_j = –40 C, V_{CCIO} +5%) and maximum conditions (T_j = 100 C, V_{CCIO} = –5%).

(3) LVDS data rate is supported for 840 Mbps using internal differential termination.

MultiVolt I/O Interface

The Stratix architecture supports the MultiVolt I/O interface feature, which allows Stratix devices in all packages to interface with systems of different supply voltages.

The Stratix `VCCINT` pins must always be connected to a 1.5-V power supply. With a 1.5-V V_{CCINT} level, input pins are 1.5-V, 1.8-V, 2.5-V, and 3.3-V tolerant. The `VCCIO` pins can be connected to either a 1.5-V, 1.8-V, 2.5-V, or 3.3-V power supply, depending on the output requirements.

The output levels are compatible with systems of the same voltage as the power supply (i.e., when VCCIO pins are connected to a 1.5-V power supply, the output levels are compatible with 1.5-V systems). When VCCIO pins are connected to a 3.3-V power supply, the output high is 3.3 V and is compatible with 3.3-V or 5.0-V systems.

Table 2–36 summarizes Stratix MultiVolt I/O support.

Table 2–36. Stratix MultiVolt I/O Support Note (1)

V_{CCIO} (V)	Input Signal (5)					Output Signal (6)				
	1.5 V	1.8 V	2.5 V	3.3 V	5.0 V	1.5 V	1.8 V	2.5 V	3.3 V	5.0 V
1.5	v	v	v (2)	v (2)		v				
1.8	v (2)	v	v (2)	v (2)		v (3)	v			
2.5			v	v		v (3)	v (3)	v		
3.3			v (2)	v	v (4)	v (3)	v (3)	v (3)	v	v

Notes to Table 2–36:

(1) To drive inputs higher than V_{CCIO} but less than 4.1 V, disable the PCI clamping diode. However, to drive 5.0-V inputs to the device, enable the PCI clamping diode to prevent V_I from rising above 4.0 V.

(2) The input pin current may be slightly higher than the typical value.

(3) Although V_{CCIO} specifies the voltage necessary for the Stratix device to drive out, a receiving device powered at a different level can still interface with the Stratix device if it has inputs that tolerate the V_{CCIO} value.

(4) Stratix devices can be 5.0-V tolerant with the use of an external resistor and the internal PCI clamp diode.

(5) This is the external signal that is driving the Stratix device.

(6) This represents the system voltage that Stratix supports when a VCCIO pin is connected to a specific voltage level. For example, when VCCIO is 3.3 V and if the I/O standard is LVTTL/LVCMOS, the output high of the signal coming out from Stratix is 3.3 V and is compatible with 3.3-V or 5.0-V systems.

High-Speed Differential I/O Support

Stratix devices contain dedicated circuitry for supporting differential standards at speeds up to 840 Mbps. The following differential I/O standards are supported in the Stratix device: LVDS, LVPECL, HyperTransport, and 3.3-V PCML.

There are four dedicated high-speed PLLs in the EP1S10 to EP1S25 devices and eight dedicated high-speed PLLs in the EP1S30 to EP1S80 devices to multiply reference clocks and drive high-speed differential SERDES channels.

f See the Stratix device pin-outs at **www.altera.com** for additional high speed DIFFIO pin information for Stratix devices.

Table 2–37 shows the number of channels that each fast PLL can clock in EP1S10, EP1S20, and EP1S25 devices. Tables 2–38 through Table 2–41 show this information for EP1S30, EP1S40, EP1S60, and EP1S80 devices.

Table 2–37. EP1S10, EP1S20 & EP1S25 Device Differential Channels (Part 1 of 2) *Note (1)*

Device	Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs			
					PLL 1	PLL 2	PLL 3	PLL 4
EP1S10	484-pin FineLine BGA	Transmitter *(2)*	20	840 *(4)*	5	5	5	5
				840 *(3)*	10	10	10	10
		Receiver	20	840 *(4)*	5	5	5	5
				840 *(3)*	10	10	10	10
	672-pin FineLine BGA 672-pin BGA	Transmitter *(2)*	36	624 *(4)*	9	9	9	9
				624 *(3)*	18	18	18	18
		Receiver	36	624 *(4)*	9	9	9	9
				624 *(3)*	18	18	18	18
	780-pin FineLine BGA	Transmitter *(2)*	44	840 *(4)*	11	11	11	11
				840 *(3)*	22	22	22	22
		Receiver	44	840 *(4)*	11	11	11	11
				840 *(3)*	22	22	22	22
EP1S20	484-pin FineLine BGA	Transmitter *(2)*	24	840 *(4)*	6	6	6	6
				840 *(3)*	12	12	12	12
		Receiver	20	840 *(4)*	5	5	5	5
				840 *(3)*	10	10	10	10
	672-pin FineLine BGA 672-pin BGA	Transmitter *(2)*	48	624 *(4)*	12	12	12	12
				624 *(3)*	24	24	24	24
		Receiver	50	624 *(4)*	13	12	12	13
				624 *(3)*	25	25	25	25
	780-pin FineLine BGA	Transmitter *(2)*	66	840 *(4)*	17	16	16	17
				840 *(3)*	33	33	33	33
		Receiver	66	840 *(4)*	17	16	16	17
				840 *(3)*	33	33	33	33

Table 2–37. EP1S10, EP1S20 & EP1S25 Device Differential Channels (Part 2 of 2) *Note (1)*

Device	Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs			
					PLL 1	PLL 2	PLL 3	PLL 4
EP1S25	672-pin FineLine BGA 672-pin BGA	Transmitter *(2)*	56	624 *(4)*	14	14	14	14
				624 *(3)*	28	28	28	28
		Receiver	58	624 *(4)*	14	15	15	14
				624 *(3)*	29	29	29	29
	780-pin FineLine BGA	Transmitter *(2)*	70	840 *(4)*	18	17	17	18
				840 *(3)*	35	35	35	35
		Receiver	66	840 *(4)*	17	16	16	17
				840 *(3)*	33	33	33	33
	1,020-pin FineLine BGA	Transmitter *(2)*	78	840 *(4)*	19	20	20	19
				840 *(3)*	39	39	39	39
		Receiver	78	840 *(4)*	19	20	20	19
				840 *(3)*	39	39	39	39

Notes to Table 2–37:

(1) The first row for each transmitter or receiver reports the number of channels driven directly by the PLL. The second row below it shows the maximum channels a PLL can drive if cross bank channels are used from the adjacent center PLL. For example, in the 484-pin FineLine BGA EP1S10 device, PLL 1 can drive a maximum of five channels at 840 Mbps or a maximum of 10 channels at 840 Mbps. The Quartus II software may also merge receiver and transmitter PLLs when a receiver is driving a transmitter. In this case, one fast PLL can drive both the maximum numbers of receiver and transmitter channels.

(2) The number of channels listed includes the transmitter clock output (`tx_outclock`) channel. If the design requires a DDR clock, it can use an extra data channel.

(3) These channels span across two I/O banks per side of the device. When a center PLL clocks channels in the opposite bank on the same side of the device it is called cross-bank PLL support. Both center PLLs can clock cross-bank channels simultaneously if, for example, PLL_1 is clocking all receiver channels and PLL_2 is clocking all transmitter channels. You cannot have two adjacent PLLs simultaneously clocking cross-bank receiver channels or two adjacent PLLs simultaneously clocking transmitter channels. Cross-bank allows for all receiver channels on one side of the device to be clocked on one clock while all transmitter channels on the device are clocked on the other center PLL. Crossbank PLLs are supported at full-speed, 840 Mbps. For wire-bond devices, the full-speed is 624 Mbps.

(4) These values show the channels available for each PLL without crossing another bank.

When you span two I/O banks using cross-bank support, you can route only two load enable signals total between the PLLs. When you enable `rx_data_align`, you use both `rxloadena` and `txloadena` of a PLL. That leaves no `loadena` for the second PLL.

The only way you can use the `rx_data_align` is if one of the following is true:

- The receiver PLL is only clocking receive channels (no resources for the transmitter)
- If all channels can fit in one I/O bank

Table 2–38. EP1S30 Differential Channels *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2), (3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
780-pin FineLine BGA	Transmitter *(4)*	70	840	18	17	17	18	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	35	35	35	35	*(6)*	*(6)*	*(6)*	*(6)*
	Receiver	66	840	17	16	16	17	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	33	33	33	33	*(6)*	*(6)*	*(6)*	*(6)*
956-pin BGA	Transmitter *(4)*	80	840	19	20	20	19	20	20	20	20
			840 *(5)*	39	39	39	39	20	20	20	20
	Receiver	80	840	20	20	20	20	19	20	20	19
			840 *(5)*	40	40	40	40	19	20	20	19
1,020-pin FineLine BGA	Transmitter *(4)*	80 (2) *(7)*	840	19 (1)	20	20	19 (1)	20	20	20	20
			840 *(5)*,*(8)*	39 (1)	39 (1)	39 (1)	39 (1)	20	20	20	20
	Receiver	80 (2) *(7)*	840	20	20	20	20	19 (1)	20	20	19 (1)
			840 *(5)*,*(8)*	40	40	40	40	19 (1)	20	20	19 (1)

Table 2–39. EP1S40 Differential Channels (Part 1 of 2) *Note (1)*

Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2), (3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
780-pin FineLine BGA	Transmitter *(4)*	68	840	18	16	16	18	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	34	34	34	34	*(6)*	*(6)*	*(6)*	*(6)*
	Receiver	66	840	17	16	16	17	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	33	33	33	33	*(6)*	*(6)*	*(6)*	*(6)*

Table 2–39. EP1S40 Differential Channels (Part 2 of 2) *Note (1)*

Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
956-pin BGA	Transmitter *(4)*	80	840	18	17	17	18	20	20	20	20
			840 *(5)*	35	35	35	35	20	20	20	20
	Receiver	80	840	20	20	20	20	18	17	17	18
			840 *(5)*	40	40	40	40	18	17	17	18
1,020-pin FineLine BGA	Transmitter *(4)*	80 (10) *(7)*	840	18 (2)	17 (3)	17 (3)	18 (2)	20	20	20	20
			840 *(5)*, *(8)*	35 (5)	35 (5)	35 (5)	35 (5)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	18 (2)	17 (3)	17 (3)	18 (2)
			840 *(5)*, *(8)*	40	40	40	40	18 (2)	17 (3)	17 (3)	18 (2)
1,508-pin FineLine BGA	Transmitter *(4)*	80 (10) *(7)*	840	18 (2)	17 (3)	17 (3)	18 (2)	20	20	20	20
			840 *(5)*, *(8)*	35 (5)	35 (5)	35 (5)	35 (5)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	18 (2)	17 (3)	17 (3)	18 (2)
			840 *(5)*, *(8)*	40	40	40	40	18 (2)	17 (3)	17 (3)	18 (2)

Table 2–40. EP1S60 Differential Channels (Part 1 of 2) *Note (1)*

Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
956-pin BGA	Transmitter *(4)*	80	840	12	10	10	12	20	20	20	20
			840 *(5)*, *(8)*	22	22	22	22	20	20	20	20
	Receiver	80	840	20	20	20	20	12	10	10	12
			840 *(5)*, *(8)*	40	40	40	40	12	10	10	12

Table 2–40. EP1S60 Differential Channels (Part 2 of 2) *Note (1)*

Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
1,020-pin FineLine BGA	Transmitter *(4)*	80 (12) *(7)*	840	12 (2)	10 (4)	10 (4)	12 (2)	20	20	20	20
			840 *(5)*, *(8)*	22 (6)	22 (6)	22 (6)	22 (6)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	12 (8)	10 (10)	10 (10)	12 (8)
			840 *(5)*, *(8)*	40	40	40	40	12 (8)	10 (10)	10 (10)	12 (8)
1,508-pin FineLine BGA	Transmitter *(4)*	80 (36) *(7)*	840	12 (8)	10 (10)	10 (10)	12 (8)	20	20	20	20
			840 *(5)*,*(8)*	22 (18)	22 (18)	22 (18)	22 (18)	20	20	20	20
	Receiver	80 (36) *(7)*	840	20	20	20	20	12 (8)	10 (10)	10 (10)	12 (8)
			840 *(5)*,*(8)*	40	40	40	40	12 (8)	10 (10)	10 (10)	12 (8)

Table 2–41. EP1S80 Differential Channels (Part 1 of 2) *Note (1)*

Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
956-pin BGA	Transmitter *(4)*	80 (40) *(7)*	840	10	10	10	10	20	20	20	20
			840 *(5)*,*(8)*	20	20	20	20	20	20	20	20
	Receiver	80	840	20	20	20	20	10	10	10	10
			840 *(5)*,*(8)*	40	40	40	40	10	10	10	10
1,020-pin FineLine BGA	Transmitter *(4)*	92 (12) *(7)*	840	10 (2)	10 (4)	10 (4)	10 (2)	20	20	20	20
			840 *(5)*,*(8)*	20 (6)	20 (6)	20 (6)	20 (6)	20	20	20	20
	Receiver	90 (10) *(7)*	840	20	20	20	20	10 (2)	10 (3)	10 (3)	10 (2)
			840 *(5)*,*(8)*	40	40	40	40	10 (2)	10 (3)	10 (3)	10 (2)

Table 2–41. EP1S80 Differential Channels (Part 2 of 2) Note (1)

Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs (2), (3)			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
1,508-pin FineLine BGA	Transmitter (4)	80 (72) (7)	840	10 (10)	10 (10)	10 (10)	10 (10)	20 (8)	20 (8)	20 (8)	20 (8)
			840 (5),(8)	20 (20)	20 (20)	20 (20)	20 (20)	20 (8)	20 (8)	20 (8)	20 (8)
	Receiver	80 (56) (7)	840	20	20	20	20	10 (14)	10 (14)	10 (14)	10 (14)
			840 (5),(8)	40	40	40	40	10 (14)	10 (14)	10 (14)	10 (14)

Notes to Tables 2–38 through 2–41:

(1) The first row for each transmitter or receiver reports the number of channels driven directly by the PLL. The second row below it shows the maximum channels a PLL can drive if cross bank channels are used from the adjacent center PLL. For example, in the 780-pin FineLine BGA EP1S30 device, PLL 1 can drive a maximum of 18 transmitter channels at 840 Mbps or a maximum of 35 transmitter channels at 840 Mbps. The Quartus II software may also merge transmitter and receiver PLLs when a receiver is driving a transmitter. In this case, one fast PLL can drive both the maximum numbers of receiver and transmitter channels.

(2) Some of the channels accessible by the center fast PLL and the channels accessible by the corner fast PLL overlap. Therefore, the total number of channels is not the addition of the number of channels accessible by PLLs 1, 2, 3, and 4 with the number of channels accessible by PLLs 7, 8, 9, and 10. For more information on which channels overlap, see the Stratix device pin-outs at **www.altera.com**.

(3) The corner fast PLLs in this device support a data rate of 840 Mbps for channels labeled "high" speed in the device pin-outs at **www.altera.com**.

(4) The numbers of channels listed include the transmitter clock output (`tx_outclock`) channel. An extra data channel can be used if a DDR clock is needed.

(5) These channels span across two I/O banks per side of the device. When a center PLL clocks channels in the opposite bank on the same side of the device it is called cross-bank PLL support. Both center PLLs can clock cross-bank channels simultaneously if say PLL_1 is clocking all receiver channels and PLL_2 is clocking all transmitter channels. You cannot have two adjacent PLLs simultaneously clocking cross-bank receiver channels or two adjacent PLLs simultaneously clocking transmitter channels. Cross-bank allows for all receiver channels on one side of the device to be clocked on one clock while all transmitter channels on the device are clocked on the other center PLL. Crossbank PLLs are supported at full-speed, 840 Mbps. For wire-bond devices, the full-speed is 624 Mbps.

(6) PLLs 7, 8, 9, and 10 are not available in this device.

(7) The number in parentheses is the number of slow-speed channels, guaranteed to operate at up to 462 Mbps. These channels are independent of the high-speed differential channels. For the location of these channels, see the device pin-outs at **www.altera.com**.

(8) See the Stratix device pin-outs at **www.altera.com**. Channels marked "high" speed are 840 MBps and "low" speed channels are 462 MBps.

The high-speed differential I/O circuitry supports the following high speed I/O interconnect standards and applications:

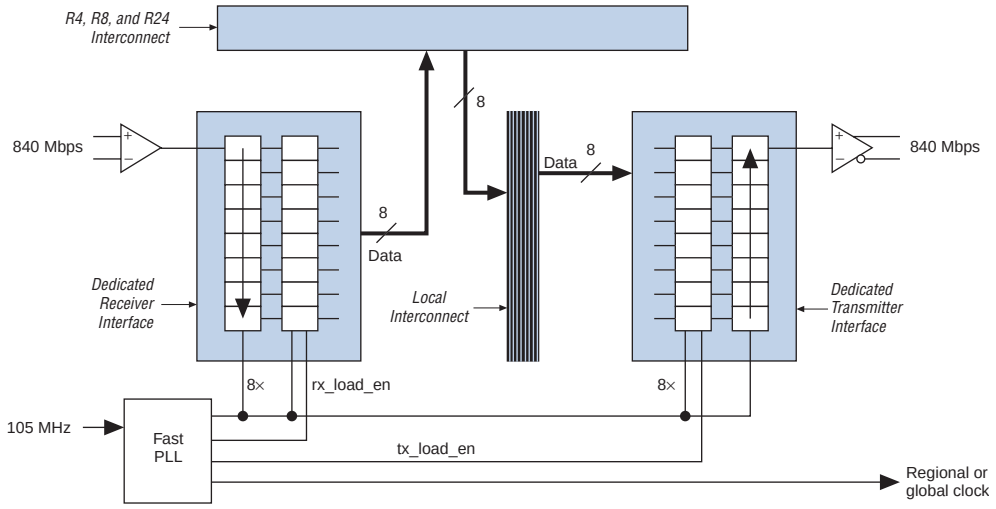
- UTOPIA IV
- SPI-4 Phase 2 (POS-PHY Level 4)
- SFI-4
- 10G Ethernet XSBI

- RapidIO
- HyperTransport

Dedicated Circuitry

Stratix devices support source-synchronous interfacing with LVDS, LVPECL, 3.3-V PCML, or HyperTransport signaling at up to 840 Mbps. Stratix devices can transmit or receive serial channels along with a low-speed or high-speed clock. The receiving device PLL multiplies the clock by a integer factor W (W = 1 through 32). For example, a HyperTransport application where the data rate is 800 Mbps and the clock rate is 400 MHz would require that W be set to 2. The SERDES factor J determines the parallel data width to deserialize from receivers or to serialize for transmitters. The SERDES factor J can be set to 4, 7, 8, or 10 and does not have to equal the PLL clock-multiplication W value. For a J factor of 1, the Stratix device bypasses the SERDES block. For a J factor of 2, the Stratix device bypasses the SERDES block, and the DDR input and output registers are used in the IOE. See Figure 2–73.

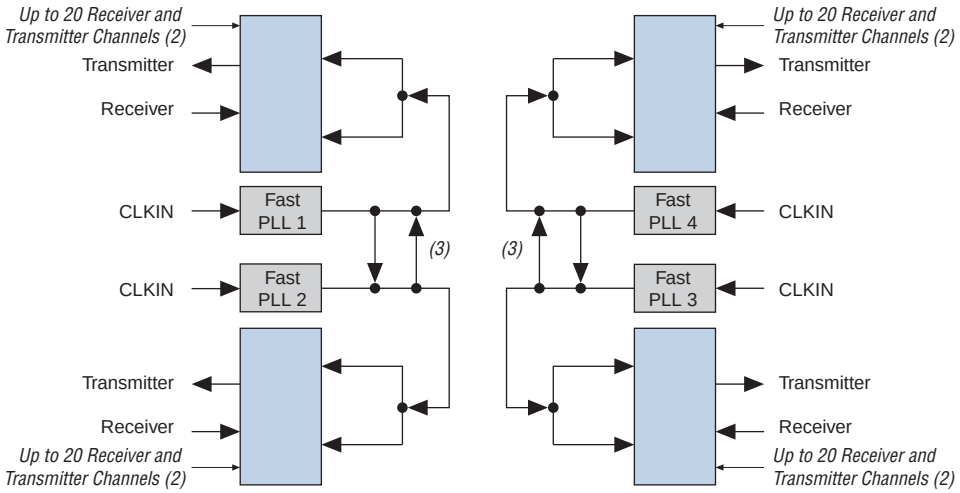
Figure 2–73. High-Speed Differential I/O Receiver / Transmitter Interface Example



An external pin or global or regional clock can drive the fast PLLs, which can output up to three clocks: two multiplied high-speed differential I/O clocks to drive the SERDES block and/or external pin, and a low-speed clock to drive the logic array.

The Quartus II MegaWizard® Plug-In Manager only allows the implementation of up to 20 receiver or 20 transmitter channels for each fast PLL. These channels operate at up to 840 Mbps. The receiver and transmitter channels are interleaved such that each I/O bank on the left and right side of the device has one receiver channel and one transmitter channel per LAB row. Figure 2–74 shows the fast PLL and channel layout in EP1S10, EP1S20, and EP1S25 devices. Figure 2–75 shows the fast PLL and channel layout in the EP1S30 to EP1S80 devices.

Figure 2–74. Fast PLL & Channel Layout in the EP1S10, EP1S20 or EP1S25 Devices *Note (1)*



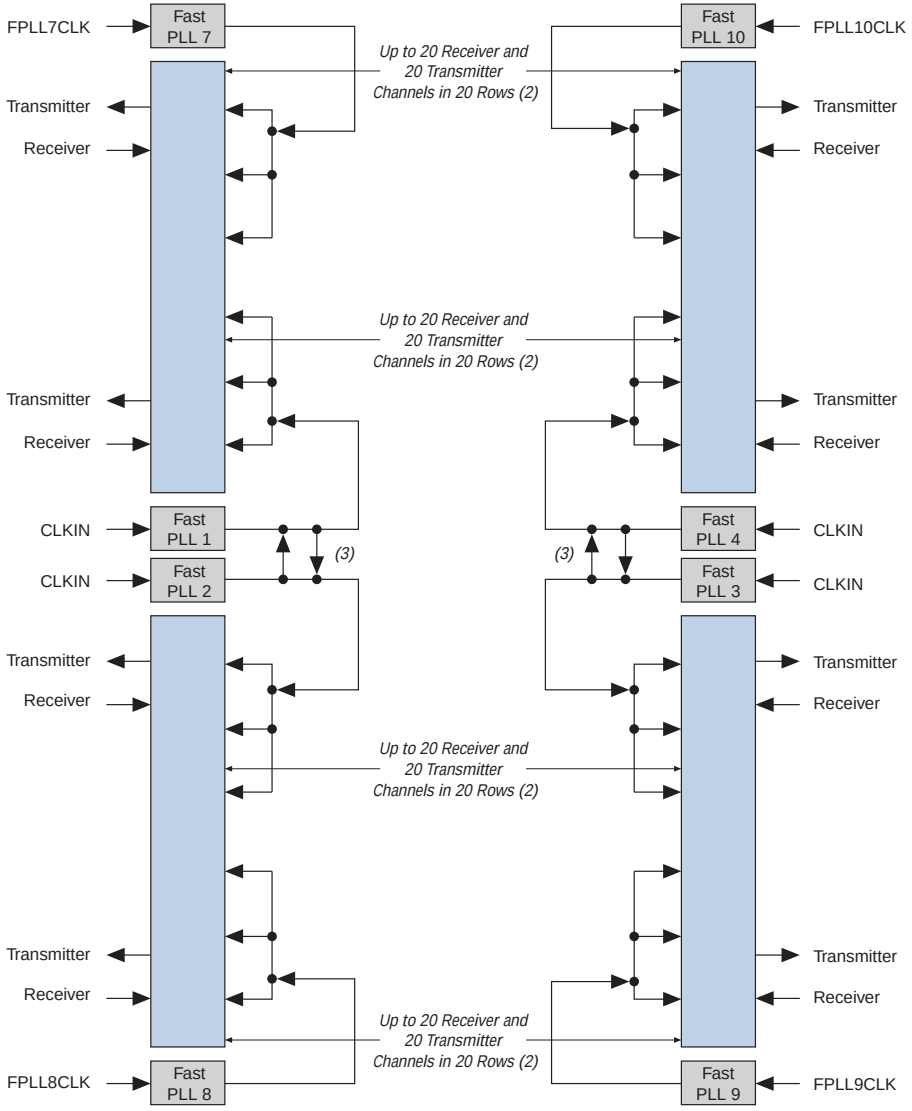
Notes to Figure 2–74:

(1) Wire-bond packages support up to 624 Mbps.

(2) See Table 2–41 for the number of channels each device supports.

(3) There is a multiplexer here to select the PLL clock source. If a PLL uses this multiplexer to clock channels outside of its bank quadrant, those clocked channels support up to 840 Mbps for "high" speed channels and 462 Mbps for "low" speed channels, as labeled in the device pin-outs at **www.altera.com**.

Figure 2–75. Fast PLL & Channel Layout in the EP1S30 to EP1S80 Devices *Note (1)*



Notes to Figure 2–75:

(1) Wire-bond packages support up to 624 Mbps.

(2) See Table 2–38 through 2–41 for the number of channels each device supports.

(3) There is a multiplexer here to select the PLL clock source. If a PLL uses this multiplexer to clock channels outside of its bank quadrant, those clocked channels support up to 840 Mbps for "high" speed channels and 462 Mbps for "low" speed channels as labeled in the device pin-outs at **www.altera.com**.

The transmitter external clock output is transmitted on a data channel. The `txclk` pin for each bank is located in between data transmitter pins. For ×1 clocks (e.g., 622 Mbps, 622 MHz), the high-speed PLL clock bypasses the SERDES to drive the output pins. For half-rate clocks (e.g., 622 Mbps, 311 MHz) or any other even-numbered factor such as 1/4, 1/7, 1/8, or 1/10, the SERDES automatically generates the clock in the Quartus II software.

For systems that require more than four or eight high-speed differential I/O clock domains, a SERDES bypass implementation is possible using IOEs.

Byte Alignment

For high-speed source synchronous interfaces such as POS-PHY 4, XSBI, RapidIO, and HyperTransport technology, the source synchronous clock rate is not a byte- or SERDES-rate multiple of the data rate. Byte alignment is necessary for these protocols since the source synchronous clock does not provide a byte or word boundary since the clock is one half the data rate, not one eighth. The Stratix device's high-speed differential I/O circuitry provides dedicated data realignment circuitry for user-controlled byte boundary shifting. This simplifies designs while saving LE resources. An input signal to each fast PLL can stall deserializer parallel data outputs by one bit period. You can use an LE-based state machine to signal the shift of receiver byte boundaries until a specified pattern is detected to indicate byte alignment.

Power Sequencing & Hot Socketing

Because Stratix devices can be used in a mixed-voltage environment, they have been designed specifically to tolerate any possible power-up sequence. Therefore, the `VCCIO` and `VCCINT` power supplies may be powered in any order.

Although you can power up or down the `VCCIO` and `VCCINT` power supplies in any sequence, you should not power down any I/O banks that contain configuration pins while leaving other I/O banks powered on. For power up and power down, all supplies (`VCCINT` and all `VCCIO` power planes) must be powered up and down within 100 ms of each other. This prevents I/O pins from driving out.

Signals can be driven into Stratix devices before and during power up without damaging the device. In addition, Stratix devices do not drive out during power up. Once operating conditions are reached and the device is configured, Stratix devices operate as specified by the user. For more information, see *Hot Socketing* in the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook, Volume 2*.

3. Configuration & Testing

S51003-1.3

IEEE Std. 1149.1 (JTAG) Boundary-Scan Support

All Stratix® devices provide JTAG BST circuitry that complies with the IEEE Std. 1149.1a-1990 specification. JTAG boundary-scan testing can be performed either before or after, but not during configuration. Stratix devices can also use the JTAG port for configuration together with either the Quartus® II software or hardware using either Jam Files (**.jam**) or Jam Byte-Code Files (**.jbc**).

Stratix devices support IOE I/O standard setting reconfiguration through the JTAG BST chain. The JTAG chain can update the I/O standard for all input and output pins any time before or during user mode through the `CONFIG_IO` instruction. You can use this ability for JTAG testing before configuration when some of the Stratix pins drive or receive from other devices on the board using voltage-referenced standards. Since the Stratix device may not be configured before JTAG testing, the I/O pins may not be configured for appropriate electrical standards for chip-to-chip communication. Programming those I/O standards via JTAG allows you to fully test the I/O connection to other devices.

The enhanced PLL reconfiguration bits are part of the JTAG chain before configuration and after power-up. After device configuration, the PLL reconfiguration bits are not part of the JTAG chain.

The JTAG pins support 1.5-V/1.8-V or 2.5-V/3.3-V I/O standards. The `TDO` pin voltage is determined by the V_{CCIO} of the bank where it resides. The `VCCSEL` pin selects whether the JTAG inputs are 1.5-V, 1.8-V, 2.5-V, or 3.3-V compatible.

Stratix devices also use the JTAG port to monitor the logic operation of the device with the SignalTap® II embedded logic analyzer. Stratix devices support the JTAG instructions shown in Table 3–1.

The Quartus II software has an Auto Usercode feature where you can choose to use the checksum value of a programming file as the JTAG user code. If selected, the checksum is automatically loaded to the `USERCODE` register. In the Settings dialog box in the Assignments menu, click **Device & Pin Options**, then **General,** and then turn on the **Auto Usercode** option.

Table 3–1. Stratix JTAG Instructions

JTAG Instruction	Instruction Code	Description
SAMPLE/PRELOAD	00 0000 0101	Allows a snapshot of signals at the device pins to be captured and examined during normal device operation, and permits an initial data pattern to be output at the device pins. Also used by the SignalTap II embedded logic analyzer.
EXTEST *(1)*	00 0000 0000	Allows the external circuitry and board-level interconnects to be tested by forcing a test pattern at the output pins and capturing test results at the input pins.
BYPASS	11 1111 1111	Places the 1-bit bypass register between the `TDI` and `TDO` pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation.
USERCODE	00 0000 0111	Selects the 32-bit `USERCODE` register and places it between the `TDI` and `TDO` pins, allowing the `USERCODE` to be serially shifted out of `TDO`.
IDCODE	00 0000 0110	Selects the `IDCODE` register and places it between `TDI` and `TDO`, allowing the `IDCODE` to be serially shifted out of `TDO`.
`HIGHZ` *(1)*	00 0000 1011	Places the 1-bit bypass register between the `TDI` and `TDO` pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation, while tri-stating all of the I/O pins.
`CLAMP` *(1)*	00 0000 1010	Places the 1-bit bypass register between the `TDI` and `TDO` pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation while holding I/O pins to a state defined by the data in the boundary-scan register.
ICR instructions		Used when configuring an Stratix device via the JTAG port with a MasterBlaster™, ByteBlasterMV™, or ByteBlaster™ II download cable, or when using a Jam File or Jam Byte-Code File via an embedded processor or JRunner.
PULSE_NCONFIG	00 0000 0001	Emulates pulsing the `nCONFIG` pin low to trigger reconfiguration even though the physical pin is unaffected.
CONFIG_IO	00 0000 1101	Allows configuration of I/O standards through the JTAG chain for JTAG testing. Can be executed before, after, or during configuration. Stops configuration if executed during configuration. Once issued, the `CONFIG_IO` instruction will hold `nSTATUS` low to reset the configuration device. `nSTATUS` is held low until the device is reconfigured.
SignalTap II instructions		Monitors internal device operation with the SignalTap II embedded logic analyzer.

Note to Table 3–1:

(1) Bus hold and weak pull-up resistor features override the high-impedance state of `HIGHZ`, `CLAMP`, and `EXTEST`.

The Stratix device instruction register length is 10 bits and the USERCODE register length is 32 bits. Tables 3–2 and 3–3 show the boundary-scan register length and device IDCODE information for Stratix devices.

Table 3–2. Stratix Boundary-Scan Register Length

Device	Boundary-Scan Register Length
EP1S10	1,317
EP1S20	1,797
EP1S25	2,157
EP1S30	2,253
EP1S40	2,529
EP1S60	3,129
EP1S80	3,777

Table 3–3. 32-Bit Stratix Device IDCODE

Device	IDCODE (32 Bits) *(1)*			
	Version (4 Bits)	Part Number (16 Bits)	Manufacturer Identity (11 Bits)	LSB (1 Bit) *(2)*
EP1S10	0000	0010 0000 0000 0001	000 0110 1110	1
EP1S20	0000	0010 0000 0000 0010	000 0110 1110	1
EP1S25	0000	0010 0000 0000 0011	000 0110 1110	1
EP1S30	0000	0010 0000 0000 0100	000 0110 1110	1
EP1S40	0000	0010 0000 0000 0101	000 0110 1110	1
EP1S60	0000	0010 0000 0000 0110	000 0110 1110	1
EP1S80	0000	0010 0000 0000 0111	000 0110 1110	1

Notes to Tables 3–2 and 3–3:

(1) The most significant bit (MSB) is on the left.

(2) The IDCODE's least significant bit (LSB) is always 1.

Figure 3–1 shows the timing requirements for the JTAG signals.

Figure 3–1. Stratix JTAG Waveforms

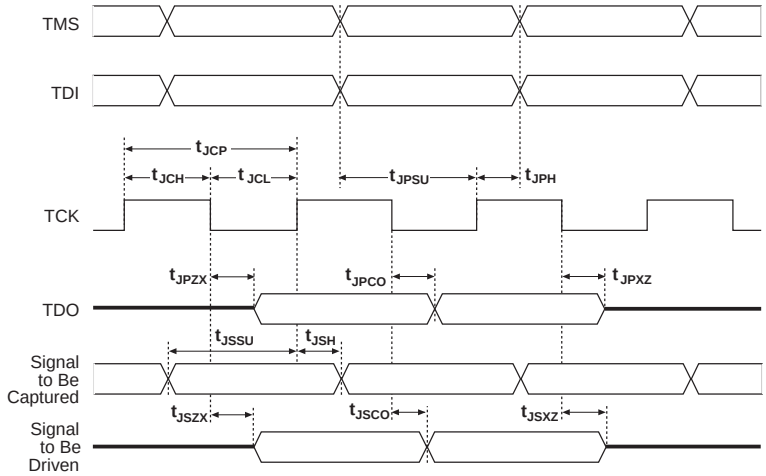


Table 3–4 shows the JTAG timing parameters and values for Stratix devices.

Table 3–4. Stratix JTAG Timing Parameters & Values

Symbol	Parameter	Min	Max	Unit
t_{JCP}	TCK clock period	100		ns
t_{JCH}	TCK clock high time	50		ns
t_{JCL}	TCK clock low time	50		ns
t_{JPSU}	JTAG port setup time	20		ns
t_{JPH}	JTAG port hold time	45		ns
t_{JPCO}	JTAG port clock to output		25	ns
t_{JPZX}	JTAG port high impedance to valid output		25	ns
t_{JPXZ}	JTAG port valid output to high impedance		25	ns
t_{JSSU}	Capture register setup time	20		ns
t_{JSH}	Capture register hold time	45		ns
t_{JSCO}	Update register clock to output		35	ns
t_{JSZX}	Update register high impedance to valid output		35	ns
t_{JSXZ}	Update register valid output to high impedance		35	ns

1 Stratix, Stratix II, Cyclone®, and Cyclone II devices must be within the first 17 devices in a JTAG chain. All of these devices have the same JTAG controller. If any of the Stratix, Stratix II, Cyclone, and Cyclone II devices are in the 18th or after they will fail configuration. This does not affect SignalTap II.

f For more information on JTAG, see the following documents:

- *AN 39: IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*
- *Jam Programming & Test Language Specification*

SignalTap II Embedded Logic Analyzer

Stratix devices feature the SignalTap II embedded logic analyzer, which monitors design operation over a period of time through the IEEE Std. 1149.1 (JTAG) circuitry. You can analyze internal logic at speed without bringing internal signals to the I/O pins. This feature is particularly important for advanced packages, such as FineLine BGA® packages, because it can be difficult to add a connection to a pin during the debugging process after a board is designed and manufactured.

Configuration

The logic, circuitry, and interconnects in the Stratix architecture are configured with CMOS SRAM elements. Altera® devices are reconfigurable. Because every device is tested with a high-coverage production test program, you do not have to perform fault testing and can focus on simulation and design verification.

Stratix devices are configured at system power-up with data stored in an Altera serial configuration device or provided by a system controller. Altera offers in-system programmability (ISP)-capable configuration devices that configure Stratix devices via a serial data stream. Stratix devices can be configured in under 100 ms using 8-bit parallel data at 100 MHz. The Stratix device's optimized interface allows microprocessors to configure it serially or in parallel, and synchronously or asynchronously. The interface also enables microprocessors to treat Stratix devices as memory and configure them by writing to a virtual memory location, making reconfiguration easy. After a Stratix device has been configured, it can be reconfigured in-circuit by resetting the device and loading new data. Real-time changes can be made during system operation, enabling innovative reconfigurable computing applications.

Operating Modes

The Stratix architecture uses SRAM configuration elements that require configuration data to be loaded each time the circuit powers up. The process of physically loading the SRAM data into the device is called configuration. During initialization, which occurs immediately after

configuration, the device resets registers, enables I/O pins, and begins to operate as a logic device. The I/O pins are tri-stated during power-up, and before and during configuration. Together, the configuration and initialization processes are called command mode. Normal device operation is called user mode.

SRAM configuration elements allow Stratix devices to be reconfigured in-circuit by loading new configuration data into the device. With real-time reconfiguration, the device is forced into command mode with a device pin. The configuration process loads different configuration data, reinitializes the device, and resumes user-mode operation. You can perform in-field upgrades by distributing new configuration files either within the system or remotely.

PORSEL is a dedicated input pin used to select POR delay times of 2 ms or 100 ms during power-up. When the PORSEL pin is connected to ground, the POR time is 100 ms; when the PORSEL pin is connected to V_{CC}, the POR time is 2 ms.

The nIO_PULLUP pin enables a built-in weak pull-up resistor to pull all user I/O pins to V_{CCIO} before and during device configuration. If nIO_PULLUP is connected to V_{CC} during configuration, the weak pull-ups on all user I/O pins are disabled. If connected to ground, the pull-ups are enabled during configuration. The nIO_PULLUP pin can be pulled to 1.5, 1.8, 2.5, or 3.3 V for a logic level high.

VCCSEL is a dedicated input that is used to choose whether all dedicated configuration and JTAG input pins can accept 1.5 V/1.8 V or 2.5 V/3.3 V during configuration. A logic low sets 3.3 V/2.5 V, and a logic high sets 1.8 V/1.5 V. VCCSEL affects the following pins: TDI, TMS, TCK, TRST, MSEL0, MSEL1, MSEL2, nCONFIG, nCE, DCLK, PLL_ENA, CONF_DONE, nSTATUS. The VCCSEL pin can be pulled to 1.5, 1.8, 2.5, or 3.3 V for a logic level high.

The VCCSEL signal does not control the dual-purpose configuration pins such as the DATA[7..0] and PPA pins (nWS, nRS, CS, nCS, and RDYnBSY). During configuration, these dual-purpose pins will drive out voltage levels corresponding to the V_{CCIO} supply voltage that powers the I/O bank containing the pin. After configuration, the dual-purpose pins use I/O standards specified in the user design.

TDO and nCEO drive out at the same voltages as the V_{CCIO} supply that powers the I/O bank containing the pin. Users must select the V_{CCIO} supply for bank containing TDO accordingly. For example, when using the ByteBlaster™ MV cable, the V_{CCIO} for the bank containing TDO must be powered up at 3.3 V.

Configuring Stratix FPGAs with JRunner

JRunner is a software driver that configures Altera FPGAs, including Stratix FPGAs, through the ByteBlaster II or ByteBlasterMV cables in JTAG mode. The programming input file supported is in Raw Binary File (**.rbf**) format. JRunner also requires a Chain Description File (**.cdf**) generated by the Quartus II software. JRunner is targeted for embedded JTAG configuration. The source code is developed for the Windows NT operating system (OS), but can be customized to run on other platforms. For more information on the JRunner software driver, see the JRunner Software Driver: An Embedded Solution to the JTAG Configuration White Paper and the source files on the Altera web site (**www.altera.com**).

Configuration Schemes

You can load the configuration data for a Stratix device with one of five configuration schemes (see Table 3–5), chosen on the basis of the target application. You can use a configuration device, intelligent controller, or the JTAG port to configure a Stratix device. A configuration device can automatically configure a Stratix device at system power-up.

Multiple Stratix devices can be configured in any of five configuration schemes by connecting the configuration enable (nCE) and configuration enable output (nCEO) pins on each device.

Table 3–5. Data Sources for Configuration

Configuration Scheme	Data Source
Configuration device	Enhanced or EPC2 configuration device
Passive serial (PS)	MasterBlaster, ByteBlasterMV, or ByteBlaster II download cable or serial data source
Passive parallel asynchronous (PPA)	Parallel data source
Fast passive parallel	Parallel data source
JTAG	MasterBlaster, ByteBlasterMV, or ByteBlaster II download cable, a microprocessor with a Jam or JBC file, or JRunner

Partial Reconfiguration

The enhanced PLLs within the Stratix device family support partial reconfiguration of their multiply, divide, and time delay settings without reconfiguring the entire device. You can use either serial data from the logic array or regular I/O pins to program the PLL's counter settings in a serial chain. This option provides considerable flexibility for frequency

synthesis, allowing real-time variation of the PLL frequency and delay. The rest of the device is functional while reconfiguring the PLL. See the *Stratix Architecture* chapter of the *Stratix Device Handbook, Volume 1* for more information on Stratix PLLs.

Remote Update Configuration Modes

Stratix devices also support remote configuration using an Altera enhanced configuration device (e.g., EPC16, EPC8, and EPC4 devices) with page mode selection. Factory configuration data is stored in the default page of the configuration device. This is the default configuration that contains the design required to control remote updates and handle or recover from errors. You write the factory configuration once into the flash memory or configuration device. Remote update data can update any of the remaining pages of the configuration device. If there is an error or corruption in a remote update configuration, the configuration device reverts back to the factory configuration information.

There are two remote configuration modes: remote and local configuration. You can use the remote update configuration mode for all three configuration modes: serial, parallel synchronous, and parallel asynchronous. Configuration devices (for example, EPC16 devices) only support serial and parallel synchronous modes. Asynchronous parallel mode allows remote updates when an intelligent host is used to configure the Stratix device. This host must support page mode settings similar to an EPC16 device.

Remote Update Mode

When the Stratix device is first powered up in remote update programming mode, it loads the configuration located at page address "`000`." The factory configuration should always be located at page address "`000`," and should never be remotely updated. The factory configuration contains the required logic to perform the following operations:

- Determine the page address / load location for the next application's configuration data
- Recover from a previous configuration error
- Receive new configuration data and write it into the configuration device

The factory configuration is the default and takes control if an error occurs while loading the application configuration.

While in the factory configuration, the factory-configuration logic performs the following operations:

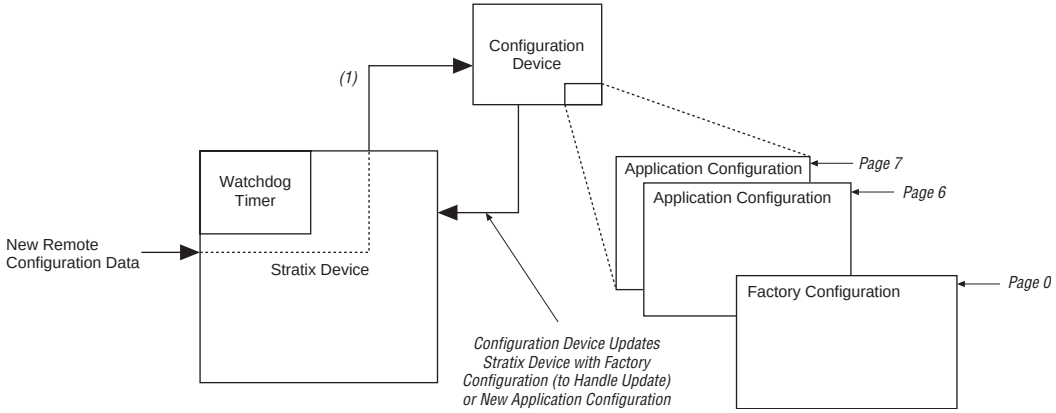
- Loads a remote update-control register to determine the page address of the new application configuration
- Determines whether to enable a user watchdog timer for the application configuration
- Determines what the watchdog timer setting should be if it is enabled

The user watchdog timer is a counter that must be continually reset within a specific amount of time in the user mode of an application configuration to ensure that valid configuration occurred during a remote update. Only valid application configurations designed for remote update can reset the user watchdog timer in user mode. If a valid application configuration does not reset the user watchdog timer in a specific amount of time, the timer updates a status register and loads the factory configuration. The user watchdog timer is automatically disabled for factory configurations.

If an error occurs in loading the application configuration, the configuration logic writes a status register to specify the cause of the error. Once this occurs, the Stratix device automatically loads the factory configuration, which reads the status register and determines the reason for reconfiguration. Based on the reason, the factory configuration will take appropriate steps and will write the remote update control register to specify the next application configuration page to be loaded.

When the Stratix device successfully loads the application configuration, it enters into user mode. The Stratix device then executes the main application of the user. Intellectual property (IP), such as a Nios® (16-bit ISA) and Nios® II (32-bit ISA) embedded processors, can help the Stratix device determine when remote update is coming. The Nios embedded processor or user logic receives incoming data, writes it to the configuration device, and loads the factory configuration. The factory configuration will read the remote update status register and determine the valid application configuration to load. Figure 3–2 shows the Stratix remote update. Figure 3–3 shows the transition diagram for remote update mode.

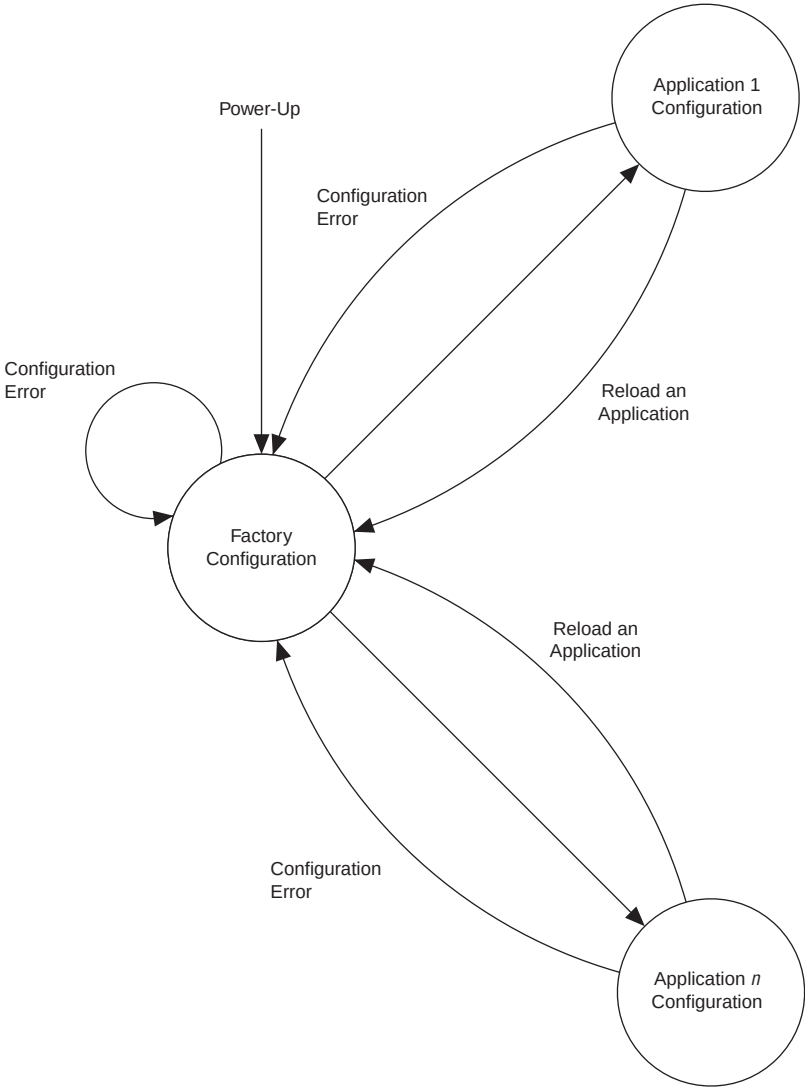
Figure 3–2. Stratix Device Remote Update



Note to Figure 3–2:

(1) When the Stratix device is configured with the factory configuration, it can handle update data from EPC16, EPC8, or EPC4 configuration device pages and point to the next page in the configuration device.

Figure 3–3. Remote Update Transition Diagram *Notes (1), (2)*



Notes to Figure 3–3:

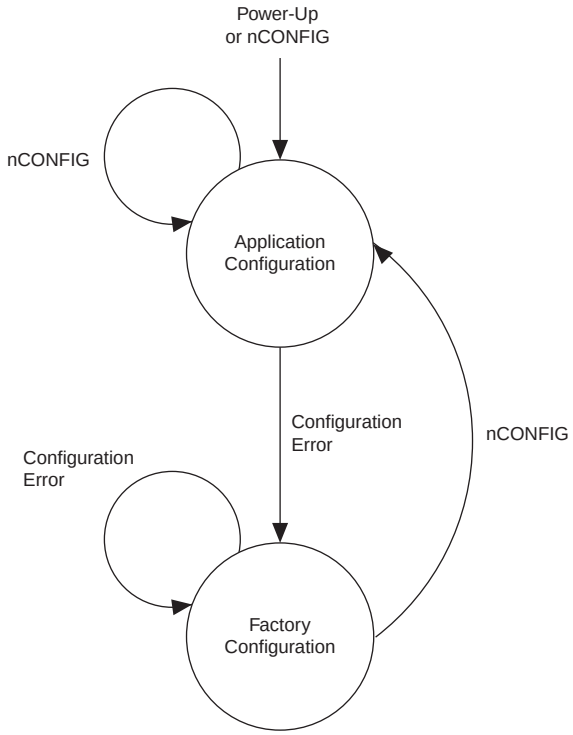
(1) Remote update of Application Configuration is controlled by a Nios embedded processor or user logic programmed in the Factory or Application configurations.

(2) Up to seven pages can be specified allowing up to seven different configuration applications.

Local Update Mode

Local update mode is a simplified version of the remote update. This feature is intended for simple systems that need to load a single application configuration immediately upon power up without loading the factory configuration first. Local update designs have only one application configuration to load, so it does not require a factory configuration to determine which application configuration to use. Figure 3–4 shows the transition diagram for local update mode.

Figure 3–4. Local Update Transition Diagram



Stratix Automated Single Event Upset (SEU) Detection

Stratix devices offer on-chip circuitry for automated checking of single event upset (SEU) detection. FPGA devices that operate at high elevations or in close proximity to earth's North or South Pole require periodic checks to ensure continued data integrity. The error detection cyclic redundancy check (CRC) feature controlled by the **Device & Pin Options** dialog box in the Quartus II software uses a 32-bit CRC circuit to ensure data reliability and is one of the best options for mitigating SEU.

For Stratix, the CRC is computed by the Quartus II software and downloaded into the device as a part of the configuration bit stream. The `CRC_ERROR` pin reports a soft error when configuration SRAM data is corrupted, triggering device reconfiguration.

Custom-Built Circuitry

Dedicated circuitry is built in the Stratix devices to perform error detection automatically. You can use the built-in dedicated circuitry for error detection using CRC feature in Stratix devices, eliminating the need for external logic. This circuitry will perform error detection automatically when enabled. This error detection circuitry in Stratix devices constantly checks for errors in the configuration SRAM cells while the device is in user mode. You can monitor one external pin for the error and use it to trigger a re-configuration cycle. Select the desired time between checks by adjusting a built-in clock divider.

Software Interface

In the Quartus II software version 4.1 and later, you can turn on the automated error detection CRC feature in the **Device & Pin Options** dialog box. This dialog box allows you to enable the feature and set the internal frequency of the CRC between 400 kHz to 100 MHz. This controls the rate that the CRC circuitry verifies the internal configuration SRAM bits in the FPGA device.

For more information on CRC, see *AN 357: Error Detection Using CRC in Altera FPGA Devices*.

Temperature Sensing Diode

Stratix devices include a diode-connected transistor for use as a temperature sensor in power management. This diode is used with an external digital thermometer device such as a MAX1617A or MAX1619 from MAXIM Integrated Products. These devices steer bias current through the Stratix diode, measuring forward voltage and converting this reading to temperature in the form of an 8-bit signed number (7 bits plus sign). The external device's output represents the junction temperature of the Stratix device and can be used for intelligent power management.

The diode requires two pins (`tempdiodep` and `tempdioden`) on the Stratix device to connect to the external temperature-sensing device, as shown in Figure 3–5. The temperature sensing diode is a passive element and therefore can be used before the Stratix device is powered.

Figure 3–5. External Temperature-Sensing Diode

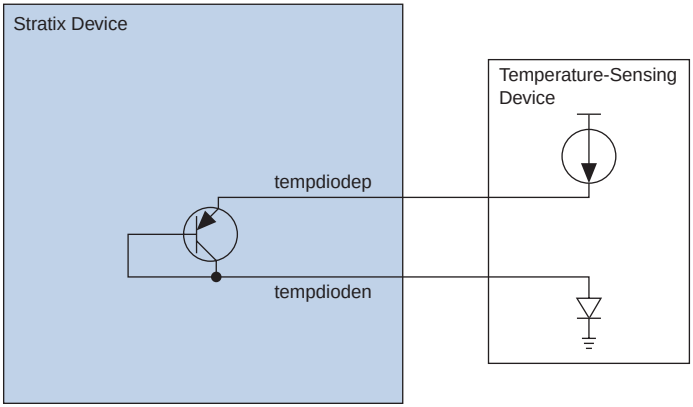


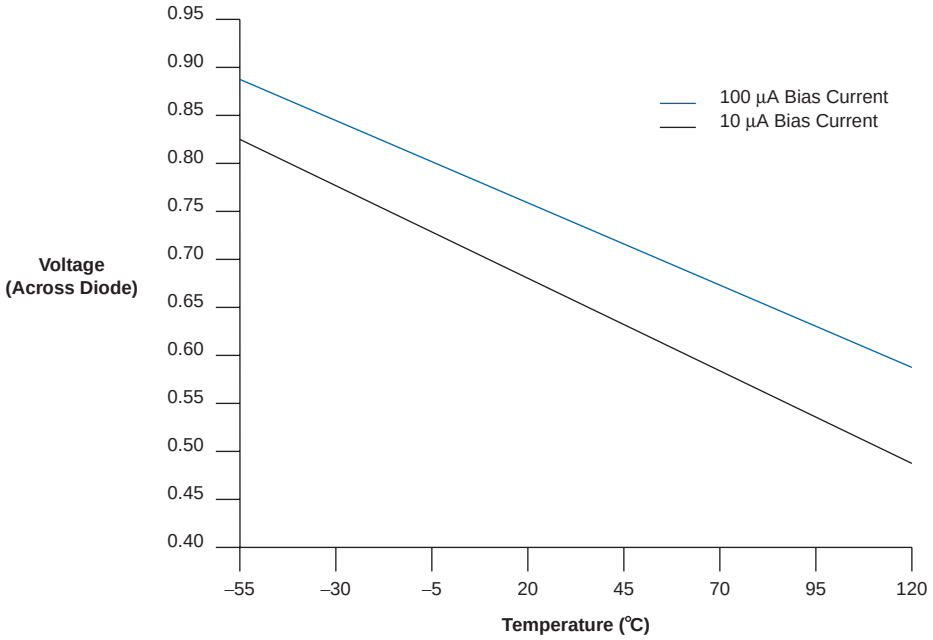
Table 3–6 shows the specifications for bias voltage and current of the Stratix temperature sensing diode.

Table 3–6. Temperature-Sensing Diode Electrical Characteristics

Parameter	Minimum	Typical	Maximum	Unit
I_{BIAS} high	80	100	120	µA
I_{BIAS} low	8	10	12	µA
$V_{BP} – V_{BN}$	0.3		0.9	V
V_{BN}		0.7		V
Series resistance			3	W

The temperature-sensing diode works for the entire operating range shown in Figure 3–6.

Figure 3–6. Temperature vs. Temperature-Sensing Diode Voltage



4. DC & Switching Characteristics

S51004-3.4

Operating Conditions

Stratix® devices are offered in both commercial and industrial grades. Industrial devices are offered in -6 and -7 speed grades and commercial devices are offered in -5 (fastest), -6, -7, and -8 speed grades. This section specifies the operation conditions for operating junction temperature, V_{CCINT} and V_{CCIO} voltage levels, and input voltage requirements. The voltage specifications in this section are specified at the pins of the device (and not the power supply). If the device operates outside these ranges, then all DC and AC specifications are not guaranteed. Furthermore, the reliability of the device may be affected. The timing parameters in this chapter apply to both commercial and industrial temperature ranges unless otherwise stated.

Tables 4–1 through 4–8 provide information on absolute maximum ratings.

Table 4–1. Stratix Device Absolute Maximum Ratings Notes (1), (2)

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCINT}	Supply voltage	With respect to ground	–0.5	2.4	V
V_{CCIO}			–0.5	4.6	V
V_I	DC input voltage *(3)*		–0.5	4.6	V
I_{OUT}	DC output current, per pin		–25	40	mA
T_{STG}	Storage temperature	No bias	–65	150	°C
T_J	Junction temperature	BGA packages under bias		135	°C

Table 4–2. Stratix Device Recommended Operating Conditions (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCINT}	Supply voltage for internal logic and input buffers	*(4)*	1.425	1.575	V

Table 4–2. Stratix Device Recommended Operating Conditions (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Supply voltage for output buffers, 3.3-V operation	*(4)*, *(5)*	3.00 (3.135)	3.60 (3.465)	V
	Supply voltage for output buffers, 2.5-V operation	*(4)*	2.375	2.625	V
	Supply voltage for output buffers, 1.8-V operation	*(4)*	1.71	1.89	V
	Supply voltage for output buffers, 1.5-V operation	*(4)*	1.4	1.6	V
V_I	Input voltage	*(3)*, *(6)*	–0.5	4.0	V
V_O	Output voltage		0	V_{CCIO}	V
T_J	Operating junction temperature	For commercial use	0	85	°C
		For industrial use	–40	100	°C

Table 4–3. Stratix Device DC Operating Conditions Note (7) (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
I_I	Input pin leakage current	$V_I = V_{CCIOmax}$ to 0 V *(8)*	–10		10	µA
I_{OZ}	Tri-stated I/O pin leakage current	$V_O = V_{CCIOmax}$ to 0 V *(8)*	–10		10	µA
I_{CC0}	V_{CC} supply current (standby) (All memory blocks in power-down mode)	V_I = ground, no load, no toggling inputs				mA
		EP1S10. V_I = ground, no load, no toggling inputs		37		mA
		EP1S20. V_I = ground, no load, no toggling inputs		65		mA
		EP1S25. V_I = ground, no load, no toggling inputs		90		mA
		EP1S30. V_I = ground, no load, no toggling inputs		114		mA
		EP1S40. V_I = ground, no load, no toggling inputs		145		mA
		EP1S60. V_I = ground, no load, no toggling inputs		200		mA
		EP1S80. V_I = ground, no load, no toggling inputs		277		mA

Table 4–3. Stratix Device DC Operating Conditions Note (7) (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
R_{CONF}	Value of I/O pin pull-up resistor before and during configuration	V_{CCIO} = 3.0 V *(9)*	20		50	kΩ
		V_{CCIO} = 2.375 V *(9)*	30		80	kΩ
		V_{CCIO} = 1.71 V *(9)*	60		150	kΩ

Table 4–4. LVTTL Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.6	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
V_{OH}	High-level output voltage	I_{OH} = –4 to –24 mA *(10)*	2.4		V
V_{OL}	Low-level output voltage	I_{OL} = 4 to 24 mA *(10)*		0.45	V

Table 4–5. LVCMOS Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.6	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
V_{OH}	High-level output voltage	V_{CCIO} = 3.0, I_{OH} = –0.1 mA	V_{CCIO} – 0.2		V
V_{OL}	Low-level output voltage	V_{CCIO} = 3.0, I_{OL} = 0.1 mA		0.2	V

Table 4–6. 2.5-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		2.375	2.625	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
V_{OH}	High-level output voltage	I_{OH} = –1 mA *(10)*	2.0		V
V_{OL}	Low-level output voltage	I_{OL} = 1 mA *(10)*		0.4	V

Table 4–7. 1.8-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.95	V
V_{IH}	High-level input voltage		$0.65 \times V_{CCIO}$	2.25	V
V_{IL}	Low-level input voltage		–0.3	$0.35 \times V_{CCIO}$	V
V_{OH}	High-level output voltage	I_{OH} = –2 to –8 mA *(10)*	V_{CCIO} – 0.45		V
V_{OL}	Low-level output voltage	I_{OL} = 2 to 8 mA *(10)*		0.45	V

Table 4–8. 1.5-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		1.4	1.6	V
V_{IH}	High-level input voltage		$0.65 \times V_{CCIO}$	V_{CCIO} + 0.3	V
V_{IL}	Low-level input voltage		–0.3	$0.35 \times V_{CCIO}$	V
V_{OH}	High-level output voltage	I_{OH} = –2 mA *(10)*	$0.75 \times V_{CCIO}$		V
V_{OL}	Low-level output voltage	I_{OL} = 2 mA *(10)*		$0.25 \times V_{CCIO}$	V

Notes to Tables 4–1 through 4–8:

(1) See the *Operating Requirements for Altera Devices Data Sheet*.

(2) Conditions beyond those listed in Table 4–1 may cause permanent damage to a device. Additionally, device operation at the absolute maximum ratings for extended periods of time may have adverse affects on the device.

(3) Minimum DC input is –0.5 V. During transitions, the inputs may undershoot to –2.0 V for input currents less than 100 mA and periods shorter than 20 ns, or overshoot to the voltage shown in Table 4–9, based on input duty cycle for input currents less than 100 mA. The overshoot is dependent upon duty cycle of the signal. The DC case is equivalent to 100% duty cycle.

(4) Maximum V_{CC} rise time is 100 ms, and V_{CC} must rise monotonically.

(5) V_{CCIO} maximum and minimum conditions for LVPECL, LVDS, and 3.3-V PCML are shown in parentheses.

(6) All pins, including dedicated inputs, clock, I/O, and JTAG pins, may be driven before V_{CCINT} and V_{CCIO} are powered.

(7) Typical values are for T_A = 25°C, V_{CCINT} = 1.5 V, and V_{CCIO} = 1.5 V, 1.8 V, 2.5 V, and 3.3 V.

(8) This value is specified for normal device operation. The value may vary during power-up. This applies for all V_{CCIO} settings (3.3, 2.5, 1.8, and 1.5 V).

(9) Pin pull-up resistance values will lower if an external source drives the pin higher than V_{CCIO}.

(10) Drive strength is programmable according to the values shown in the *Stratix Architecture* chapter of the *Stratix Device Handbook, Volume 1*.

Table 4–9. Overshoot Input Voltage with Respect to Duty Cycle (Part 1 of 2)

Vin (V)	Maximum Duty Cycle (%)
4.0	100
4.1	90
4.2	50

Table 4–9. Overshoot Input Voltage with Respect to Duty Cycle (Part 2 of 2)

Vin (V)	Maximum Duty Cycle (%)
4.3	30
4.4	17
4.5	10

Figures 4–1 and 4–2 show receiver input and transmitter output waveforms, respectively, for all differential I/O standards (LVDS, 3.3-V PCML, LVPECL, and HyperTransport technology).

Figure 4–1. Receiver Input Waveforms for Differential I/O Standards

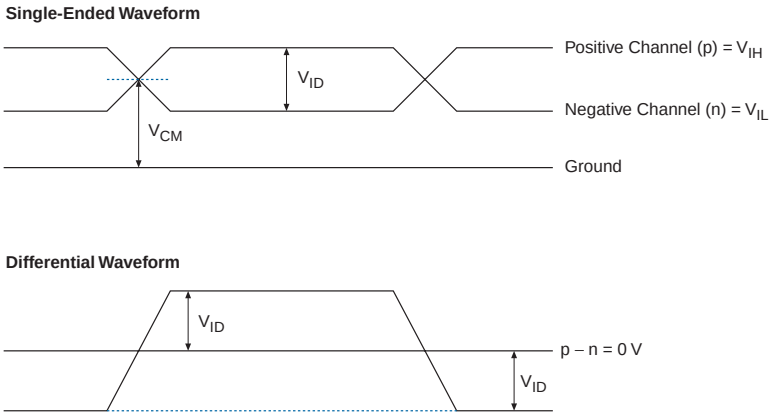
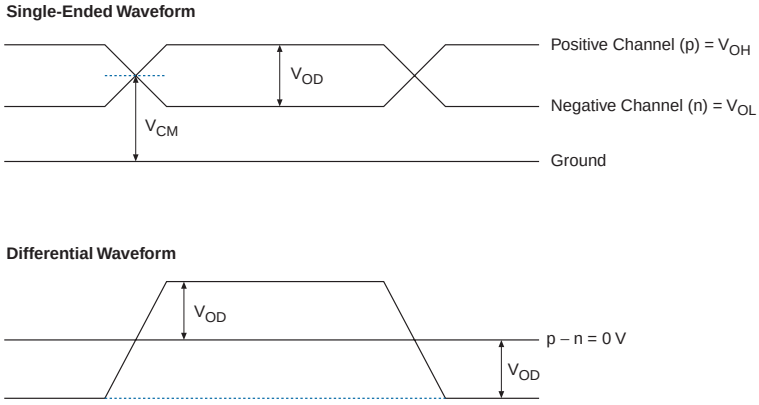


Figure 4–2. Transmitter Output Waveforms for Differential I/O Standards



Tables 4–10 through 4–33 recommend operating conditions, DC operating conditions, and capacitance for 1.5-V Stratix devices.

Table 4–10. 3.3-V LVDS I/O Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	I/O supply voltage		3.135	3.3	3.465	V
V_{ID} *(6)*	Input differential voltage swing (single-ended)	0.1 V ≤ V_{CM} < 1.1 V *W* = 1 through 10	300		1,000	mV
		1.1 V ≤ V_{CM} ≤ 1.6 V *W* = 1	200		1,000	mV
		1.1 V ≤ V_{CM} ≤ 1.6 V *W* = 2 through10	100		1,000	mV
		1.6 V < V_{CM} ≤ 1.8 V *W* = 1 through 10	300		1,000	mV

Table 4–10. 3.3-V LVDS I/O Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{ICM}	Input common mode voltage *(6)*	LVDS 0.3 V ≤V_{ID} ≤1.0 V *W* = 1 through 10	100		1,100	mV
		LVDS 0.3 V ≤V_{ID} ≤1.0 V *W* = 1 through 10	1,600		1,800	mV
		LVDS 0.2 V ≤V_{ID} ≤1.0 V *W* = 1	1,100		1,600	mV
		LVDS 0.1 V ≤V_{ID} ≤1.0 V *W* = 2 through 10	1,100		1,600	mV
V_{OD} *(1)*	Output differential voltage (single-ended)	R_L = 100 Ω	250	375	550	mV
Δ V_{OD}	Change in V_{OD} between high and low	R_L = 100 Ω			50	mV
V_{OCM}	Output common mode voltage	R_L = 100 Ω	1,125	1,200	1,375	mV
Δ V_{OCM}	Change in V_{OCM} between high and low	R_L = 100 Ω			50	mV
R_L	Receiver differential input discrete resistor (external to Stratix devices)		90	100	110	Ω

Table 4–11. 3.3-V PCML Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	I/O supply voltage		3.135	3.3	3.465	V
V_{ID} (peak-to-peak)	Input differential voltage swing (single-ended)		300		600	mV
V_{ICM}	Input common mode voltage		1.5		3.465	V
V_{OD}	Output differential voltage (single-ended)		300	370	500	mV
ΔV_{OD}	Change in V_{OD} between high and low				50	mV
V_{OCM}	Output common mode voltage		2.5	2.85	3.3	V
ΔV_{OCM}	Change in V_{OCM} between high and low				50	mV
V_T	Output termination voltage			V_{CCIO}		V
R_1	Output external pull-up resistors		45	50	55	Ω
R_2	Output external pull-up resistors		45	50	55	Ω

Table 4–12. LVPECL Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	I/O supply voltage		3.135	3.3	3.465	V
V_{ID} (peak-to-peak)	Input differential voltage swing (single-ended)		300		1,000	mV
V_{ICM}	Input common mode voltage		1		2	V
V_{OD}	Output differential voltage (single-ended)	R_L = 100 Ω	525	700	970	mV
V_{OCM}	Output common mode voltage	R_L = 100 Ω	1.5	1.7	1.9	V
R_L	Receiver differential input resistor		90	100	110	Ω

Table 4–13. HyperTransport Technology Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	I/O supply voltage		2.375	2.5	2.625	V
V_{ID} (peak-to-peak)	Input differential voltage swing (single-ended)		300		900	mV
V_{ICM}	Input common mode voltage		300		900	mV
V_{OD}	Output differential voltage (single-ended)	R_L = 100 Ω	380	485	820	mV
Δ V_{OD}	Change in V_{OD} between high and low	R_L = 100 Ω			50	mV
V_{OCM}	Output common mode voltage	R_L = 100 Ω	440	650	780	mV
Δ V_{OCM}	Change in V_{OCM} between high and low	R_L = 100 Ω			50	mV
R_L	Receiver differential input resistor		90	100	110	Ω

Table 4–14. 3.3-V PCI Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.3	3.6	V
V_{IH}	High-level input voltage		0.5 × V_{CCIO}		V_{CCIO} + 0.5	V
V_{IL}	Low-level input voltage		–0.5		0.3 × V_{CCIO}	V
V_{OH}	High-level output voltage	I_{OUT} = –500 µA	0.9 × V_{CCIO}			V
V_{OL}	Low-level output voltage	I_{OUT} = 1,500 µA			0.1 × V_{CCIO}	V

Table 4–15. PCI-X 1.0 Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0		3.6	V
V_{IH}	High-level input voltage		0.5 × V_{CCIO}		V_{CCIO} + 0.5	V
V_{IL}	Low-level input voltage		–0.5		0.35 × V_{CCIO}	V
V_{IPU}	Input pull-up voltage		0.7 × V_{CCIO}			V
V_{OH}	High-level output voltage	I_{OUT} = –500 µA	0.9 × V_{CCIO}			V
V_{OL}	Low-level output voltage	I_{OUT} = 1,500 µA			0.1 × V_{CCIO}	V

Table 4–16. GTL+ I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{TT}	Termination voltage		1.35	1.5	1.65	V
V_{REF}	Reference voltage		0.88	1.0	1.12	V
V_{IH}	High-level input voltage		V_{REF} + 0.1			V
V_{IL}	Low-level input voltage				V_{REF} – 0.1	V
V_{OL}	Low-level output voltage	I_{OL} = 34 mA *(3)*			0.65	V

Table 4–17. GTL I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{TT}	Termination voltage		1.14	1.2	1.26	V
V_{REF}	Reference voltage		0.74	0.8	0.86	V
V_{IH}	High-level input voltage		V_{REF} + 0.05			V
V_{IL}	Low-level input voltage				V_{REF} – 0.05	V
V_{OL}	Low-level output voltage	I_{OL} = 40 mA *(3)*			0.4	V

Table 4–18. SSTL-18 Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.8	1.95	V
V_{REF}	Reference voltage		0.8	0.9	1.0	V
V_{TT}	Termination voltage		V_{REF} – 0.04	V_{REF}	V_{REF} + 0.04	V
$V_{IH(DC)}$	High-level DC input voltage		V_{REF} + 0.125			V
$V_{IL(DC)}$	Low-level DC input voltage				V_{REF} – 0.125	V
$V_{IH(AC)}$	High-level AC input voltage		V_{REF} + 0.275			V
$V_{IL(AC)}$	Low-level AC input voltage				V_{REF} – 0.275	V
V_{OH}	High-level output voltage	I_{OH} = –6.7 mA *(3)*	V_{TT} + 0.475			V
V_{OL}	Low-level output voltage	I_{OL} = 6.7 mA *(3)*			V_{TT} – 0.475	V

Table 4–19. SSTL-18 Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.8	1.95	V
V_{REF}	Reference voltage		0.8	0.9	1.0	V
V_{TT}	Termination voltage		V_{REF} – 0.04	V_{REF}	V_{REF} + 0.04	V
$V_{IH(DC)}$	High-level DC input voltage		V_{REF} + 0.125			V
$V_{IL(DC)}$	Low-level DC input voltage				V_{REF} – 0.125	V
$V_{IH(AC)}$	High-level AC input voltage		V_{REF} + 0.275			V
$V_{IL(AC)}$	Low-level AC input voltage				V_{REF} – 0.275	V
V_{OH}	High-level output voltage	I_{OH} = –13.4 mA *(3)*	V_{TT} + 0.630			V
V_{OL}	Low-level output voltage	I_{OL} = 13.4 mA *(3)*			V_{TT} – 0.630	V

Table 4–20. SSTL-2 Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		2.375	2.5	2.625	V
V_{TT}	Termination voltage		V_{REF} – 0.04	V_{REF}	V_{REF} + 0.04	V
V_{REF}	Reference voltage		1.15	1.25	1.35	V
$V_{IH(DC)}$	High-level DC input voltage		V_{REF} + 0.18		3.0	V
$V_{IL(DC)}$	Low-level DC input voltage		–0.3		V_{REF} – 0.18	V
$V_{IH(AC)}$	High-level AC input voltage		V_{REF} + 0.35			V
$V_{IL(AC)}$	Low-level AC input voltage				V_{REF} – 0.35	V
V_{OH}	High-level output voltage	I_{OH} = –8.1 mA *(3)*	V_{TT} + 0.57			V
V_{OL}	Low-level output voltage	I_{OL} = 8.1 mA *(3)*			V_{TT} – 0.57	V

Table 4–21. SSTL-2 Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		2.375	2.5	2.625	V
V_{TT}	Termination voltage		V_{REF} – 0.04	V_{REF}	V_{REF} + 0.04	V
V_{REF}	Reference voltage		1.15	1.25	1.35	V
$V_{IH(DC)}$	High-level DC input voltage		V_{REF} + 0.18		V_{CCIO} + 0.3	V
$V_{IL(DC)}$	Low-level DC input voltage		–0.3		V_{REF} – 0.18	V
$V_{IH(AC)}$	High-level AC input voltage		V_{REF} + 0.35			V
$V_{IL(AC)}$	Low-level AC input voltage				V_{REF} – 0.35	V
V_{OH}	High-level output voltage	I_{OH} = –16.4 mA *(3)*	V_{TT} + 0.76			V
V_{OL}	Low-level output voltage	I_{OL} = 16.4 mA *(3)*			V_{TT} – 0.76	V

Table 4–22. SSTL-3 Class I Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.3	3.6	V
V_{TT}	Termination voltage		V_{REF} – 0.05	V_{REF}	V_{REF} + 0.05	V
V_{REF}	Reference voltage		1.3	1.5	1.7	V
$V_{IH(DC)}$	High-level DC input voltage		V_{REF} + 0.2		V_{CCIO} + 0.3	V
$V_{IL(DC)}$	Low-level DC input voltage		–0.3		V_{REF} – 0.2	V
$V_{IH(AC)}$	High-level AC input voltage		V_{REF} + 0.4			V

Table 4–22. SSTL-3 Class I Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{IL(AC)}$	Low-level AC input voltage				V_{REF} – 0.4	V
V_{OH}	High-level output voltage	I_{OH} = –8 mA *(3)*	V_{TT} + 0.6			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA *(3)*			V_{TT} – 0.6	V

Table 4–23. SSTL-3 Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.3	3.6	V
V_{TT}	Termination voltage		V_{REF} – 0.05	V_{REF}	V_{REF} + 0.05	V
V_{REF}	Reference voltage		1.3	1.5	1.7	V
$V_{IH(DC)}$	High-level DC input voltage		V_{REF} + 0.2		V_{CCIO} + 0.3	V
$V_{IL(DC)}$	Low-level DC input voltage		–0.3		V_{REF} – 0.2	V
$V_{IH(AC)}$	High-level AC input voltage		V_{REF} + 0.4			V
$V_{IL(AC)}$	Low-level AC input voltage				V_{REF} – 0.4	V
V_{OH}	High-level output voltage	I_{OH} = –16 mA *(3)*	V_{TT} + 0.8			V
V_{OL}	Low-level output voltage	I_{OL} = 16 mA *(3)*			V_{TT} – 0.8	V

Table 4–24. 3.3-V AGP 2× Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.15	3.3	3.45	V
V_{REF}	Reference voltage		0.39 × V_{CCIO}		0.41 × V_{CCIO}	V
V_{IH}	High-level input voltage *(4)*		0.5 × V_{CCIO}		V_{CCIO} + 0.5	V
V_{IL}	Low-level input voltage *(4)*				0.3 × V_{CCIO}	V
V_{OH}	High-level output voltage	I_{OUT} = –0.5 mA	0.9 × V_{CCIO}		3.6	V
V_{OL}	Low-level output voltage	I_{OUT} = 1.5 mA			0.1 × V_{CCIO}	V

Table 4–25. 3.3-V AGP 1× Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.15	3.3	3.45	V
V_{IH}	High-level input voltage *(4)*		0.5 × V_{CCIO}		V_{CCIO} + 0.5	V
V_{IL}	Low-level input voltage *(4)*				0.3 × V_{CCIO}	V

Table 4–25. 3.3-V AGP 1× Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{OH}	High-level output voltage	I_{OUT} = –0.5 mA	0.9 × V_{CCIO}		3.6	V
V_{OL}	Low-level output voltage	I_{OUT} = 1.5 mA			0.1 × V_{CCIO}	V

Table 4–26. 1.5-V HSTL Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.4	1.5	1.6	V
V_{REF}	Input reference voltage		0.68	0.75	0.9	V
V_{TT}	Termination voltage		0.7	0.75	0.8	V
V_{IH} (DC)	DC high-level input voltage		V_{REF} + 0.1			V
V_{IL} (DC)	DC low-level input voltage		–0.3		V_{REF} – 0.1	V
V_{IH} (AC)	AC high-level input voltage		V_{REF} + 0.2			V
V_{IL} (AC)	AC low-level input voltage				V_{REF} – 0.2	V
V_{OH}	High-level output voltage	I_{OH} = –8 mA *(3)*	V_{CCIO} – 0.4			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA *(3)*			0.4	V

Table 4–27. 1.5-V HSTL Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.4	1.5	1.6	V
V_{REF}	Input reference voltage		0.68	0.75	0.9	V
V_{TT}	Termination voltage		0.7	0.75	0.8	V
V_{IH} (DC)	DC high-level input voltage		V_{REF} + 0.1			V
V_{IL} (DC)	DC low-level input voltage		–0.3		V_{REF} – 0.1	V
V_{IH} (AC)	AC high-level input voltage		V_{REF} + 0.2			V
V_{IL} (AC)	AC low-level input voltage				V_{REF} – 0.2	V
V_{OH}	High-level output voltage	I_{OH} = –16 mA *(3)*	V_{CCIO} – 0.4			V
V_{OL}	Low-level output voltage	I_{OL} = 16 mA *(3)*			0.4	V

Table 4–28. 1.8-V HSTL Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.80	1.95	V
V_{REF}	Input reference voltage		0.70	0.90	0.95	V
V_{TT}	Termination voltage			$V_{CCIO} \times 0.5$		V
V_{IH} (DC)	DC high-level input voltage		V_{REF} + 0.1			V
V_{IL} (DC)	DC low-level input voltage		–0.5		V_{REF} – 0.1	V
V_{IH} (AC)	AC high-level input voltage		V_{REF} + 0.2			V
V_{IL} (AC)	AC low-level input voltage				V_{REF} – 0.2	V
V_{OH}	High-level output voltage	I_{OH} = –8 mA *(3)*	V_{CCIO} – 0.4			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA *(3)*			0.4	V

Table 4–29. 1.8-V HSTL Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.80	1.95	V
V_{REF}	Input reference voltage		0.70	0.90	0.95	V
V_{TT}	Termination voltage			$V_{CCIO} \times 0.5$		V
V_{IH} (DC)	DC high-level input voltage		V_{REF} + 0.1			V
V_{IL} (DC)	DC low-level input voltage		–0.5		V_{REF} – 0.1	V
V_{IH} (AC)	AC high-level input voltage		V_{REF} + 0.2			V
V_{IL} (AC)	AC low-level input voltage				V_{REF} – 0.2	V
V_{OH}	High-level output voltage	I_{OH} = –16 mA *(3)*	V_{CCIO} – 0.4			V
V_{OL}	Low-level output voltage	I_{OL} = 16 mA *(3)*			0.4	V

Table 4–30. 1.5-V Differential HSTL Class I & Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	I/O supply voltage		1.4	1.5	1.6	V
V_{DIF} (DC)	DC input differential voltage		0.2			V
V_{CM} (DC)	DC common mode input voltage		0.68		0.9	V
V_{DIF} (AC)	AC differential input voltage		0.4			V

Table 4–31. CTT I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		2.05	3.3	3.6	V
V_{TT}/V_{REF}	Termination and input reference voltage		1.35	1.5	1.65	V
V_{IH}	High-level input voltage		V_{REF} + 0.2			V
V_{IL}	Low-level input voltage				V_{REF} – 0.2	V
V_{OH}	High-level output voltage	I_{OH} = –8 mA	V_{REF} + 0.4			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA			V_{REF} – 0.4	V
I_O	Output leakage current (when output is high *Z*)	GND ≤V_{OUT} ≤ V_{CCIO}	–10		10	µA

Table 4–32. Bus Hold Parameters

Parameter	Conditions	V_{CCIO} Level								Unit
		1.5 V		1.8 V		2.5 V		3.3 V		
		Min	Max	Min	Max	Min	Max	Min	Max	
Low sustaining current	V_{IN} > V_{IL} (maximum)	25		30		50		70		µA
High sustaining current	V_{IN} < V_{IH} (minimum)	-25		–30		–50		–70		µA
Low overdrive current	0 V < V_{IN} < V_{CCIO}		160		200		300		500	µA
High overdrive current	0 V < V_{IN} < V_{CCIO}		-160		–200		–300		–500	µA
Bus-hold trip point		0.5	1.0	0.68	1.07	0.7	1.7	0.8	2.0	V

Table 4–33. Stratix Device Capacitance Note (5)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
C_{IOTB}	Input capacitance on I/O pins in I/O banks 3, 4, 7, and 8.		11.5		pF
C_{IOLR}	Input capacitance on I/O pins in I/O banks 1, 2, 5, and 6, including high-speed differential receiver and transmitter pins.		8.2		pF
C_{CLKTB}	Input capacitance on top/bottom clock input pins: `CLK[4:7]` and `CLK[12:15]`.		11.5		pF
C_{CLKLR}	Input capacitance on left/right clock inputs: `CLK1`, `CLK3`, `CLK8`, `CLK10`.		7.8		pF
C_{CLKLR+}	Input capacitance on left/right clock inputs: `CLK0`, `CLK2`, `CLK9`, and `CLK11`.		4.4		pF

Notes to Tables 4–10 through 4–33:

(1) When `tx_outclock` port of `altlvds_tx` megafunction is 717 MHz, $V_{OD(min)}$ = 235 mV on the output clock pin.

(2) Pin pull-up resistance values will lower if an external source drives the pin higher than V_{CCIO}.

(3) Drive strength is programmable according to the values shown in the *Stratix Architecture* chapter of the *Stratix Device Handbook, Volume 1*.

(4) V_{REF} specifies the center point of the switching range.

(5) Capacitance is sample-tested only. Capacitance is measured using time-domain reflections (TDR). Measurement accuracy is within ±0.5 pF.

(6) V_{IO} and V_{CM} have multiple ranges and values for J=1 through 10.

Power Consumption

Altera® offers two ways to calculate power for a design: the Altera web power calculator and the PowerGauge™ feature in the Quartus® II software.

The interactive power calculator on the Altera web site is typically used prior to designing the FPGA in order to get a magnitude estimate of the device power. The Quartus II software PowerGauge feature allows you to apply test vectors against your design for more accurate power consumption modeling.

In both cases, these calculations should only be used as an estimation of power, not as a specification.

Stratix devices require a certain amount of power-up current to successfully power up because of the small process geometry on which they are fabricated.

Table 4–34 shows the maximum power-up current (I_{CCINT}) required to power a Stratix device. This specification is for commercial operating conditions. Measurements were performed with an isolated Stratix device on the board to characterize the power-up current of an isolated

device. Decoupling capacitors were not used in this measurement. To factor in the current for decoupling capacitors, sum up the current for each capacitor using the following equation:

$$I = C\,(dV / dt)$$

If the regulator or power supply minimum output current is more than the Stratix device requires, then the device may consume more current than the maximum current listed in Table 4–34. However, the device does not require any more current to successfully power up than what is listed in Table 4–34.

Table 4–34. Stratix Power-Up Current (I_{CCINT}) Requirements *Note (1)*

Device	Power-Up Current Requirement		Unit
	Typical	Maximum	
EP1S10	250	700	mA
EP1S20	400	1,200	mA
EP1S25	500	1,500	mA
EP1S30	550	1,900	mA
EP1S40	650	2,300	mA
EP1S60	800	2,600	mA
EP1S80	1,000	3,000	mA

Note to Table 4–34:

(1) The maximum test conditions are for 0° C and typical test conditions are for 40° C.

The exact amount of current consumed varies according to the process, temperature, and power ramp rate. Stratix devices typically require less current during power up than shown in Table 4–34. The user-mode current during device operation is generally higher than the power-up current.

The duration of the I_{CCINT} power-up requirement depends on the V_{CCINT} voltage supply rise time. The power-up current consumption drops when the V_{CCINT} supply reaches approximately 0.75 V.

Timing Model

The DirectDrive™ technology and MultiTrack™ interconnect ensure predictable performance, accurate simulation, and accurate timing analysis across all Stratix device densities and speed grades. This section describes and specifies the performance, internal, external, and PLL timing specifications.

All specifications are representative of worst-case supply voltage and junction temperature conditions.

Preliminary & Final Timing

Timing models can have either preliminary or final status. The Quartus II software issues an informational message during the design compilation if the timing models are preliminary. Table 4–35 shows the status of the Stratix device timing models.

Preliminary status means the timing model is subject to change. Initially, timing numbers are created using simulation results, process data, and other known parameters. These tests are used to make the preliminary numbers as close to the actual timing parameters as possible.

Final timing numbers are based on actual device operation and testing. These numbers reflect the actual performance of the device under worst-case voltage and junction temperature conditions.

Table 4–35. Stratix Device Timing Model Status

Device	Preliminary	Final
EP1S10		v
EP1S20		v
EP1S25		v
EP1S30		v
EP1S40		v
EP1S60		v
EP1S80		v

Performance

Table 4–36 shows Stratix performance for some common designs. All performance values were obtained with Quartus II software compilation of LPM, or MegaCore® functions for the FIR and FFT designs.

Table 4–36. Stratix Performance (Part 1 of 2) *Notes (1), (2)*

Applications		Resources Used			Performance				
		LEs	TriMatrix Memory Blocks	DSP Blocks	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Units
LE	16-to-1 multiplexer *(1)*	22	0	0	407.83	324.56	288.68	228.67	MHz
	32-to-1 multiplexer *(3)*	46	0	0	318.26	255.29	242.89	185.18	MHz
	16-bit counter	16	0	0	422.11	422.11	390.01	348.67	MHz
	64-bit counter	64	0	0	321.85	290.52	261.23	220.5	MHz
TriMatrix memory M512 block	Simple dual-port RAM 32 × 18 bit	0	1	0	317.76	277.62	241.48	205.21	MHz
	FIFO 32 × 18 bit	30	1	0	319.18	278.86	242.54	206.14	MHz
TriMatrix memory M4K block	Simple dual-port RAM 128 × 36 bit	0	1	0	290.86	255.55	222.27	188.89	MHz
	True dual-port RAM 128 × 18 bit	0	1	0	290.86	255.55	222.27	188.89	MHz
	FIFO 128 × 36 bit	34	1	0	290.86	255.55	222.27	188.89	MHz
TriMatrix memory M-RAM block	Single port RAM 4K × 144 bit	1	1	0	255.95	223.06	194.06	164.93	MHz
	Simple dual-port RAM 4K × 144 bit	0	1	0	255.95	233.06	194.06	164.93	MHz
	True dual-port RAM 4K × 144 bit	0	1	0	255.95	233.06	194.06	164.93	MHz
	Single port RAM 8K × 72 bit	0	1	0	278.94	243.19	211.59	179.82	MHz
	Simple dual-port RAM 8K × 72 bit	0	1	0	255.95	223.06	194.06	164.93	MHz
	True dual-port RAM 8K × 72 bit	0	1	0	255.95	223.06	194.06	164.93	MHz
	Single port RAM 16K × 36 bit	0	1	0	280.66	254.32	221.28	188.00	MHz
	Simple dual-port RAM 16K × 36 bit	0	1	0	269.83	237.69	206.82	175.74	MHz

Table 4–36. Stratix Performance (Part 2 of 2) *Notes (1), (2)*

Applications		Resources Used			Performance				
		LEs	TriMatrix Memory Blocks	DSP Blocks	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Units
TriMatrix memory M-RAM block	True dual-port RAM 16K × 36 bit	0	1	0	269.83	237.69	206.82	175.74	MHz
	Single port RAM 32K × 18 bit	0	1	0	275.86	244.55	212.76	180.83	MHz
	Simple dual-port RAM 32K × 18 bit	0	1	0	275.86	244.55	212.76	180.83	MHz
	True dual-port RAM 32K × 18 bit	0	1	0	275.86	244.55	212.76	180.83	MHz
	Single port RAM 64K × 9 bit	0	1	0	287.85	253.29	220.36	187.26	MHz
	Simple dual-port RAM 64K × 9 bit	0	1	0	287.85	253.29	220.36	187.26	MHz
	True dual-port RAM 64K × 9 bit	0	1	0	287.85	253.29	220.36	187.26	MHz
DSP block	9 × 9-bit multiplier *(3)*	0	0	1	335.0	293.94	255.68	217.24	MHz
	18 × 18-bit multiplier *(4)*	0	0	1	278.78	237.41	206.52	175.50	MHz
	36 × 36-bit multiplier *(4)*	0	0	1	148.25	134.71	117.16	99.59	MHz
	36 × 36-bit multiplier *(5)*	0	0	1	278.78	237.41	206.52	175.5	MHz
	18-bit, 4-tap FIR filter	0	0	1	278.78	237.41	206.52	175.50	MHz
Larger Designs	8-bit, 16-tap parallel FIR filter	58	0	4	141.26	133.49	114.88	100.28	MHz
	8-bit, 1,024-point FFT function	870	5	1	261.09	235.51	205.21	175.22	MHz

Notes to Table 4–36:

(1) These design performance numbers were obtained using the Quartus II software.
(2) Numbers not listed will be included in a future version of the data sheet.
(3) This application uses registered inputs and outputs.
(4) This application uses registered multiplier input and output stages within the DSP block.
(5) This application uses registered multiplier input, pipeline, and output stages within the DSP block.

Internal Timing Parameters

Internal timing parameters are specified on a speed grade basis independent of device density. Tables 4–37 through 4–42 describe the Stratix device internal timing microparameters for LEs, IOEs, TriMatrix™ memory structures, DSP blocks, and MultiTrack interconnects.

Table 4–37. LE Internal Timing Microparameter Descriptions

Symbol	Parameter
t_{SU}	LE register setup time before clock
t_{H}	LE register hold time after clock
t_{CO}	LE register clock-to-output delay
t_{LUT}	LE combinatorial LUT delay for data-in to data-out
t_{CLR}	Minimum clear pulse width
t_{PRE}	Minimum preset pulse width
t_{CLKHL}	Register minimum clock high or low time. The maximum core clock frequency can be calculated by $1/(2 \times t_{CLKHL})$.

Table 4–38. IOE Internal Timing Microparameter Descriptions

Symbol	Parameter
t_{SU_R}	Row IOE input register setup time
t_{SU_C}	Column IOE input register setup time
t_{H}	IOE input and output register hold time after clock
t_{CO_R}	Row IOE input and output register clock-to-output delay
t_{CO_C}	Column IOE input and output register clock-to-output delay
$t_{PIN2COMBOUT_R}$	Row input pin to IOE combinatorial output
$t_{PIN2COMBOUT_C}$	Column input pin to IOE combinatorial output
$t_{COMBIN2PIN_R}$	Row IOE data input to combinatorial output pin
$t_{COMBIN2PIN_C}$	Column IOE data input to combinatorial output pin
t_{CLR}	Minimum clear pulse width
t_{PRE}	Minimum preset pulse width
t_{CLKHL}	Register minimum clock high or low time. The maximum I/O clock frequency can be calculated by $1/(2 \times t_{CLKHL})$. Performance may also be affected by I/O timing, use of PLL, and I/O programmable settings.

Table 4–39. DSP Block Internal Timing Microparameter Descriptions

Symbol	Parameter
t_{SU}	Input, pipeline, and output register setup time before clock
t_{H}	Input, pipeline, and output register hold time after clock
t_{CO}	Input, pipeline, and output register clock-to-output delay
$t_{INREG2PIPE9}$	Input Register to DSP Block pipeline register in 9 × 9-bit mode
$t_{INREG2PIPE18}$	Input Register to DSP Block pipeline register in 18 × 18-bit mode
$t_{PIPE2OUTREG2ADD}$	DSP Block Pipeline Register to output register delay in Two-Multipliers Adder mode
$t_{PIPE2OUTREG4ADD}$	DSP Block Pipeline Register to output register delay in Four-Multipliers Adder mode
t_{PD9}	Combinatorial input to output delay for 9 × 9
t_{PD18}	Combinatorial input to output delay for 18 × 18
t_{PD36}	Combinatorial input to output delay for 36 × 36
t_{CLR}	Minimum clear pulse width
t_{CLKHL}	Register minimum clock high or low time. This is a limit on the min time for the clock on the registers in these blocks. The actual performance is dependent upon the internal point-to-point delays in the blocks and may give slower performance as shown in Table 4–36 on page 4–20 and as reported by the timing analyzer in the Quartus II software.

Table 4–40. M512 Block Internal Timing Microparameter Descriptions

Symbol	Parameter
t_{M512RC}	Synchronous read cycle time
t_{M512WC}	Synchronous write cycle time
$t_{M512WERESU}$	Write or read enable setup time before clock
$t_{M512WEREH}$	Write or read enable hold time after clock
$t_{M512CLKENSU}$	Clock enable setup time before clock
$t_{M512CLKENH}$	Clock enable hold time after clock
$t_{M512DATASU}$	Data setup time before clock
$t_{M512DATAH}$	Data hold time after clock
$t_{M512WADDRSU}$	Write address setup time before clock
$t_{M512WADDRH}$	Write address hold time after clock
$t_{M512RADDRSU}$	Read address setup time before clock
$t_{M512RADDRH}$	Read address hold time after clock
$t_{M512DATACO1}$	Clock-to-output delay when using output registers
$t_{M512DATACO2}$	Clock-to-output delay without output registers
$t_{M512CLKHL}$	Register minimum clock high or low time. This is a limit on the min time for the clock on the registers in these blocks. The actual performance is dependent upon the internal point-to-point delays in the blocks and may give slower performance as shown in Table 4–36 on page 4–20 and as reported by the timing analyzer in the Quartus II software.
$t_{M512CLR}$	Minimum clear pulse width

Table 4–41. M4K Block Internal Timing Microparameter Descriptions (Part 1 of 2)

Symbol	Parameter
t_{M4KRC}	Synchronous read cycle time
t_{M4KWC}	Synchronous write cycle time
$t_{M4KWERESU}$	Write or read enable setup time before clock
$t_{M4KWEREH}$	Write or read enable hold time after clock
$t_{M4KCLKENSU}$	Clock enable setup time before clock
$t_{M4KCLKENH}$	Clock enable hold time after clock
$t_{M4KBESU}$	Byte enable setup time before clock
t_{M4KBEH}	Byte enable hold time after clock
$t_{M4KDATAASU}$	A port data setup time before clock

Table 4–41. M4K Block Internal Timing Microparameter Descriptions (Part 2 of 2)

Symbol	Parameter
$t_{M4KDATAAH}$	A port data hold time after clock
$t_{M4KADDRASU}$	A port address setup time before clock
$t_{M4KADDRAH}$	A port address hold time after clock
$t_{M4KDATABSU}$	B port data setup time before clock
$t_{M4KDATABH}$	B port data hold time after clock
$t_{M4KADDRBSU}$	B port address setup time before clock
$t_{M4KADDRBH}$	B port address hold time after clock
$t_{M4KDATACO1}$	Clock-to-output delay when using output registers
$t_{M4KDATACO2}$	Clock-to-output delay without output registers
$t_{M4KCLKHL}$	Register minimum clock high or low time. This is a limit on the min time for the clock on the registers in these blocks. The actual performance is dependent upon the internal point-to-point delays in the blocks and may give slower performance as shown inTable 4–36 on page 4–20 and as reported by the timing analyzer in the Quartus II software.
t_{M4KCLR}	Minimum clear pulse width

Table 4–42. M-RAM Block Internal Timing Microparameter Descriptions (Part 1 of 2)

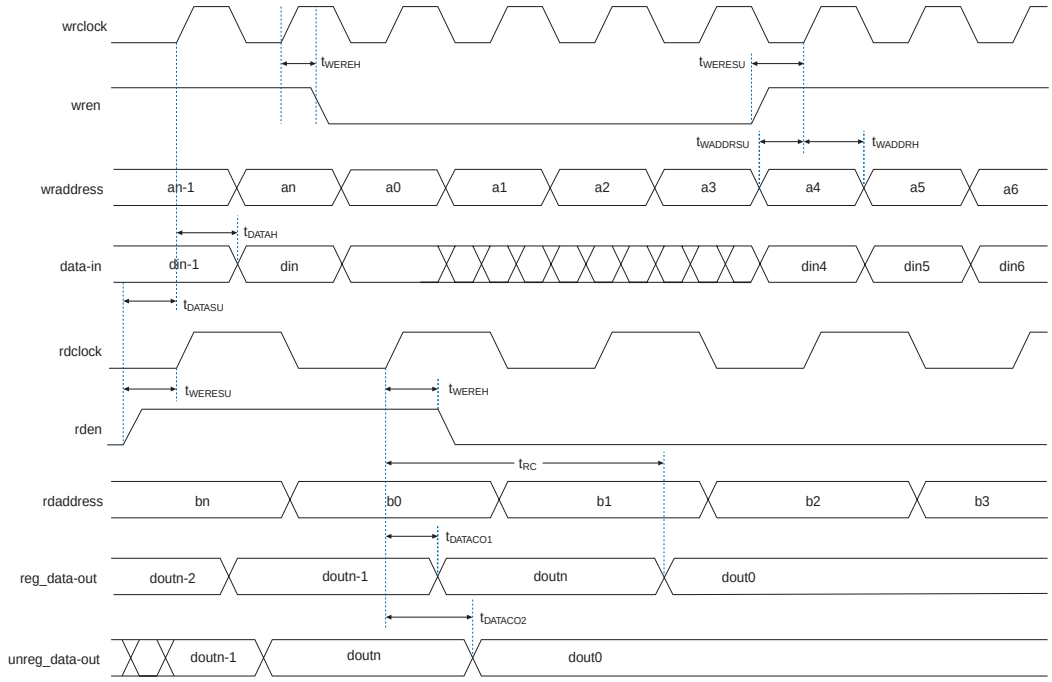
Symbol	Parameter
t_{MRAMRC}	Synchronous read cycle time
t_{MRAMWC}	Synchronous write cycle time
$t_{MRAMWERESU}$	Write or read enable setup time before clock
$t_{MRAMWEREH}$	Write or read enable hold time after clock
$t_{MRAMCLKENSU}$	Clock enable setup time before clock
$t_{MRAMCLKENH}$	Clock enable hold time after clock
$t_{MRAMBESU}$	Byte enable setup time before clock
$t_{MRAMBEH}$	Byte enable hold time after clock
$t_{MRAMDATAASU}$	A port data setup time before clock
$t_{MRAMDATAAH}$	A port data hold time after clock
$t_{MRAMADDRASU}$	A port address setup time before clock
$t_{MRAMADDRAH}$	A port address hold time after clock
$t_{MRAMDATABSU}$	B port setup time before clock

Table 4–42. M-RAM Block Internal Timing Microparameter Descriptions (Part 2 of 2)

Symbol	Parameter
$t_{MRAMDATABH}$	B port hold time after clock
$t_{MRAMADDRBSU}$	B port address setup time before clock
$t_{MRAMADDRBH}$	B port address hold time after clock
$t_{MRAMDATACO1}$	Clock-to-output delay when using output registers
$t_{MRAMDATACO2}$	Clock-to-output delay without output registers
$t_{MRAMCLKHL}$	Register minimum clock high or low time. This is a limit on the min time for the clock on the registers in these blocks. The actual performance is dependent upon the internal point-to-point delays in the blocks and may give slower performance as shown in Table 4–36 on page 4–20 and as reported by the timing analyzer in the Quartus II software.
$t_{MRAMCLR}$	Minimum clear pulse width.

Figure 4–3 shows the TriMatrix memory waveforms for the M512, M4K, and M-RAM timing parameters shown in Tables 4–40 through 4–42.

Figure 4–3. Dual-Port RAM Timing Microparameter Waveform



Internal timing parameters are specified on a speed grade basis independent of device density. Tables 4–44 through 4–50 show the internal timing microparameters for LEs, IOEs, TriMatrix memory structures, DSP blocks, and MultiTrack interconnects.

Table 4–43. Routing Delay Internal Timing Microparameter Descriptions (Part 1 of 2)

Symbol	Parameter
t_{R4}	Delay for an R4 line with average loading; covers a distance of four LAB columns.
t_{R8}	Delay for an R8 line with average loading; covers a distance of eight LAB columns.
t_{R24}	Delay for an R24 line with average loading; covers a distance of 24 LAB columns.

Table 4–43. Routing Delay Internal Timing Microparameter Descriptions (Part 2 of 2)

Symbol	Parameter
t_{C4}	Delay for a C4 line with average loading; covers a distance of four LAB rows.
t_{C8}	Delay for a C8 line with average loading; covers a distance of eight LAB rows.
t_{C16}	Delay for a C16 line with average loading; covers a distance of 16 LAB rows.
t_{LOCAL}	Local interconnect delay, for connections within a LAB, and for the final routing hop of connections to LABs, DSP blocks, RAM blocks and I/Os.

Table 4–44. LE Internal Timing Microparameters

Parameter	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{SU}	10		10		11		13		ps
t_{H}	100		100		114		135		ps
t_{CO}		156		176		202		238	ps
t_{LUT}		366		459		527		621	ps
t_{CLR}	100		100		114		135		ps
t_{PRE}	100		100		114		135		ps
t_{CLKHL}	1000		1111		1190		1400		ps

Table 4–45. IOE Internal TSU Microparameter by Device Density (Part 1 of 2)

Device	Symbol	-5		-6		-7		-8		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
EP1S10	t_{SU_R}	76		80		80		80		ps
	t_{SU_C}	176		80		80		80		ps
EP1S20	t_{SU_R}	76		80		80		80		ps
	t_{SU_C}	76		80		80		80		ps
EP1S25	t_{SU_R}	276		280		280		280		ps
	t_{SU_C}	276		280		280		280		ps
EP1S30	t_{SU_R}	76		80		80		80		ps
	t_{SU_C}	176		180		180		180		ps

Table 4–45. IOE Internal TSU Microparameter by Device Density (Part 2 of 2)

Device	Symbol	-5		-6		-7		-8		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
EP1S40	t_{SU_R}	76		80		80		80		ps
	t_{SU_C}	376		380		380		380		ps
EP1S60	t_{SU_R}	276		280		280		280		ps
	t_{SU_C}	276		280		280		280		ps
EP1S80	t_{SU_R}	426		430		430		430		ps
	t_{SU_C}	76		80		80		80		ps

Table 4–46. IOE Internal Timing Microparameters

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_H	68		71		82		96		ps
t_{CO_R}		171		179		206		242	ps
t_{CO_C}		171		179		206		242	ps
$t_{PIN2COMBOUT_R}$		1,234		1,295		1,490		1,753	ps
$t_{PIN2COMBOUT_C}$		1,087		1,141		1,312		1,544	ps
$t_{COMBIN2PIN_R}$		3,894		4,089		4,089		4,089	ps
$t_{COMBIN2PIN_C}$		4,299		4,494		4,494		4,494	ps
t_{CLR}	276		289		333		392		ps
t_{PRE}	260		273		313		369		ps
t_{CLKHL}	1,000		1,111		1,190		1,400		ps

Table 4–47. DSP Block Internal Timing Microparameters (Part 1 of 2)

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{SU}	0		0		0		0		ps
t_H	67		75		86		101		ps
t_{CO}		142		158		181		214	ps
$t_{INREG2PIPE9}$		2,613		2,982		3,429		4,035	ps
$t_{INREG2PIPE18}$		3,390		3,993		4,591		5,402	ps

Table 4–47. DSP Block Internal Timing Microparameters (Part 2 of 2)

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
$t_{PIPE2OUTREG2ADD}$		2,002		2,203		2,533		2,980	ps
$t_{PIPE2OUTREG4ADD}$		2,899		3,189		3,667		4,314	ps
t_{PD9}		3,709		4,081		4,692		5,520	ps
t_{PD18}		4,795		5,275		6,065		7,135	ps
t_{PD36}		7,495		8,245		9,481		11,154	ps
t_{CLR}	450		500		575		676		ps
t_{CLKHL}	1,350		1,500		1,724		2,029		ps

Table 4–48. M512 Block Internal Timing Microparameters

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{M512RC}		3,340		3,816		4,387		5,162	ps
t_{M512WC}		3,138		3,590		4,128		4,860	ps
$t_{M512WERESU}$	110		123		141		166		ps
$t_{M512WEREH}$	34		38		43		51		ps
$t_{M512CLKENSU}$	215		215		247		290		ps
$t_{M512CLKENH}$	–70		–70		–81		–95		ps
$t_{M512DATASU}$	110		123		141		166		ps
$t_{M512DATAH}$	34		38		43		51		ps
$t_{M512WADDRSU}$	110		123		141		166		ps
$t_{M512WADDRH}$	34		38		43		51		ps
$t_{M512RADDRSU}$	110		123		141		166		ps
$t_{M512RADDRH}$	34		38		43		51		ps
$t_{M512DATACO1}$		424		472		541		637	ps
$t_{M512DATACO2}$		3,366		3,846		4,421		5,203	ps
$t_{M512CLKHL}$	1,000		1,111		1,190		1,400		ps
$t_{M512CLR}$	170		189		217		255		ps

Table 4–49. M4K Block Internal Timing Microparameters

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{M4KRC}		3,807		4,320		4,967		5,844	ps
t_{M4KWC}		2,556		2,840		3,265		3,842	ps
$t_{M4KWERESU}$	131		149		171		202		ps
$t_{M4KWEREH}$	34		38		43		51		ps
$t_{M4KCLKENSU}$	193		215		247		290		ps
$t_{M4KCLKENH}$	–63		–70		–81		–95		ps
$t_{M4KBESU}$	131		149		171		202		ps
t_{M4KBEH}	34		38		43		51		ps
$t_{M4KDATAASU}$	131		149		171		202		ps
$t_{M4KDATAAH}$	34		38		43		51		ps
$t_{M4KADDRASU}$	131		149		171		202		ps
$t_{M4KADDRAH}$	34		38		43		51		ps
$t_{M4KDATABSU}$	131		149		171		202		ps
$t_{M4KDATABH}$	34		38		43		51		ps
$t_{M4KADDRBSU}$	131		149		171		202		ps
$t_{M4KADDRBH}$	34		38		43		51		ps
$t_{M4KDATACO1}$		571		635		729		858	ps
$t_{M4KDATACO2}$		3,984		4,507		5,182		6,097	ps
$t_{M4KCLKHL}$	1,000		1,111		1,190		1,400		ps
t_{M4KCLR}	170		189		217		255		ps

Table 4–50. M-RAM Block Internal Timing Microparameters (Part 1 of 2)

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{MRAMRC}		4,364		4,838		5,562		6,544	ps
t_{MRAMWC}		3,654		4,127		4,746		5,583	ps
$t_{MRAMWERESU}$	25		25		28		33		ps
$t_{MRAMWEREH}$	18		20		23		27		ps
$t_{MRAMCLKENSU}$	99		111		127		150		ps
$t_{MRAMCLKENH}$	–48		–53		–61		–72		ps

Table 4–50. M-RAM Block Internal Timing Microparameters (Part 2 of 2)

Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
$t_{MRAMBESU}$	25		25		28		33		ps
$t_{MRAMBEH}$	18		20		23		27		ps
$t_{MRAMDATAASU}$	25		25		28		33		ps
$t_{MRAMDATAAH}$	18		20		23		27		ps
$t_{MRAMADDRASU}$	25		25		28		33		ps
$t_{MRAMADDRAH}$	18		20		23		27		ps
$t_{MRAMDATABSU}$	25		25		28		33		ps
$t_{MRAMDATABH}$	18		20		23		27		ps
$t_{MRAMADDRBSU}$	25		25		28		33		ps
$t_{MRAMADDRBH}$	18		20		23		27		ps
$t_{MRAMDATACO1}$		1,038		1,053		1,210		1,424	ps
$t_{MRAMDATACO2}$		4,362		4,939		5,678		6,681	ps
$t_{MRAMCLKHL}$	1,000		1,111		1,190		1,400		ps
$t_{MRAMCLR}$	135		150		172		202		ps

Table 4–51. Routing Delay Internal Timing Parameters

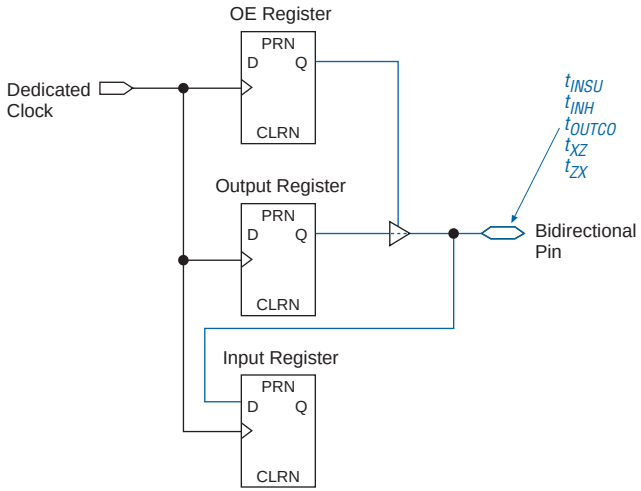
Symbol	-5		-6		-7		-8		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{R4}		268		295		339		390	ps
t_{R8}		371		349		401		461	ps
t_{R24}		465		512		588		676	ps
t_{C4}		440		484		557		641	ps
t_{C8}		577		634		730		840	ps
t_{C16}		445		489		563		647	ps
t_{LOCAL}		313		345		396		455	ps

Routing delays vary depending on the load on that specific routing line. The Quartus II software reports the routing delay information when running the timing analysis for a design.

External Timing Parameters

External timing parameters are specified by device density and speed grade. Figure 4–4 shows the pin-to-pin timing model for bidirectional IOE pin timing. All registers are within the IOE.

Figure 4–4. External Timing in Stratix Devices



All external timing parameters reported in this section are defined with respect to the dedicated clock pin as the starting point. All external I/O timing parameters shown are for 3.3-V LVTTL I/O standard with the 24-mA current strength and fast slew rate. For external I/O timing using standards other than LVTTL or for different current strengths, use the I/O standard input and output delay adders in Tables 4–103 through 4–108.

Table 4–52 shows the external I/O timing parameters when using fast regional clock networks.

Table 4–52. Stratix Fast Regional Clock External I/O Timing Parameters *Notes (1), (2)*

Symbol	Parameter
t_{INSU}	Setup time for input or bidirectional pin using IOE input register with fast regional clock fed by FCLK pin
t_{INH}	Hold time for input or bidirectional pin using IOE input register with fast regional clock fed by FCLK pin
t_{OUTCO}	Clock-to-output delay output or bidirectional pin using IOE output register with fast regional clock fed by FCLK pin
t_{XZ}	Synchronous IOE output enable register to output pin disable delay using fast regional clock fed by FCLK pin
t_{ZX}	Synchronous IOE output enable register to output pin enable delay using fast regional clock fed by FCLK pin

Notes to Table 4–52:

(1) These timing parameters are sample-tested only.

(2) These timing parameters are for column and row IOE pins. You should use the Quartus II software to verify the external timing for any pin.

Table 4–53 shows the external I/O timing parameters when using regional clock networks.

Table 4–53. Stratix Regional Clock External I/O Timing Parameters (Part 1 of 2) *Notes (1), (2)*

Symbol	Parameter
t_{INSU}	Setup time for input or bidirectional pin using IOE input register with regional clock fed by CLK pin
t_{INH}	Hold time for input or bidirectional pin using IOE input register with regional clock fed by CLK pin
t_{OUTCO}	Clock-to-output delay output or bidirectional pin using IOE output register with regional clock fed by CLK pin
$t_{INSUPLL}$	Setup time for input or bidirectional pin using IOE input register with regional clock fed by Enhanced PLL with default phase setting
t_{INHPLL}	Hold time for input or bidirectional pin using IOE input register with regional clock fed by Enhanced PLL with default phase setting
$t_{OUTCOPLL}$	Clock-to-output delay output or bidirectional pin using IOE output register with regional clock Enhanced PLL with default phase setting

Table 4–53. Stratix Regional Clock External I/O Timing Parameters (Part 2 of 2) *Notes (1), (2)*

Symbol	Parameter
t_{XZPLL}	Synchronous IOE output enable register to output pin disable delay using regional clock fed by Enhanced PLL with default phase setting
t_{ZXPLL}	Synchronous IOE output enable register to output pin enable delay using regional clock fed by Enhanced PLL with default phase setting

Notes to Table 4–53:

(1) These timing parameters are sample-tested only.

(2) These timing parameters are for column and row IOE pins. You should use the Quartus II software to verify the external timing for any pin.

Table 4–54 shows the external I/O timing parameters when using global clock networks.

Table 4–54. Stratix Global Clock External I/O Timing Parameters *Notes (1), (2)*

Symbol	Parameter
t_{INSU}	Setup time for input or bidirectional pin using IOE input register with global clock fed by CLK pin
t_{INH}	Hold time for input or bidirectional pin using IOE input register with global clock fed by CLK pin
t_{OUTCO}	Clock-to-output delay output or bidirectional pin using IOE output register with global clock fed by CLK pin
$t_{INSUPLL}$	Setup time for input or bidirectional pin using IOE input register with global clock fed by Enhanced PLL with default phase setting
t_{INHPLL}	Hold time for input or bidirectional pin using IOE input register with global clock fed by Enhanced PLL with default phase setting
$t_{OUTCOPLL}$	Clock-to-output delay output or bidirectional pin using IOE output register with global clock Enhanced PLL with default phase setting
t_{XZPLL}	Synchronous IOE output enable register to output pin disable delay using global clock fed by Enhanced PLL with default phase setting
t_{ZXPLL}	Synchronous IOE output enable register to output pin enable delay using global clock fed by Enhanced PLL with default phase setting

Notes to Table 4–54:

(1) These timing parameters are sample-tested only.

(2) These timing parameters are for column and row IOE pins. You should use the Quartus II software to verify the external timing for any pin.

Stratix External I/O Timing

These timing parameters are for both column IOE and row IOE pins. In EP1S30 devices and above, you can decrease the t_{SU} time by using the FPLLCLK, but may get positive hold time in EP1S60 and EP1S80 devices. You should use the Quartus II software to verify the external devices for any pin.

Tables 4–55 through 4–60 show the external timing parameters on column and row pins for EP1S10 devices.

Table 4–55. EP1S10 External I/O Timing on Column Pins Using Fast Regional Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.238		2.325		2.668		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.240	4.549	2.240	4.836	2.240	5.218	NA	NA	ns
t_{XZ}	2.180	4.423	2.180	4.704	2.180	5.094	NA	NA	ns
t_{ZX}	2.180	4.423	2.180	4.704	2.180	5.094	NA	NA	ns

Table 4–56. EP1S10 External I/O Timing on Column Pins Using Regional Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max			
t_{INSU}	1.992		2.054		2.359		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.395	4.795	2.395	5.107	2.395	5.527	NA	NA	ns
t_{XZ}	2.335	4.669	2.335	4.975	2.335	5.403	NA	NA	ns
t_{ZX}	2.335	4.669	2.335	4.975	2.335	5.403	NA	NA	ns
$t_{INSUPLL}$	0.975		0.985		1.097		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA	NA	ns
$t_{OUTCOPLL}$	1.262	2.636	1.262	2.680	1.262	2.769	NA	NA	ns
t_{XZPLL}	1.202	2.510	1.202	2.548	1.202	2.645	NA	NA	ns
t_{ZXPLL}	1.202	2.510	1.202	2.548	1.202	2.645	NA	NA	ns

Table 4–57. EP1S10 External I/O Timing on Column Pins Using Global Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.647		1.692		1.940		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.619	5.184	2.619	5.515	2.619	5.999	NA	NA	ns
t_{XZ}	2.559	5.058	2.559	5.383	2.559	5.875	NA	NA	ns
t_{ZX}	2.559	5.058	2.559	5.383	2.559	5.875	NA	NA	ns
$t_{INSUPLL}$	1.239		1.229		1.374		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.109	2.372	1.109	2.436	1.109	2.492	NA	NA	ns
t_{XZPLL}	1.049	2.246	1.049	2.304	1.049	2.368	NA	NA	ns
t_{ZXPLL}	1.049	2.246	1.049	2.304	1.049	2.368	NA	NA	ns

Table 4–58. EP1S10 External I/O Timing on Row Pin Using Fast Regional Clock Network *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.212		2.403		2.759		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.391	4.838	2.391	5.159	2.391	5.569	NA	NA	ns
t_{XZ}	2.418	4.892	2.418	5.215	2.418	5.637	NA	NA	ns
t_{ZX}	2.418	4.892	2.418	5.215	2.418	5.637	NA	NA	ns

Table 4–59. EP1S10 External I/O Timing on Row Pins Using Regional Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.161		2.336		2.685		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.434	4.889	2.434	5.226	2.434	5.643	NA	NA	ns
t_{XZ}	2.461	4.493	2.461	5.282	2.461	5.711	NA	NA	ns
t_{ZX}	2.461	4.493	2.461	5.282	2.461	5.711	NA	NA	ns
$t_{INSUPLL}$	1.057		1.172		1.315		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.327	2.773	1.327	2.848	1.327	2.940	NA	NA	ns
t_{XZPLL}	1.354	2.827	1.354	2.904	1.354	3.008	NA	NA	ns
t_{ZXPLL}	1.354	2.827	1.354	2.904	1.354	3.008	NA	NA	ns

Table 4–60. EP1S10 External I/O Timing on Row Pins Using Global Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.787		1.944		2.232		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.647	5.263	2.647	5.618	2.647	6.069	NA	NA	ns
t_{XZ}	2.674	5.317	2.674	5.674	2.674	6.164	NA	NA	ns
t_{ZX}	2.674	5.317	2.674	5.674	2.674	6.164	NA	NA	ns
$t_{INSUPLL}$	1.371		1.1472		1.654		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.144	2.459	1.144	2.548	1.144	2.601	NA	NA	ns
t_{XZPLL}	1.171	2.513	1.171	2.604	1.171	2.669	NA	NA	ns
t^{ZXPLL}	1.171	2.513	1.171	2.604	1.171	2.669	NA	NA	ns

Note to Tables 4–55 to 4–60:

(1) Only EP1S25, EP1S30, and EP1S40 have speed grade of -8.

Tables 4–61 through 4–66 show the external timing parameters on column and row pins for EP1S20 devices.

Table 4–61. EP1S20 External I/O Timing on Column Pins Using Fast Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.065		2.245		2.576		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.283	4.622	2.283	4.916	2.283	5.310	NA	NA	ns
t_{XZ}	2.223	4.496	2.223	4.784	2.223	5.186	NA	NA	ns
t_{ZX}	2.223	4.496	2.223	4.784	2.223	5.186	NA	NA	ns

Table 4–62. EP1S20 External I/O Timing on Column Pins Using Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.541		1.680		1.931		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.597	5.146	2.597	5.481	2.597	5.955	NA	NA	ns
t_{XZ}	2.537	5.020	2.537	5.349	2.537	5.831	NA	NA	ns
t_{ZX}	2.537	5.020	2.537	5.349	2.537	5.831	NA	NA	ns
$t_{INSUPLL}$	0.777		0.818		0.937		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.296	2.690	1.296	2.801	1.296	2.876	NA	NA	ns
t_{XZPLL}	1.236	2.564	1.236	2.669	1.236	2.752	NA	NA	ns
t_{ZXPLL}	1.236	2.564	1.236	2.669	1.236	2.752	NA	NA	ns

Table 4–63. EP1S20 External I/O Timing on Column Pins Using Global Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.351		1.479		1.699		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.732	5.380	2.732	5.728	2.732	6.240	NA	NA	ns
t_{XZ}	2.672	5.254	2.672	5.596	2.672	6.116	NA	NA	ns
t_{ZX}	2.672	5.254	2.672	5.596	2.672	6.116	NA	NA	ns
$t_{INSUPLL}$	0.923		0.971		1.098		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.210	2.544	1.210	2.648	1.210	2.715	NA	NA	ns
t_{XZPLL}	1.150	2.418	1.150	2.516	1.150	2.591	NA	NA	ns
t_{ZXPLL}	1.150	2.418	1.150	2.516	1.150	2.591	NA	NA	ns

Table 4–64. EP1S20 External I/O Timing on Row Pins Using Fast Regional Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.032		2.207		2.535		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.492	5.018	2.492	5.355	2.492	5.793	NA	NA	ns
t_{XZ}	2.519	5.072	2.519	5.411	2.519	5.861	NA	NA	ns
t_{ZX}	2.519	5.072	2.519	5.411	2.519	5.861	NA	NA	ns

Table 4–65. EP1S20 External I/O Timing on Row Pins Using Regional Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.815		1.967		2.258		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.633	5.235	2.663	5.595	2.663	6.070	NA	NA	ns
t_{XZ}	2.660	5.289	2.660	5.651	2.660	6.138	NA	NA	ns
t_{ZX}	2.660	5.289	2.660	5.651	2.660	6.138	NA	NA	ns
$t_{INSUPLL}$	1.060		1.112		1.277		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.325	2.770	1.325	2.908	1.325	2.978	NA	NA	ns
t_{XZPLL}	1.352	2.824	1.352	2.964	1.352	3.046	NA	NA	ns
t_{ZXPLL}	1.352	2.824	1.352	2.964	1.352	3.046	NA	NA	ns

Table 4–66. EP1S20 External I/O Timing on Row Pins Using Global Clock Networks *Note (1)*

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.742		1.887		2.170		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.674	5.308	2.674	5.675	2.674	6.158	NA	NA	ns
t_{XZ}	2.701	5.362	2.701	5.731	2.701	6.226	NA	NA	ns
t_{ZX}	2.701	5.362	2.701	5.731	2.701	6.226	NA	NA	ns
$t_{INSUPLL}$	1.353		1.418		1.613		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.158	2.447	1.158	2.602	1.158	2.642	NA	NA	ns
t_{XZPLL}	1.185	2.531	1.158	2.602	1.185	2.710	NA	NA	ns
t_{ZXPLL}	1.185	2.531	1.158	2.602	1.185	2.710	NA	NA	ns

Note to Tables 4–61 to 4–66:

(1) Only EP1S25, EP1S30, and EP1S40 have a speed grade of -8.

Tables 4–67 through 4–72 show the external timing parameters on column and row pins for EP1S25 devices.

Table 4–67. EP1S25 External I/O Timing on Column Pins Using Fast Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.412		2.613		2.968		3.468		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.196	4.475	2.196	4.748	2.196	5.118	2.196	5.603	ns
t_{XZ}	2.136	4.349	2.136	4.616	2.136	4.994	2.136	5.488	ns
t_{ZX}	2.136	4.349	2.136	4.616	2.136	4.994	2.136	5.488	ns

Table 4–68. EP1S25 External I/O Timing on Column Pins Using Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.535		1.661		1.877		2.125		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.739	5.396	2.739	5.746	2.739	6.262	2.739	6.946	ns
t_{XZ}	2.679	5.270	2.679	5.614	2.679	6.138	2.679	6.831	ns
t_{ZX}	2.679	5.270	2.679	5.614	2.679	6.138	2.679	6.831	ns
$t_{INSUPLL}$	0.934		0.980		1.092		1.231		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.316	2.733	1.316	2.839	1.316	2.921	1.316	3.110	ns
t^{XZPLL}	1.256	2.607	1.256	2.707	1.256	2.797	1.256	2.995	ns
t_{ZXPLL}	1.256	2.607	1.256	2.707	1.256	2.797	1.256	2.995	ns

Table 4–69. EP1S25 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.371		1.471		1.657		1.916		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.809	5.516	2.809	5.890	2.809	6.429	2.809	7.155	ns
t_{XZ}	2.749	5.390	2.749	5.758	2.749	6.305	2.749	7.040	ns
t_{ZX}	2.749	5.390	2.749	5.758	2.749	6.305	2.749	7.040	ns
$t_{INSUPLL}$	1.271		1.327		1.491		1.677		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.124	2.396	1.124	2.492	1.124	2.522	1.124	2.602	ns
t_{XZPLL}	1.064	2.270	1.064	2.360	1.064	2.398	1.064	2.487	ns
t_{ZXPLL}	1.064	2.270	1.064	2.360	1.064	2.398	1.064	2.487	ns

Table 4–70. EP1S25 External I/O Timing on Row Pins Using Fast Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.429		2.631		2.990		3.503		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.376	4.821	2.376	5.131	2.376	5.538	2.376	6.063	ns
t_{XZ}	2.403	4.875	2.403	5.187	2.403	5.606	2.403	6.145	ns
t_{ZX}	2.403	4.875	2.403	5.187	2.403	5.606	2.403	6.145	ns

Table 4–71. EP1S25 External I/O Timing on Row Pins Using Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.793		1.927		2.182		2.542		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.759	5.457	2.759	5.835	2.759	6.346	2.759	7.024	ns
t_{XZ}	2.786	5.511	2.786	5.891	2.786	6.414	2.786	7.106	ns
t_{ZX}	2.786	5.511	2.786	5.891	2.786	6.414	2.786	7.106	ns
$t_{INSUPLL}$	1.169		1.221		1.373		1.600		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.375	2.861	1.375	2.999	1.375	3.082	1.375	3.174	ns
t_{XZPLL}	1.402	2.915	1.402	3.055	1.402	3.150	1.402	3.256	ns
t_{ZXPLL}	1.402	2.915	1.402	3.055	1.402	3.150	1.402	3.256	ns

Table 4–72. EP1S25 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.665		1.779		2.012		2.372		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.834	5.585	2.834	5.983	2.834	6.516	2.834	7.194	ns
t_{XZ}	2.861	5.639	2.861	6.039	2.861	6.584	2.861	7.276	ns
t_{ZX}	2.861	5.639	2.861	6.039	2.861	6.584	2.861	7.276	ns
$t_{INSUPLL}$	1.538		1.606		1.816		2.121		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.164	2.492	1.164	2.614	1.164	2.639	1.164	2.653	ns
t_{XZPLL}	1.191	2.546	1.191	2.670	1.191	2.707	1.191	2.735	ns
t_{ZXPLL}	1.191	2.546	1.191	2.670	1.191	2.707	1.191	2.735	ns

Tables 4–73 through 4–78 show the external timing parameters on column and row pins for EP1S30 devices.

Table 4–73. EP1S30 External I/O Timing on Column Pins Using Fast Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.502		2.680		3.062		3.591		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.473	4.965	2.473	5.329	2.473	5.784	2.473	6.392	ns
t_{XZ}	2.413	4.839	2.413	5.197	2.413	5.660	2.413	6.277	ns
t_{ZX}	2.413	4.839	2.413	5.197	2.413	5.660	2.413	6.277	ns

Table 4–74. EP1S30 External I/O Timing on Column Pins Using Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.286		2.426		2.769		3.249		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.641	5.225	2.641	5.629	2.641	6.130	2.641	6.796	ns
t_{XZ}	2.581	5.099	2.581	5.497	2.581	6.006	2.581	6.681	ns
t_{ZX}	2.581	5.099	2.581	5.497	2.581	6.006	2.581	6.681	ns
$t_{INSUPLL}$	1.200		1.185		1.344		1.662		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.108	2.367	1.108	2.534	1.108	2.569	1.108	2.517	ns
t_{XZPLL}	1.048	2.241	1.048	2.402	1.048	2.445	1.048	2.402	ns
t_{ZXPLL}	1.048	2.241	1.048	2.402	1.048	2.445	1.048	2.402	ns

Table 4–75. EP1S30 External I/O Timing on Column Pins Using Global Clock Networks (Part 1 of 2)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.935		2.029		2.310		2.709		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.814	5.532	2.814	5.980	2.814	6.536	2.814	7.274	ns

Table 4–75. EP1S30 External I/O Timing on Column Pins Using Global Clock Networks (Part 2 of 2)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{XZ}	2.754	5.406	2.754	5.848	2.754	6.412	2.754	7.159	ns
t_{ZX}	2.754	5.406	2.754	5.848	2.754	6.412	2.754	7.159	ns
$t_{INSUPLL}$	1.265		1.236		1.403		1.756		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.068	2.302	1.068	2.483	1.068	2.510	1.068	2.423	ns
t_{XZPLL}	1.008	2.176	1.008	2.351	1.008	2.386	1.008	2.308	ns
t_{ZXPLL}	1.008	2.176	1.008	2.351	1.008	2.386	1.008	2.308	ns

Table 4–76. EP1S30 External I/O Timing on Row Pins Using Fast Regional Clock Networks

Parameters	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.616		2.808		3.223		3.797		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.542	5.114	2.542	5.502	2.542	5.965	2.542	6.581	ns
t_{XZ}	2.569	5.168	2.569	5.558	2.569	6.033	2.569	6.663	ns
t_{ZX}	2.569	5.168	2.569	5.558	2.569	6.033	2.569	6.663	ns

Table 4–77. EP1S30 External I/O Timing on Row Pins Using Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.322		2.467		2.828		3.342		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.731	5.408	2.731	5.843	2.731	6.360	2.731	7.036	ns
t_{XZ}	2.758	5.462	2.758	5.899	2.758	6.428	2.758	7.118	ns
t_{ZX}	2.758	5.462	2.758	5.899	2.758	6.428	2.758	7.118	ns
$t_{INSUPLL}$	1.291		1.283		1.469		1.832		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.192	2.539	1.192	2.737	1.192	2.786	1.192	2.742	ns
t_{XZPLL}	1.219	2.539	1.219	2.793	1.219	2.854	1.219	2.824	ns
t_{ZXPLL}	1.219	2.539	1.219	2.793	1.219	2.854	1.219	2.824	ns

Table 4–78. EP1S30 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.995		2.089		2.398		2.830		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.917	5.735	2.917	6.221	2.917	6.790	2.917	7.548	ns
t_{XZ}	2.944	5.789	2.944	6.277	2.944	6.858	2.944	7.630	ns
t_{ZX}	2.944	5.789	2.944	6.277	2.944	6.858	2.944	7.630	ns
$t_{INSUPLL}$	1.337		1.312		1.508		1.902		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.164	2.493	1.164	2.708	1.164	2.747	1.164	2.672	ns
t_{XZPLL}	1.191	2.547	1.191	2.764	1.191	2.815	1.191	2.754	ns
t_{ZXPLL}	1.191	2.547	1.191	2.764	1.191	2.815	1.191	2.754	ns

Tables 4–79 through 4–84 show the external timing parameters on column and row pins for EP1S40 devices.

Table 4–79. EP1S40 External I/O Timing on Column Pins Using Fast Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.696		2.907		3.290		2.899		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.506	5.015	2.506	5.348	2.506	5.809	2.698	7.286	ns
t_{XZ}	2.446	4.889	2.446	5.216	2.446	5.685	2.638	7.171	ns
t_{ZX}	2.446	4.889	2.446	5.216	2.446	5.685	2.638	7.171	ns

Table 4–80. EP1S40 External I/O Timing on Column Pins Using Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.413		2.581		2.914		2.938		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.668	5.254	2.668	5.628	2.668	6.132	2.869	7.307	ns
t_{XZ}	2.608	5.128	2.608	5.496	2.608	6.008	2.809	7.192	ns
t_{ZX}	2.608	5.128	2.608	5.496	2.608	6.008	2.809	7.192	ns
$t_{INSUPLL}$	1.385		1.376		1.609		1.837		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.117	2.382	1.117	2.552	1.117	2.504	1.117	2.542	ns
t_{XZPLL}	1.057	2.256	1,057	2.420	1.057	2.380	1.057	2.427	ns
t_{ZXPLL}	1.057	2.256	1,057	2.420	1.057	2.380	1.057	2.427	ns

Table 4–81. EP1S40 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.126		2.268		2.558		2.930		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.856	5.585	2.856	5.987	2.856	6.541	2.847	7.253	ns
t_{XZ}	2.796	5.459	2.796	5.855	2.796	6.417	2.787	7.138	ns
t_{ZX}	2.796	5.459	2.796	5.855	2.796	6.417	2.787	7.138	ns
$t_{INSUPLL}$	1.466		1.455		1.711		1.906		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.092	2.345	1.092	2.510	1.092	2.455	1.089	2.473	ns
t_{XZPLL}	1.032	2.219	1.032	2.378	1.032	2.331	1.029	2.358	ns
t_{ZXPLL}	1.032	2.219	1.032	2.378	1.032	2.331	1.029	2.358	ns

Table 4–82. EP1S40 External I/O Timing on Row Pins Using Fast Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.472		2.685		3.083		3.056		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.631	5.258	2.631	5.625	2.631	6.105	2.745	7.324	ns
t_{XZ}	2.658	5.312	2.658	5.681	2.658	6.173	2.772	7.406	ns
t_{ZX}	2.658	5.312	2.658	5.681	2.658	6.173	2.772	7.406	ns

Table 4–83. EP1S40 External I/O Timing on Row Pins Using Regional Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.349		2.526		2.898		2.952		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.725	5.381	2.725	5.784	2.725	6.290	2.725	7.426	ns
t_{XZ}	2.752	5.435	2.752	5.840	2.752	6.358	2.936	7.508	ns
t_{ZX}	2.752	5.435	2.752	5.840	2.752	6.358	2.936	7.508	ns
$t_{INSUPLL}$	1.328		1.322		1.605		1.883		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.169	2.502	1.169	2.698	1.169	2.650	1.169	2.691	ns
t_{XZPLL}	1.196	2.556	1.196	2.754	1.196	2.718	1.196	2.773	ns
t_{ZXPLL}	1.196	2.556	1.196	2.754	1.196	2.718	1.196	2.773	ns

Table 4–84. EP1S40 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.020		2.171		2.491		2.898		ns
t_{INH}	0.000		0.000		0.000		0.000		ns
t_{OUTCO}	2.912	5.710	2.912	6.139	2.912	6.697	2.931	7.480	ns
t_{XZ}	2.939	5.764	2.939	6.195	2.939	6.765	2.958	7.562	ns
t_{ZX}	2.939	5.764	2.939	6.195	2.939	6.765	2.958	7.562	ns
$t_{INSUPLL}$	1.370		1.368		1.654		1.881		ns
t_{INHPLL}	0.000		0.000		0.000		0.000		ns
$t_{OUTCOPLL}$	1.144	2.460	1.144	2.652	1.144	2.601	1.170	2.693	ns
t_{XZPLL}	1.171	2.514	1.171	2.708	1.171	2.669	1.197	2.775	ns
t_{ZXPLL}	1.171	2.514	1.171	2.708	1.171	2.669	1.197	2.775	ns

Tables 4–85 through 4–90 show the external timing parameters on column and row pins for EP1S60 devices.

Table 4–85. EP1S60 External I/O Timing on Column Pins Using Fast Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	3.029		3.277		3.733		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.446	4.871	2.446	5.215	2.446	5.685	NA	NA	ns
t_{XZ}	2.386	4.745	2.386	5.083	2.386	5.561	NA	NA	ns
t_{ZX}	2.386	4.745	2.386	5.083	2.386	5.561	NA	NA	ns

Table 4–86. EP1S60 External I/O Timing on Column Pins Using Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.491		2.691		3.060		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.767	5.409	2.767	5.801	2.767	6.358	NA	NA	ns
t_{XZ}	2.707	5.283	2.707	5.669	2.707	6.234	NA	NA	ns
t_{ZX}	2.707	5.283	2.707	5.669	2.707	6.234	NA	NA	ns
$t_{INSUPLL}$	1.233		1.270		1.438		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.078	2.278	1.078	2.395	1.078	2.428	NA	NA	ns
t_{XZPLL}	1.018	2.152	1.018	2.263	1.018	2.304	NA	NA	ns
t_{ZXPLL}	1.018	2.152	1.018	2.263	1.018	2.304	NA	NA	ns

Table 4–87. EP1S60 External I/O Timing on Column Pins Using Global Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.000		2.152		2.441		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	3.051	5.900	3.051	6.340	3.051	6.977	NA	NA	ns
t_{XZ}	2.991	5.774	2.991	6.208	2.991	6.853	NA	NA	ns
t_{ZX}	2.991	5.774	2.991	6.208	2.991	6.853	NA	NA	ns
$t_{INSUPLL}$	1.315		1.362		1.543		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.029	2.196	1.029	2.303	1.029	2.323	NA	NA	ns
t_{XZPLL}	0.969	2.070	0.969	2.171	0.969	2.199	NA	NA	ns
t_{ZXPLL}	0.969	2.070	0.969	2.171	0.969	2.199	NA	NA	ns

Table 4–88. EP1S60 External I/O Timing on Row Pins Using Fast Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	3.144		3.393		3.867		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.643	5.275	2.643	5.654	2.643	6.140	NA	NA	ns
t_{XZ}	2.670	5.329	2.670	5.710	2.670	6.208	NA	NA	ns
t_{ZX}	2.670	5.329	2.670	5.710	2.670	6.208	NA	NA	ns

Table 4–89. EP1S60 External I/O Timing on Row Pins Using Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.775		2.990		3.407		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.867	5.644	2.867	6.057	2.867	6.600	NA	NA	ns
t_{XZ}	2.894	5.698	2.894	6.113	2.894	6.668	NA	NA	ns
t_{ZX}	2.894	5.698	2.894	6.113	2.894	6.668	NA	NA	ns
$t_{INSUPLL}$	1.523		1.577		1.791		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.174	2.507	1.174	2.643	1.174	2.664	NA	NA	ns
t_{XZPLL}	1.201	2.561	1.201	2.699	1.201	2.732	NA	NA	ns
t_{ZXPLL}	1.201	2.561	1.201	2.699	1.201	2.732	NA	NA	ns

Table 4–90. EP1S60 External I/O Timing on Row Pins Using Global Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.232		2.393		2.721		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	3.182	6.187	3.182	6.654	3.182	7.286	NA	NA	ns
t_{XZ}	3.209	6.241	3.209	6.710	3.209	7.354	NA	NA	ns
t_{ZX}	3.209	6.241	3.209	6.710	3.209	7.354	NA	NA	ns
$t_{INSUPLL}$	1.651		1.612		1.833		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.154	2.469	1.154	2.608	1.154	2.622	NA	NA	ns
t_{XZPLL}	1.181	2.523	1.181	2.664	1.181	2.690	NA	NA	ns
t_{ZXPLL}	1.181	2.523	1.181	2.664	1.181	2.690	NA	NA	ns

Note to Tables 4–85 to 4–90:

(1) Only EP1S25, EP1S30, and EP1S40 devices have the -8 speed grade.

Tables 4–91 through 4–96 show the external timing parameters on column and row pins for EP1S80 devices.

Table 4–91. EP1S80 External I/O Timing on Column Pins Using Fast Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.328		2.528		2.900		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.422	4.830	2.422	5.169	2.422	5.633	NA	NA	ns
t_{XZ}	2.362	4.704	2.362	5.037	2.362	5.509	NA	NA	ns
t_{ZX}	2.362	4.704	2.362	5.037	2.362	5.509	NA	NA	ns

Table 4–92. EP1S80 External I/O Timing on Column Pins Using Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.760		1.912		2.194		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.761	5.398	2.761	5.785	2.761	6.339	NA	NA	ns
t_{XZ}	2.701	5.272	2.701	5.653	2.701	6.215	NA	NA	ns
t_{ZX}	2.701	5.272	2.701	5.653	2.701	6.215	NA	NA	ns
$t_{INSUPLL}$	0.462		0.606		0.785		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.661	2.849	1.661	2.859	1.661	2.881	NA	NA	ns
t_{XZPLL}	1.601	2.723	1.601	2.727	1.601	2.757	NA	NA	ns
t_{ZXPLL}	1.601	2.723	1.601	2.727	1.601	2.757	NA	NA	ns

Table 4–93. EP1S80 External I/O Timing on Column Pins Using Global Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	0.884		0.976		1.118		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	3.267	6.274	3.267	6.721	3.267	7.415	NA	NA	ns
t_{XZ}	3.207	6.148	3.207	6.589	3.207	7.291	NA	NA	ns
t_{ZX}	3.207	6.148	3.207	6.589	3.207	7.291	NA	NA	ns
$t_{INSUPLL}$	0.506		0.656		0.838		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.635	2.805	1.635	2.809	1.635	2.828	NA	NA	ns
t_{XZPLL}	1.575	2.679	1.575	2.677	1.575	2.704	NA	NA	ns
t_{ZXPLL}	1.575	2.679	1.575	2.677	1.575	2.704	NA	NA	ns

Table 4–94. EP1S80 External I/O Timing on Row Pins Using Fast Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.792		2.993		3.386		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.619	5.235	2.619	5.609	2.619	6.086	NA	NA	ns
t_{XZ}	2.646	5.289	2.646	5.665	2.646	6.154	NA	NA	ns
t_{ZX}	2.646	5.289	2.646	5.665	2.646	6.154	NA	NA	ns

Table 4–95. EP1S80 External I/O Timing on Row Pins Using Regional Clock Networks Note (1)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	2.295		2.454		2.767		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	2.917	5.732	2.917	6.148	2.917	6.705	NA	NA	ns
t_{XZ}	2.944	5.786	2.944	6.204	2.944	6.773	NA	NA	ns
t_{ZX}	2.944	5.786	2.944	6.204	2.944	6.773	NA	NA	ns
$t_{INSUPLL}$	1.011		1.161		1.372		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.808	3.169	1.808	3.209	1.808	3.233	NA	NA	ns
t_{XZPLL}	1.835	3.223	1.835	3.265	1.835	3.301	NA	NA	ns
t_{ZXPLL}	1.835	3.223	1.835	3.265	1.835	3.301	NA	NA	ns

Table 4–96. EP1S80 External I/O Timing on Rows Using Pin Global Clock Networks Note (1)

Symbol	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
t_{INSU}	1.362		1.451		1.613		NA		ns
t_{INH}	0.000		0.000		0.000		NA		ns
t_{OUTCO}	3.457	6.665	3.457	7.151	3.457	7.859	NA	NA	ns
t_{XZ}	3.484	6.719	3.484	7.207	3.484	7.927	NA	NA	ns
t_{ZX}	3.484	6.719	3.484	7.207	3.484	7.927	NA	NA	ns
$t_{INSUPLL}$	o.994		1.143		1.351		NA		ns
t_{INHPLL}	0.000		0.000		0.000		NA		ns
$t_{OUTCOPLL}$	1.821	3.186	1.821	3.227	1.821	3.254	NA	NA	ns
t_{XZPLL}	1.848	3.240	1.848	3.283	1.848	3.322	NA	NA	ns
t_{ZXPLL}	1.848	3.240	1.848	3.283	1.848	3.322	NA	NA	ns

Note to Tables 4–91 to 4–96:

(1) Only EP1S25, EP1S30, and EP1S40 devices have the -8 speed grade.

Definition of I/O Skew

I/O skew is defined as the absolute value of the worst-case difference in clock-to-out times (t_{CO}) between any two output registers fed by a common clock source.

I/O bank skew is made up of the following components:

- Clock network skews: This is the difference between the arrival times of the clock at the clock input port of the two IOE registers.
- Package skews: This is the package trace length differences between (I/O pad A to I/O pin A) and (I/O pad B to I/O pin B).

Figure 4–5 shows an example of two IOE registers located in the same bank, being fed by a common clock source. The clock can come from an input pin or from a PLL output.

Figure 4–5. I/O Skew within an I/O Bank

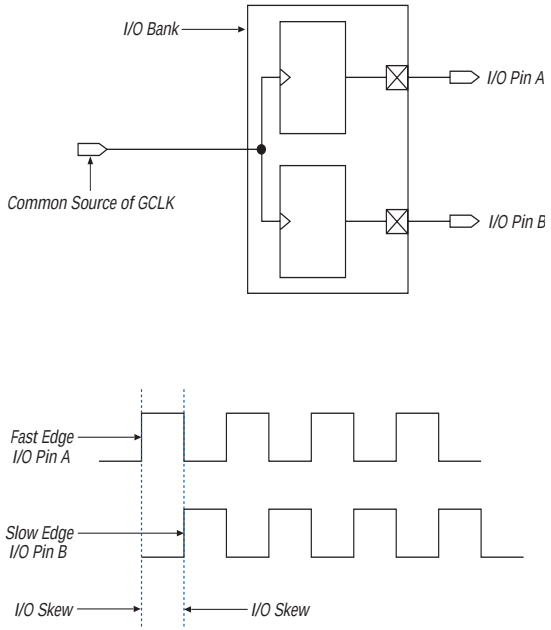


Figure 4–6 shows the case where four IOE registers are located in two different I/O banks.

Figure 4–6. I/O Skew Across Two I/O Banks

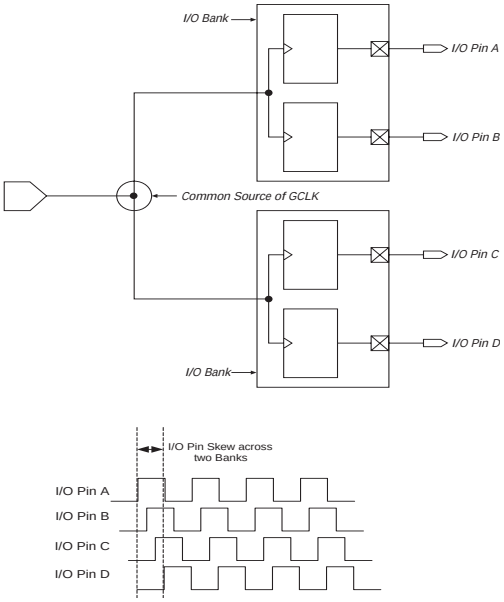


Table 4–97 defines the timing parameters used to define the timing for horizontal I/O pins (side banks 1, 2, 5, 6) and vertical I/O pins (top and bottom banks 3, 4, 7, 8). The timing parameters define the skew within an I/O bank, across two neighboring I/O banks on the same side of the device, across all horizontal I/O banks, across all vertical I/O banks, and the skew for the overall device.

Table 4–97. Output Pin Timing Skew Definitions (Part 1 of 2)

Symbol	Definition
t_{SB_HIO}	Row I/O (HIO) within one I/O bank *(1)*
t_{SB_VIO}	Column I/O (VIO) within one I/O bank *(1)*
t_{SS_HIO}	Row I/O (HIO) same side of the device, across two banks *(2)*
t_{SS_VIO}	Column I/O (VIO) same side of the device, across two banks *(2)*

Table 4–97. Output Pin Timing Skew Definitions (Part 2 of 2)

Symbol	Definition
t_{LR_HIO}	Across all HIO banks (1, 2, 5, 6); across four similar type I/O banks
t_{TB_VIO}	Across all VIO banks (3, 4, 7, 8); across four similar type I/O banks
$t_{OVERALL}$	Output timing skew for all I/O pins on the device.

Notes to Table 4–97:

(1) See Figure 4–5 on page 4–57.

(2) See Figure 4–6 on page 4–58.

Table 4–98 shows the I/O skews when using the same global or regional clock to feed IOE registers in I/O banks around each device. These values can be used for calculating the timing budget on the output (write) side of a memory interface. These values already factor in the package skew.

Table 4–98. Output Skew for Stratix by Device Density

Symbol	Skew (ps) *(1)*		
	EP1S10 to EP1S30	EP1S40	EP1S60 & EP1S80
t_{SB_HIO}	90	290	500
t_{SB_VIO}	160	290	500
t_{SS_HIO}	90	460	600
t_{SS_VIO}	180	520	630
t_{LR_HIO}	150	490	600
t_{TB_VIO}	190	580	670
$t_{OVERALL}$	430	630	880

Note to Table 4–98:

(1) The skew numbers in Table 4–98 account for worst case package skews.

Skew on Input Pins

Table 4–99 shows the package skews that were considered to get the worst case I/O skew value. You can use these values, for example, when calculating the timing budget on the input (read) side of a memory interface.

Table 4–99. Package Skew on Input Pins

Package Parameter	Worst-Case Skew (ps)
Pins in the same I/O bank	50
Pins in top/bottom (vertical I/O) banks	50
Pins in left/right side (horizontal I/O) banks	50
Pins across the entire device	100

PLL Counter & Clock Network Skews

Table 4–100 shows the clock skews between different clock outputs from the Stratix device PLL.

Table 4–100. PLL Counter & Clock Network Skews

Parameter	Worst-Case Skew (ps)
Clock skew between two external clock outputs driven by the same counter	100
Clock skew between two external clock outputs driven by the different counters with the same settings	150
Dual-purpose PLL dedicated clock output used as I/O pin vs. regular I/O pin	270 *(1)*
Clock skew between any two outputs of the PLL that drive global clock networks	150

Note to Table 4–100:

(1) The Quartus II software models 270 ps of delay on the PLL dedicated clock output (`PLL6_OUT[3..0]p/n` and `PLL5_OUT[3..0]p/n`) pins both when used as clocks and when used as I/O pins.

I/O Timing Measurement Methodology

Different I/O standards require different baseline loading techniques for reporting timing delays. Altera characterizes timing delays with the required termination and loading for each I/O standard. The timing information is specified from the input clock pin up to the output pin of

the FPGA device. The Quartus II software calculates the I/O timing for each I/O standard with a default baseline loading as specified by the I/O standard.

Altera measures clock-to-output delays (t_{CO}) at worst-case process, minimum voltage, and maximum temperature (PVT) for the 3.3-V LVTTL I/O standard with 24 mA (default case) current drive strength setting and fast slew rate setting. I/O adder delays are measured to calculate the t_{CO} change at worst-case PVT across all I/O standards and current drive strength settings with the default loading shown in Table 4–101 on page 4–62. Timing derating data for additional loading is taken for t_{CO} across worst-case PVT for all I/O standards and drive strength settings. These three pieces of data are used to predict the timing at the output pin.

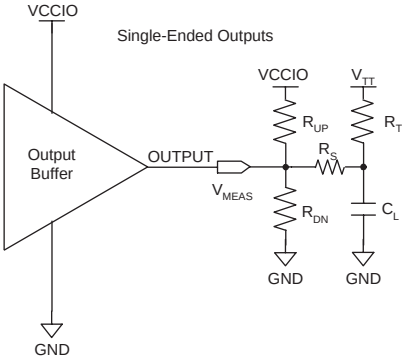
t_{CO} at pin = t_{OUTCO} max for 3.3-V 24 mA LVTTL + I/O Adder + Output Delay Adder for Loading

Simulation using IBIS models is required to determine the delays on the PCB traces in addition to the output pin delay timing reported by the Quartus II software and the timing model in the device handbook.

1. Simulate the output driver of choice into the generalized test setup using values from Table 4–101 on page 4–62.
2. Record the time to VMEAS.
3. Simulate the output driver of choice into the actual PCB trace and load, using the appropriate IBIS input buffer model or an equivalent capacitance value to represent the load.
4. Record the time to VMEAS.
5. Compare the results of steps 2 and 4. The increase or decrease in delay should be added to or subtracted from the I/O Standard Output Adder delays to yield the actual worst-case propagation delay (clock-to-input) of the PCB trace.

The Quartus II software reports maximum timing with the conditions shown in Table 4–101 on page 4–62 using the proceeding equation. Figure 4–7 on page 4–62 shows the model of the circuit that is represented by the Quartus II output timing.

Figure 4–7. Output Delay Timing Reporting Setup Modeled by Quartus II



Notes to Figure 4–7:

(1) Output pin timing is reported at the output pin of the FPGA device. Additional delays for loading and board trace delay need to be accounted for with IBIS model simulations.

(2) V_{CCINT} is 1.42-V unless otherwise specified.

Table 4–101. Reporting Methodology For Maximum Timing For Single-Ended Output Pins (Part 1 of 2)
Notes (1), (2), (3)

I/O Standard	Loading and Termination							Measurement Point
	R_{UP} Ω	R_{DN} Ω	R_S Ω	R_T Ω	V_{CCIO} (V)	VTT (V)	C_L (pF)	V_{MEAS}
3.3-V LVTTL	–	–	0	–	2.950	2.95	10	1.500
2.5-V LVTTL	–	–	0	–	2.370	2.37	10	1.200
1.8-V LVTTL	–	–	0	–	1.650	1.65	10	0.880
1.5-V LVTTL	–	–	0	–	1.400	1.40	10	0.750
3.3-V LVCMOS	–	–	0	–	2.950	2.95	10	1.500
2.5-V LVCMOS	–	–	0	–	2.370	2.37	10	1.200
1.8-V LVCMOS	–	–	0	–	1.650	1.65	10	0.880
1.5-V LVCMOS	–	–	0	–	1.400	1.40	10	0.750
3.3-V GTL	–	–	0	25	2.950	1.14	30	0.740
2.5-V GTL	–	–	0	25	2.370	1.14	30	0.740
3.3-V GTL+	–	–	0	25	2.950	1.35	30	0.880
2.5-V GTL+	–	–	0	25	2.370	1.35	30	0.880
3.3-V SSTL-3 Class II	–	–	25	25	2.950	1.25	30	1.250

Table 4–101. Reporting Methodology For Maximum Timing For Single-Ended Output Pins (Part 2 of 2)
Notes (1), (2), (3)

I/O Standard	Loading and Termination							Measurement Point
	R_{UP} Ω	R_{DN} Ω	R_S Ω	R_T Ω	V_{CCIO} (V)	VTT (V)	C_L (pF)	V_{MEAS}
3.3-V SSTL-3 Class I	–	–	25	50	2.950	1.250	30	1.250
2.5-V SSTL-2 Class II	–	–	25	25	2.370	1.110	30	1.110
2.5-V SSTL-2 Class I	–	–	25	50	2.370	1.110	30	1.110
1.8-V SSTL-18 Class II	–	–	25	25	1.650	0.760	30	0.760
1.8-V SSTL-18 Class I	–	–	25	50	1.650	0.760	30	0.760
1.5-V HSTL Class II	–	–	0	25	1.400	0.700	20	0.680
1.5-V HSTL Class I	–	–	0	50	1.400	0.700	20	0.680
1.8-V HSTL Class II	–	–	0	25	1.650	0.700	20	0.880
1.8-V HSTL Class I	–	–	0	50	1.650	0.700	20	0.880
3.3-V PCI *(4)*	–/25	25/–	0	–	2.950	2.950	10	0.841/1.814
3.3-V PCI-X 1.0 *(4)*	–/25	25/–	0	–	2.950	2.950	10	0.841/1.814
3.3-V Compact PCI *(4)*	–/25	25/–	0	–	2.950	2.950	10	0.841/1.814
3.3-V AGP 1X *(4)*	–/25	25/–	0	–	2.950	2.950	10	0.841/1.814
3.3-V CTT	–	–	25	50	2.050	1.350	30	1.350

Notes to Table 4–101:

(1) Input measurement point at internal node is $0.5 \times V_{CCINT}$.

(2) Output measuring point for data is V_{MEAS}.

(3) Input stimulus edge rate is 0 to V_{CCINT} in 0.5 ns (internal signal) from the driver preceding the IO buffer.

(4) The first value is for output rising edge and the second value is for output falling edge. The hyphen (-) indicates infinite resistance or disconnection.

Table 4–102 shows the reporting methodology used by the Quartus II software for minimum timing information for output pins.

Table 4–102. Reporting Methodology For Minimum Timing For Single-Ended Output Pins (Part 1 of 2)
Notes (1), (2), (3)

I/O Standard	Loading and Termination							Measurement Point
	R_{UP} Ω	R_{DN} Ω	R_S Ω	R_T Ω	V_{CCIO} (V)	VTT (V)	C_L (pF)	V_{MEAS}
3.3-V LVTTL	–	–	0	–	3.600	3.600	10	1.800
2.5-V LVTTL	–	–	0	–	2.630	2.630	10	1.200
1.8-V LVTTL	–	–	0	–	1.950	1.950	10	0.880
1.5-V LVTTL	–	–	0	–	1.600	1.600	10	0.750
3.3-V LVCMOS	–	–	0	–	3.600	3.600	10	1.800
2.5-V LVCMOS	–	–	0	–	2.630	2.630	10	1.200
1.8-V LVCMOS	–	–	0	–	1.950	1.950	10	0.880
1.5-V LVCMOS	–	–	0	–	1.600	1.600	10	0.750
3.3-V GTL	–	–	0	25	3.600	1.260	30	0.860
2.5-V GTL	–	–	0	25	2.630	1.260	30	0.860
3.3-V GTL+	–	–	0	25	3.600	1.650	30	1.120
2.5-V GTL+	–	–	0	25	2.630	1.650	30	1.120
3.3-V SSTL-3 Class II	–	–	25	25	3.600	1.750	30	1.750
3.3-V SSTL-3 Class I	–	–	25	50	3.600	1.750	30	1.750
2.5-V SSTL-2 Class II	–	–	25	25	2.630	1.390	30	1.390
2.5-V SSTL-2 Class I	–	–	25	50	2.630	1.390	30	1.390
1.8-V SSTL-18 Class II	–	–	25	25	1.950	1.040	30	1.040
1.8-V SSTL-18 Class I	–	–	25	50	1.950	1.040	30	1.040
1.5-V HSTL Class II	–	–	0	25	1.600	0.800	20	0.900
1.5-V HSTL Class I	–	–	0	50	1.600	0.800	20	0.900
1.8-V HSTL Class II	–	–	0	25	1.950	0.900	20	1.000
1.8-V HSTL Class I	–	–	0	50	1.950	0.900	20	1.000
3.3-V PCI *(4)*	–/25	25/–	0	–	3.600	1.950	10	1.026/2.214
3.3-V PCI-X 1.0 *(4)*	–/25	25/–	0	–	3.600	1.950	10	1.026/2.214
3.3-V Compact PCI *(4)*	–/25	25/–	0	–	3.600	3.600	10	1.026/2.214
3.3-V AGP 1× *(4)*	–/25	25/–	0	–	3.600	3.600	10	1.026/2.214

Table 4–102. Reporting Methodology For Minimum Timing For Single-Ended Output Pins (Part 2 of 2)
Notes (1), (2), (3)

I/O Standard	Loading and Termination							Measurement Point
	R_{UP} Ω	R_{DN} Ω	R_S Ω	R_T Ω	V_{CCIO} (V)	VTT (V)	C_L (pF)	V_{MEAS}
3.3-V CTT	–	–	25	50	3.600	1.650	30	1.650

Notes to Table 4–102:

(1) Input measurement point at internal node is $0.5 \times V_{CCINT}$.

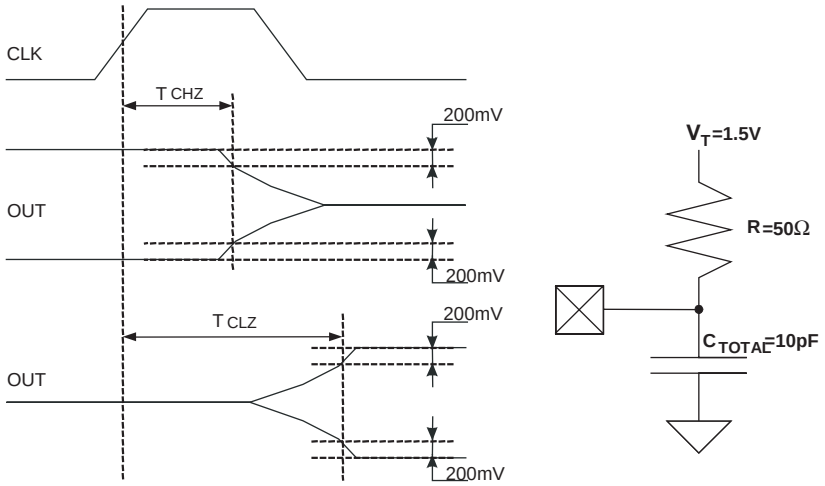
(2) Output measuring point for data is V_{MEAS}. When two values are given, the first is the measurement point on the rising edge and the other is for the falling edge.

(3) Input stimulus edge rate is 0 to V_{CCINT} in 0.5 ns (internal signal) from the driver preceding the I/O buffer.

(4) The first value is for the output rising edge and the second value is for the output falling edge. The hyphen (-) indicates infinite resistance or disconnection.

Figure 4–8 shows the measurement setup for output disable and output enable timing. The T_{CHZ} stands for clock to high Z time delay and is the same as T_{XZ}. The T_{CLZ} stands for clock to low Z (driving) time delay and is the same as T_{ZX}.

Figure 4–8. Measurement Setup for T_{XZ} and T_{ZX}



External I/O Delay Parameters

External I/O delay timing parameters for I/O standard input and output adders and programmable input and output delays are specified by speed grade independent of device density. All of the timing parameters in this section apply to both flip-chip and wire-bond packages.

Tables 4–103 and 4–104 show the input adder delays associated with column and row I/O pins. If an I/O standard is selected other than 3.3-V LVTTL or LVCMOS, add the selected delay to the external t_{INSU} and $t_{INSUPLL}$ I/O parameters shown in Tables 4–54 through 4–96.

Table 4–103. Stratix I/O Standard Column Pin Input Delay Adders

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
LVCMOS		0		0		0		0	ps
3.3-V LVTTL		0		0		0		0	ps
2.5-V LVTTL		19		19		22		26	ps
1.8-V LVTTL		221		232		266		313	ps
1.5-V LVTTL		352		369		425		500	ps
GTL		–45		–48		–55		–64	ps
GTL+		–75		–79		–91		–107	ps
3.3-V PCI		0		0		0		0	ps
3.3-V PCI-X 1.0		0		0		0		0	ps
Compact PCI		0		0		0		0	ps
AGP 1×		0		0		0		0	ps
AGP 2×		0		0		0		0	ps
CTT		120		126		144		170	ps
SSTL-3 Class I		–162		–171		–196		–231	ps
SSTL-3 Class II		–162		–171		–196		–231	ps
SSTL-2 Class I		–202		–213		–244		–287	ps
SSTL-2 Class II		–202		–213		–244		–287	ps
SSTL-18 Class I		78		81		94		110	ps
SSTL-18 Class II		78		81		94		110	ps
1.5-V HSTL Class I		–76		–80		–92		–108	ps
1.5-V HSTL Class II		–76		–80		–92		–108	ps
1.8-V HSTL Class I		–52		–55		–63		–74	ps
1.8-V HSTL Class II		–52		–55		–63		–74	ps

Table 4–104. Stratix I/O Standard Row Pin Input Delay Adders

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
LVCMOS		0		0		0		0	ps
3.3-V LVTTL		0		0		0		0	ps
2.5-V LVTTL		21		22		25		29	ps
1.8-V LVTTL		181		190		218		257	ps
1.5-V LVTTL		300		315		362		426	ps
GTL+		–152		–160		–184		–216	ps
CTT		–168		–177		–203		–239	ps
SSTL-3 Class I		–193		–203		–234		–275	ps
SSTL-3 Class II		–193		–203		–234		–275	ps
SSTL-2 Class I		–262		–276		–317		–373	ps
SSTL-2 Class II		–262		–276		–317		–373	ps
SSTL-18 Class I		–105		–111		–127		–150	ps
SSTL-18 Class II		0		0		0		0	ps
1.5-V HSTL Class I		–151		–159		–183		–215	ps
1.8-V HSTL Class I		–126		–133		–153		–179	ps
LVDS		–149		–157		–180		–212	ps
LVPECL		–149		–157		–180		–212	ps
3.3-V PCML		–65		–69		–79		–93	ps
HyperTransport		77		–81		–93		–110	ps

Tables 4–105 through 4–108 show the output adder delays associated with column and row I/O pins for both fast and slow slew rates. If an I/O standard is selected other than 3.3-V LVTTL 4mA or LVCMOS 2 mA with a fast slew rate, add the selected delay to the external t_{OUTCO}, $t_{OUTCOPLL}$, t_{XZ}, t_{ZX}, t_{XZPLL}, and t_{ZXPLL} I/O parameters shown in Table 4–55 on page 4–36 through Table 4–96 on page 4–56.

Table 4–105. Stratix I/O Standard Output Delay Adders for Fast Slew Rate on Column Pins (Part 1 of 2)

Parameter		-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		1,895		1,990		1,990		1,990	ps
	4 mA		956		1,004		1,004		1,004	ps
	8 mA		189		198		198		198	ps
	12 mA		0		0		0		0	ps
	24 mA		–157		–165		–165		–165	ps
3.3-V LVTTL	4 mA		1,895		1,990		1,990		1,990	ps
	8 mA		1,347		1,414		1,414		1,414	ps
	12 mA		636		668		668		668	ps
	16 mA		561		589		589		589	ps
	24 mA		0		0		0		0	ps
2.5-V LVTTL	2 mA		2,517		2,643		2,643		2,643	ps
	8 mA		834		875		875		875	ps
	12 mA		504		529		529		529	ps
	16 mA		194		203		203		203	ps
1.8-V LVTTL	2 mA		1,304		1,369		1,369		1,369	ps
	8 mA		960		1,008		1,008		1,008	ps
	12 mA		960		1,008		1,008		1,008	ps
1.5-V LVTTL	2 mA		6,680		7,014		7,014		7,014	ps
	4 mA		3,275		3,439		3,439		3,439	ps
	8 mA		1,589		1,668		1,668		1,668	ps
GTL			16		17		17		17	ps
GTL+			9		9		9		9	ps
3.3-V PCI			50		52		52		52	ps
3.3-V PCI-X 1.0			50		52		52		52	ps
Compact PCI			50		52		52		52	ps
AGP 1×			50		52		52		52	ps
AGP 2×			1,895		1,990		1,990		1,990	ps

Table 4–105. Stratix I/O Standard Output Delay Adders for Fast Slew Rate on Column Pins (Part 2 of 2)

Parameter	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
	Min	Max	Min	Max	Min	Max	Min	Max	
CTT		973		1,021		1,021		1,021	ps
SSTL-3 Class I		719		755		755		755	ps
SSTL-3 Class II		146		153		153		153	ps
SSTL-2 Class I		678		712		712		712	ps
SSTL-2 Class II		223		234		234		234	ps
SSTL-18 Class I		1,032		1,083		1,083		1,083	ps
SSTL-18 Class II		447		469		469		469	ps
1.5-V HSTL Class I		660		693		693		693	ps
1.5-V HSTL Class II		537		564		564		564	ps
1.8-V HSTL Class I		304		319		319		319	ps
1.8-V HSTL Class II		231		242		242		242	ps

Table 4–106. Stratix I/O Standard Output Delay Adders for Fast Slew Rate on Row Pins (Part 1 of 2)

Parameter		-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		1,518		1,594		1,594		1,594	ps
	4 mA		746		783		783		783	ps
	8 mA		96		100		100		100	ps
	12 mA		0		0		0		0	ps
3.3-V LVTTL	4 mA		1,518		1,594		1,594		1,594	ps
	8 mA		1,038		1,090		1,090		1,090	ps
	12 mA		521		547		547		547	ps
	16 mA		414		434		434		434	ps
	24 mA		0		0		0		0	ps
2.5-V LVTTL	2 mA		2,032		2,133		2,133		2,133	ps
	8 mA		699		734		734		734	ps
	12 mA		374		392		392		392	ps
	16 mA		165		173		173		173	ps
1.8-V LVTTL	2 mA		3,714		3,899		3,899		3,899	ps
	8 mA		1,055		1,107		1,107		1,107	ps
	12 mA		830		871		871		871	ps

Table 4–106. Stratix I/O Standard Output Delay Adders for Fast Slew Rate on Row Pins (Part 2 of 2)

Parameter		-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
1.5-V LVTTL	2 mA		5,460		5,733		5,733		5,733	ps
	4 mA		2,690		2,824		2,824		2,824	ps
	8 mA		1,398		1,468		1,468		1,468	ps
GTL+			6		6		6		6	ps
CTT			845		887		887		887	ps
SSTL-3 Class I			638		670		670		670	ps
SSTL-3 Class II			144		151		151		151	ps
SSTL-2 Class I			604		634		634		634	ps
SSTL-2 Class II			211		221		221		221	ps
SSTL-18 Class I			955		1,002		1,002		1,002	ps
1.5-V HSTL Class I			733		769		769		769	ps
1.8-V HSTL Class I			372		390		390		390	ps
LVDS			–196		–206		–206		–206	ps
LVPECL			–148		–156		–156		–156	ps
PCML			–147		–155		–155		–155	ps
HyperTransport technology			–93		–98		–98		–98	ps

Note to Table 4–103 through 4–106:
(1) These parameters are only available on row I/O pins.

Table 4–107. Stratix I/O Standard Output Delay Adders for Slow Slew Rate on Column Pins (Part 1 of 2)

Parameter		-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		1,822		1,913		1,913		1,913	ps
	4 mA		684		718		718		718	ps
	8 mA		233		245		245		245	ps
	12 mA		1		1		1		1	ps
	24 mA		–608		–638		–638		–638	ps

Table 4–107. Stratix I/O Standard Output Delay Adders for Slow Slew Rate on Column Pins (Part 2 of 2)

Parameter		-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
3.3-V LVTTL	4 mA		1,822		1,913		1,913		1,913	ps
	8 mA		1,586		1,665		1,665		1,665	ps
	12 mA		686		720		720		720	ps
	16 mA		630		662		662		662	ps
	24 mA		0		0		0		0	ps
2.5-V LVTTL	2 mA		2,925		3,071		3,071		3,071	ps
	8 mA		1,496		1,571		1,571		1,571	ps
	12 mA		937		984		984		984	ps
	16 mA		1,003		1,053		1,053		1,053	ps
1.8-V LVTTL	2 mA		7,101		7,456		7,456		7,456	ps
	8 mA		3,620		3,801		3,801		3,801	ps
	12 mA		3,109		3,265		3,265		3,265	ps
1.5-V LVTTL	2 mA		10,941		11,488		11,488		11,488	ps
	4 mA		7,431		7,803		7,803		7,803	ps
	8 mA		5,990		6,290		6,290		6,290	ps
GTL			–959		–1,007		–1,007		–1,007	ps
GTL+			–438		–460		–460		–460	ps
3.3-V PCI			660		693		693		693	ps
3.3-V PCI-X 1.0			660		693		693		693	ps
Compact PCI			660		693		693		693	ps
AGP 1×			660		693		693		693	ps
AGP 2×			288		303		303		303	ps
CTT			631		663		663		663	ps
SSTL-3 Class I			301		316		316		316	ps
SSTL-3 Class II			–359		–377		–377		–377	ps
SSTL-2 Class I			523		549		549		549	ps
SSTL-2 Class II			–49		–51		–51		–51	ps
SSTL-18 Class I			2,315		2,431		2,431		2,431	ps
SSTL-18 Class II			723		759		759		759	ps
1.5-V HSTL Class I			1,687		1,771		1,771		1,771	ps
1.5-V HSTL Class II			1,095		1,150		1,150		1,150	ps
1.8-V HSTL Class I			599		629		678		744	ps
1.8-V HSTL Class II			87		102		102		102	ps

Table 4–108. Stratix I/O Standard Output Delay Adders for Slow Slew Rate on Row Pins

I/O Standard		-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
LVCMOS	2 mA		1,571		1,650		1,650		1,650	ps
	4 mA		594		624		624		624	ps
	8 mA		208		218		218		218	ps
	12 mA		0		0		0		0	ps
3.3-V LVTTL	4 mA		1,571		1,650		1,650		1,650	ps
	8 mA		1,393		1,463		1,463		1,463	ps
	12 mA		596		626		626		626	ps
	16 mA		562		590		590		590	ps
2.5-V LVTTL	2 mA		2,562		2,690		2,690		2,690	ps
	8 mA		1,343		1,410		1,410		1,410	ps
	12 mA		864		907		907		907	ps
	16 mA		945		992		992		992	ps
1.8-V LVTTL	2 mA		6,306		6,621		6,621		6,621	ps
	8 mA		3,369		3,538		3,538		3,538	ps
	12 mA		2,932		3,079		3,079		3,079	ps
1.5-V LVTTL	2 mA		9,759		10,247		10,247		10,247	ps
	4 mA		6,830		7,172		7,172		7,172	ps
	8 mA		5,699		5,984		5,984		5,984	ps
GTL+			–333		–350		–350		–350	ps
CTT			591		621		621		621	ps
SSTL-3 Class I			267		280		280		280	ps
SSTL-3 Class II			–346		–363		–363		–363	ps
SSTL-2 Class I			481		505		505		505	ps
SSTL-2 Class II			–58		–61		–61		–61	ps
SSTL-18 Class I			2,207		2,317		2,317		2,317	ps
1.5-V HSTL Class I			1,966		2,064		2,064‘		2,064	ps
1.8-V HSTL Class I			1,208		1,268		1,460		1,720	ps

Tables 4–109 and 4–110 show the adder delays for the column and row IOE programmable delays. These delays are controlled with the Quartus II software logic options listed in the Parameter column.

Table 4–109. Stratix IOE Programmable Delays on Column Pins *Note (1)*

Parameter	Setting	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
Decrease input delay to internal cells	Off		3,970		4,367		5,022		5,908	ps
	Small		3,390		3,729		4,288		5,045	ps
	Medium		2,810		3,091		3,554		4,181	ps
	Large		224		235		270		318	ps
	On		224		235		270		318	ps
Decrease input delay to input register	Off		3,900		4,290		4,933		5,804	ps
	On		0		0		0		0	ps
Decrease input delay to output register	Off		1,240		1,364		1,568		1,845	ps
	On		0		0		0		0	ps
Increase delay to output pin	Off		0		0		0		0	ps
	On		397		417		417		417	ps
Increase delay to output enable pin	Off		0		0		0		0	ps
	On		338		372		427		503	ps
Increase output clock enable delay	Off		0		0		0		0	ps
	Small		540		594		683		804	ps
	Large		1,016		1,118		1,285		1,512	ps
	On		1,016		1,118		1,285		1,512	ps
Increase input clock enable delay	Off		0		0		0		0	ps
	Small		540		594		683		804	ps
	Large		1,016		1,118		1,285		1,512	ps
	On		1,016		1,118		1,285		1,512	ps
Increase output enable clock enable delay	Off		0		0		0		0	ps
	Small		540		594		683		804	ps
	Large		1,016		1,118		1,285		1,512	ps
	On		1,016		1,118		1,285		1,512	ps
Increase t_{ZX} delay to output pin	Off		0		0		0		0	ps
	On		2,199		2,309		2,309		2,309	ps

Table 4–110. Stratix IOE Programmable Delays on Row Pins *Note (1)*

Parameter	Setting	-5 Speed Grade		-6 Speed Grade		-7 Speed Grade		-8 Speed Grade		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
Decrease input delay to internal cells	Off		3,970		4,367		5,022		5,908	ps
	Small		3,390		3,729		4,288		5,045	ps
	Medium		2,810		3,091		3,554		4,181	ps
	Large		173		181		208		245	ps
	On		173		181		208		245	ps
Decrease input delay to input register	Off		3,900		4,290		4,933		5,804	ps
	On		0		0		0		0	ps
Decrease input delay to output register	Off		1,240		1,364		1,568		1,845	ps
	On		0		0		0		0	ps
Increase delay to output pin	Off		0		0		0		0	ps
	On		397		417		417		417	ps
Increase delay to output enable pin	Off		0		0		0		0	ps
	On		348		383		441		518	ps
Increase output clock enable delay	Off		0		0		0		0	ps
	Small		180		198		227		267	ps
	Large		260		286		328		386	ps
	On		260		286		328		386	ps
Increase input clock enable delay	Off		0		0		0		0	ps
	Small		180		198		227		267	ps
	Large		260		286		328		386	ps
	On		260		286		328		386	ps
Increase output enable clock enable delay	Off		0		0		0		0	ps
	Small		540		594		683		804	ps
	Large		1,016		1,118		1,285		1,512	ps
	On		1,016		1,118		1,285		1,512	ps
Increase t_{ZX} delay to output pin	Off		0		0		0		0	ps
	On		1,993		2,092		2,092		2,092	ps

Note to Table 4–109 and Table 4–110:

(1) The delay chain delays vary for different device densities. These timing values only apply to EP1S30 and EP1S40 devices. Reference the timing information reported by the Quartus II software for other devices.

The scaling factors for column output pin timing in Tables 4–111 to 4–113 are shown in units of time per pF unit of capacitance (ps / pF). Add this delay to the t_{CO} or combinatorial timing path for output or bidirectional pins in addition to the I / O adder delays shown in Tables 4–103 through 4–108 and the IOE programmable delays in Tables 4–109 and 4–110.

Table 4–111. Output Delay Adder for Loading on LVTTL/LVCMOS Output Buffers *Note (1)*

Conditions		Output Pin Adder Delay (ps/pF)				
Parameter	**Value**	**3.3-V LVTTL**	**2.5-V LVTTL**	**1.8-V LVTTL**	**1.5-V LVTTL**	**LVCMOS**
Drive Strength	24mA	15	–	–	-	8
	16mA	25	18	–	–	–
	12mA	30	25	25	–	15
	8mA	50	35	40	35	20
	4mA	60	–	–	80	30
	2mA	–	75	120	160	60

Note to Table 4–111:
(1) The timing information in this table is preliminary.

Table 4–112. Output Delay Adder for Loading on SSTL/HSTL Output Buffers *Note (1)*

Conditions	Output Pin Adder Delay (ps/pF)			
	SSTL-3	**SSTL-2**	**SSTL-1.8**	**1.5-V HSTL**
Class I	25	25	25	25
Class II	25	20	25	20

Note to Table 4–112:
(1) The timing information in this table is preliminary.

Table 4–113. Output Delay Adder for Loading on GTL+/GTL/CTT/PCI Output Buffers *Note (1)*

Conditions		Output Pin Adder Delay (ps/pF)				
Parameter	**Value**	**GTL+**	**GTL**	**CTT**	**PCI**	**AGP**
VCCIO Voltage Level	3.3V	18	18	25	20	20
	2.5V	15	18	-	-	-

Note to Table 4–113:
(1) The timing information in this table is preliminary.

Maximum Input & Output Clock Rates

Tables 4–114 through 4–119 show the maximum input clock rate for column and row pins in Stratix devices.

Table 4–114. Stratix Maximum Input Clock Rate for CLK[7..4] & CLK[15..12] Pins in Flip-Chip Packages (Part 1 of 2)

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	422	422	390	390	MHz
2.5 V	422	422	390	390	MHz
1.8 V	422	422	390	390	MHz
1.5 V	422	422	390	390	MHz
LVCMOS	422	422	390	390	MHz
GTL	300	250	200	200	MHz
GTL+	300	250	200	200	MHz
SSTL-3 Class I	400	350	300	300	MHz
SSTL-3 Class II	400	350	300	300	MHz
SSTL-2 Class I	400	350	300	300	MHz
SSTL-2 Class II	400	350	300	300	MHz
SSTL-18 Class I	400	350	300	300	MHz
SSTL-18 Class II	400	350	300	300	MHz
1.5-V HSTL Class I	400	350	300	300	MHz
1.5-V HSTL Class II	400	350	300	300	MHz
1.8-V HSTL Class I	400	350	300	300	MHz
1.8-V HSTL Class II	400	350	300	300	MHz
3.3-V PCI	422	422	390	390	MHz
3.3-V PCI-X 1.0	422	422	390	390	MHz
Compact PCI	422	422	390	390	MHz
AGP 1×	422	422	390	390	MHz
AGP 2×	422	422	390	390	MHz
CTT	300	250	200	200	MHz
Differential 1.5-V HSTL C1	400	350	300	300	MHz
LVPECL *(1)*	645	645	622	622	MHz
PCML *(1)*	300	275	275	275	MHz

Table 4–114. Stratix Maximum Input Clock Rate for CLK[7..4] & CLK[15..12] Pins in Flip-Chip Packages (Part 2 of 2)

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVDS *(1)*	645	645	622	622	MHz
HyperTransport technology *(1)*	500	500	450	450	MHz

Table 4–115. Stratix Maximum Input Clock Rate for CLK[0, 2, 9, 11] Pins & FPLL[10..7]CLK Pins in Flip-Chip Packages

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	422	422	390	390	MHz
2.5 V	422	422	390	390	MHz
1.8 V	422	422	390	390	MHz
1.5 V	422	422	390	390	MHz
LVCMOS	422	422	390	390	MHz
GTL+	300	250	200	200	MHz
SSTL-3 Class I	400	350	300	300	MHz
SSTL-3 Class II	400	350	300	300	MHz
SSTL-2 Class I	400	350	300	300	MHz
SSTL-2 Class II	400	350	300	300	MHz
SSTL-18 Class I	400	350	300	300	MHz
SSTL-18 Class II	400	350	300	300	MHz
1.5-V HSTL Class I	400	350	300	300	MHz
1.8-V HSTL Class I	400	350	300	300	MHz
CTT	300	250	200	200	MHz
Differential 1.5-V HSTL C1	400	350	300	300	MHz
LVPECL *(1)*	717	717	640	640	MHz
PCML *(1)*	400	375	350	350	MHz
LVDS *(1)*	717	717	640	640	MHz
HyperTransport technology *(1)*	717	717	640	640	MHz

Table 4–116. Stratix Maximum Input Clock Rate for CLK[1, 3, 8, 10] Pins in Flip-Chip Packages

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	422	422	390	390	MHz
2.5 V	422	422	390	390	MHz
1.8 V	422	422	390	390	MHz
1.5 V	422	422	390	390	MHz
LVCMOS	422	422	390	390	MHz
GTL+	300	250	200	200	MHz
SSTL-3 Class I	400	350	300	300	MHz
SSTL-3 Class II	400	350	300	300	MHz
SSTL-2 Class I	400	350	300	300	MHz
SSTL-2 Class II	400	350	300	300	MHz
SSTL-18 Class I	400	350	300	300	MHz
SSTL-18 Class II	400	350	300	300	MHz
1.5-V HSTL Class I	400	350	300	300	MHz
1.8-V HSTL Class I	400	350	300	300	MHz
CTT	300	250	200	200	MHz
Differential 1.5-V HSTL C1	400	350	300	300	MHz
LVPECL *(1)*	645	645	640	640	MHz
PCML *(1)*	300	275	275	275	MHz
LVDS *(1)*	645	645	640	640	MHz
HyperTransport technology *(1)*	500	500	450	450	MHz

Table 4–117. Stratix Maximum Input Clock Rate for CLK[7..4] & CLK[15..12] Pins in Wire-Bond Packages (Part 1 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	422	390	390	MHz
2.5 V	422	390	390	MHz
1.8 V	422	390	390	MHz
1.5 V	422	390	390	MHz
LVCMOS	422	390	390	MHz
GTL	250	200	200	MHz

Table 4–117. Stratix Maximum Input Clock Rate for CLK[7..4] & CLK[15..12] Pins in Wire-Bond Packages (Part 2 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
GTL+	250	200	200	MHz
SSTL-3 Class I	300	250	250	MHz
SSTL-3 Class II	300	250	250	MHz
SSTL-2 Class I	300	250	250	MHz
SSTL-2 Class II	300	250	250	MHz
SSTL-18 Class I	300	250	250	MHz
SSTL-18 Class II	300	250	250	MHz
1.5-V HSTL Class I	300	180	180	MHz
1.5-V HSTL Class II	300	180	180	MHz
1.8-V HSTL Class I	300	180	180	MHz
1.8-V HSTL Class II	300	180	180	MHz
3.3-V PCI	422	390	390	MHz
3.3-V PCI-X 1.0	422	390	390	MHz
Compact PCI	422	390	390	MHz
AGP 1×	422	390	390	MHz
AGP 2×	422	390	390	MHz
CTT	250	180	180	MHz
Differential 1.5-V HSTL C1	300	180	180	MHz
LVPECL *(1)*	422	400	400	MHz
PCML *(1)*	215	200	200	MHz
LVDS *(1)*	422	400	400	MHz
HyperTransport technology *(1)*	422	400	400	MHz

Table 4–118. Stratix Maximum Input Clock Rate for CLK[0, 2, 9, 11] Pins & FPLL[10..7]CLK Pins in Wire-Bond Packages (Part 1 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	422	390	390	MHz
2.5 V	422	390	390	MHz
1.8 V	422	390	390	MHz
1.5 V	422	390	390	MHz

Table 4–118. Stratix Maximum Input Clock Rate for CLK[0, 2, 9, 11] Pins & FPLL[10..7]CLK Pins in Wire-Bond Packages (Part 2 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVCMOS	422	390	390	MHz
GTL+	250	200	200	MHz
SSTL-3 Class I	350	300	300	MHz
SSTL-3 Class II	350	300	300	MHz
SSTL-2 Class I	350	300	300	MHz
SSTL-2 Class II	350	300	300	MHz
SSTL-18 Class I	350	300	300	MHz
SSTL-18 Class II	350	300	300	MHz
1.5-V HSTL Class I	350	300	300	MHz
1.8-V HSTL Class I	350	300	300	MHz
CTT	250	200	200	MHz
Differential 1.5-V HSTL C1	350	300	300	MHz
LVPECL *(1)*	717	640	640	MHz
PCML *(1)*	375	350	350	MHz
LVDS *(1)*	717	640	640	MHz
HyperTransport technology *(1)*	717	640	640	MHz

Table 4–119. Stratix Maximum Input Clock Rate for CLK[1, 3, 8, 10] Pins in Wire-Bond Packages (Part 1 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	422	390	390	MHz
2.5 V	422	390	390	MHz
1.8 V	422	390	390	MHz
1.5 V	422	390	390	MHz
LVCMOS	422	390	390	MHz
GTL+	250	200	200	MHz
SSTL-3 Class I	350	300	300	MHz
SSTL-3 Class II	350	300	300	MHz
SSTL-2 Class I	350	300	300	MHz
SSTL-2 Class II	350	300	300	MHz

Table 4–119. Stratix Maximum Input Clock Rate for CLK[1, 3, 8, 10] Pins in Wire-Bond Packages (Part 2 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
SSTL-18 Class I	350	300	300	MHz
SSTL-18 Class II	350	300	300	MHz
1.5-V HSTL Class I	350	300	300	MHz
1.8-V HSTL Class I	350	300	300	MHz
CTT	250	200	200	MHz
Differential 1.5-V HSTL C1	350	300	300	MHz
LVPECL *(1)*	645	622	622	MHz
PCML *(1)*	275	275	275	MHz
LVDS *(1)*	645	622	622	MHz
HyperTransport technology *(1)*	500	450	450	MHz

Note to Tables 4–114 through 4–119:

(1) These parameters are only available on row I/O pins.

Tables 4–120 through 4–123 show the maximum output clock rate for column and row pins in Stratix devices.

Table 4–120. Stratix Maximum Output Clock Rate for PLL[5, 6, 11, 12] Pins in Flip-Chip Packages (Part 1 of 2)

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	350	300	250	250	MHz
2.5 V	350	300	300	300	MHz
1.8 V	250	250	250	250	MHz
1.5 V	225	200	200	200	MHz
LVCMOS	350	300	250	250	MHz
GTL	200	167	125	125	MHz
GTL+	200	167	125	125	MHz
SSTL-3 Class I	200	167	167	133	MHz
SSTL-3 Class II	200	167	167	133	MHz
SSTL-2 Class I *(3)*	200	200	167	167	MHz
SSTL-2 Class I *(4)*	200	200	167	167	MHz
SSTL-2 Class I *(5)*	150	134	134	134	MHz

Table 4–120. Stratix Maximum Output Clock Rate for PLL[5, 6, 11, 12] Pins in Flip-Chip Packages (Part 2 of 2)

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
SSTL-2 Class II *(3)*	200	200	167	167	MHz
SSTL-2 Class II *(4)*	200	200	167	167	MHz
SSTL-2 Class II *(5)*	150	134	134	134	MHz
SSTL-18 Class I	150	133	133	133	MHz
SSTL-18 Class II	150	133	133	133	MHz
1.5-V HSTL Class I	250	225	200	200	MHz
1.5-V HSTL Class II	225	200	200	200	MHz
1.8-V HSTL Class I	250	225	200	200	MHz
1.8-V HSTL Class II	225	200	200	200	MHz
3.3-V PCI	350	300	250	250	MHz
3.3-V PCI-X 1.0	350	300	250	250	MHz
Compact PCI	350	300	250	250	MHz
AGP 1×	350	300	250	250	MHz
AGP 2×	350	300	250	250	MHz
CTT	200	200	200	200	MHz
Differential 1.5-V HSTL C1	225	200	200	200	MHz
Differential 1.8-V HSTL Class I	250	225	200	200	MHz
Differential 1.8-V HSTL Class II	225	200	200	200	MHz
Differential SSTL-2 *(6)*	200	200	167	167	MHz
LVPECL *(2)*	500	500	500	500	MHz
PCML *(2)*	350	350	350	350	MHz
LVDS *(2)*	500	500	500	500	MHz
HyperTransport technology *(2)*	350	350	350	350	MHz

Table 4–121. Stratix Maximum Output Clock Rate (Using I/O Pins) for PLL[1, 2, 3, 4] Pins in Flip-Chip Packages

I/O Standard	-5 Speed Grade	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	400	350	300	300	MHz
2.5 V	400	350	300	300	MHz
1.8 V	400	350	300	300	MHz
1.5 V	350	300	300	300	MHz
LVCMOS	400	350	300	300	MHz
GTL	200	167	125	125	MHz
GTL+	200	167	125	125	MHz
SSTL-3 Class I	167	150	133	133	MHz
SSTL-3 Class II	167	150	133	133	MHz
SSTL-2 Class I	150	133	133	133	MHz
SSTL-2 Class II	150	133	133	133	MHz
SSTL-18 Class I	150	133	133	133	MHz
SSTL-18 Class II	150	133	133	133	MHz
1.5-V HSTL Class I	250	225	200	200	MHz
1.5-V HSTL Class II	225	225	200	200	MHz
1.8-V HSTL Class I	250	225	200	200	MHz
1.8-V HSTL Class II	225	225	200	200	MHz
3.3-V PCI	250	225	200	200	MHz
3.3-V PCI-X 1.0	225	225	200	200	MHz
Compact PCI	400	350	300	300	MHz
AGP 1×	400	350	300	300	MHz
AGP 2×	400	350	300	300	MHz
CTT	300	250	200	200	MHz
LVPECL *(2)*	717	717	500	500	MHz
PCML *(2)*	420	420	420	420	MHz
LVDS *(2)*	717	717	500	500	MHz
HyperTransport technology *(2)*	420	420	420	420	MHz

Table 4–122. Stratix Maximum Output Clock Rate for PLL[5, 6, 11, 12] Pins in Wire-Bond Packages (Part 1 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	175	150	150	MHz
2.5 V	175	150	150	MHz
1.8 V	175	150	150	MHz
1.5 V	175	150	150	MHz
LVCMOS	175	150	150	MHz
GTL	125	100	100	MHz
GTL+	125	100	100	MHz
SSTL-3 Class I	110	90	90	MHz
SSTL-3 Class II	133	125	125	MHz
SSTL-2 Class I	166	133	133	MHz
SSTL-2 Class II	133	100	100	MHz
SSTL-18 Class I	110	100	100	MHz
SSTL-18 Class II	110	100	100	MHz
1.5-V HSTL Class I	167	167	167	MHz
1.5-V HSTL Class II	167	133	133	MHz
1.8-V HSTL Class I	167	167	167	MHz
1.8-V HSTL Class II	167	133	133	MHz
3.3-V PCI	167	167	167	MHz
3.3-V PCI-X 1.0	167	133	133	MHz
Compact PCI	175	150	150	MHz
AGP 1×	175	150	150	MHz
AGP 2×	175	150	150	MHz
CTT	125	100	100	MHz
Differential 1.5-V HSTL C1	167	133	133	MHz
Differential 1.8-V HSTL Class I	167	167	167	MHz
Differential 1.8-V HSTL Class II	167	133	133	MHz
Differential SSTL-2 *(1)*	110	100	100	MHz
LVPECL *(2)*	311	275	275	MHz
PCML *(2)*	250	200	200	MHz

Table 4–122. Stratix Maximum Output Clock Rate for PLL[5, 6, 11, 12] Pins in Wire-Bond Packages (Part 2 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVDS *(2)*	311	275	275	MHz
HyperTransport technology *(2)*	311	275	275	MHz

Table 4–123. Stratix Maximum Output Clock Rate (Using I/O Pins) for PLL[1, 2, 3, 4] Pins in Wire-Bond Packages (Part 1 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVTTL	200	175	175	MHz
2.5 V	200	175	175	MHz
1.8 V	200	175	175	MHz
1.5 V	200	175	175	MHz
LVCMOS	200	175	175	MHz
GTL	125	100	100	MHz
GTL+	125	100	100	MHz
SSTL-3 Class I	110	90	90	MHz
SSTL-3 Class II	150	133	133	MHz
SSTL-2 Class I	90	80	80	MHz
SSTL-2 Class II	110	100	100	MHz
SSTL-18 Class I	110	100	100	MHz
SSTL-18 Class II	110	100	100	MHz
1.5-V HSTL Class I	225	200	200	MHz
1.5-V HSTL Class II	200	167	167	MHz
1.8-V HSTL Class I	225	200	200	MHz
1.8-V HSTL Class II	200	167	167	MHz
3.3-V PCI	200	175	175	MHz
3.3-V PCI-X 1.0	200	175	175	MHz
Compact PCI	200	175	175	MHz
AGP 1×	200	175	175	MHz
AGP 2×	200	175	175	MHz
CTT	125	100	100	MHz
LVPECL *(2)*	311	270	270	MHz
PCML *(2)*	400	311	311	MHz

Table 4–123. Stratix Maximum Output Clock Rate (Using I/O Pins) for PLL[1, 2, 3, 4] Pins in Wire-Bond Packages (Part 2 of 2)

I/O Standard	-6 Speed Grade	-7 Speed Grade	-8 Speed Grade	Unit
LVDS *(2)*	400	311	311	MHz
HyperTransport technology *(2)*	420	400	400	MHz

Notes to Tables 4–120 through 4–123:

(1) Differential SSTL-2 outputs are only available on column clock pins.

(2) These parameters are only available on row I/O pins.

(3) SSTL-2 in maximum drive strength condition. See Table 4–101 on page 4–62 for more information on exact loading conditions for each I/O standard.

(4) SSTL-2 in minimum drive strength with ≤10pF output load condition.

(5) SSTL-2 in minimum drive strength with > 10pF output load condition.

(6) Differential SSTL-2 outputs are only supported on column clock pins.

High-Speed I/O Specification

Table 4–124 provides high-speed timing specifications definitions.

Table 4–124. High-Speed Timing Specifications & Terminology

High-Speed Timing Specification	Terminology
t_C	High-speed receiver/transmitter input and output clock period.
f_{HSCLK}	High-speed receiver/transmitter input and output clock frequency.
t_{RISE}	Low-to-high transmission time.
t_{FALL}	High-to-low transmission time.
Timing unit interval (TUI)	The timing budget allowed for skew, propagation delays, and data sampling window. (TUI = 1/(Receiver Input Clock Frequency × Multiplication Factor) = t_C/w).
f_{HSDR}	Maximum LVDS data transfer rate (f_{HSDR} = 1/TUI).
Channel-to-channel skew (TCCS)	The timing difference between the fastest and slowest output edges, including t_{CO} variation and clock skew. The clock is included in the TCCS measurement.
Sampling window (SW)	The period of time during which the data must be valid to be captured correctly. The setup and hold times determine the ideal strobe position within the sampling window. SW = t_{SW} (max) – t_{SW} (min).
Input jitter (peak-to-peak)	Peak-to-peak input jitter on high-speed PLLs.
Output jitter (peak-to-peak)	Peak-to-peak output jitter on high-speed PLLs.
t_{DUTY}	Duty cycle on high-speed transmitter output clock.
t_{LOCK}	Lock time for high-speed transmitter and receiver PLLs.
J	Deserialization factor (width of internal data bus).
W	PLL multiplication factor.

Tables 4–125 and 4–126 show the high-speed I/O timing for Stratix devices.

Table 4–125. High-Speed I/O Specifications for Flip-Chip Packages (Part 1 of 4) *Notes (1), (2)*

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSCLK} (Clock frequency) (LVDS, LVPECL, HyperTransport technology) $f_{HSCLK} = f_{HSDR}$ / W	*W* = 4 to 30 (Serdes used)	10		210	10		210	10		156	10		115.5	MHz
	W = 2 (Serdes bypass)	50		231	50		231	50		231	50		231	MHz
	W = 2 (Serdes used)	150		420	150		420	150		312	150		231	MHz
	W = 1 (Serdes bypass)	100		462	100		462	100		462	100		462	MHz
	W = 1 (Serdes used)	300		717	300		717	300		624	300		462	MHz
f_{HSDR} Device operation (LVDS, LVPECL, HyperTransport technology)	*J* = 10	300		840	300		840	300		640	300		462	Mbps
	J = 8	300		840	300		840	300		640	300		462	Mbps
	J = 7	300		840	300		840	300		640	300		462	Mbps
	J = 4	300		840	300		840	300		640	300		462	Mbps
	J = 2	100		462	100		462	100		640	100		462	Mbps
	J = 1 (LVDS and LVPECL only)	100		462	100		462	100		640	100		462	Mbps

Table 4–125. High-Speed I/O Specifications for Flip-Chip Packages (Part 2 of 4) Notes (1), (2)

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSCLK} (Clock frequency) (PCML) $f_{HSCLK} = f_{HSDR}$ / W	*W* = 4 to 30 (Serdes used)	10		100	10		100	10		77.75	10		77.75	MHz
	W = 2 (Serdes bypass)	50		200	50		200	50		150	50		150	MHz
	W = 2 (Serdes used)	150		200	150		200	150		155.5	150		155.5	MHz
	W = 1 (Serdes bypass)	100		250	100		250	100		200	100		200	MHz
	W = 1 (Serdes used)	300		400	300		400	300		311	300		311	MHz
f_{HSDR} Device operation (PCML)	*J* = 10	300		400	300		400	300		311	300		311	Mbps
	J = 8	300		400	300		400	300		311	300		311	Mbps
	J = 7	300		400	300		400	300		311	300		311	Mbps
	J = 4	300		400	300		400	300		311	300		311	Mbps
	J = 2	100		400	100		400	100		300	100		300	Mbps
	J = 1	100		250	100		250	100		200	100		200	Mbps
TCCS	All			200			200			300			300	ps

Table 4–125. High-Speed I/O Specifications for Flip-Chip Packages (Part 3 of 4) Notes (1), (2)

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
SW	PCML (*J* = 4, 7, 8, 10)	750			750			800			800			ps
	PCML (*J* = 2)	900			900			1,200			1,200			ps
	PCML (*J* = 1)	1,500			1,500			1,700			1,700			ps
	LVDS and LVPECL (*J* = 1)	500			500			550			550			ps
	LVDS, LVPECL, HyperTransport technology (*J* = 2 through 10)	440			440			500			500			ps
Input jitter tolerance (peak-to-peak)	All			250			250			250			250	ps
Output jitter (peak-to-peak)	All			160			160			200			200	ps
Output t_{RISE}	LVDS	80	110	120	80	110	120	80	110	120	80	110	120	ps
	HyperTransport technology	110	170	200	110	170	200	120	170	200	120	170	200	ps
	LVPECL	90	130	150	90	130	150	100	135	150	100	135	150	ps
	PCML	80	110	135	80	110	135	80	110	135	80	110	135	ps
Output t_{FALL}	LVDS	80	110	120	80	110	120	80	110	120	80	110	120	ps
	HyperTransport technology	110	170	200	110	170	200	110	170	200	110	170	200	ps
	LVPECL	90	130	160	90	130	160	100	135	160	100	135	160	ps
	PCML	105	140	175	105	140	175	110	145	175	110	145	175	ps

Table 4–125. High-Speed I/O Specifications for Flip-Chip Packages (Part 4 of 4) *Notes (1), (2)*

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t_{DUTY}	LVDS (*J* = 2 through 10)	47.5	50	52.5	47.5	50	52.5	47.5	50	52.5	47.5	50	52.5	%
	LVDS (*J* =1) and LVPECL, PCML, HyperTransport technology	45	50	55	45	50	55	45	50	55	45	50	55	%
t_{LOCK}	All			100			100			100			100	µs

Notes to Table 4–125:

(1) When J = 4, 7, 8, and 10, the SERDES block is used.

(2) When J = 2 or J = 1, the SERDES is bypassed.

Table 4–126. High-Speed I/O Specifications for Wire-Bond Packages (Part 1 of 2)

Symbol	Conditions	-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSCLK} (Clock frequency) (LVDS,LVPECL, HyperTransport technology) $f_{HSCLK} = f_{HSDR} / W$	*W* = 4 to 30 (Serdes used)	10		156	10		115.5	10		115.5	MHz
	W = 2 (Serdes bypass)	50		231	50		231	50		231	MHz
	W = 2 (Serdes used)	150		312	150		231	150		231	MHz
	W = 1 (Serdes bypass)	100		311	100		270	100		270	MHz
	W = 1 (Serdes used)	300		624	300		462	300		462	MHz
f_{HSDR} Device operation, (LVDS,LVPECL, HyperTransport technology)	J = 10	300		624	300		462	300		462	Mbps
	J = 8	300		624	300		462	300		462	Mbps
	J = 7	300		624	300		462	300		462	Mbps
	J = 4	300		624	300		462	300		462	Mbps
	J = 2	100		462	100		462	100		462	Mbps
	J = 1 (LVDS and LVPECL only)	100		311	100		270	100		270	Mbps
f_{HSCLK} (Clock frequency) (PCML) $f_{HSCLK} = f_{HSDR} / W$	*W* = 4 to 30 (Serdes used)	10		77.75							MHz
	W = 2 (Serdes bypass)	50		150	50		77.5	50		77.5	MHz
	W = 2 (Serdes used)	150		155.5							MHz
	W = 1 (Serdes bypass)	100		200	100		155	100		155	MHz
	W = 1 (Serdes used)	300		311							MHz
Device operation, f_{HSDR} (PCML)	J = 10	300		311							Mbps
	J = 8	300		311							Mbps
	J = 7	300		311							Mbps
	J = 4	300		311							Mbps
	J = 2	100		300	100		155	100		155	Mbps
	J = 1	100		200	100		155	100		155	Mbps
TCCS	All			400			400			400	ps

Table 4–126. High-Speed I/O Specifications for Wire-Bond Packages (Part 2 of 2)

Symbol	Conditions	-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
SW	PCML (J = 4, 7, 8, 10) only	800			800			800			ps
	PCML (J = 2) only	1,200			1,200			1,200			ps
	PCML (J = 1) only	1,700			1,700			1,700			ps
	LVDS and LVPECL (J = 1) only	550			550			550			ps
	LVDS, LVPECL, HyperTransport technology (J = 2 through 10) only	500			500			500			ps
Input jitter tolerance (peak-to-peak)	All			250			250			250	ps
Output jitter (peak-to-peak)	All			200			200			200	ps
Output t_{RISE}	LVDS	80	110	120	80	110	120	80	110	120	ps
	HyperTransport technology	120	170	200	120	170	200	120	170	200	ps
	LVPECL	100	135	150	100	135	150	100	135	150	ps
	PCML	80	110	135	80	110	135	80	110	135	ps
Output t_{FALL}	LVDS	80	110	120	80	110	120	80	110	120	ps
	HyperTransport	110	170	200	110	170	200	110	170	200	ps
	LVPECL	100	135	160	100	135	160	100	135	160	ps
	PCML	110	145	175	110	145	175	110	145	175	ps
t_{DUTY}	LVDS (J = 2 through10) only	47.5	50	52.5	47.5	50	52.5	47.5	50	52.5	%
	LVDS (J =1) and LVPECL, PCML, HyperTransport technology	45	50	55	45	50	55	45	50	55	%
t_{LOCK}	All			100			100			100	µs

PLL Specifications

Tables 4–127 through 4–129 describe the Stratix device enhanced PLL specifications.

Table 4–127. Enhanced PLL Specifications for -5 Speed Grades (Part 1 of 2)

Symbol	Parameter	Min	Typ	Max	Unit
f_{IN}	Input clock frequency	3 *(1)*, *(2)*		684	MHz
f_{INPFD}	Input frequency to PFD	3		420	MHz
f_{INDUTY}	Input clock duty cycle	40		60	%
$f_{EINDUTY}$	External feedback clock input duty cycle	40		60	%
$t_{INJITTER}$	Input clock period jitter			±200 *(3)*	ps
$t_{EINJITTER}$	External feedback clock period jitter			±200 *(3)*	ps
t_{FCOMP}	External feedback clock compensation time *(4)*			6	ns
f_{OUT}	Output frequency for internal global or regional clock	0.3		500	MHz
f_{OUT_EXT}	Output frequency for external clock *(3)*	0.3		526	MHz
$t_{OUTDUTY}$	Duty cycle for external clock output (when set to 50%)	45		55	%
t_{JITTER}	Period jitter for external clock output *(6)*			±100 ps for >200-MHz `outclk` ±20 mUI for <200-MHz `outclk`	ps or mUI
$t_{CONFIG5,6}$	Time required to reconfigure the scan chains for PLLs 5 and 6			$289/f_{SCANCLK}$	
$t_{CONFIG11,12}$	Time required to reconfigure the scan chains for PLLs 11 and 12			$193/f_{SCANCLK}$	
$t_{SCANCLK}$	`scanclk` frequency *(5)*			22	MHz
t_{DLOCK}	Time required to lock dynamically (after switchover or reconfiguring any non-post-scale counters/delays) *(7)*			100	µs
t_{LOCK}	Time required to lock from end of device configuration	10		400	µs
f_{VCO}	PLL internal VCO operating range	300		800 *(8)*	MHz
t_{LSKEW}	Clock skew between two external clock outputs driven by the same counter		±50		ps

Table 4–127. Enhanced PLL Specifications for -5 Speed Grades (Part 2 of 2)

Symbol	Parameter	Min	Typ	Max	Unit
t_{SKEW}	Clock skew between two external clock outputs driven by the different counters with the same settings		±75		ps
f_{SS}	Spread spectrum modulation frequency	30		150	kHz
% spread	Percentage spread for spread spectrum frequency *(10)*	0.4	0.5	0.6	%
t_{ARESET}	Minimum pulse width on `areset` signal	10			ns
$t_{ARESET_RECONFIG}$	Minimum pulse width on the `areset` signal when using PLL reconfiguration. Reset the PLL after `scandataout` goes high.	500			ns

Table 4–128. Enhanced PLL Specifications for -6 Speed Grades (Part 1 of 2)

Symbol	Parameter	Min	Typ	Max	Unit
f_{IN}	Input clock frequency	3 *(1)*, *(2)*		650	MHz
f_{INPFD}	Input frequency to PFD	3		420	MHz
f_{INDUTY}	Input clock duty cycle	40		60	%
$f_{EINDUTY}$	External feedback clock input duty cycle	40		60	%
$t_{INJITTER}$	Input clock period jitter			±200 *(3)*	ps
$t_{EINJITTER}$	External feedback clock period jitter			±200 *(3)*	ps
t_{FCOMP}	External feedback clock compensation time *(4)*			6	ns
f_{OUT}	Output frequency for internal global or regional clock	0.3		450	MHz
f_{OUT_EXT}	Output frequency for external clock *(3)*	0.3		500	MHz
$t_{OUTDUTY}$	Duty cycle for external clock output (when set to 50%)	45		55	%
t_{JITTER}	Period jitter for external clock output *(6)*			±100 ps for >200-MHz `outclk` ±20 mUI for <200-MHz `outclk`	ps or mUI
$t_{CONFIG5,6}$	Time required to reconfigure the scan chains for PLLs 5 and 6			$289/f_{SCANCLK}$	
$t_{CONFIG11,12}$	Time required to reconfigure the scan chains for PLLs 11 and 12			$193/f_{SCANCLK}$	

Table 4–128. Enhanced PLL Specifications for -6 Speed Grades (Part 2 of 2)

Symbol	Parameter	Min	Typ	Max	Unit
$t_{SCANCLK}$	`scanclk` frequency *(5)*			22	MHz
t_{DLOCK}	Time required to lock dynamically (after switchover or reconfiguring any non-post-scale counters/delays) *(7)* *(11)*	*(9)*		100	µs
t_{LOCK}	Time required to lock from end of device configuration *(11)*	10		400	µs
f_{VCO}	PLL internal VCO operating range	300		800 *(8)*	MHz
t_{LSKEW}	Clock skew between two external clock outputs driven by the same counter		±50		ps
t_{SKEW}	Clock skew between two external clock outputs driven by the different counters with the same settings		±75		ps
f_{SS}	Spread spectrum modulation frequency	30		150	kHz
% spread	Percentage spread for spread spectrum frequency *(10)*	0.4	0.5	0.6	%
t_{ARESET}	Minimum pulse width on `areset` signal	10			ns

Table 4–129. Enhanced PLL Specifications for -7 Speed Grade (Part 1 of 2)

Symbol	Parameter	Min	Typ	Max	Unit
f_{IN}	Input clock frequency	3 *(1)*, *(2)*		565	MHz
f_{INPFD}	Input frequency to PFD	3		420	MHz
f_{INDUTY}	Input clock duty cycle	40		60	%
$f_{EINDUTY}$	External feedback clock input duty cycle	40		60	%
$t_{INJITTER}$	Input clock period jitter			±200 *(3)*	ps
$t_{EINJITTER}$	External feedback clock period jitter			±200 *(3)*	ps
t_{FCOMP}	External feedback clock compensation time *(4)*			6	ns
f_{OUT}	Output frequency for internal global or regional clock	0.3		420	MHz
f_{OUT_EXT}	Output frequency for external clock *(3)*	0.3		434	MHz

Table 4–129. Enhanced PLL Specifications for -7 Speed Grade (Part 2 of 2)

Symbol	Parameter	Min	Typ	Max	Unit
$t_{OUTDUTY}$	Duty cycle for external clock output (when set to 50%)	45		55	%
t_{JITTER}	Period jitter for external clock output *(6)*			±100 ps for >200-MHz `outclk` ±20 mUI for <200-MHz `outclk`	ps or mUI
$t_{CONFIG5,6}$	Time required to reconfigure the scan chains for PLLs 5 and 6			$289/f_{SCANCLK}$	
$t_{CONFIG11,12}$	Time required to reconfigure the scan chains for PLLs 11 and 12			$193/f_{SCANCLK}$	
$t_{SCANCLK}$	`scanclk` frequency *(5)*			22	MHz
t_{DLOCK}	Time required to lock dynamically (after switchover or reconfiguring any non-post-scale counters/delays) *(7)* *(11)*	*(9)*		100	µs
t_{LOCK}	Time required to lock from end of device configuration *(11)*	10		400	µs
f_{VCO}	PLL internal VCO operating range	300		600 *(8)*	MHz
t_{LSKEW}	Clock skew between two external clock outputs driven by the same counter		±50		ps
t_{SKEW}	Clock skew between two external clock outputs driven by the different counters with the same settings		±75		ps
f_{SS}	Spread spectrum modulation frequency	30		150	kHz
% spread	Percentage spread for spread spectrum frequency *(10)*	0.5		0.6	%
t_{ARESET}	Minimum pulse width on `areset` signal	10			ns

Table 4–130. Enhanced PLL Specifications for -8 Speed Grade (Part 1 of 3)

Symbol	Parameter	Min	Typ	Max	Unit
f_{IN}	Input clock frequency	3 *(1)*, *(2)*		480	MHz
f_{INPFD}	Input frequency to PFD	3		420	MHz
f_{INDUTY}	Input clock duty cycle	40		60	%
$f_{EINDUTY}$	External feedback clock input duty cycle	40		60	%
$t_{INJITTER}$	Input clock period jitter			±200 *(3)*	ps

Table 4–130. Enhanced PLL Specifications for -8 Speed Grade (Part 2 of 3)

Symbol	Parameter	Min	Typ	Max	Unit
$t_{EINJITTER}$	External feedback clock period jitter			±200 *(3)*	ps
t_{FCOMP}	External feedback clock compensation time *(4)*			6	ns
f_{OUT}	Output frequency for internal global or regional clock	0.3		357	MHz
f_{OUT_EXT}	Output frequency for external clock *(3)*	0.3		369	MHz
$t_{OUTDUTY}$	Duty cycle for external clock output (when set to 50%)	45		55	%
t_{JITTER}	Period jitter for external clock output *(6)*			±100 ps for >200-MHz `outclk` ±20 mUI for <200-MHz `outclk`	ps or mUI
$t_{CONFIG5,6}$	Time required to reconfigure the scan chains for PLLs 5 and 6			$289/f_{SCANCLK}$	
$t_{CONFIG11,12}$	Time required to reconfigure the scan chains for PLLs 11 and 12			$193/f_{SCANCLK}$	
$t_{SCANCLK}$	`scanclk` frequency *(5)*			22	MHz
t_{DLOCK}	Time required to lock dynamically (after switchover or reconfiguring any non-post-scale counters/delays) *(7)* *(11)*	*(9)*		100	µs
t_{LOCK}	Time required to lock from end of device configuration *(11)*	10		400	µs
f_{VCO}	PLL internal VCO operating range	300		600 *(8)*	MHz

Table 4–130. Enhanced PLL Specifications for -8 Speed Grade (Part 3 of 3)

Symbol	Parameter	Min	Typ	Max	Unit
t_{LSKEW}	Clock skew between two external clock outputs driven by the same counter		±50		ps
t_{SKEW}	Clock skew between two external clock outputs driven by the different counters with the same settings		±75		ps
f_{SS}	Spread spectrum modulation frequency	30		150	kHz
% spread	Percentage spread for spread spectrum frequency *(10)*	0.5		0.6	%
t_{ARESET}	Minimum pulse width on `areset` signal	10			ns

Notes to Tables 4–127 through 4–130:

(1) The minimum input clock frequency to the PFD (f_{IN} / *N*) must be at least 3 MHz for Stratix device enhanced PLLs.

(2) Use this equation ($f_{OUT} = f_{IN} * ml(n \times \text{post-scale counter})$) in conjunction with the specified f_{INPFD} and f_{VCO} ranges to determine the allowed PLL settings.

(3) See "Maximum Input & Output Clock Rates" on page 4–76.

(4) t_{FCOMP} can also equal 50% of the input clock period multiplied by the pre-scale divider *n* (whichever is less).

(5) This parameter is timing analyzed by the Quartus II software because the `scanclk` and `scandata` ports can be driven by the logic array.

(6) Actual jitter performance may vary based on the system configuration.

(7) Total required time to reconfigure and lock is equal to $t_{DLOCK} + t_{CONFIG}$. If only post-scale counters and delays are changed, then t_{DLOCK} is equal to 0.

(8) When using the spread-spectrum feature, the minimum VCO frequency is 500 MHz. The maximum VCO frequency is determined by the speed grade selected.

(9) Lock time is a function of PLL configuration and may be significantly faster depending on bandwidth settings or feedback counter change increment.

(10) Exact, user-controllable value depends on the PLL settings.

(11) The LOCK circuit on Stratix PLLs does not work for industrial devices below -20C unless the PFD frequency > 200 MHz. See the *Stratix FPGA Errata Sheet* for more information on the PLL.

Tables 4–131 through 4–133 describe the Stratix device fast PLL specifications.

Table 4–131. Fast PLL Specifications for -5 & -6 Speed Grade Devices

Symbol	Parameter	Min	Max	Unit
f_{IN}	CLKIN frequency *(1)*, *(2)*, *(3)*	10	717	MHz
f_{INPFD}	Input frequency to PFD	10	500	MHz
f_{OUT}	Output frequency for internal global or regional clock *(3)*	9.375	420	MHz
f_{OUT_DIFFIO}	Output frequency for external clock driven out on a differential I/O data channel *(2)*	*(5)*	*(5)*	
f_{VCO}	VCO operating frequency	300	1,000	MHz
t_{INDUTY}	CLKIN duty cycle	40	60	%
$t_{INJITTER}$	Period jitter for CLKIN pin		±200	ps
t_{DUTY}	Duty cycle for DFFIO 1× CLKOUT pin *(6)*	45	55	%
t_{JITTER}	Period jitter for DIFFIO clock out *(6)*		*(5)*	ps
t_{LOCK}	Time required for PLL to acquire lock	10	100	µs
m	Multiplication factors for *m* counter *(6)*	1	32	Integer
*l*0, *l*1, *g*0	Multiplication factors for *l*0, *l*1, and *g*0 counter *(7)*, *(8)*	1	32	Integer
t_{ARESET}	Minimum pulse width on areset signal	10		ns

Table 4–132. Fast PLL Specifications for -7 Speed Grades (Part 1 of 2)

Symbol	Parameter	Min	Max	Unit
f_{IN}	CLKIN frequency *(1)*, *(3)*	10	640	MHz
f_{INPFD}	Input frequency to PFD	10	500	MHz
f_{OUT}	Output frequency for internal global or regional clock *(4)*	9.375	420	MHz
f_{OUT_DIFFIO}	Output frequency for external clock driven out on a differential I/O data channel	*(5)*	*(5)*	MHz
f_{VCO}	VCO operating frequency	300	700	MHz
t_{INDUTY}	CLKIN duty cycle	40	60	%
$t_{INJITTER}$	Period jitter for CLKIN pin		±200	ps
t_{DUTY}	Duty cycle for DFFIO 1× CLKOUT pin *(6)*	45	55	%

Table 4–132. Fast PLL Specifications for -7 Speed Grades (Part 2 of 2)

Symbol	Parameter	Min	Max	Unit
t_{JITTER}	Period jitter for DIFFIO clock out *(6)*		*(5)*	ps
t_{LOCK}	Time required for PLL to acquire lock	10	100	µs
m	Multiplication factors for *m* counter *(7)*	1	32	Integer
*l*0, *l*1, *g*0	Multiplication factors for *l*0, *l*1, and *g*0 counter *(7)*, *(8)*	1	32	Integer
t_{ARESET}	Minimum pulse width on `areset` signal	10		ns

Table 4–133. Fast PLL Specifications for -8 Speed Grades (Part 1 of 2)

Symbol	Parameter	Min	Max	Unit
f_{IN}	`CLKIN` frequency *(1)*, *(3)*	10	460	MHz
f_{INPFD}	Input frequency to PFD	10	500	MHz
f_{OUT}	Output frequency for internal global or regional clock *(4)*	9.375	420	MHz
f_{OUT_DIFFIO}	Output frequency for external clock driven out on a differential I/O data channel	*(5)*	*(5)*	MHz
f_{VCO}	VCO operating frequency	300	700	MHz
t_{INDUTY}	`CLKIN` duty cycle	40	60	%
$t_{INJITTER}$	Period jitter for `CLKIN` pin		±200	ps
t_{DUTY}	Duty cycle for `DFFIO` 1× `CLKOUT` pin *(6)*	45	55	%
t_{JITTER}	Period jitter for DIFFIO clock out *(6)*		*(5)*	ps
t_{LOCK}	Time required for PLL to acquire lock	10	100	µs
m	Multiplication factors for *m* counter *(7)*	1	32	Integer
*l*0, *l*1, *g*0	Multiplication factors for *l*0, *l*1, and *g*0 counter *(7)*, *(8)*	1	32	Integer

Table 4–133. Fast PLL Specifications for -8 Speed Grades (Part 2 of 2)				
Symbol	**Parameter**	**Min**	**Max**	**Unit**
t_{ARESET}	Minimum pulse width on `areset` signal	10		ns

Notes to Tables 4–131 through 4–133:

(1) See "Maximum Input & Output Clock Rates" on page 4–76.

(2) PLLs 7, 8, 9, and 10 in the EP1S80 device support up to 717-MHz input and output.

(3) Use this equation ($f_{OUT} = f_{IN} * ml(n \times \text{post-scale counter})$) in conjunction with the specified f_{INPFD} and f_{VCO} ranges to determine the allowed PLL settings.

(4) When using the SERDES, high-speed differential I/O mode supports a maximum output frequency of 210 MHz to the global or regional clocks (that is, the maximum data rate 840 Mbps divided by the smallest SERDES J factor of 4).

(5) Refer to the section "High-Speed I/O Specification" on page 4–87 for more information.

(6) This parameter is for high-speed differential I/O mode only.

(7) These counters have a maximum of 32 if programmed for 50/50 duty cycle. Otherwise, they have a maximum of 16.

(8) High-speed differential I/O mode supports W = 1 to 16 and J = 4, 7, 8, or 10.

DLL Specifications

Table 4–134 reports the jitter for the DLL in the DQS phase shift reference circuit.

Table 4–134. DLL Jitter for DQS Phase Shift Reference Circuit	
Frequency (MHz)	**DLL Jitter (ps)**
197 to 200	± 100
160 to 196	± 300
100 to 159	± 500

f For more information on DLL jitter, see the *DDR SRAM* section in the *Stratix Architecture* chapter of the *Stratix Device Handbook, Volume 1*.

Table 4–135 lists the Stratix DLL low frequency limit for full phase shift across all PVT conditions. The Stratix DLL can be used below these frequencies, but it will not achieve the full phase shift requested across all

process and operating conditions. Run the timing analyzer in the Quartus II software at the fast and slow operating conditions to see the phase shift range that is achieved below these frequencies.

Table 4–135. Stratix DLL Low Frequency Limit for Full Phase Shift

Phase Shift	Minimum Frequency for Full Phase Shift	Unit
72°	119	MHz
90°	149	MHz

5. Reference & Ordering Information

S51005-2.1

Software

Stratix® devices are supported by the Altera® Quartus® II design software, which provides a comprehensive environment for system-on-a-programmable-chip (SOPC) design. The Quartus II software includes HDL and schematic design entry, compilation and logic synthesis, full simulation and advanced timing analysis, SignalTap® II logic analyzer, and device configuration. See the *Design Software Selector Guide* for more details on the Quartus II software features.

The Quartus II software supports the Windows XP / 2000 / NT / 98, Sun Solaris, Linux Red Hat v7.1 and HP-UX operating systems. It also supports seamless integration with industry-leading EDA tools through the NativeLink® interface.

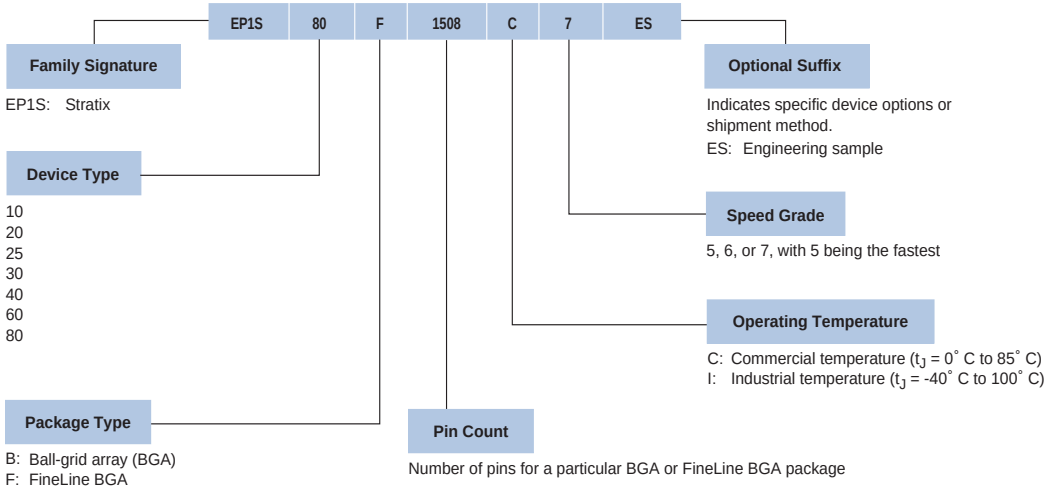
Device Pin-Outs

Stratix device pin-outs can be found on the Altera web site **(www.altera.com).**

Ordering Information

Figure 5–1 describes the ordering codes for Stratix devices. For more information on a specific package, see the *Package Information for Stratix Devices* chapter.

Figure 5–1. Stratix Device Packaging Ordering Information



Index

A

Accumulator 2–63
Adder / Output Blocks 2–61
Adder / Subtractor 2–63
  Accumulator 2–63
AGP 1x Specifications 4–13
AGP 2x Specifications 4–13
Architecture 2–1
  36 x 36 Multiply Mode 2–66
  addnsub Signal 2–8
  Block Diagram 2–2
  Bus Hold 2–121
  Byte Alignment 2–140
  Carry-Select Chain 2–11
  Clear & Preset Logic Control 2–13
  Combined Resources 2–78
  Dedicated Circuitry 2–137
  Device Resources 2–3
  Device Routing Scheme 2–20
  Digital Signal Processing Block 2–52
  Direct Link Connection 2–5
  Dynamic Arithmetic Mode 2–10
    in LE 2–11
  Four-Multipliers
    Adder Mode 2–68
  Functional Description 2–1
  LAB
    Interconnects 2–4
    Logic Array Blocks 2–3
    Structure 2–4
  LE Operating Modes 2–8
  Logic Elements 2–6
  Modes of Operation 2–64
  Multiplier Size & Configurations per DSP block 2–70
  Multiply-Accumulator Mode 2–67
  MultiTrack Interconnect 2–14
  Normal Mode 2–9
    in LE 2–9
  Open-Drain Output 2–120
  Power Sequencing & Hot Socketing 2–140
  Programmable Drive Strength 2–119
  Programmable Pull-Up Resistor 2–122
  Simple Multiplier Mode 2–64
  Single-Port Mode 2–51
  Slew-Rate Control 2–120
  Two-Multipliers
    Adder Mode 2–67
    Adder Mode Implementing Complex Multiply 2–68

C

Class I Specifications 4–11, 4–12
Class II Specifications 4–11, 4–12, 4–13
Clocks
  Clock Feedback 2–96
  Clock Multiplication & Division 2–88, 2–101
  Clock Switchover 2–88
  Delay 2–97
  EP1S10, EP1S20 & EP1S25
    Device I / O Clock Groups 2–80
  EP1S25, EP1S20 & EP1S10 Device Fast Clock Pin Connections to Fast Regional Clocks 2–77
  EP1S30 Device Fast Regional Clock Pin Connections to Fast Regional Clocks 2–78
  EP1S30, EP1S40, EP1S60, EP1S80
    Device I / O Clock Groups 2–81
  External Clock
    Inputs 2–102
    Outputs 2–92, 2–103
    Outputs for Enhanced PLLs 11 & 12 2–95
    Outputs for PLLs 5 & 6 2–93
  Fast Regional Clock External I / O Timing Parameters 4–34
  Fast Regional Clock Network 2–76

Global & Hierarchical Clocking 2–73
Global & Regional Clock Connections
from Side Pins & Fast PLL Outputs 2–85
from Top Clock Pins & Enhanced PLL Outputs 2–86
Global Clock External I/O Timing Parameters 4–35
Global Clock Network 2–74
Global Clocking 2–75
Independent Clock Mode 2–44
Input/Output
Clock Mode 2–46
Simple Dual-Port Mode 2–48
True Dual-Port Mode 2–47
Maximum Input & Output Clock Rates 4–76
Maximum Input Clock Rate
for CLK
(0, 2, 9, 11) Pins in
Flip-Chip Packages 4–77
Wire-Bond Packages 4–79
(1, 3, 8, 10) Pins in
Flip-Chip Packages 4–78
Wire-Bond Packages 4–80
(7..4) & CLK(15..12) Pins in
Flip-Chip Packages 4–76
Wire-Bond Packages 4–78
Maximum Output Clock Rate
for PLL
(1, 2, 3, 4) Pins in
Flip-Chip Packages 4–83
Wire-Bond Packages 4–85
(5, 6, 11, 12) Pins in
Flip-Chip Packages 4–81
Wire-Bond Packages 4–84
Phase & Delay Shifting 2–96
Phase Delay 2–96
PLL Clock Networks 2–73
Read/Write Clock Mode 2–49
in Simple Dual-Port Mode 2–50
Regional Clock 2–75
External I/O Timing Parameters 4–34
Regional Clock Bus 2–79
Regional Clock Network 2–75
Spread-Spectrum Clocking 2–98
Configuration 3–5
32-Bit IDCODE 3–3
and Testing 3–1
Data Sources for Configuration 3–7
Local Update Mode 3–12
Local Update Transition Diagram 3–12
Operating Modes 3–5
Partial Reconfiguration 3–7
Remote Update 3–8
Remote Update Transition Diagram 3–11
Schemes 3–7
SignalTap II Embedded Logic Analyzer 3–5
Stratix FPGAs with JRunner 3–7
Control Signals 2–104

D

DC Switching
Absolute Maximum Ratings 4–1
Bus Hold Parameters 4–16
Capacitance 4–17
DC & Switching Characteristics 4–1
External Timing Parameters 4–33
Operating Conditions 4–1
Performance 4–20
Power Consumption 4–17
Recommended Operating Conditions 4–1
DDR
Double-Data Rate I/O Pins 2–111
Device Features
EP1S10, EP1S20, EP1S25, EP1S30, 1–3
EP1S40, EP1S60, EP1S80, 1–3

Differential HSTL Specifications 4–15
DSP
Block Diagram
Configuration
for 18 x 18-Bit 2–55
for 9 x 9-Bit 2–56
Block Interconnect Interface 2–71
Block Interface 2–70
Block Signal Sources & Destinations 2–73
Blocks
Arranged in Columns 2–53
in Stratix Devices 2–54
Input Register Modes 2–60
Input Registers 2–58
Multiplier
2–60
Block 2–57
Signed Representation 2–60
Sub-Block 2–57
Sub-Blocks Using Input Shift Register Connections 2–59
Pipeline / Post Multiply Register 2–61

E

EP1S10 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–36
Global Clock External I/O Timing Parameters 4–37
Regional Clock External I/O Timing Parameters 4–36
Row Pin
Fast Regional Clock External I/O Timing Parameters 4–37
Global Clock External I/O Timing Parameters 4–38
Regional Clock External I/O Timing Parameters 4–38
EP1S20 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–39
Global Clock External I/O Timing Parameters 4–40
Regional Clock External I/O Timing Parameters 4–39
Row Pin
Fast Regional Clock External I/O Timing Parameters 4–40
Global Clock External I/O Timing Parameters 4–41
Regional Clock External I/O Timing Parameters 4–41
EP1S25 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–42
Global Clock External I/O Timing Parameters 4–43
Regional Clock External I/O Timing Parameters 4–42
Row Pin
Fast Regional Clock External I/O Timing Parameters 4–43
Global Clock External I/O Timing Parameters 4–44
Regional Clock External I/O Timing Parameters 4–44
EP1S30 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–45
Global Clock External I/O Timing Parameters 4–45
Regional Clock External I/O Timing Parameters 4–45
Row Pin
Fast Regional Clock External I/O Timing Parameters 4–46
Global Clock External I/O Timing Parameters 4–47
Regional Clock External I/O Timing Parameters 4–47
EP1S40 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–48
Global Clock External I/O Timing Parameters 4–49
Regional Clock External I/O Timing Parameters 4–48
Row Pin

Fast Regional Clock External I/O Timing Parameters 4–49
Global Clock External I/O Timing Parameters 4–50
Regional Clock External I/O Timing Parameters 4–50
EP1S60 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–51
Global Clock External I/O Timing Parameters 4–52
Regional Clock External I/O Timing Parameters 4–51
M-RAM
Interface Locations 2–38
Row Pin
Fast Regional Clock External I/O Timing Parameters 4–52
Global Clock External I/O Timing Parameters 4–53
Regional Clock External I/O Timing Parameters 4–53
EP1S80 Devices
Column Pin
Fast Regional Clock External I/O Timing Parameters 4–54
Global Clock External I/O Timing Parameters 4–55
Regional Clock External I/O Timing Parameters 4–54
Global Clock External I/O Timing Parameters 4–56
Row Pin
Fast Regional Clock External I/O Timing Parameters 4–55
Regional Clock External I/O Timing Parameters 4–56

H

HSTL
Class I Specifications 4–14, 4–15
Class II Specifications 4–14, 4–15

I

I/O
Standards
1.5-V 4–14, 4–15
I/O Specifications 4–4
1.8-V
I/O Specifications 4–4
2.5-V
I/O Specifications 4–3
3.3-V 4–13
LVDS I/O Specifications 4–6
PCI Specifications 4–9
PCML Specifications 4–8
Advanced I/O Standard Support 2–122
Column I/O Block Connection to the Interconnect 2–107
Column Pin
Input Delay Adders 4–66
Control Signal Selection per IOE 2–109
CTT I/O Specifications 4–16
Differential LVDS Input On-Chip Termination 2–128
External I/O Delay Parameters 4–66
GTL+ I/O Specifications 4–10
High-Speed Differential I/O Support 2–130
HyperTransport Technology Specifications 4–9
I/O Banks 2–125
I/O Structure 2–104
I/O Support by Bank 2–126
IOE Structure 2–105
LVCMOS Specifications 4–3
LVDS Performance on Fast PLL Input 2–103
LVPECL Specifications 4–8
LVTTL Specifications 4–3
MultiVolt I/O Interface 2–129
MultiVolt I/O Support 2–130
Output Delay Adders for Fast Slew Rate on Column Pins 4–68
Output Delay Adders for Fast Slew Rate on Row Pins 4–69
Output Delay Adders for Slow Slew Rate on Column Pins 4–70
Package Options & I/O Pin Counts 1–4
Receiver Input Waveforms for Differential

I/O Standards 4–5
Row I/O Block Connection to the Interconnect 2–106
Row Pin
Input Delay Adders 4–67
Signal Path through the I/O Block 2–108
SSTL-18 4–11
SSTL-2 4–12
SSTL-3 4–12, 4–13
Stratix IOE in Bidirectional I/O Configuration 2–110
Supported I/O Standards 2–123
Transmitter Output Waveforms for Differential I/O Standards 4–6
Interconnect
C4 Connections 2–18
DSP Block Interface to Interconnect 2–72
Left-Facing M-RAM to Interconnect Interface 2–40
LUT Chain Register Chain Interconnects 2–17
M-RAM
Column Unit Interface to Interconnect 2–42
Row Unit Interface to Interconnect 2–41
R4 Connections 2–15
IOE
Internal Timing Microparameters 4–29

J

JTAG
Boundary-Scan
Register Length 3–3
Support 3–1
Stratix JTAG
Instructions 3–2
Waveforms 3–4

L

LAB
Control Signals 2–5
Wide Control Signals 2–6
LUT
Chain & Register Chain 2–8

M

Memory Architecture
Byte Enable for M4K
RAM Block
2–32
Byte Enable for M-RAM
Block
2–35
External RAM Interfacing 2–115
M4K
Block Internal Timing
Microparameter
Descriptions 4–24
Microparameters 4–31
RAM Block
2–30
Configurations (Simple Dual-Port) 2–31
Configurations (True Dual-Port) 2–31
Control Signals 2–33
LAB Row Interface 2–33
M512
Block Internal Timing
Microparameter
Descriptions 4–24
Microparameters 4–30
RAM Block
Architecture 2–27
Configurations (Simple Dual-Port RAM) 2–27
Control Signals 2–29
LAB Row Interface 2–30
Memory Block Size 2–26
Memory Modes 2–21
M-RAM
Block
2–34
Configurations (Simple Dual-Port) 2–34
Configurations (True Dual-Port) 2–35
Block Control Signals 2–37
Block Internal Timing
Microparameter
Descriptions 4–25
Combined Byte Selection for x144

Mode 2–36
Row & Column Interface Unit Signals 2–43
Parity Bit Support 2–24
Shift Register
Memory Configuration 2–26
Support 2–25
Simple Dual-Port & Single-Port Memory Configurations 2–23
Stratix IOE in DDR Input I/O Configuration 2–112
Stratix IOE in DDR Output I/O Configuration 2–114
TriMatrix Memory 2–21
True Dual-Port Memory Configuration 2–22

O

Ordering Information 5–1
Device Pin-Outs 5–1
Packaging Ordering Information 5–2
Reference & Ordering Information 5–1
Output Registers 2–64
Output Selection Multiplexer 2–64

P

Packaging
BGA Package Sizes 1–4
Device Speed Grades 1–5
FineLine BGA Package Sizes 1–5
PCI-X 1.0 Specifications 4–10
Phase Shifting 2–103
PLL
Advanced Clear & Enable Control 2–98
Dynamically Programmable Counters & Delays in Stratix Device Enhanced PLLs 2–91
Enhanced
Fast PLLs 2–81
Fast PLL 2–100
Channel Layout EP1S10, EP1S20 or EP1S25 Devices 2–138
Channel Layout EP1S30 to EP1S80 Devices 2–139
Port I/O Standards 2–102
I/O Standards Supported for Enhanced PLL Pins 2–94
Lock Detect & Programmable Gated Locked 2–98
PLL Locations 2–84
Programmable Bandwidth 2–91
Programmable Delay Chain 2–111
Programmable Duty Cycle 2–98
Reconfiguration 2–90

T

Testing
Temperature Sensing Diode 3–13
Electrical Characteristics 3–14
External 3–14
Temperature vs. Temperature-Sensing Diode Voltage 3–15
Timing
DSP
Block Internal Timing Microparameter Descriptions 4–23
Microparameters 4–29
Dual-Port RAM Timing Microparameter Waveform 4–27
External Timing in Stratix Devices 4–33
High-Speed I/O Timing 4–87
High-Speed Timing Specifications & Terminology 4–87
Internal Parameters 4–22
IOE Internal Timing Microparameter Descriptions 4–22
LE Internal Timing Microparameters 4–28
Logic Elements Internal Timing Microparameter Descriptions 4–22
Model 4–19
PLL Timing 4–94
Preliminary & Final 4–19
Stratix Device Timing Model Status 4–19
Stratix JTAG
Timing Parameters & Values 3–4
TriMatrix Memory
TriMatrix Memory Features 2–21

With down-spread modulation, the peak of the modulated waveform is the actual target frequency. Therefore, the system never exceeds the maximum clock speed. To maintain reliable communication, the entire system / subsystem should use the Stratix or Stratix GX device as the clock source. Communication could fail if the Stratix or Stratix GX logic array is clocked by the spread-spectrum clock, but the data it receives from another device is not.

Since spread spectrum affects the *m* counter values, all spread-spectrum PLL outputs are affected. Therefore, if only one spread-spectrum signal is needed, the clock signal should use a separate PLL without other outputs from that PLL.

No special considerations are needed when using spread spectrum with the clock switchover feature. This is because the clock switchover feature does not affect the *m* and *n* counter values, which are the counter values that are switching when using spread spectrum.

PLL Reconfiguration

f See *AN 282: Implementing PLL Reconfiguration in Stratix & Stratix GX Devices* for information on PLL reconfiguration.

Enhanced PLL Pins

Table 1–9 shows the physical pins and their purpose for the Enhanced PLLs. For `inclk` port connections to pins see "Clocking" on page 1–39.

Table 1–9. Enhanced PLL Pins (Part 1 of 2)

Pin	Description
`CLK4p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 6.
`CLK5p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 6.
`CLK6p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 12.
`CLK7p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 12.
`CLK12p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 11.
`CLK13p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 11.
`CLK14p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 5.
`CLK15p/n`	Single-ended or differential pins that can drive the `inclk` port for PLL 5.
`PLL5_FBp/n`	Single-ended or differential pins that can drive the `fbin` port for PLL 5.
`PLL6_FBp/n`	Single-ended or differential pins that can drive the `fbin` port for PLL 6.
`PLLENABLE`	Dedicated input pin that drives the `pllena` port of all or a set of PLLs. If you do not use this pin, connect it to ground.

Table 1–9. Enhanced PLL Pins (Part 2 of 2)

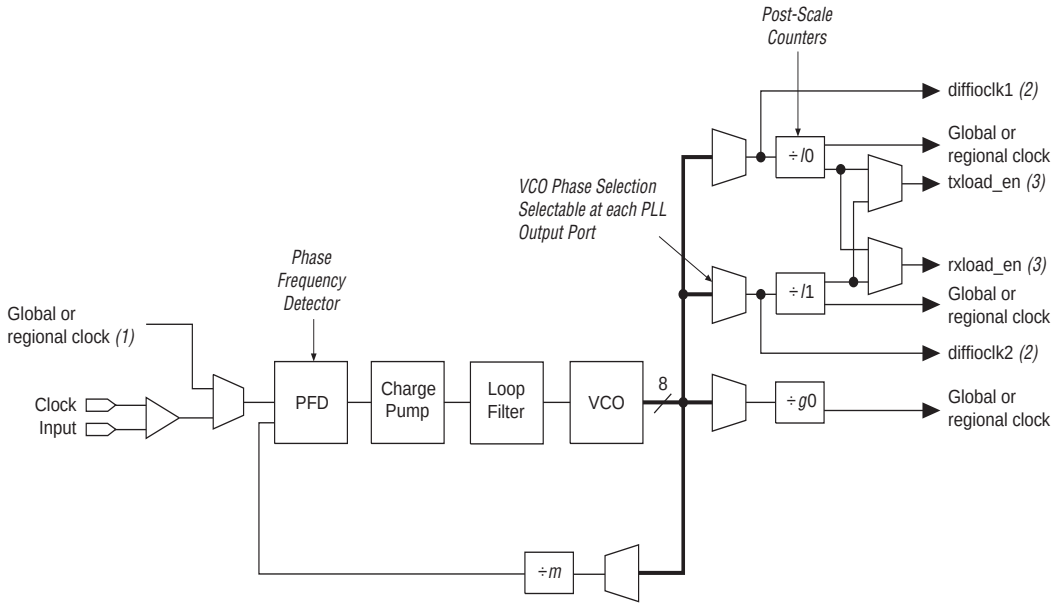
Pin	Description
PLL5_OUT[3..0]p/n	Single-ended or differential pins driven by extclk[3..0] ports from PLL 5.
PLL6_OUT[3..0]p/n	Single-ended or differential pins driven by extclk[3..0] ports from PLL 6.
PLL11_OUT, CLK13n	Single-ended output pin driven by clk0 port from PLL 11.
PLL12_OUT, CLK6n	Single-ended output pin driven by clk0 port from PLL 12.
VCCA_PLL5	Analog power for PLL 5. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL5	Guard ring power for PLL 5. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL5	Analog ground for PLL 5. You can connect this pin to the GND plane on the board.
GNDG_PLL5	Guard ring ground for PLL 5. You can connect this pin to the GND plane on the board.
VCCA_PLL6	Analog power for PLL 6. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL6	Guard ring power for PLL 6. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL6	Analog ground for PLL 6. You can connect this pin to the GND plane on the board.
GNDG_PLL6	Guard ring ground for PLL 6. You can connect this pin to the GND plane on the board.
VCCA_PLL11	Analog power for PLL 11. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL11	Guard ring power for PLL 11. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL11	Analog ground for PLL 11. You can connect this pin to the GND plane on the board.
GNDG_PLL11	Guard ring ground for PLL 11.You can connect this pin to the GND plane on the board.
VCCA_PLL12	Analog power for PLL 12. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL12	Guard ring power for PLL 12. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL12	Analog ground for PLL 12. You can connect this pin to the GND plane on the board.
GNDG_PLL12	Guard ring ground for PLL 12. You can connect this pin to the GND plane on the board.
VCC_PLL5_OUTA	External clock output V_{CCIO} power for PLL5_OUT0p, PLL5_OUT0n, PLL5_OUT1p, and PLL5_OUT1n outputs from PLL 5.
VCC_PLL5_OUTB	External clock output V_{CCIO} power for PLL5_OUT2p, PLL5_OUT2n, PLL5_OUT3p, and PLL5_OUT3n outputs from PLL 5.
VCC_PLL6_OUTA	External clock output V_{CCIO} power for PLL5_OUT0p, PLL5_OUT0n, PLL5_OUT1p, and PLL5_OUT1n outputs from PLL 6.
VCC_PLL6_OUTB	External clock output V_{CCIO} power for PLL5_OUT2p, PLL5_OUT2n, PLL5_OUT3p, and PLL5_OUT3n outputs from PLL 6.

Fast PLLs

Stratix devices contain up to eight fast PLLs and Stratix GX devices contain up to four fast PLLs. Both device PLLs have high-speed differential I/O interface ability along with general-purpose features. Figure 1–17 shows a diagram of the fast PLL. This section discusses the

general purpose abilities of the Fast PLL. For information on the high-speed differential I/O interface capabilities, see the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter.

Figure 1–17. Stratix & Stratix GX Fast PLL Block Diagram



Notes to Figure 1–17:

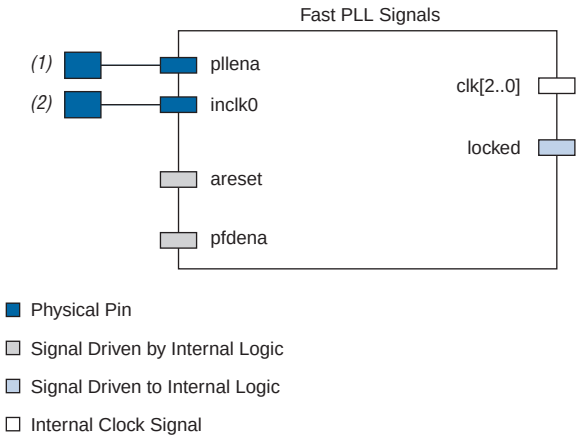
(1) The global or regional clock input can be driven by an output from another PLL or any dedicated CLK or FCLK pin. It cannot be driven by internally-generated global signals.

(2) In high-speed differential I/O support mode, this high-speed PLL clock feeds the SERDES. Stratix and Stratix GX devices only support one rate of data transfer per fast PLL in high-speed differential I/O support mode.

(3) This signal is a high-speed differential I/O support SERDES control signal.

Figure 1–18 shows all possible ports related to fast PLLs.

Figure 1–18. Fast PLL Ports & Physical Destinations



Notes to Figure 1–18:

(1) This input pin is shared by all enhanced and fast PLLs.

(2) This input pin is either single-ended or differential.

Tables 1–10 and 1–11 show the description of all fast PLL ports.

Table 1–10. Fast PLL Input Signals

Name	Description	Source	Destination
`inclk1`	Reference clock input to PLL	Pin	PFD
`pllena`	Enable pin for enabling or disabling all or a set of PLLs – active high	Pin	PLL control signal
`areset`	Signal used to reset the PLL which re-synchronizes all the counter outputs—active high	Logic array	PLL control signal
`pfdena`	Enables the up/down outputs from the phase-frequency detector—active high	Logic array	PFD

Table 1–11. Fast PLL Output Signals

Name	Description	Source	Destination
`clk[2..0]`	PLL outputs driving regional or global clock	PLL counter	Internal clock
`locked`	Lock output from lock detect circuit—active high	PLL lock detect	Logic array

Clock Multiplication & Division

Stratix and Stratix GX device fast PLLs provide clock synthesis for PLL output ports using *m* / (post scaler) scaling factors. The input clock is multiplied by the *m* feedback factor. Each output port has a unique post scale counter to divide down the high-frequency VCO. There is one multiply counter, *m*, per fast PLL with a range of 1 to 32. There are three post-scale counters (*g*0, *l*0, and *l*1) for the regional and global clock output ports. All post-scale counters range from 1 to 32. If the design uses a high-speed serial interface, you can set the output counter to 1 to allow the high-speed VCO frequency to drive the SERDES.

External Clock Outputs

Each fast PLL supports differential or single-ended outputs for source-synchronous transmitters or for general-purpose external clocks. There are no dedicated external clock output pins. The fast PLL global or regional outputs can drive any I / O pin as an external clock output pin. The I / O standards supported by any particular bank determines what standards are possible for an external clock output driven by the fast PLL in that bank. See the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook, Volume 2* or the *Stratix GX Device Handbook, Volume 2* for output standard support.

Table 1–12 shows the I / O standards supported by fast PLL input pins.

Table 1–12. Fast PLL Port I/O Standards (Part 1 of 2)

I/O Standard	Input	
	INCLK	PLLENABLE
LVTTL	v	v
LVCMOS	v	v
2.5 V	v	
1.8 V	v	
1.5 V	v	
3.3-V PCI		
3.3-V PCI-X 1.0		
LVPECL	v	
PCML	v	
LVDS	v	
HyperTransport technology	v	
Differential HSTL	v	

Table 1–12. Fast PLL Port I/O Standards (Part 2 of 2)

I/O Standard	Input	
	INCLK	PLLENABLE
Differential SSTL		
3.3-V GTL		
3.3-V GTL+	v	
1.5-V HSTL Class I	v	
1.5-V HSTL Class II		
1.8-V HSTL Class I	v	
1.8-V HSTL Class II		
SSTL-18 Class I	v	
SSTL-18 Class II		
SSTL-2 Class I	v	
SSTL-2 Class II	v	
SSTL-3 Class I	v	
SSTL-3 Class II	v	
AGP (1× and 2×)		
CTT	v	

Phase Shifting

Stratix and Stratix GX device fast PLLs have advanced clock shift ability to provide programmable phase shift. These parameters are set in the Quartus II software.

The Quartus II software automatically sets the phase taps and counter settings according to the phase shift entry. Enter a desired phase shift and the Quartus II software automatically sets the closest setting achievable. This type of phase shift is not reconfigurable during system operation. You can enter a phase shift (in degrees or time units) for each PLL clock output port or for all outputs together in one shift. You can perform phase shifting in time units with a resolution range of 125 to 416.66 ps to create a function of frequency input and the multiplication and division factors (that is, it is a function of the VCO period), with the finest step being equal to an eighth (×0.125) of the VCO period. Each clock output counter can choose a different phase of the VCO period from up to eight taps for individual fine-step selection. Also, each clock output counter can use a unique initial count setting to achieve individual coarse shift selection in steps of one VCO period. The combination of coarse and grain shifts allows phase shifting for the entire input clock period.

The equation to determine the precision of phase in degrees is: 45° ÷ post-scale counter value. Therefore, the maximum step size is 45°, and smaller steps are possible depending on the multiplication and division ratio necessary on the output counter port.

This type of phase shift provides the highest precision since it is the least sensitive to process, supply, and temperature variation.

Programmable Duty Cycle

The programmable duty cycle allows the fast PLL to generate clock outputs with a variable duty cycle. This feature is supported on each fast PLL post-scale counter. *g*0, *l*0, and *l*1 all support programmable duty. You use a low- and high-time count setting for the post-scale counters to set the duty cycle.

The Quartus II software uses the frequency input and multiply / divide rate desired to select the post-scale counter, which determines the possible choices for each duty cycle. The precision of the duty cycle is determined by the post-scale counter value chosen on an output. The precision is defined by 50% divided by the post-scale counter value. The closest value to 100% is not achievable for a given counter value. For example, if the *g*0 counter is 10, then steps of 5% are possible for duty cycle choices between 5 to 90%.

If the device uses external feedback, you must set the duty cycle for the counter driving off the device to 50%.

Control Signals

The lock output indicates a stable clock output signal in phase with the reference clock. Unlike enhanced PLLs, fast PLLs do not have a lock filter counter.

The `pllenable` pin is a dedicated pin that enables / disables both PLLs. When the `pllenable` pin is low, the clock output ports are driven by `GND` and all the PLLs go out of lock. When the `pllenable` pin goes high again, the PLLs relock and resynchronize to the input clocks. You can choose which PLLs are controlled by the `pllenable` by connecting the `pllenable` input port of the `altpll` megafunction to the common `pllenable` input pin.

The `areset` signals are reset / resynchronization inputs for each fast PLL. The Stratix and Stratix GX devices can drive these input signals from an input pin or from LEs. When driven high, the PLL counters reset, clearing the PLL output and placing the PLL out of lock. The VCO sets back to its nominal setting (~700 MHz). When driven low again, the PLL

resynchronizes to its input clock as it relocks. If the target VCO frequency is below this nominal frequency, then the output frequency starts at a higher value then desired as it locks.

The `pfdena` signals control the PFD output with a programmable gate. If you disable the PFD, the VCO operates at its last set value of control voltage and frequency with some long-term drift to a lower frequency. The system continues running when the PLL goes out of lock or the input clock disables. By maintaining the last locked frequency, the system has time to store its current settings before shutting down.

If the PLL loses lock for any reason (for example, because of excessive `inclk` jitter, clock switchover, PLL reconfiguration, or power supply noise), the PLL must be reset with the `areset` signal to guarantee correct phase relationship between the PLL output clocks. If the phase relationship between the input clock and the output clock and between different output clocks from the PLL is not important in your design, it is not necessary to reset the PLL.

Pins

Table 1–13 shows the physical pins and their purpose for the Fast PLLs. For inclk port connections to pins see "Clocking" on page 1–39.

Table 1–13. Fast PLL Pins (Part 1 of 3)

Pin	Description
CLK0p/n	Single-ended or differential pins that can drive the inclk port for PLL 1 or 7.
CLK1p/n	Single-ended or differential pins that can drive the inclk port for PLL 1.
CLK2p/n	Single-ended or differential pins that can drive the inclk port for PLL 2 or 8.
CLK3p/n	Single-ended or differential pins that can drive the inclk port for PLL 2.
CLK8p/n	Single-ended or differential pins that can drive the inclk port for PLL 3 or 9. *(1)*
CLK9p/n	Single-ended or differential pins that can drive the inclk port for PLL 3. *(1)*
CLK10p/n	Single-ended or differential pins that can drive the inclk port for PLL 4 or 10. *(1)*
CLK11p/n	Single-ended or differential pins that can drive the inclk port for PLL 4. *(1)*
FPLL7CLKp/n	Single-ended or differential pins that can drive the inclk port for PLL 7.
FPLL8CLKp/n	Single-ended or differential pins that can drive the inclk port for PLL 8.
FPLL9CLKp/n	Single-ended or differential pins that can drive the inclk port for PLL 9. *(1)*
FPLL10CLKp/n	Single-ended or differential pins that can drive the inclk port for PLL 10. *(1)*
PLLENABLE	Dedicated input pin that drives the pllena port of all or a set of PLLs. If you do not use this pin, connect it to ground.
VCCA_PLL1	Analog power for PLL 1. Connect this pin to 1.5 V, even if the PLL is not used.

Table 1–13. Fast PLL Pins (Part 2 of 3)

Pin	Description
VCCG_PLL1	Guard ring power for PLL 1. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL1	Analog ground for PLL 1. You can connect this pin to the GND plane on the board.
GNDG_PLL1	Guard ring ground for PLL 1. You can connect this pin to the GND plane on the board.
VCCA_PLL2	Analog power for PLL 2. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL2	Guard ring power for PLL 2. Connect this pin to1.5 V, even if the PLL is not used.
GNDA_PLL2	Analog ground for PLL 2. You can connect this pin to the GND plane on the board.
GNDG_PLL2	Guard ring ground for PLL 2. You can connect this pin to the GND plane on the board.
VCCA_PLL3	Analog power for PLL 3. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
VCCG_PLL3	Guard ring power for PLL 3. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
GNDA_PLL3	Analog ground for PLL 3. You can connect this pin to the GND plane on the board. *(1)*
GNDG_PLL3	Guard ring ground for PLL 3. You can connect this pin to the GND plane on the board. *(1)*
VCCA_PLL4	Analog power for PLL 4. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
VCCG_PLL4	Guard ring power for PLL 4. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
GNDA_PLL4	Analog ground for PLL 4. You can connect this pin to the GND plane on the board. *(1)*
GNDG_PLL4	Guard ring ground for PLL 4. You can connect this pin to the GND plane on the board. *(1)*
VCCA_PLL7	Analog power for PLL 7. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL7	Guard ring power for PLL 7. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL7	Analog ground for PLL 7. You can connect this pin to the GND plane on the board.
GNDG_PLL7	Guard ring ground for PLL 7. You can connect this pin to the GND plane on the board.
VCCA_PLL8	Analog power for PLL 8. Connect this pin to 1.5 V, even if the PLL is not used.
VCCG_PLL8	Guard ring power for PLL 8. Connect this pin to 1.5 V, even if the PLL is not used.
GNDA_PLL8	Analog ground for PLL 8. You can connect this pin to the GND plane on the board.
GNDG_PLL8	Guard ring ground for PLL 8. You can connect this pin to the GND plane on the board.
VCCA_PLL9	Analog power for PLL 9. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
VCCG_PLL9	Guard ring power for PLL 9. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*

Table 1–13. Fast PLL Pins (Part 3 of 3)

Pin	Description
GNDA_PLL9	Analog ground for PLL 9. You can connect this pin to the GND plane on the board. *(1)*
GNDG_PLL9	Guard ring ground for PLL 9. You can connect this pin to the GND plane on the board. *(1)*
VCCA_PLL10	Analog power for PLL 10. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
VCCG_PLL10	Guard ring power for PLL 10. Connect this pin to 1.5 V, even if the PLL is not used. *(1)*
GNDA_PLL10	Analog ground for PLL 10. Connect this pin to the GND plane on the board. *(1)*
GNDG_PLL10	Guard ring ground for PLL 10. You can connect this pin to the GND plane on the board. *(1)*

Note to Table 1–13:

(1) PLLs 3, 4, 9, and 10 are not available on Stratix GX devices for general-purpose configuration. These PLLs are part of the HSSI block. See *AN 236: Using Source-Synchronous Signaling with DPA in Stratix GX Devices* for more information.

Clocking

Stratix and Stratix GX devices provide a hierarchical clock structure and multiple PLLs with advanced features. The large number of clocking resources in combination with the clock synthesis precision provided by enhanced and fast PLLs provides a complete clock management solution.

Global & Hierarchical Clocking

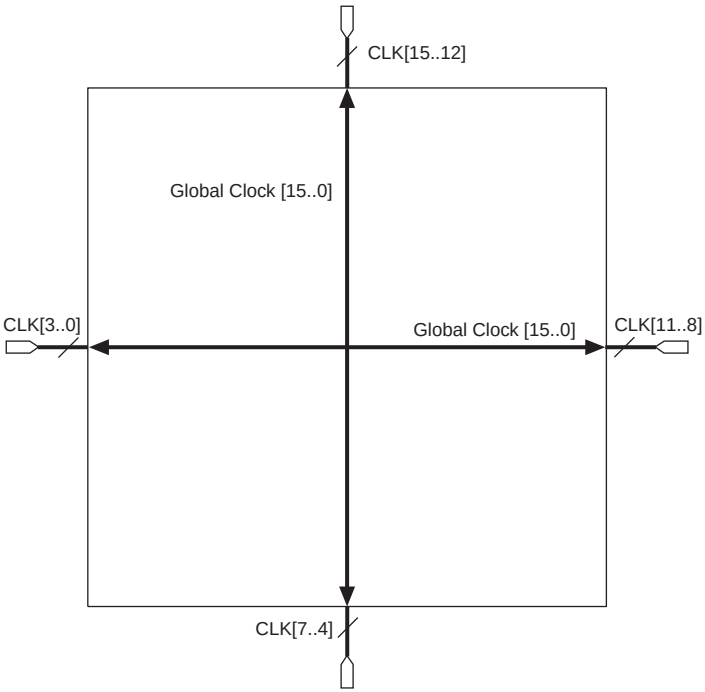
Stratix and Stratix GX devices provide 16 dedicated global clock networks, 16 regional clock networks (4 per device quadrant), and 8 dedicated fast regional clock networks. These clocks are organized into a hierarchical clock structure that allows for up to 22 clocks per device region with low skew and delay. This hierarchical clocking scheme provides up to 48 unique clock domains within Stratix and Stratix GX devices.

There are 16 dedicated clock pins (CLK[15..0]) on Stratix devices and 12 dedicated clock pins (CLK[11..0]) on Stratix GX devices to drive either the global or regional clock networks. Four clock pins drive each side of the Stratix device, as shown in Figures 1–19 and 1–20. On Stratix GX devices, four clock pins drive the top, left, and bottom sides of the device. The clocks on the right side of the device are not available for general-purpose PLLs. Enhanced and fast PLL outputs can also drive the global and regional clock networks.

Global Clock Network

These clocks drive throughout the entire device, feeding all device quadrants. All resources within the device—IOEs, LEs, DSP blocks, and all memory blocks—can use the global clock networks as clock sources. These resources can also be used for control signals, such as clock enables and synchronous or asynchronous clears fed from the external pin. Internal logic can also drive the global clock networks for internally generated global clocks and asynchronous clears, clock enables, or other control signals with large fanout. Figure 1–19 shows the 16 dedicated `CLK` pins driving global clock networks.

Figure 1–19. Global Clocking

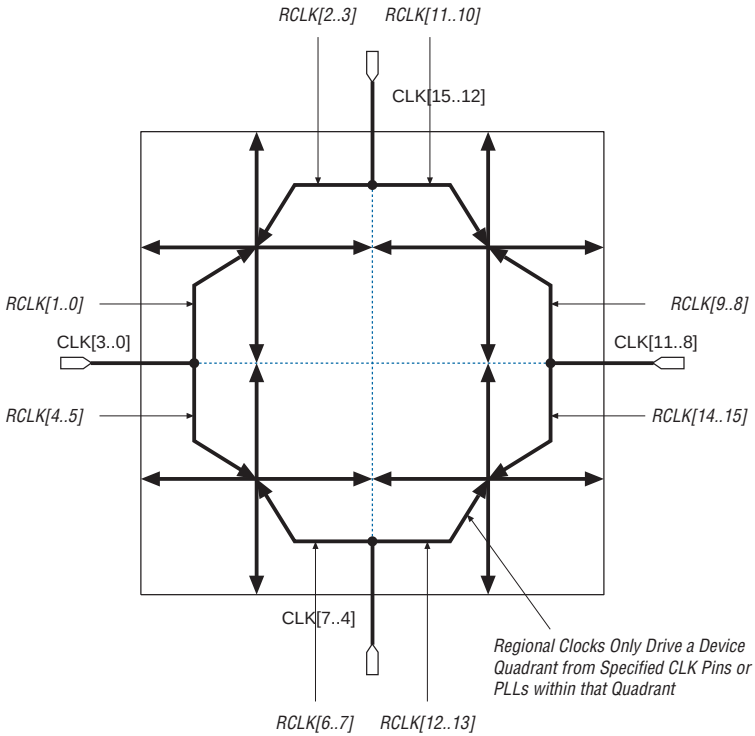


Regional Clock Network

There are four regional clock networks within each quadrant of the Stratix or Stratix GX device that are driven by the same dedicated `CLK[15..0]` input pins or from PLL outputs. From a top view of the silicon, `RCLK[0..3]` are in the top-left quadrant, `RCLK[8..11]` are in the top-right quadrant, `RCLK[4..7]` are in the bottom-left quadrant, and

RCLK[12..15] are in the bottom-right quadrant. The regional clock networks only pertain to the quadrant they drive into. The regional clock networks provide the lowest clock delay and skew for logic contained within a single quadrant. RCLK clock networks cannot be driven by internal logic. The CLK clock pins symmetrically drive the RCLK networks within a particular quadrant, as shown in Figure 1–20. See Figures 1–21 and 1–22 for RCLK connections from PLLs and CLK pins.

Figure 1–20. Regional Clocks



Clock Input Connections

Two CLK pins drive each enhanced PLL. You can use either one or both pins for clock switchover inputs into the PLL. Either pin can be the primary clock source for clock switchover, which is controlled in the Quartus II software. Enhanced PLLs 5 and 6 also have feedback input pins as shown in Table 1–14.

Input clocks for fast PLLs 1, 2, 3, and 4 come from CLK pins. Stratix GX devices use PLLs 3 and 4 in the HSSI block only. A multiplexer chooses one of two possible CLK pins to drive each PLL. This multiplexer is not a clock switchover multiplexer and is only used for clock input connectivity.

Either a FPLLCLK input pin or a CLK pin can drive the fast PLLs in the corners (7, 8, 9, and 10) when used for general purpose. CLK pins cannot drive these fast PLLs in high-speed differential I/O mode. PLLs 9 and 10 are used for the HSSI block in Stratix GX devices and are not available.

Table 1–14 shows which PLLs are available for each Stratix device and which input clock pin drives which PLLs.

Table 1–14. Stratix Clock Input Sources For Enhanced & Fast PLLs (Part 1 of 2)

Clock Input Pins	All Stratix Devices						EP1S30, EP1S40, EP1S60 & EP1S80 Devices Only				EP1S40 *(3)*, EP1S60 & EP1S80 Devices Only	
	PLL 1 *(1)*	PLL 2 *(1)*	PLL 3 *(1)*	PLL 4 *(1)*	PLL 5 *(2)*	PLL 6 *(2)*	PLL 7 *(1)*	PLL 8 *(1)*	PLL 9 *(1)*	PLL 10 *(1)*	PLL 11 *(2)*	PLL 12 *(2)*
CLK0p/n	v						v					
CLK1p/n	v											
CLK2p/n		v						v				
CLK3p/n		v										
CLK4p/n						v						
CLK5p/n						v						
CLK6p/n												v
CLK7p/n												v
CLK8p/n			v						v			
CLK9p/n			v									
CLK10p/n				v						v		
CLK11p/n				v								
CLK12p/n											v	
CLK13p/n											v	
CLK14p/n					v							
CLK15p/n					v							

Table 1–14. Stratix Clock Input Sources For Enhanced & Fast PLLs (Part 2 of 2)

Clock Input Pins	All Stratix Devices						EP1S30, EP1S40, EP1S60 & EP1S80 Devices Only				EP1S40 (3), EP1S60 & EP1S80 Devices Only	
	PLL 1 (1)	PLL 2 (1)	PLL 3 (1)	PLL 4 (1)	PLL 5 (2)	PLL 6 (2)	PLL 7 (1)	PLL 8 (1)	PLL 9 (1)	PLL 10 (1)	PLL 11 (2)	PLL 12 (2)
`FPll7clk`							v					
`FPll8clk`								v				
`FPll9clk`									v			
`FPll10clk`										v		
Clock Feedback Input Pins												
`Pll5_fbp/n`					v							
`Pll6_fbp/n`						v						

Notes to Table 1–14:

(1) This is a fast PLL. The global or regional clocks in a fast PLL's quadrant can drive the fast PLL input. A pin or other PLL must drive the global or regional source. The source cannot be driven by internally generated logic before driving the fast PLL.

(2) This is an enhanced PLL.

(3) The EP1S40 device in the F780 package does not support PLLs 11 and 12.

Clock Output Connections

Enhanced PLLs have outputs for two regional clock outputs and four global outputs. There is line sharing between clock pins, global and regional clock networks and all PLL outputs. Check Tables 1–15 and 1–16 and Figures 1–21 and 1–22 to make sure that the clocking scheme is valid. The Quartus II software automatically maps to regional and global clocks to avoid any restrictions. Enhanced PLLs 5 and 6 drive out to single-ended pins as shown in Table 1–15. PLLs 11 and 12 drive out to single-ended pins.

You can connect each fast PLL 1, 2, 3, or 4 outputs (*g*0, *l*0, and *l*1) to either a global or a regional clock. (PLLs 3 and 4 are not available on Stratix GX devices.) There is line sharing between clock pins, `FPLLCLK` pins, global and regional clock networks and all PLL outputs. Check Figures 1–21 and 1–22 to make sure that the clocking is valid. The Quartus II software automatically maps to regional and global clocks to avoid any restrictions.

Table 1–15 shows the global and regional clocks that each PLL drives outputs to for Stratix devices. Table 1–16 shows the global and regional clock network each of the CLK and FPLLCLK pins drive when bypassing the PLL.

Table 1–15. Stratix Global & Regional Clock Output Line Sharing for Enhanced & Fast PLLs (Part 1 of 2)

Clock Network	All Devices						EP1S30, EP1S40, EP1S60 & EP1S80 Devices Only				EP1S40 (5), EP1S60 & EP1S80 Devices Only	
	PLL 1 (1)	PLL 2 (1)	PLL 3 (1)	PLL 4 (1)	PLL 5 (2)	PLL 6 (2)	PLL 7 (1)	PLL 8 (1)	PLL 9 (1)	PLL 10 (1)	PLL 11 (2)	PLL 12 (2)
GCLK0	v	v					v	v				
GCLK1	v	v					v	v				
GCLK2	v	v					v	v				
GCLK3	v	v					v	v				
GCLK4						v						v
GCLK5						v						v
GCLK6						v						v
GCLK7						v						v
GCLK8			v	v					v	v		
GCLK9			v	v					v	v		
GCLK10			v	v					v	v		
GCLK11			v	v					v	v		
GCLK12					v						v	
GCLK13					v						v	
GCLK14					v						v	
GCLK15					v						v	
RCLK0	v	v					v					
RCLK1	v	v					v					
RCLK2					v						v	
RCLK3					v						v	
RCLK4	v	v						v				
RCLK5	v	v						v				

Table 1–15. Stratix Global & Regional Clock Output Line Sharing for Enhanced & Fast PLLs (Part 2 of 2)

Clock Network	All Devices						EP1S30, EP1S40, EP1S60 & EP1S80 Devices Only				EP1S40 (5), EP1S60 & EP1S80 Devices Only	
	PLL 1 (1)	PLL 2 (1)	PLL 3 (1)	PLL 4 (1)	PLL 5 (2)	PLL 6 (2)	PLL 7 (1)	PLL 8 (1)	PLL 9 (1)	PLL 10 (1)	PLL 11 (2)	PLL 12 (2)
RCLK6						v						v
RCLK7						v						v
RCLK8			v	v						v		
RCLK9			v	v						v		
RCLK10					v						v	
RCLK11					v						v	
RCLK12						v						v
RCLK13						v						v
RCLK14			v	v					v			
RCLK15			v	v					v			
External Clock Output												
PLL5_OUT [3..0]p/n					v							
PLL6_OUT [3..0]p/n						v						
PLL11_OUT (3)											v	
PLL12_OUT (4)												v

Notes to Table 1–15:

(1) This is a fast PLL.
(2) This is an enhanced PLL.
(3) This pin is a tri-purpose pin; it can be an I/O pin, `CLK13n`, or used for PLL 11 output.
(4) This pin is a tri-purpose pin; it can be an I/O pin, `CLK7n`, or used for PLL 12 output.
(5) The EP1S40 device in the F780 package does not support PLLs 11 and 12.

Table 1–16. Stratix CLK & FPLLCLK Input Pin Connections to Global & Regional Clock Networks Note (1)

Clock Network	CLK Pins																FPLLCLK (2)			
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	7	8	9	10
GCLK0	v																v	v		
GCLK1		v															v	v		
GCLK2			v														v	v		
GCLK3				v													v	v		
GCLK4					v															
GCLK5						v														
GCLK6							v													
GCLK7								v												
GCLK8									v										v	v
GCLK9										v									v	v
GCLK10											v								v	v
GCLK11												v							v	v
GCLK12													v							
GCLK13														v						
GCLK14															v					
GCLK15																v				
RCLK0	v																v			
RCLK1		v															v			
RCLK2															v			v		
RCLK3																v		v		
RCLK4			v																	
RCLK5				v																
RCLK6					v															
RCLK7						v														
RCLK8											v								v	
RCLK9												v							v	
RCLK10													v							v
RCLK11														v						v
RCLK12							v													

Table 1–16. Stratix CLK & FPLLCLK Input Pin Connections to Global & Regional Clock Networks Note (1)

Clock Network	CLK Pins																FPLLCLK (2)			
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	7	8	9	10
RCLK13								v												
RCLK14									v											
RCLK15										v										

Notes to Table 1–16:

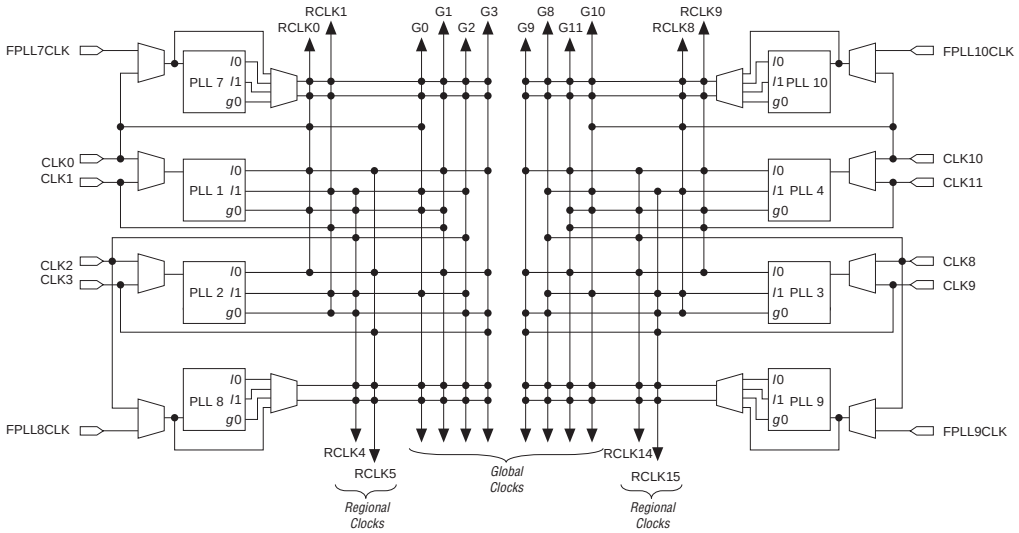
(1) The CLK and FPLLCLK pins cannot drive.

(2) The FPLLCLK pin is only available in EP1S80, EP1S60, EP1S40, and EP1S30 devices.

The fast PLLs also drive high-speed SERDES clocks for differential I/O interfacing. For information on these FPLLCLK pins, see the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter.

Figure 1–21 shows the global and regional clock input and output connections from the enhanced. Figure 1–21 shows graphically the same information as Tables 1–15 and 1–16 but with the added detail of where each specific PLL output port drives to.

Figure 1–21. Global & Regional Clock Connections from Side Clock Pins & Fast PLL Outputs



Notes to Figures 1–21:

(1) The global or regional clocks in a fast PLL's quadrant can drive the fast PLL input. A dedicated pin or other PLL must drive the global or regional source. The source cannot be driven by internally generated logic before driving the fast PLL.

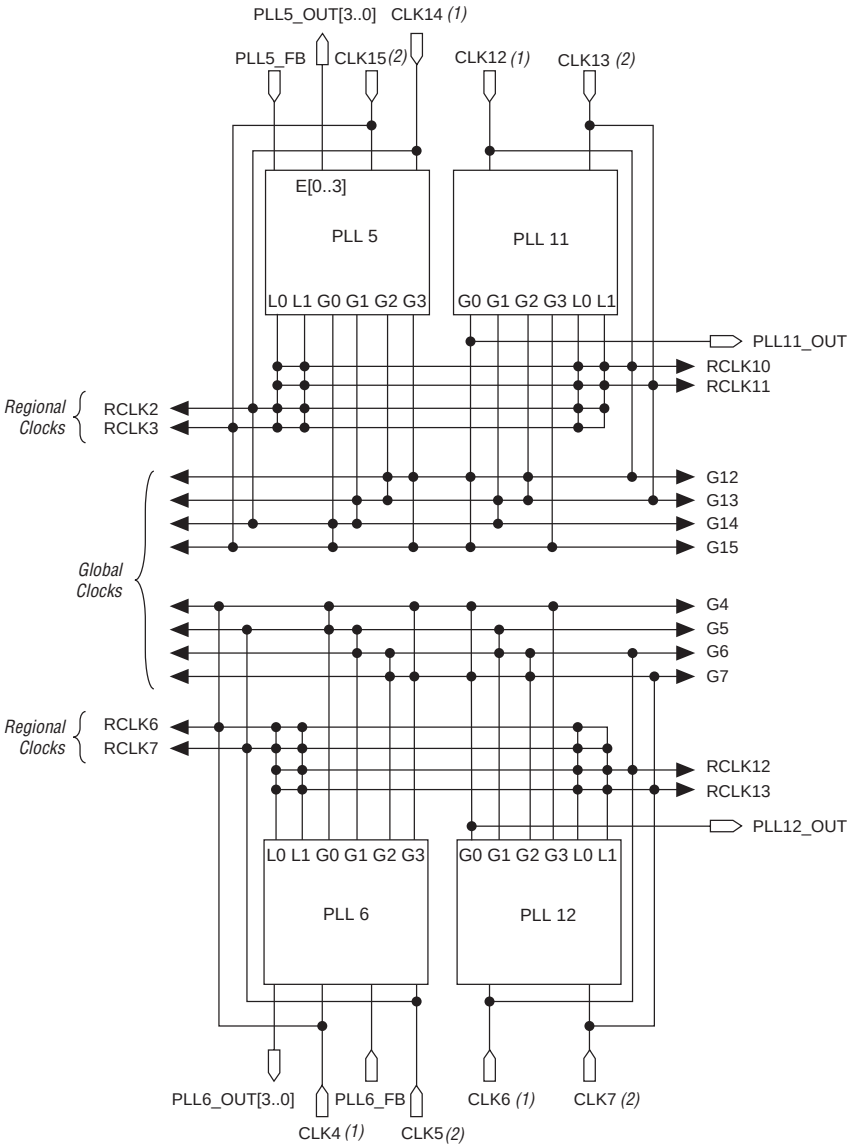
(2) PLLs 3, 4, 9, and 10 are used for the HSSI block in Stratix GX devices and are not available for this use.

When using a fast PLL to compensate for clock delays to drive logic on the chip, the clock delay from the input pin to the clock input port of the PLL is compensated only if the clock is fed by the dedicated input pin closest to the PLL. If the fast PLL gets its input clock from a global or regional clock or from another dedicated clock pin, which does not directly feed the fast PLL, the clock signal is first routed onto a global clock network. The signal then drives into the PLL. In this case, the clock delay is not fully compensated and the delay compensation is equal to the clock delay from the dedicated clock pin closest to the PLL to the clock input port of the PLL.

For example, if you use `CLK0` to feed PLL 7, the input clock path delay is not fully compensated, but if `FPLL7CLK` feeds PLL 7, the input clock path delay is fully compensated.

Figure 1–22 shows the global and regional clock input and output connections from the fast PLLs. Figure 1–22 shows graphically the same information as Tables 1–15 and 1–16 but with the added detail of where each specific PLL output port drives to.

Figure 1–22. Global & Regional Clock Connections from Top Clock Pins & Enhanced PLL Outputs



Notes to Figures 1–22:

(1) CLK4, CLK6, CLK12, and CLK14 feed the corresponding PLL's `inclk0` port.

(2) CLK5, CLK7, CLK13, and CLK15 feed the corresponding PLL's `inclk1` port.

Board Layout

The enhanced and fast PLL circuits in Stratix and Stratix GX devices contain analog components embedded in a digital device. These analog components have separate power and ground pins to minimize noise generated by the digital components. Both Stratix and Stratix GX enhanced and fast PLLs use separate VCC and ground pins to isolate circuitry and improve noise resistance.

VCCA & GNDA

Each enhanced and fast PLL uses separate VCC and ground pin pairs for their analog circuitry. The analog circuit power and ground pin for each PLL is called PLL<*PLL number*>_VCCA and PLL<*PLL number*>_GNDA. Connect the VCCA power pin to a 1.5-V power supply, even if you do not use the PLL. Isolate the power connected to VCCA from the power to the rest of the Stratix and Stratix GX device or any other digital device on the board. You can use one of three different methods of isolating the VCCA pin: separate V_{CCA} power planes, a partitioned V_{CCA} island within the V_{CCINT} plane, and thick V_{CCA} traces.

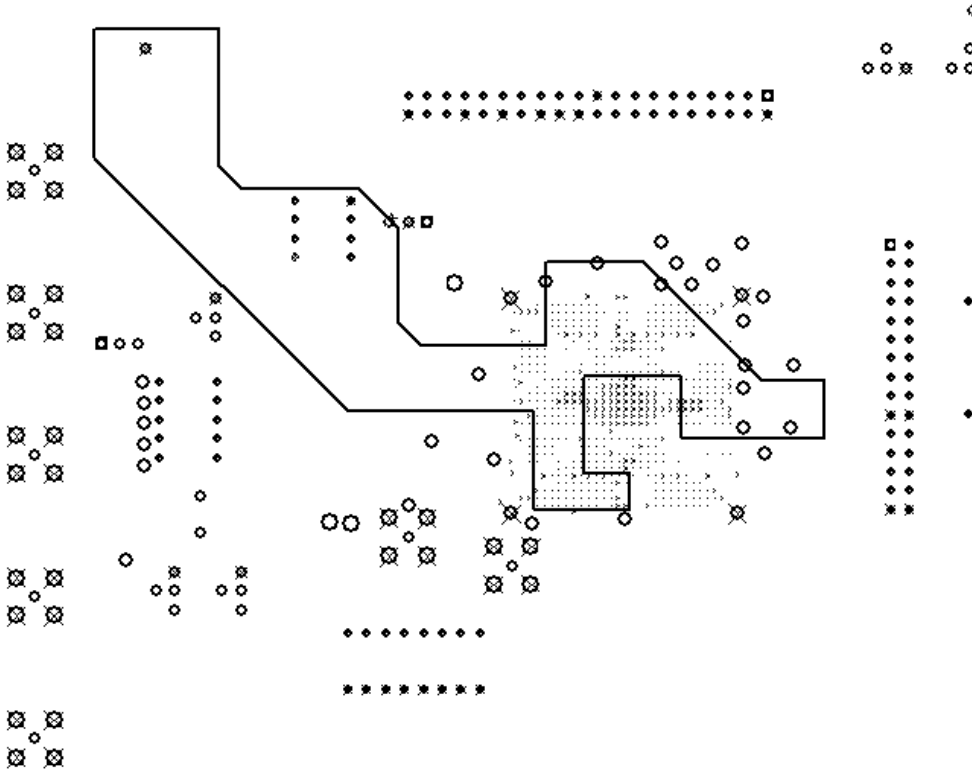
Separate V_{CCA} Power Plane

A mixed signal system is already partitioned into analog and digital sections, each with its own power planes on the board. To isolate the VCCA pin using a separate V_{CCA} power plane, connect the VCCA pin to the analog 1.5-V power plane.

Partitioned V_{CCA} Island within V_{CCINT} Plane

Fully digital systems do not have a separate analog power plane on the board. Because it is expensive to add new planes to the board, you can create islands for VCCA_PLL. Figure 1–23 shows an example board layout with an analog power island. The dielectric boundary that creates the island should be 25 mils thick. Figure 1–23 shows a partitioned plane within V_{CCINT} for V_{CCA}.

Figure 1–23. V_{CCINT} Plane Partitioned for V_{CCA} Island

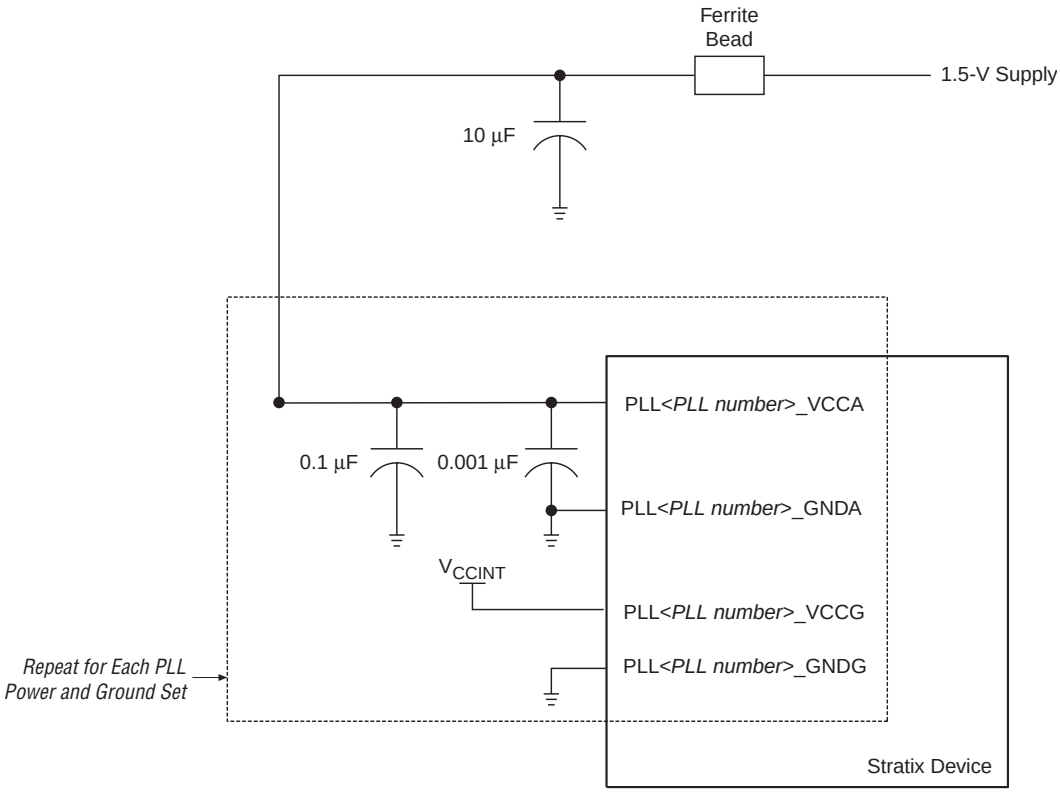


Thick V_{CCA} Trace

Because of board constraints, you might not be able to partition a V_{CCA} island. Instead, run a thick trace from the power supply to each V_{CCA} pin. The traces should be at least 20 mils thick.

In each of these three cases, you should filter each V_{CCA} pin with a decoupling circuit shown in Figure 1–24. Place a ferrite bead that exhibits high impedance at frequencies of 50 MHz or higher and a 10-µF tantalum parallel capacitor where the power enters the board. Decouple each V_{CCA} pin with a 0.1-µF and 0.001-µF parallel combination of ceramic capacitors located as close as possible to the Stratix or Stratix GX device. You can connect the GNDA pins directly to the same ground plane as the device's digital ground.

Figure 1–24. PLL Power Schematic for Stratix or Stratix GX PLLs



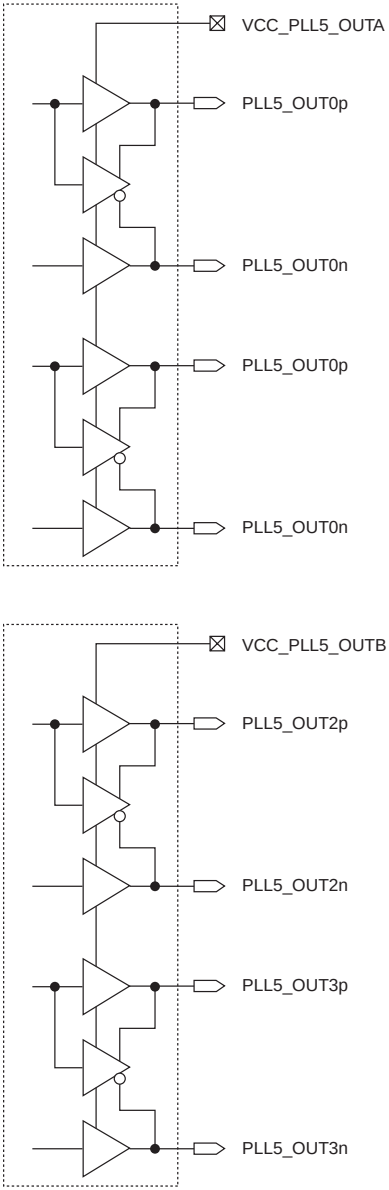
VCCG & GNDG

The guard ring power and ground pins are called PLL<*PLL number*>_VCCG and PLL<*PLL number*>_GNDG. The guard ring isolates the PLL circuit from the rest of the device. Connect these guard ring V_{CCG} pins to the quietest digital supply on the board. In most systems, this is the digital 1.5-V supply supplied to the device's V_{CCINT} pins. Connect the V_{CCG} pins to a power supply even if you do not use the PLL. You can connect the GNDG pins directly to the same ground plane as the device's digital ground. See Figure 1–24.

External Clock Output Power

Enhanced PLLs 5 and 6 also have isolated power pins for their dedicated external clock outputs (VCC_PLL5_OUTA and VCC_PLL5_OUTB, or VCC_PLL6_OUTA and VCC_PLL6_OUTB, respectively). PLLs 5 and 6 both have two banks of outputs. Each bank is powered by a unique output power, OUTA or OUTB, as illustrated in Figure 1–25. These outputs can by powered by 3.3, 2.5, 1.8, or 1.5 V depending on the I/O standard for the clock output in the A or B groups.

Figure 1–25. External Clock Output Pin Association to Output Power Note (1)

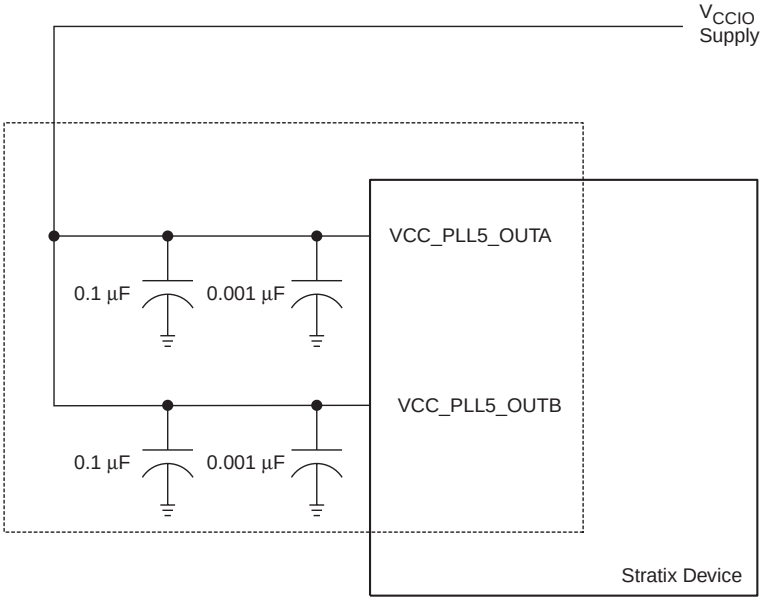


Note to Figure 1–25:

(1) These pins apply to PLL 5. The figure for PLL 6 is similar, except that the pin names begin with the prefix PLL6 instead of PLL5.

Filter each isolated power pin with a decoupling circuit shown in Figure 1–26. Decouple the isolated power pins with a 0.1-µF and a 0.001-µF parallel combination of ceramic capacitors located as close as possible to the Stratix device.

Figure 1–26. Stratix PLL External Clock Output Power Ball Connections
Note (1)



Note to Figure 1–26:
(1) Figure 1–26 also applies to VCC_PLL6_OUTA/B.

Guidelines

Use the following guidelines for optimal jitter performance on the external clock outputs from enhanced PLLs 5 and 6. If all outputs are running at the same frequency, these guidelines are not necessary to improve performance.

- When driving two or more clock outputs from PLL 5 or 6, separate the outputs into the two groups shown in Figure 1–24. For example, if you are driving 100- and 200-MHz clock outputs off-chip from PLL 5, place one output on `PLL5_OUT0p` (powered by `VCC_PLL5_OUTA`) and the other output on `PLL5_OUT2p` (powered by `VCC_PLL5_OUTB`). Since the output buffers are powered by different pins, they are less susceptible to bimodal jitter. Bimodal jitter is a deterministic jitter not caused by the PLL but rather by coincident edges of clock outputs that are multiples of each other.
- Use phase shift to ensure edges are not coincident on all the clock outputs.
- Use phase shift to skew clock edges with respect to each other for best jitter performance.

  1 Delay shift (time delay elements) are no longer supported in Stratix PLLs. Use the phase shift feature to implement the desired time shift.

- If you cannot drive multiple clocks of different frequencies and phase shifts or isolate banks, you should control the drive capability on the lower frequency clock. Reducing how much current the output buffer has to supply can reduce the noise. Minimize capacitive load on the slower frequency output and configure the output buffer to drive slow slew rate and lower current strength. The higher frequency output should have an improved performance, but this may degrade the performance of your lower frequency clock output.

Conclusion

Stratix and Stratix GX device enhanced PLLs provide you with complete control of your clocks and system timing. These PLLs are capable of offering flexible system level clock management that was previously only available in discrete PLL devices. The embedded PLLs meet and exceed the features offered by these high-end discrete devices, reducing the need for other timing devices in the system.

Section II. Memory

This section provides information on the TriMatrix™ Embedded Memory blocks internal to Stratix® devices and the supported external memory interfaces.

It contains the following chapters:

- Chapter 2, TriMatrix Embedded Memory Blocks in Stratix & Stratix GX Devices
- Chapter 3, External Memory Interfaces in Stratix & Stratix GX Devices

The *QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices* chapter is removed in this version of the *Stratix Device Handbook*. The information is available in *AN 349: Interfacing QDR SRAM with Stratix and Stratix GX Devices*.

Revision History

The table below shows the revision history for Chapters 2 and 3.

Chapter	Date/Version	Changes Made	Comments
2	July 2005, v3.3	• Updated "Implementing True Dual-Port Mode" section.	
	January 2005, v3.2	• Minor technical content update.	
	September 2004, v3.1	• Updated Note 1 in Figure 2–12 on page 2–22. • Updated description about using two different clocks in a dual-port RAM on page 2–27. • Deleted description of M-RAM block and document references on page 2–27.	
	April 2004, v3.0	• Synchronous occurrences are renamed to pipelined. • Pseudo-synchronous occurrences are renamed flow-through. • Added AND gate to Figure 2–12.	
	July 2003, v2.0	• Updated performance specification for TriMatrix memory in Table 2-1. • Added addressing example for a RAM that is using mixed-width mode, page 2-9. • Added Note 1 to Tables 2-9 and 2-10, Note 3 to Figure 2-11, and Note 2 to Figures 2-12 and 2-13.	

Chapter	Date/Version	Changes Made	Comments
3	June 2006, v3.3	• Changed the name of the chapter from External Memory Interfaces to External Memory Interfaces in Stratix & Stratix GX Devices to reflect its shared status between those device handbooks. • Added cross reference regarding frequency limits for 72 and 90° phase shift for DQS.	
	July 2005, v3.2	• Updated mathematical symbols in Table 3–3. • Updated "DQS Phase-Shift Circuitry" section.	
	September 2004, v3.1	• Moved Figure 8 to become Figure 1, "Example of Where a DQS Signal is Center-Aligned in the IOE" on page 3–3. • Updated Table 3–1 on page 3–10, updated Note 4. Note 4, 5, and 6, are now Note 5, 6, and 7, respectively. • Updated Table 3–2 on page 3–10. • Updated Table 3–3 on page 3–13. • Updated Note on page 3–14. • Moved the "External Memory Standards" on page 3–1 to follow the Introduction section. • Moved "Conclusion" on page 3–27 to end of chapter.	
	April 2004, v3.0	• Chapter renamed Chapter 3, External Memory Interfaces in Stratix & Stratix GX Devices. • Table 3–1: DDR SDRAM - side banks row added, ZBT SRAM row updated. • Added Tables 3–2 and 3–4. • DQSn pins removed (page 3-5) • Deleted "QDR SRAM Interfacing" figure. • Replaced "t_{ZX} & t_{XZ} Timing Diagram."	
	November 2003, v2.1	• Removed support for series and parallel on-chip termination.	
	July 2003, v2.0	• altddio_bidir function is used for DQS in versions before Quartus II 3.0. (page 3-2) • Updated naming convention for DQS pins on page 3-9 to match pin tables. • Clarified input clock to PLL must come from an external input pin on page 3-12.	

2. TriMatrix Embedded Memory Blocks in Stratix & Stratix GX Devices

S52003-3.3

Introduction

Stratix® and Stratix GX devices feature the TriMatrix™ memory structure, composed of three sizes of embedded RAM blocks. TriMatrix memory includes 512-bit M512 blocks, 4-Kbit M4K blocks, and 512-Kbit M-RAM blocks, each of which is configurable to support a wide range of features. Offering up to 10 Mbits of RAM and up to 12 terabits per second of device memory bandwidth, the TriMatrix memory structure makes the Stratix and Stratix GX families ideal for memory-intensive applications.

TriMatrix Memory

TriMatrix memory structures can implement a wide variety of complex memory functions. For example, use the small M512 blocks for first-in first-out (FIFO) functions and clock domain buffering where memory bandwidth is critical. The M4K blocks are an ideal size for applications requiring medium-sized memory, such as asynchronous transfer mode (ATM) cell processing. M-RAM blocks enhance programmable logic device (PLD) memory capabilities for large buffering applications, such as internet protocol (IP) packet buffering and system cache.

TriMatrix memory blocks support various memory configurations, including single-port, simple dual-port, true dual-port (also known as bidirectional dual-port), shift-register, ROM, and FIFO mode. The TriMatrix memory architecture also includes advanced features and capabilities, such as byte enable support, parity-bit support, and mixed-port width support. This chapter describes the various TriMatrix memory modes and features.

Table 2–1 summarizes the features supported by the three sizes of TriMatrix memory.

f For more information on selecting which memory block to use, see *AN 207: TriMatrix Memory Selection Using the Quartus II Software.*

Table 2–1. Summary of TriMatrix Memory Features

Feature	M512 Block	M4K Block	M-RAM Block
Performance	319 MHz	290 MHz	287 MHz
Total RAM bits (including parity bits)	576	4,608	589,824
Configurations	512 × 1 256 × 2 128 × 4 64 × 8 64 × 9 32 × 16 32 × 18	4K × 1 2K × 2 1K × 4 512 × 8 512 × 9 256 × 16 256 × 18 128 × 32 128 × 36	64K × 8 64K × 9 32K × 16 32K × 18 16K × 32 16K × 36 8K × 64 8K × 72 4K × 128 4K × 144
Parity bits	v	v	v
Byte enable		v	v
Single-port memory	v	v	v
Simple dual-port memory	v	v	v
True dual-port memory		v	v
Embedded shift register	v	v	
ROM	v	v	
FIFO buffer	v	v	v
Simple dual-port mixed width support	v	v	v
True dual-port mixed width support		v	v
Memory initialization file (**.mif**)	v	v	
Mixed-clock mode	v	v	v
Power-up condition	Outputs cleared	Outputs cleared	Outputs unknown
Register clears	Input and output registers *(1)*	Input and output registers *(2)*	Output registers
Same-port read-during-write	New data available at positive clock edge	New data available at positive clock edge	New data available at positive clock edge
Mixed-port read-during-write	Outputs set to unknown or old data	Outputs set to unknown or old data	Unknown output

Notes to Table 2–1:

(1) The `rden` register on the M512 memory block does not have a clear port.

(2) On the M4K block, asserting the clear port of the `rden` and byte enable registers drives the output of these registers high.

The extremely high memory bandwidth of the Stratix and Stratix GX device families is a result of increased memory capacity and speed. Table 2–2 shows the memory capacity for TriMatrix memory blocks in each Stratix device. Table 2–3 shows the memory capacity for TriMatrix memory blocks in each Stratix GX device.

Table 2–2. TriMatrix Memory Distribution in Stratix Devices

Device	M512 Columns/Blocks	M4K Columns/Blocks	M-RAM Blocks	Total RAM Bits
EP1S10	4 / 94	2 / 60	1	920,448
EP1S20	6 / 194	2 / 82	2	1,669,248
EP1S25	6 / 224	3 / 138	2	1,944,576
EP1S30	7 / 295	3 / 171	4	3,317,184
EP1S40	8 / 384	3 / 183	4	3,423,744
EP1S60	10 / 574	4 / 292	6	5,215,104
EP1S80	11 / 767	4 / 364	9	7,427,520

Table 2–3. TriMatrix Memory Distribution in Stratix GX Devices

Device	M512 Columns/Blocks	M4K Columns/Blocks	M-RAM Blocks	Total RAM Bits
EP1SGX10	4 / 94	2 / 60	1	920,448
EP1SGX25	6 / 224	3 / 138	2	1,944,576
EP1SGX40	8 / 384	3 / 183	4	3,423,744

Clear Signals

When applied to input registers, the asynchronous clear signal for the TriMatrix embedded memory immediately clears the input registers. However, the output of the memory block does not show the effects until the next clock edge. When applied to output registers, the asynchronous clear signal clears the output registers and the effects are seen immediately.

Parity Bit Support

The memory blocks support a parity bit for each byte. Parity bits are in addition to the amount of memory in each RAM block. For example, the M512 block has 576 bits, 64 of which are optionally used for parity bit

storage. The parity bit, along with logic implemented in logic elements (LEs), can implement parity checking for error detection to ensure data integrity. Parity-size data words can also store user-specified control bits.

Byte Enable Support

In the M4K and M-RAM blocks, byte enables can mask the input data so that only specific bytes of data are written. The unwritten bytes retain the previous written value. The write enable signals (`wren`), in conjunction with the byte enable signals (`byteena`), controls the RAM block's write operations. The default value for the `byteena` signals is high (enabled), in which case writing is controlled only by the `wren` signals.

Asserting the clear port of the byte enable registers drives the byte enable signals to their default high level.

M4K Blocks

M4K blocks support byte writes when the write port has a data width of 16, 18, 32, or 36 bits. Table 2–4 summarizes the byte selection.

Table 2–4. Byte Enable for M4K Blocks *Notes (1), (2)*

byteena	datain × 18	datain × 36
[0] = 1	[8..0]	[8..0]
[1] = 1	[17..9]	[17..9]
[2] = 1	–	[26..18]
[3] = 1	–	[35..27]

Notes to Table 2–4:

(1) Any combination of byte enables is possible.

(2) Byte enables can be used in the same manner with 8-bit words, i.e., in × 16 and × 32 modes.

M-RAM Blocks

M-RAM blocks support byte enables for the ×16, ×18, ×32, ×36, ×64, and ×72 modes. In the ×128 or ×144 simple dual-port mode, the two sets of `byteena` signals (`byteena_a` and `byteena_b`) combine to form the necessary 16 byte enables. Tables 2–5 and 2–6 summarize the byte selection.

Table 2–5. Byte Enable for M-RAM Blocks *Notes (1), (2)*

byteena	datain ×18	datain ×36	datain ×72
[0] = 1	[8..0]	[8..0]	[8..0]
[1] = 1	[17..9]	[17..9]	[17..9]
[2] = 1	–	[26..18]	[26..18]
[3] = 1	–	[35..27]	[35..27]
[4] = 1	–	–	[44..36]
[5] = 1	–	–	[53..45]
[6] = 1	–	–	[62..54]
[7] = 1	–	–	[71..63]

Notes to Table 2–5:

(1) Any combination of byte enables is possible.

(2) Byte enables can be used in the same manner with 8-bit words, that is, in ×16, ×32, and ×64 modes.

Table 2–6. M-RAM Combined Byte Selection for ×144 Mode (Part 1 of 2), *Notes (1), (2)*

byteena_a	datain ×144
[0] = 1	[8..0]
[1] = 1	[17..9]
[2] = 1	[26..18]
[3] = 1	[35..27]
[4] = 1	[44..36]
[5] = 1	[53..45]
[6] = 1	[62..54]
[7] = 1	[71..63]
[8] = 1	[80..72]
[9] = 1	[89..81]
[10] = 1	[98..90]
[11] = 1	[107..99]

Table 2–6. M-RAM Combined Byte Selection for ×144 Mode (Part 2 of 2), *Notes (1), (2)*

byteena_a	datain ×144
[12] = 1	[116..108]
[13] = 1	[125..117]
[14] = 1	[134..126]
[15] = 1	[143..135]

Notes to Table 2–6:

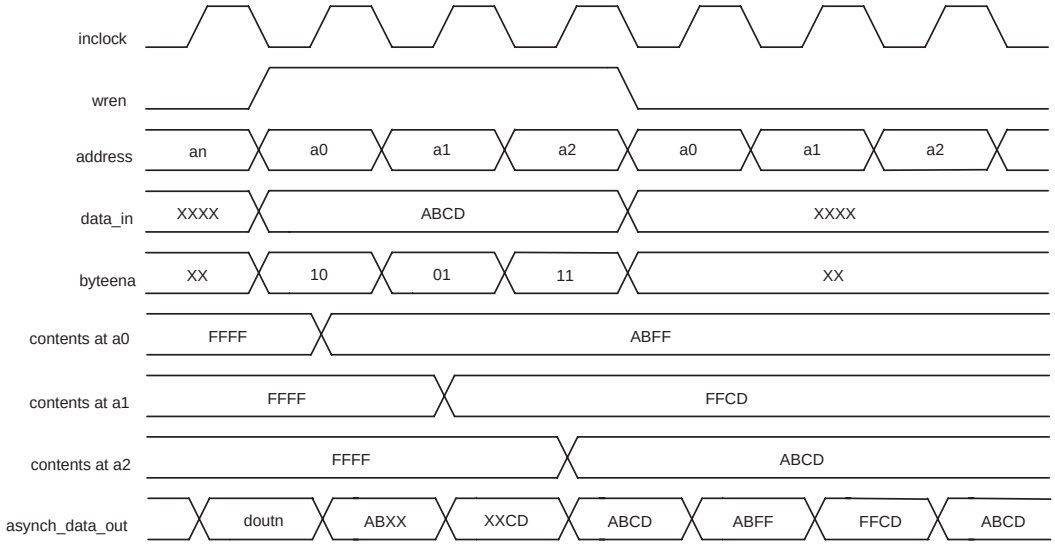
(1) Any combination of byte enables is possible.

(2) Byte enables can be used in the same manner with 8-bit words, i.e., in ×16, ×32, ×64, and ×128 modes.

Byte Enable Functional Waveform

Figure 2–1 shows how both the `wren` and the `byteena` signals control the write operations of the RAM.

Figure 2–1. Byte Enable Functional Waveform *Note (1)*



Note to Figure 2–1:

(1) For more information on simulation output when a read-during-write occurs at the same address location, see "Read-During-Write Operation at the Same Address" on page 2–25.

Using TriMatrix Memory

The TriMatrix memory blocks include input registers that synchronize writes and output registers to pipeline designs and improve system performance. All TriMatrix memory blocks are pipelined, meaning that all inputs are registered, but outputs are either registered or combinatorial. TriMatrix memory can emulate a flow-through memory by using combinatorial outputs.

f For more information, see *AN 210: Converting Memory from Asynchronous to Synchronous for Stratix & Stratix GX Designs*.

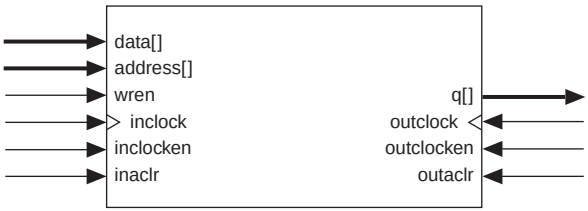
Depending on the TriMatrix memory block type, the memory can have various modes, including:

- Single-port
- Simple dual-port
- True dual-port (bidirectional dual-port)
- Shift-register
- ROM
- FIFO

Implementing Single-Port Mode

Single-port mode supports non-simultaneous reads and writes. Figure 2–2 shows the single-port memory configuration for TriMatrix memory. All memory block types support the single-port mode.

Figure 2–2. Single-Port Memory *Note (1)*



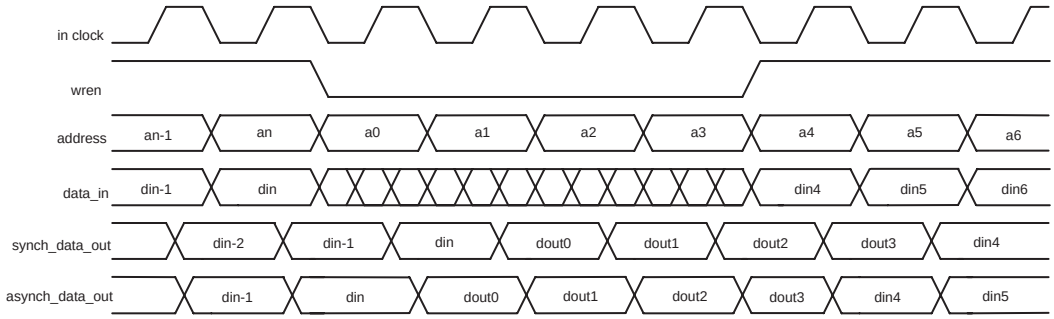
Note to Figure 2–2:
(1) Two single-port memory blocks can be implemented in a single M4K block.

M4K memory blocks can also be divided in half and used for two independent single-port RAM blocks. The Altera Quartus II software automatically uses this single-port memory packing when running low on memory resources. To force two single-port memories into one M4K block, first ensure that each of the two independent RAM blocks is equal to or less than half the size of the M4K block. Second, assign both single-port RAMs to the same M4K block.

In the single-port RAM configuration, the outputs can only be in read-during-write mode, which means that during the write operation, data written to the RAM flows through to the RAM outputs. When the output registers are bypassed, the new data is available on the rising edge of the same clock cycle it was written on. For more information about read-during-write mode, see "Read-During-Write Operation at the Same Address" on page 2–25.

Figure 2–3 shows timing waveforms for read and write operations in single-port mode.

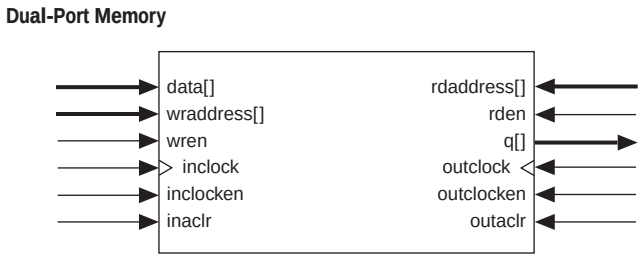
Figure 2–3. Single-Port Timing Waveforms



Implementing Simple Dual-Port Mode

Simple dual-port memory supports a simultaneous read and write. Figure 2–4 shows the simple dual-port memory configuration for TriMatrix memory. All memory block types support this configuration.

Figure 2–4. Simple Dual-Port Memory *Note (1)*



Note to Figure 2–4:

(1) Simple dual-port RAM supports read / write clock mode in addition to the input / output clock mode shown.

TriMatrix memory supports mixed-width configurations, allowing different read and write port widths. When using mixed-width mode, the LSB is written to or read from first. For example, take a RAM that is set up in mixed-width mode with write data width ×8 and read data width ×2. If a binary 00000001 is written to write dress 0, the following is read out of the ×2 output side:

Read Address	×2 data
00	01(LSB of ×8 data)
01	00
10	00
11	00(MSB of ×8 data)

Tables 2–7 to 2–9 show the mixed width configurations for the M512, M4K, and M-RAM blocks, respectively.

Table 2–7. M512 Block Mixed-Width Configurations (Simple Dual-Port Mode)

Read Port	Write Port						
	512 × 1	256 × 2	128 × 4	64 × 8	32 × 16	64 × 9	32 × 18
512 × 1	v	v	v	v	v		
256 × 2	v	v	v	v	v		
128 × 4	v	v	v		v		
64 × 8	v	v		v			
32 × 16	v	v	v		v		
64 × 9						v	
32 × 18							v

Table 2–8. M4K Block Mixed-Width Configurations (Simple Dual-Port Mode) (Part 1 of 2)

Read Port	Write Port								
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	128 × 32	512 × 9	256 × 18	128 × 36
4K × 1	v	v	v	v	v	v			
2K × 2	v	v	v	v	v	v			
1K × 4	v	v	v	v	v	v			
512 × 8	v	v	v	v	v	v			
256 × 16	v	v	v	v	v	v			

Table 2–8. M4K Block Mixed-Width Configurations (Simple Dual-Port Mode) (Part 2 of 2)

Read Port	Write Port								
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	128 × 32	512 × 9	256 × 18	128 × 36
128 × 32	v	v	v	v	v	v			
512 × 9							v	v	v
256 × 18							v	v	v
128 × 36							v	v	v

Table 2–9. M-RAM Block Mixed-Width Configurations (Simple Dual-Port Mode)

Read Port	Write Port				
	64K × 9	32K × 18	16K × 36	8K × 72	4K × 144
64K × 9	v	v	v	v	
32K × 18	v	v	v	v	
16K × 36	v	v	v	v	
8K × 72	v	v	v	v	
4K × 144					v

M512 blocks support serializer and deserializer (SERDES) applications. By using the mixed-width support in combination with double data rate (DDR) I/O standards, the block can function as a SERDES to support low-speed serial I/O standards using global or regional clocks.

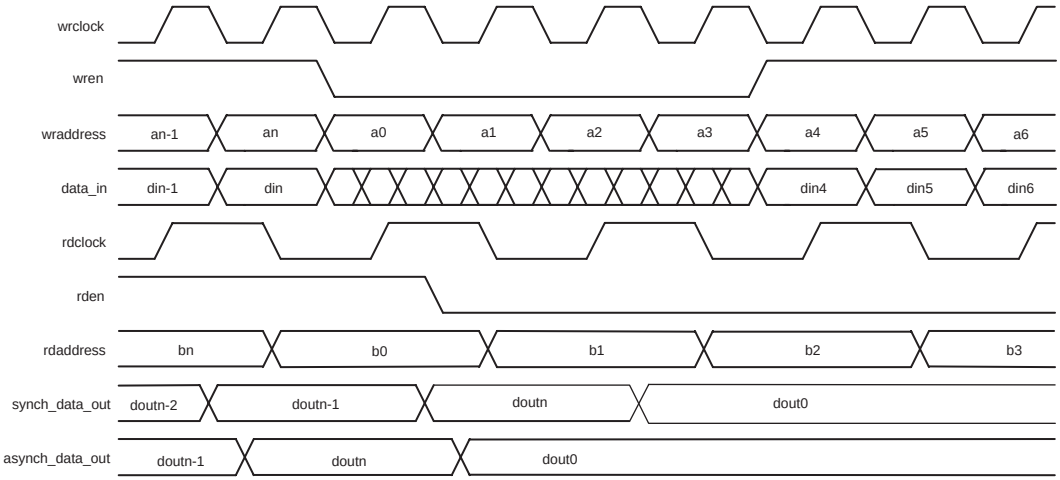
f For more information on Stratix device I/O structure see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1*. For more information on Stratix GX device I/O structure see the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

In simple dual-port mode, the M512 and M4K blocks have one write enable and one read enable signal. The M512 does not support a clear port on the `rden` register. On the M4K block, asserting the clear port of the `rden` register drives `rden` high, which allows the read operation to occur. When the read enable is deactivated, the current data is retained at the output ports. If the read enable is activated during a write operation with the same address location selected, the simple dual-port RAM output is either unknown or can be set to output the old data stored at the memory address. For more information, see "Read-During-Write Operation at the Same Address" on page 2–25.

M-RAM blocks have one write enable signal in simple dual-port mode. To perform a write operation, the write enable is held high. The M-RAM block is always enabled for read operation. If the read address and the write address select the same address location during a write operation, the M-RAM block output is unknown.

Figure 2–5 shows timing waveforms for read and write operations in simple dual-port mode.

Figure 2–5. Simple Dual-Port Timing Waveforms *Note (1)*



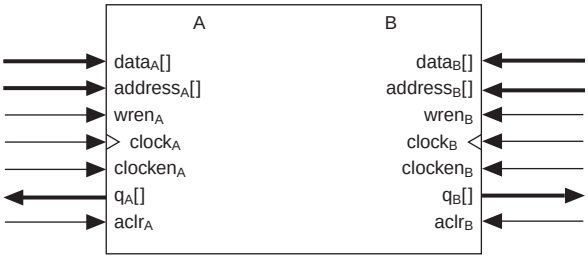
Note to Figure 2–5:

(1) The `rden` signal is not available in the M-RAM block. A M-RAM block in simple dual-port mode is always reading out the data stored at the current read address location.

Implementing True Dual-Port Mode

M4K and M-RAM blocks offer a true dual-port mode to support any combination of two-port operations: two reads, two writes, or one read and one write at two different clock frequencies. Figure 2–6 shows the true dual-port memory configuration for TriMatrix memory.

Figure 2–6. True Dual-Port Memory *Note (1)*



Note to Figure 2–6:

(1) True dual-port memory supports input/output clock mode in addition to the independent clock mode shown.

The widest bit configuration of the M4K and M-RAM blocks in true dual-port mode is 256 × 16-bit (× 18-bit with parity) and 8K × 64-bit (× 72-bit with parity), respectively. The 128 × 32-bit (× 36-bit with parity) configuration of the M4K block and the 4K × 128-bit (× 144-bit with parity) configuration of the M-RAM block are unavailable because the number of output drivers is equivalent to the maximum bit width of the respective memory block. Because true dual-port RAM has outputs on two ports, the maximum width of the true dual-port RAM equals half of the total number of output drivers. Tables 2–10 and 2–11 list the possible M4K RAM block and M-RAM block configurations, respectively.

Table 2–10. M4K Block Mixed-Port Width Configurations (True Dual-Port)

Port A	Port B						
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	512 × 9	256 × 18
4K × 1	v	v	v	v	v		
2K × 2	v	v	v	v	v		
1K × 4	v	v	v	v	v		
512 × 8	v	v	v	v	v		
256 × 16	v	v	v	v	v		
512 × 9						v	v
256 × 18						v	v

Table 2–11. M-RAM Block Mixed-Port Width Configurations (True Dual-Port)

Port A	Port B			
	64K × 9	32K × 18	16K × 36	8K × 72
64K × 9	v	v	v	v
32K × 18	v	v	v	v
16K × 36	v	v	v	v
8K × 72	v	v	v	v

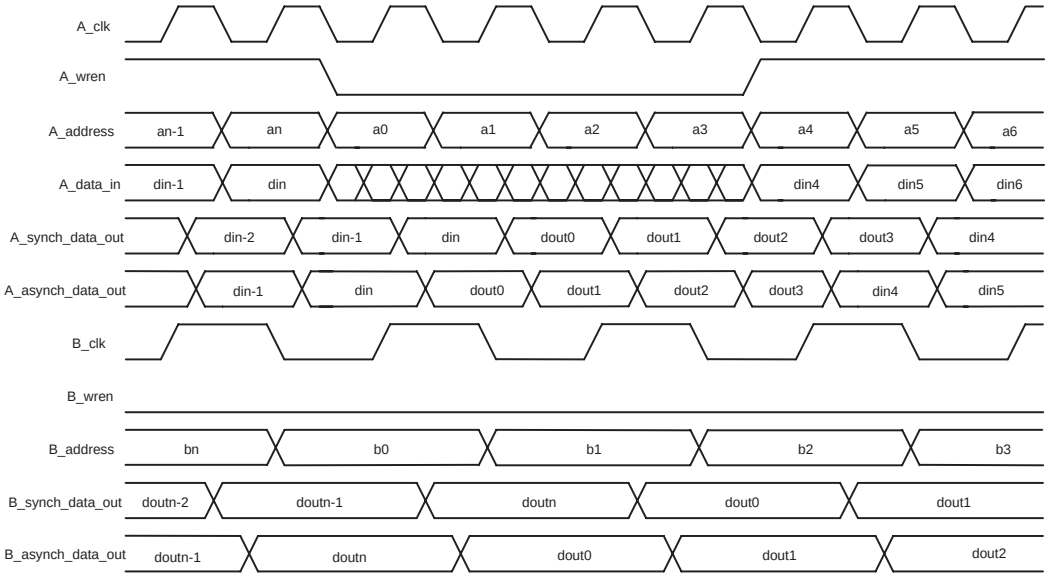
In true dual-port configuration, the RAM outputs can only be configured for read-during-write mode. This means that during write operation, data being written to the A or B port of the RAM flows through to the A or B outputs, respectively. When the output registers are bypassed, the new data is available on the rising edge of the same clock cycle it was written on. For waveforms and information on mixed-port read-during-write mode, see "Read-During-Write Operation at the Same Address" on page 2–25.

Potential write contentions must be resolved external to the RAM because writing to the same address location at both ports results in unknown data storage at that location. Data is written on the rising edge of the write clock for the M-RAM block. For a valid write operation to the same address of the M-RAM block, the rising edge of the write clock for port A must occur following the maximum write cycle time interval after the rising edge of the write clock for port B. Since data is written into the M512 and M4K blocks at the falling edge of the write clock, the rising edge of the write clock for port A should occur following half of the maximum write cycle time interval after the falling edge of the write clock for port B. If this timing is not met, the data stored in that particular address is invalid.

f See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1* for the maximum synchronous write cycle time.

Figure 2–7 shows true dual-port timing waveforms for write operation at port A and read operation at port B.

Figure 2–7. True Dual-Port Timing Waveforms



Implementing Shift-Register Mode

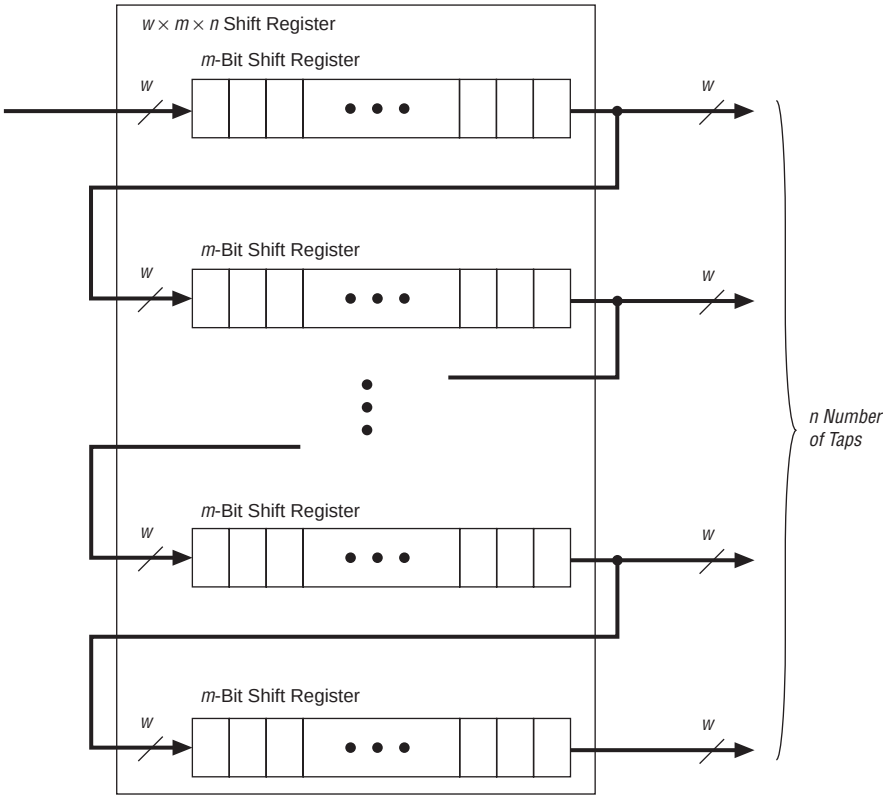
Embedded memory block configurations can implement shift registers for digital signal processing (DSP) applications, such as finite impulse response (FIR) filters, pseudo-random number generators, multi-channel filtering, and auto-correlation and cross-correlation functions. These and other DSP applications require local data storage, traditionally implemented with standard flip-flops that can quickly consume many logic cells for large shift registers. A more efficient alternative is to use embedded memory as a shift register block, which saves logic cell and routing resources and provides a more efficient implementation.

The size of a $(w \times m \times n)$ shift register is determined by the input data width (w), the length of the taps (m), and the number of taps (n). The size of a $(w \times m \times n)$ shift register must be less than or equal to the maximum number of memory bits in the respective block: 576 bits for the M512 block and 4,608 bits for the M4K block. In addition, the size of $w \times n$ must be less than or equal to the maximum width of the respective block: 18 bits for the M512 block and 36 bits for the M4K block. If a larger shift register is required, the memory blocks can be cascaded together.

1 M-RAM blocks do not support the shift-register mode.

Data is written into each address location at the falling edge of the clock and read from the address at the rising edge of the clock. The shift-register mode logic automatically controls the positive and negative edge clocking to shift the data in one clock cycle. Figure 2–8 shows the TriMatrix memory block in the shift-register mode.

Figure 2–8. Shift-Register Memory Configuration



Implementing ROM Mode

The M512 and the M4K blocks support ROM mode. Use a memory initialization file (**.mif**) to initialize the ROM contents of M512 and M4K blocks. The M-RAM block does not support ROM mode.

All Stratix memory configurations must have synchronous inputs; therefore, the address lines of the ROM are registered. The outputs can be registered or combinatorial. The ROM read operation is identical to the read operation in the single-port RAM configuration.

Implementing FIFO Buffers

While the small M512 memory blocks are ideal for designs with many shallow FIFO buffers, all three memory sizes support FIFO mode.

All memory configurations have synchronous inputs; however, the FIFO buffer outputs are always combinatorial. Simultaneous read and write from an empty FIFO is not supported.

Clock Modes

Depending on the TriMatrix memory mode, independent, input / output, read / write, and / or single-port clock modes are available. Table 2–12 shows the clock modes supported by the TriMatrix memory modes.

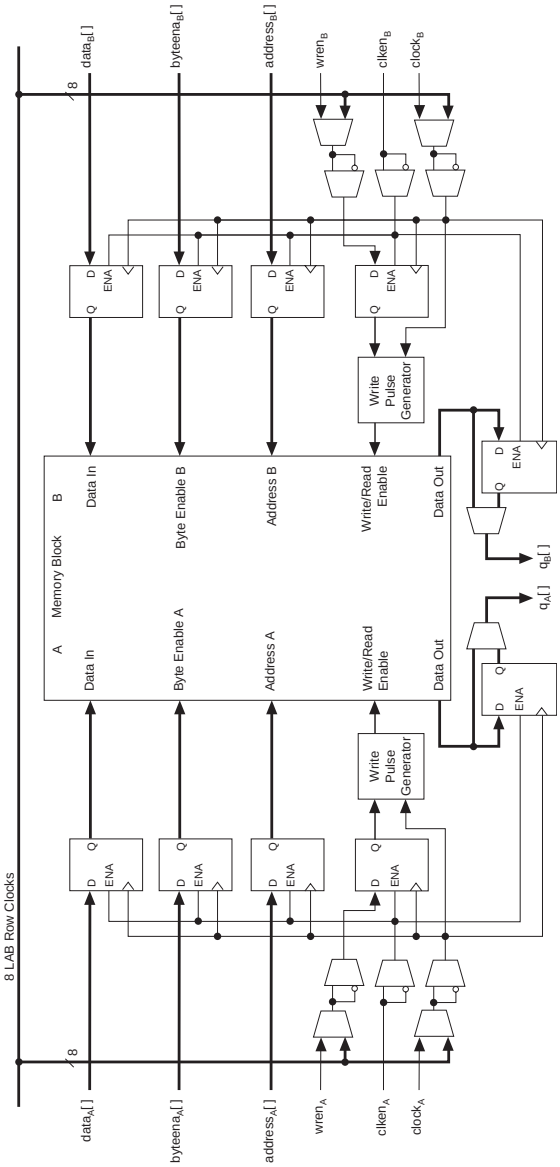
Table 2–12. TriMatrix Memory Clock Modes

Clocking Mode	True-Dual Port Mode	Simple Dual-Port Mode	Single-Port Mode
Independent	v		
Input/output	v	v	
Read/write		v	
Single-port			v

Independent Clock Mode

The TriMatrix memory blocks can implement independent clock mode for true dual-port memory. In this mode, a separate clock is available for each port (A and B). Clock A controls all registers on the port A side, while clock B controls all registers on the port B side. Each port also supports independent clock enables and asynchronous clear signals for port A and B registers. Figure 2–9 shows a TriMatrix memory block in independent clock mode.

Figure 2–9. Independent Clock Mode *Note (1), (2)*



Note to Figure 2–9:

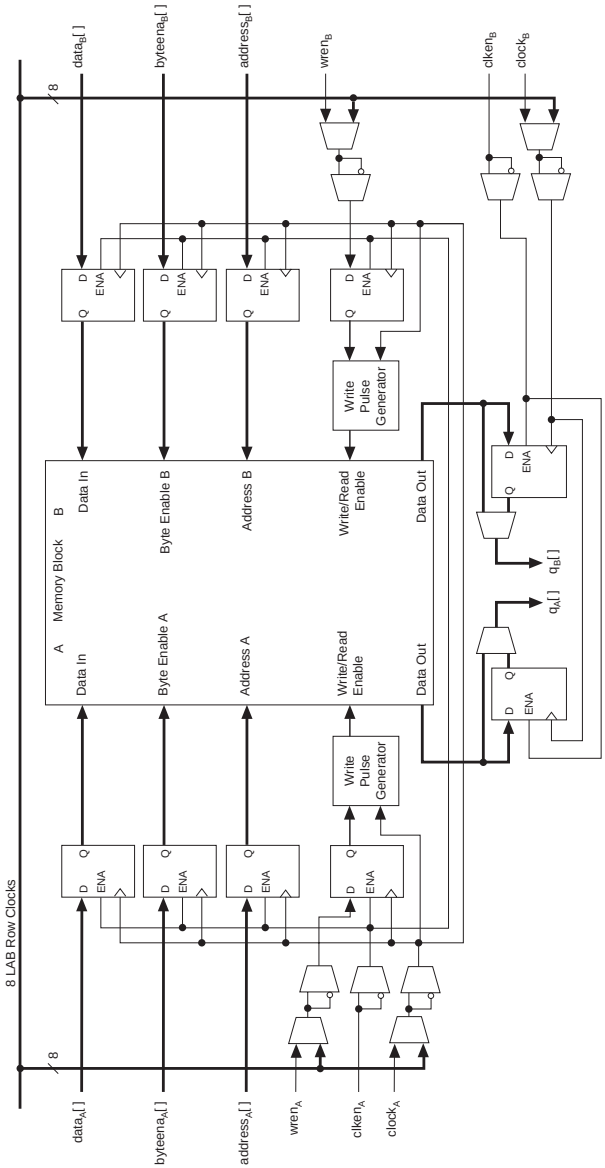
(1) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

(2) All registers shown have asynchronous clear ports, except when using the M-RAM. M-RAM blocks have asynchronous clear ports on their output registers only.

Input/Output Clock Mode

The TriMatrix memory blocks can implement input/output clock mode for true and simple dual-port memory. On each of the two ports, A and B, one clock controls all registers for inputs into the memory block: data input, `wren`, and address. The other clock controls the block's data output registers. Each memory block port also supports independent clock enables and asynchronous clear signals for input and output registers. Figures 2–10 and 2–11 show the memory block in input/output clock mode for true and simple dual-port modes, respectively.

Figure 2–10. Input/Output Clock Mode in True Dual-Port Mode *Note (1)*

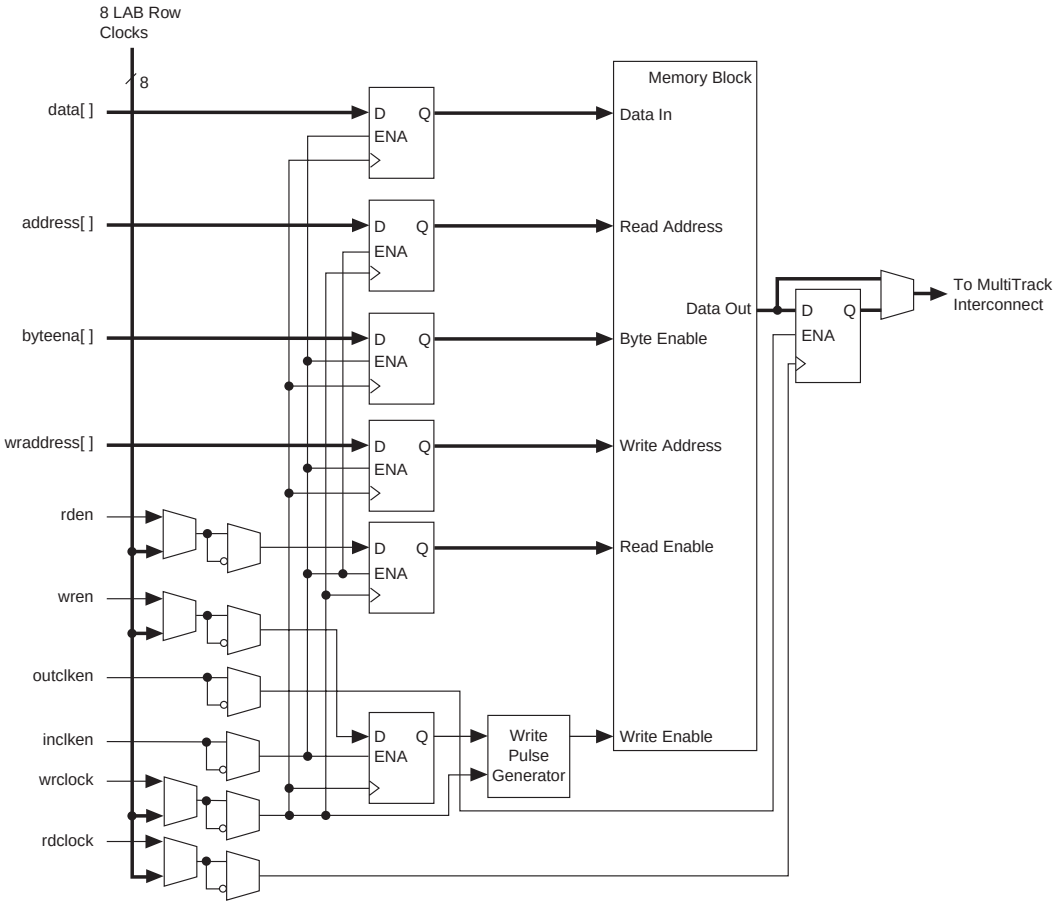


Note to Figure 2–10:

(1) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

All registers shown have asynchronous clear ports, except when using the M-RAM. M-RAM blocks have asynchronous clear ports on their output registers only.

Figure 2–11. Input/Output Clock Mode in Simple Dual-Port Mode *Notes (1), (2), (3), (4)*



Notes to Figure 2–11:

(1) The rden signal is not available in the M-RAM block. A M-RAM block in simple dual-port mode is always reading out the data stored at the current read address location.

(2) For more information on the MultiTrack™ interconnect, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

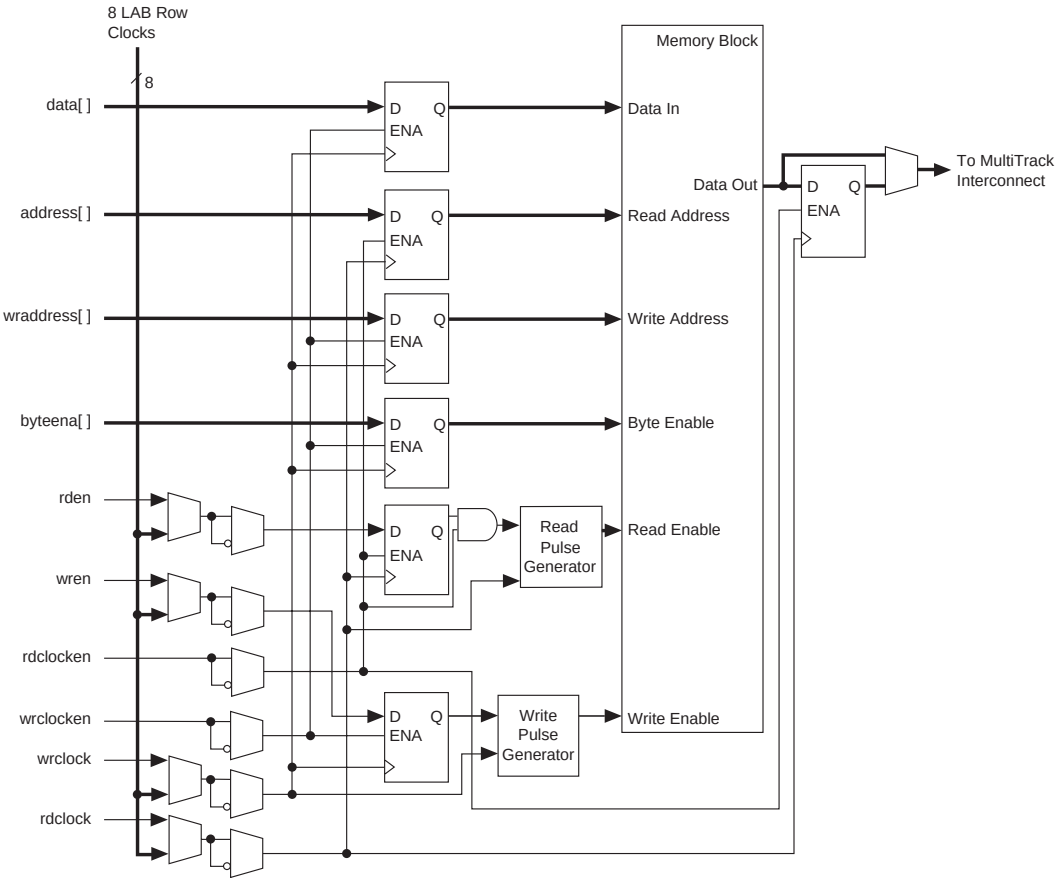
(3) All registers shown have asynchronous clear ports, except when using the M-RAM. M-RAM blocks have asynchronous clear ports on their output registers only.

(4) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Read/Write Clock Mode

The TriMatrix memory blocks can implement read / write clock mode for simple dual-port memory. This mode can use up to two clocks. The write clock controls the block's data inputs, `wraddress`, and `wren`. The read clock controls the data output, `rdaddress`, and `rden`. The memory blocks support independent clock enables for each clock and asynchronous clear signals for the read- and write-side registers. Figure 2–12 shows a memory block in read / write clock mode.

Figure 2–12. Read/Write Clock Mode in Simple Dual-Port Mode *Notes (1), (2), (3)*



Notes to Figure 2–12:

(1) For more information on the MultiTrack interconnect, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

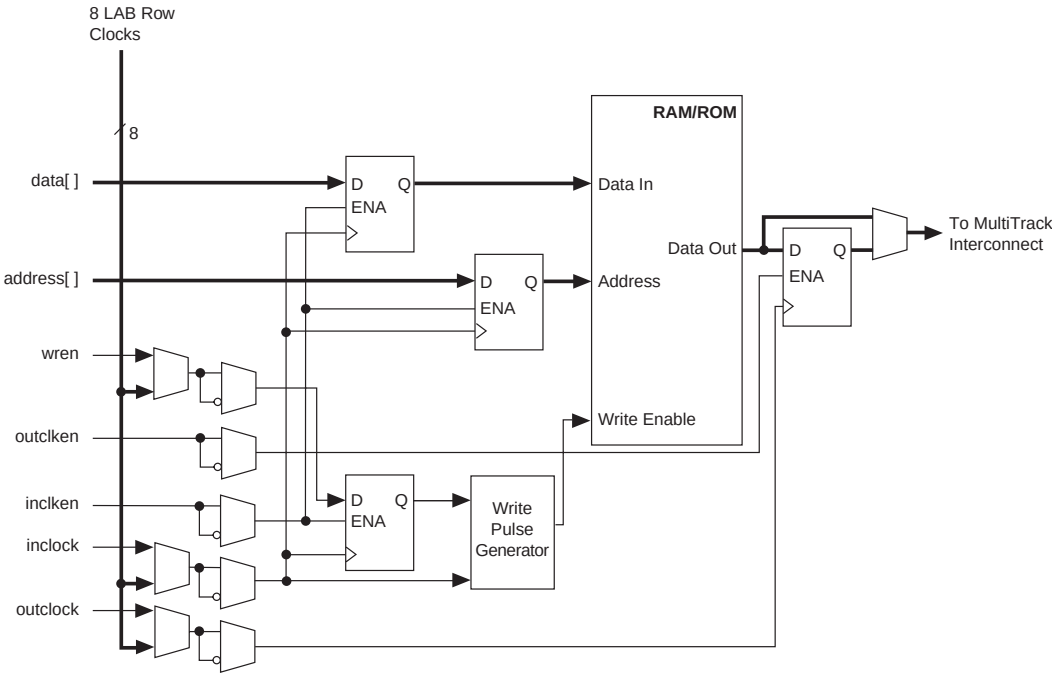
(2) All registers shown have asynchronous clear ports, except when using the M-RAM. M-RAM blocks have asynchronous clear ports on their output registers only.

(3) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Single-Port Mode

The TriMatrix memory blocks can implement single-port clock mode for single-port memory mode. Single-port mode is used when simultaneous reads and writes are not required. See Figure 2–13. A single block in a memory block can support up to two single-port mode RAM blocks in M4K blocks.

Figure 2–13. Single-Port Mode *Notes (1), (2), (3)*



Notes to Figure 2–13:

(1) For more information on the MultiTrack interconnect, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

(2) All registers shown have asynchronous clear ports, except when using the M-RAM. M-RAM blocks have asynchronous clear ports on their output registers only.

(3) Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Designing With TriMatrix Memory

When instantiating TriMatrix memory you must understand the various features that set it apart from other memory architectures. The following sections describe some of the important attributes and functionality of TriMatrix memory.

f For information on the difference between APEX-style memory and TriMatrix memory, see the *Transitioning APEX Designs to Stratix Devices* chapter.

Selecting TriMatrix Memory Blocks

The Quartus II software automatically partitions user-defined memory into embedded memory blocks using the most efficient size combinations. The memory can also be manually assigned to a specific block size or a mixture of block sizes. Table 2–1 on page 2–2 is a guide for selecting a TriMatrix memory block size based on supported features.

1 Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

f For more information on selecting which memory block to use, see *AN 207: TriMatrix Memory Selection Using the Quartus II Software*.

1 Violating the setup or hold time on the address registers could corrupt the memory contents. This applies to both read and write operations.

Pipeline & Flow-Through Modes

TriMatrix memory architecture implements synchronous (pipelined) RAM by registering both the input and output signals to the RAM block. All TriMatrix memory inputs are registered providing synchronous write cycles. In synchronous operation, RAM generates its own self-timed strobe write enable (`wren`) signal derived from the global or regional clock. In contrast, a circuit using asynchronous RAM must generate the RAM `wren` signal while ensuring its data and address signals meet setup and hold time specifications relative to the `wren` signal. The output registers can be bypassed.

In an asynchronous memory neither the input nor the output is registered. While Stratix and Stratix GX devices do not support asynchronous memory, they do support a flow-through read where the output data is available during the clock cycle when the read address is driven into it. Flow-through reading is possible in the simple and true dual-port modes of the M512 and M4K blocks by clocking the read enable and read address registers on the negative clock edge and bypassing the output registers.

f For more information, see *AN 210: Converting Memory from Asynchronous to Synchronous for Stratix & Stratix GX Devices*.

Power-up Conditions & Memory Initialization

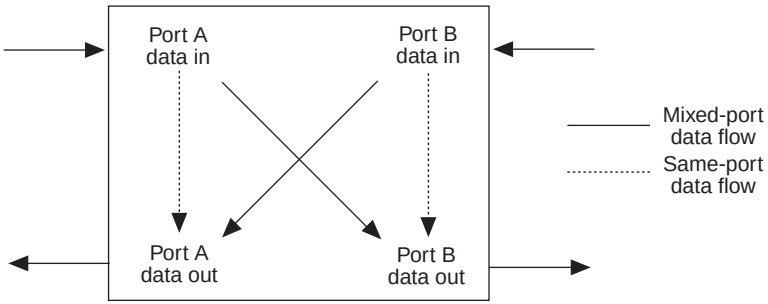
Upon power-up, TriMatrix memory is in an idle state. The M512 and M4K block outputs always power-up to zero, regardless of whether the output registers are used or bypassed. Even if a memory initialization file is used to pre-load the contents of the RAM block, the outputs still power-up cleared. For example, if address `0` is pre-initialized to `FF`, the M512 and M4K blocks power-up with the output at `00`.

M-RAM blocks do not support memory initialization files; therefore, they cannot be pre-loaded with data upon power-up. M-RAM blocks combinatorial outputs and memory controls always power-up to an unknown state. If M-RAM block outputs are registered, the registers power-up cleared. The undefined output appears one clock cycle later. The output remains undefined until a read operation is performed on an address that has been written to.

Read-During-Write Operation at the Same Address

The following two sections describe the functionality of the various RAM configurations when reading from an address during a write operation at that same address. There are two types of read-during-write operations: same-port and mixed-port. Figure 2–14 illustrates the difference in data flow between same-port and mixed-port read-during-write.

Figure 2–14. Read-During-Write Data Flow

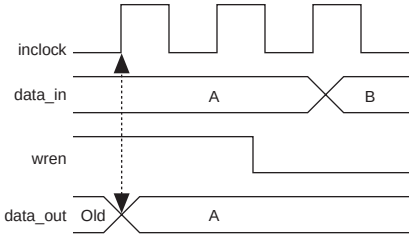


Same-Port Read-During-Write Mode

For read-during-write operation of a single-port RAM or the same port of a true dual-port RAM, the new data is available on the rising edge of the same clock cycle it was written on. This behavior is valid on all memory-block sizes. See Figure 2–15 for a sample functional waveform.

When using byte enables in true dual-port RAM mode, the outputs for the masked bytes on the same port are unknown. (See Figure 2–1 on page 2–6.) The non-masked bytes are read out as shown in Figure 2–15.

Figure 2–15. Same-Port Read-During-Write Functionality *Note (1)*



Note to Figure 2–15:
(1) Outputs are not registered.

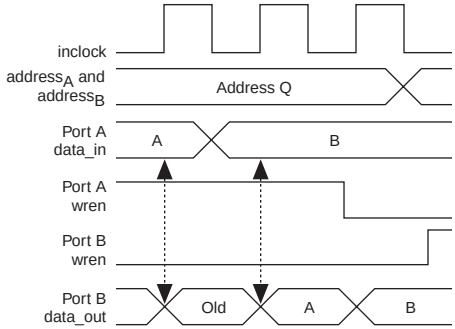
Mixed-Port Read-During-Write Mode

This mode is used when a RAM in simple or true dual-port mode has one port reading and the other port writing to the same address location with the same clock.

The `READ_DURING_WRITE_MODE_MIXED_PORTS` parameter for M512 and M4K memory blocks determines whether to output the old data at the address or a "don't care" value. Setting this parameter to `OLD_DATA` outputs the old data at that address. Setting this parameter to `DONT_CARE` outputs a "don't care" or unknown value. See Figures 2–16 and 2–17 for sample functional waveforms showing this operation. These figures assume that the outputs are not registered.

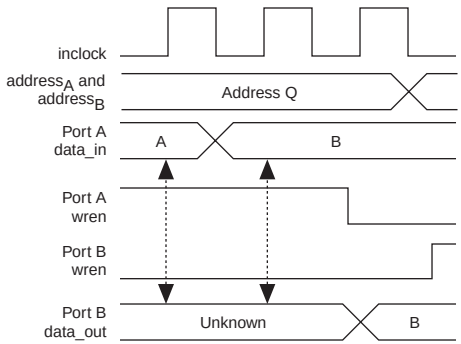
The `DONT_CARE` setting allows memory implementation in any TriMatrix memory block. The `OLD_DATA` setting restricts memory implementation to only M512 or M4K memory blocks. Selecting `DONT_CARE` gives the compiler more flexibility when placing memory functions into TriMatrix memory.

Figure 2–16. Mixed-Port Read-During-Write: OLD_DATA



For mixed-port read-during-write operation of the same address location of a M-RAM block, the RAM outputs are unknown, as shown in Figure 2–17.

Figure 2–17. Mixed-Port Read-During-Write: DONT_CARE



Mixed-port read-during-write is not supported when two different clocks are used in a dual-port RAM. The output value will be unknown during a mixed-port read-during-write operation.

Conclusion

TriMatrix memory, an enhanced RAM architecture with extremely high memory bandwidth in Stratix and Stratix GX devices, gives advanced control of memory applications with features such as byte enables, parity bit storage, and shift-register mode, as well as mixed-port width support and true dual-port mode.

3. External Memory Interfaces in Stratix & Stratix GX Devices

S52008-3.3

Introduction

Stratix® and Stratix GX devices support a broad range of external memory interfaces such as double data rate (DDR) SDRAM, RLDRAM II, quad data rate (QDR) SRAM, QDRII SRAM, zero bus turnaround (ZBT) SRAM, and single data rate (SDR) SDRAM. The dedicated phase-shift circuitry allows the Stratix and Stratix GX devices to interface at twice the system clock speed with an external memory (up to 200 MHz / 400 Mbps).

Typical I / O architectures transmit a single data word on each positive clock edge and are limited to the associated clock speed using this protocol. To achieve a 400-megabits per second (Mbps) transfer rate, a SDR system requires a 400-MHz clock. Many new applications have introduced a DDR I / O architecture as an alternative to SDR architectures. While SDR architectures capture data on one edge of a clock, the DDR architectures captures data on both the rising and falling edges of the clock, doubling the throughput for a given clock frequency and accelerating performance. For example, a 200-MHz clock can capture a 400-Mbps data stream, enhancing system performance and simplifying board design.

Most current memory architectures use a DDR I / O interface. These DDR memory standards cover a broad range of applications for embedded processor systems, image processing, storage, communications, and networking. This chapter describes the hardware features in Stratix and Stratix GX devices that facilitate the high-speed memory interfacing for each memory standard. It then briefly explains how each memory standard uses the features of the Stratix and Stratix GX devices.

f You can use this document with *AN 329: ZBT SRAM Controller Reference Design for Stratix & Stratix GX Devices*, *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*, and *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

External Memory Standards

The following sections provide an overview on using the Stratix and Stratix GX device external memory interfacing features.

DDR SDRAM

DDR SDRAM is a memory architecture that transmits and receives data at twice the clock speed of traditional SDR architectures. These devices transfer data on both the rising and falling edge of the clock signal.

Interface Pins

DDR devices use interface pins including data, data strobe, clock, command, and address pins. Data is sent and captured at twice the clock rate by transferring data on both the positive and negative edge of a clock. The commands and addresses only use one active edge of a clock.

Connect the memory device's DQ and DQS pins to the DQ and DQS pins, respectively, as listed in the Stratix and Stratix GX devices pin table. DDR SDRAM also uses active-high data mask pins for writes. You can connect DM pins to any of the I/O pins in the same bank as the DQ pins of the FPGA. There is one DM pin per DQS/DQ group.

DDR SDRAM ×16 devices use two DQS pins, and each DQS pin is associated with eight DQ pins. However, this is not the same as the ×16 mode in Stratix and Stratix GX devices. To support a ×16 DDR SDRAM, you need to configure the Stratix and Stratix GX FPGAs to use two sets of DQ pins in ×8 mode. Similarly if your ×32 memory device uses four DQS pins where each DQS pin is associated with eight DQ pins, you need to configure the Stratix and Stratix GX FPGA to use four sets of pins in ×8 mode.

You can also use any I/O pins in banks 1, 2, 5, or 6 to interface with DDR SDRAM devices. These banks do not have dedicated circuitry, though.

You can also use any of the user I/O pins for commands and addresses to the DDR SDRAM.

f For more information, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*.

If the DDR SDRAM device supports ECC, the design uses a DQS/DQ group for ECC pins. You can use any of the user I/O pins for commands and addresses.

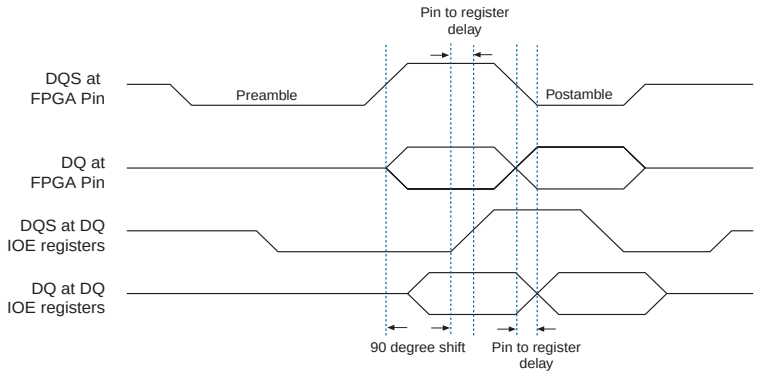
Because of the symmetrical setup and hold time for the command and address pins at the memory, you might need to generate these signals from the system clock's negative edge.

The clocks to the SDRAM device are called CK and CK#. Use any of the user I/O pins via the DDR registers to generate the CK and CK# signals to meet the DDR SDRAM t_{DQSS} requirement. The memory device's t_{DQSS} requires that the DQS signal's positive edge write operations must be within 25% of the positive edge of the DDR SDRAM clock input. Using user I/O pins for CK and CK# ensures that any PVT variations seen by the DQS signal are tracked by these pins, too.

Read & Write Operations

When reading from the DDR SDRAM, the DQS signal coming into the Stratix and Stratix GX device is edge-aligned with the DQ pins. The dedicated circuitry center-aligns the DQS signal with respect to the DQ signals and the shifted DQS bus drives the clock input of the DDR input registers. The DDR input registers bring the data from the DQ signals to the device. The system clock clocks the DQS output enable and output paths. The -90° shifted clock clocks the DQ output enable and output paths. Figure 3–1 shows an example of the DQ and DQS relationship during a burst-of-two read. It shows where the DQS signal is center-aligned in the IOE.

Figure 3–1. Example of Where a DQS Signal is Center-Aligned in the IOE



When writing to the DDR SDRAM, the DQS signal must be center-aligned with the DQ pins. Two PLL outputs are needed to generate the DQS signal and to clock the DQ pins. The DQS are clocked by the 0° phase-shift PLL output, while the DQ pins are clocked by the -90° phase-shifted PLL output. Figure 3–2 shows the DQS and DQ relationship during a DDR SDRAM burst-of-two write.

Figure 3–2. DQ & DQS Relationship During a Burst-of-Two Write

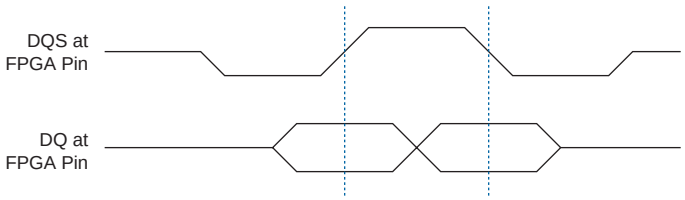
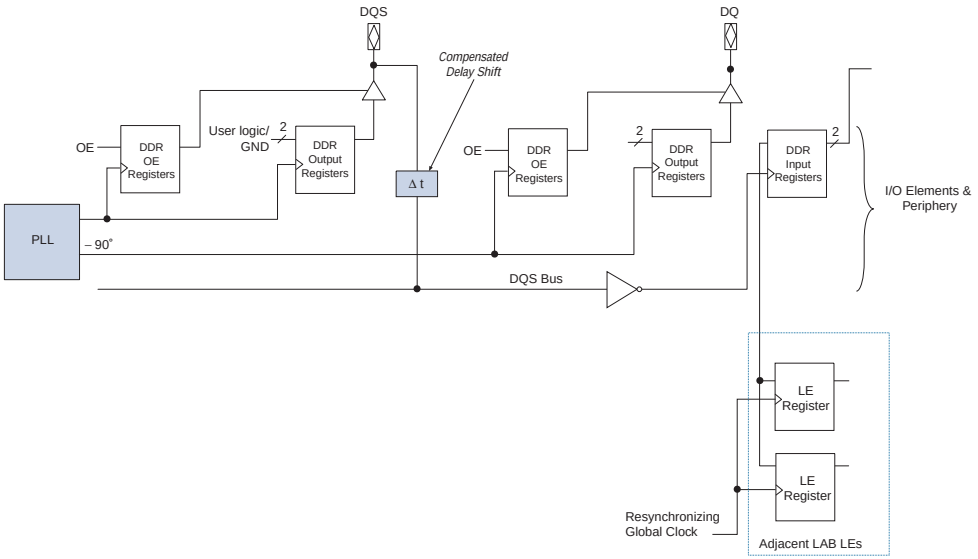


Figure 3–3 shows DDR SDRAM interfacing from the I/O through the dedicated circuitry to the logic array. When the DQS pin acts as an input strobe, the dedicated circuitry shifts the incoming DQS pin by either 72° or 90° and clocks the DDR input registers. Because of the DDR input registers architecture in Stratix and Stratix GX devices, the shifted DQS signal must be inverted. The DDR registers outputs are sent to two LE registers to be synchronized with the system clock.

f Refer to the *DC & Switching Characteristics* chapter in volume 1 of the *Stratix Device Handbook* for frequency limits regarding the 72 and 90° phase shift for DQS.

Figure 3–3. DDR SDRAM Interfacing



f For more information on DDR SDRAM specifications, see JEDEC standard publications JESD79C from **www.jedec.org**, or see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*.

RLDRAM II

RLDRAM II provides fast random access as well as high bandwidth and high density, making this memory technology ideal for high-speed network and communication data storage applications. The fast random access speeds in RLDRAM II devices make them a viable alternative to SRAM devices at a lower cost. Additionally, RLDRAM II devices have minimal latency to support designs that require fast response times.

Interface Pins

RLDRAM II devices use interface pins such as data, clock, command, and address pins. There are two types of RLDRAM II memory: common I/O (CIO) and separate I/O (SIO). The data pins in RLDRAM II CIO device are bidirectional while the data pins in a RLDRAM II SIO device are uni-directional. Instead of bidirectional data strobes, RLDRAM II uses differential free-running read and write clocks to accompany the data. As in DDR SDRAM, data is sent and captured at twice the clock rate by transferring data on both the positive and negative edge of a clock. The commands and addresses still only use one active edge of a clock.

If the data pins are bidirectional, connect them to the Stratix and Stratix GX device DQ pins. If the data pins are uni-directional, connect the RLDRAM II device Q ports to the Stratix and Stratix GX device DQ pins and connect the D ports to any user I/O pins in I/O banks 3, 4, 7, and 8. RLDRAM II also uses active-high data mask pins for writes. You can connect DM pins to any of the I/O pins in the same bank as the DQ pins of the FPGA. When interfacing with SIO devices, connect the DM pins to any of the I/O pins in the same bank as the D pins. There is one DM pin per DQS/DQ group.

Connect the read clock pins (QK) to Stratix and Stratix GX device DQS pins. You must configure the DQS signals as bidirectional pins. However, since QK pins are output-only pins from the memory, RLDRAM memory interfacing in Stratix and Stratix GX devices requires that you ground the DQS and DQSn pin output enables. The Stratix and Stratix GX devices use the shifted QK signal from the DQS logic block to capture data. You can leave the QK# signal of the RLDRAM II device unconnected.

RLDRAM II devices have both input clocks (CK and CK#) and write clocks (DK and DK#). Use the external clock buffer to generate CK, CK#, DK, and DK# to meet the CK, CK#, DK, and DK# skew requirements from the RLDRAM II device. If you are interfacing with multiple RLDRAM II devices, perform IBIS simulations to analyze the loading effects on the clock pair.

You can use any of the user I/O pins for commands and addresses. RLDRAM II also offers QVLD pins to indicate the read data availability. Connect the QVLD pins to the Stratix and Stratix GX device DQVLD pins, listed in the pin table.

Read & Write Operations

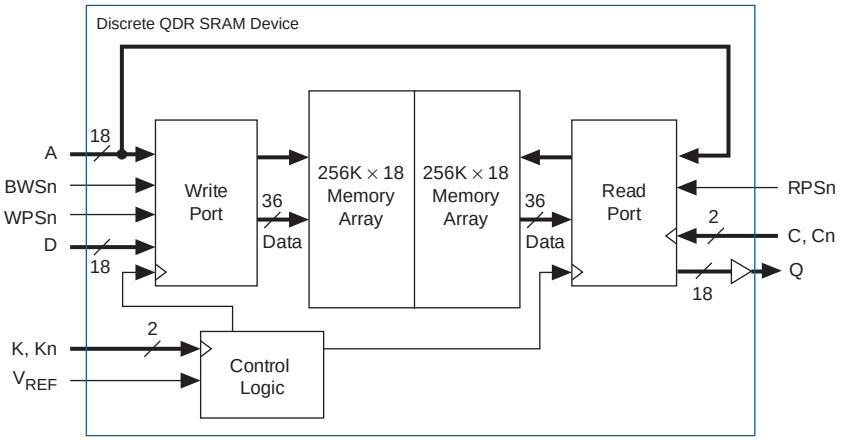
When reading from the RLDRAM II device, data is sent edge-aligned with the read clock QK or QK# signal. When writing to the RLDRAM II device, data must be center-aligned with the write clock (DK or DK# signal). The Stratix and Stratix GX device RLDRAM II interface uses the

same scheme as in DDR SDRAM interfaces whereby the dedicated circuitry is used during reads to center-align the data and the read clock inside the FPGA and the PLL center-aligns the data and write clock outputs. The data and clock relationship for reads and writes in RLDRAM II is similar to those in DDR SDRAM as already depicted in Figure 3–1 on page 3–3 and Figure 3–3 on page 3–4.

QDR & QDRII SRAM

QDR SRAM provides independent read and write ports that eliminate the need for bus turnaround. The memory uses two sets of clocks: K and Kn for write access, and optional C and Cn for read accesses, where Kn and Cn are the inverse of the K and C clocks, respectively. You can use differential HSTL I/O pins to drive the QDR SRAM clock into the Stratix and Stratix GX devices. The separate write data and read data ports permit a transfer rate up to four words on every cycle through the DDR circuitry. Stratix and Stratix GX devices support both burst-of-two and burst-of-four QDR SRAM architectures, with clock cycles up to 167 MHz using the 1.5-V HSTL Class I or Class II I/O standard. Figure 3–4 shows the block diagram for QDR SRAM burst-of-two architecture.

Figure 3–4. QDR SRAM Block Diagram for Burst-of-Two Architecture



QDRII SRAM is a second generation of QDR SRAM devices. It can transfer four words per clock cycle, fulfilling the requirements facing next-generation communications system designers. QDRII SRAM devices provide concurrent reads and writes, zero latency, and increased data throughput. Stratix and Stratix GX devices support QDRII SRAM at speeds up to 200 MHz since the timing requirements for QDRII SRAM are not as strict as QDR SRAM.

Interface Pins

QDR and QDRII SRAM uses two separate, uni-directional data ports for read and write operations, enabling quad data-rate data transfer. Both QDR and QDRII SRAM use shared address lines for reads and writes. Stratix and Stratix GX devices utilize dedicated DDR I/O circuitry for the input and output data bus and the K and Kn output clock signals.

Both QDR and QDRII SRAM burst-of-two devices sample the read address on the rising edge of the K clock and sample the write address on the rising edge of the Kn clock while QDR and QDRII SRAM burst-of-four devices sample both read and write addresses on the K clock's rising edge. You can use any of the Stratix and Stratix GX device user I/O pins in I/O banks 3, 4, 7, and 8 for the D write data ports, commands, and addresses.

QDR SRAM uses the following clock signals: input clocks K and Kn and output clocks C and Cn. In addition to the aforementioned two pairs of clocks, QDRII SRAM also uses echo clocks CQ and CQn. Clocks Cn, Kn, and CQn are logical complements of clocks C, K, and CQ respectively. Clocks C, Cn, K, and Kn are inputs to the QDRII SRAM while clocks CQ and CQn are outputs from the QDRII SRAM. Stratix and Stratix GX devices use single-clock mode for single-device QDR and QDRII SRAM interfacing where the K and Kn are used for both read and write operations, and the C and Cn clocks are unused. Use both C or Cn and K or Kn clocks when interfacing with a bank of multiple QDRII SRAM devices with a single controller.

You can generate C, Cn, K, and Kn clocks using any of the I/O registers in I/O banks 3, 4, 7, or 8 via the DDR registers. Due to strict skew requirements between K and Kn signals, use adjacent pins to generate the clock pair. Surround the pair with buffer pins tied to V_{CC} and ground for better noise immunity from other signals.

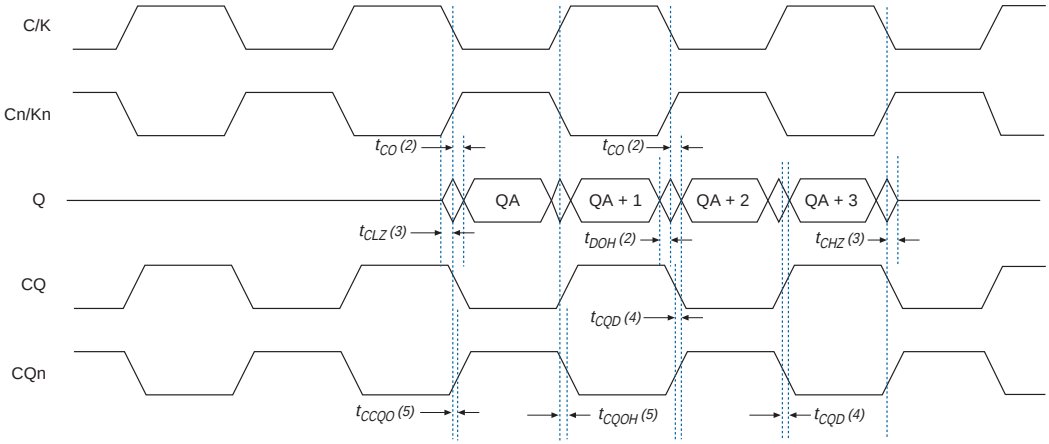
In general, all output signals to the QDR and QDRII SRAM should use the top and bottom banks (I/O banks 3, 4, 7, or 8). You can place the input signals from the QDR and QDRII SRAM in any I/O banks.

Read & Write Operations

Figure 3–5 shows the data and clock relationships in QDRII SRAM devices at the memory pins during reads. QDR and QDRII SRAM devices send data within a t_{CO} time after each rising edge of the input clock C or Cn in multi-clock mode, or the input clock K or Kn in single clock mode. Data is valid until t_{DOH} time, after each rising edge of the C or Cn in multi-

clock mode, or K or Kn in single clock mode. The edge-aligned CQ and CQn clocks accompany the read data for data capture in Stratix and Stratix GX devices.

Figure 3–5. Data & Clock Relationship During a QDRII SRAM Read *Note (1)*



Notes to Figure 3–5:

(1) The timing parameter nomenclature is based on the Cypress QDRII SRAM data sheet for CY7C1313V18.

(2) CO is the data clock-to-out time and t_{DOH} is the data output hold time between burst.

(3) t_{CLZ} and t_{CHZ} are bus turn-on and turn-off times respectively.

(4) t_{CQD} is the skew between CQn and data edges.

(5) t_{CQQO} and t_{CQOH} are skew between the C or Cn (or K or Kn in single-clock mode) and the CQ or CQn clocks.

When writing to QDRII SRAM devices, data is generated by the write clock, while the K clock is 90° shifted from the write clock, creating a center-aligned arrangement.

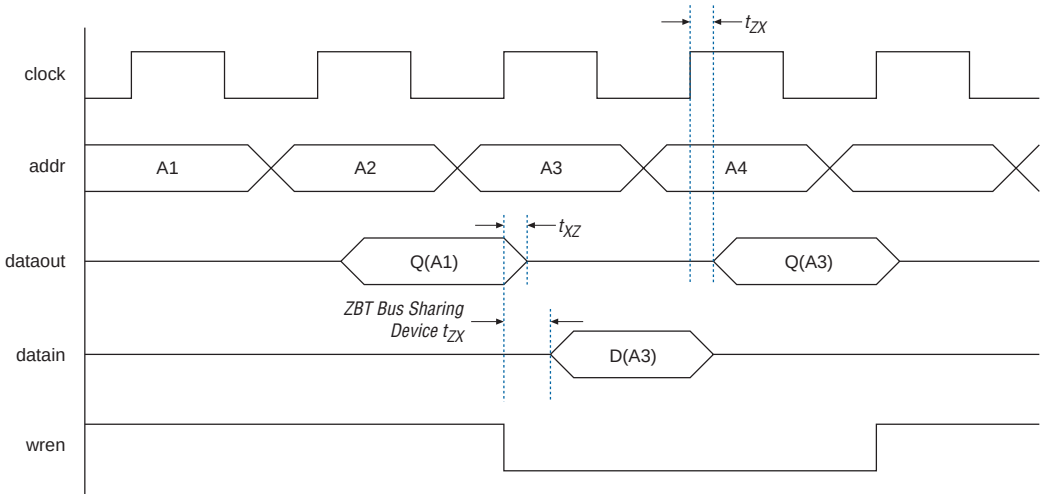
f Go to **www.qdrsram.com** for the QDR SRAM and QDRII SRAM specifications. For more information on QDR and QDRII SRAM interfaces in Stratix and Stratix GX devices, see *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

ZBT SRAM

ZBT SRAM eliminate dead bus cycles when turning a bidirectional bus around between reads and writes or between writes and reads. ZBT allows for 100% bus utilization because ZBT SRAM can be read or written on every clock cycle. Bus contention can occur when shifting from a write cycle to a read cycle or vice versa with no idle cycles in between. ZBT SRAM allows small amounts of bus contention. To avoid bus contention, the output clock-to-low-impedance time (t_{ZX}) must be greater

than the clock-to-high-impedance time (t_{XZ}). Stratix and Stratix GX device I/O pins can interface with ZBT SRAM devices at up to 200 MHz and can meet ZBT t_{CO} and t_{SU} timing requirements by controlling phase delay in clocks to the OE or output and input registers using an enhanced PLL. Figure 3–6 shows a flow-through ZBT SRAM operation where A1 and A3 are read addresses and A2 and A4 are write addresses. For pipelined ZBT SRAM operation, data is delayed by another clock cycle. Stratix and Stratix GX devices support up to 200-MHz ZBT SRAM operation using the 2.5-V or 3.3-V LVTTL I/O standard.

Figure 3–6. t_{ZX} & t_{XZ} Timing Diagram



Interface Pins

ZBT SRAM uses one system clock input for all clocking purposes. Only the rising edge of this clock is used, since ZBT SRAM uses a single data rate scheme. The data bus, DQ, is bidirectional. There are three control signals to the ZBT SRAM: `RW_N`, `BW_N`, and `ADV_LD_N`. You can use any of the Stratix and Stratix GX device user I/O pins to interface to the ZBT SRAM device.

f For more information on ZBT SRAM Interfaces in Stratix devices, see *AN 329: ZBT SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

DDR Memory Support Overview

Table 3–1 shows the external RAM support in Stratix EP1S10 through EP1S40 devices and all Stratix GX devices. Table 3–2 shows the external RAM support in Stratix EP1S60 and EP1S80 devices.

Table 3–1. External RAM Support in Stratix EP1S10 through EP1S40 & All Stratix GX Devices

DDR Memory Type	I/O Standard	Maximum Clock Rate (MHz)						
		-5 Speed Grade	-6 Speed Grade		-7 Speed Grade		-8 Speed Grade	
		Flip-Chip	Flip-Chip	Wire-Bond	Flip-Chip	Wire-Bond	Flip-Chip	Wire-Bond
DDR SDRAM *(1)*, *(2)*	SSTL-2	200	167	133	133	100	100	100
DDR SDRAM - side banks *(2)*, *(3)*, *(4)*	SSTL-2	150	133	110	133	100	100	100
RLDRAM II *(4)*	1.8-V HSTL	200	*(5)*	*(5)*	*(5)*	*(5)*	*(5)*	*(5)*
QDR SRAM *(6)*	1.5-V HSTL	167	167	133	133	100	100	100
QDRII SRAM *(6)*	1.5-V HSTL	200	167	133	133	100	100	100
ZBT SRAM *(7)*	LVTTL	200	200	200	167	167	133	133

Notes to Table 3–1:

(1) These maximum clock rates apply if the Stratix device uses DQS phase-shift circuitry to interface with DDR SDRAM. DQS phase-shift circuitry is only available on the top and bottom I/O banks (I/O banks 3, 4, 7, and 8).

(2) For more information on DDR SDRAM, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices.*

(3) DDR SDRAM is supported on the Stratix device side I/O banks (I/O banks 1, 2, 5, and 6) without dedicated DQS phase-shift circuitry. The read DQS signal is ignored in this mode.

(4) These performance specifications are preliminary.

(5) This device does not support RLDRAM II.

(6) For more information on QDR or QDRII SRAM, see *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

(7) For more information on ZBT SRAM, see *AN 329: ZBT SRAM Controller Reference Design for Stratix and Stratix GX Devices.*

Table 3–2. External RAM Support in Stratix EP1S60 & EP1S80 (Part 1 of 2)

DDR Memory Type	I/O Standard	Maximum Clock Rate (MHz)		
		-5 Speed Grade	-6 Speed Grade	-7 Speed Grade
DDR SDRAM *(1)*, *(2)*	SSTL-2	167	167	133
DDR SDRAM - side banks *(2)*, *(3)*	SSTL-2	150	133	133
QDR SRAM *(4)*	1.5-V HSTL	133	133	133
QDRII SRAM *(4)*	1.5-V HSTL	167	167	133

Table 3–2. External RAM Support in Stratix EP1S60 & EP1S80 (Part 2 of 2)

DDR Memory Type	I/O Standard	Maximum Clock Rate (MHz)		
		-5 Speed Grade	-6 Speed Grade	-7 Speed Grade
ZBT SRAM *(5)*	LVTTL	200	200	167

Notes to Table 3–2:

(1) These maximum clock rates apply if the Stratix device uses DQS phase-shift circuitry to interface with DDR SDRAM. DQS phase-shift circuitry is only available on the top and bottom I/O banks (I/O banks 3, 4, 7, and 8).

(2) For more information on DDR SDRAM, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices.*

(3) DDR SDRAM is supported on the side banks (I/O banks 1, 2, 5, and 6) with no dedicated DQS phase-shift circuitry. The read DQS signal is ignored in this mode.

(4) For more information on QDR or QDRII SRAM, see *AN 349: QDR SRAM Controller Reference Design for Stratix & Stratix GX Devices*.

(5) For more information on ZBT SRAM, see *AN 329: ZBT SRAM Controller Reference Design for Stratix and Stratix GX Devices.*

Stratix and Stratix GX devices support the data strobe or read clock signal (DQS) used in DDR SDRAM, and RLDRAM II devices. DQS signals are associated with a group of data (DQ) pins.

Stratix and Stratix GX devices contain dedicated circuitry to shift the incoming DQS signals by 0°, 72°, and 90°. The DQS phase-shift circuitry uses a frequency reference to dynamically generate control signals for the delay chains in each of the DQS pins, allowing it to compensate for process, voltage, and temperature (PVT) variations. The dedicated circuitry also creates consistent margins that meet your data sampling window requirements.

f Refer to the *DC & Switching Characteristics* chapter in volume 1 of the *Stratix Device Handbook* for frequency limits regarding the 72 and 90° phase shift for DQS.

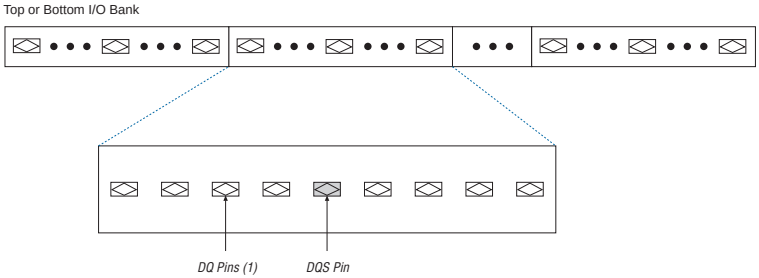
In addition to the DQS dedicated phase-shift circuitry, every I/O element (IOE) in Stratix and Stratix GX devices contains six registers and one latch to achieve DDR operation. There is also a programmable delay chain in the IOE that can help reduce contention when interfacing with ZBT SRAM devices.

DDR Memory Interface Pins

Stratix and Stratix GX devices use data (DQ), data strobe (DQS), and clock pins to interface with DDR SDRAM and RLDRAM II devices. This section explains the pins used in the DDR SDRAM and RLDRAM II interfaces. For QDR, QDRII, and ZBT SRAM interfaces, see the "External Memory Standards" section.

Figure 3–7 shows the DQ and DQS pins in ×8 mode.

Figure 3–7. Stratix & Stratix GX Device DQ & DQS Groups in × 8 Mode



Note to Figure 3–7:

(1) There are at least eight DQ pins per group.

Data & Data Strobe Pins

Stratix and Stratix GX data pins for the DDR memory interfaces are called DQ pins. The Stratix and Stratix GX device I/O banks at the top (I/O banks 3 and 4) and the bottom (I/O banks 7 and 8) of the device support DDR SDRAM and RLDRAM II up to 200 MHz. These pins support DQS signals with DQ bus modes of ×8, ×16, or ×32. Stratix and Stratix GX devices can support either bidirectional data strobes or uni-directional read clocks. Depending on the external memory interface, either the memory device's read data strobes or read clocks feed the DQS pins.

For ×8 mode, there are up to 20 groups of programmable DQS and DQ pins—10 groups in I/O banks 3 and 4 and 10 groups in I/O banks 7 and 8 (see Table 3–3). Each group consists of one DQS pin and a set of eight DQ pins.

For ×16 mode, there are up to eight groups of programmable DQS and DQ pins—four groups in I/O banks 3 and 4, and four groups in I/O banks 7 and 8. The EP1S20 device supports seven ×16 mode groups. The EP1S10 device does not support ×16 mode. All other devices support the full eight groups. See Table 3–3. Each group consists of one DQS and 16 DQ pins. In ×16 mode, `DQS1T`, `DQS3T`, `DQS6T`, and `DQS8T` pins on the top side of the device, and `DQS1B`, `DQS3B`, `DQS6B`, and `DQS8B` pins on the

bottom side of the device are dedicated DQS pins. The DQS2T, DQS7T, DQS2B, and DQS7B pins are dedicated DQS pins for ×32 mode, and each group consists of one DQS and 32 DQ pins.

Table 3–3. DQS & DQ Bus Mode Support *Note (1)*

Device	Package	Number of ×8 Groups	Number of ×16 Groups	Number of ×32 Groups
EP1S10	672-pin BGA 672-pin FineLine BGA®	12 *(2)*	0	0
	484-pin FineLine BGA 780-pin FineLine BGA	16 *(3)*	0	4
EP1S20	484-pin FineLine BGA	18 *(4)*	7 *(5)*	4
	672-pin BGA 672-pin FineLine BGA	16 *(3)*	7 *(5)*	4
	780-pin FineLine BGA	20	7 *(5)*	4
EP1S25	672-pin BGA 672-pin FineLine BGA	16 *(3)*	8	4
	780-pin FineLine BGA 1,020-pin FineLine BGA	20	8	4
EP1S30	956-pin BGA 780-pin FineLine BGA 1,020-pin FineLine BGA	20	8	4
EP1S40	956-pin BGA 1,020-pin FineLine BGA 1,508-pin FineLine BGA	20	8	4
EP1S60	956-pin BGA 1,020-pin FineLine BGA 1,508-pin FineLine BGA	20	8	4
EP1S80	956-pin BGA 1,508-pin FineLine BGA 1,923-pin FineLine BGA	20	8	4

Notes to Table 3–3:

(1) For V_{REF} guidelines, see the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter of the *Stratix Device Handbook, Volume 2* or the *Stratix GX Handbook, Volume 2*.

(2) These packages have six groups in I/O banks 3 and 4 and six groups in I/O banks 7 and 8.

(3) These packages have eight groups in I/O banks 3 and 4 and eight groups in I/O banks 7 and 8.

(4) This package has nine groups in I/O banks 3 and 4 and nine groups in I/O banks 7 and 8.

(5) These packages have three groups in I/O banks 3 and 4 and four groups in I/O banks 7 and 8.

The DQS pins are marked in the Stratix and Stratix GX device pin table as `DQS[9..0]T` or `DQS[9..0]B`, where `T` stands for top and `B` for bottom. The corresponding DQ pins are marked as `DQ[9..0]T[7..0]`, where `[9..0]` indicates which DQS group the pins belong to. The numbering scheme starts from right to left on the package bottom view. When not used as DQ or DQS pins, these pins are available as user I/O pins.

You can also create a design in a mode other than the ×8, ×16, or ×32 mode. The Quartus® II software uses the next larger mode with the unused DQ pins available as regular use I/O pins. For example, if you create a design for ×9 mode for an RLDRAM II interface (nine DQ pins driven by one DQS pin), the Quartus II software implements a ×16 mode with seven DQ pins unconnected to the DQS bus. These seven unused DQ pins can be used as regular I/O pins.

1 On the top and bottom side of the device, the DQ and DQS pins must be configured as bidirectional DDR pins to enable the DQS phase-shift circuitry. If you only want to use the DQ and/or DQS pins as inputs, you need to set the output enable of the DQ and/or DQS pins to ground. Use the `altdqs` and `altdq` megafunctions to configure the DQS and DQ pins, respectively. However, you should use the Altera® IP Toolbench to create the data path for your memory interfaces.

Stratix and Stratix GX device side I/O banks (I/O banks 1, 2, 5, and 6) support SDR SDRAM, ZBT SRAM, QDR SRAM, QDRII SRAM, and DDR SDRAM interfaces and can use any of the user I/O pins in these banks for the interface. Since these I/O banks do not have any dedicated circuitry for memory interfacing, they can support DDR SDRAM up to 150 MHz in -5 speed grade devices. However, these I/O banks do not support the HSTL-18 Class II I/O standard, which is required to interface with RLDRAM II.

Clock Pins

You can use any of the DDR I/O registers in the top or bottom bank of the device (I/O banks 3, 4, 7, or 8) to generate clocks to the memory device. You can also use any of the DDR I/O registers in the side I/O banks 1, 2, 5, or 6 to generate clocks for DDR SDRAM interfaces on the side I/O banks (not using the DQS circuitry).

Command & Address Pins

You can use any of the user I/O pins in the top or bottom bank of the device (I/O banks 3, 4, 7, or 8) for commands and addresses. For DDR SDRAM, you can also use any of the user I/O pins in the side I/O banks 1, 2, 5, or 6, regardless of whether you use the DQS phase-shift circuitry or not.

Other Pins (Parity, DM, ECC & QVLD Pins)

You can use any of the DQ pins for the parity pins in Stratix and Stratix GX devices. However, this may mean that you are using the next larger DQS/DQ mode. For example, if you need a parity bit for each byte of data, you are actually going to have nine DQ pins per DQS pin. The Quartus II software then implements a ×16 mode, with the seven unused DQ pins available as user I/O pins.

The data mask (DM) pins are only required when writing to DDR SDRAM and RLDRAM II devices. A low signal on the DM pins indicates that the write is valid. If the DM signal is high, the memory masks the DQ signals. You can use any of the I/O pins in the same bank as the DQ pins for the DM signals. Each group of DQS and DQ signals requires a DM pin. The DDR register, clocked by the –90° shifted clock, creates the DM signals, similar to DQ output signals.

Some DDR SDRAM devices support error correction coding (ECC), which is a method of detecting and automatically correcting errors in data transmission. Connect the DDR ECC pins to a Stratix and Stratix GX device DQS/DQ group. In 72-bit DDR SDRAM, there are eight ECC pins in addition to the 64 data pins. The memory controller needs extra logic to encode and decode the ECC data.

QVLD pins are used in RLDRAM II interfacing to indicate the read data availability. There is one QVLD pin per RLDRAM II device. A high on QVLD indicates that the memory is outputting the data requested. Similar to DQ inputs, this signal is edge-aligned with the RLDRAM II read clocks, QK and QK#, and is sent half a clock cycle before data starts coming out of the memory. You can connect QVLD pins to any of the I/O pins in the same bank as the DQ pins for the QVLD signals.

DQS Phase-Shift Circuitry

Two single phase-shifting reference circuits are located on the top and bottom of the Stratix and Stratix GX devices. Each circuit is driven by a system reference clock that is of the same frequency as the DQS signal. Clock pins `CLK[15..12]p` feed the phase-shift circuitry on the top of the device and clock pins `CLK[7..4]p` feed the phase-shift circuitry on the

bottom of the device. The phase-shift circuitry cannot be fed from other sources such as the LE or the PLL internal output clocks. This phase-shift circuitry is used for DDR SDRAM and RLDRAM II interfaces. For best performance, turn off the input reference clock to the DQS phase-shift circuitry when reading from the DDR SDRAM or RLDRAM II. This is to avoid any DLL jitter incorrectly shifting the DQS signal while the FPGA is capturing data.

1 The I/O pins in I/O banks 1, 2, 5, and 6 can interface with the DDR SDRAM at up to 150 MHz. See *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*.

A compensated delay element on each DQS pin allows for either a 90° or a 72° phase shift, which automatically centers input DQS signals with the data valid window of their corresponding DQ data signals. The DQS signals drive a local DQS bus within the top and bottom I/O banks. This DQS bus is an additional resource to the I/O clocks and clocks DQ input registers with the DQS signal.

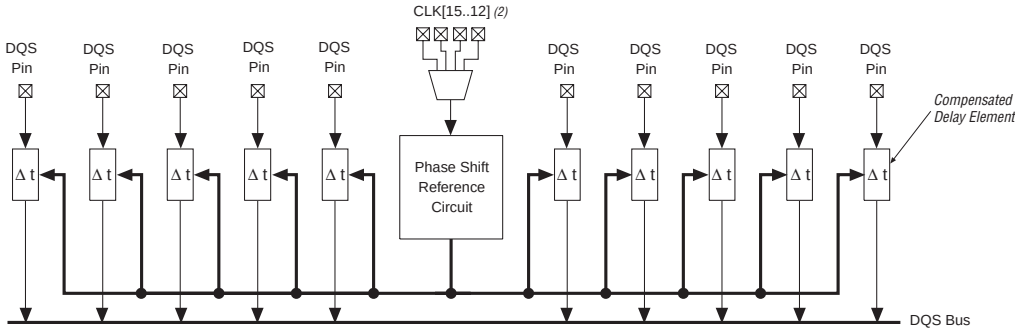
f Refer to the *DC & Switching Characteristics* chapter in volume 1 of the *Stratix Device Handbook* for frequency limits regarding the 72 and 90° phase shift for DQS.

The phase-shifting reference circuit on the top of the device controls the compensated delay elements for all 10 DQS pins located at the top of the device. The phase-shifting reference circuit on the bottom of the device controls the compensated delay elements for all 10 DQS pins located on the bottom of the device. All 10 delay elements (DQS signals) on either the top or bottom of the device shift by the same degree amount. For example, all 10 DQS pins on the top of the device can be shifted by 90° and all 10 DQS pins on the bottom of the device can be shifted by 72°. The reference circuit requires a maximum of 256 system reference clock cycles to set the correct phase on the DQS delay elements.

1 This applies only to the initial phase calculation. Altera recommends that you enable the DLL during the refresh cycle of the DDR SDRAM. Enabling the DLL for the duration of the minimum refresh time is sufficient for recalculating the phase shift.

Figure 3–8 shows the phase-shift reference circuit control of each DQS delay shift on the top of the device. This same circuit is duplicated on the bottom of the device.

Figure 3–8. DQS & DQSn Pins & the DQS Phase-Shift Circuitry *Note (1)*



Notes to Figure 3–8:

(1) There are up to 10 DQS and DQSn pins available on the top or the bottom of the Stratix and Stratix GX devices.

(2) Clock pins `CLK[15..12]p` feed the phase-shift circuitry on the top of the device and clock pins `CLK[7..4]p` feed the phase circuitry on the bottom of the device. The reference clock can also be used in the logic array.

The phase-shift circuitry is only used during read transactions where the DQS pins are acting as input clocks or strobes. The phase-shift circuitry can shift the incoming DQS signal by 0°, 72°, and 90°. The shifted DQS signal is then inverted and used as a clock or a strobe at the DQ IOE input registers.

f Refer to the *DC & Switching Characteristics* chapter in volume 1 of the *Stratix Device Handbook* for frequency limits regarding the 72 and 90° phase shift for DQS.

The DQS phase-shift circuitry is bypassed when 0° shift is chosen. The routing delay between the pins and the IOE registers is matched with high precision for both the DQ and DQS signal when the 72° or 90° phase shift is used. With the 0° phase shift, the skew between DQ and the DQS signals at the IOE register has been minimized. See Table 3–4 for the Quartus II software reported number on the DQ and DQS path to the IOE when the DQS is set to 0° phase shift.

Table 3–4. Quartus II Reported Number on the DQS Path to the IOE *Note (1)*

Speed Grade	DQ2IOE	DQS2IOE	Unit
-5	0.908	1.008	ns
-6	0.956	1.061	ns
-7	1.098	1.281	ns

Table 3–4. Quartus II Reported Number on the DQS Path to the IOE *Note (1)*

Speed Grade	DQ2IOE	DQS2IOE	Unit
-8	1.293	1.635	ns

Note to Table 3–4:

(1) These are reported by Quartus II version 4.0. Check the latest version of the Quartus II software for the most current information.

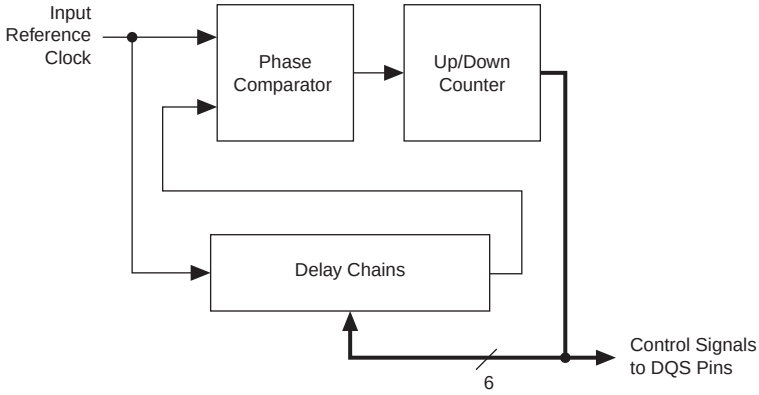
To generate the correct phase shift, you must provide a clock signal of the same frequency as the DQS signal to the DQS phase-shift circuitry. Any of the `CLK[15..12]p` clock pins can feed the phase circuitry on the top of the device (I/O banks 3 and 4) and any of the `CLK[7..4]p` clock pins can feed the phase circuitry on the bottom of the device (I/O banks 7 and 8). Both the top and bottom phase-shift circuits need unique clock pins for the reference clock. You cannot use any internal clock sources to feed the phase-shift circuitry, but you can route internal clock sources off-chip and then back into one of the allowable clock input pins.

DLL

The DQS phase-shift circuitry uses a DLL to dynamically measure the clock period needed by the DQS pin (see Figure 3–9). The DQS phase-shift circuitry then uses the clock period to generate the correct phase shift. The DLL in the Stratix and Stratix GX devices DQS phase-shift circuitry can operate between 100 and 200 MHz. The phase-shift circuitry needs a maximum of 256 clock cycles to calculate the correct phase shift. Data sent during these clock cycles may not be properly captured.

1 You can still use the DQS phase-shift circuitry for DDR SDRAM interfaces that are less than 100 MHz. The DQS signal is shifted by about 2.5 ns. This shifted DQS signal is not in the center of the DQ signals, but it is shifted enough to capture the correct data in this low-frequency application.

Figure 3–9. Simplified Diagram of the DQS Phase-Shift Circuitry



The input reference clock goes into the DLL to a chain of delay elements. The phase comparator compares the signal coming out of the end of the delay element chain to the input reference clock. The phase comparator then issues the upndn signal to the up / down counter. This signal increments or decrements a six-bit delay setting (control signals to DQS pins) that increases or decreases the delay through the delay element chain to bring the input reference clock and the signals coming out of the delay element chain in phase.

The shifted DQS signal then goes to the DQS bus to clock the IOE input registers of the DQ pins. It cannot go into the logic array for other purposes.

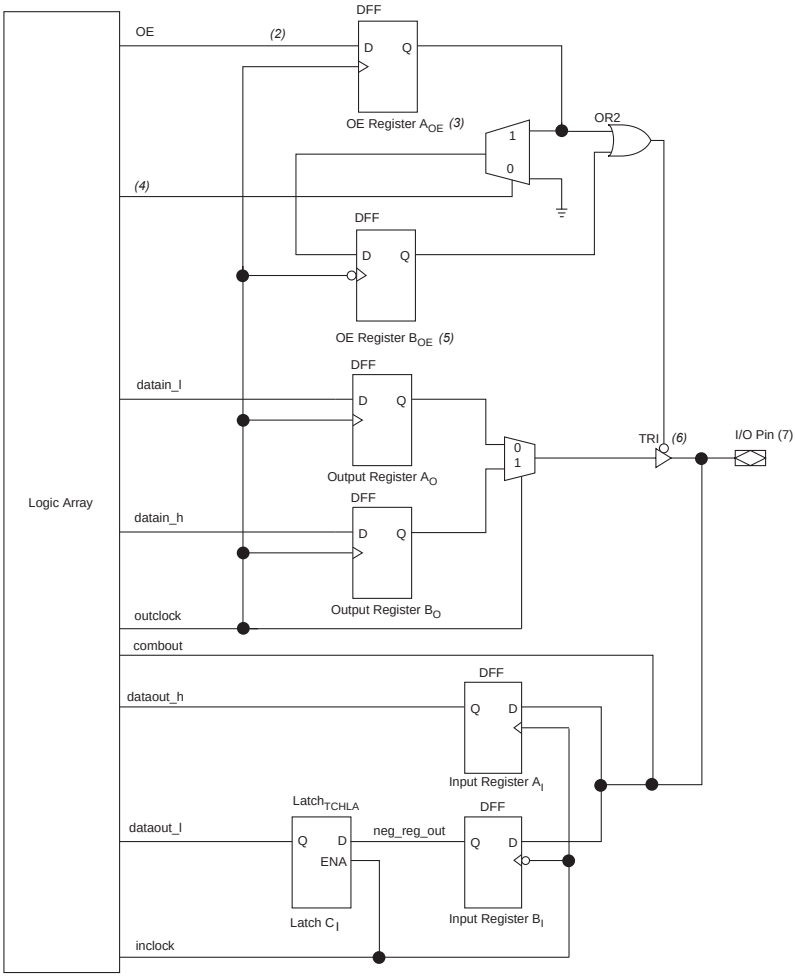
For external memory interfaces that use a bidirectional read strobe like DDR SDRAM, the DQS signal is low before going to or coming from a high-impedance state (see Figure 3–1 on page 3–3). The state where DQS is low just after a high-impedance state is called the preamble and the state where DQS is low just before it returns to high-impedance state is called the postamble. There are preamble and postamble specifications for both read and write operations in DDR SDRAM. To ensure data is not lost when there is noise on the DQS line at the end of a read postamble time, you need to add soft postamble circuitry to disable the clocks at the DQ IOE registers.

f For more information, the DQS Postamble soft logic is described in *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*. The Altera DDR SDRAM controller MegaCore® generates this logic as open-source code.

DDR Registers

Each Stratix and Stratix GX IOE contains six registers and one latch. Two registers and a latch are used for input, two registers are used for output, and two registers are used for output enable control. The second output enable register provides the write preamble for the DQS strobe in the DDR external memory interfaces. This negative-edge output enable register extends the high-impedance state of the pin by a half clock cycle to provide the external memory's DQS preamble time specification. Figure 3–10 shows the six registers and the latch in the Stratix and Stratix GX IOE and Figure 3–11 shows how the second OE register extends the DQS high impedance state by half a clock cycle during a write operation.

Figure 3–10. Bidirectional DDR I/O Path in Stratix & Stratix GX Devices *Note (1)*



Notes to Figure 3–10:

(1) All control signals can be inverted at the IOE. No programmable delay chains are shown in this diagram.

(2) The OE signal is active low, but the Quartus II software implements this as active high and automatically adds an inverter before input to the A_{OE} register during compilation.

(3) The A_{OE} register generates the enable signal for general-purpose DDR I/O applications.

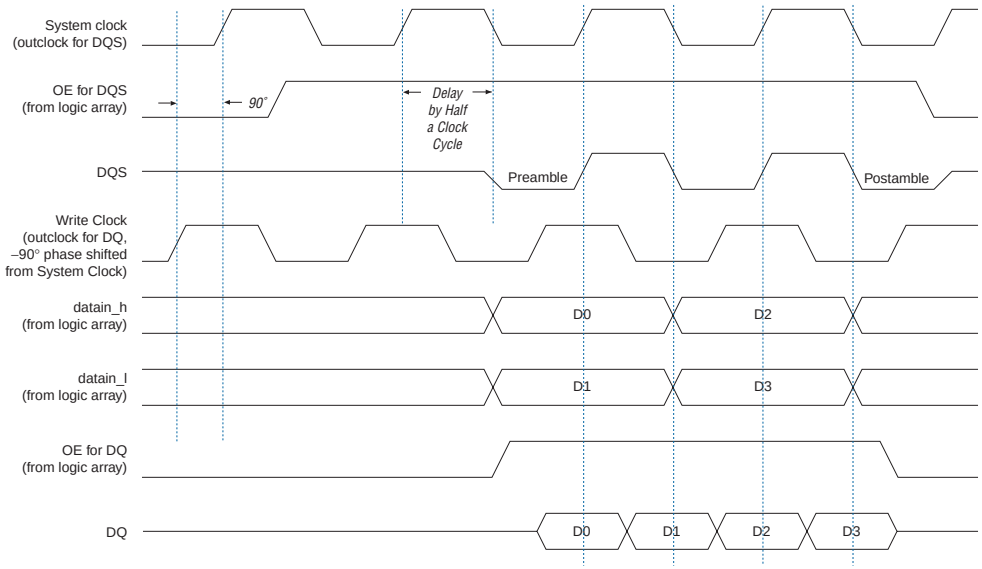
(4) This select line is to choose whether the OE signal should be delayed by half-a-clock cycle.

(5) The B_{OE} register generates the delayed enable signal for the write strobes and write clock for memory interfaces.

(6) The tristate enable is active low by default. You can design it to be active high. The combinational control path for the tristate is not shown in this diagram.

(7) You can also have combinational output to the I/O pin; this path is not shown in the diagram.

Figure 3–11. Extending the OE Disable by Half-a-Clock Cycle for a Write Transaction *Note (1)*

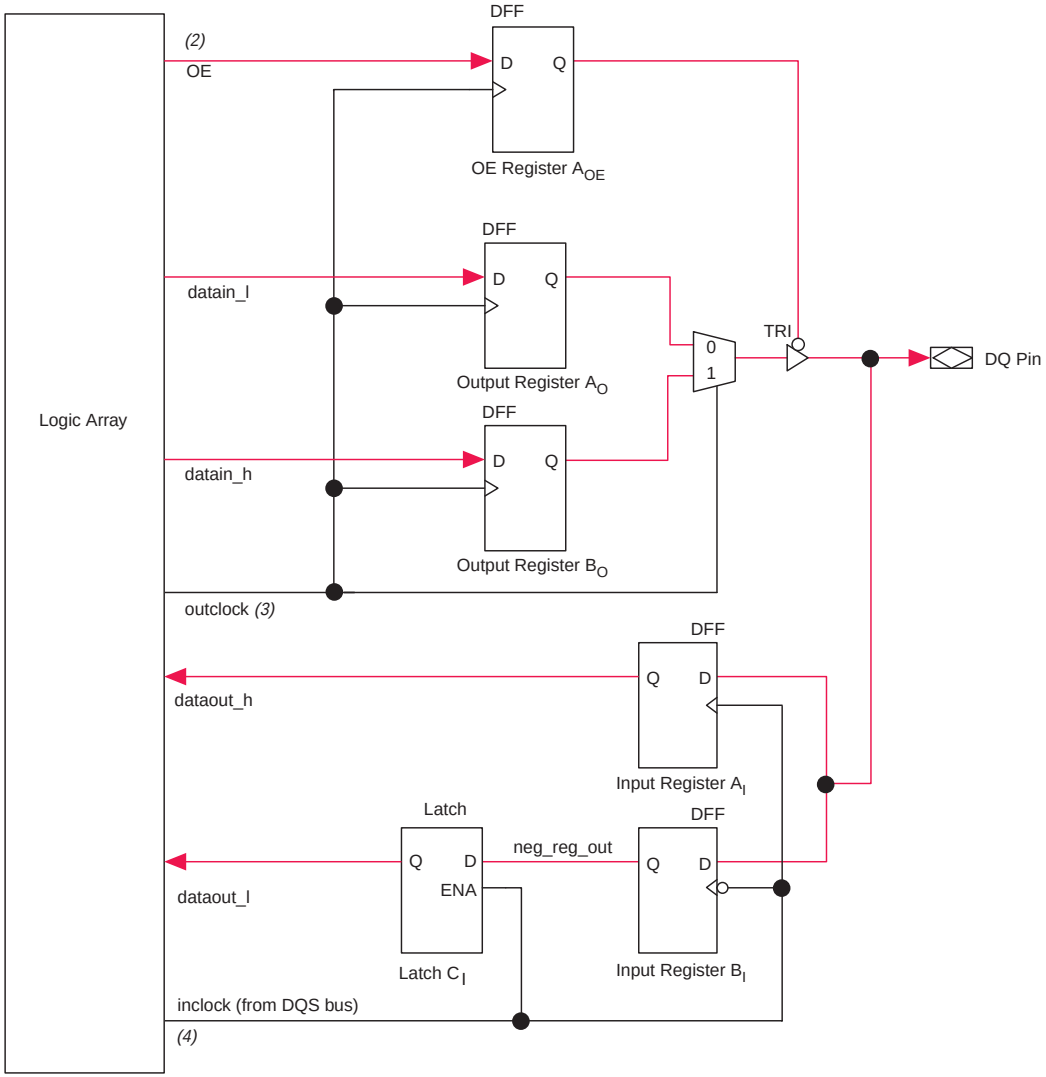


Note to Figure 3–11:

(1) The waveform reflects the software simulation result. The OE signal is an active low on the device. However, the Quartus II software implements this signal as an active high and automatically adds an inverter before the A_{OE} register D input.

Figures 3–12 and 3–13 summarize the IOE registers used for the DQ and DQS signals.

Figure 3–12. DQ Configuration in Stratix & Stratix GX IOE *Note (1)*



Notes to Figure 3–12:

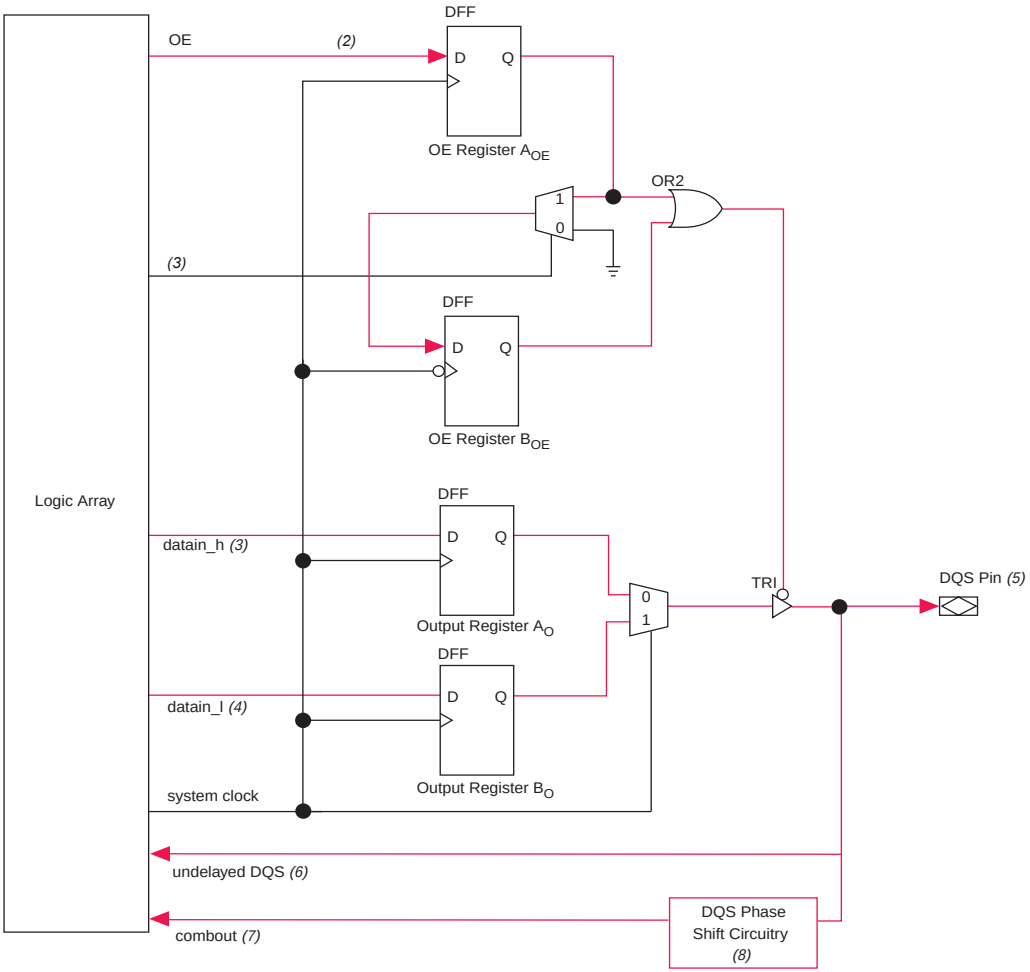
(1) You can use the `altdq` megafunction to generate the DQ signals.

(2) The `OE` signal is active low, but the Quartus II software implements this as active high and automatically adds an inverter before the `OE` register A_{OE} during compilation.

(3) The `outclock` signal is phase shifted –90° from the system clock.

(4) The shifted DQS signal must be inverted before going to the IOE. The inversion is automatic if you use the `altdq` megafunction to generate the DQ signals.

Figure 3–13. DQS Configuration in Stratix & Stratix GX IOE *Note (1)*



Notes to Figure 3–13:

(1) You can use the `altdqs` megafunction to generate the DQS signals.

(2) The `OE` signal is active low, but the Quartus II software implements this as active high and automatically adds an inverter before `OE` register A_{OE} during compilation.

(3) The select line can be chosen in the `altdqs` MegaWizard Plug-In Manager.

(4) The `datain_l` and `datain_h` pins are usually connected to V_{CC} and ground, respectively.

(5) DQS postamble handling is not shown in this diagram. For more information, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*.

(6) This undelayed DQS signal goes to the LE for the soft postamble circuitry.

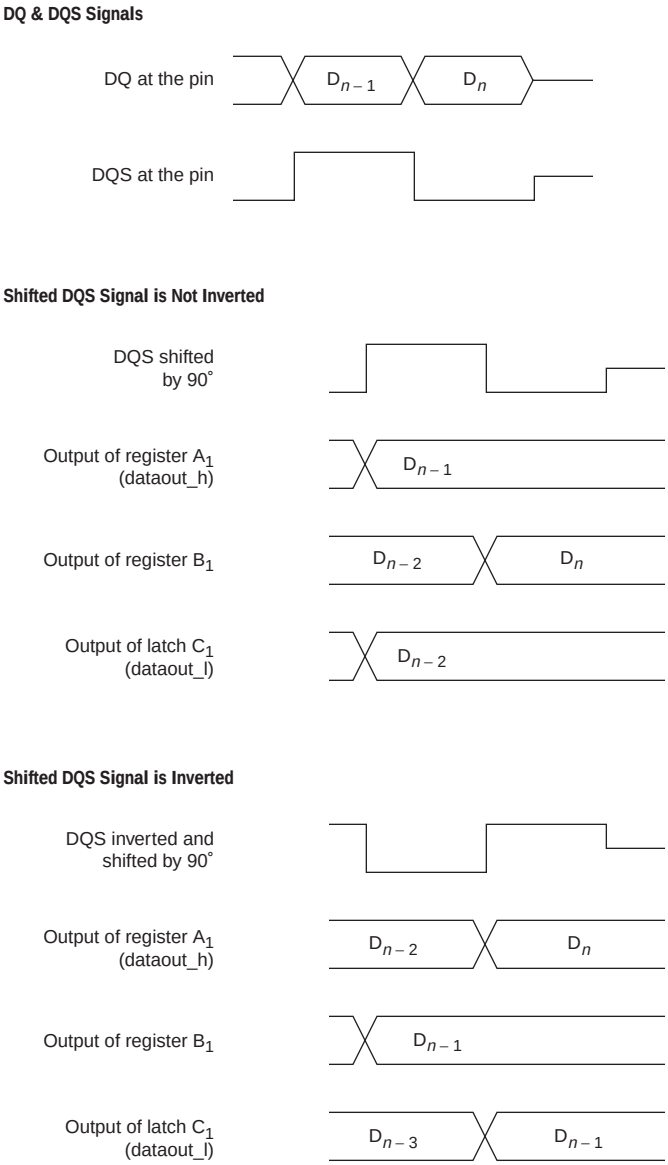
(7) You must invert this signal before it reaches the DQ IOE. This signal is automatically inverted if you use the `altdq` megafunction to generate the DQ signals. Connect this port to the `inclock` port in the `altdq` megafunction.

(8) DQS phase-shift circuitry is only available on DQS pins.

The Stratix and Stratix GX DDR IOE structure requires you to invert the incoming DQS signal by using a NOT gate to ensure proper data transfer. The `altdq` megafunction automatically adds the inverter when it generates the DQ signals. As shown in Figure 3–10, the `inclock` signal's rising edge clocks the A_I register, `inclock` signal's falling edge clocks the B_I register, and latch C_I is opened when `inclock` is one. In a DDR memory read operation, the last data coincides with DQS being low. If you do not invert the DQS pin, you do not get this last data because the latch does not open until the next rising edge of the DQS signal. The NOT gate is inserted automatically if the `altdg` megafunction is used; otherwise you need to add the NOT gate manually.

Figure 3–14 shows waveforms of the circuit shown in Figure 3–12. The second set of waveforms in Figure 3–14 shows what happens if the shifted DQS signal is not inverted; the last data, D_n, does not get latched into the logic array as DQS goes to tristate after the read postamble time. The third set of waveforms in Figure 3–14 shows a proper read operation with the DQS signal inverted after the 90° shift; the last data D_n does get latched. In this case the outputs of register A_I and latch C_I, which correspond to `dataout_h` and `dataout_l` ports, are now switched because of the DQS inversion.

Figure 3–14. DQ Captures with Non-Inverted & Inverted Shifted DQS



PLL

When using the Stratix and Stratix GX top and bottom I/O banks (I/O banks 3, 4, 7, or 8) to interface with a DDR memory, at least one PLL with two outputs is needed to generate the system clock and the write clock. The system clock generates the DQS write signals, commands, and addresses. The write clock is –90° shifted from the system clock and generates the DQ signals during writes.

When using the Stratix and Stratix GX side I/O banks 1, 2, 5, or 6 to interface with DDR SDRAM devices, two PLLs may be needed per I/O bank for best performance. The side I/O banks do not have dedicated circuitry, so one PLL captures data from the DDR SDRAM and another PLL generates the write signals, commands, and addresses to the DDR SDRAM device. Stratix and Stratix GX devices side I/O banks can support DDR SDRAM up to 150 MHz.

f For more information, see *AN 342: Interfacing DDR SDRAM with Stratix & Stratix GX Devices*.

Conclusion

Stratix and Stratix GX devices support SDR SDRAM, DDR SDRAM, RLDRAM II, QDR SDRAM, QDRII SRAM, and ZBT SRAM external memories. Stratix and Stratix GX devices feature high-speed interfaces that transfer data between external memory devices at up to 200 MHz/400 Mbps. Phase-shift circuitry in the Stratix and Stratix GX devices allows you to ensure that clock edges are properly aligned.

Section III. I/O Standards

This section provides information on Stratix® single-ended, voltage-referenced, and differential I/O standards.

It contains the following chapters:

- Chapter 4, Selectable I/O Standards in Stratix & Stratix GX Devices
- Chapter 5, High-Speed Differential I/O Interfaces in Stratix Devices

Revision History

The table below shows the revision history for Chapters 4 and 5.

Chapter	Date/Version	Changes Made	Comments
4	June 2006, v3.4	• Updated "AC Hot Socketing Specification" section.	
	July 2005, v3.3	• Updated "Non-Voltage-Referenced Standards" section. • Minor change to Table 4–6.	
	January 2005, v3.2	• Updated content throughout.	

Chapter	Date/Version	Changes Made	Comments
	September 2004, v3.1	• Table 4–1 on page 4–1: renamed table, updated table, and added Note 1. • Deleted Figure named “1.5-V Differential HSTL Class II Termination.” • Updated text describing “SSTL-18 Class I & II - EIA/JEDEC Preliminary Standard JC42.3” on page 4–11. • Updated HyperTransport data rates on page 4–17. • Changed HyperTransport device speed from 800 MHz to 400 MHz on page 4–17. • Added four rows to Table 4–2 on page 4–18: 1.5-V HSTL Class I, 1.8-V HSTL Class I, 1.5-V HSTL Class II, and 1.8-V HSTL Class II. • Changed title of Table 4–3 on page 4–21. • Updated Table 4–4 on page 4–22. • Updated Figure 4–20 on page 4–29. • Added description of which clock pins support differential on-chip termination on page 4–30. • Updated description of flip-chip packages on page 4–31. • Changed title of Figure 4–21 on page 4–31. • Updated milliamps for non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA packages on page 4–35. • Updated equation for FineLine BGA package on page 4–35. • Updated milliamps in non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA packages onpage 4–37.	
	April 2004, v3.0	• Updated notes to Figure 4–18. • New information added to the “Hot Socketing” section. • New information added to the “Differential Pad Placement Guidelines” section.	
	November 2003, v2.2	• Removed support for series and parallel on-chip termination. • Updated Figure 4–22.	
	October 2003, v2.1	• Added the Output Enable Group Logic Option in Quartus II and Toggle Rate Logic Option in Quartus II sections. • Updated notes to Table 4–10.	
	July 2003, v2.0	• Renamed impedance matching to series termination throughout Chapter. • Removed wide range specs for LVTTL and LVCMOS standards pages 4-3 to 4-5. • Relaxed restriction of input pins next to differential pins for flipchip packages (pages 4-20, 4-35, and 4-36). • Added Drive Strength section on page 4-26. • Removed text “for 10 ns or less” from AC Hot socketing specification on page 4-27. • Added Series Termination column to Table 4-9.	

Chapter	Date/Version	Changes Made	Comments
5	July 2005, v3.2	Updated Table 5–14 on page 5–58.	
	September 2004, v3.1	• Updated Note 3 in Table 5–10 on page 5–54. • Updated Table 5–7 on page 5–34. • Updated Table 5–8 on page 5–36. • Updated description of "R_D Differential Termination" on page 5–46. • Updated Note 5 in Table 5–14 on page 5–58. • Updated Notes 2, 5, and 7 in Table 5–11 on page 5–56 through Table 5–14 on page 5–58. • Added new text about spanning two I/O banks on page 5–60.	
	April 2004, v3.0	• Updated notes for Figure 5–17. • Updated Table 5–7, 5–8, and 5–10. • "Data Alignment with Clock" section, last sentence: change made from 90 degrees to 180 degrees.	
	November 2003, v2.2	• Removed support for series and parallel on-chip termination. • Updated the number of channels per PLL in Tables 5-10 through 5-14.	
	October 2003, v2.1	• Added -8 speed grade device information, including Tables 5-7 and 5-8.	
	July 2003, v2.0	• Format changes throughout Chapter. • Relaxed restriction of input pins next to differential pins for flip chip packages in Figure 5-1, Note 5. • Wire bond package performance specification for "high" speed channels was increased to 624 Mbps from 462 Mbps throughout Chapter. • Updated high-speed I/O specification for J=2 in Tables 5-7 and 5-8. • Updated Tables 5-10 to 5-14 to reflect PLL cross-bank support for high-speed differential channels at full speed. • Increased maximum output clock frequency to 462 to 500 MHz on page 5-66.	

4. Selectable I/O Standards in Stratix & Stratix GX Devices

S52004-3.4

Introduction

The proliferation of I/O standards and the need for higher I/O performance have made it critical that devices have flexible I/O capabilities. Stratix® and Stratix GX programmable logic devices (PLDs) feature programmable I/O pins that support a wide range of industry I/O standards, permitting increased design flexibility. These I/O capabilities enable fast time-to-market and high-performance solutions to meet the demands of complex system designs. Additionally, Stratix and Stratix GX devices simplify system board design and make it easy to connect to microprocessors, peripherals, memories, gate arrays, programmable logic circuits, and standard logic functions.

This chapter provides guidelines for using one or more industry I/O standards in Stratix and Stratix GX devices, including:

- Stratix and Stratix GX I/O standards
- High-speed interfaces
- Stratix and Stratix GX I/O banks
- Programmable current drive strength
- Hot socketing
- Differential on-chip termination
- I/O pad placement guidelines
- Quartus® II software support

Stratix & Stratix GX I/O Standards

Stratix and Stratix GX devices support a wide range of industry I/O standards as shown in the *Stratix Device Family Data Sheet* section in the *Stratix Device Handbook, Volume 1* and the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*. Several applications that use these I/O standards are listed in Table 4–1.

Table 4–1. I/O Standard Applications & Performance (Part 1 of 2) *Note (1)*

I/O Standard	Application	Performance
3.3-V LVTTL/LVCMOS	General purpose	350 MHz
2.5-V LVTTL/LVCMOS	General purpose	350 MHz
1.8-V LVTTL/LVCMOS	General purpose	250 MHz
1.5-V LVCMOS	General purpose	225 MHz
PCI/CompactPCI	PC/embedded systems	66 MHz

Table 4–1. I/O Standard Applications & Performance (Part 2 of 2) *Note (1)*

I/O Standard	Application	Performance
PCI-X 1.0	PC/embedded systems	133 MHz
AGP 1× and 2×	Graphics processors	66 to 133 MHz
SSTL-3 Class I and II	SDRAM	167 MHz
SSTL-2 Class I and II	DDR I SDRAM	160 to 400 Mbps
HSTL Class I	QDR SRAM/SRAM/CSIX	150 to 225 MHz
HSTL Class II	QDR SRAM/SRAM/CSIX	150 to 250 MHz
Differential HSTL	Clock interfaces	150 to 225 MHz
GTL	Backplane driver	200 MHz
GTL+	Pentium processor interface	133 to 200 MHz
LVDS	Communications	840 Mbps
HyperTransport technology	Motherboard interfaces	800 Mbps
LVPECL	PHY interface	840 Mbps
PCML	Communications	840 Mbps
Differential SSTL-2	DDR I SDRAM	160 to 400 Mbps
CTT	Back planes and bus interfaces	200 MHz

Note to Table 4–1:

(1) These performance values are dependent on device speed grade, package type (flip-chip or wirebond) and location of I/Os (top/bottom or left/right). See the *DC & Switching Characteristics* chapter of the *Stratix Device Handbook, Volume 1*.

3.3-V Low Voltage Transistor-Transistor Logic (LVTTL) - EIA/JEDEC Standard JESD8-B

The 3.3-V LVTTL I/O standard is a general-purpose, single-ended standard used for 3.3-V applications. The LVTTL standard defines the DC interface parameters for digital circuits operating from a 3.0-V or 3.3-V power supply and driving or being driven by LVTTL-compatible devices.

The LVTTL input standard specifies a wider input voltage range of $-0.5\ V \leq V_I \leq 3.8\ V$. Altera allows an input voltage range of $-0.5\ V \leq V_I \leq 4.1\ V$. The LVTTL standard does not require input reference voltages or board terminations.

Stratix and Stratix GX devices support both input and output levels for 3.3-V LVTTL operation.

3.3-V LVCMOS - EIA/JEDEC Standard JESD8-B

The 3.3-V low voltage complementary metal oxide semiconductor (LVCMOS) I/O standard is a general-purpose, single-ended standard used for 3.3-V applications. The LVCMOS standard defines the DC interface parameters for digital circuits operating from a 3.0-V or 3.3-V power supply and driving or being driven by LVCMOS-compatible devices.

The LVCMOS standard specifies the same input voltage requirements as LVTTL ($-0.5\ V \leq V_I \leq 3.8\ V$). The output buffer drives to the rail to meet the minimum high-level output voltage requirements. The 3.3-V I/O standard does not require input reference voltages or board terminations.

Stratix and Stratix GX devices support both input and output levels for 3.3-V LVCMOS operation.

2.5-V LVTTL Normal Voltage Range - EIA/JEDEC Standard EIA/JESD8-5

The 2.5-V I/O standard is used for 2.5-V LVTTL applications. This standard defines the DC interface parameters for high-speed, low-voltage, non-terminated digital circuits driving or being driven by other 2.5-V devices. The input and output voltage ranges are:

- The 2.5-V normal range input standards specify an input voltage range of $-0.3\ V \leq V_I \leq 3.0\ V$.
- The normal range minimum high-level output voltage requirement (V_{OH}) is 2.1 V.

Stratix and Stratix GX devices support both input and output levels for 2.5-V LVTTL operation.

2.5-V LVCMOS Normal Voltage Range - EIA/JEDEC Standard EIA/JESD8-5

The 2.5-V I/O standard is used for 2.5-V LVCMOS applications. This standard defines the DC interface parameters for high-speed, low-voltage, non-terminated digital circuits driving or being driven by other 2.5-V parts. The input and output voltage ranges are:

- The 2.5-V normal range input standards specify an input voltage range of $-0.5\ V \leq V_I \leq 3.0\ V$.
- The normal range minimum V_{OH} requirement is 2.1 V.

Stratix and Stratix GX devices support both input and output levels for 2.5-V LVCMOS operation.

1.8-V LVTTL Normal Voltage Range - EIA/JEDEC Standard EIA/JESD8-7

The 1.8-V I/O standard is used for 1.8-V LVTTL applications. This standard defines the DC interface parameters for high-speed, low-voltage, non-terminated digital circuits driving or being driven by other 1.8-V parts. The input and output voltage ranges are:

- The 1.8-V normal range input standards specify an input voltage range of $-0.5\ V \leq V_I \leq 2.3\ V$.
- The normal range minimum V_{OH} requirement is $V_{CCIO} - 0.45\ V$.

Stratix and Stratix GX devices support both input and output levels for 1.8-V LVTTL operation.

1.8-V LVCMOS Normal Voltage Range - EIA/JEDEC Standard EIA/JESD8-7

The 1.8-V I/O standard is used for 1.8-V LVCMOS applications. This standard defines the DC interface parameters for high-speed, low-voltage, non-terminated digital circuits driving or being driven by other 1.8-V devices. The input and output voltage ranges are:

- The 1.8-V normal range input standards specify an input voltage range of $-0.5\ V \leq V_I \leq 2.5\ V$.
- The normal range minimum V_{OH} requirement is $V_{CCIO} - 0.45\ V$.

Stratix and Stratix GX devices support both input and output levels for 1.8-V LVCMOS operation.

1.5-V LVCMOS Normal Voltage Range - EIA/JEDEC Standard JESD8-11

The 1.5-V I/O standard is used for 1.5-V applications. This standard defines the DC interface parameters for high-speed, low-voltage, non-terminated digital circuits driving or being driven by other 1.5-V devices. The input and output voltage ranges are:

- The 1.5-V normal range input standards specify an input voltage range of $-0.5\ V \leq V_I \leq 2.0\ V$.
- The normal range minimum V_{OH} requirement is 1.05 V.

Stratix and Stratix GX devices support both input and output levels for 1.5-V LVCMOS operation.

1.5-V HSTL Class I & II - EIA/JEDEC Standard EIA/JESD8-6

The high-speed transceiver logic (HSTL) I/O standard is used for applications designed to operate in the 0.0- to 1.5-V HSTL logic switching range. This standard defines single ended input and output specifications for all HSTL-compliant digital integrated circuits. The single ended input standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. Stratix and Stratix GX devices support both input and output levels specified by the 1.5-V HSTL I/O standard. The input clock is implemented using dedicated differential input buffers. Two single-ended output buffers are automatically programmed to have opposite polarity so as to implement a differential output clock. Additionally, the 1.5-V HSTL I/O standard in Stratix and Stratix GX devices is compatible with the 1.8-V HSTL I/O standard in APEX™ 20KE and APEX 20KC devices because the input and output voltage thresholds are compatible. See Figures 4–1 and 4–2. Stratix and Stratix GX devices support both input and output levels with V_{REF} and V_{TT}.

Figure 4–1. HSTL Class I Termination

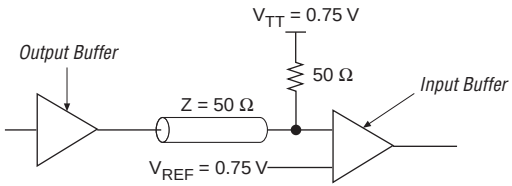
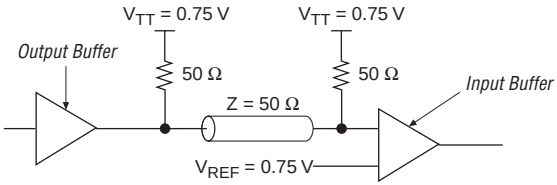


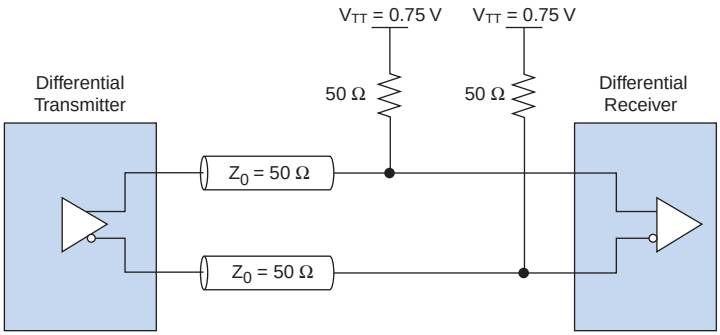
Figure 4–2. HSTL Class II Termination



1.5-V Differential HSTL - EIA/JEDEC Standard EIA/JESD8-6

The differential HSTL I/O standard is used for applications designed to operate in the 0.0- to 1.5-V HSTL logic switching range such as quad data rate (QDR) memory clock interfaces. The differential HSTL specification is the same as the single ended HSTL specification. The standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. Differential HSTL does not require an input reference voltage, however, it does require a 50 Ω resistor termination resistor to V_{TT} at the input buffer (see Figure 4–3). Stratix and Stratix GX devices support both input and output clock levels for 1.5-V differential HSTL. The input clock is implemented using dedicated differential input buffer. Two single-ended output buffers are automatically programmed to have opposite polarity so as to implement a differential output clock.

Figure 4–3. 1.5-V Differential HSTL Class I Termination



3.3-V PCI Local Bus - PCI Special Interest Group PCI Local Bus Specification Rev. 2.3

The PCI local bus specification is used for applications that interface to the PCI local bus, which provides a processor-independent data path between highly integrated peripheral controller components, peripheral add-in boards, and processor/memory systems. The conventional PCI specification revision 2.3 defines the PCI hardware environment including the protocol, electrical, mechanical, and configuration specifications for the PCI devices and expansion boards. This standard requires 3.3-V V_{CCIO}. Stratix and Stratix GX devices are fully compliant with the 3.3-V *PCI Local Bus Specification Revision 2.3* and meet 64-bit/66-MHz operating frequency and timing requirements. The 3.3-V PCI standard does not require input reference voltages or board terminations. Stratix and Stratix GX devices support both input and output levels.

3.3-V PCI-X 1.0 Local Bus - PCI-SIG PCI-X Local Bus Specification Revision 1.0a

The PCI-X 1.0 standard is used for applications that interface to the PCI local bus. The standard enables the design of systems and devices that operate at clock speeds up to 133 MHz, or 1 gigabit per second (Gbps) for a 64-bit bus. The PCI-X 1.0 protocol enhancements enable devices to operate much more efficiently, providing more usable bandwidth at any clock frequency. By using the PCI-X 1.0 standard, devices can be designed to meet PCI-X 1.0 requirements and operate as conventional 33- and 66-MHz PCI devices when installed in those systems. This standard requires 3.3-V $V_{CCIO.}$ Stratix and Stratix GX devices are fully compliant with the 3.3-V *PCI-X Specification Revision 1.0a* and meet the 133-MHz operating frequency and timing requirements. The 3.3-V PCI standard does not require input reference voltages or board terminations. Stratix and Stratix GX devices support both input and output levels.

3.3-V Compact PCI Bus - PCI SIG PCI Local Bus Specification Revision 2.3

The Compact PCI local bus specification is used for applications that interface to the PCI local bus. It follows the *PCI Local Bus Specification Revision 2.3* plus additional requirements in PCI Industrial Computers Manufacturing Group (PICMG) specifications PICMG 2.0 R3.0, CompactPCI specification, and the hot swap requirements in PICMG 2.1 R2.0, CompactPCI Hot Swap Specification. This standard has similar electrical requirements as LVTTL and requires 3.3-V $V_{CCIO.}$ Stratix and Stratix GX devices are compliant with the Compact PCI electrical requirements. The 3.3-V PCI standard does not require input reference voltages or board terminations. Stratix and Stratix GX devices support both input and output levels.

3.3-V 1× AGP - Intel Corporation Accelerated Graphics Port Interface Specification 2.0

The AGP interface is a platform bus specification that enables high-performance graphics by providing a dedicated high-speed port for the movement of large blocks of 3-dimensional texture data between a PC's graphics controller and system memory. The 1× AGP I/O standard is a single-ended standard used for 3.3-V graphics applications. The 1× AGP input standard specifies an input voltage range of $-0.5\ V \leq V_I \leq V_{CCIO} + 0.5\ V$. The 1× AGP standard does not require input reference voltages or board terminations. Stratix and Stratix GX devices support both input and output levels.

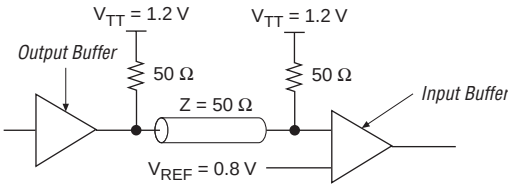
3.3-V 2× AGP - Intel Corporation Accelerated Graphics Port Interface Specification 2.0

The 2× AGP I/O standard is a voltage-referenced, single-ended standard used for 3.3-V graphics applications. The 2× AGP input standard specifies an input voltage range of $-0.5V \leq V_I \leq V_{CCIO} + 0.5V$. The 2× AGP standard does not require board terminations. Stratix and Stratix GX devices support both input and output levels.

GTL - EIA/JEDEC Standard EIA/JESD8-3

The GTL I/O standard is a low-level, high-speed back plane standard used for a wide range of applications from ASICs and processors to interface logic devices. The GTL standard defines the DC interface parameters for digital circuits operating from power supplies of 2.5, 3.3, and 5.0 V. The GTL standard is an open-drain standard, and Stratix and Stratix GX devices support a 2.5- or 3.3-V V_{CCIO} to meet this standard. GTL requires a 0.8-V V_{REF} and open-drain outputs with a 1.2-V V_{TT} (see Figure 4–4). Stratix and Stratix GX devices support both input and output levels.

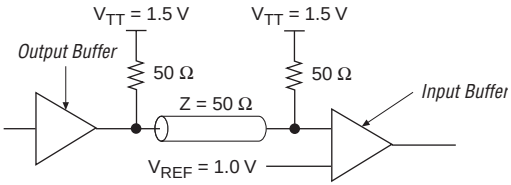
Figure 4–4. GTL Termination



GTL+

The GTL+ I/O standard is used for high-speed back plane drivers and Pentium processor interfaces. The GTL+ standard defines the DC interface parameters for digital circuits operating from power supplies of 2.5, 3.3, and 5.0 V. The GTL+ standard is an open-drain standard, and Stratix and Stratix GX devices support a 2.5- or 3.3-V V_{CCIO} to meet this standard. GTL+ requires a 1.0-V V_{REF} and open-drain outputs with a 1.5-V V_{TT} (see Figure 4–5). Stratix and Stratix GX devices support both input and output levels.

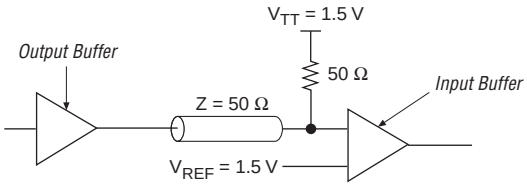
Figure 4–5. GTL+ Termination



CTT - EIA/JEDEC Standard JESD8-4

The CTT I/O standard is used for backplanes and memory bus interfaces. The CTT standard defines the DC interface parameters for digital circuits operating from 2.5- and 3.3-V power supplies. The CTT standard does not require special circuitry to interface with LVTTL or LVCMOS devices when the CTT driver is not terminated. The CTT standard requires a 1.5-V V_{REF} and a 1.5-V V_{TT} (see Figure 4–6). Stratix and Stratix GX devices support both input and output levels.

Figure 4–6. CTT Termination



SSTL-3 Class I & II - EIA/JEDEC Standard JESD8-8

The SSTL-3 I/O standard is a 3.3-V memory bus standard used for applications such as high-speed SDRAM interfaces. This standard defines the input and output specifications for devices that operate in the SSTL-3 logic switching range of 0.0 to 3.3 V. The SSTL-3 standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. SSTL-3 requires a 1.5-V V_{REF} and a 1.5-V V_{TT} to which the series and termination resistors are connected (see Figures 4–7 and 4–8). Stratix and Stratix GX devices support both input and output levels.

Figure 4–7. SSTL-3 Class I Termination

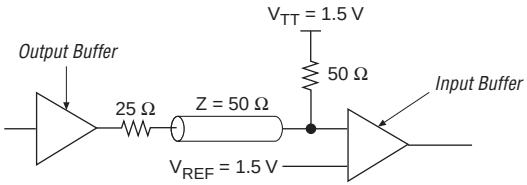
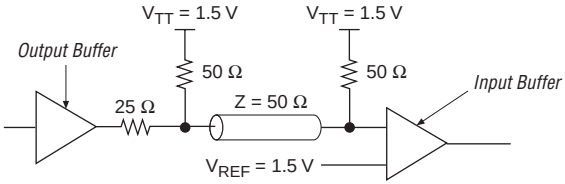


Figure 4–8. SSTL-3 Class II Termination



SSTL-2 Class I & II - EIA/JEDEC Standard JESD8-9A

The SSTL-2 I/O standard is a 2.5-V memory bus standard used for applications such as high-speed DDR SDRAM interfaces. This standard defines the input and output specifications for devices that operate in the SSTL-2 logic switching range of 0.0 to 2.5 V. This standard improves operation in conditions where a bus must be isolated from large stubs. The SSTL-2 standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. SSTL-2 requires a 1.25-V V_{REF} and a 1.25-V V_{TT} to which the series and termination resistors are connected (see Figures 4–9 and 4–10). Stratix and Stratix GX devices support both input and output levels.

Figure 4–9. SSTL-2 Class I Termination

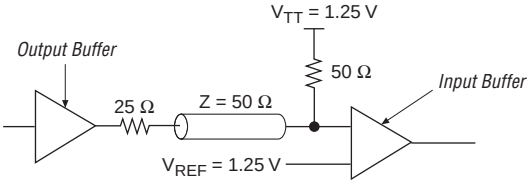
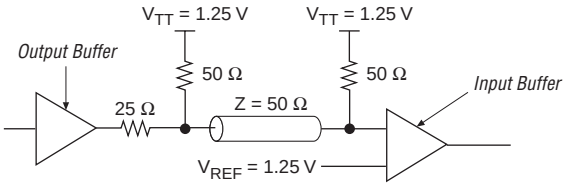


Figure 4–10. SSTL-2 Class II Termination



SSTL-18 Class I & II - EIA/JEDEC Preliminary Standard JC42.3

The SSTL-18 I/O standard is a 1.8-V memory bus standard. This standard is similar to SSTL-2 and defines input and output specifications for devices that are designed to operate in the SSTL-18 logic switching range 0.0 to 1.8 V. SSTL-18 requires a 0.9-V V_{REF} and a 0.9-V V_{TT} to which the series and termination resistors are connected. See Figures 4–11 and 4–12 for details on SSTL-18 Class I and II termination. Stratix and Stratix GX devices support both input and output levels.

Figure 4–11. SSTL-18 Class I Termination

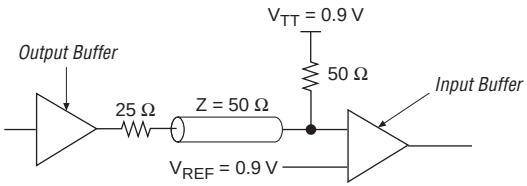
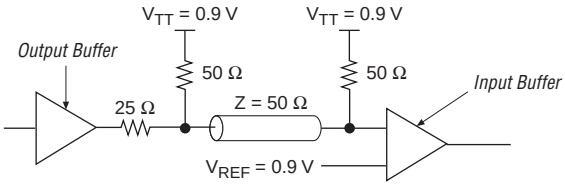


Figure 4–12. SSTL-18 Class II Termination

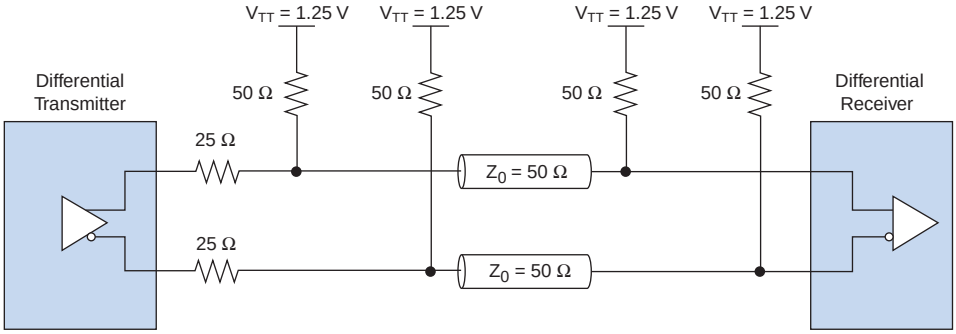


Differential SSTL-2 - EIA/JEDEC Standard JESD8-9A

The differential SSTL-2 I/O standard is a 2.5-V standard used for applications such as high-speed DDR SDRAM clock interfaces. This standard supports differential signals in systems using the SSTL-2

standard and supplements the SSTL-2 standard for differential clocks. The differential SSTL-2 standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. The differential SSTL-2 standard does not require an input reference voltage differential. See Figure 4–13 for details on differential SSTL-2 termination. Stratix and Stratix GX devices support output clock levels for differential SSTL-2 Class II operation. The output clock is implemented using two single-ended output buffers which are programmed to have opposite polarity.

Figure 4–13. Differential SSTL-2 Class II Termination



LVDS - ANSI/TIA/EIA Standard ANSI/TIA/EIA-644

The LVDS I/O standard is a differential high-speed, low-voltage swing, low-power, general-purpose I/O interface standard requiring a 3.3-V V_{CCIO}. This standard is used in applications requiring high-bandwidth data transfer, backplane drivers, and clock distribution. The ANSI/TIA/EIA-644 standard specifies LVDS transmitters and receivers capable of operating at recommended maximum data signaling rates of 655 Mbps. However, devices can operate at slower speeds if needed, and there is a theoretical maximum of 1.923 Gbps. Stratix and Stratix GX devices meet the ANSI/TIA/EIA-644 standard.

Due to the low voltage swing of the LVDS I/O standard, the electromagnetic interference (EMI) effects are much smaller than CMOS, TTL, and PECL. This low EMI makes LVDS ideal for applications with low EMI requirements or noise immunity requirements. The LVDS standard does not require an input reference voltage, however, it does require a 100 Ω termination resistor between the two signals at the input buffer. Stratix and Stratix GX devices include an optional differential LVDS termination resistor within the device using differential on-chip termination. Stratix and Stratix GX devices support both input and output levels.

f For more information on the LVDS I/O standard in Stratix devices, see the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter.

LVPECL

The LVPECL I/O standard is a differential interface standard requiring a 3.3-V V_{CCIO}. The standard is used in applications involving video graphics, telecommunications, data communications, and clock distribution. The high-speed, low-voltage swing LVPECL I/O standard uses a positive power supply and is similar to LVDS, however, LVPECL has a larger differential output voltage swing than LVDS. The LVPECL standard does not require an input reference voltage, but it does require a 100-Ω termination resistor between the two signals at the input buffer. See Figures 4–14 and 4–15 for two alternate termination schemes for LVPECL. Stratix and Stratix GX devices support both input and output levels.

Figure 4–14. LVPECL DC Coupled Termination

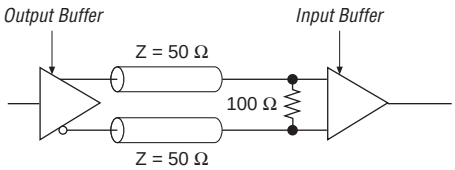
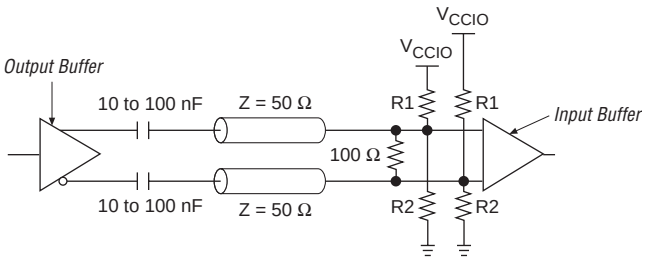


Figure 4–15. LVPECL AC Coupled Termination



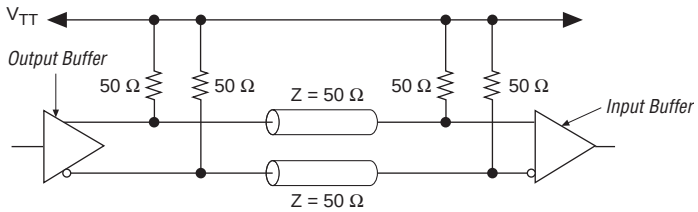
Pseudo Current Mode Logic (PCML)

The PCML I/O standard is a differential high-speed, low-power I/O interface standard used in applications such as networking and telecommunications. The standard requires a 3.3-V V_{CCIO}. The PCML I/O standard consumes less power than the LVPECL I/O standard. The

PCML standard is similar to LVPECL, but PCML has a reduced voltage swing, which allows for a faster switching time and lower power consumption. The PCML standard uses open drain outputs and requires a differential output signal. See Figure 4–16 for details on PCML termination. Stratix and Stratix GX devices support both input and output levels.

Additionally, Stratix GX devices support 1.5-V PCML as described in the *Stratix GX Device Handbook, Volume 1*.

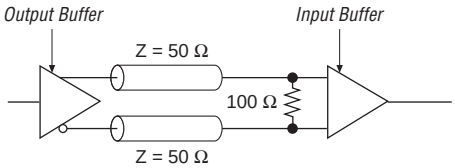
Figure 4–16. PCML Termination



HyperTransport Technology - HyperTransport Consortium

The HyperTransport technology I/O standard is a differential high-speed, high-performance I/O interface standard requiring a 2.5-V V_{CCIO}. This standard is used in applications such as high-performance networking, telecommunications, embedded systems, consumer electronics, and Internet connectivity devices. The HyperTransport technology I/O standard is a point-to-point standard in which each HyperTransport technology bus consists of two point-to-point unidirectional links. Each link is 2 to 32 bits. The HyperTransport technology standard does not require an input reference voltage. However, it does require a 100-Ω termination resistor between the two signals at the input buffer. See Figure 4–17 for details on HyperTransport technology termination. Stratix and Stratix GX devices support both input and output levels.

Figure 4–17. HyperTransport Technology Termination



f See the *Stratix Device Family Data Sheet* section in the *Stratix Device Handbook, Volume 1*; the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*; and the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter for more information on differential I/O standards.

High-Speed Interfaces

In addition to current industry physical I/O standards, Stratix and Stratix GX devices also support a variety of emerging high-speed interfaces. This section provides an overview of these interfaces.

OIF-SPI4.2

This implementation agreement is widely used in the industry for OC-192 and 10-Gbps multi-service system interfaces. SONET and SDH are synchronous transmission systems over which data packets are transferred. POS-PHY Level 4 is a standard interface for switches and routers, and defines the operation between a physical layer (PHY) device and link layer devices (ATM, Internet protocol, and Gigabit Ethernet) for bandwidths of OC-192 ATM, POS, and 10-Gigabit Ethernet applications. Some key POS-PHY Level 4 system features include:

- Large selection of POS-PHY Level 4-based PHYs
- Independent of data protocol
- Wide industry support
- LVDS I/O standard to improve signal integrity
- Inband addressing/control
- Out of band flow control
- Scalable architecture
- Over 622-Mbps operation
- Dynamic interface timing mode

POS-PHY Level 4 operates at a wide range of frequencies.

OIF-SFI4.1

This implementation agreement is widely used in the industry for interfacing physical layer (PHY) to the serializer-deserializer (SERDES) devices in OC-192 and 10 Gbps multi-service systems. The POS-PHY Level 4 interface standard defines the SFI-4 standard. POS-PHY Level 4: SFI-4 is a standardized 16-bit × 622-Mbps line-side interface for 10-Gbps applications. Internet LAN and WAN architectures use telecommunication SONET protocols for data transferring data over the PHY layer. SFI-4 interfaces between OC-192 SERDES and SONET framers.

10 Gigabit Ethernet Sixteen Bit Interface (XSBI) - IEEE Draft Standard P802.3ae/D2.0

10 Gigabit Ethernet XSBI is an interface standard for LANs, metropolitan area networks (MANs), storage area networks (SANs), and WANs. 10 Gigabit Ethernet XSBI provides many features for efficient, effective high-speed networking, including easy migration to higher performance levels without disruption, lower cost of ownership including acquisition and support versus other alternatives, familiar management tools and common skills, ability to support new applications and data protocols, flexibility in network design, and multiple vendor sourcing and interoperability.

Under the ISO Open Systems Interconnection (OSI) model, Ethernet is a Layer 2 protocol. 10 Gigabit Ethernet XSBI uses the IEEE 802.3 Ethernet media access control (MAC) protocol, Ethernet frame format, and the minimum / maximum frame size. An Ethernet PHY corresponding to OSI layer 1 connects the media to the MAC layer that corresponds to OSI layer 2. The PHY is divided into a physical media dependent (PMD) element, such as optical transceivers, and a physical coding sub-layer (PCS), which has coding and a serializer / multiplexor. This standard defines two PHY types, including the LAN PHY and the WAN PHY, which are distinguished by the PCS. The 10 Gigabit Ethernet XSBI standard is a full-duplex technology standard that can increase the speed and distance of Ethernet.

RapidIO Interconnect Specification Revision 1.1

The RapidIO interface is a communications standard used to connect devices on a circuit board and circuit boards on a backplane. RapidIO is a packet-switched interconnect standard designed for embedded systems such as those used in networking and communications. The RapidIO interface standard is a high-performance interconnect interface used for transferring data and control information between microprocessors, DSPs, system memory, communications and network processors, and peripheral devices in a system.

RapidIO replaces existing peripheral bus and processor technologies such as PCI. Some features of RapidIO include multiprocessing support, an open standard, flexible topologies, higher bandwidth, low latency, error management support in hardware, small silicon footprint, widely available process and I / O technologies, and transparency to existing applications and operating system software. The RapidIO standard provides 10-Gbps device bandwidth using 8-bit-wide input and output data ports. RapidIO uses LVDS technology, has the capability to be scaled to multi-GHz frequencies, and features a 10-bit interface.

HyperTransport Technology - HyperTransport Consortium

The HyperTransport technology I/O standard is a differential high-speed, high performance I/O interface standard developed for communications and networking chip-to-chip communications. HyperTransport technology is used in applications such as high-performance networking, telecommunications, embedded systems, consumer electronics, and Internet connectivity devices. The HyperTransport technology I/O standard is a point-to-point (one source connected to exactly one destination) standard that provides a high-performance interconnect between integrated circuits in a system, such as on a motherboard.

Stratix devices support HyperTransport technology at data rates up to 800 Mbps and 32 bits in each direction. HyperTransport technology uses an enhanced differential signaling technology to improve performance. HyperTransport technology supports data widths of 2, 4, 8, 16, or 32 bits in each direction. HyperTransport technology in Stratix and Stratix GX devices operates at multiple clock speeds up to 400 MHz.

UTOPIA Level 4 – ATM Forum Technical Committee Standard AF-PHY-0144.001

The UTOPIA Level 4 frame-based interface standard allows device manufacturers and network developers to develop components that can operate at data rates up to 10 Gbps. This standard increases interface speeds using LVDS I/O and advanced silicon technologies for fast data transfers.

UTOPIA Level 4 provides new control techniques and a 32-, 16-, or 8-bit LVDS bus, a symmetric transmit/receive bus structure for easier application design and testability, nominal data rates of 10 Gbps, in-band control of cell delimiters and flow control to minimize pin count, source-synchronous clocking, and supports variable length packet systems. UTOPIA Level 4 handles sustained data rates for OC-192 and supports ATM cells. UTOPIA Level 4 also supports interconnections across motherboards, daughtercards, and backplane interfaces.

Stratix & Stratix GX I/O Banks

Stratix devices have eight I/O banks in addition to the four enhanced PLL external clock output banks, as shown in Table 4–2 and Figure 4–18. I/O banks 3, 4, 7, and 8 support all single-ended I/O standards. I/O banks 1, 2, 5, and 6 support differential HSTL (on input clocks), LVDS, LVPECL, PCML, and HyperTransport technology, as well as all single-ended I/O standards except HSTL Class II, GTL, SSTL-18 Class II, PCI/PCI-X 1.0, and 1×/2× AGP. The four enhanced PLL external clock output banks (I/O banks 9, 10, 11, and 12) support clock outputs all single-ended I/O

standards in addition to differential SSTL-2 and HSTL (both on the output clock only). Since Stratix devices support both non-voltage-referenced and voltage-referenced I/O standards, there are different guidelines when working with either separately or when working with both.

Table 4–2. I/O Standards Supported in Stratix I/O Banks (Part 1 of 2)

I/O Standard	I/O Bank								Enhanced PLL External Clock Output Banks			
	1	2	3	4	5	6	7	8	9	10	11	12
3.3-V LVTTL/LVCMOS	v	v	v	v	v	v	v	v	v	v	v	v
2.5-V LVTTL/LVCMOS	v	v	v	v	v	v	v	v	v	v	v	v
1.8-V LVTTL/LVCMOS	v	v	v	v	v	v	v	v	v	v	v	v
1.5-V LVCMOS	v	v	v	v	v	v	v	v	v	v	v	v
PCI/PCIX//Compact PCI			v	v			v	v	v	v	v	v
AGP 1×			v	v			v	v	v	v	v	v
AGP 2×			v	v			v	v	v	v	v	v
SSTL-3 Class I	v	v	v	v	v	v	v	v	v	v	v	v
SSTL-3 Class II	v	v	v	v	v	v	v	v	v	v	v	v
SSTL-2 Class I	v	v	v	v	v	v	v	v	v	v	v	v
SSTL-2 Class II	v	v	v	v	v	v	v	v	v	v	v	v
SSTL-18 Class I	v	v	v	v	v	v	v	v	v	v	v	v
SSTL-18 Class II			v	v			v	v	v	v	v	v
Differential SSTL-2 (output clocks)									v	v	v	v
HSTL Class I	v	v	v	v	v	v	v	v	v	v	v	v
1.5-V HSTL Class I	v	v	v	v	v	v	v	v	v	v	v	v
1.8-V HSTL Class I	v	v	v	v	v	v	v	v	v	v	v	v
HSTL Class II			v	v			v	v	v	v	v	v
1.5-V HSTL Class II			v	v			v	v	v	v	v	v
1.8-V HSTL Class II			v	v			v	v	v	v	v	v
Differential HSTL (input clocks)	v	v	v	v	v	v	v	v				
Differential HSTL (output clocks)									v	v	v	v
GTL			v	v			v	v	v	v	v	v

Table 4–2. I/O Standards Supported in Stratix I/O Banks (Part 2 of 2)

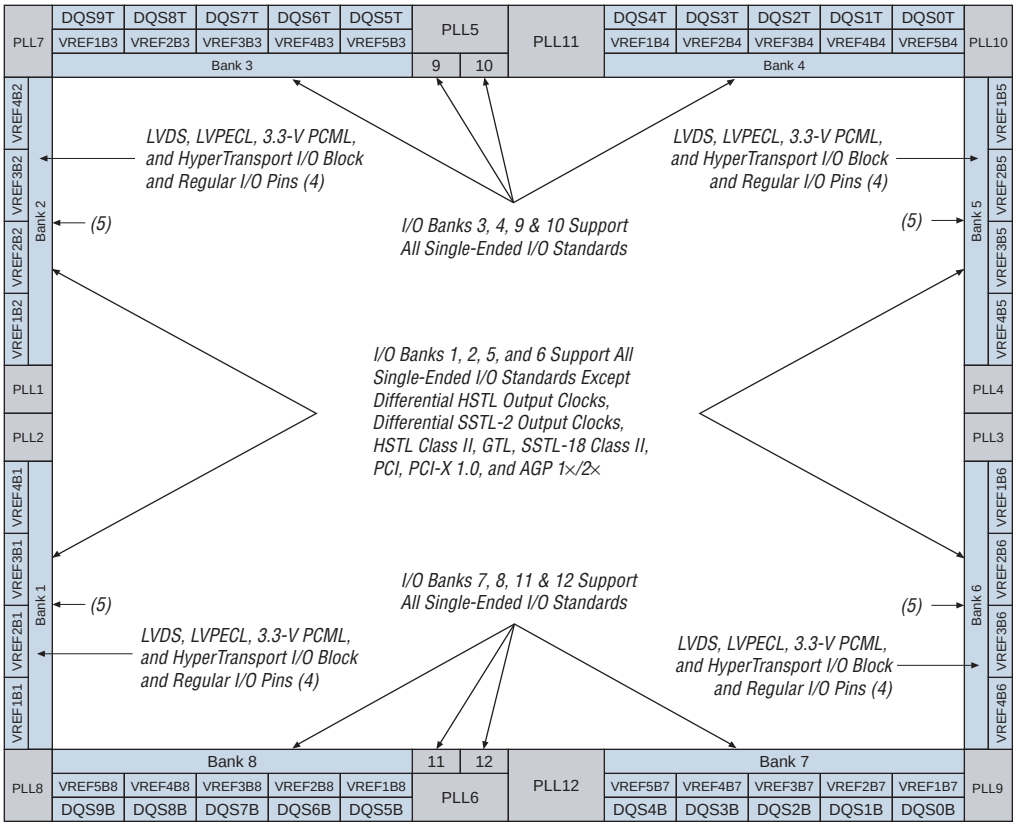
I/O Standard	I/O Bank								Enhanced PLL External Clock Output Banks			
	1	2	3	4	5	6	7	8	9	10	11	12
GTL+	v	v	v	v	v	v	v	v	v	v	v	v
CTT	v	v	v	v	v	v	v	v	v	v	v	v
LVDS	v	v	*(1)*	*(1)*	v	v	*(1)*	*(1)*	*(2)*	*(2)*	*(2)*	*(2)*
HyperTransport technology	v	v	*(1)*	*(1)*	v	v	*(1)*	*(1)*	*(2)*	*(2)*	*(2)*	*(2)*
LVPECL	v	v	*(1)*	*(1)*	v	v	*(1)*	*(1)*	*(2)*	*(2)*	*(2)*	*(2)*
PCML	v	v	*(1)*	*(1)*	v	v	*(1)*	*(1)*	*(2)*	*(2)*	*(2)*	*(2)*

Notes to Table 4–2:

(1) This I/O standard is only supported on input clocks in this I/O bank.

(2) This I/O standard is only supported on output clocks in this I/O bank.

Figure 4–18. Stratix I/O Banks *Notes (1), (2), (3)*



Notes to Figure 4–18:

(1) Figure 4–18 is a top view of the silicon die. This corresponds to a top-down view for non-flip-chip packages, but is a reverse view for flip-chip packages.

(2) Figure 4–18 is a graphic representation only. See the pin list and the Quartus II software for exact locations.

(3) Banks 9 through 12 are enhanced PLL external clock output banks.

(4) If the high-speed differential I/O pins are not used for high-speed differential signaling, they can support all of the I/O standards except HSTL Class II, GTL, SSTL-18 Class II, PCI, PCI-X 1.0, and AGP 1×/2×.

(5) For guidelines on placing single-ended I/O pads next to differential I/O pads, see "I/O Pad Placement Guidelines" on page 4–30.

Tables 4–3 and 4–4 list the I/O standards that Stratix GX enhanced and fast PLL pins support. Figure 4–19 shows the I/O standards that each Stratix GX I/O bank supports.

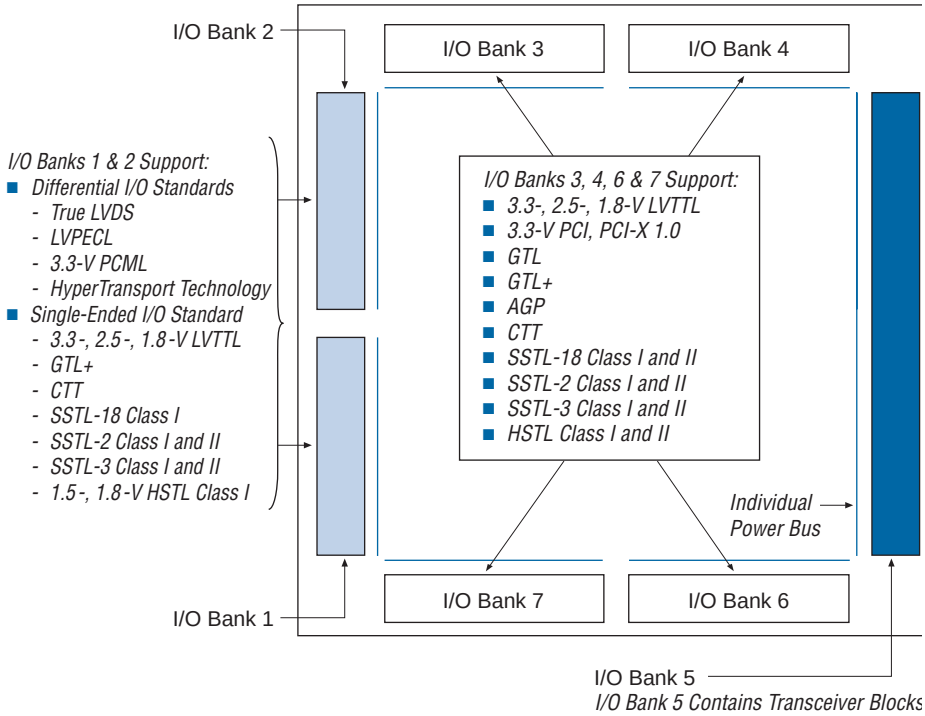
Table 4–3. I/O Standards Supported in Stratix & Stratix GX Enhanced PLL Pins

I/O Standard	Input			Output
	INCLK	FBIN	PLLENABLE	EXTCLK
LVTTL	v	v	v	v
LVCMOS	v	v	v	v
2.5 V	v	v		v
1.8 V	v	v		v
1.5 V	v	v		v
3.3-V PCI	v	v		v
3.3-V PCI-X 1.0	v	v		v
LVPECL	v	v		v
3.3-V PCML	v	v		v
LVDS	v	v		v
HyperTransport technology	v	v		v
Differential HSTL	v			v
Differential SSTL				v
3.3-V GTL	v	v		v
3.3-V GTL+	v	v		v
1.5-V HSTL Class I	v	v		v
1.5-V HSTL Class II	v	v		v
SSTL-18 Class I	v	v		v
SSTL-18 Class II	v	v		v
SSTL-2 Class I	v	v		v
SSTL-2 Class II	v	v		v
SSTL-3 Class I	v	v		v
SSTL-3 Class II	v	v		v
AGP (1× and 2×)	v	v		v
CTT	v	v		v

Table 4–4. I/O Standards Supported in Stratix & Stratix GX Fast PLL Pins

I/O Standard	Input	
	INCLK	PLLENABLE
LVTTL	v	v
LVCMOS	v	v
2.5 V	v	
1.8 V	v	
1.5 V	v	
3.3-V PCI		
3.3-V PCI-X 1.0		
LVPECL	v	
3.3-V PCML	v	
LVDS	v	
HyperTransport technology	v	
Differential HSTL	v	
Differential SSTL		
3.3-V GTL		
3.3-V GTL+		
1.5V HSTL Class I	v	
1.5V HSTL Class II		
SSTL-18 Class I	v	
SSTL-18 Class II		
SSTL-2 Class I	v	
SSTL-2 Class II	v	
SSTL-3 Class I	v	
SSTL-3 Class II	v	
AGP (1× and 2×)		
CTT	v	

Figure 4–19. Stratix GX I/O Banks



There is some flexibility with the number of I/O standards each Stratix I/O bank can simultaneously support. The following sections provide guidelines for mixing non-voltage-referenced and voltage-referenced I/O standards in Stratix devices.

Non-Voltage-Referenced Standards

Each Stratix I/O bank has its own VCCIO pins and supports only one V_{CCIO}, either 1.5, 1.8, 2.5 or 3.3 V. A Stratix I/O bank can simultaneously support any number of input signals with different I/O standard assignments, as shown in Table 4–5.

Table 4–5. Acceptable Input Levels for LVTTL/LVCMOS

Bank V_{CCIO}	Acceptable Input Levels			
	3.3 V	2.5 V	1.8 V	1.5 V
3.3 V	v	v		
2.5 V	v	v		
1.8 V	v *(2)*	v *(2)*	v	v *(1)*
1.5 V	v *(2)*	v *(2)*	v	v

Notes to Table 4–5:

(1) Because the input signal will not drive to the rail, the input buffer does not completely shut off, and the I/O current will be slightly higher than the default value.

(2) These input values overdrive the input buffer, so the pin leakage current will be slightly higher than the default value.

For output signals, a single I/O bank can only support non-voltage-referenced output signals driving at the same voltage as V_{CCIO}. A Stratix I/O bank can only have one V_{CCIO} value, so it can only drive out that one value for non-voltage referenced signals. For example, an I/O bank with a 2.5-V V_{CCIO} setting can support 2.5-V LVTTL inputs and outputs, HyperTransport technology inputs and outputs, and 3.3-V LVCMOS inputs (not output or bidirectional pins).

1 If the output buffer overdrives the input buffer, you must turn on the **Allow voltage overdrive for LVTTL/LVCMOS** option in the Quartus II software. To see this option, click the **Device & Pin Options** button in the **Device** page of the **Settings** dialog box (Assignments menu). Then click the **Pin Placement** tab in the **Device & Pin Options** dialog box.

Voltage-Referenced Standards

To accommodate voltage-referenced I/O standards, each Stratix I/O bank supports multiple VREF pins feeding a common VREF bus. The number of available VREF pins increases as device density increases. If these pins are not used as VREF pins, they can not be used as generic I/O pins.

An I/O bank featuring single-ended or differential standards can support voltage-referenced standards as long as all voltage-referenced standards use the same V_{REF} setting. For example, although one I/O bank can implement both SSTL-3 and SSTL-2 I/O standards, I/O pins using these standards must be in different banks since they require different V_{REF} values

For voltage-referenced inputs, the receiver compares the input voltage to the voltage reference and does not take into account the V_{CCIO} setting. Therefore, the V_{CCIO} setting is irrelevant for voltage referenced inputs.

Voltage-referenced bidirectional and output signals must be the same as the I/O bank's V_{CCIO} voltage. For example, although you can place an SSTL-2 input pin in any I/O bank with a 1.25-V V_{REF} level, you can only place SSTL-2 output pins in an I/O bank with a 2.5-V V_{CCIO}.

Mixing Voltage Referenced & Non-Voltage Referenced Standards

Non-voltage referenced and voltage referenced pins can safely be mixed in a bank by applying each of the rule-sets individually. For example, on I/O bank can support SSTL-3 inputs and 1.8-V LVCMOS inputs and outputs with a 1.8-V V_{CCIO} and a 1.5-V V_{REF}. Similarly, an I/O bank can support 1.5-V LVCMOS, 3.3-V LVTTL (inputs, but not outputs), and HSTL I/O standards with a 1.5-V V_{CCIO} and 0.75-V V_{REF}.

For the voltage-referenced examples, see the "I/O Pad Placement Guidelines" section. For details on how the Quartus II software supports I/O standards, see the "Quartus II Software Support"section.

Drive Strength

Each I/O standard supported by Stratix and Stratix GX devices drives out a minimum drive strength. When an I/O is configured as LVTTL or LVCMOS I/O standards, you can specify the current drive strength, as summarized in Table 4–7.

Standard Current Drive Strength

Each I/O standard supported by Stratix and Stratix GX devices drives out a minimum drive strength. Table 4–6 summarizes the minimum drive strength of each I/O standard.

Table 4–6. Minimum Current Drive Strength of Each I/O Standard

I/O Standard	Current Strength, I_{OL}/I_{OH} (mA)
GTL	40 *(1)*
GTL+	34 *(1)*
SSTL-3 Class I	8
SSTL-3 Class II	16
SSTL-2 Class I	8.1
SSTL-2 Class II	16.4
SSTL-18 Class I	6.7
SSTL-18 Class II	13.4
1.5-V HSTL Class I	8
1.5-V HSTL Class II	16
CTT	8
AGP 1×	I_{OL} = 1.5, I_{OH} = –0.5

Note to Table 4–6:

(1) Because this I/O standard uses an open drain buffer, this value refers to I_{OL}.

When the SSTL-2 Class I and II I/O standards are implemented on top or bottom I/O pins, the drive strength is designed to be higher than the drive strength of the buffer when implemented on side I/O pins. This allows the top or bottom I/O pins to support 200-MHz operation with the standard 35-pF load. At the same time, the current consumption when using top or bottom I/O pins is higher than the side I/O pins. The high current strength may not be necessary for certain applications where the value of the load is less than the standard test load (e.g., DDR interface). The Quartus II software allows you to reduce the drive strength when the I/O pins are used for the SSTL-2 Class I or Class II I/O standard and being implemented on the top or bottom I/O through the Current Strength setting. Select the minimum strength for lower drive strength.

Programmable Current Drive Strength

The Stratix and Stratix GX device I/O pins support various output current drive settings as shown in Table 4–7. These programmable drive strength settings help decrease the effects of simultaneously switching outputs (SSO) in conjunction with reducing system noise. The supported settings ensure that the device driver meets the I_{OH} and I_{OL} specifications for the corresponding I/O standard.

Table 4–7. Programmable Drive Strength

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA)
3.3-V LVTTL	24 *(1)*, 16, 12, 8, 4
3.3-V LVCMOS	24 *(2)*, 12 *(1)*, 8, 4, 2
2.5-V LVTTL/LVCMOS	16 *(1)*, 12, 8, 2
1.8-V LVTTL/LVCMOS	12 *(1)*, 8, 2
1.5-V LVCMOS	8 *(1)*, 4, 2

Notes to Table 4–7:

(1) This is the Quartus II software default current setting.

(2) I/O banks 1, 2, 5, and 6 do not support this setting.

These drive-strength settings are programmable on a per-pin basis (for output and bidirectional pins only) using the Quartus II software. To modify the current strength of a particular pin, see "Programmable Drive Strength Settings" on page 4–40.

Hot Socketing

Stratix devices support hot socketing without any external components. In a hot socketing situation, a device's output buffers are turned off during system power-up or power-down. Stratix and Stratix GX devices support any power-up or power-down sequence (V_{CCIO} and V_{CCINT}) to simplify designs. For mixed-voltage environments, you can drive signals into the device before or during power-up or power-down without damaging the device. Stratix and Stratix GX devices do not drive out until the device is configured and has attained proper operating conditions.

Even though you can power up or down the V_{CCIO} and V_{CCINT} power supplies in any sequence you should not power down any I/O bank(s) that contains the configuration pins while leaving other I/O banks powered on. For power up and power down, all supplies (V_{CCINT} and all V_{CCIO} power planes) must be powered up and down within 100 ms of one another. This prevents I/O pins from driving out.

You can power up or power down the V_{CCIO} and V_{CCINT} pins in any sequence. The power supply ramp rates can range from 100 ns to 100 ms. During hot socketing, the I/O pin capacitance is less than 15 pF and the clock pin capacitance is less than 20 pF.

DC Hot Socketing Specification

The hot socketing DC specification is $| I_{IOPIN} | < 300 \mu A$.

AC Hot Socketing Specification

The hot socketing AC specification is $| I_{IOPIN} | < 8$ mA for 10 ns or less.

This specification takes into account the pin capacitance, but not board trace and external loading capacitance. Additional capacitance for trace, connector, and loading must be considered separately.

I_{IOPIN} is the current at any user I/O pin on the device. The DC specification applies when all VCC supplies to the device are stable in the powered-up or powered-down conditions. For the AC specification, the peak current duration because of power-up transients is 10 ns or less. For more information, refer to the *Hot-Socketing & Power-Sequencing Feature & Testing for Altera Devices* white paper.

I/O Termination

Although single-ended, non-voltage-referenced I/O standards do not require termination, Altera recommends using external termination to improve signal integrity where required.

The following I/O standards do not require termination:

- LVTTL
- LVCMOS
- 2.5 V
- 1.8 V
- 1.5 V
- 3.3-V PCI/Compact PCI
- 3.3-V PCI-X 1.0
- 3.3-V AGP 1×

Voltage-Referenced I/O Standards

Voltage-referenced I/O standards require both an input reference voltage, $V_{REF,}$ and a termination voltage, V_{TT}. Off-chip termination on the board should be used for series and parallel termination.

For more information on termination for voltage-referenced I/O standards, see the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook, Volume 2*; or the *Stratix GX Device Handbook, Volume 2*.

Differential I/O Standards

Differential I/O standards typically require a termination resistor between the two signals at the receiver. The termination resistor must match the differential load impedance of the bus. Stratix and Stratix GX devices provide an optional differential termination on-chip resistor when using LVDS.

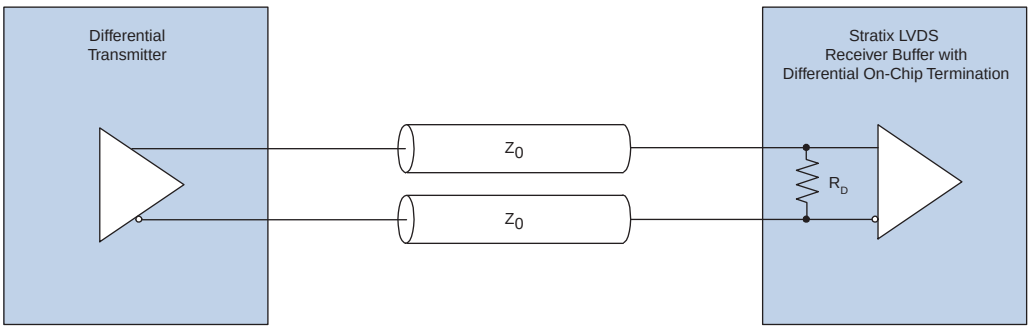
See the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter for more information on differential I/O standards and their interfaces.

For differential I/O standards, I/O banks support differential termination when V_{CCIO} equals 3.3 V.

Differential Termination (R_D)

Stratix devices support differential on-chip termination for source-synchronous LVDS signaling. The differential termination resistors are adjacent to the differential input buffers on the device. This placement eliminates stub effects, improving the signal integrity of the serial link. Using differential on-chip termination resistors also saves board space. Figure 4–20 shows the differential termination connections for Stratix and Stratix GX devices.

Figure 4–20. Differential Termination



Differential termination for Stratix devices is supported for the left and right I/O banks. Differential termination for Stratix GX devices is supported for the left, source-synchronous I/O bank. Some of the clock input pins are in the top and bottom I/O banks, which do not support differential termination. Clock pins CLK[1,3,8,10] support differential on-chip termination. Clock pins CLK[0,2,9,11], CLK[4-7], and CLK[12-15] do not support differential on-chip termination.

Transceiver Termination

Stratix GX devices feature built-in on-chip termination within the transceiver at both the transmit and receive buffers. This termination improves signal integrity and provides support for the 1.5-V PCML I/O standard.

I/O Pad Placement Guidelines

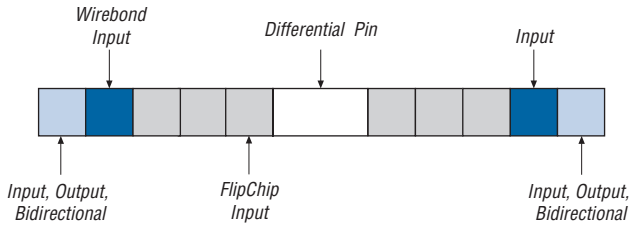
This section provides pad placement guidelines for the programmable I/O standards supported by Stratix and Stratix GX devices and includes essential information for designing systems using the devices' selectable I/O capabilities. These guidelines will reduce noise problems so that FPGA devices can maintain an acceptable noise level on the line from the V_{CCIO} supply. Since Altera FPGAs require that a separate V_{CCIO} power each bank, these noise issues do not have any effect when crossing bank boundaries and these guidelines do not apply. Although pad placement rules need not be considered between I/O banks, some rules must be considered if you are using a VREF signal in a PLLOUT bank. Note that the signals in the PLLOUT banks share the V_{REF} supply with neighboring I/O banks and, therefore, must adhere to the V_{REF} rules discussed in "VREF Pad Placement Guidelines".

Differential Pad Placement Guidelines

To avoid cross coupling and maintain an acceptable noise level on the V_{CCIO} supply, there are restrictions on the placement of single-ended I/O pads in relation to differential pads. Use the following guidelines for placing single-ended pads with respect to differential pads in Stratix devices. These guidelines apply for LVDS, HyperTransport technology, LVPECL, and PCML I/O standards. The differential pad placement guidelines do not apply for differential HSTL and differential SSTL output clocks since each differential output clock is essentially implemented using two single-ended output buffers. These rules do not apply to differential HSTL input clocks either even though the dedicated input buffers are used. However, both differential HSTL and differential SSTL output standards must adhere to the single-ended (VREF) pad placement restrictions discussed in "VREF Pad Placement Guidelines".

- For flip-chip packages, there are no restrictions for placement of single-ended input signals with respect to differential signals (see Figure 4–21). For wire-bond packages, single ended input pads may only be placed four or more pads away from a differential pad.
- Single-ended outputs and bidirectional pads may only be placed five or more pads away from a differential pad (see Figure 4–21), regardless of package type.

Figure 4–21. Legal Pin Placement *Note (1)*



Note to Figure 4–21:
(1) Input pads on a flip-chip packages have no restrictions.

VREF Pad Placement Guidelines

Restrictions on the placement of single-ended voltage-referenced I/O pads with respect to VREF pads help maintain an acceptable noise level on the V_{CCIO} supply and to prevent output switching noise from shifting the VREF rail. The following guidelines are for placing single-ended pads in Stratix devices.

Input Pins

Each VREF pad supports a maximum of 40 input pads with up to 20 on each side of the VREF pad.

Output Pins

When a voltage referenced input or bidirectional pad does not exist in a bank, there is no limit to the number of output pads that can be implemented in that bank. When a voltage referenced input exists, each VREF pad supports 20 outputs for thermally enhanced FineLine BGA® and thermally enhanced BGA cavity up packages or 15 outputs for Non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA packages.

Bidirectional Pins

Bidirectional pads must satisfy input and output guidelines simultaneously. If the bidirectional pads are all controlled by the same OE and there are no other outputs or voltage referenced inputs in the bank, then there is no case where there is a voltage referenced input active at the same time as an output. Therefore, the output limitation does not apply. However, since the bidirectional pads are linked to the same OE, the bidirectional pads act as inputs at the same time. Therefore, the input limitation of 40 input pads (20 on each side of the VREF pad) applies.

If any of the bidirectional pads are controlled by different output enables (OE) and there are no other outputs or voltage referenced inputs in the bank, then there may be a case where one group of bidirectional pads is acting as inputs while another group is acting as outputs. In such cases, apply the formulas shown in Table 4–8.

Table 4–8. Input-Only Bidirectional Pin Limitation Formulas

Package Type	Formula
Thermally enhanced FineLine BGA and thermally enhanced BGA cavity up	*<Total number of bidirectional pads> – <Total number of pads from the smallest group of pads controlled by an OE>* ≤20 (per VREF pad)
Non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA	*<Total number of bidirectional pads> – <Total number of pads from the smallest group of pads controlled by an OE>* ≤15 (per VREF pad).

Consider a thermally enhanced FineLine BGA package with eight bidirectional pads controlled by OE1, eight bidirectional pads controlled by OE2, and six bidirectional pads controlled by OE3. While this totals 22 bidirectional pads, it is safely allowable because there would be a maximum of 16 outputs per VREF pad possible assuming the worst case where OE1 and OE2 are active and OE3 is inactive. This is particularly relevant in DDR SDRAM applications.

When at least one additional voltage referenced input and no other outputs exist in the same VREF bank, then the bidirectional pad limitation must simultaneously adhere to the input and output limitations. See the following equation.

<Total number of bidirectional pads> + *<Total number of input pads>* ≤40 (20 on each side of the VREF pad)

The previous equation accounts for the input limitations, but you must apply the appropriate equation from Table 4–9 to determine the output limitations.

Table 4–9. Bidirectional pad Limitation Formulas (Where VREF Inputs Exist)

Package Type	Formula
Thermally enhanced FineLine BGA and thermally enhanced BGA cavity up	<*Total number of bidirectional pads*> ≤20 (per VREF pad)
Non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA	<*Total number of bidirectional pads*> ≤15 (per VREF pad)

When at least one additional output exists but no voltage referenced inputs exist, apply the appropriate formula from Table 4–10.

Table 4–10. Bidirectional Pad Limitation Formulas (Where VREF Outputs Exist)

Package Type	Formula
Thermally enhanced FineLine BGA and thermally enhanced BGA cavity up	<*Total number of bidirectional pads*> + <*Total number of additional output pads*> – <*Total number of pads from the smallest group of pads controlled by an OE*> ≤20 (per VREF pad)
Non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA	<*Total number of bidirectional pads*> + <*Total number of additional output pads*> – <*Total number of pads from the smallest group of pads controlled by an OE*> ≤15 (per VREF pad)

When additional voltage referenced inputs and other outputs exist in the same VREF bank, then the bidirectional pad limitation must again simultaneously adhere to the input and output limitations. See the following equation.

<*Total number of bidirectional pads*> + <*Total number of input pads*> ≤40 (20 on each side of the VREF pad)

The previous equation accounts for the input limitations, but you must apply the appropriate equation from Table 4–9 to determine the output limitations.

Table 4–11. Bidirectional Pad Limitation Formulas (Multiple VREF Inputs & Outputs)

Package Type	Formula
Thermally enhanced FineLine BGA and thermally enhanced BGA cavity up	*<Total number of bidirectional pads>* + *<Total number of additional output pads>* ≤20 (per VREF pad)
non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA	*<Total number of bidirectional pads>* + *<Total number of additional output pads>* ≤15 (per VREF pad)

In addition to the pad placement guidelines, use the following guidelines when working with V_{REF} standards:

- Each bank can only have a single V_{CCIO} voltage level and a single V_{REF} voltage level at a given time. Pins of different I/O standards can share the bank if they have compatible V_{CCIO} values (see Table 4–12 for more details).
- In all cases listed above, the Quartus II software generates an error message for illegally placed pads.

Output Enable Group Logic Option in Quartus II

The Quartus II software can check a design to make sure that the pad placement does not violate the rules mentioned above. When the software checks the design, if the design contains more bidirectional pins than what is allowed, the Quartus II software returns a fitting error. When all the bidirectional pins are either input or output but not both (for example, in a DDR memory interface), you can use the **Output Enable Group Logic** option. Turning on this option directs the Quartus II Fitter to view the specified nodes as an output enable group. This way, the Fitter does not violate the requirements for the maximum number of pins driving out of a V_{REF} bank when a voltaged-referenced input pin or bidirectional pin is present.

In a design that implements DDR memory interface with dq, dqs and dm pins utilized, there are two ways to enable the above logic options. You can enable the logic options through the Assignment Editor or by adding the following assignments to your project's ESF file:

```
OPTIONS_FOR_INDIVIDUAL_NODES_ONLY
{
    dq  : OUTPUT_ENABLE_GROUP 1;
    dqs : OUTPUT_ENABLE_GROUP 1;
```

```
    dm  : OUTPUT_ENABLE_GROUP 1;
}
```

As a result, the Quartus II Fitter does not count the bidirectional pin potential outputs, and the number of V_{REF} bank outputs remains in the legal range.

Toggle Rate Logic Option in Quartus II

You should specify the pin's output toggling rate in order to perform a stricter pad placement check in the Quartus II software. Specify the frequency at which a pin toggles in the Quartus II Assignment Editor. This option is useful for adjusting the pin toggle rate in order to place them closer to differential pins. The option directs the Quartus II Fitter toggle-rate checking while allowing you to place a single-ended pin closer to a differential pin.

DC Guidelines

Variables affecting the DC current draw include package type and desired termination methods. This section provides information on each of these variables and also shows how to calculate the DC current for pin placement.

1 The Quartus II software automatically takes these variables into account during compilation.

For any 10 consecutive output pads in an I/O bank, Altera recommends a maximum current of 200 mA for thermally enhanced FineLine BGA and thermally enhanced BGA cavity up packages and 164 mA for non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA packages. The following equation shows the current density limitation equation for thermally enhanced FineLine BGA and thermally enhanced BGA cavity up packages:

$$\sum_{pin}^{pin + 9} I_{pin} < 200 \text{ mA}$$

The following equation shows the current density limitation equation for non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA packages:

$$\sum_{pin}^{pin+9} I_{pin} < 164\ mA$$

Table 4–12 shows the DC current specification per pin for each I/O standard. I/O standards not shown in the table do not exceed these current limitations.

Table 4–12. I/O Standard DC Specification *Note (1)*

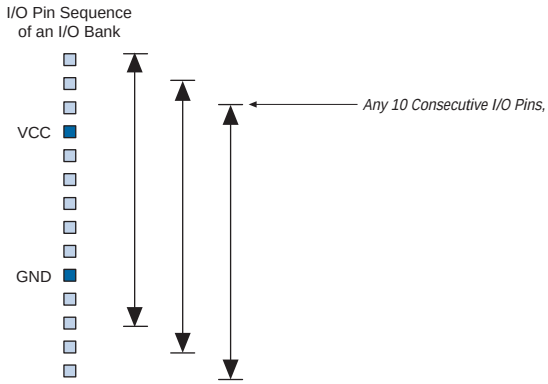
Pin I/O Standard	I_{PIN} (mA)		
	3.3-V V_{CCIO}	2.5-V V_{CCIO}	1.5-V V_{CCIO}
GTL	40	40	-
GTL+	34	34	-
SSTL-3 Class I	8	-	-
SSTL-3 Class II	16	-	-
CTT	8	-	-
SSTL-2 Class I	-	8.1	-
SSTL-2 Class II	-	16.4	-
HSTL Class I	-	-	8
HSTL Class II	-	-	16

Note to Table 4–12:

(1) The current rating on a V_{REF} pin is less than 10µA.

f For more information on Altera device packaging, see the *Package Information for Stratix Devices* chapter in the *Stratix Device Handbook, Volume 2*.

Figure 4–22. Current Draw Limitation Guidelines



Any 10 consecutive I/O pads cannot exceed 200 mA in thermally enhanced FineLine BGA and thermally enhanced BGA cavity up packages or 164 mA in non-thermally enhanced cavity up and non-thermally enhanced FineLine BGA packages.

For example, consider a case where a group of 10 consecutive pads are configured as follows for a thermally enhanced FineLine BGA and thermally enhanced BGA cavity up package:

- Number of SSTL-3 Class I output pads = 3
- Number of GTL+ output pads = 4
- The rest of the surrounding I/O pads in the consecutive group of 10 are unused

In this case, the total current draw for these 10 consecutive I/O pads would be:

(# of SSTL-3 Class I pads × 8 mA) +
(# of GTL+ output pads × 34 mA) = (3 × 8 mA) + (4 × 34 mA) = 160 mA

In the above example, the total current draw for all 10 consecutive I/O pads is less than 200 mA.

Power Source of Various I/O Standards

For Stratix and Stratix GX devices, the I/O standards are powered by different power sources. To determine which source powers the input buffers, see Table 4–13. All output buffers are powered by V_{CCIO}.

Table 4–13. The Relationships Between Various I/O Standards and the Power Sources

I/O Standard	Power Source
2.5V/3.3V LVTTL	V_{CCIO}
PCI/PCI-X 1.0	V_{CCIO}
AGP	V_{CCIO}
1.5V/1.8V	V_{CCIO}
GTL	V_{CCINT}
GTL+	V_{CCINT}
SSTL	V_{CCINT}
HSTL	V_{CCINT}
CTT	V_{CCINT}
LVDS	V_{CCINT}
LVPECL	V_{CCINT}
PCML	V_{CCINT}
HyperTransport	V_{CCINT}

Quartus II Software Support

You specify which programmable I/O standards to use for Stratix and Stratix GX devices with the Quartus II software. This section describes Quartus II implementation, placement, and assignment guidelines, including

- Compiler Settings
- Device & Pin Options
- Assign Pins
- Programmable Drive Strength Settings
- I/O Banks in the Floorplan View
- Auto Placement & Verification

Compiler Settings

You make Compiler settings in the **Compiler Settings** dialog box (Processing menu). Click the **Chips & Devices** tab to specify the device family, specific device, package, pin count, and speed grade to use for your design.

Device & Pin Options

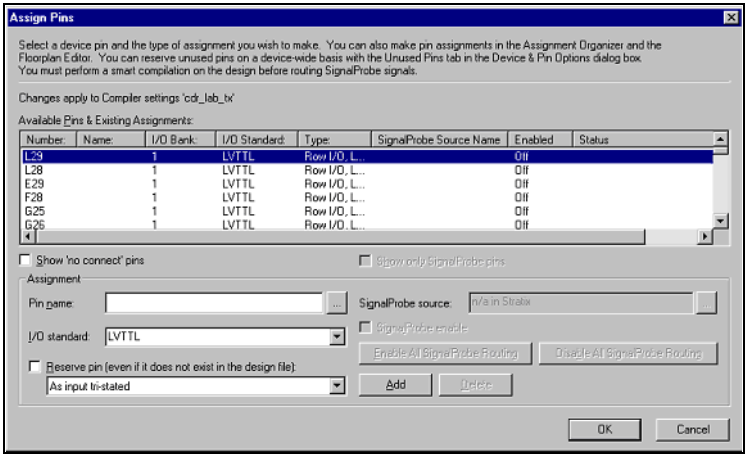
Click **Device & Pin Options** in the **Compiler Settings** dialog box to access the I/O pin settings. For example, in the **Voltage** tab you can select a default I/O standard for all pins for the targeted device. I/O pins that do not have a specific I/O standard assignment default this standard. Click **OK** when you are done setting I/O pin options to return to the **Compiler Settings** dialog box.

Assign Pins

Click **Assign Pins** in the **Compiler Settings** dialog box to view the device's pin settings and pin assignments (see Figure 4–23). You can view the pin settings under **Available Pins & Existing Assignments**. The listing does not include V_{REF} pins because they are dedicated pins. The information for each pin includes:

- Number
- Name
- I/O Bank
- I/O Standard
- Type (e.g., row or column I/O and differential or control)
- SignalProbe Source Name
- Enabled (that is, whether SignalProbe routing is enabled or disabled
- Status

Figure 4–23. Assign Pins



When you assign an I/O standard that requires a reference voltage to an I/O pin, the Quartus II software automatically assigns V_{REF} pins. See the Quartus II Help for instructions on how to use an I/O standard for a pin.

Programmable Drive Strength Settings

To make programmable drive strength settings, perform the following steps:

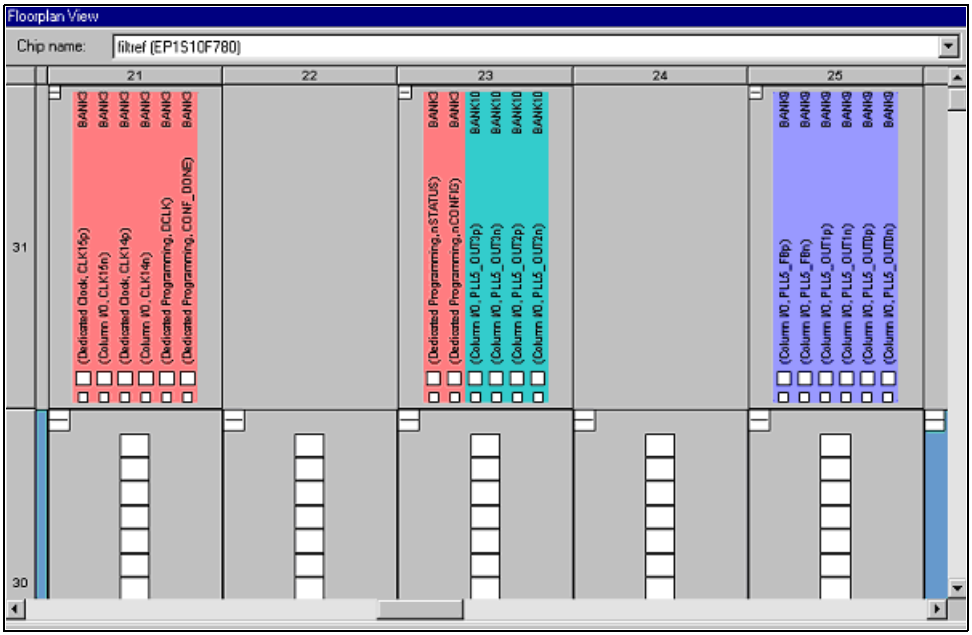
1. In the Tools menu, choose **Assignment Organizer**.
2. Choose the **Edit specific entity & node settings for:** setting, then select the output or bidirectional pin to specify the current strength for.
3. In the **Assignment Categories** dialog box, select **Options for Individual Nodes Only**.
4. Select **Click here to add a new assignment**.
5. In the **Assignment** dialog box, set the **Name** field to **Current Strength** and set the **Setting** field to the desired, allowable value.
6. Click **Add**.
7. Click **Apply**, then **OK**.

I/O Banks in the Floorplan View

You can view the arrangement of the device I/O banks in the **Floorplan View** (View menu) as shown in Figure 4–24. You can assign multiple I/O standards to the I/O pins in any given I/O bank as long as the V_{CCIO} of the standards is the same. Pins that belong to the same I/O bank must use the same V_{CCIO} signal.

Each device I/O pin belongs to a specific, numbered I/O bank. The Quartus II software color codes the I/O bank to which each I/O pin and V_{CCIO} pin belong. Turn on the **Show I/O Banks** option to display the I/O bank color and the bank numbers for each pin.

Figure 4–24. Floorplan View Window



Auto Placement & Verification of Selectable I/O Standards

The Quartus II software automatically verifies the placement for all I/O and V_{REF} pins and performs the following actions.

- Automatically places I/O pins of different V_{REF} standards without pin assignments in separate I/O banks and enables the V_{REF} pins of these I/O banks.
- Verifies that voltage-referenced I/O pins requiring different V_{REF} levels are not placed in the same bank.
- Reports an error message if the current limit is exceeded for a Stratix or Stratix GX power bank, as determined by the equation documented in "DC Guidelines" on page 4–35.
- Reserves the unused high-speed differential I/O channels and regular user I/O pins in the high-speed differential I/O banks when any of the high-speed differential I/O channels are being used.
- Automatically assigns V_{REF} pins and I/O pins such that the current requirements are met and I/O standards are placed properly.

Conclusion

Stratix and Stratix GX devices provide the I/O capabilities to allow you to work with current and emerging I/O standards and requirements. Today's complex designs demand increased flexibility to work with the wide variety of available I/O standards and to simplify board design. With Stratix and Stratix GX device features, such as hot socketing and differential on-chip termination, you can reduce board design interface costs and increase your development flexibility.

More Information

For more information, see the following sources:

- The *Stratix Device Family Data Sheet* section in the *Stratix Device Handbook, Volume 1*
- The *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*
- The *High-Speed Differential I/O Interfaces in Stratix Devices* chapter
- *AN 224: High-Speed Board Layout Guidelines*

References

For more information, see the following references:

- Stub Series Terminated Logic for 2.5-V (SSTL-2), JESD8-9B, Electronic Industries Association, December 2000.
- High-Speed Transceiver Logic (HSTL) – A 1.5-V Output Buffer Supply Voltage Based Interface Standard for Digital Integrated Circuits, EIA/JESD8-6, Electronic Industries Association, August 1995.
- 1.5-V +/- 0.1 V (Normal Range) and 0.9 V – 1.6 V (Wide Range) Power Supply Voltage and Interface Standard for Non-terminated Digital Integrated Circuits, JESD8-11, Electronic Industries Association, October 2000.
- 1.8-V +/- 0.15 V (Normal Range) and 1.2 V – 1.95 V (Wide Range) Power Supply Voltage and Interface Standard for Non-terminated Digital Integrated Circuits, JESD8-7, Electronic Industries Association, February 1997.
- Center-Tap-Terminated (CTT) Low-Level, High-Speed Interface Standard for Digital Integrated Circuits, JESD8-9A, Electronic Industries Association, November 1993.
- 2.5-V +/- 0.2V (Normal Range) and 1.8-V to 2.7V (Wide Range) Power Supply Voltage and Interface Standard for Non-terminated Digital Integrated Circuits, JESD8-5, Electronic Industries Association, October 1995.
- Interface Standard for Nominal 3V/ 3.3-V Supply Digital Integrated Circuits, JESD8-B, Electronic Industries Association, September 1999.
- Gunning Transceiver Logic (GTL) Low-Level, High-Speed Interface Standard for Digital Integrated Circuits, JESD8-3, Electronic Industries Association, November 1993.

- Accelerated Graphics Port Interface Specification 2.0, Intel Corporation.
- Stub Series Terminated Logic for 1.8-V (SSTL-18), Preliminary JC42.3, Electronic Industries Association.
- PCI Local Bus Specification, Revision 2.2, PCI Special Interest Group, December 1998.
- PCI-X Local Bus Specification, Revision 1.0a, PCI Special Interest Group.
- UTOPIA Level 4, AF-PHY-0144.001, ATM Technical Committee.
- POS-PHY Level 4: SPI-4, OIF-SPI4-02.0, Optical Internetworking Forum.
- POS-PHY Level 4: SFI-4, OIF-SFI4-01.0, Optical Internetworking Forum.
- Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits, ANSI / TIA / EIA-644, American National Standards Institute / Telecommunications Industry / Electronic Industries Association, October 1995.

5. High-Speed Differential I/O Interfaces in Stratix Devices

S52005-3.2

Introduction

To achieve high data transfer rates, Stratix® devices support True-LVDS™ differential I/O interfaces which have dedicated serializer/deserializer (SERDES) circuitry for each differential I/O pair. Stratix SERDES circuitry transmits and receives up to 840 megabits per second (Mbps) per channel. The differential I/O interfaces in Stratix devices support many high-speed I/O standards, such as LVDS, LVPECL, PCML, and HyperTransport™ technology. Stratix device high-speed modules are designed to provide solutions for many leading protocols such as SPI-4 Phase 2, SFI-4, 10G Ethernet XSBI, RapidIO, HyperTransport technology, and UTOPIA-4.

The SERDES transmitter is designed to serialize 4-, 7-, 8-, or 10-bit wide words and transmit them across either a cable or printed circuit board (PCB). The SERDES receiver takes the serialized data and reconstructs the bits into a 4-, 7-, 8-, or 10-bit-wide parallel word. The SERDES contains the necessary high-frequency circuitry, multiplexer, demultiplexer, clock, and data manipulation circuitry. You can use double data rate I/O (DDRIO) circuitry to transmit or receive differential data in by-one (×1) or by-two (×2) modes.

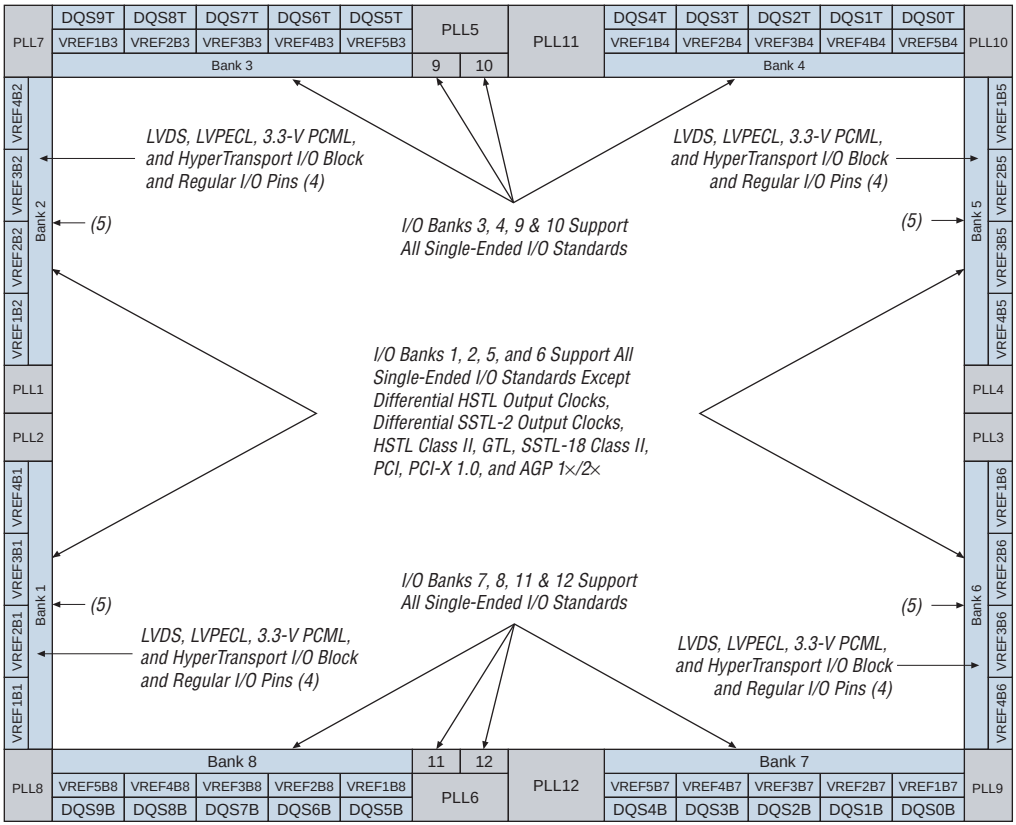
1 Contact Altera Applications for more information on other *B* values that the Stratix devices support and using ×7-mode in the Quartus® II software. Stratix devices currently only support *B* = 1 and B = 7 in ×7 mode.

This chapter describes the high-speed differential I/O capabilities of Stratix programmable logic devices (PLDs) and provides guidelines for their optimal use. You should use this document in conjunction with the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1*. Consideration of the critical issues of controlled impedance of traces and connectors, differential routing, termination techniques, and DC balance gets the best performance from the device. Therefore, an elementary knowledge of high-speed clock-forwarding techniques is also helpful.

Stratix I/O Banks

Stratix devices contain eight I/O banks, as shown in Figure 5–1. The two I/O banks on each side contain circuitry to support high-speed LVDS, LVPECL, PCML, HSTL Class I and II, SSTL-2 Class I and II, and HyperTransport inputs and outputs.

Figure 5–1. Stratix I/O Banks *Notes (1), (2), (3)*



Notes to Figure 5–1:

(1) Figure 5–1 is a top view of the Stratix silicon die, which corresponds to a top-down view of non-flip-chip packages and a bottom-up view of flip-chip packages.

(2) Figure 5–1 is a graphic representation only. See the pin list and the Quartus II software for exact locations.

(3) Banks 9 through 12 are enhanced PLL external clock output banks.

(4) If the high-speed differential I/O pins are not used for high-speed differential signaling, they can support all of the I/O standards except HSTL Class I and II, GTL, SSTL-18 Class II, PCI, PCI-X 1.0, and AGP 1× /2×.

(5) See "Differential Pad Placement Guidelines" on page 4–30. You can only place single-ended output/bidirectional pads five or more pads away from a differential pad. Use the **Show Pads** view in the Quartus II Floorplan Editor to locate these pads. The Quartus II software gives an error message for illegal output or bidirectional pin placement next to a high-speed differential I/O pin.

Stratix Differential I/O Standards

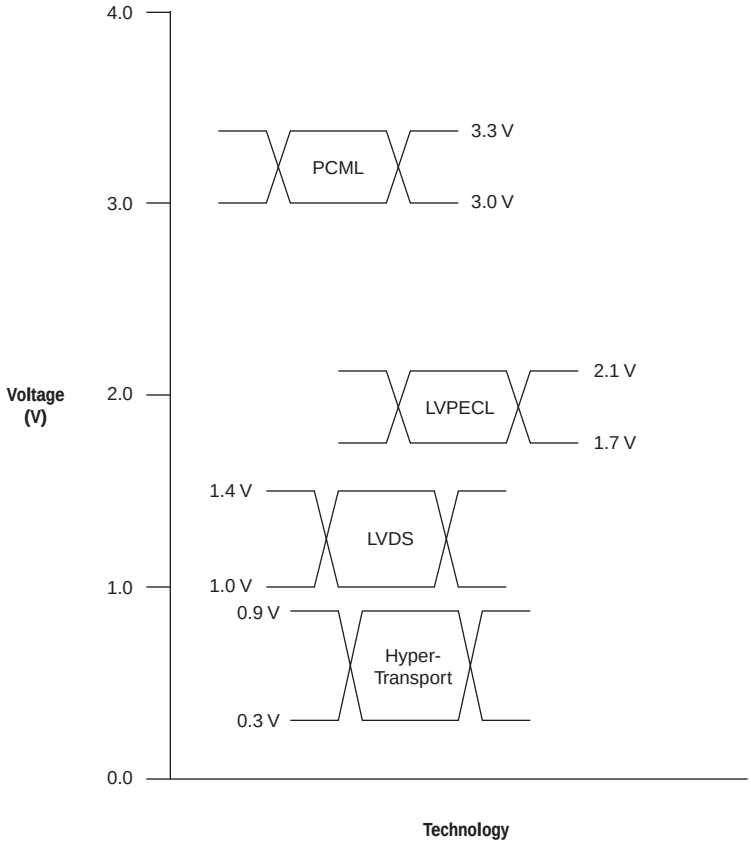
Stratix devices provide a multi-protocol interface that allows communication between a variety of I/O standards, including LVDS, HyperTransport technology, LVPECL, PCML, HSTL Class I and II, and

SSTL-2 Class I and II. This feature makes the Stratix device family ideal for applications that require multiple I/O standards, such as a protocol translator.

f For more information on termination for Stratix I/O standards, see "Differential I/O Termination" on page 5–46.

Figure 5–2 compares the voltage levels between differential I/O standards supported in all the Stratix devices.

Figure 5–2. Differential I/O Standards Supported by Stratix Devices



LVDS

The LVDS I/O standard is a differential high-speed, low-voltage swing, low-power, general-purpose I/O interface standard requiring a 3.3-V V_{CCIO}. This standard is used in applications requiring high-bandwidth data transfer, backplane drivers, and clock distribution. The ANSI/TIA/EIA-644 standard specifies LVDS transmitters and receivers capable of operating at recommended maximum data signaling rates of 655 Mbps. However, devices can operate at slower speeds if needed, and there is a theoretical maximum of 1.923 Gbps. Stratix devices meet the ANSI/TIA/EIA-644 standard.

Due to the low voltage swing of the LVDS I/O standard, the electromagnetic interference (EMI) effects are much smaller than CMOS, transistor-to-transistor logic (TTL), and PECL. This low EMI makes LVDS ideal for applications with low EMI requirements or noise immunity requirements. The LVDS standard specifies a differential output voltage range of 0.25 V × V_{OD} ≤0.45 V. The LVDS standard does not require an input reference voltage, however, it does require a 100-Ω termination resistor between the two signals at the input buffer. Stratix devices include an optional differential termination resistor within the device. See Section I, Stratix Device Family Data Sheet of the *Stratix Device Handbook, Volume 1* for the LVDS parameters.

HyperTransport Technology

The HyperTransport technology I/O standard is a differential high-speed, high-performance I/O interface standard requiring a 2.5-V VCCIO. This standard is used in applications such as high-performance networking, telecommunications, embedded systems, consumer electronics, and Internet connectivity devices. The HyperTransport technology I/O standard is a point-to-point standard in which each HyperTransport technology bus consists of two point-to-point unidirectional links. Each link is 2 to 32 bits. See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* for the HyperTransport parameters.

LVPECL

The LVPECL I/O standard is a differential interface standard requiring a 3.3-V $V_{CCIO.}$ The standard is used in applications involving video graphics, telecommunications, data communications, and clock distribution. The high-speed, low-voltage swing LVPECL I/O standard uses a positive power supply and is similar to LVDS, however, LVPECL has a larger differential output voltage swing than LVDS. See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* for the LVPECL signaling characteristics.

PCML

The PCML I/O standard is a differential high-speed, low-power I/O interface standard used in applications such as networking and telecommunications. The standard requires a 3.3-V V_{CCIO}. The PCML I/O standard achieves better performance and consumes less power than the LVPECL I/O standard. The PCML standard is similar to LVPECL, but PCML has a reduced voltage swing, which allows for a faster switching time and lower power consumption.See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* for the PCML signaling characteristics.

Differential HSTL (Class I & II)

The differential HSTL I/O standard is used for applications designed to operate in the 0.0- to 1.5-V HSTL logic switching range such as quad data rate (QDR) memory clock interfaces. The differential HSTL specification is the same as the single ended HSTL specification. The standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. The differential HSTL I/O standard is only available on the input and output clocks. See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* for the HSTL signaling characteristics

Differential SSTL-2 (Class I & II)

The differential SSTL-2 I/O standard is a 2.5-V memory bus standard used for applications such as high-speed double data rate (DDR) SDRAM interfaces. This standard defines the input and output specifications for devices that operate in the SSTL-2 logic switching range of 0.0 to 2.5 V. This standard improves operation in conditions where a bus must be isolated from large stubs. The SSTL-2 standard specifies an input voltage range of $-0.3\ V \leq V_I \leq V_{CCIO} + 0.3\ V$. Stratix devices support both input and output levels. The differential SSTL-2 I/O standard is only available on output clocks. See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* for the SSTL-2 signaling characteristics.

Stratix Differential I/O Pin Location

The differential I/O pins are located on the I/O banks on the right and left side of the Stratix device. Table 5–1 shows the location of the Stratix device high-speed differential I/O buffers. When the I/O pins in the I/O banks that support differential I/O standards are not used for high-speed

signaling, you can configure them as any of the other supported I/O standards. DDRIO capabilities are detailed in "SERDES Bypass DDR Differential Signaling" on page 5–42.

Table 5–1. I/O Pin Locations on Each Side of Stratix Devices

Device Side *(1)*	Differential Input	Differential Output	DDRIO
Left	v	v	v
Right	v	v	v
Top			v
Bottom			v

Note to Table 5–1:

(1) Device sides are relative to pin A1 in the upper left corner of the device (top view of the package).

Principles of SERDES Operation

Stratix devices support source-synchronous differential signaling up to 840 Mbps. Serial data is transmitted and received along with a low-frequency clock. The PLL can multiply the incoming low-frequency clock by a factor of 1 to 10. The SERDES factor *J* can be 4, 7, 8, or 10 and does not have to equal the clock multiplication value. ×1 and ×2 operation is also possible by bypassing the SERDES; it is explained in "SERDES Bypass DDR Differential Interface Review" on page 5–42.

On the receiver side, the high-frequency clock generated by the PLL shifts the serial data through a shift register (also called deserializer). The parallel data is clocked out to the logic array synchronized with the low-frequency clock. On the transmitter side, the parallel data from the logic array is first clocked into a parallel-in, serial-out shift register synchronized with the low-frequency clock and then transmitted out by the output buffers.

There are four dedicated fast PLLs in EP1S10 to EP1S25 devices, and eight in EP1S30 to EP1S80 devices. These PLLs are used for the SERDES operations as well as general-purpose use.

The differential channels and the high-speed PLL layout in Stratix devices are described in the "Differential I/O Interface & Fast PLLs" section on page 5–16.

Stratix Differential I/O Receiver Operation

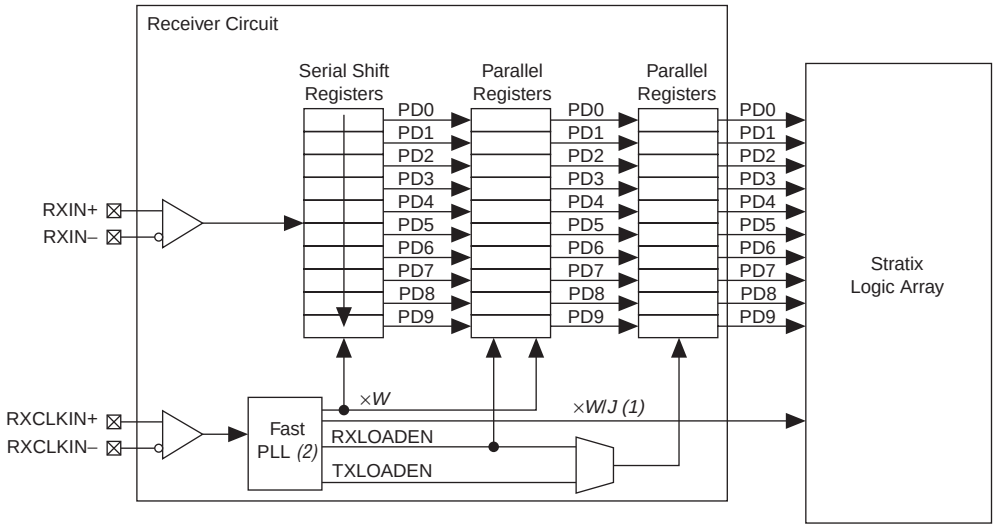
You can configure any of the Stratix differential input channels as a receiver channel (see Figure 5–3). The differential receiver deserializes the incoming high-speed data. The input shift register continuously clocks the incoming data on the negative transition of the high-frequency clock generated by the PLL clock (×*W*).

The data in the serial shift register is shifted into a parallel register by the RXLOADEN signal generated by the fast PLL counter circuitry on the third falling edge of the high-frequency clock. However, you can select which falling edge of the high frequency clock loads the data into the parallel register, using the data-realignment circuit. For more information on the data-realignment circuit, see "Data Realignment Principles of Operation" on page 5–25.

In normal mode, the enable signal RXLOADEN loads the parallel data into the next parallel register on the second rising edge of the low-frequency clock. You can also load data to the parallel register through the TXLOADEN signal when using the data-realignment circuit.

Figure 5–3 shows the block diagram of a single SERDES receiver channel. Figure 5–4 shows the timing relationship between the data and clocks in Stratix devices in ×10 mode. *W* is the low-frequency multiplier and *J* is data parallelization division factor.

Figure 5–3. Stratix High-Speed Interface Deserialized in ×10 Mode

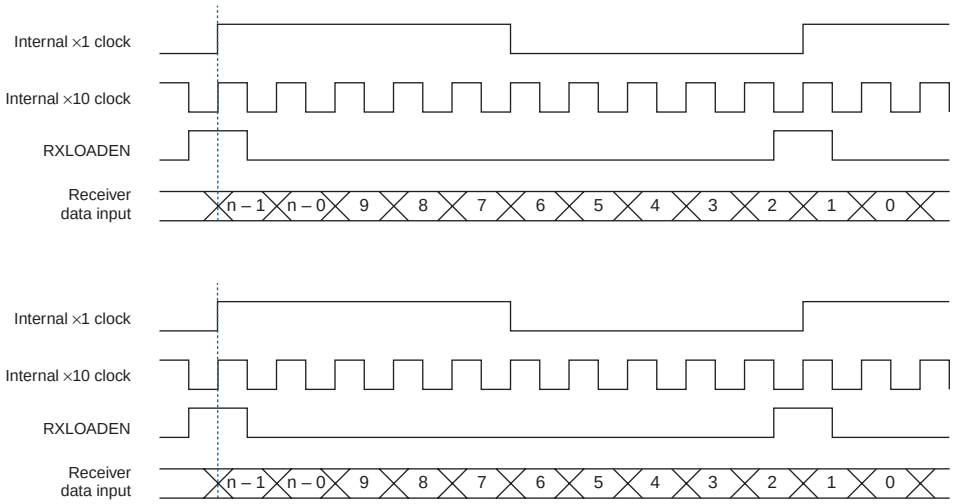


Notes to Figure 5–3:

(1) W = 1, 2, 4, 7, 8, or 10.
J = 4, 7, 8, or 10.
W does not have to equal J. When J = 1 or 2, the deserializer is bypassed. When J = 2, the device uses DDRIO registers.

(2) This figure does not show additional circuitry for clock or data manipulation.

Figure 5–4. Receiver Timing Diagram



Stratix Differential I/O Transmitter Operation

You can configure any of the Stratix differential output channels as a transmitter channel. The differential transmitter is used to serialize outbound parallel data.

The logic array sends parallel data to the SERDES transmitter circuit when the TXLOADEN signal is asserted. This signal is generated by the high-speed counter circuitry of the logic array low-frequency clock's rising edge. The data is then transferred from the parallel register into the serial shift register by the TXLOADEN signal on the third rising edge of the high-frequency clock.

Figure 5–5 shows the block diagram of a single SERDES transmitter channel and Figure 5–6 shows the timing relationship between the data and clocks in Stratix devices in ×10 mode. *W* is the low-frequency multiplier and *J* is the data parallelization division factor.

Figure 5–5. Stratix High-Speed Interface Serialized in ×10 Mode

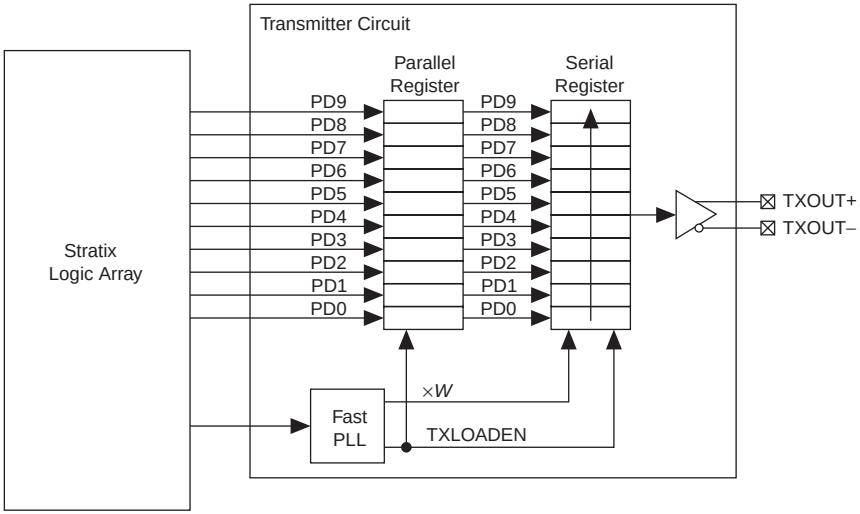
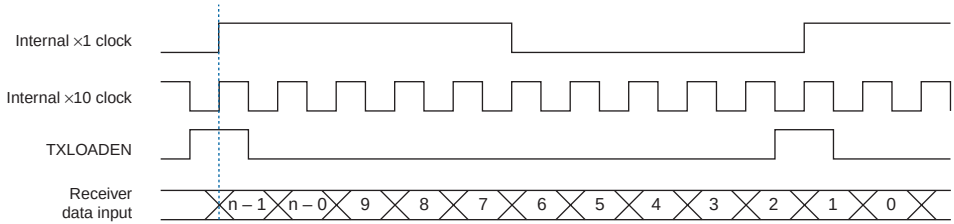


Figure 5–6. Transmitter Timing Diagram



Transmitter Clock Output

Different applications and protocols call for various clocking schemes. Some applications require you to center-align the rising or falling clock edge with the data. Other applications require a divide version of the transmitted clock, or the clock and data to be at the same high-speed frequency. The Stratix device transmitter clock output is versatile and easily programmed for all such applications.

Stratix devices transmit data using the source-synchronous scheme, where the clock is transmitted along with the serialized data to the receiving device. Unlike APEX™ 20KE and APEX II devices, Stratix devices do not have a fixed transmitter clock output pin. The Altera® Quartus II software generates the transmitter clock output by using a fast clock to drive a transmitter `dataout` channel. Therefore, you can place the transmitter clock pair close to the data channels, reducing clock-to-data skew and increasing system margins. This approach is more flexible, as any channel can drive a clock, not just specially designated clock pins.

Divided-Down Transmitter Clock Output

You can divide down the high-frequency clock by 2, 4, 8, or 10, depending on the system requirements. The various options allow Stratix devices to accommodate many different types of protocols. The divided-down clock is generated by an additional transmitting data channel.

Table 5–2 shows the divided-down version of the high-frequency clock and the selected serialization factor *J* (described in pervious sections). The Quartus II software automatically generates the data input to the additional transmitter data channel.

Table 5–2. Differential Transmitter Output Clock Division

J	Data Input	Output Clock Divided By *(1)*
4	1010	2
4	0011	4
8	10101010	2
8	00110011	4
8	11000011	8
10	1010101010	2
10	1110000011	10

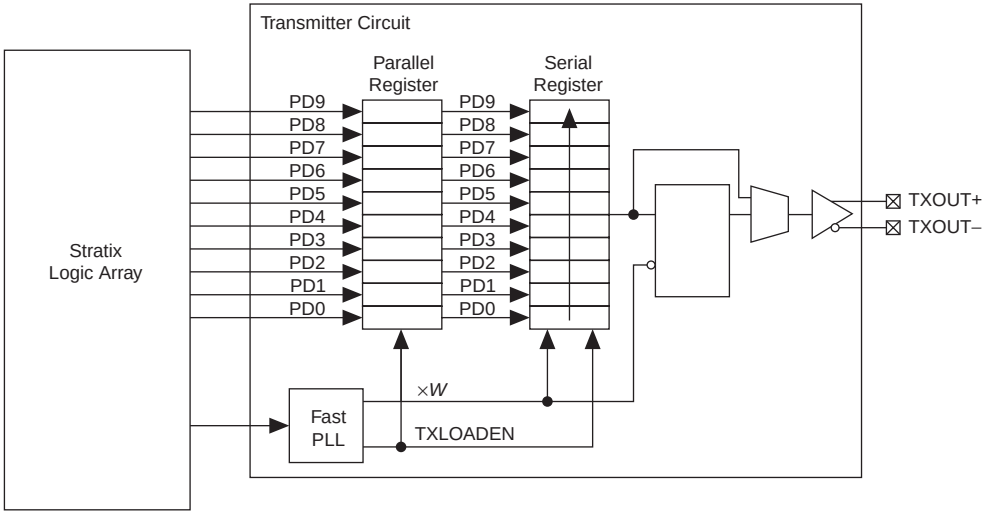
Note to Table 5–2:

(1) This value is usually referred to as *B*.

Center-Aligned Transmitter Clock Output

A negative-edge-triggered D flipflop (DFF) register is located between the serial register of each data channel and its output buffer, as show in Figure 5–7. The negative-edge-triggered DFF register is used when center-aligned data is required. For center alignment, the DFF only shifts the output from the channel used as the transmitter clock out. The transmitter data channels bypass the negative-edge DFF. When you use the DFF register, the data is transmitted at the negative edge of the multiplied clock. This delays the transmitted clock output relative to the data channels by half the multiplied clock cycle. This is used for HyperTransport technology, but can also be used for any interface requiring center alignment.

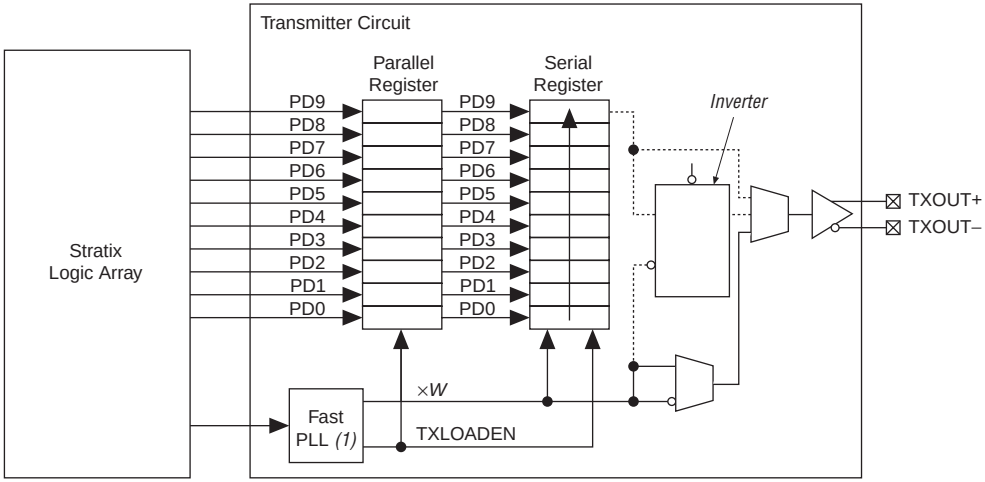
Figure 5–7. Stratix Programmable Transmitter Clock



SDR Transmitter Clock Output

You can route the high-frequency clock internally generated by the PLL out as a transmitter clock output on any of the differential channels. The high-frequency clock output allows Stratix devices to support applications that require a 1-to-1 relationship between the clock and data. The path of the high-speed clock is shown in Figure 5–8. A programmable inverter allows you to drive the signal out on either the negative edge of the clock or 180º out of phase with the streaming data.

Figure 5–8. High-Speed 1-to-1 Transmitter Clock Output



Note to Figure 5–8:

(1) This figure does not show additional circuitry for clock or data manipulation.

Using SERDES to Implement DDR

Some designs require a 2-to-1 data-to-clock ratio. These systems are usually based on Rapid I/O, SPI-4 Phase 2 (POS_PHY Level 4), or HyperTransport interfaces, and support various data rates. Stratix devices meet this requirement for such applications by providing a variable clock division factor. The SERDES clock division factor is set to 2 for double data rate (DDR).

An additional differential channel (as described in "Transmitter Clock Output" on page 5–10) is automatically configured to produce the transmitter clock output signal with half the frequency of the data.

For example, when a system is required to transmit 6.4 Gbps with a 2-to-1 clock-to-data ratio, program the SERDES with eight high-speed channels running at 800 Mbps each. When you set the output clock division factor (2 for this example), the Quartus II software automatically assigns a ninth channel as the transmitter clock output. You can edge- or center-align the transmitter clock by selecting the default PLL phase or selecting the negative-edge transmitter clock output. On the receiver side, the clock signal is connected to the receiver PLL's clock.

The multiplication factor *W* is also calculated automatically. The data rate divides by the input clock frequency to calculate the *W* factor. The deserialization factor (*J*) may be 4, 7, 8, or 10.

Figure 5–9 shows a DDR clock-to-data timing relationship with the clock center-aligned with respect to data. Figure 5–10 shows the connection between the receiver and transmitter circuits.

Figure 5–9. DDR Clock-to-Data Relationship

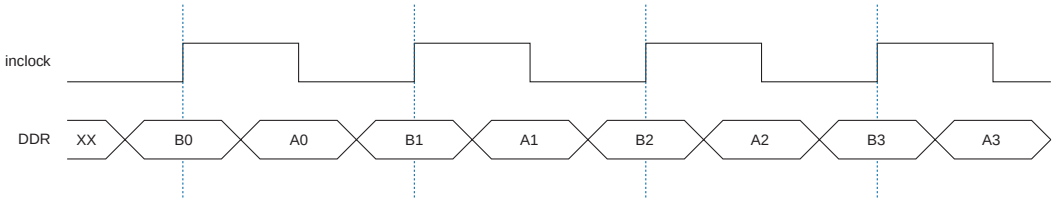
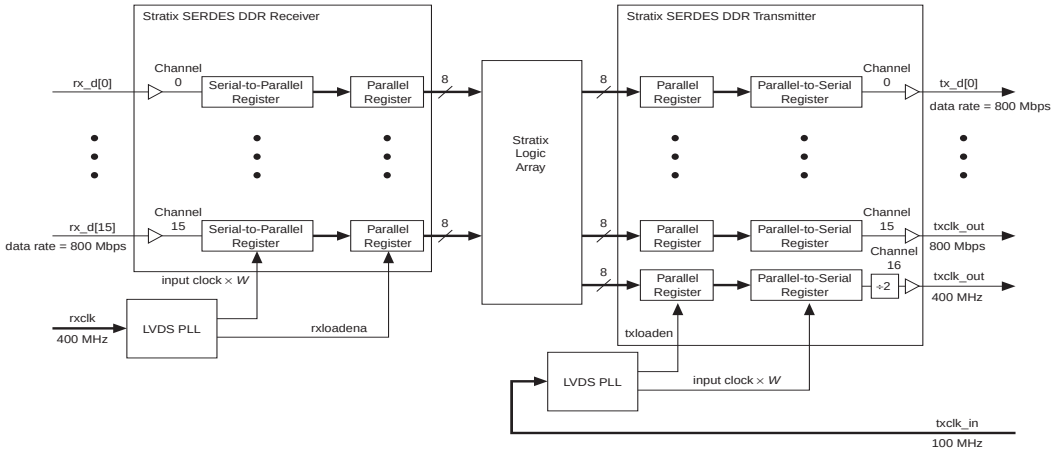


Figure 5–10. DDR Receiver & Transmitter Circuit Connection



Using SERDES to Implement SDR

Stratix devices support systems based on single data rate (SDR) operations applications by allowing you to directly transmit out the multiplied clock (as described in "SDR Transmitter Clock Output" on page 5–12). These systems are usually based on Utopia-4, SFI-4, or XSBI interfaces, and support various data rates.

An additional differential channel is automatically configured to produce the transmitter clock output signal and is transmitted along with the data.

For example, when a system is required to transmit 10 Gbps with a 1-to-1 clock-to-data ratio, program the SERDES with sixteen high-speed channels running at 624 Mbps each. The Quartus II software

automatically assigns a seventeenth channel as the transmitter clock output. You can edge- or center-align the transmitter clock output by selecting the default PLL phase or selecting the 90° phase of the PLL output. On the receiver side, the clock signal is connected to the receiver PLL's clock input, and you can assign identical clock-to-data alignment.

The multiplication factor *W* is calculated automatically. The data rate is dividing by the input clock frequency to calculate the *W* factor. The deserialization factor *J* may be 4, 7, 8, or 10.

Figure 5–11 shows an SDR clock-to-data timing relationship, with clock center aligned with respect to data. Figure 5–12 shows the connection between the receiver and transmitter circuits.

Figure 5–11. SDR Clock-to-Data Relationship

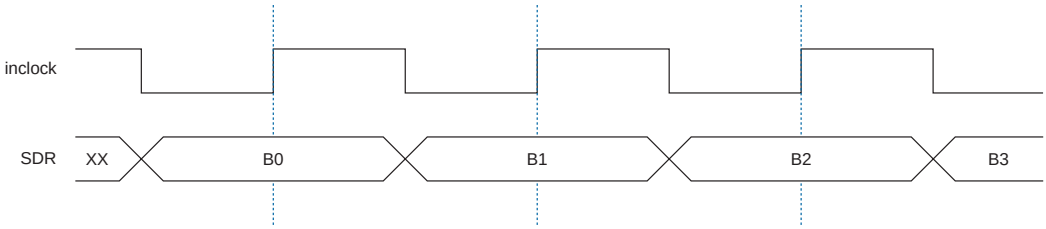
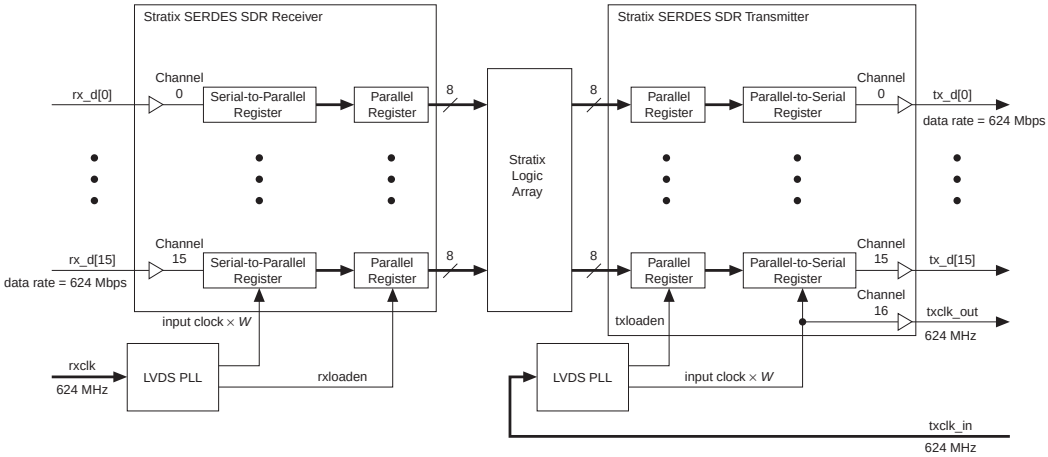


Figure 5–12. SDR Receiver & Transmitter Circuit Connection



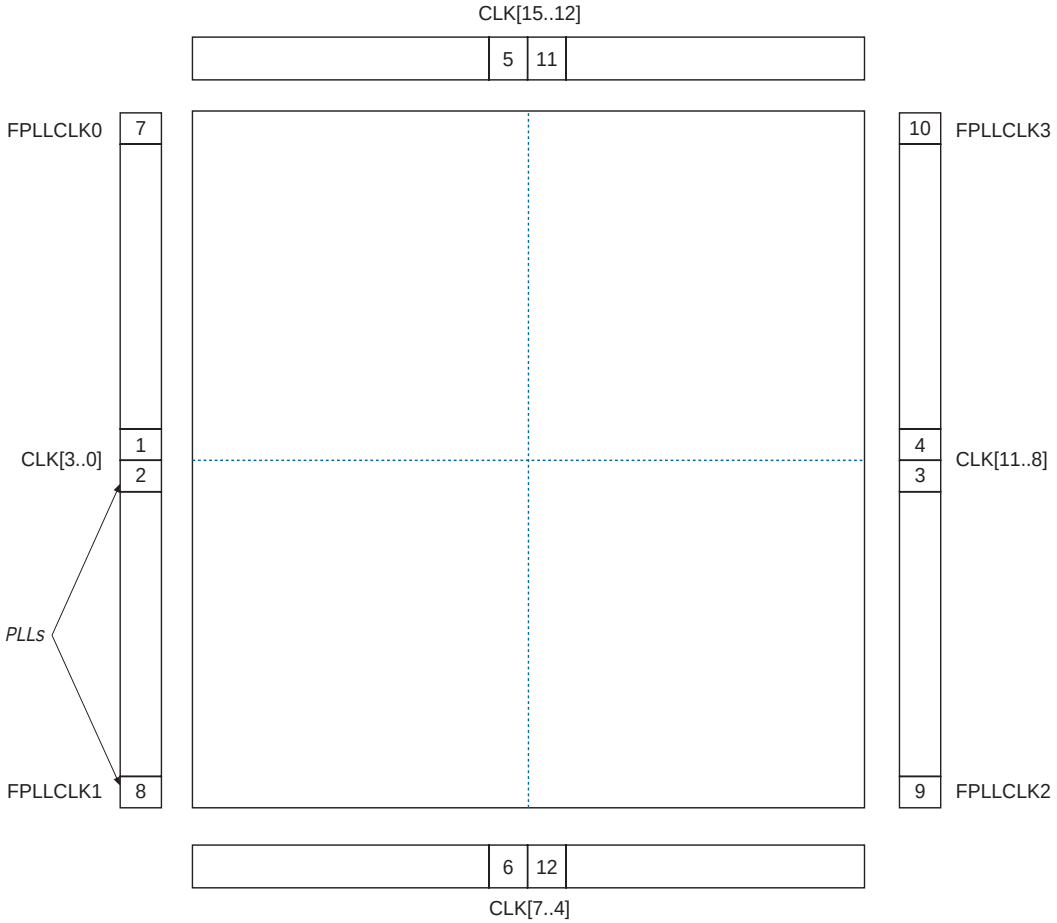
Differential I/O Interface & Fast PLLs

Stratix devices provide 16 dedicated global clocks, 8 dedicated fast regional I/O pins, and up to 16 regional clocks (four per device quadrant) that are fed from the dedicated global clock pins or PLL outputs. The 16 dedicated global clocks are driven either by global clock input pins that support all I/O standards or from enhanced and fast PLL outputs.

Stratix devices use the fast PLLs to implement clock multiplication and division to support the SERDES circuitry. The input clock is either multiplied by the *W* feedback factor and/or divided by the *J* factor. The resulting clocks are distributed to SERDES, local, or global clock lines.

Fast PLLs are placed in the center of the left and right sides for EP1S10 to EP1S25 devices. For EP1S30 to EP1S80 devices, fast PLLs are placed in the center of the left and right sides, as well as the device corners (see Figure 5–13). These fast PLLs drive a dedicated clock network to the SERDES in the rows above and below or top and bottom of the device as shown in Figure 5–13.

Figure 5–13. Stratix Fast PLL Positions & Clock Naming Convention *Note (1)*



Notes to Figure 5–13:

(1) Dedicated clock input pins on the right and left sides do not support PCI or PCI-X 1.0.

(2) PLLs 7, 8, 9, and 10 are not available on the EP1S30 device in the 780-pin FineLine BGA® package.

Clock Input & Fast PLL Output Relationship

Table 5–3 summarizes the PLL interface to the input clocks and the enable signal (ENA). Table 5–4 summarizes the clock networks each fast PLL can connect to across all Stratix family devices.

Table 5–3. Fast PLL Clock Inputs (Including Feedback Clocks) & Enables Note (1)

Input Pin	All Stratix Devices				EP1S30 to EP1S80 Devices Only			
	PLL 1	PLL 2	PLL 3	PLL 4	PLL 7	PLL 8	PLL 9	PLL 10
CLK0 *(2)*	v				v *(3)*			
CLK1	v							
CLK2 *(2)*		v				v *(3)*		
CLK3		v						
CLK4								
CLK5								
CLK6								
CLK7								
CLK8			v				v *(3)*	
CLK9 *(2)*			v					
CLK10				v				v *(3)*
CLK11 *(2)*				v				
CLK12								
CLK13								
CLK14								
CLK15								
ENA	v	v	v	v	v	v	v	v
FPLL7CLK					v			
FPLL8CLK						v		
FPLL9CLK							v	
FPLL10CLK								v

Notes to Table 5–3:

(1) PLLs 5, 6, 11, and 12 are not fast PLLs.

(2) Clock pins CLK0, CLK2, CLK9, CLK11, and pins FPLL[7..10]CLK do not support differential on-chip termination.

(3) Either a FPLLCLK pin or a CLK pin can drive the corner fast PLLs (PLL7, PLL8, PLL9, and PLL10) when used for general purpose. CLK pins cannot drive these fast PLLs in high-speed differential I/O mode.

Table 5–4. Fast PLL Relationship with Stratix Clock Networks (Part 1 of 2) *Notes (1), (2)*

Output Signal	All Stratix Devices				EP1S30 to EP1S80 Devices Only			
	PLL 1	PLL 2	PLL 3	PLL 4	PLL 7	PLL 8	PLL 9	PLL 10
GCLK0	v							
GCLK1	v							
GCLK2		v						
GCLK3		v						
GCLK4			v					
GCLK9			v					
GCLK10				v				
GCLK11				v				
RCLK1	v	v			v			
RCLK2	v	v			v			
RCLK3	v	v				v		
RCLK4	v	v				v		
RCLK9			v	v			v	
RCLK10			v	v			v	
RCLK11			v	v				v
RCLK12			v	v				v
DIFFIOCLK1	v							
DIFFIOCLK2	v							
DIFFIOCLK3		v						
DIFFIOCLK4		v						
DIFFIOCLK5			v					
DIFFIOCLK6			v					
DIFFIOCLK7				v				
DIFFIOCLK8				v				
DIFFIOCLK9					v			
DIFFIOCLK10					v			
DIFFIOCLK11						v		
DIFFIOCLK12						v		
DIFFIOCLK13							v	

Table 5–4. Fast PLL Relationship with Stratix Clock Networks (Part 2 of 2) Notes (1), (2)

Output Signal	All Stratix Devices				EP1S30 to EP1S80 Devices Only			
	PLL 1	PLL 2	PLL 3	PLL 4	PLL 7	PLL 8	PLL 9	PLL 10
DIFFIOCLK14							v	
DIFFIOCLK15								v
DIFFIOCLK16								v

Notes to Table 5–4:

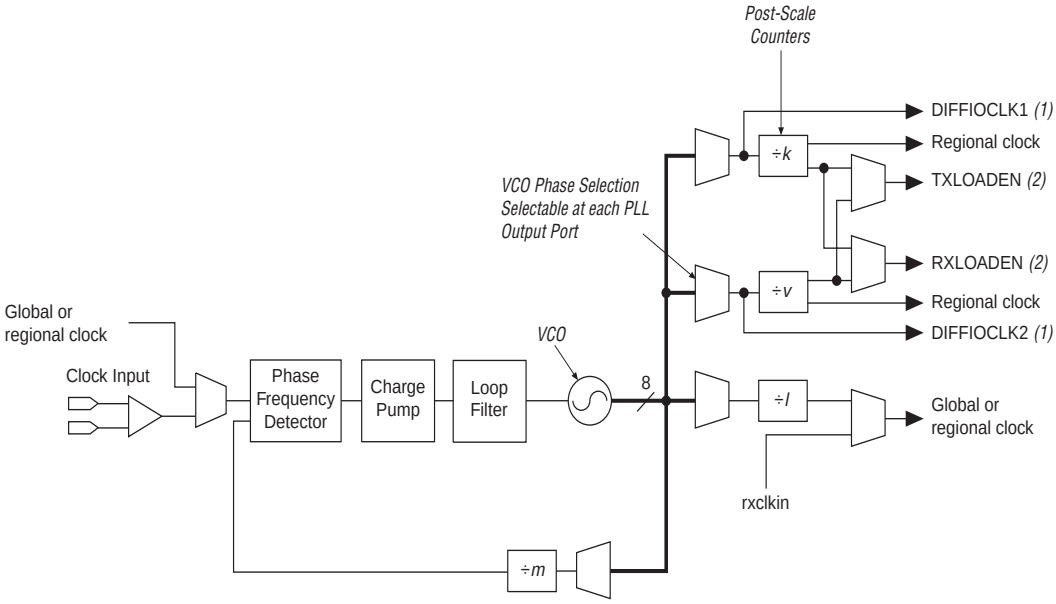
(1) PLLs 5, 6, 11, and 12 are not fast PLLs.

(2) The input clock for PLLs used to clock receiver the `rx_inclock` port on the `altlvds_rx` megafunction must be driven by a dedicated clock pin (`CLK[3..0]` and `CLK[8..11]`) or the corner pins that clock the corner PLLs (`FPLL[10..7]CLK`).

Fast PLL Specifications

You can drive the fast PLLs by an external pin or any one of the sectional clocks [21..0]. You can connect the clock input directly to local or global clock lines, as shown in Figure 5–14. You cannot use the sectional-clock inputs to the fast PLL's input multiplexer for the receiver PLL. You can only use the sectional clock inputs in the transmitter only mode or as a general purpose PLL.

Figure 5–14. Fast PLL Block Diagram



Notes to Figure 5–14:

(1) In high-speed differential I/O mode, the high-speed PLL clock feeds the SERDES. Stratix devices only support one rate of data transfer per fast PLL in high-speed differential I/O mode.

(2) Control signal for high-speed differential I/O SERDES.

You can multiply the input clock by a factor of 1 to 16. The multiplied clock is used for high-speed serialization or deserialization operations. Fast PLL specifications are shown in the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1*. The voltage controlled oscillators (VCOs) are designed to operate within the frequency range of 300 to 840 MHz, to provide data rates of up to 840 Mbps.

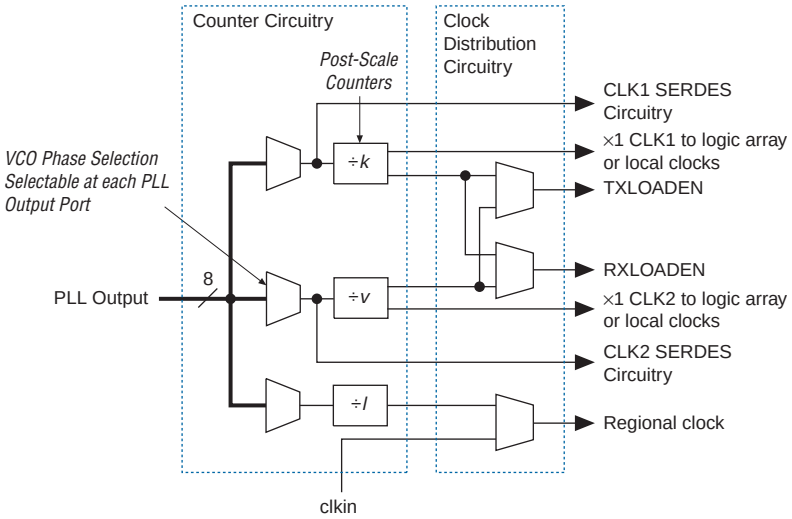
High-Speed Phase Adjust

There are eight phases of the multiplied clock at the PLL output, each delayed by 45° from the previous clock and synchronized with the original clock. The three multiplexers (shown in Figure 5–14) select one of the delayed, multiplied clocks. The PLL output drives the three counters *k*, *v*, and *l*. You can program the three individual post scale counters (*k*, *v*, and *l*) independently for division ratio or phase. The selected PLL output is used for the serialization or deserialization process in SERDES.

Counter Circuitry

The multiplied clocks bypass the counter taps *k* and *v* to directly feed the SERDES serial registers. These two taps also feed to the quadrant local clock network and the dedicated RXLOADENA or TXLOADENA pins, as shown in Figure 5–15. Both *k* and *v* are utilized simultaneously during the data-realignment procedure. When the design does not use the data realignment, both TXLOADEN and RXLOADEN pins use a single counter.

Figure 5–15. Fast PLL Connection to Logic Array



The Stratix device fast PLL has another GCLK connection for general-purpose applications. The third tap *l* feeds the quadrant local clock as well as the global clock network. You can use the *l* counter's multiplexer for applications requiring the device to connect the incoming clock directly to the local or global clocks. You can program the multiplexer to connect the RXCLKIN signal directly to the local or global clock lines. Figure 5–15 shows the connection between the incoming clock, the *l* tap, and the local or global clock lines.

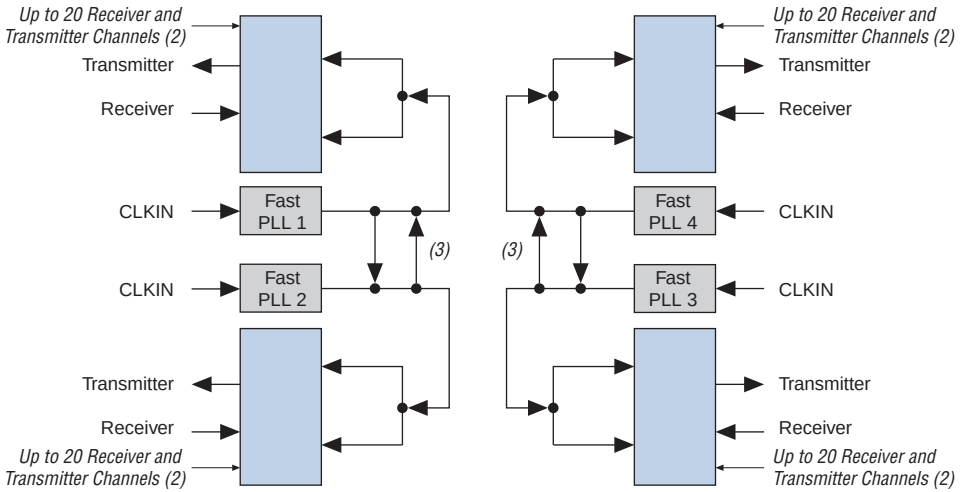
The differential clock selection is made per differential bank. Since the length of the clock tree limits the performance, each fast PLL should drive only one differential bank.

Fast PLL SERDES Channel Support

The Quartus II MegaWizard Plug-In Manager only allows you to implement up to 20 receiver or 20 transmitter channels for each fast PLL. These channels operate at up to 840 Mbps. For more information on implementing more than 20 channels, see "Fast PLLs" on page 5–52. The receiver and transmitter channels are interleaved such that each I / O bank on the left and right side of the device has one receiver channel and one transmitter channel per row. Figure 5–16 shows the fast PLL and channel layout in EP1S10, EP1S20, and EP1S25 devices. Figure 5–17 shows the fast PLL and channel layout in EP1S30 to EP1S80 devices.

f For more the number of channels in each device, see Tables 5–10 through 5–14.

Figure 5–16. Fast PLL & Channel Layout in EP1S10, EP1S20 & EP1S25 Devices *Note (1)*



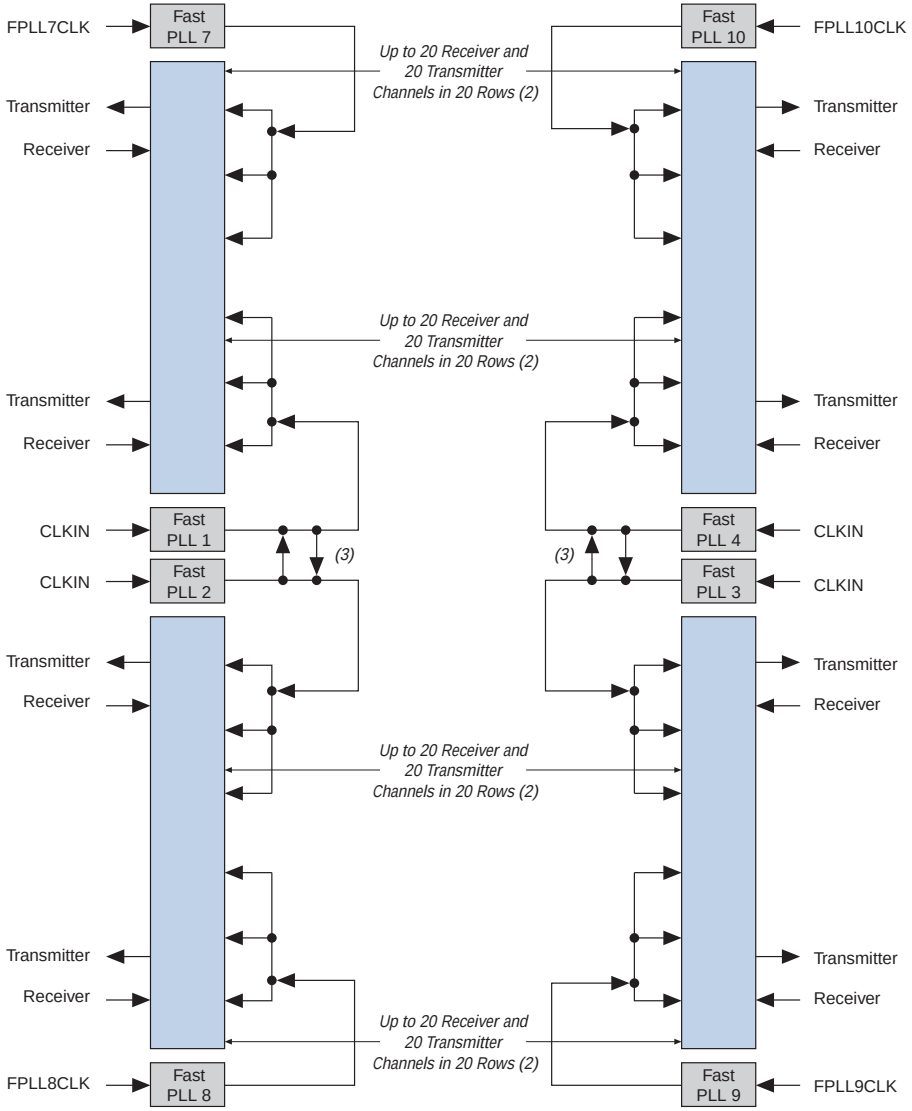
Notes to Figure 5–16:

(1) Wire-bond packages only support up to 624 Mbps until characterization shows otherwise.

(2) See Tables 5–10 through 5–14 for the exact number of channels each package and device density supports.

(3) There is a multiplexer here to select the PLL clock source. If a PLL uses this multiplexer to clock channels outside of its bank quadrant (e.g., if PLL 2 clocks PLL 1's channel region), those clocked channels support up to 840 Mbps.

Figure 5–17. Fast PLL & Channel Layout in EP1S30 to EP1S80 Devices *Note (1)*



Notes to Figure 5–17:

(1) Wire-bond packages only support up to 624-Mbps until characterization shows otherwise.

(2) See Tables 5–10 through 5–14 for the exact number of channels each package and device density supports.

(3) There is a multiplexer here to select the PLL clock source. If a PLL uses this multiplexer to clock channels outside of its bank quadrant (e.g., if PLL 2 clocks PLL 1's channel region), those clocked channels support up to 840 Mbps.

Advanced Clear & Enable Control

There are several control signals for clearing and enabling PLLs and their outputs. You can use these signals to control PLL resynchronization and to gate PLL output clocks for low-power applications.

The PLLENABLE pin is a dedicated pin that enables and disables Stratix device enhanced and fast PLLs. When the PLLENABLE pin is low, the clock output ports are driven by GND and all the PLLs go out of lock. When the PLLENABLE pin goes high again, the PLLs relock and resynchronize to the input clocks.

The reset signals are reset/resynchronization inputs for each enhanced PLL. Stratix devices can drive these input signals from an input pin or from LEs. When driven high, the PLL counters reset, clearing the PLL output and placing the PLL out of lock. When driven low again, the PLL resynchronizes to its input as it relocks.

Receiver Data Realignment

Most systems using serial differential I/O data transmission require a certain data-realignment circuit. Stratix devices contain embedded data-realignment circuitry. While normal I/O operation guarantees that data is captured, it does not guarantee the parallelization boundary, as this point is randomly determined based on the power-up of both communicating devices. The data-realignment circuitry corrects for bit misalignments by shifting, or delaying, data bits.

Data Realignment Principles of Operation

Stratix devices use a realignment and clock distribution circuitry (described in "Counter Circuitry" on page 5–22) for data realignment.

Set the internal `rx_data_align` node end high to assert the data-realignment circuitry. When this node is switched from a low to a high state, the realignment circuitry is activated and the data is delayed by one bit. To ensure the rising edge of the `rx_data_align` node end is latched into the PLL, the `rx_data_align` node end should stay high for at least two low-frequency clock cycles.

An external circuit or an internal custom-made state machine using LEs can generate the signal to pull the `rx_data_align` node end to a high state.

When the data realignment circuitry is activated, it generates an internal pulse Sync S1 or Sync S2 that disables one of the two counters used for the SERDES operation (described in "Counter Circuitry" on page 5–22). One counter is disabled for one high-frequency clock cycle, delaying the

RXLOADEN signal and dropping the first incoming bit of the serial input data stream located in the first serial register of the SERDES circuitry (shown in Figure 5–3 on page 5–8).

Figure 5–18 shows the function-timing diagram of a Stratix SERDES in normal ×8 mode, and Figure 5–19 shows the function-timing diagrams of a Stratix SERDES when data realignment is used.

Figure 5–18. SERDES Function Timing Diagram in Normal Operation

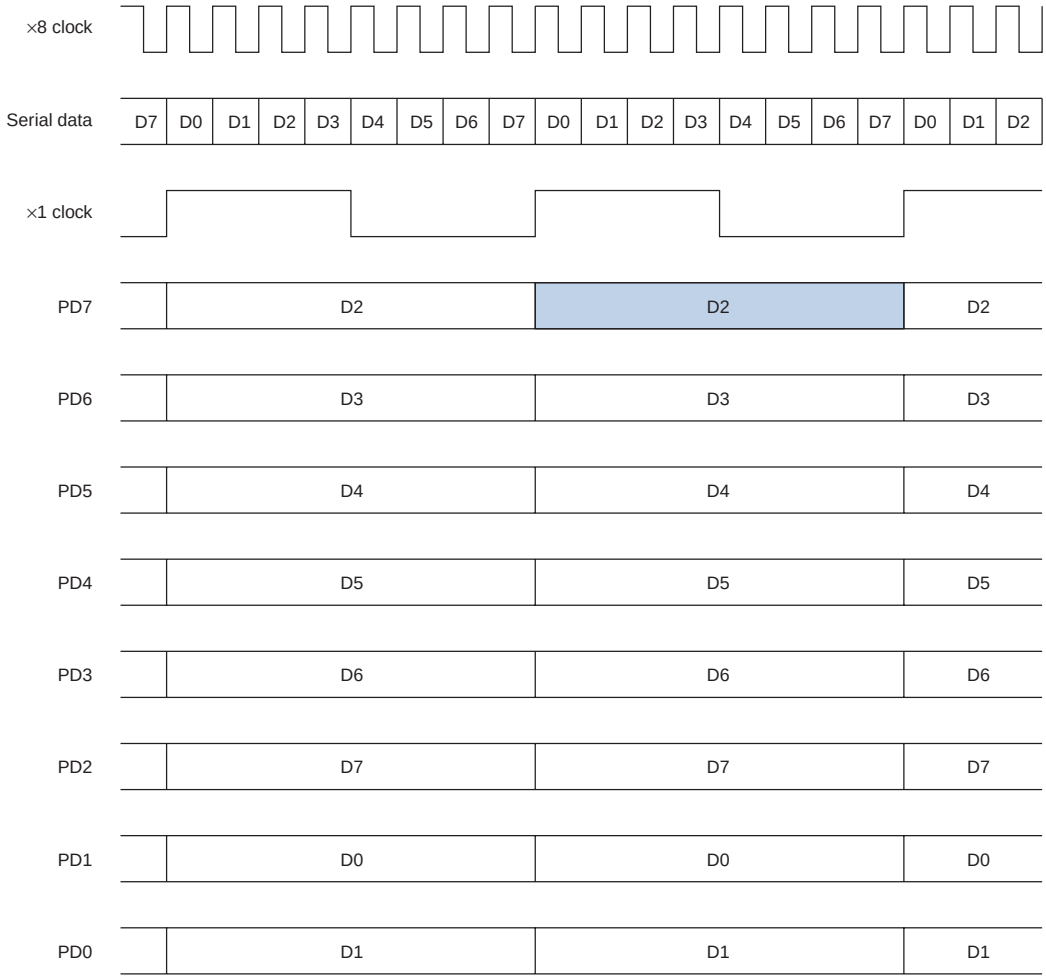
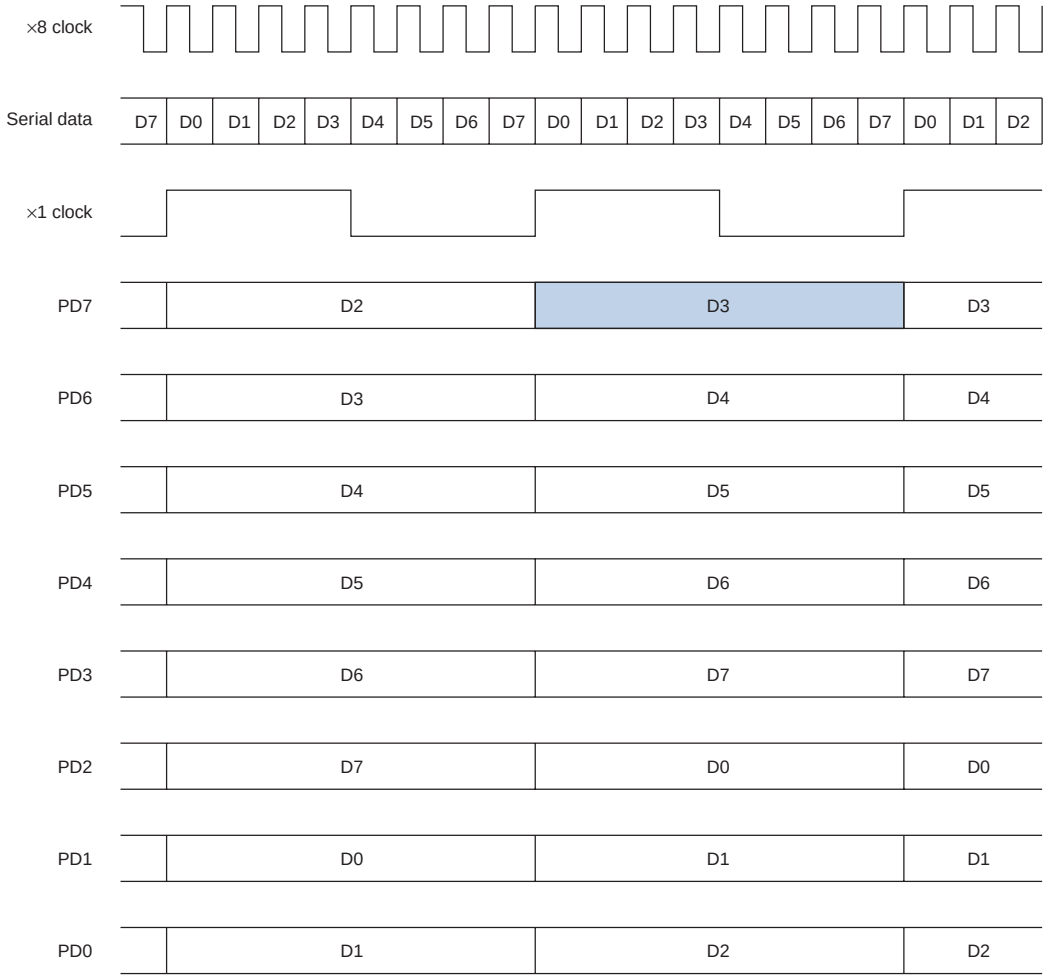


Figure 5–19. SERDES Function Timing Diagram with Data-Realignment Operation



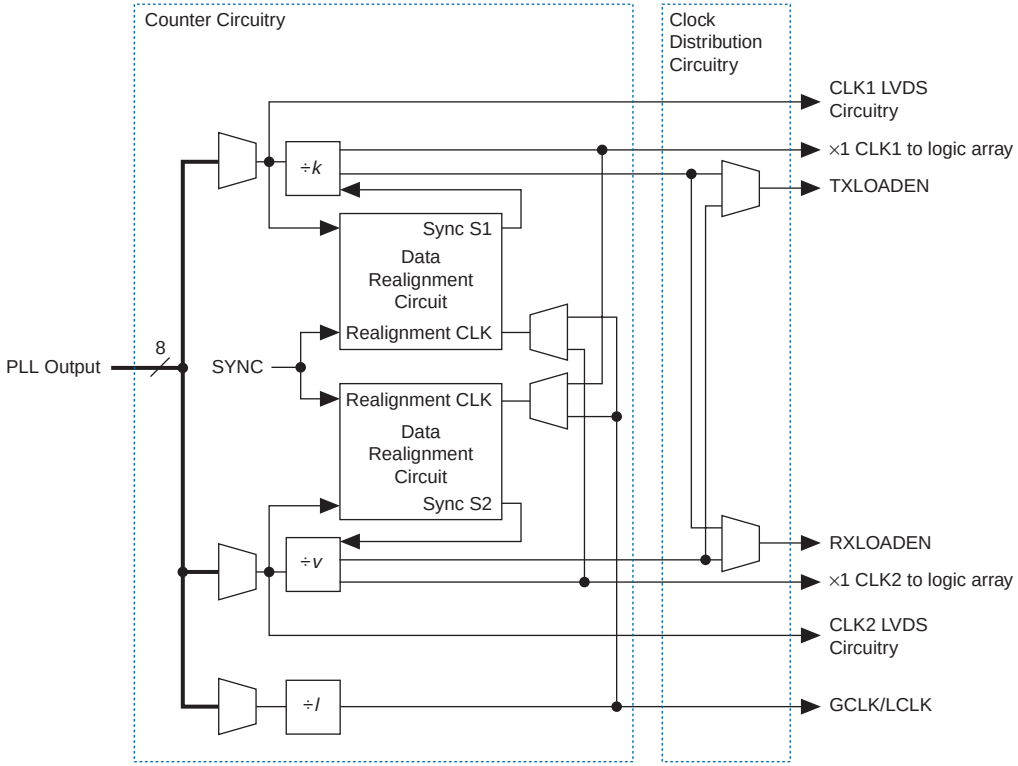
Generating the TXLOADEN Signal

The TXLOADEN signal controls the transfer of data between the SERDES circuitry and the logic array when data realignment is used. To prevent the interruption of the TXLOADEN signal during data realignment, both *k* and *v* counter are used.

In normal operation the TXLOADEN signal is generated by the *k* counter. However, during the data-realignment operation this signal is generated by either counter. When the *k* counter is used for realignment, the

TXLOADEN signal is generated by the *v* counter, and when the *v* counter is used for realignment, the TXLOADEN signal is generated by the *k* counter, as shown in Figure 5–20.

Figure 5–20. Realignment Circuit TXLOADEN Signal Control *Note (1)*



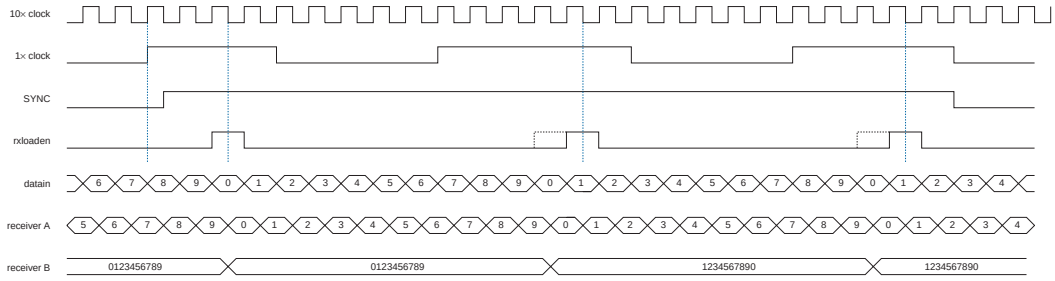
Note to Figure 5–20:

(1) This figure does not show additional realignment circuitry.

Realignment Implementation

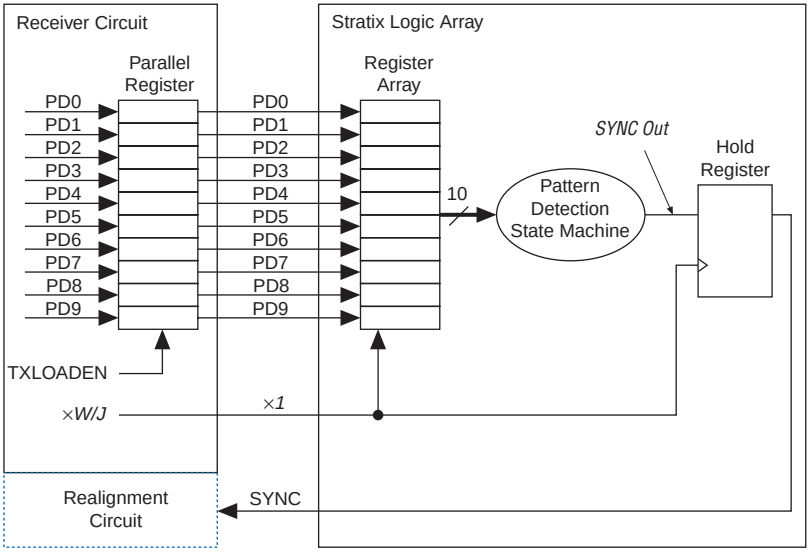
The realignment signal (SYNC) is used for data realignment and reframing. An external pin (RX_DATA_ALIGN) or an internal signal controls the `rx_data_align` node end. When the `rx_data_align` node end is asserted high for at least two low-frequency clock cycles, the RXLOADEN signal is delayed by one high-frequency clock period and the parallel bits shift by one bit. Figure 5–21 shows the timing relationship between the high-frequency clock, the RXLOADEN signal, and the parallel data.

Figure 5–21. Realignment by rx_data_align Node End



A state machine can generate the realignment signal to control the alignment procedure. Figure 5–22 shows the connection between the realignment signal and the `rx_data_align` node end.

Figure 5–22. SYNC Signal Path to Realignment Circuit



To guarantee that the `rx_data_align` signal generated by a user state machine is latched correctly by the counters, the user circuit must meet certain requirements.

- The design must include an input synchronizing register to ensure that data is synchronized to the ×1 clock.

- After the pattern detection state machine, use another synchronizing register to capture the generated SYNC signal and synchronize it to the ×1 clock.
- Since the skew in the path from the output of this synchronizing register to the PLL is undefined, the state machine must generate a pulse that is high for two ×1 clock periods.
- Since the SYNC generator circuitry only generates a single fast clock period pulse for each SYNC pulse, you cannot generate additional SYNC pulses until the comparator signal is reset low.
- To guarantee the pattern detection state machine does not incorrectly generate multiple SYNC pulses to shift a single bit, the state machine must hold the SYNC signal low for at least three ×1 clock periods between pulses.

Source-Synchronous Timing Budget

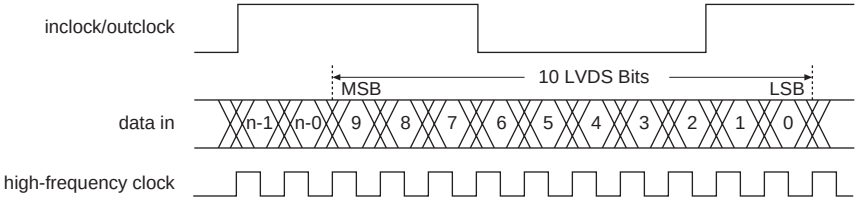
This section discusses the timing budget, waveforms, and specifications for source-synchronous signaling in Stratix devices. LVDS, LVPECL, PCML, and HyperTransport I/O standards enable high-speed data transmission. This high data-transmission rate results in better overall system performance. To take advantage of fast system performance, you must understand how to analyze timing for these high-speed signals. Timing analysis for the differential block is different from traditional synchronous timing analysis techniques.

Rather than focusing on clock-to-output and setup times, source-synchronous timing analysis is based on the skew between the data and the clock signals. High-speed differential data transmission requires you to use timing parameters provided by IC vendors and to consider board skew, cable skew, and clock jitter. This section defines the source-synchronous differential data orientation timing parameters, and timing budget definitions for Stratix devices, and explains how to use these timing parameters to determine a design's maximum performance.

Differential Data Orientation

There is a set relationship between an external clock and the incoming data. For operation at 840 Mbps and $W = 10$, the external clock is multiplied by 10 and phase-aligned by the PLL to coincide with the sampling window of each data bit. The third falling edge of high-frequency clock is used to strobe the incoming high-speed data. Therefore, the first two bits belong to the previous cycle. Figure 5–23 shows the data bit orientation of the ×10 mode as defined in the Quartus II software.

Figure 5–23. Bit Orientation in the Quartus II Software



Differential I/O Bit Position

Data synchronization is necessary for successful data transmission at high frequencies. Figure 5–24 shows the data bit orientation for a receiver channel operating in ×8 mode. Similar positioning exists for the most significant bits (MSBs) and least significant bits (LSBs) after deserialization, as listed in Table 5–5.

Figure 5–24. Bit Order for One Channel of Differential Data

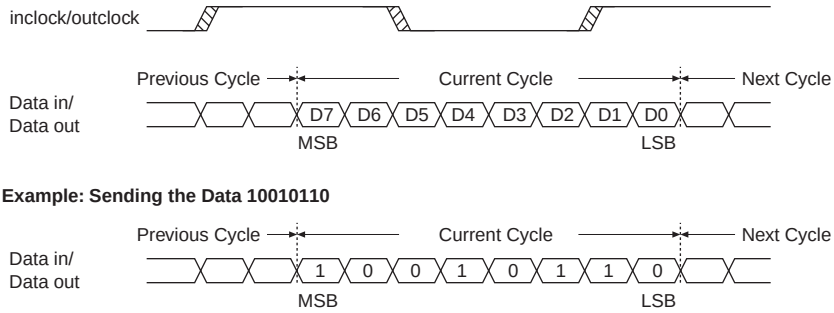


Table 5–5 shows the conventions for differential bit naming for 18 differential channels. However, the MSB and LSB are increased with the number of channels used in a system.

Table 5–5. LVDS Bit Naming

Receiver Data Channel Number	Internal 8-Bit Parallel Data	
	MSB Position	LSB Position
1	7	0
2	15	8
3	23	16
4	31	24
5	39	32
6	47	40
7	55	48
8	63	56
9	71	64
10	79	72
11	87	80
12	95	88
13	103	96
14	111	104
15	119	112
16	127	120
17	135	128
18	143	136

Timing Definition

The specifications used to define high-speed timing are described in Table 5–6.

Table 5–6. High-Speed Timing Specifications & Terminology (Part 1 of 2)

High-Speed Timing Specification	Terminology
t_C	High-speed receiver/transmitter input and output clock period.
f_{HSCLK}	High-speed receiver/transmitter input and output clock frequency.
t_{RISE}	Low-to-high transmission time.

Table 5–6. High-Speed Timing Specifications & Terminology (Part 2 of 2)

High-Speed Timing Specification	Terminology
t_{FALL}	High-to-low transmission time.
Timing unit interval (TUI)	The timing budget allowed for skew, propagation delays, and data sampling window. (TUI = 1/(Receiver Input Clock Frequency × Multiplication Factor) = t_C/w).
f_{HSDR}	Maximum LVDS data transfer rate (f_{HSDR} = 1/TUI).
Channel-to-channel skew (TCCS)	The timing difference between the fastest and slowest output edges, including t_{CO} variation and clock skew. The clock is included in the TCCS measurement.
Sampling window (SW)	The period of time during which the data must be valid in order for you to capture it correctly. The setup and hold times determine the ideal strobe position within the sampling window. SW = t_{SW} (max) – t_{SW} (min).
Input jitter (peak-to-peak)	Peak-to-peak input jitter on high-speed PLLs.
Output jitter (peak-to-peak)	Peak-to-peak output jitter on high-speed PLLs.
t_{DUTY}	Duty cycle on high-speed transmitter output clock.
t_{LOCK}	Lock time for high-speed transmitter and receiver PLLs.

Tables 5–7 and 5–8 show the high-speed I/O timing for Stratix devices

Table 5–7. High-Speed I/O Specifications for Flip-Chip Packages (Part 1 of 3) *Notes (1), (2)*

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSCLK} (Clock frequency) (LVDS, LVPECL, HyperTransport technology) $f_{HSCLK} = f_{HSDR} / W$	*W* = 4 to 30	10		210	10		210	10		156	10		115.5	MHz
	W = 2 (Serdes bypass)	50		231	50		231	50		231	50		231	MHz
	W = 2 (Serdes used)	150		420	150		420	150		312	150		231	MHz
	W = 1 (Serdes bypass)	100		462	100		462	100		462	100		462	MHz
	W = 1 (Serdes used)	300		717	300		717	300		624	300		462	MHz
f_{HSDR} Device operation (LVDS, LVPECL, HyperTransport technology)	*J* = 10	300		840	300		840	300		640	300		462	Mbps
	J = 8	300		840	300		840	300		640	300		462	Mbps
	J = 7	300		840	300		840	300		640	300		462	Mbps
	J = 4	300		840	300		840	300		640	300		462	Mbps
	J = 2	100		462	100		462	100		640	100		462	Mbps
	J = 1 (LVDS and LVPECL only)	100		462	100		462	100		640	100		462	Mbps
f_{HSCLK} (Clock frequency) (PCML) $f_{HSCLK} = f_{HSDR} / W$	*W* = 4 to 30 (Serdes used)	10		100	10		100	10		77.75	10		77.75	MHz
	W = 2 (Serdes bypass)	50		200	50		200	50		150	50		150	MHz
	W = 2 (Serdes used)	150		200	150		200	150		155.5	150		155.5	MHz
	W = 1 (Serdes bypass)	100		250	100		250	100		200	100		200	MHz
	W = 1 (Serdes used)	300		400	300		400	300		311	300		311	MHz

Table 5–7. High-Speed I/O Specifications for Flip-Chip Packages (Part 2 of 3) *Notes (1), (2)*

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSDR} Device operation (PCML)	*J* = 10	300		400	300		400	300		311	300		311	Mbps
	J = 8	300		400	300		400	300		311	300		311	Mbps
	J = 7	300		400	300		400	300		311	300		311	Mbps
	J = 4	300		400	300		400	300		311	300		311	Mbps
	J = 2	100		400	100		400	100		300	100		300	Mbps
	J = 1	100		250	100		250	100		200	100		200	Mbps
TCCS	All			200			200			300			300	ps
SW	PCML (*J* = 4, 7, 8, 10)	750			750			800			800			ps
	PCML (*J* = 2)	900			900			1,200			1,200			ps
	PCML (*J* = 1)	1,500			1,500			1,700			1,700			ps
	LVDS and LVPECL (*J* = 1)	500			500			550			550			ps
	LVDS, LVPECL, HyperTransport technology (*J* = 2 through 10)	440			440			500			500			ps
Input jitter tolerance (peak-to-peak)	All			250			250			250			250	ps
Output jitter (peak-to-peak)	All			160			160			200			200	ps
Output t_{RISE}	LVDS	80	110	120	80	110	120	80	110	120	80	110	120	ps
	HyperTransport technology	110	170	200	110	170	200	120	170	200	120	170	200	ps
	LVPECL	90	130	150	90	130	150	100	135	150	100	135	150	ps
	PCML	80	110	135	80	110	135	80	110	135	80	110	135	ps

Table 5–7. High-Speed I/O Specifications for Flip-Chip Packages (Part 3 of 3) *Notes (1), (2)*

Symbol	Conditions	-5 Speed Grade			-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Output t_{FALL}	LVDS	80	110	120	80	110	120	80	110	120	80	110	120	ps
	HyperTransport technology	110	170	200	110	170	200	110	170	200	110	170	200	ps
	LVPECL	90	130	160	90	130	160	100	135	160	100	135	160	ps
	PCML	105	140	175	105	140	175	110	145	175	110	145	175	ps
t_{DUTY}	LVDS (*J* = 2 through 10)	47.5	50	52.5	47.5	50	52.5	47.5	50	52.5	47.5	50	52.5	%
	LVDS (*J* =1) and LVPECL, PCML, HyperTransport technology	45	50	55	45	50	55	45	50	55	45	50	55	%
t_{LOCK}	All			100			100			100			100	µs

Notes to Table 5–7:

(1) When *J* = 4, 7, 8, and 10, the SERDES block is used.

(2) When *J* = 2 or J = 1, the SERDES is bypassed.

Table 5–8. High-Speed I/O Specifications for Wire-Bond Packages (Part 1 of 3)

Symbol	Conditions	-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSCLK} (Clock frequency) (LVDS,LVPECL, HyperTransport technology) $f_{HSCLK} = f_{HSDR} / W$	*W* = 4 to 30 (Serdes used)	10		156	10		115.5	10		115.5	MHz
	W = 2 (Serdes bypass)	50		231	50		231	50		231	MHz
	W = 2 (Serdes used)	150		312	150		231	150		231	MHz
	W = 1 (Serdes bypass)	100		311	100		270	100		270	MHz
	W = 1 (Serdes used)	300		624	300		462	300		462	MHz

Table 5–8. High-Speed I/O Specifications for Wire-Bond Packages (Part 2 of 3)

Symbol	Conditions	-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
f_{HSDR} Device operation, (LVDS,LVPECL, HyperTransport technology)	J = 10	300		624	300		462	300		462	Mbps
	J = 8	300		624	300		462	300		462	Mbps
	J = 7	300		624	300		462	300		462	Mbps
	J = 4	300		624	300		462	300		462	Mbps
	J = 2	100		462	100		462	100		462	Mbps
	J = 1 (LVDS and LVPECL only)	100		311	100		270	100		270	Mbps
f_{HSCLK} (Clock frequency) (PCML) $f_{HSCLK} = f_{HSDR} / W$	*W* = 4 to 30 (Serdes used)	10		77.75							MHz
	W = 2 (Serdes bypass)	50		150	50		77.5	50		77.5	MHz
	W = 2 (Serdes used)	150		155.5							MHz
	W = 1 (Serdes bypass)	100		200	100		155	100		155	MHz
	W = 1 (Serdes used)	300		311							MHz
Device operation, f_{HSDR} (PCML)	J = 10	300		311							Mbps
	J = 8	300		311							Mbps
	J = 7	300		311							Mbps
	J = 4	300		311							Mbps
	J = 2	100		300	100		155	100		155	Mbps
	J = 1	100		200	100		155	100		155	Mbps
TCCS	All			400			400			400	ps
SW	PCML (J = 4, 7, 8, 10) only	800			800			800			ps
	PCML (J = 2) only	1,200			1,200			1,200			ps
	PCML (J = 1) only	1,700			1,700			1,700			ps
	LVDS and LVPECL (J = 1) only	550			550			550			ps
	LVDS, LVPECL, HyperTransport technology (J = 2 through 10) only	500			500			500			ps

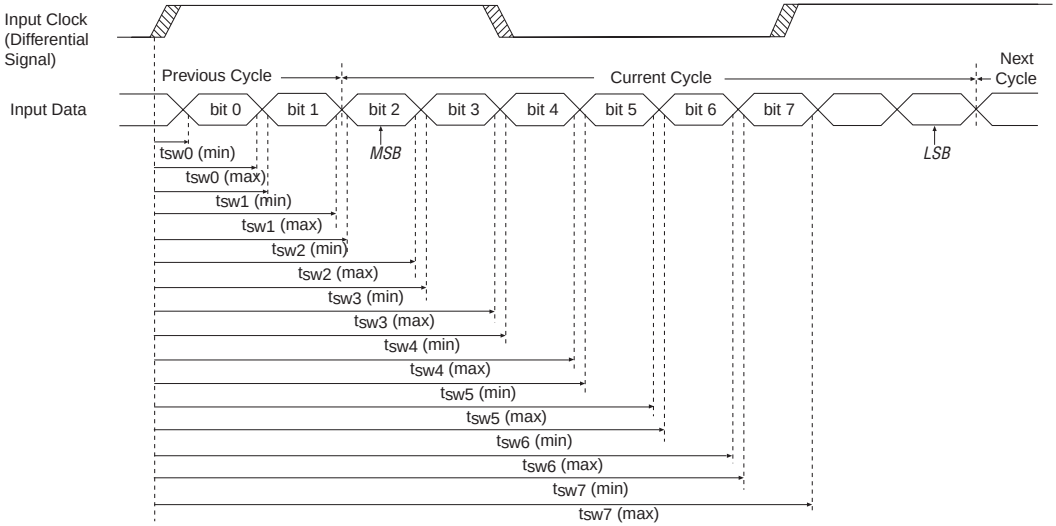
Table 5–8. High-Speed I/O Specifications for Wire-Bond Packages (Part 3 of 3)

Symbol	Conditions	-6 Speed Grade			-7 Speed Grade			-8 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input jitter tolerance (peak-to-peak)	All			250			250			250	ps
Output jitter (peak-to-peak)	All			200			200			200	ps
Output t_{RISE}	LVDS	80	110	120	80	110	120	80	110	120	ps
	HyperTransport technology	120	170	200	120	170	200	120	170	200	ps
	LVPECL	100	135	150	100	135	150	100	135	150	ps
	PCML	80	110	135	80	110	135	80	110	135	ps
Output t_{FALL}	LVDS	80	110	120	80	110	120	80	110	120	ps
	HyperTransport	110	170	200	110	170	200	110	170	200	ps
	LVPECL	100	135	160	100	135	160	100	135	160	ps
	PCML	110	145	175	110	145	175	110	145	175	ps
t_{DUTY}	LVDS (J =2..10) only	47.5	50	52.5	47.5	50	52.5	47.5	50	52.5	%
	LVDS (J =1) and LVPECL, PCML, HyperTransport technology	45	50	55	45	50	55	45	50	55	%
t_{LOCK}	All			100			100			100	µs

Input Timing Waveform

Figure 5–25 illustrates the essential operations and the timing relationship between the clock cycle and the incoming serial data. For a functional description of the SERDES, see "Principles of SERDES Operation" on page 5–6.

Figure 5–25. Input Timing Waveform *Note (1)*



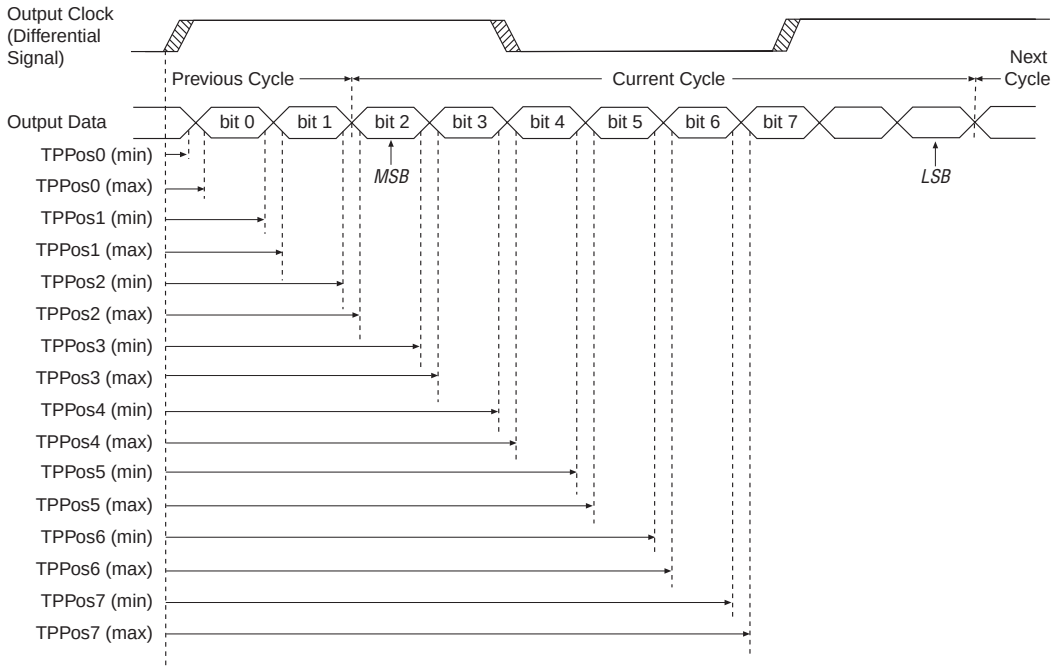
Note to Figure 5–25:

(1) The timing specifications are referenced at a 100-mV differential voltage.

Output Timing

The output timing waveform in Figure 5–26 illustrates the relationship between the output clock and the serial output data stream.

Figure 5–26. Output Timing Waveform *Note (1)*



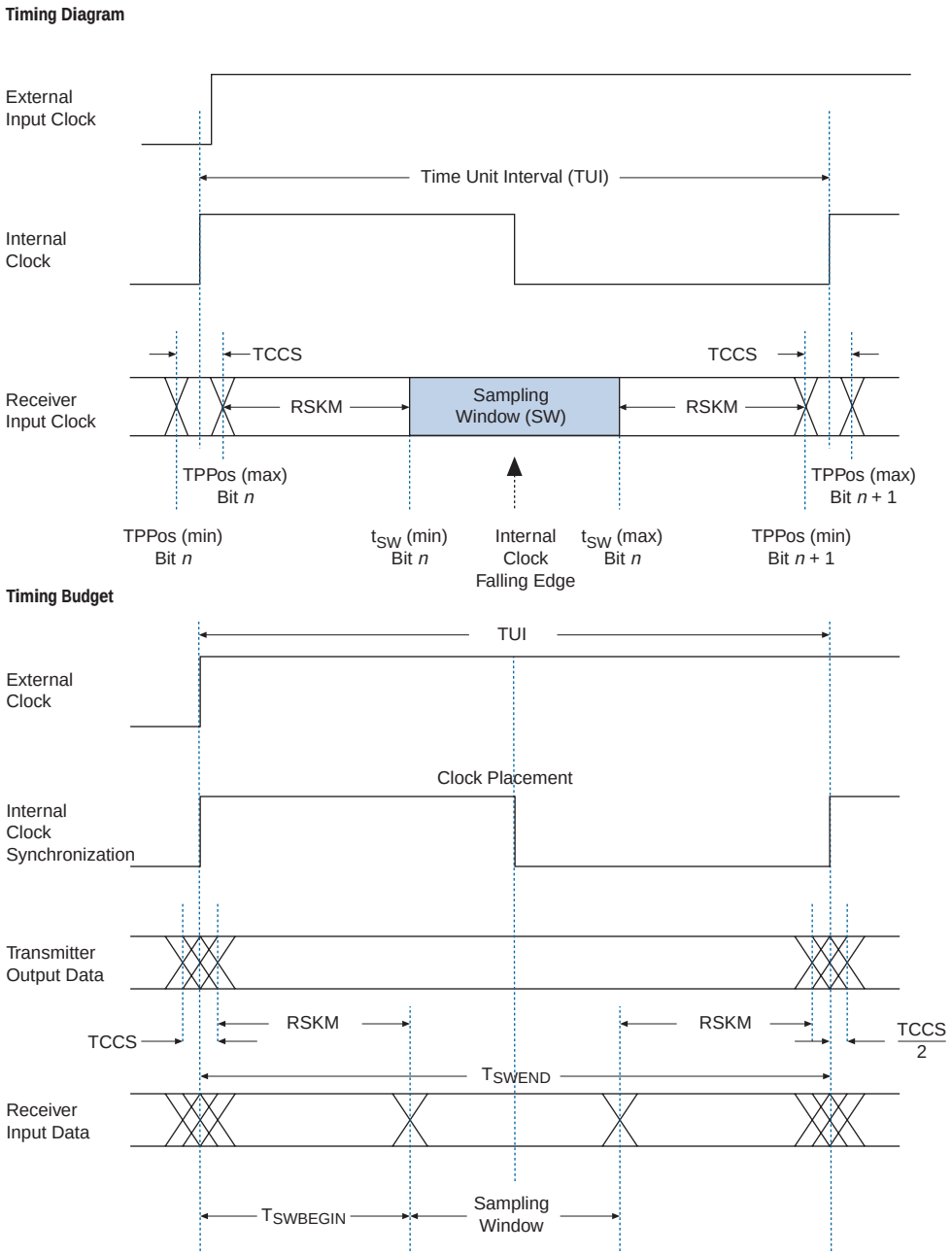
Note to Figure 5–26:

(1) The timing specifications are referenced at a 250-mV differential voltage.

Receiver Skew Margin

Change in system environment, such as temperature, media (cable, connector, or PCB) loading effect, a receiver's inherent setup and hold, and internal skew, reduces the sampling window for the receiver. The timing margin between receiver's clock input and the data input sampling window is known as RSKM. Figure 5–27 illustrates the relationship between the parameter and the receiver's sampling window.

Figure 5–27. Differential High-Speed Timing Diagram & Timing Budget



Switching Characteristics

Timing specifications for Stratix devices are listed in Tables 5–7 and 5–8. You can also find Stratix device timing information in the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1*.

Timing Analysis

Differential timing analysis is based on skew between data and the clock signals. For static timing analysis, the timing characteristics of the differential I/O standards are guaranteed by design and depend on the frequency at which they are operated. Use the values in the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* to calculate system timing margins for various I/O protocols. For detailed descriptions and implementations of these protocols, see the Altera web site at **www.altera.com**.

SERDES Bypass DDR Differential Signaling

Each Stratix device high-speed differential I/O channel can transmit or receive data in by-two (×2) mode at up to 624 Mbps using PLLs. These pins do not require dedicated SERDES circuitry and they implement serialization and deserialization with minimal logic.

SERDES Bypass DDR Differential Interface Review

Stratix devices use dedicated DDR circuitry to implement ×2 differential signaling. Although SDR circuitry samples data only at the positive edge of the clock, DDR captures data on both the rising and falling edges for twice the transfer rate of SDR. Stratix device shift registers, internal global PLLs, and I/O cells can perform serial-to-parallel conversions on incoming data and parallel-to-serial conversion on outgoing data.

SERDES Clock Domains

The SERDES bypass differential signaling can use any of the many clock domains available in Stratix devices. These clock domains fall into four categories: global, regional, fast regional, and internally generated.

General-purpose PLLs generate the global clock domains. The fast PLLs can generate additional global clocks domains. Each PLL features two taps that directly drive two unique global clock networks. A dedicated clock pin drives each general-purpose PLL. These clock lines are utilized when designing for speeds up to 420 Mbps. Tables 5–3 and 5–4 on page 5–19, respectively, show the available clocks in Stratix devices.

SERDES Bypass DDR Differential Signaling Receiver Operation

The SERDES bypass differential signaling receiver uses the Stratix device DDR input circuitry to receive high-speed serial data. The DDR input circuitry consists of a pair of shift registers used to capture the high-speed serial data, and a latch.

One register captures the data on the positive edge of the clock (generated by PLL) and the other register captures the data on the negative edge of the clock. Because the data captured on the negative edge is delayed by one-half of the clock cycle, it is latched before it interfaces with the system logic.

Figure 5–28 shows the DDR timing relationship between the incoming serial data and the clock. In this example, the `inclock` signal is running at half the speed of the incoming data. However, other combinations are also possible. Figure 5–29 shows the DDR input and the other modules used in a Flexible-LVDS receiver design to interface with the system logic.

Figure 5–28. ×2 Timing Relation between Incoming Serial Data & Clock

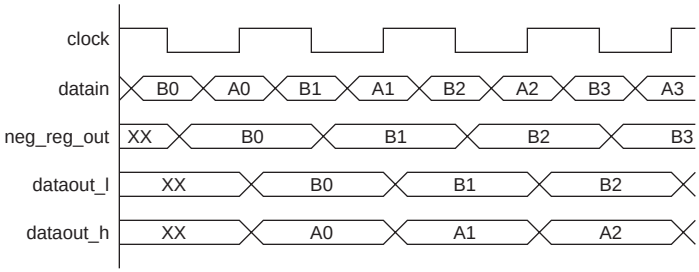
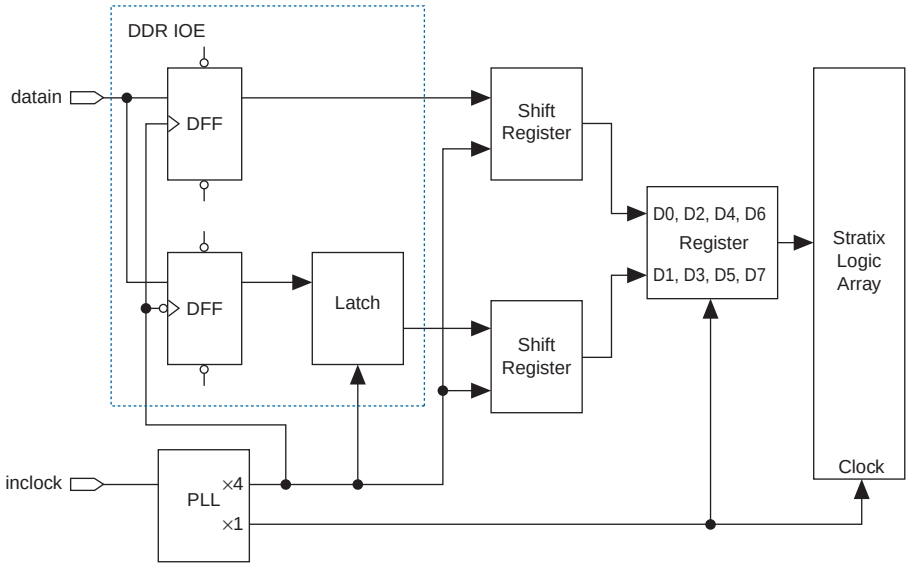


Figure 5–29. ×2 Data Rate Receiver Channel with Deserialization Factor of 8



SERDES Bypass DDR Differential Signaling Transmitter Operation

The ×2 differential signaling transmitter uses the Stratix device DDR output circuitry to transmit high-speed serial data. The DDR output circuitry consists of a pair of shift registers and a multiplexer. The shift registers capture the parallel data on the clock's rising edge (generated by the PLL), and a multiplexer transmits the data in sync with the clock. Figure 5–30 shows the DDR timing relation between the parallel data and the clock. In this example, the `inclock` signal is running at half the speed of the data. However, other combinations are possible. Figure 5–31 shows the DDR output and the other modules used in a ×2 transmitter design to interface with the system logic.

Figure 5–30. ×2 Timing Relation between Parallel Data & Clock

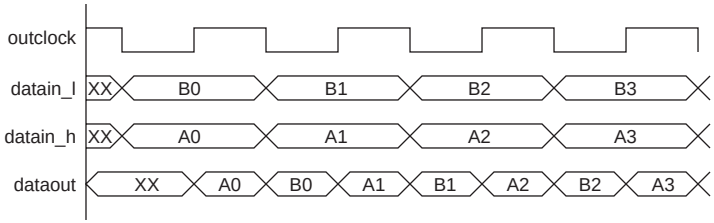
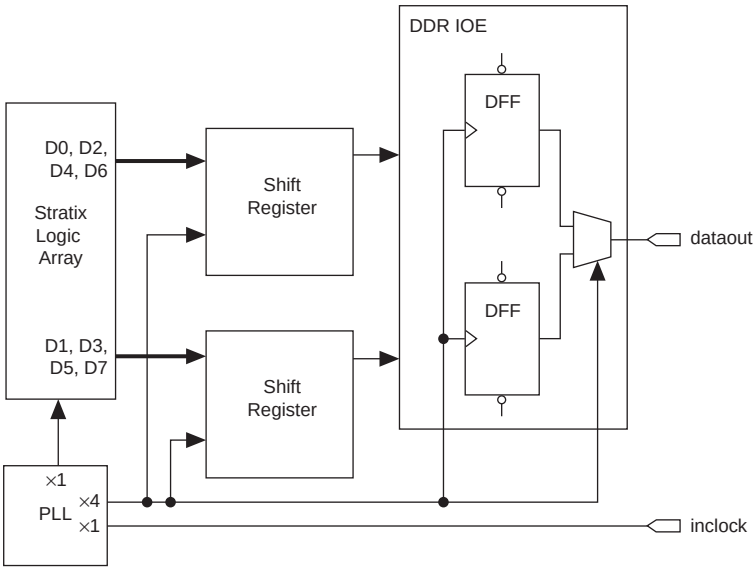


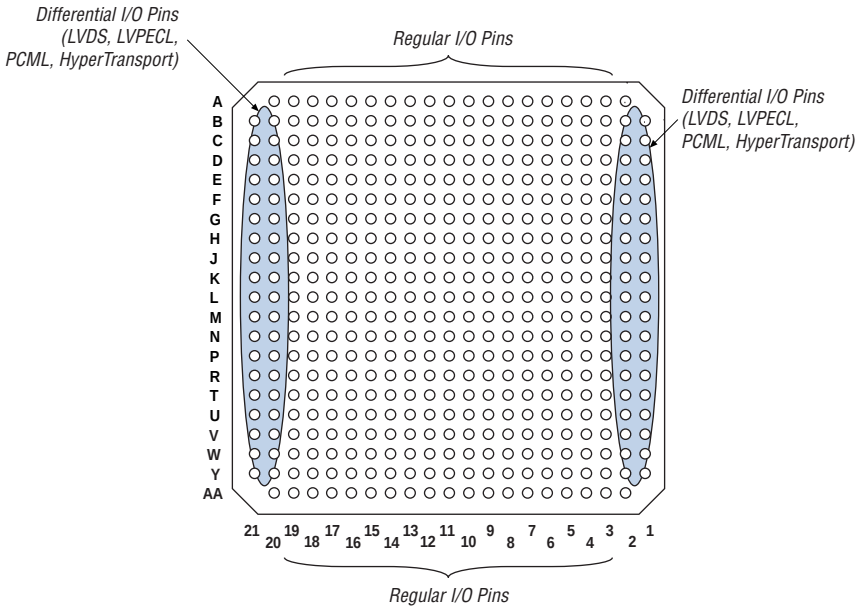
Figure 5–31. ×2 Data Rate Transmitter Channel with Serialization Factor of 8



High-Speed Interface Pin Locations

Stratix high-speed interface pins are located at the edge of the package to limit the possible mismatch between a pair of high-speed signals. Stratix devices have eight programmable I/O banks. Figure 5–32 shows the I/O pins and their location relative to the package.

Figure 5–32. Differential I/O Pin Locations



Differential I/O Termination

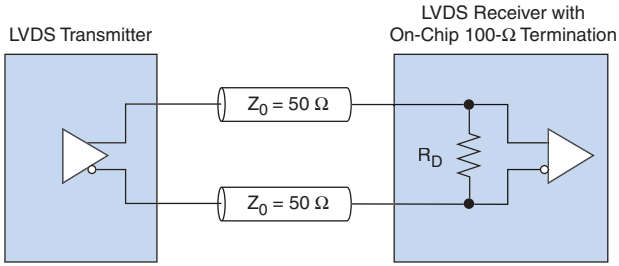
Stratix devices implement differential on-chip termination to reduce reflections and maintain signal integrity. On-chip termination also minimizes the number of external resistors required. This simplifies board design and places the resistors closer to the package, eliminating small stubs that can still lead to reflections.

R_D Differential Termination

Stratix devices support differential on-chip termination for the LVDS I/O standard. External termination is required on output pins for PCML transmitters. HyperTransport, LVPECL, and LVDS receivers require 100 ohm termination at the input pins. Figure 5–33 shows the device with differential termination for the LVDS I/O standard.

f For more information on differential on-chip termination technology, see the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter.

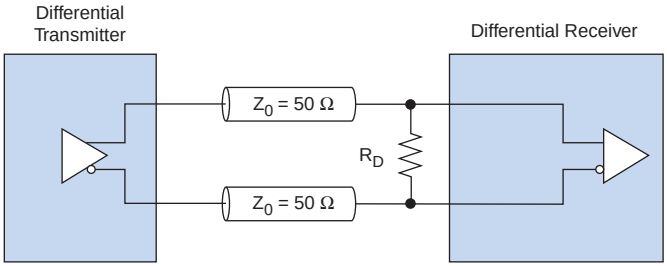
Figure 5–33. LVDS Differential On-Chip Termination



HyperTransport & LVPECL Differential Termination

HyperTransport and LVPECL I/O standards are terminated by an external 100-Ω resistor on the input pin. Figure 5–34 shows the device with differential termination for the HyperTransport or LVPECL I/O standard.

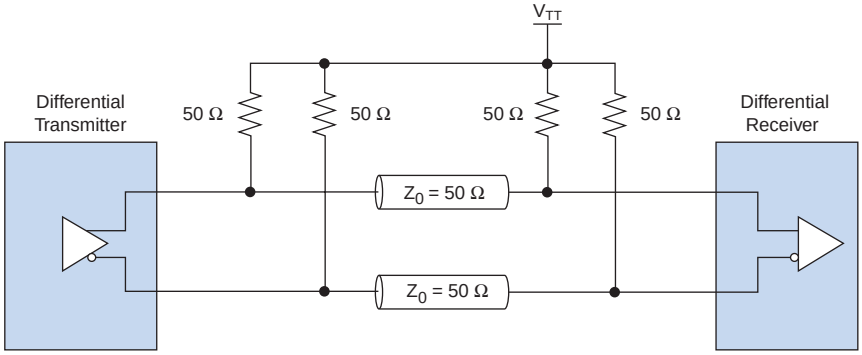
Figure 5–34. HyperTransport & LVPECL Differential Termination



PCML Differential Termination

The PCML I/O technology is an alternative to the LVDS I/O technology, and use an external voltage source (V_{TT}), a pair of 100-Ω resistors on the input side and a pair of 50-Ω resistors on the output side. Figure 5–35 shows the device with differential termination for PCML I/O standard.

Figure 5–35. PCML Differential Termination



Differential HSTL Termination

The HSTL Class I and II I/O standards require a 0.75-V V_{REF} and a 0.75-V V_{TT}. Figures 5–36 and 5–37 show the device with differential termination for HSTL Class I and II I/O standard.

Figure 5–36. Differential HSTL Class I Termination

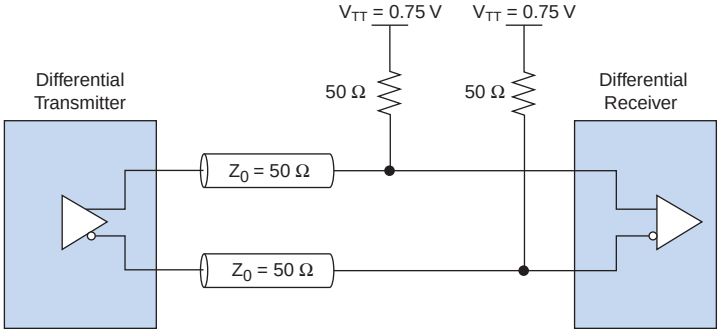
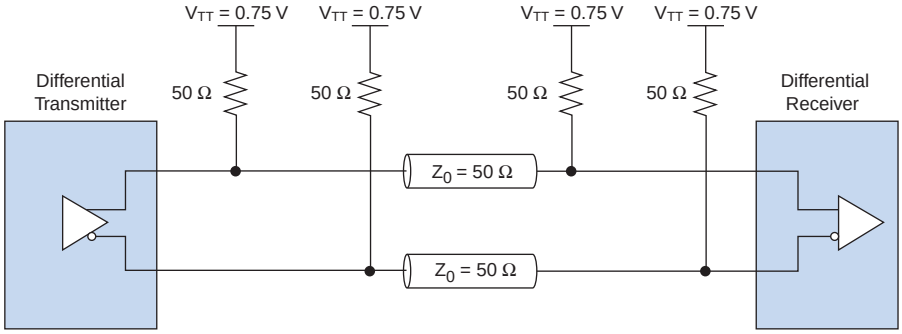


Figure 5–37. Differential HSTL Class II Termination



Differential SSTL-2 Termination

The SSTL-2 Class I and II I/O standards require a 1.25-V V_{REF} and a 1.25-V V_{TT}. Figures 5–37 and 5–38 show the device with differential termination for SSTL-2 Class I and II I/O standard.

Figure 5–38. Differential SSTL-2 Class I Termination

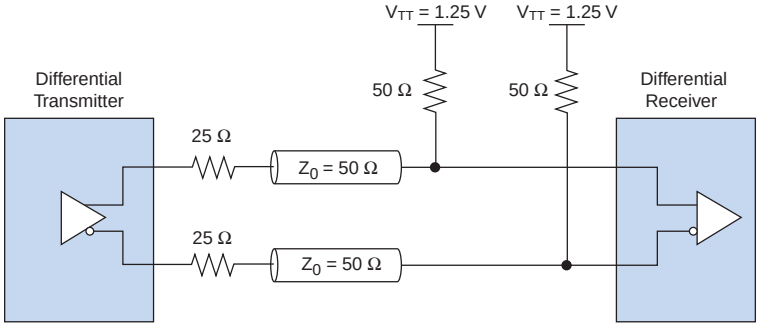
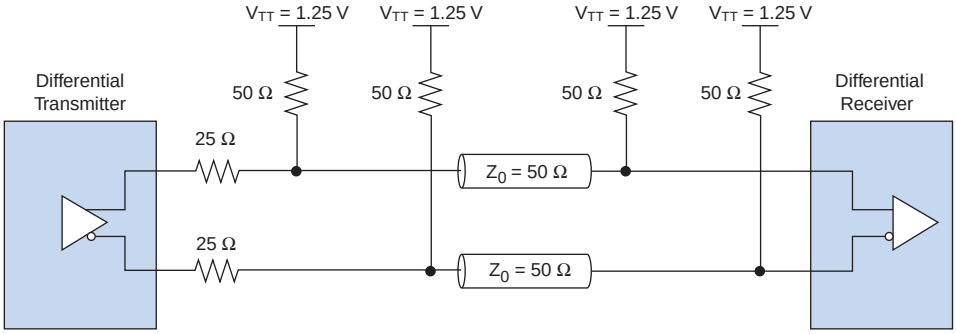


Figure 5–39. Differential SSTL-2 Class II Termination



Board Design Consideration

This section is a brief explanation of how to get the optimal performance from the Stratix high-speed I/O block and ensure first-time success in implementing a functional design with optimal signal quality. For more information on detailed board layout recommendation and I/O pin terminations see *AN 224: High-Speed Board Layout Guidelines*.

You must consider the critical issues of controlled impedance of traces and connectors, differential routing, and termination techniques to get the best performance from the IC. For more information, use this chapter and the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1*.

The Stratix high-speed module generates signals that travel over the media at frequencies as high as 840 Mbps. Board designers should use the following general guidelines:

- Baseboard designs on controlled differential impedance. Calculate and compare all parameters such as trace width, trace thickness, and the distance between two differential traces.
- Place external reference resistors as close to receiver input pins as possible.
- Use surface mount components.
- Avoid 90° or 45° corners.
- Use high-performance connectors such as HS-3 connectors for backplane designs. High-performance connectors are provided by Teradyne Corp **(www.teradyne.com)** or Tyco International Ltd. (**www.tyco.com**).
- Design backplane and card traces so that trace impedance matches the connector's and/or the termination's impedance.
- Keep equal number of vias for both signal traces.

- Create equal trace lengths to avoid skew between signals. Unequal trace lengths also result in misplaced crossing points and system margins as the TCCS value increases.
- Limit vias because they cause discontinuities.
- Use the common bypass capacitor values such as 0.001 µF, 0.01 µF, and 0.1 µF to decouple the fast PLL power and ground planes.
- Keep switching TTL signals away from differential signals to avoid possible noise coupling.
- Do not route TTL clock signals to areas under or above the differential signals.

Software Support

This section provides information on using the Quartus II software to create Stratix designs with LVDS transmitters or receivers. You can use the `altlvds` megafunction in the Quartus II software to implement the SERDES circuitry. You must bypass the SERDES circuitry in ×1 and ×2 mode designs and use the `altddio` megafunction to implement the deserialization instead. You can use either the logic array or the M512 RAM blocks closest to the differential pins for deserialization in SERDES bypass mode.

Differential Pins in Stratix

Stratix device differential pins are located in I/O banks 1, 2, 5, and 6 (see Figure 5–1 on page 5–2). Each bank has differential transmitter and differential receiver pin pairs. You can use each differential transmitter pin pair as either a differential data pin pair or a differential clock pin pair because Stratix devices do not have dedicated LVDS `tx_outclock` pin pairs. The differential receiver pin pairs can only function as differential data pin pairs. You can use these differential pins as regular user I/O pins when not used as differential pins. When using differential signaling in an I/O bank, you cannot place non-differential output or bidirectional pads within five I/O pads of either side of the differential pins to avoid a decrease in performance on the LVDS signals.

You only need to make assignments to the positive pin of the pin-pair. The Quartus II software automatically reserves and makes the same assignment to the negative pin. If you do not assign any differential I/O standard to the differential pins, the Quartus II software sets them as LVDS differential pins during fitting, if the design uses the SERDES circuitry. Additionally, if you bypass the SERDES circuitry, you can still use the differential pins by assigning a differential I/O standard to the pins in the Quartus II software. However, when you bypass the SERDES circuitry in the ×1 and ×2 mode, you must assign the correct differential I/O standard to the associated pins in the Assignment Organizer. For more information on how to use the Assignment Organizer, see the Quartus II On-Line Help.

Stratix devices can drive the PLL_LOCK signal to both output pins and internal logic. As a result, you do not need a dedicated LOCK pin for your PLLs. In addition, there is only one PLL_ENABLE pin that enables all the PLLs on the device, including the fast PLLs. You must use either the LVTTL or LVCMOS I/O standard with this pin.

Table 5–9 displays the LVDS pins in Stratix devices.

Table 5–9. LVDS Pin Names

Pin Names	Functions
DIFFIO_TX#p	Transmitter positive data or output clock pin
DIFFIO_TX#n	Transmitter negative data or output clock pin
DIFFIO_RX#p	Receiver positive data pin
DIFFIO_RX#n	Receiver negative data pin
FPLLCLK#p	Positive input clock pin to the corner fast PLLs *(1)*, *(2)*
FPLLCLK#n	Negative input clock pin to the corner fast PLLs *(1)*, *(2)*
CLK#p	Positive input clock pin *(2)*
CLK#n	Negative input clock pin *(2)*

Notes to Table 5–9:

(1) The FPLLCLK pin-pair is only available in EP1S30, EP1S40, EP1S60, EP1S80 devices.

(2) Either a FPLLCLK pin or a CLK pin can drive the corner fast PLLs (PLL7, PLL8, PLL9, and PLL10) when used for general purpose. CLK pins cannot drive these fast PLLs in high-speed differential I/O mode.

Fast PLLs

Each fast PLL features a multiplexed input path from a global or regional clock net. A clock pin or an output from another PLL in the device can drive the input path. The input clock for PLLs used to clock receiver the rx_inclock port on the altlvds_rx megafunction must be driven by a dedicated clock pin (CLK[3..0,8..11]) or the corner pins that clock the corner PLLs (FPLL[10..7]CLK). EP1S10, EP1S20, and EP1S25 devices have a total of four fast PLLs located in the center of both sides of the device (see Figure 5–16 on page 5–23). EP1S30 and larger devices have two additional fast PLLs per side at the top and bottom corners of the device. As shown in Figure 5–17 on page 5–24, the corner fast PLL shares an I/O bank with the closest center fast PLL (e.g., PLLs 1 and 7 share an I/O bank). The maximum input clock frequency for enhanced PLLs is 684 MHz and 717 MHz for fast PLLs.

f For more information on Stratix PLLs, see the *General-Purpose PLLs in Stratix & Stratix GX Devices* chapter.

One fast PLL can drive the 20 transmitter channels and 20 receiver channels closest to it with data rates of up to 840 Mbps. Wire-bond packages support a data rate of 624 Mbps. The corner fast PLLs in EP1S80 devices support data rates of up to 840 Mbps. See Tables 5–10 through 5–14 for the number of high-speed differential channels in a particular Stratix device density and package.

Since the fast PLL drives the 20 closest differential channels, there are coverage overlaps in the EP1S30 and larger devices that have two fast PLLs per I/O bank. In these devices, either the center fast PLL or the corner fast PLL can drive the differential channels in the middle of the I/O bank.

Fast PLLs can drive more than 20 transmitter and 20 receiver channels (see Tables 5–10 through 5–14 and Figures 5–16, and 5–17 for the number of channels each PLL can drive). In addition, the center fast PLLs can drive either one I/O bank or both I/O banks on the same side (left or right) of the device, while the corner fast PLLs can only drive the differential channels in its I/O bank. Neither fast PLL can drive the differential channels in the opposite side of the device.

The center fast PLLs can only drive two I/O at 840 Mbps. For example, EP1S20 device fast PLL 1 can drive all 33 differential channels on its side (17 channels from I/O bank 2 and 16 channels from I/O bank 1) running at 840 Mbps in 4× mode. When a center fast PLL drives the opposite bank on the same side of the device, the other center fast PLL cannot drive any differential channels on the device.

See Tables 5–10 through 5–14 for the maximum number of channels that one fast PLL can drive. The number of channels is also listed in the Quartus II software. The Quartus II software gives an error message if you try to compile a design exceeding the maximum number of channels.

f Additional high-speed DIFFIO pin information for Stratix devices is available in Volume 3 of the *Stratix Device Handbook*.

Table 5–10 shows the number of channels and fast PLLs in EP1S10, EP1S20, and EP1S25 devices. Tables 5–11 through 5–14 show this information for EP1S30, EP1S40, EP1S60, and EP1S80 devices.

Table 5–10. EP1S10, EP1S20 & EP1S25 Device Differential Channels (Part 1 of 2) Note (1)

Device	Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs			
					PLL 1	PLL 2	PLL 3	PLL 4
EP1S10	484-pin FineLine BGA	Transmitter *(2)*	20	840	5	5	5	5
				840 *(3)*	10	10	10	10
		Receiver	20	840	5	5	5	5
				840 *(3)*	10	10	10	10
	672-pin FineLine BGA 672-pin BGA	Transmitter *(2)*	36	624 *(4)*	9	9	9	9
				624 *(3)*	18	18	18	18
		Receiver	36	624 *(4)*	9	9	9	9
				624 *(3)*	18	18	18	18
	780-pin FineLine BGA	Transmitter *(2)*	44	840	11	11	11	11
				840 *(3)*	22	22	22	22
		Receiver	44	840	11	11	11	11
				840 *(3)*	22	22	22	22
EP1S20	484-pin FineLine BGA	Transmitter *(2)*	24	840	6	6	6	6
				840 *(3)*	12	12	12	12
		Receiver	20	840	5	5	5	5
				840 *(3)*	10	10	10	10
	672-pin FineLine BGA 672-pin BGA	Transmitter *(2)*	48	624 *(4)*	12	12	12	12
				624 *(3)*	24	24	24	24
		Receiver	50	624 *(4)*	13	12	12	13
				624 *(3)*	25	25	25	25
	780-pin FineLine BGA	Transmitter *(2)*	66	840	17	16	16	17
				840 *(3)*	33	33	33	33
		Receiver	66	840	17	16	16	17
				840 *(3)*	33	33	33	33

Table 5–10. EP1S10, EP1S20 & EP1S25 Device Differential Channels (Part 2 of 2) *Note (1)*

Device	Package	Transmitter/ Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs			
					PLL 1	PLL 2	PLL 3	PLL 4
EP1S25	672-pin FineLine BGA 672-pin BGA	Transmitter *(2)*	56	624 *(4)*	14	14	14	14
				624 *(3)*	28	28	28	28
		Receiver	58	624 *(4)*	14	15	15	14
				624 *(3)*	29	29	29	29
	780-pin FineLine BGA	Transmitter *(2)*	70	840	18	17	17	18
				840 *(3)*	35	35	35	35
		Receiver	66	840	17	16	16	17
				840 *(3)*	33	33	33	33
	1,020-pin FineLine BGA	Transmitter *(2)*	78	840	19	20	20	19
				840 *(3)*	39	39	39	39
		Receiver	78	840	19	20	20	19
				840 *(3)*	39	39	39	39

Notes to Table 5–10:

(1) The first row for each transmitter or receiver reports the number of channels driven directly by the PLL. The second row below it shows the maximum channels a PLL can drive if cross bank channels are used from the adjacent center PLL. For example, in the 484-pin FineLine BGA EP1S10 device, PLL 1 can drive a maximum of five channels at 840 Mbps or a maximum of 10 channels at 840 Mbps. The Quartus II software may also merge receiver and transmitter PLLs when a receiver is driving a transmitter. In this case, one fast PLL can drive both the maximum numbers of receiver and transmitter channels.

(2) The number of channels listed includes the transmitter clock output (`tx_outclock`) channel. If the design requires a DDR clock, it can use an extra data channel.

(3) These channels span across two I/O banks per side of the device. When a center PLL clocks channels in the opposite bank on the same side of the device it is called cross-bank PLL support. Both center PLLs can clock cross-bank channels simultaneously if, for example, PLL_1 is clocking all RX channels and PLL_2 is clocking all TX channels. You cannot have two adjacent PLLs simultaneously clocking cross-bank RX channels or two adjacent PLLs simultaneously clocking TX channels. Cross-bank allows for all receiver channels on one side of the device to be clocked on one clock while all transmitter channels on the device are clocked on the other center PLL. Crossbank PLLs are supported at full-speed, 840 Mbps. For wire-bond devices, the full-speed is 624 Mbps.

(4) These values show the channels available for each PLL without crossing another bank.

Table 5–11. EP1S30 Differential Channels *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
780-pin FineLine BGA	Transmitter *(4)*	70	840	18	17	17	18	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	35	35	35	35	*(6)*	*(6)*	*(6)*	*(6)*
	Receiver	66	840	17	16	16	17	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	33	33	33	33	*(6)*	*(6)*	*(6)*	*(6)*
956-pin FineLine BGA	Transmitter *(4)*	80 *(7)*	840	19	20	20	19	20	20	20	20
			840 *(5)*	39	39	39	39	20	20	20	20
	Receiver	80 *(7)*	840	20	20	20	20	19	20	20	19
			840 *(5)*	40	40	40	40	19	20	20	19
1,020-pin FineLine BGA	Transmitter *(4)*	80 (2) *(7)*	840	19 (1)	20	20	19 (1)	20	20	20	20
			840 *(5)*,*(8)*	39 (1)	39 (1)	39 (1)	39 (1)	20	20	20	20
	Receiver	80 (2) *(7)*	840	20	20	20	20	19 (1)	20	20	19 (1)
			840 *(5)*,*(8)*	40	40	40	40	19 (1)	20	20	19 (1)

Table 5–12. EP1S40 Differential Channels (Part 1 of 2) *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
780-pin FineLine BGA	Transmitter *(4)*	68	840	18	16	16	18	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	34	34	34	34	*(6)*	*(6)*	*(6)*	*(6)*
	Receiver	66	840	17	16	16	17	*(6)*	*(6)*	*(6)*	*(6)*
			840 *(5)*	33	33	33	33	*(6)*	*(6)*	*(6)*	*(6)*
956-pin FineLine BGA	Transmitter *(4)*	80	840	18	17	17	18	20	20	20	20
			840 *(5)*	35	35	35	35	20	20	20	20
	Receiver	80	840	20	20	20	20	18	17	17	18
			840 *(5)*	40	40	40	40	18	17	17	18

Table 5–12. EP1S40 Differential Channels (Part 2 of 2) *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
1,020-pin FineLine BGA	Transmitter *(4)*	80 (10) *(7)*	840	18 (2)	17 (3)	17 (3)	18 (2)	20	20	20	20
			840 *(5)*, *(8)*	35 (5)	35 (5)	35 (5)	35 (5)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	18 (2)	17 (3)	17 (3)	18 (2)
			840 *(5)*, *(8)*	40	40	40	40	18 (2)	17 (3)	17 (3)	18 (2)
1,508-pin FineLine BGA	Transmitter *(4)*	80 (10) *(7)*	840	18 (2)	17 (3)	17 (3)	18 (2)	20	20	20	20
			840 *(5)*, *(8)*	35 (5)	35 (5)	35 (5)	35 (5)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	18 (2)	17 (3)	17 (3)	18 (2)
			840 *(5)*, *(8)*	40	40	40	40	18 (2)	17 (3)	17 (3)	18 (2)

Table 5–13. EP1S60 Differential Channels (Part 1 of 2) *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
956-pin FineLine BGA	Transmitter *(4)*	80	840	12	10	10	12	20	20	20	20
			840 *(5)*, *(8)*	22	22	22	22	20	20	20	20
	Receiver	80	840	20	20	20	20	12	10	10	12
			840 *(5)*, *(8)*	40	40	40	40	12	10	10	12
1,020-pin FineLine BGA	Transmitter *(4)*	80 (12) *(7)*	840	12 (2)	10 (4)	10 (4)	12 (2)	20	20	20	20
			840 *(5)*, *(8)*	22 (6)	22 (6)	22 (6)	22 (6)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	12 (8)	10 (10)	10 (10)	12 (8)
			840 *(5)*, *(8)*	40	40	40	40	12 (8)	10 (10)	10 (10)	12 (8)

Table 5–13. EP1S60 Differential Channels (Part 2 of 2) *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*, *(3)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
1,508-pin FineLine BGA	Transmitter *(4)*	80 (36) *(7)*	840	12 (8)	10 (10)	10 (10)	12 (8)	20	20	20	20
			840 *(5)*,*(8)*	22 (18)	22 (18)	22 (18)	22 (18)	20	20	20	20
	Receiver	80 (36) *(7)*	840	20	20	20	20	12 (8)	10 (10)	10 (10)	12 (8)
			840 *(5)*,*(8)*	40	40	40	40	12 (8)	10 (10)	10 (10)	12 (8)

Table 5–14. EP1S80 Differential Channels (Part 1 of 2) *Note (1)*

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs *(2)*			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
956-pin FineLine BGA	Transmitter *(4)*	80 (40) *(7)*	840	10	10	10	10	20	20	20	20
			840 *(5)*,*(8)*	20	20	20	20	20	20	20	20
	Receiver	80	840	20	20	20	20	10	10	10	10
			840 *(5)*,*(8)*	40	40	40	40	10	10	10	10
1,020-pin FineLine BGA	Transmitter *(4)*	80 (12) *(7)*	840	10 (2)	10 (4)	10 (4)	10 (2)	20	20	20	20
			840 *(5)*,*(8)*	20 (6)	20 (6)	20 (6)	20 (6)	20	20	20	20
	Receiver	80 (10) *(7)*	840	20	20	20	20	10 (2)	10 (3)	10 (3)	10 (2)
			840 *(5)*,*(8)*	40	40	40	40	10 (2)	10 (3)	10 (3)	10 (2)

Table 5–14. EP1S80 Differential Channels (Part 2 of 2) Note (1)

Package	Transmitter /Receiver	Total Channels	Maximum Speed (Mbps)	Center Fast PLLs				Corner Fast PLLs (2)			
				PLL1	PLL2	PLL3	PLL4	PLL7	PLL8	PLL9	PLL10
1,508-pin FineLine BGA	Transmitter (4)	80 (72) (7)	840	10 (10)	10 (10)	10 (10)	10 (10)	20 (8)	20 (8)	20 (8)	20 (8)
			840 (5),(8)	20 (20)	20 (20)	20 (20)	20 (20)	20 (8)	20 (8)	20 (8)	20 (8)
	Receiver	80 (56) (7)	840	20	20	20	20	10 (14)	10 (14)	10 (14)	10 (14)
			840 (5),(8)	40	40	40	40	10 (14)	10 (14)	10 (14)	10 (14)

Notes to Tables 5–11 through 5–14.

(1) The first row for each transmitter or receiver reports the number of channels driven directly by the PLL. The second row below it shows the maximum channels a PLL can drive if cross bank channels are used from the adjacent center PLL. For example, in the 780-pin FineLine BGA EP1S30 device, PLL 1 can drive a maximum of 18 transmitter channels at 840 Mbps or a maximum of 35 transmitter channels at 840 Mbps. The Quartus II software may also merge transmitter and receiver PLLs when a receiver is driving a transmitter. In this case, one fast PLL can drive both the maximum numbers of receiver and transmitter channels.

(2) Some of the channels accessible by the center fast PLL and the channels accessible by the corner fast PLL overlap. Therefore, the total number of channels is not the addition of the number of channels accessible by PLLs 1, 2, 3, and 4 with the number of channels accessible by PLLs 7, 8, 9, and 10. For more information on which channels overlap, see the *Fast PLL to High-Speed I/O Connections* section in the relevant device pin table available on the web (**www.altera.com**).

(3) The corner fast PLLs in this device support a data rate of 840 Mbps for channels labeled "high" speed in the device pin tables.

(4) The numbers of channels listed include the transmitter clock output (`tx_outclock`) channel. You can use an extra data channel if you need a DDR clock.

(5) These channels span across two I/O banks per side of the device. When a center PLL clocks channels in the opposite bank on the same side of the device it is called cross-bank PLL support. Both center PLLs can clock cross-bank channels simultaneously if, for example, PLL_1 is clocking all receiver channels and PLL_2 is clocking all transmitter channels. You cannot have two adjacent PLLs simultaneously clocking cross-bank receiver channels or two adjacent PLLs simultaneously clocking transmitter channels. Cross-bank allows for all receiver channels on one side of the device to be clocked on one clock while all transmitter channels on the device are clocked on the other center PLL. Crossbank PLLs are supported at full-speed, 840 Mbps. For wire-bond devices, the full-speed is 624 Mbps.

(6) PLLs 7, 8, 9, and 10 are not available in this device.

(7) The number in parentheses is the number of slow-speed channels, guaranteed to operate at up to 462 Mbps. These channels are independent of the high-speed differential channels. For the location of these channels, see the *Fast PLL to High-Speed I/O Connections* section in the relevant device pin table available on the web (**www.altera.com**).

(8) See device pin-outs channels marked "high" speed are 840 Mbps and "low" speed channels are 462 MBps.

The Quartus II software may also merge transmitter and receiver PLLs when a receiver block is driving a transmitter block if the **Use Common PLLs for Rx and Tx** option is set for both modules. The Quartus II software does not merge the PLLs in multiple transmitter-only or multiple receiver-only modules fed by the same clock.

When you span two I/O banks using cross-bank support, you can route only two load enable signals total between the plls. When you enable `rx_data_align`, you use both `rxloadena` and `txloadena` of a PLL. That leaves no `loadena` for the second PLL.

The only way you can use the `rx_data_align` is if one of the following is true:

- The RX PLL is only clocking RX channels (no resources for TX)
- If all channels can fit in one I/O bank

LVDS Receiver Block

You only need to enter the input clock frequency, deserialization factor, and the input data rate to implement an LVDS receiver block. The Quartus II software then automatically sets the clock boost (*W*) factor for the receiver. In addition, you can also indicate the clock and data alignment for the receiver or add the `pll_enable`, `rx_data_align`, and `rx_locked` output ports. Table 5–15 explains the function of the available ports in the LVDS receiver block.

Table 5–15. LVDS Receiver Ports

Port Name	Direction	Function	Input Port Source/Output Port Destination
`rx_in[number_of_channels - 1..0]`	Input	Input data channel	Pin
`rx_inclock`	Input	Reference input clock	Pin or output from a PLL
`rx_pll_enable`	Input	Enables fast PLL	Pin *(1)*, *(2)*, *(3)*
`rx_data_align`	Input	Control for the data realignment circuitry	Pin or logic array *(1)*, *(3)*, *(4)*
`rx_locked`	Output	Fast PLL locked pin	Pin or logic array *(1)*, *(3)*
`rx_out[Deserialization_factor * number_of_channels -1..0]`	Output	De-serialized data	Logic array
`rx_outclock`	Output	Internal reference clock	Logic array

Notes to Table 5–15:

(1) This is an optional port.

(2) Only one `rx_pll_enable` pin is necessary to enable all the PLLs in the device.

(3) This is a non-differential pin.

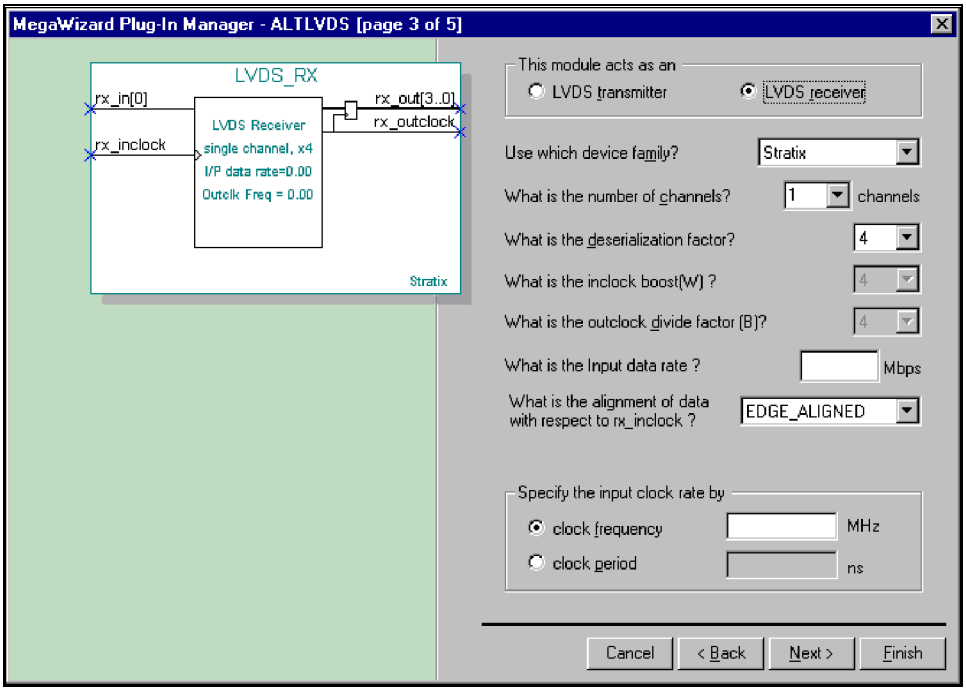
(4) See "Realignment Implementation" on page 5–28 for more information. For guaranteed performance and data alignment, you must synchronize `rx_data_align` with `rx_outclock`.

Use the `altlvds` MegaWizard Plug-In Manager to create an LVDS receiver block. The following sections explain the parameters available in the Plug-In Manager when creating an LVDS receiver block.

Page 3 of the altlvds_rx MegaWizard Plug-In Manager

On page 3 of the `altlvds` MegaWizard Plug-In Manager, you can choose to create either an LVDS transmitter or receiver. Depending on what you select, the MegaWizard Plug-In Manager provides you with different options. Figure 5–40 shows page 3 of the `altlvds` MegaWizard Plug-In Manager with options for creating an LVDS receiver.

Figure 5–40. Page 3 of the altlvds_rx MegaWizard Plug-In Manager



Number of Channels

The **What is the number of channels?** parameter specifies the number of receiver channels required and the width of `rx_out` port. To set a fast PLL to drive over 20 channels, type the required number in the Quartus II window instead of choosing a number from the drop-down menu, which only has selections of up to 20 channels.

Deserialization Factor

Use the **What is the deserialization factor?** parameter to specify the number of bits per channel. The Stratix LVDS receiver supports 4, 7, 8, and 10 for deserialization factor (*J*) values. Based on the factor specified, the Quartus II software determines the multiplication and / or division factor for the LVDS PLL to deserialize the data.

See Table 5–5 for the differential bit naming convention. The parallel data for the nth channel spans from the MSB (`rx_out` bit $[(J \times n) - 1]$) to the LSB (`rx_out` bit $[J \times (n - 1)]$), where *J* is the deserialization factor. The total width of the receiver `rx_out` port is equal to the number of channels multiplied by your deserialization factor.

Input Data Rate

The **What is the inclock boost(W)?** parameter sets the data rate coming into the receiver and is usually the deserialization factor (*J*) multiplied by the `inclock` frequency. This parameter's value must be larger than the input clock frequency and has a maximum input data rate of 840 Mbps for Stratix devices. You do not have to provide a value for the inclock boost (*W*) when designing with Stratix devices because the Quartus II software can calculate it automatically from this parameter and the clock frequency or clock period.

The `rx_outclock` frequency is $(W / J) \times$ input frequency. The parallel data coming out of the receiver has the same frequency as the `rx_outclock` port. The clock-to-data alignment of the parallel data output from the receiver depends on the **What is the alignment of data with respect to rx_inclock?** parameter.

Data Alignment with Clock

The **What is the alignment of data with respect to rx_inclock?** parameter adjusts the clock-to-data skew. For most applications, the data is source synchronous to the clock. However, there are applications where you must center-align the data with respect to the clock. You can use the **What is the alignment of data with respect to rx_inclock?** parameter to align the input data with respect to the `rx_inclock` port. The MegaWizard Plug-In automatically calculates the phase for the fast PLL outputs from the **What is the alignment of data with respect to rx_inclock?** parameter. This parameter's default value is `EDGE_ALIGNED`, and other values available from the pull-down menu are `EDGE_ALIGNED`, `CENTER_ALIGNED`, `45_DEGREES`, `135_DEGREES`, `180_DEGREES`, `225_DEGREES`, `270_DEGREES`, and `315_DEGREES`. `CENTER_ALIGNED` is the same as 90 degrees aligned and is useful for applications like HyperTransport technology.

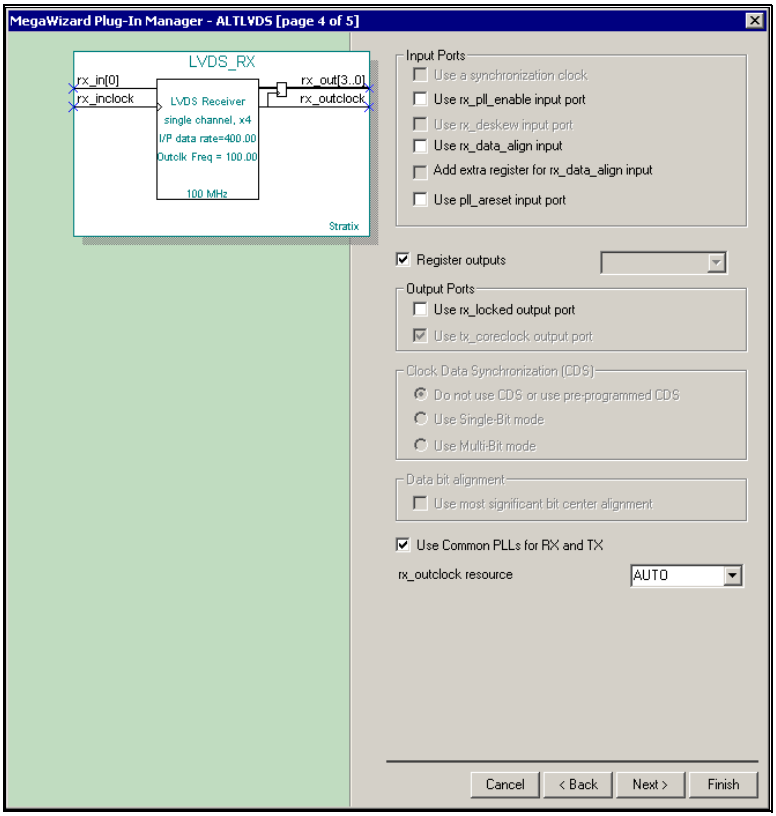
Clock Frequency or Clock Period

The fields in the **Specify the input clock rate by** box specify the input frequency or the period of the input clock going into the fast PLL. When using the same input clock to feed a transmitter and receiver simultaneously, the Quartus II software can use one fast PLL for both the transmitter and receiver.

Page 4 of the altlvds_rx MegaWizard Plug-In Manager

This section describes the parameters found on page 4 of the `altlvds_rx` MegaWizard Plug-In Manager (see Figure 5–41).

Figure 5–41. Page 4 of the altlvds_rx MegaWizard Plug-In Manager



Data Realignment

Check the **Use the "rx_data_align" input port** box within the **Input Ports** box to add the `rx_data_align` output port and enable the data realignment circuitry in Stratix SERDES. See "Receiver Data Realignment" on page 5–25 for more information. If necessary, you can create a state machine to send a pulse to the `rx_data_align` port to realign the data coming in the LVDS receiver. You need to assert the port for at least two clock cycles to enable the data realignment circuitry. Go to the Altera web site at **www.altera.com** for a sample design written in Verilog HDL.

For guaranteed performance when using data realignment, check the **Add Extra registers for rx_data_align input** box when using the `rx_data_align` port. The Quartus II software places one synchronization register in the LE closest to the `rx_data_align` port.

Register Outputs

Check the **Register outputs** box to register the receiver's output data. The register acts as the module's register boundary. If the module fed by the receiver does not have a register boundary for the data, turn this option on. The number of registers used is proportional to the deserialization factor (*J*). The Quartus II software places the synchronization registers in the LEs closest to the SERDES circuitry.

Use Common PLL for Both Transmitter & Receiver

Check the **Use Common PLLs for Rx and Tx** box to place both the LVDS transmitter and the LVDS receiver in the same Stratix device I/O bank. The Quartus II software allows the transmitter and receiver to share the same fast PLL when they use the same input clock. Although you must separate the transmitter and receiver modules in your design, the Quartus II software merges the fast PLLs when appropriate and give you the following message:

Receiver fast PLL <*lvds_rx PLL name*> and transmitter fast PLL <*lvds_tx PLL name*> are merged together

The Quartus II software provides the following message when it cannot merge the fast PLLs for the LVDS transmitter and receiver pair in the design:

Can't merge transmitter-only fast PLL <*lvds_tx PLL name*> and receiver-only fast PLL <*lvds_rx PLL name*>

rx_outclock Resource

You can use either the global or regional clock for the `rx_outclock` signal. If you select **Auto** in the Quartus II software, the tool uses any available lines.

LVDS Transmitter Module

The Quartus II software calculates the inclock boost (*W*) factor for the LVDS transmitter based on input data rate, input clock frequency, and the deserialization factor. In addition to setting the data and clock alignment, you can also set the outclock divide factor (*B*) for the transmitter output clock and add the `pll_enable`, `tx_locked`, and `tx_coreclock` ports. Table 5–16 explains the function of the available ports in the LVDS transmitter block.

Table 5–16. LVDS Transmitter Ports

Port Name	Direction	Function	Input port Source/Output port Destination
`tx_in[Deserialization_factor * number_of_channels - 1..0]`	Input	Input data	Logic array
`tx_inclock`	Input	Reference input clock	Pin or output clock from a PLL
`tx_pll_enable`	Input	Fast PLL enable	Pin *(1)*, *(2)*, *(3)*
`tx_out[number_of_channels - 1..0]`	Output	Serialized LVDS data signal	Pin
`tx_outclock`	Output	External reference clock	Pin
`tx_coreclock`	Output	Internal reference clock	Pin, logic array, or input clock to a fast PLL *(1)*
`tx_locked`	Output	Fast PLL locked pin	Pin or logic array *(1)*, *(2)*, *(3)*

Notes to Table 5–16:

(1) This is an optional port.

(2) Only one `tx_pll_enable` pin is necessary to enable all the PLLs in the device.

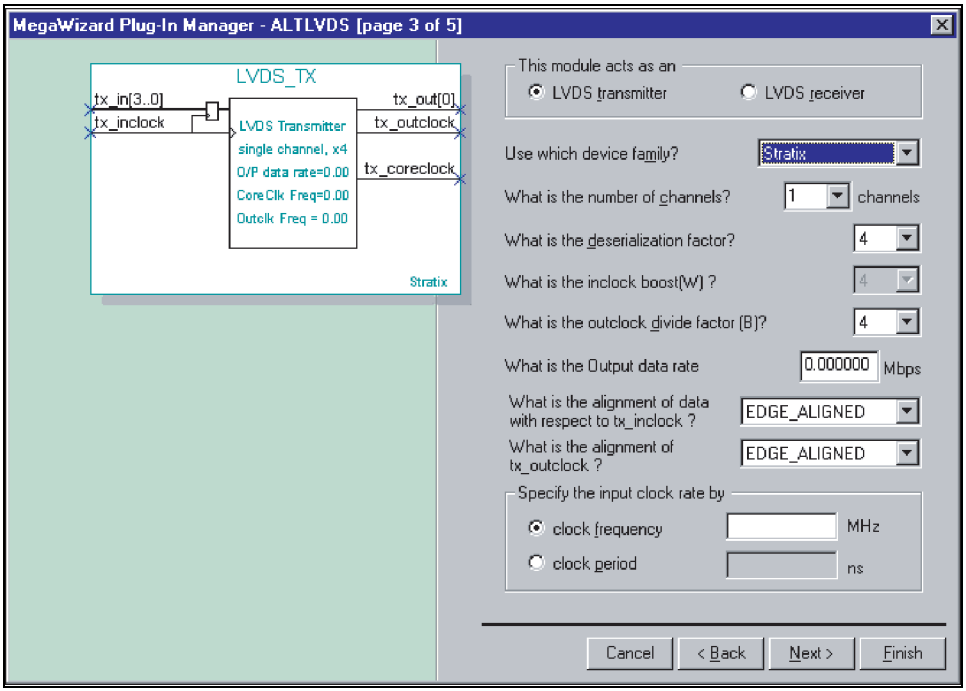
(3) This is a non-differential pin.

You can also use the `altlvds` MegaWizard Plug-In Manager to create an LVDS transmitter block. The following sections explain the parameters available in the Plug-In Manager when creating an LVDS transmitter block.

Page 3 of the altlvds_tx MegaWizard Plug-In Manager

This section describes the parameters found on page 3 of the `altlvds_tx` MegaWizard Plug-In Manager (see Figure 5–42).

Figure 5–42. Page 3 of the Transmitter altlvds MegaWizard Plug-In Manager



Number of Channels

The **What is the number of channels?** parameter specifies the number of transmitter channels required and the width of the `tx_in` port. You can have more than 20 channels in a transmitter or receiver module by typing in the required number instead of choosing a number from the drop down menu, which only has selections of up to 20 channels.

Deserialization Factor

The **What is the deserialization factor?** parameter specifies the number of bits per channel. The transmitter block supports deserialization factors of 4, 7, 8, and 10. Based on the factor specified, the Quartus II software determines the multiplication and / or division factor for the LVDS PLL in order to serialize the data.

Table 5–5 on page 5–32 lists the differential bit naming convention. The parallel data for the nth channel spans from the MSB (`rx_out` bit $[(J \times n) - 1]$) to the LSB (`rx_out` bit $[J \times (n - 1)]$), where J is the

deserialization factor. The total width of the `tx_in` port of the transmitter is equal to the number of channels multiplied by the deserialization factor.

Outclock Divide Factor

The **What is the Output data rate?** parameter specifies the ratio of the `tx_outclock` frequency compared to the data rate. The default value for this parameter is the value of the deserialization factor parameter. The `tx_outclock` frequency is equal to [*W* / *B*] x input clock frequency. There is also an optional `tx_coreclock` port which has the same frequency as the [*W* / *J*] × input frequency.

The outclock divide factor is useful for applications that do not require the data rate to be the same as the clock frequency. For example, HyperTransport technology uses a half-clock data rate scheme where the clock frequency is half the data rate. Table 5–17 shows the supported outclock divide factor for a given deserialization factor.

Table 5–17. Deserialization Factor (J) vs. Outclock Divide Factor (B)

Deserialization Factor (*J*)	Outclock Divide Factor (*B*)
4	1, 2, 4
7	1, 7*(1)*
8	1, 2, 4, 8
10	1, 2, 10

Note to Table 5–17:
(1) The clock does not have a 50% duty cycle when b=7 in x7 mode.

Output Data Rate

The **What is the Output data rate** parameter specifies the data rate out of the fast PLL and determines the input clock boost / multiplication factor needed for the transmitter. This parameter must be larger than the input clock frequency and has a maximum rate of 840 Mbps for Stratix devices. The input clock boost factor (*W*) is the output data rate divided by the input clock frequency. The Stratix SERDES circuitry supports input clock boost factors of 4, 7, 8, or 10. The maximum output data rate is 840 Mbps, while the clock has a maximum output of 500 MHz.

Data Alignment with Clock

Use the **What is the alignment of data with respect to tx_inclock?** parameter and the **What is the alignment of tx_outclock?** to align the input and output data, respectively, with the clock. For most applications, the data is edge-aligned with the clock. However, there are applications where the data must be center-aligned with respect to the clock. With

Stratix devices, you can align the input data with respect to the `tx_inclock` port and align the output data with respect to the `tx_outclock` port. The MegaWizard Plug-In Manager uses the alignment of input and output data to automatically calculate the phase for the fast PLL outputs. Both of these parameters default to `EDGE_ALIGNED`, and other values are `CENTER_ALIGNED`, `45_DEGREES`, `135_DEGREES`, `180_DEGREES`, `225_DEGREES`, `270_DEGREES`, and `315_DEGREES`. `CENTER_ALIGNED` is the same as 180 degrees aligned and is required for the HyperTransport technology I/O standard.

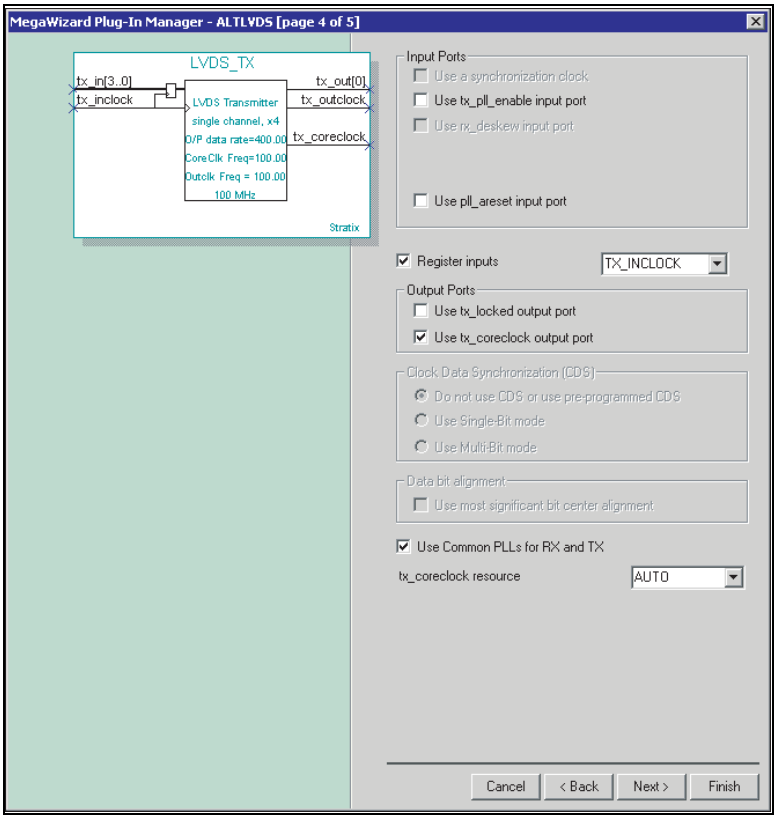
Clock Frequency & Clock Period

The fields in the **Specify the input clock rate by** box specify either the frequency or the period of the input clock going into the fast PLL. However, you cannot specify both. If your design uses the same input clock to feed a transmitter and a receiver module simultaneously, the Quartus II software can merge the fast PLLs for both the transmitter and receiver when the **Use common PLLs for Tx & Rx** option is turned on.

Page 4 of the altlvds_tx MegaWizard Plug-In Manager

This section describes the parameters found on page 4 of the `altlvds_tx` MegaWizard Plug-In Manager (see Figure 5–43).

Figure 5–43. Page 4 of the Transmitter altlvds MegaWizard Plug-In Manager



Registered Inputs

Check the **Register inputs** box if the input data to the transmitter is not registered just before it feeds the transmitter module. You can choose either `tx_clkin` or `tx_coreclk` to clock the transmitter data (`tx_in[]`) signal. This serves as the register boundary. The number of registers used is proportional to the deserialization factor (*J*). The Quartus II software places the synchronization registers with the LEs in the same row and closest to the SERDES circuitry.

Use Common PLL for Transmitter & Receiver

Check the **Use Common PLLs for Rx and Tx** box to place both the LVDS transmitter and receiver in the same I/O bank in Stratix devices. The Quartus II software also allows the transmitter and receiver to share the PLL when the same input clock is used for both. Although you must

separate the transmitter and receiver in your design, the Quartus II software merges the fast PLLs when appropriate and gives you the following message:

Receiver fast PLL <lvds_rx pll name> and transmitter fast PLL <lvds_tx pll name> are merged together

The Quartus II software gives the following message when it cannot merge the fast PLLs for the LVDS transmitter and receiver pair in the design:

```
Can't merge transmitter-only fast PLL
<lvds_tx pll name> and receiver-only fast PLL
<lvds_rx pll name>
```

tx_outclock Resource

You can use either the global or regional clock for the `tx_outclock` signal. If you select **Auto** in the Quartus II software, the tool uses any available lines.

SERDES Bypass Mode

You can bypass the SERDES block if your data rate is less than 624 Mbps, and you must bypass the SERDES block for the ×1 and ×2 LVDS modules.

Since you cannot route the fast PLL output to an output pin, you must create additional DDR I/O circuitry for the transmitter clock output. To create an ×*J* transmitter output clock, instantiate an `alt_ddio` megafunction clocked by the ×*J* clock with `datain_h` connected to V_{CC} and `datain_l` connected to `GND`.

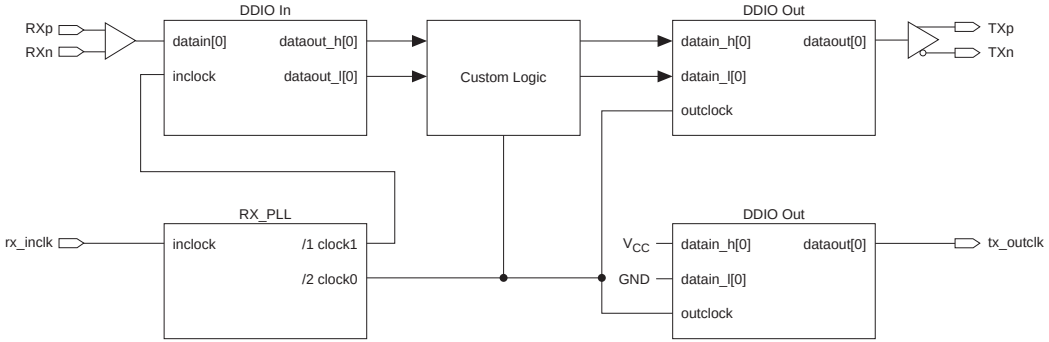
×1 Mode

For ×1 mode, you only need to specify the I/O standard of the pins to tell the Quartus II software that you are using differential signaling. However, Altera recommends using the DDRIO circuitry when the input or output data rate is higher than 231 Mbps. The maximum output clock frequency for ×1 mode is 420 MHz.

×2 Mode

You must use the DDRIO circuitry for ×2 mode. The Quartus II software provides the `altddio_in` and `altddio_out` megafunctions to use for ×2 receiver and ×2 transmitter, respectively. The maximum data rate in ×2 mode is 624 Mbps. Figure 5–44 shows the schematic for using DDR circuitry in ×2 mode.

Figure 5–44. LVDS x2 Mode Schematic Using DDR I/O Circuitry



The transmitter output clock requires extra DDR output circuitry that has the input high and input low connected to V_{CC} and GND respectively. The output clock frequency is the same as the input frequency of the DDR output circuitry.

Other Modes

For other modes, you can still to use the DDR circuitry for better frequency performance. You can use either the LEs or the M512 RAM block for the deserialization.

M512 RAM Block as Serializer/Deserializer Interface

In addition to using the DDR circuitry and the M512 RAM block, you need two extra counters per memory block to provide the address for the memory: a fast counter powering up at 0 and a slow counter powering up at 2. The M512 RAM block is configured as a simple dual-port memory block, where the read enable and the write enable signals are always tied high. Figures 5–45 and 5–46 show the block diagram for the SERDES bypass receiver and SERDES bypass transmitter, respectively.

Figure 5–45. SERDES Bypass LVDS Receiver Using M512 RAM Block as the Deserializer

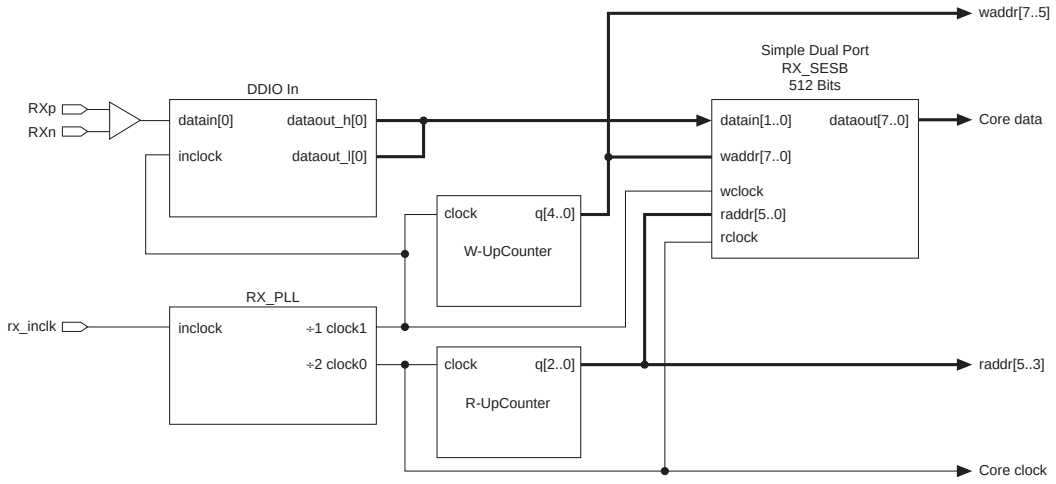
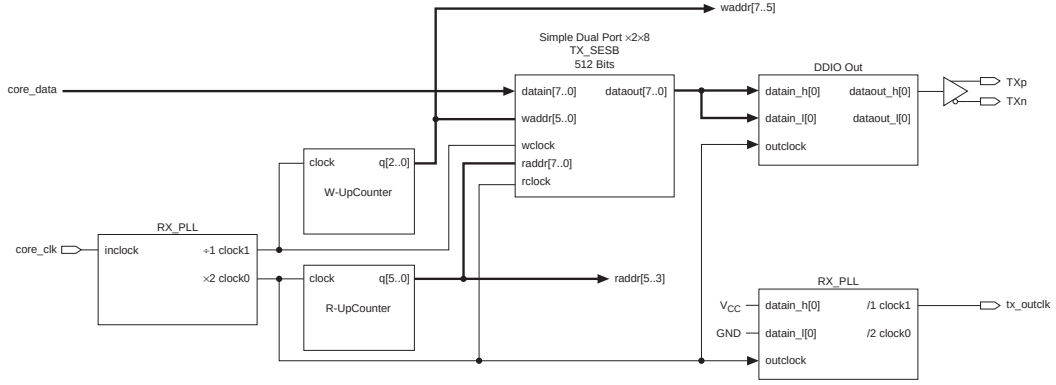


Figure 5–46. SERDES Bypass LVDS Transmitter Using M512 RAM Block as Deserializer



For the transmitter, the read counter is the fast counter and the write counter is the slow counter. For the receiver, the write counter is the fast counter and the read counter is the slow counter. Tables 5–18 and 5–19 provide the address counter configurations for the transmitter and the receiver, respectively.

Table 5–18. Address Counters for SERDES Bypass LVDS Receiver

M512 Mode	Deserialization Factor	Write Up-Counter (Fast Counter)		Read Up-Counter (Slow Counter)		Invalid Initial Cycles	
		Width	Starts at	Width	Starts at	Write	Read
×2×4	4	4	0	3	2	12	6
×2×8	8	5	0	3	2	24	6
×4×16	8	5	0	3	2	24	6
×2×16	16	6	0	3	2	48	6

Table 5–19. Address Counters for SERDES Bypass LVDS Transmitter

M512 Mode	Deserialization Factor	Write Up-Counter (Fast Counter)		Read Up-Counter (Slow Counter)		Invalid Initial Cycles	
		Width	Starts at	Width	Starts at	Write	Read
×2×4	4	4	0	3	2	2	4
×2×8	8	5	0	3	2	2	8
×4×16	8	5	0	3	2	2	8
×2×16	16	6	0	3	2	2	16

In different M512 memory configurations, the counter width is smaller than the address width, so you must ground some of the most significant address bits. Table 5–20 summarizes the address width, the counter width, and the number of bits to be grounded.

Table 5–20. Address & Counter Width

M512 Mode	Write Counter Width	Read Counter Width	Write Address Width	Read Address Width	Number of Grounded Bits	
					Write Address	Read Address
×2×4	4	3	8	7	4	4
×2×8	5	3	8	6	3	3
×4×16	6	3	7	5	1	2
×2×16	5	3	8	5	3	2

Logic Array as Serializer/Deserializer Interface

The design can use the `lpm_shift_reg` megafunction instead of a simple dual port memory block to serialize / deserialize data. The receiver requires an extra flip-flop clocked by the slow clock to latch on to the deserialized data. The transmitter requires a counter to generate the enable signal for the shift register to indicate the times to load and serialize the data. Figures 5–47 and 5–48 show the schematic of the ×8 LVDS receiver and ×8 LVDS transmitter, respectively, with the logic array performing the deserialization.

This scheme can also be used for APEX II and Mercury device flexible LVDS solutions.

Figure 5–47. SERDES Bypass LVDS Receiver with Logic Array as Deserializer

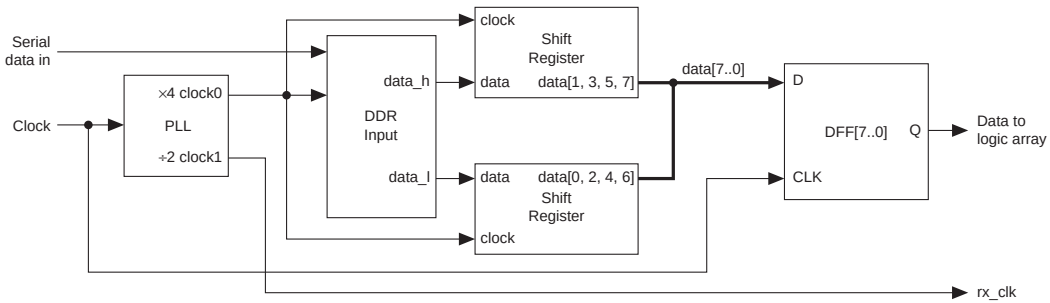
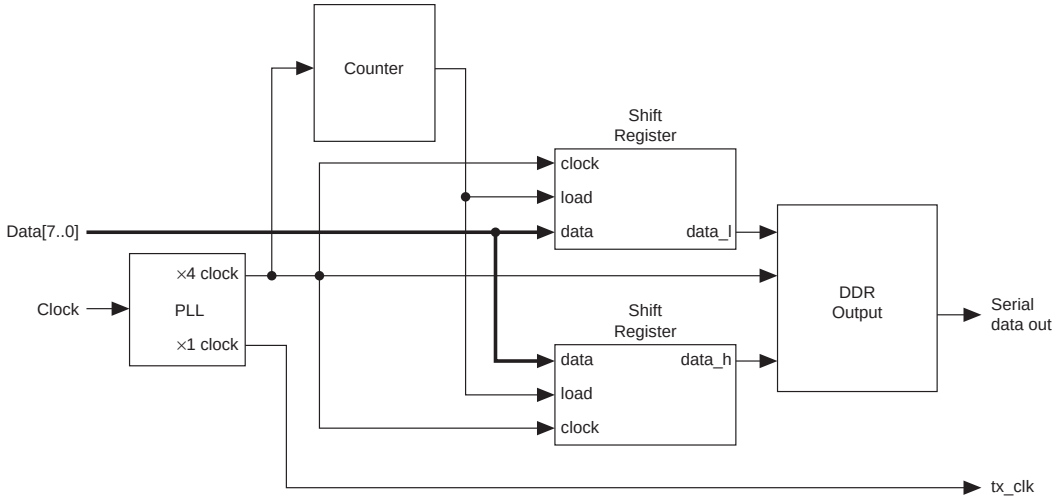


Figure 5–48. SERDES Bypass LVDS Transmitter with Logic Array as Deserializer



Summary

The Stratix device family of flexible, high-performance, high-density PLDs delivers the performance and bandwidth necessary for complex system-on-a-programmable-chip (SOPC) solutions. Stratix devices support multiple I/O protocols to interface with other devices within the system. Stratix devices can easily implement processing-intensive data-path functions that are received and transmitted at high speeds. The Stratix family of devices combines a high-performance enhanced PLD architecture with dedicated I/O circuitry in order to provide I/O standard performances of up to 840 Mbps.

Section IV. Digital Signal Processing (DSP)

This section provides information for design and optimization of digital signal processing (DSP) functions and arithmetic operations in the on-chip DSP blocks.

It contains the following chapters:

- Chapter 6, DSP Blocks in Stratix & Stratix GX Devices
- Chapter 7, Implementing High Performance DSP Functions in Stratix & Stratix GX Devices

Revision History

The table below shows the revision history for Chapters 6 and 7.

Chapter	Date/Version	Changes Made
6	July 2005, v2.2	• Changed *Stratix GX FPGA Family* data sheet reference to *Stratix GX Device Handbook, Volume 1*.
	September 2004, v2.1	• Updated "Software Support" on page 6–28. • Deleted "Quartus II DSP Megafunctions" section. It was replaced by the updated "Software Support" on page 6–28 • Replaced references to AN 193 and AN 194 with a new reference on page 6–28.
	July 2003, v2.0	• Minor content change.
	April 2003, v1.0	• No new changes in *Stratix Device Handbook* v2.0.
7	September 2004, v1.1	• Corrected spelling error.
	April 2003, v1.0	• No new changes in *Stratix Device Handbook* v2.0.

6. DSP Blocks in Stratix & Stratix GX Devices

S52006-2.2

Introduction

Traditionally, designers had to make a trade-off between the flexibility of off-the-shelf digital signal processors and the performance of custom-built devices. Altera® Stratix® and Stratix GX devices eliminate the need for this trade-off by providing exceptional performance combined with the flexibility of programmable logic devices (PLDs). Stratix and Stratix GX devices have dedicated digital signal processing (DSP) blocks, which have high-speed parallel processing capabilities, that are optimized for DSP applications. DSP blocks are ideal for implementing DSP applications that need high data throughput.

The most commonly used DSP functions are finite impulse response (FIR) filters, complex FIR filters, infinite impulse response (IIR) filters, fast Fourier transform (FFT) functions, discrete cosine transform (DCT) functions, and correlators. These functions are the building blocks for more complex systems such as wideband code division multiple access (W-CDMA) basestations, voice over Internet protocol (VoIP), and high-definition television (HDTV).

Although these functions are complex, they all use similar building blocks such as multiply-adders and multiply-accumulators. Stratix and Stratix GX DSP blocks combine five arithmetic operations—multiplication, addition, subtraction, accumulation, and summation—to meet the requirements of complex functions and to provide improved performance.

This chapter describes the Stratix and Stratix GX DSP blocks, and explains how you can use them to implement high-performance DSP functions. It addresses the following topics:

- Architecture
- Operational Modes
- Software Support

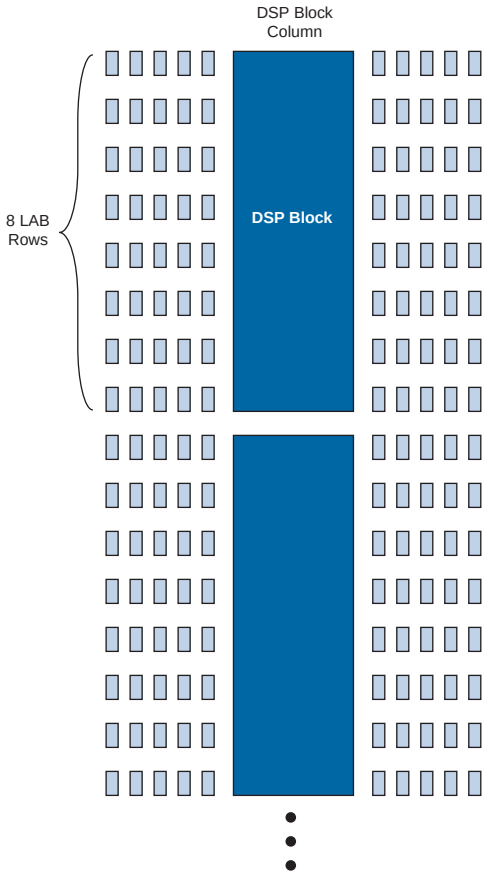
f See the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* and the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1* for more information on Stratix and Stratix GX devices, respectively.

DSP Block Overview

Each Stratix and Stratix GX device has two columns of DSP blocks that efficiently implement multiplication, multiply accumulate (MAC), and filtering functions. Figure 6–1 shows one of the columns with surrounding LAB rows. You can configure each DSP block to support:

- Eight 9 × 9 bit multipliers
- Four 18 × 18 bit multipliers
- One 36 × 36 bit multiplier

Figure 6–1. DSP Blocks Arranged in Columns



The multipliers can then feed an adder or an accumulator block, depending on the DSP block operational mode. Additionally, you can use the DSP block input registers as shift registers to implement applications such as FIR filters efficiently. The number of DSP blocks per column

increases with device density. Tables 6–1 and 6–2 describe the number of DSP blocks in each Stratix and Stratix GX device, respectively, and the multipliers that you can implement.

Table 6–1. Number of DSP Blocks in Stratix Devices Note (1)

Device	DSP Blocks	9 × 9 Multipliers	18 × 18 Multipliers	36 × 36 Multipliers
EP1S10	6	48	24	6
EP1S20	10	80	40	10
EP1S25	10	80	40	10
EP1S30	12	96	48	12
EP1S40	14	112	56	14
EP1S60	18	144	72	18
EP1S80	22	176	88	22

Table 6–2. Number of DSP Blocks in Stratix GX Devices Note (1)

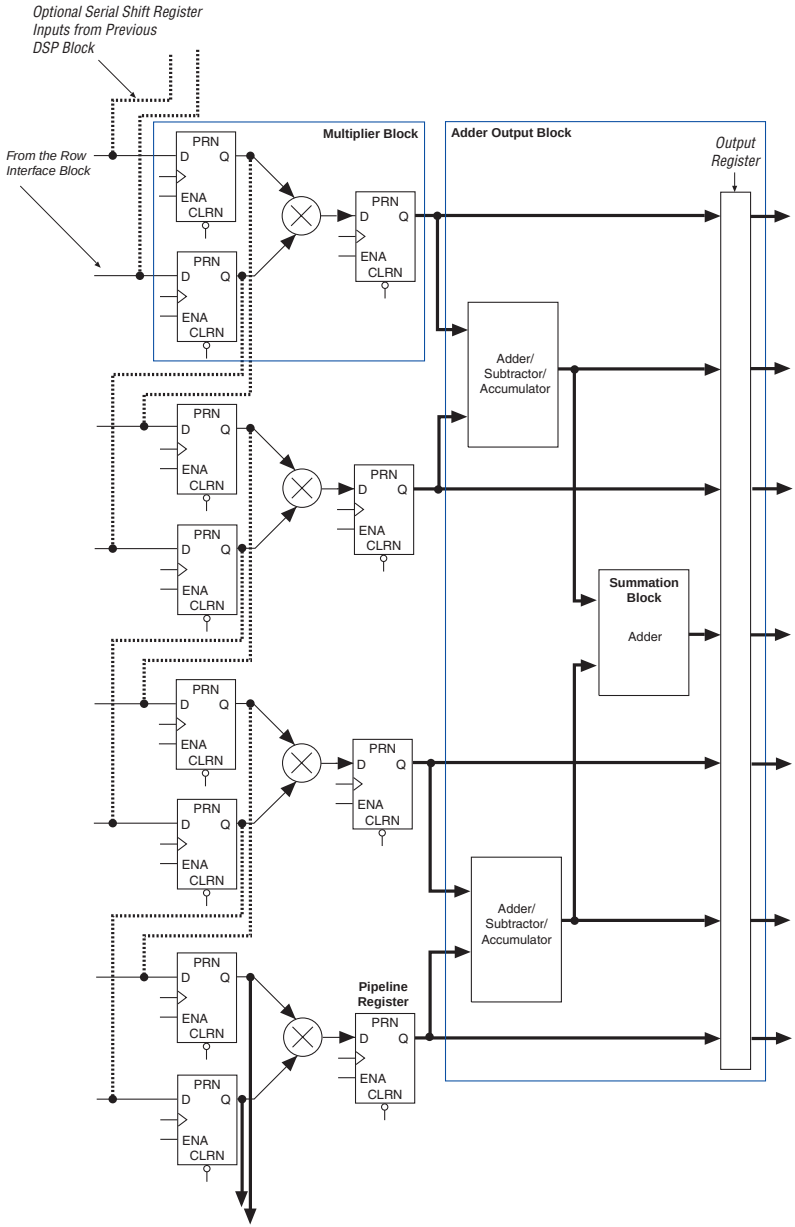
Device	DSP Blocks	9 × 9 Multipliers	18 × 18 Multipliers	36 × 36 Multipliers
EP1SGX10C	6	48	24	6
EP1SGX10D	6	48	24	6
EP1SGX25C	10	80	40	10
EP1SGX25D	10	80	40	10
EP1SGX25F	10	80	40	10
EP1SGX40D	14	112	56	14
EP1SGX40G	14	112	56	14

Note to Tables 6–1 and 6–2:

(1) Each device has either the number of 9 × 9-, 18 × 18-, or 36 × 36-bit multipliers shown.The total number of multipliers for each device is not the sum of all the multipliers.

Figure 6–2 shows the DSP block operating as an 18 × 18 multiplier.

Figure 6–2. DSP Block in 18 × 18 Mode



Architecture

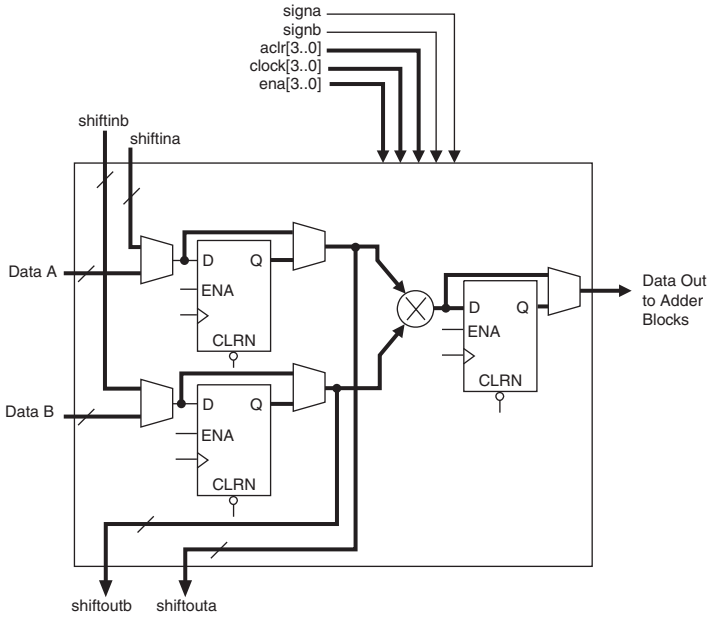
The DSP block consists of the following elements:

- A multiplier block
- An adder / subtractor / accumulator block
- A summation block
- An output interface
- Output registers
- Routing and control signals

Multiplier Block

Each multiplier block has input registers, a multiplier stage, and a pipeline register. See Figure 6–3.

Figure 6–3. Multiplier Block Architecture



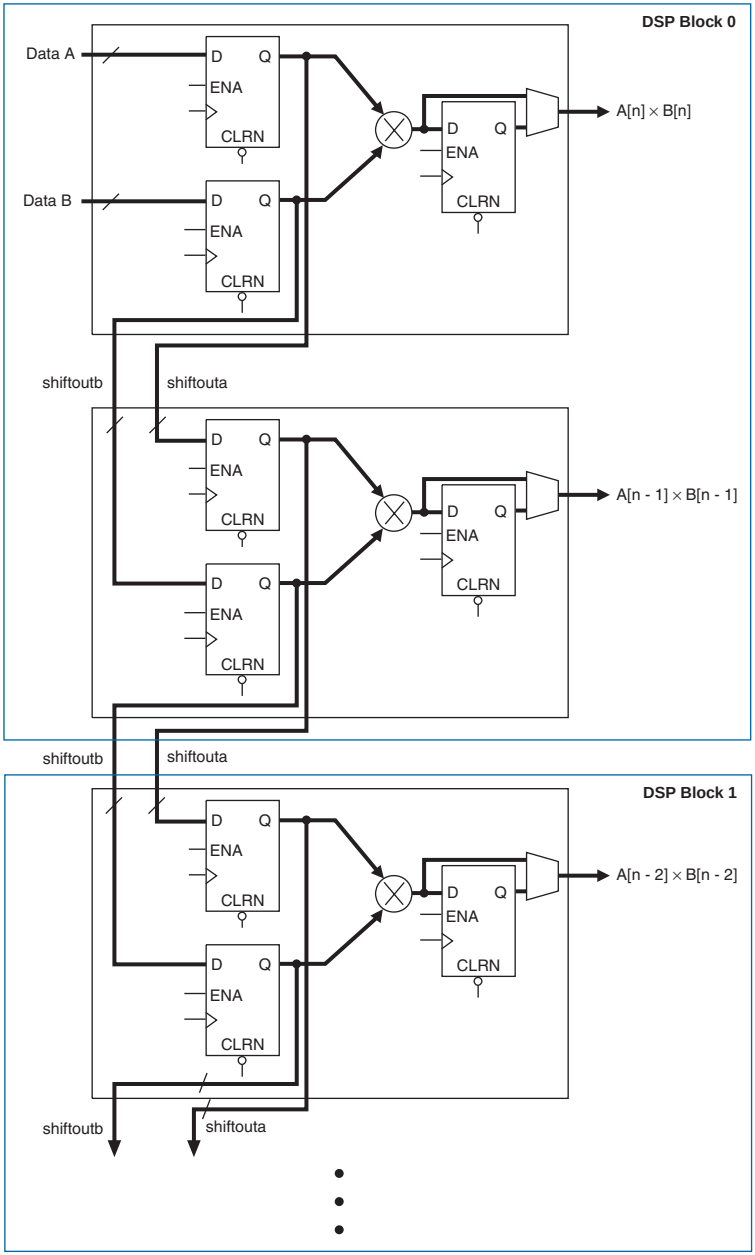
Input Registers

Each operand feeds an input register or the multiplier directly. The DSP block has the following signals (one of each controls every input and output register):

- `clock[3..0]`
- `ena[3..0]`
- `aclr[3..0]`

The input registers feed the multiplier and drive two dedicated shift output lines, `shiftouta` and `shiftoutb`. The shift outputs from one multiplier block directly feed the adjacent multiplier block in the same DSP block (or the next DSP block), as shown in Figure 6–4 on page 6–7, to form a shift register chain. This chain can terminate in any block, i.e., you can create any length of shift register chain up to 224 registers. A shift register is useful in DSP applications such as FIR filters. When implementing 9 × 9 and 18 × 18 multipliers, you do not need external logic to create the shift register chain because the input shift registers are internal to the DSP block. This implementation greatly reduces the required LE count and routing resources, and produces repeatable timing.

Figure 6–4. Shift Register Chain



Multiplier Stage

The multiplier stage supports 9 × 9, 18 × 18, or 36 × 36 multiplication. (The multiplier stage also support smaller multipliers. See "Operational Modes" on page 6–18 for details.) Based on the data width, a single DSP block can perform many multiplications in parallel.

The multiplier operands can be signed or unsigned numbers. Two signals, `signa` and `signb`, indicate the representation of the two operands. For example, a logic 1 on the `signa` signal indicates that data A is a signed number; a logic 0 indicates an unsigned number. The result of the multiplication is signed if any one of the operands is a signed number, as shown in Table 6–3.

Table 6–3. Multiplier Signed Representation

Data A	Data B	Result
Unsigned	Unsigned	Unsigned
Unsigned	Signed	Signed
Signed	Unsigned	Signed
Signed	Signed	Signed

The `signa` and `signb` signals affect the entire DSP block. Therefore, all of the data A inputs feeding the same DSP block must have the same sign representation. Similarly, all of the data B inputs feeding the same DSP block must have the same sign representation. The multiplier offers full precision regardless of the sign representation.

1 By default, the Altera Quartus® II software sets the multiplier to perform unsigned multiplication when the `signa` and `signb` signals are not used.

Pipeline Registers

The output from the multiplier can feed a pipeline register or be bypassed. You can use pipeline registers for any multiplier size; pipelining is useful for increasing the DSP block performance, particularly when using subsequent adder stages.

1 In the DSP block, pipelining improves the performance of 36 × 36 multipliers. For 18 × 18 multipliers and smaller, pipelining adds latency but does not improve performance.

Adder/Output Block

The adder / output block has the following elements (See Figure 6–5 on page 6–10):

- An adder / subtractor / accumulator block
- A summation block
- An output select multiplexer
- Output registers

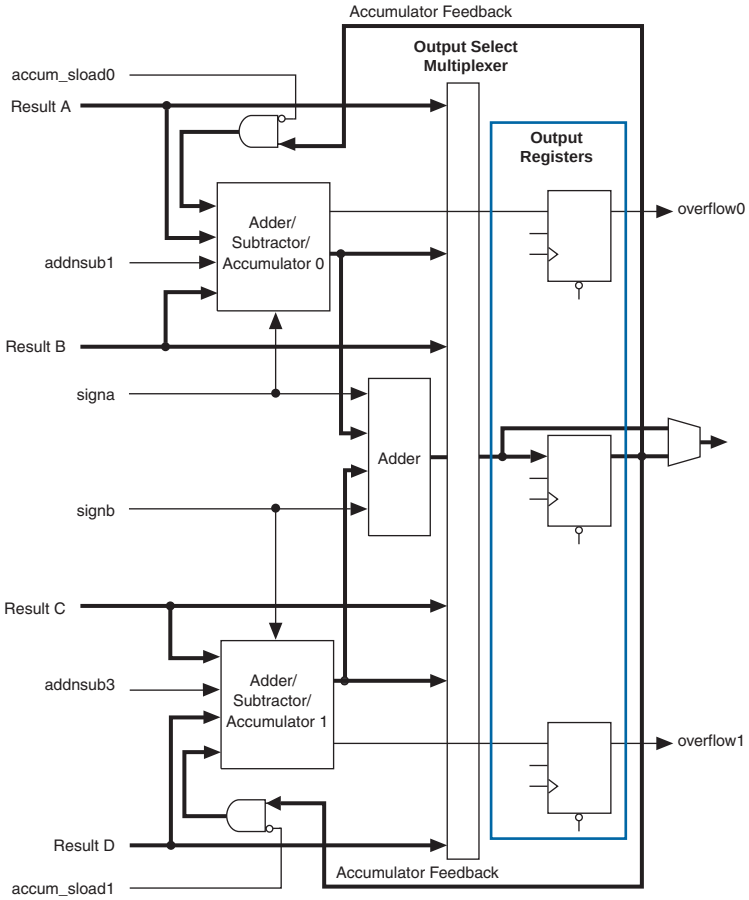
You can configure the adder / output block as:

- A pure output interface
- An accumulator
- A simple one-level adder
- A two-level adder with dynamic addition / subtraction control on the first-level adder
- The final stage of a 36-bit multiplier

The output select multiplexer sets the output of the DSP block. You can register the adder / output block's output using the output registers.

1 You cannot use the adder / output block independently of the multiplier.

Figure 6–5. Adder/Output Block



Adder/Subtractor/Accumulator Block

The adder / subtractor / accumulator is the first level of the adder / output block. You can configure the block as an accumulator or as an adder / subtractor.

Accumulator

When the adder / subtractor / accumulator is configured as an accumulator, the output of the adder / output block feeds back to the accumulator as shown in Figure 6–5. You can use the

`accum_sload[1..0]` signals to clear the accumulator asynchronously. This action is not the same as resetting the output registers. You can clear the accumulation and begin a new one without losing any clock cycles.

The `overflow` signal goes high on the positive edge of the clock when the accumulator overflows or underflows. In the next clock cycle, however, the `overflow` signal resets to zero even though an overflow (or underflow) occurred in the previous clock cycle. Use a latch to preserve the overflow condition indefinitely (until the latch is cleared).

Adder/Subtractor

The `addnsub[1..0]` signals select addition or subtraction: high for addition and low for subtraction. You can control the `addnsub[1..0]` signals using external logic; therefore, the first-level block can switch from an adder to a subtractor dynamically, simply by changing the `addnsub[1..0]` signals. If the first stage is configured as a subtractor, the output is A - B and C - D.

The adder / subtractor also uses two signals, `signa` and `signb`, like the multiplier block. These signals indicate the sign representation of both operands together. You can register the signals with a latency of 1 or 2 clock cycles.

Summation Block

The output from the adder / subtractor feeds to an optional summation block, which is an adder block that sums the outputs of the adder / subtractor. The summation block is important in applications such as FIR filters.

Output Select Multiplexer

The outputs from the various elements of the adder / output block are routed through an output select multiplexer. Based on the DSP block operational mode, the outputs of the multiplier block, adder / subtractor / accumulator, or summation block feed straight to the output, bypassing the remaining blocks in the DSP block.

1 The output select multiplier configuration is configured automatically by software.

Output Registers

You can use the output registers to register the DSP block output. Like the input registers, the output registers are controlled by the four `clock[3..0]`, `aclr[3..0]`, and `ena[3..0]` signals. You can use the output registers in any DSP block operational mode.

1 The output registers form part of the accumulator in the multiply-accumulate mode.

Routing Structure & Control Signals

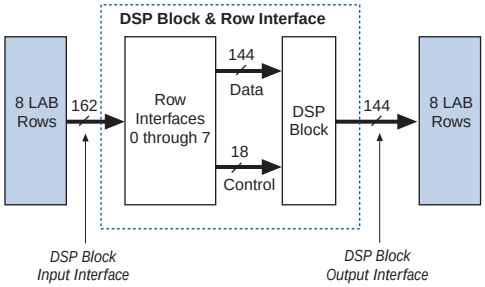
This section describes the interface between the DSP blocks and the row interface blocks. It also describes how the DSP block generates control signals and how the signals route from the row interface to the DSP block.

DSP Block Interface

The DSP blocks are organized in columns, which provides efficient horizontal communication between the blocks and the column-based memory blocks. The DSP block communicates with other parts of the device through an input and output interface. Each DSP block, including the input and output interface, is 8 logic array blocks (LABs) long.

The DSP block and row interface blocks consist of eight blocks that connect to eight adjacent LAB rows on the left and right. Each of the eight blocks has two regions: right and left, one per row. The DSP block receives 144 data input signals and 18 control signals for a total of 162 input signals. This block drives out 144 data output signals; 2 of the data signals can be used as overflow signals (`overflow`). Figure 6–6 provides an overview of the DSP block and its interface to adjacent LABs.

Figure 6–6. DSP Block Interface to Adjacent LABs



Input Interface

The DSP block input interface has 162 input signals from adjacent LABs; 18 data signals per row and 18 control signals per block.

Output Interface

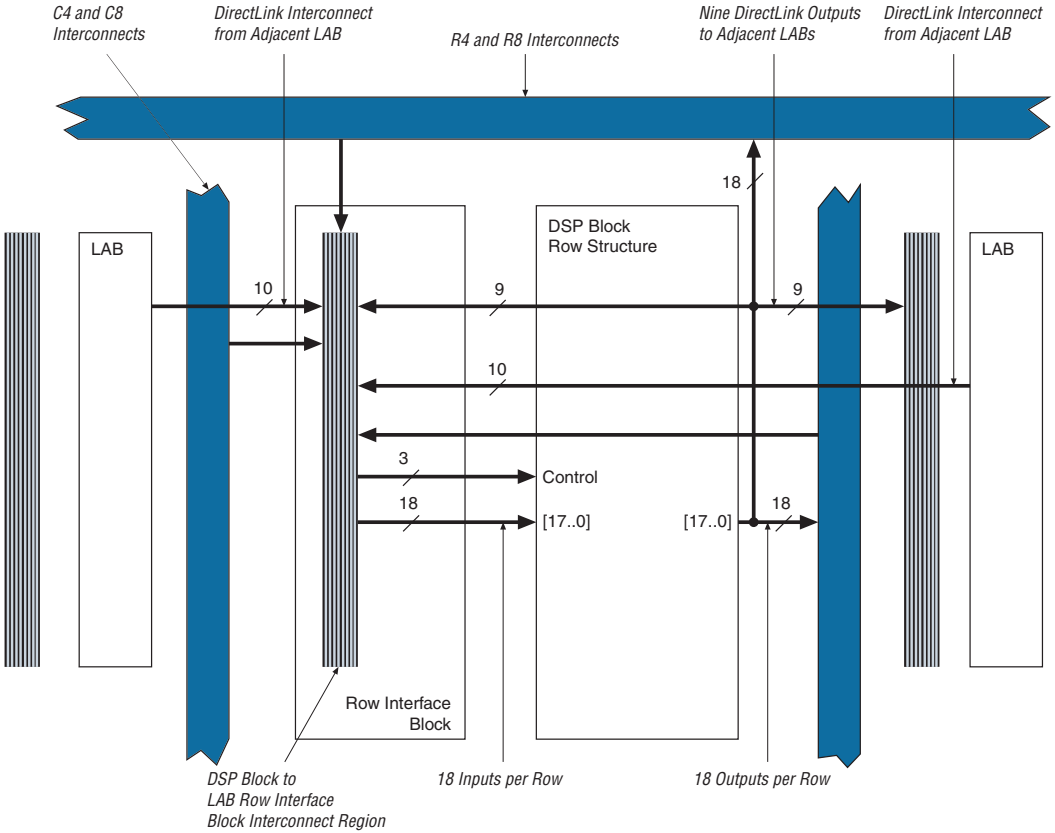
The DSP block output interface drives 144 outputs to adjacent LABs, 18 signals per row from 8 rows.

Because the DSP block outputs communicate horizontally, and because each DSP block row has more outputs than an LAB (18 from the DSP block compared to 10 from an LAB), the DSP block has double the number of row channel drivers compared to an LAB. The DSP block has the same number of row channels, but the row channels are staggered as if there were two LABs within each block. The DSP blocks have the same number of column channels as LABs because DSP blocks communicate primarily through row channels.

Row Interface Block

Each row interface block connects to the DSP block row structure with 21 signals. Because each DSP block has eight row interface blocks, this block receives 162 signals from the eight row interfaces. Of the 162 signals, 144 are data inputs and 18 are control signals. Figure 6–7 on page 6–14 shows one row block within the DSP block.

Figure 6–7. DSP Row Interface Block



Control Signals in the Row Interface Block

The DSP block has a set of input registers, a pipeline register, and an output register. Each register is grouped in banks that share the same clock and clear resources:

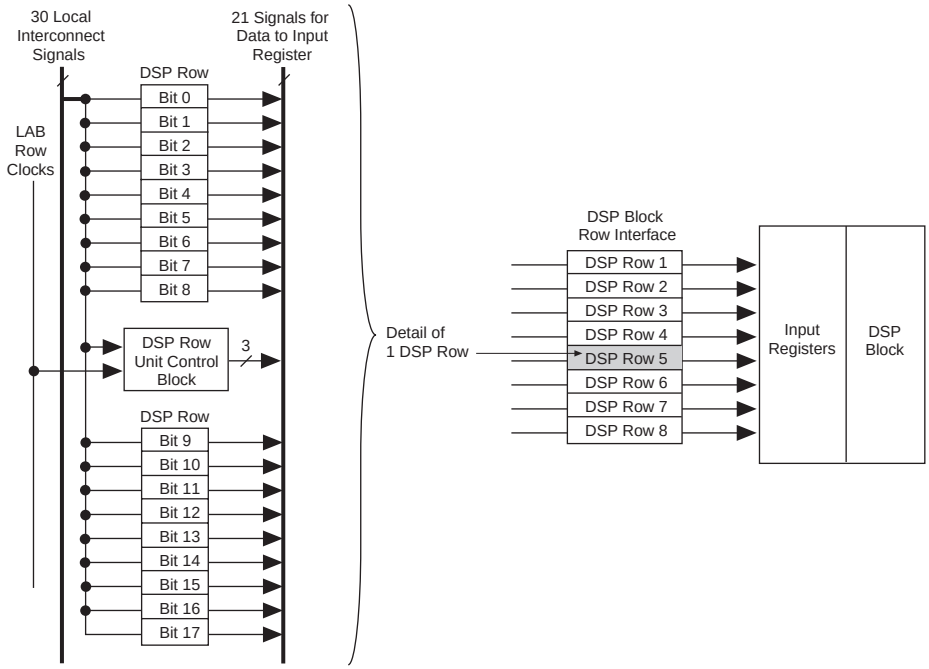
- 1- to 9-bit banks for the input register
- 1- to 18-bit banks for the pipeline register
- 18 bits for the output register

The row interface block generates the control signals and routes them to the DSP block. Each DSP block has 18 control signals:

- Four clock signals (`clock[3..0]`), which are available to each bank of DSP blocks
- Four clear signals (`aclr[3..0]`), which are available to each bank of DSP blocks
- Four clock enable signals (`ena[3..0]`), which the whole DSP block can use
- `signa` and `signb`, which are specific to each DSP block
- `addnsub[1..0]` signals
- `accum_sload[1..0]` signals

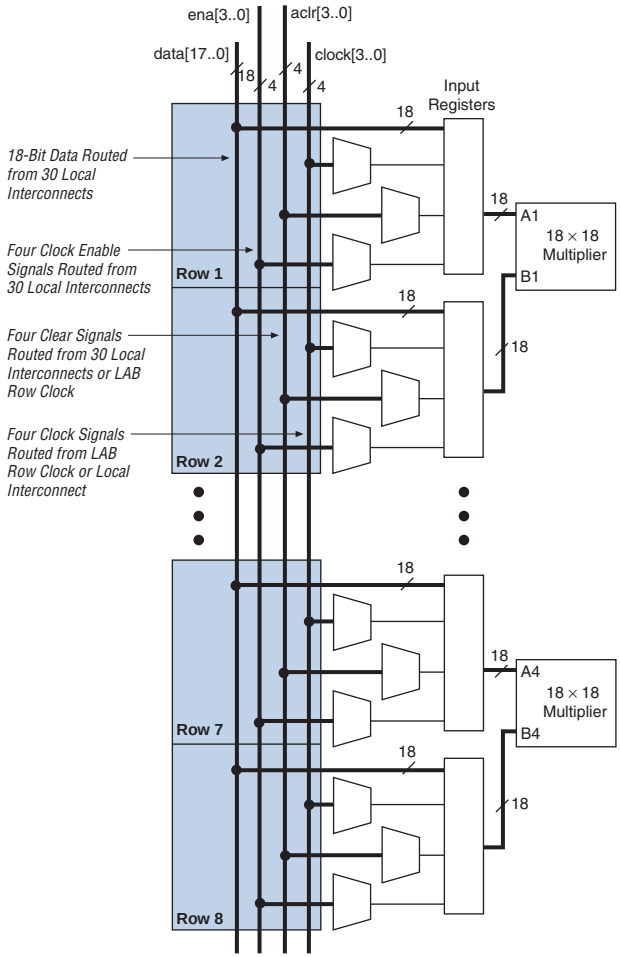
The `signa`, `signb`, and `addnsub[1..0]`, `accum_sload[1..0]` signals have independent clocks and clears and can be registered individually. When each 18 × 18 multiplier in the DSP block splits in half to two 9 × 9 multipliers, each 9 × 9 multiplier has independent control signals. Figure 6–8 shows the DSP block row interface and shows how it generates the data and control signals.

Figure 6–8. DSP Block Row Interface



The DSP block interface generates the clock signals from LAB row clocks or the local interconnect. The clear signals are generated from the local interconnects within each DSP block row interface or from LAB row clocks. The four clock enable signals are generated from the 30 local interconnects from the same LAB rows that generate the clock signals. The clock enable is paired with the clock because the enable logic is implemented at the interface. Figure 6–9 shows the signal distribution within the row interface block.

Figure 6–9. DSP Block Row Interface Signal Distribution



Each row block provides 18 bits of data to the multiplier (i.e., one of the operands to the multiplier), which are routed through the 30 local interconnects within each DSP row interface block. Any signal in the device can be the source of the 18-bit multiplier data, by connecting to the local row interconnect through any row or column.

Each control signal routes through one of the eight rows of the DSP block. Table 6–4 shows the 18 control signals and the row to which each one routes.

Table 6–4. Control Signals in DSP Block

Signal Name	Row	Description
signa	1	DSP block-wide signed and unsigned control signals for all multipliers. The multiplier outputs are unsigned only if both signa and signb are low.
signb	6	
addnsub1	3	Controls addition or subtraction of the two one-level adders. The addnsub0 signal controls the top two one-level adders; the addnsub1 signal controls the bottom two one-level adders. A high indicates addition; a low indicates subtraction.
addnsub3	7	
accum_sload0	2	Resets the feedback input to the accumulator. The signal asynchronously clears the accumulator and allows new accumulation to begin without losing any clock cycles. The accum_sload0 controls the top two one-level adders, and the accum_sload1 controls the bottom two one-level adders. A low is for normal accumulation operations and a high is for zeroing the accumulator.
accum_sload1	7	
clock0	3	DSP block-wide clock signals.
clock1	4	
clock2	5	
clock3	6	
aclr0	1	DSP block-wide clear signals.
aclr1	4	
aclr2	5	
aclr3	7	
ena[3..0]	Same rows as the Clock Signals	DSP block-wide clock enable signals.

Input/Output Data Interface Routing

The 30 local interconnects generate the 18 inputs to the row interface blocks. The 21 outputs of the row interface block are the inputs to the DSP row block (see Figure 6–7 on page 6–14).

The row interface block has DirectLink™ connections that connect the DSP block input or output signals to the left and right adjacent LABs at each row. (The DirectLink connections provide interconnects between LABs and adjacent blocks.) The DirectLink connection reduces the use of row and column interconnects, providing higher performance and flexibility.

Each row interface block receives 10 DirectLink connections from the right adjacent LABs and 10 from the left adjacent LABs. Additionally, the row interface block receives signals from the DSP block, making a total of 30 local interconnects for each row interface block. All of the row and column resources within the DSP block can access this interconnect region (see Figure 6–7 on page 6–14).

A DSP block has nine outputs that drive the right adjacent LAB and nine that drive the left adjacent LAB through DirectLink interconnects. All 18 outputs drive any row or column.

Operational Modes

You can use the DSP block in one of four operational modes, depending on your application needs (see Table 6–4). The Quartus II software has built-in megafunctions that you can use to control the mode. After you have made your parameter settings using the megafunction's MegaWizard® Plug-In, the Quartus II software automatically configures the DSP block.

Table 6–5. DSP Block Operational Modes

Mode	9 × 9	18 × 18	36 × 36
Simple multiplier	Eight multipliers with eight product outputs	Four multipliers with four product outputs	One multiplier
Multiply accumulator	Two 34-bit multiply-accumulate blocks	Two 52-bit multiply-accumulate blocks	–
Two-multiplier adder	Four two-multiplier adders	Two two-multiplier adders	–
Four-multiplier adder	Two four-multiplier adders	One four-multiplier adder	–

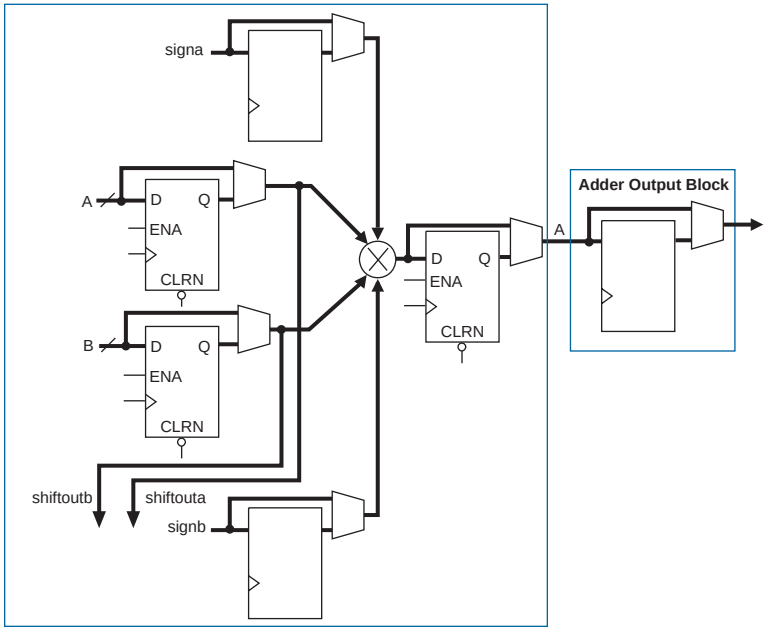
Simple Multiplier Mode

In simple multiplier mode, the DSP block performs individual multiplication operations for general-purpose multipliers and for applications such as equalizer coefficient updates that require many individual multiplication operations.

9- & 18-Bit Multipliers

You can configure each DSP block multiplier for 9 or 18 bits. A single DSP block can support up to 8 individual 9-bit or smaller multipliers, or up to 4 individual multipliers with operand widths between 10- and 18-bits. Figure 6–10 shows the simple multiplier mode.

Figure 6–10. Simple Multiplier Mode



The multiplier operands can accept signed integers, unsigned integers, or a combination. The `signa` and `signb` signals are dynamic and can be registered in the DSP block. Additionally, you can register the multiplier inputs and results independently. Pipelining the result, using the pipeline registers in the block, increases the performance of the DSP block.

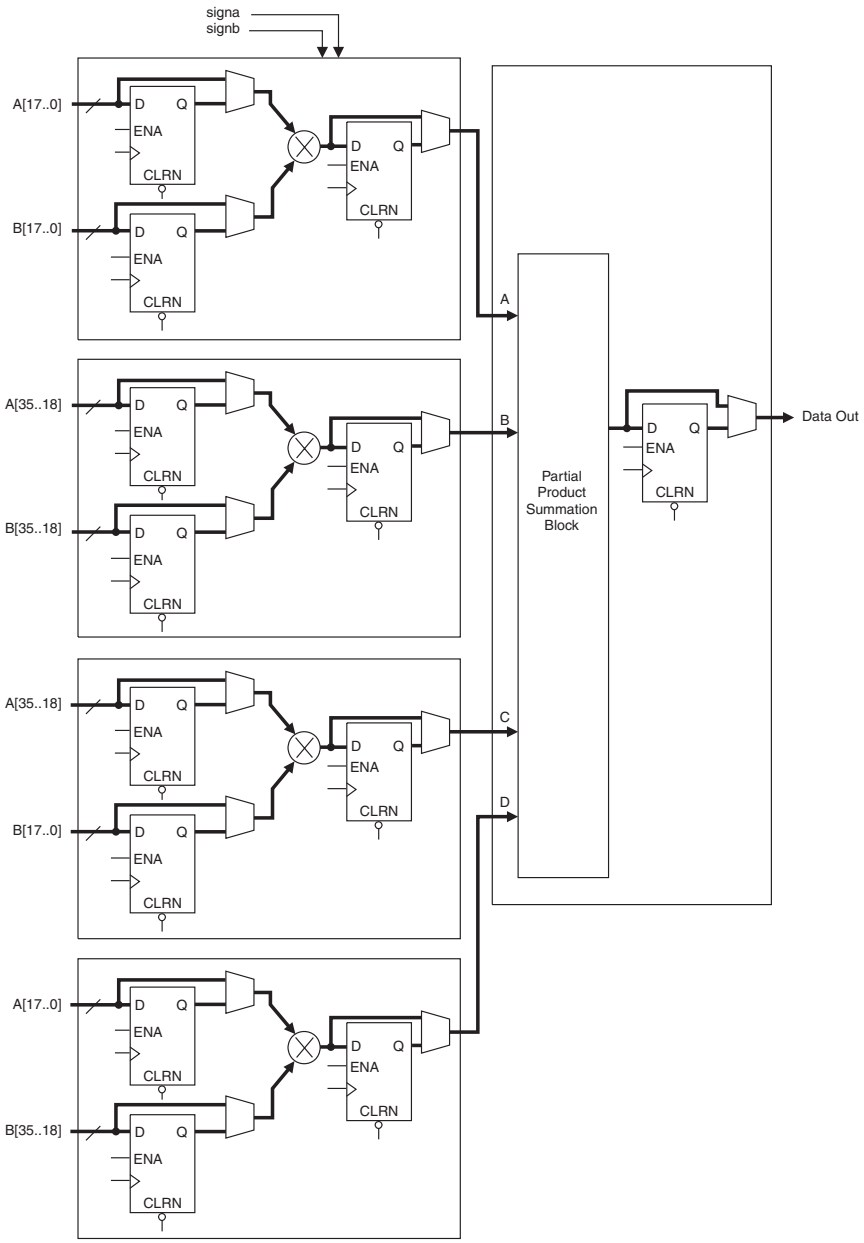
36-Bit Multiplier

The 36-bit multiplier is a subset of the simple multiplier mode. It uses the entire DSP block to implement one 36 × 36-bit multiplier. The four 18-bit multipliers are fed part of each input, as shown in Figure 6–11 on page 6–21. The adder / output block adds the partial products using the

summation block. You can use pipeline registers between the multiplier stage and the summation block. The 36 × 36-bit multiplier supports signed and unsigned operation.

The 36-bit multiplier is useful when your application needs more than 18-bit precision, for example, for mantissa multiplication of precision floating-point arithmetic applications.

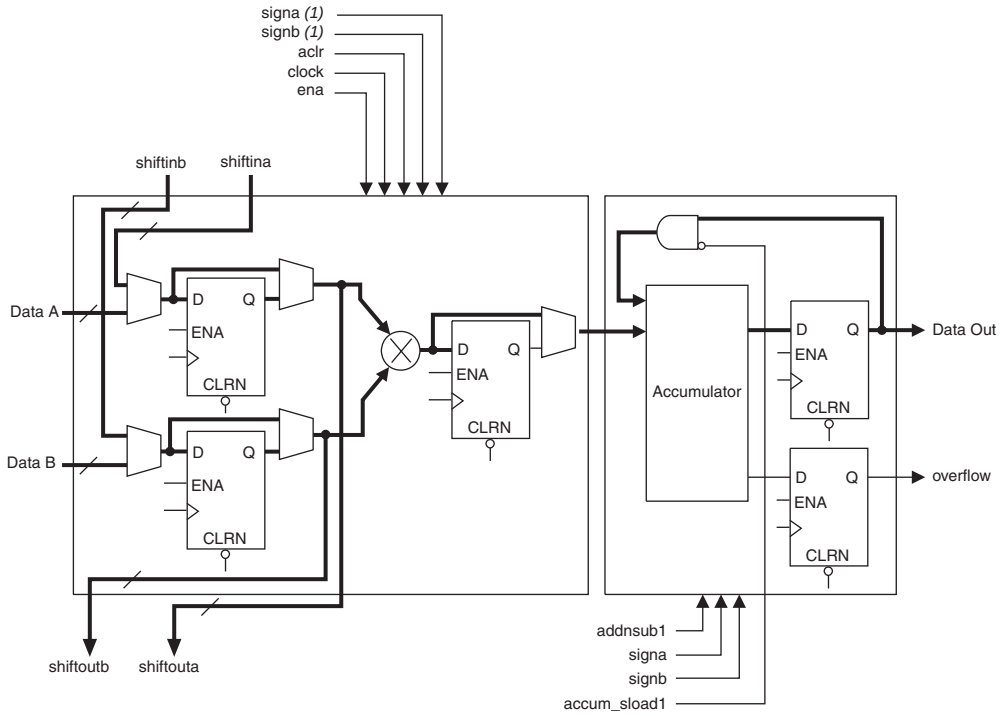
Figure 6–11. 36-Bit Multiplier



Multiply Accumulator Mode

In multiply accumulator mode, the output of the multiplier stage feeds the adder / output block, which is configured as an accumulator or subtractor (see Figure 6–12). You can implement up to two independent 18-bit multiply accumulators in one DSP block. The Quartus II software implements smaller multiplier-accumulators by tying the unused low-order bits of an 18-bit multiplier to ground.

Figure 6–12. Multiply Accumulator Mode



Note to Figure 6–12:

(1) The `signa` and `signb` signals are the same in the multiplier stage and the adder / output block.

The multiply accumulator output can be up to 52 bits wide for a maximum 36-bit result with 16-bits of accumulation. In this mode, the DSP block uses output registers and the `accum_sload` and `overflow` signals. The `accum_sload[1..0]` signal synchronously loads the multiplier result to the accumulator output. This signal can be unregistered or registered once or twice. The DSP block can then begin a new accumulation without losing any clock cycles. The `overflow` signal indicates an overflow or underflow in the accumulator. This signal is

cleared for the next accumulation cycle, and you can use an external latch to preserve the signal. You can use the `addnsub[1..0]` signals to perform accumulation or subtraction dynamically.

1 If you want to use DSP blocks and your design only has an accumulator, you can use a multiply by one followed by an accumulator to force the software to implement the logic in the DSP block.

Two-Multiplier Adder Mode

The two-multiplier adder mode uses the adder / output block to add or subtract the outputs of the multiplier block, which is useful for applications such as FFT functions and complex FIR filters. Additionally, in this mode, the DSP block outputs two sums or differences for multipliers up to 18 bits, or 4 sums or differences for 9-bit or smaller multipliers. A single DSP block can implement one 18 × 18-bit complex multiplier or two 9 × 9-bit complex multipliers.

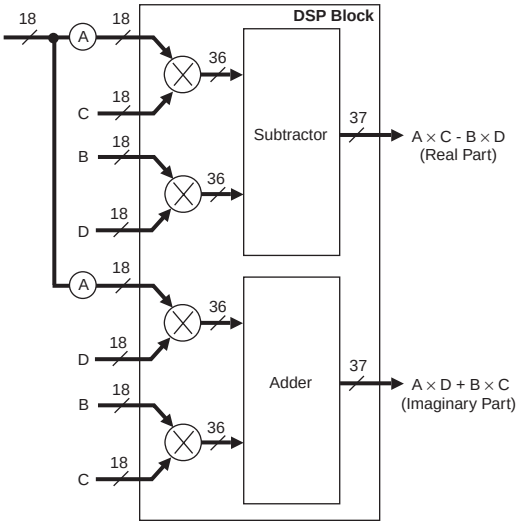
A complex multiplication can be written as:

$$(a + jb) \times (c + jd) = (a \times c - b \times d) + j \times (a \times d + b \times c)$$

In this mode, a single DSP block calculates the real part ($a \times c - b \times d$) using one adder / subtractor / accumulator and the imaginary part ($a \times d + b \times c$) using another adder / subtractor / accumulator for data up to 18 bits. Figure 6–13 shows an 18-bit complex multiplication. For data widths up to 9 bits, the DSP block can perform two complex multiplications using four one-level adders. Resources outside of the DSP block route each input to the two multiplier inputs.

1 You can only use the adder block if it follows multiplication operations.

Figure 6–13. Complex Multiplier Implemented Using Two-Multiplier Adder Mode



Four-Multiplier Adder Mode

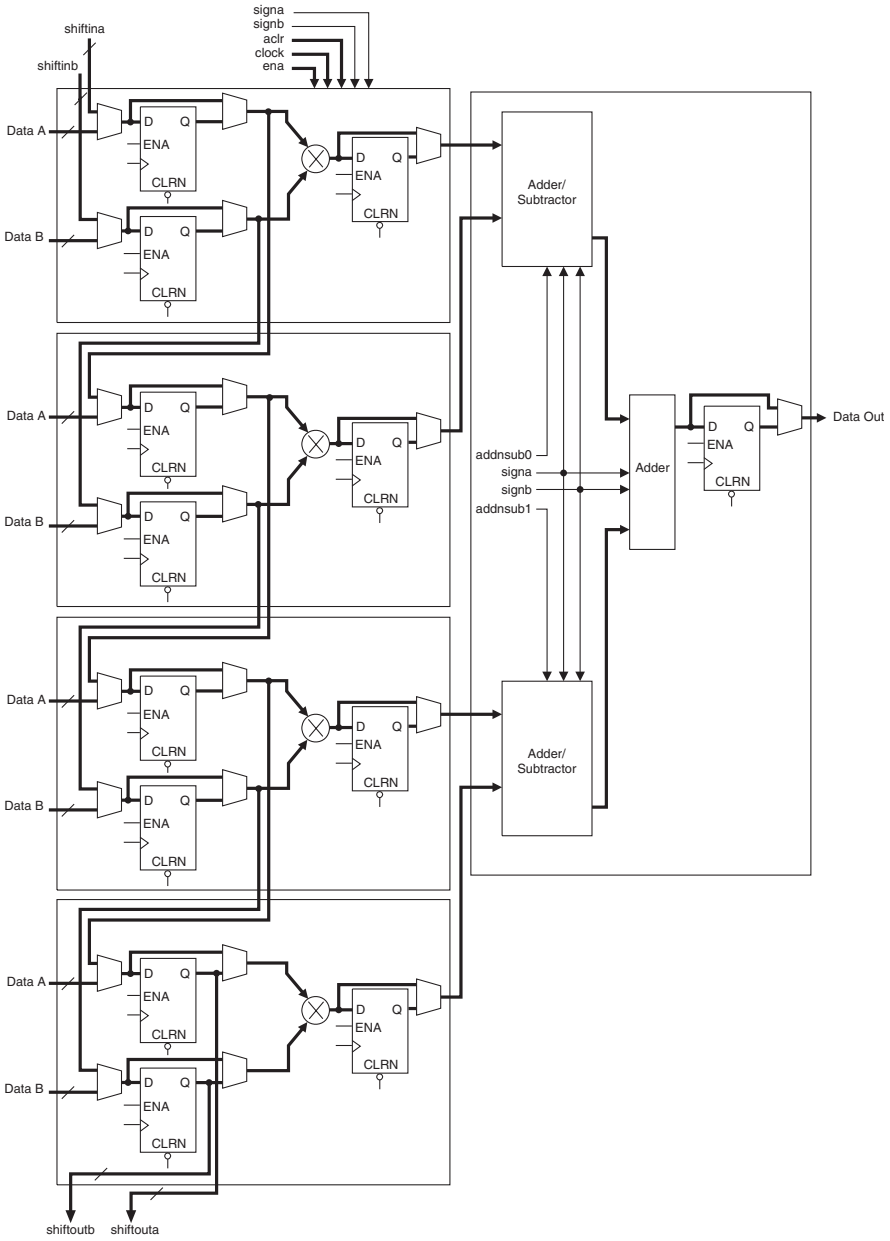
In the four-multiplier adder mode, which you can use for 1-dimensional and 2-dimensional filtering applications, the DSP block adds the results of two adder / subtractor / accumulators in a final stage (the summation block).

1 You can only use the adder block if it follows multiplication operations.

9- & 18-Bit Summation Blocks

A single DSP block can implement one 18 × 18 or two 9 × 9 summation blocks (see Figure 6–14 on page 6–25). The multiplier product widths must be the same size.

Figure 6–14. Four-Multiplier Adder Mode



FIR Filters

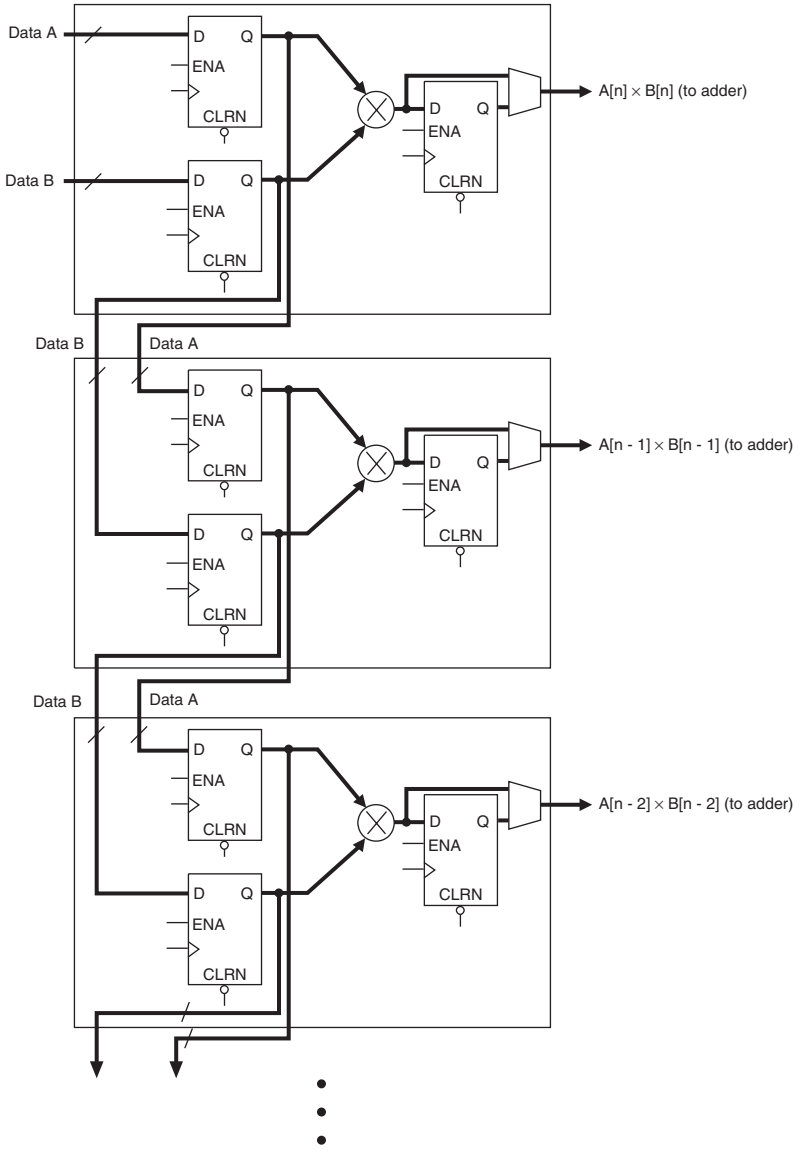
The four-multiplier adder mode can be used for FIR filter and complex FIR filter applications. The DSP block combines a four-multiplier adder with the input registers configured as shift registers. One set of shift inputs contains the filter data, while the other holds the coefficients, which can be loaded serially or in parallel (see Figure 6–15).

The input shift register eliminates the need for shift registers external to the DSP block (e.g., implemented in device logic elements). This architecture simplifies filter design and improves performance because the DSP block implements all of the filter circuitry.

1 Serial shift inputs in 36-bit simple multiplier mode require external registers.

One DSP block can implement an entire 18-bit FIR filter with up to four taps. For FIR filters larger than four taps, you can cascade DSP blocks with additional adder stages implemented in logic elements.

Figure 6–15. Input Shift Registers Configured for a FIR Filter



Software Support

Altera provides two distinct methods for implementing various modes of the DSP block in your design: instantiation and inference. Both methods use the following three Quartus II megafunctions:

- lpm_mult
- altmult_add
- altmult_accum

You can instantiate the megafunctions in the Quartus II software to use the DSP block. Alternatively, with inference, you can create a HDL design an synthesize it using a third-party synthesis tool like LeonardoSpectrum or Synplify or Quartus II Native Synthesis that infers the appropriate megafunction by recognizing multipliers, multiplier adders, and multiplier accumulators. Using either method, the Quartus II software maps the functionality to the DSP blocks during compilation.

f See the *Implementing High-Performance DSP Functions in Stratix & Stratix GX Devices* chapter in the *Stratix Device Handbook, Volume 2* or the *Stratix GX Device Handbook, Volume 2* for more information on using DSP blocks to implement high-performance DSP functions such as FIR filters, IIR filters, and discreet cosine transforms (DCTs).

f See Quartus II On-Line Help for instructions on using the megafunctions and the MegaWizard Plug-In Manager.

f For more information on DSP block inference support, see the *Recommended HDL Coding Styles* chapter of the *Quartus II Development Software Handbook v4.1, Volume 1*.

Conclusion

The Stratix and Stratix GX device DSP blocks are optimized to support DSP applications that need high data throughput, such as FIR filters, FFT functions, and encoders. These DSP blocks are flexible and can be configured in one of four operational modes to suit any application need. The DSP block's adder / subtractor / accumulator and the summation blocks minimize the amount of logic resources used and provide efficient routing. This efficiency results in improved performance and data throughput for DSP applications. The Quartus II software, together with the LeonardoSpectrum and Synplify software, provides a complete and easy-to-use flow for implementing functionality in the DSP block.

7. Implementing High Performance DSP Functions in Stratix & Stratix GX Devices

S52007-1.1

Introduction

Digital signal processing (DSP) is a rapidly advancing field. With products increasing in complexity, designers face the challenge of selecting a solution with both flexibility and high performance that can meet fast time-to-market requirements. DSP processors offer flexibility, but they lack real-time performance, while application-specific standard products (ASSPs) and application-specific integrated circuits (ASICs) offer performance, but they are inflexible. Only programmable logic devices (PLDs) offer both flexibility and high performance to meet advanced design challenges.

The mathematical theory underlying basic DSP building blocks—such as the finite impulse response (FIR) filter, infinite impulse response (IIR) filter, fast fourier transform (FFT), and direct cosine transform (DCT)—is computationally intensive. Altera® Stratix® and Stratix GX devices feature dedicated DSP blocks optimized for implementing arithmetic operations, such as multiply, multiply-add, and multiply-accumulate.

In addition to DSP blocks, Stratix and Stratix GX devices have TriMatrix™ embedded memory blocks that feature various sizes that can be used for data buffering, which is important for most DSP applications. These dedicated hardware features make Stratix and Stratix GX devices an ideal DSP solution.

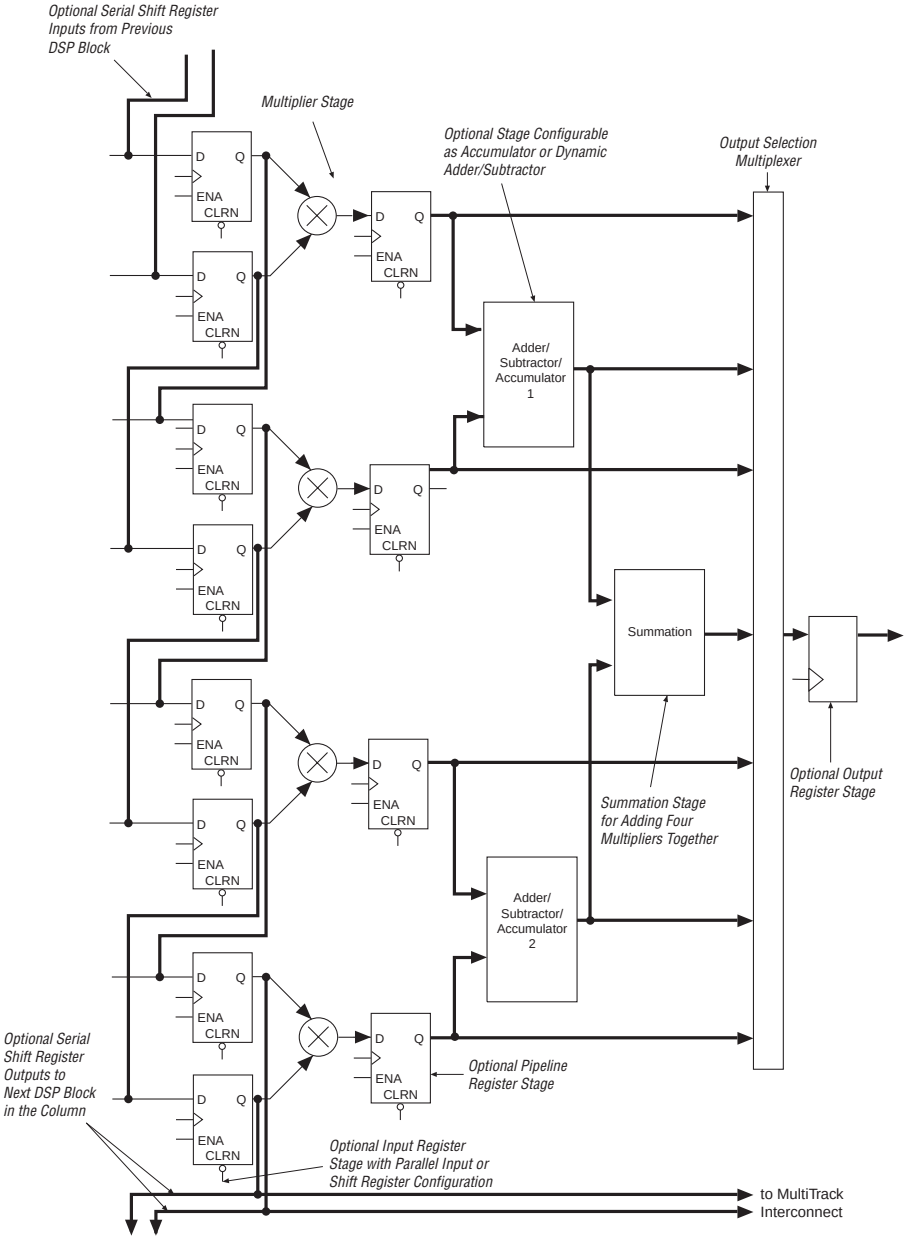
This chapter describes the implementation of high performance DSP functions, including filters, transforms, and arithmetic functions, using Stratix and Stratix GX DSP blocks. The following topics are discussed:

- FIR filters
- IIR filters
- Matrix manipulation
- Discrete Cosine Transform
- Arithmetic functions

Stratix & Stratix GX DSP Block Overview

Stratix and Stratix GX devices feature DSP blocks that can efficiently implement DSP functions, including multiply, multiply-add, and multiply-accumulate. The DSP blocks also have three built-in registers sets: the input registers, the pipeline registers at the multiplier output, and the output registers. Figure 7–1 shows the DSP block operating in the 18 × 18-bit mode.

Figure 7–1. DSP Block Diagram for 18 x 18-bit Mode



The DSP blocks are organized into columns enabling efficient horizontal communication with adjacent TriMatrix memory blocks. Tables 7–1 and 7–2 show the DSP block resources in Stratix and Stratix GX devices, respectively.

Table 7–1. DSP Block Resources in Stratix Devices

Device	DSP Blocks	Maximum 9 × 9 Multipliers	Maximum 18 × 18 Multipliers	Maximum 36 × 36 Multipliers
EP1S10	6	48	24	6
EP1S20	10	80	40	10
EP1S25	10	80	40	10
EP1S30	12	96	48	12
EP1S40	14	112	56	14
EP1S60	18	144	72	18
EP1S80	22	176	88	22

Table 7–2. DSP Block Resources in Stratix GX Devices

Device	DSP Blocks	Maximum 9 × 9 Multipliers	Maximum 18 × 18 Multipliers	Maximum 36 × 36 Multipliers
EP1SGX10C	6	48	24	6
EP1SGX10D	6	48	24	6
EP1SGX25C	10	80	40	10
EP1SGX25D	10	80	40	10
EP1SGX25F	10	80	40	10
EP1SGX40D	14	112	56	14
EP1SGX40G	14	112	56	14

Each DSP block supports either eight 9 × 9-bit multipliers, four 18-bit multipliers, or one 36 × 36-bit multiplier. These multipliers can feed an adder or an accumulator unit based on the operation mode. Table 7–3 shows the different operation modes for the DSP blocks.

Table 7–3. Operation Modes for DSP Blocks

DSP Block Mode	Number & Size of Multipliers per DSP Block		
	9 x 9-bit	18 x 18-bit	36 x 36-bit
Simple multiplier	Eight multipliers with eight product outputs	Four multipliers with four product outputs	One multiplier with one product output
Multiply-accumulate	Two multiply and accumulate (34 bit)	Two multiply and accumulate (52 bit)	
Two-multipliers adder	4 two-multipliers adders	2 two-multipliers adders	
Four-multipliers adder	2 four-multipliers adder	1 four-multipliers adder	

Implementing multipliers, multiply-adders, and multiply-accumulators in the DSP blocks has a performance advantage over logic cell implementation. Using DSP blocks also reduces logic cell and routing resource consumption. To achieve higher performance, register each stage of the DSP block to allow pipelining. For implementing applications, such as FIR filters, efficiently use the input registers of the DSP block as shift registers.

f For more information on DSP blocks, see the *DSP Blocks in Stratix & Stratix GX Devices* chapter.

TriMatrix Memory Overview

Stratix and Stratix GX devices feature the TriMatrix memory structure, composed of three sizes of embedded RAM blocks. These include the 512-bit size M512 block, the 4-Kbit size M4K block, and the 512-Kbit size M-RAM block. Each block is configurable to support a wide range of features.

Tables 7–4 and 7–5 show the number of memory blocks in each Stratix and Stratix GX device, respectively.

Table 7–4. TriMatrix Memory Resources in Stratix Devices (Part 1 of 2)

Device	M512	M4K	M-RAM
EP1S10	94	60	1
EP1S20	194	82	2
EP1S25	224	138	2

Table 7–4. TriMatrix Memory Resources in Stratix Devices (Part 2 of 2)

Device	M512	M4K	M-RAM
EP1S30	295	171	4
EP1S40	384	183	4
EP1S60	574	292	6
EP1S80	767	364	9

Table 7–5. TriMatrix Memory Resources in Stratix GX Devices

Device	M512	M4K	M-RAM
EP1SGX10C	94	60	1
EP1SGX10D	94	60	1
EP1SGX25C	224	138	2
EP1SGX25D	224	138	2
EP1SGX25F	224	138	2
EP1SGX40D	384	183	4
EP1SGX40G	384	183	4

Most DSP applications require local data storage for intermediate buffering or for filter storage. The TriMatrix memory blocks enable efficient use of available resources for each application.

The M512 and M4K memory blocks can implement shift registers for applications, such as multi-channel filtering, auto-correlation, and cross-correlation functions. Implementing shift registers in embedded memory blocks reduces logic cell and routing resource consumption.

f For more information on TriMatrix memory blocks, see the *TriMatrix Embedded Memory Blocks in Stratix & Stratix GX Devices* chapter.

DSP Function Overview

The following sections describe commonly used DSP functions. Each section illustrates the implementation of a basic DSP building block, including FIR and IIR filters, in Stratix and Stratix GX devices using DSP blocks and TriMatrix memory blocks.

Finite Impulse Response (FIR) Filters

This section describes the basic theory and implementation of basic FIR filters, time-domain multiplexed (TDM) FIR filters, and interpolation and decimation polyphase FIR filters. An introduction to the complex FIR filter is also presented in this section.

FIR Filter Background

Digital communications systems use FIR filters for a variety of functions, including waveform shaping, anti-aliasing, band selection, decimation / interpolation, and low pass filtering. The basic structure of a FIR filter consists of a series of multiplications followed by an addition.

The following equation represents an FIR filter operation:

$$y(n) = x(n) \otimes n(n)$$

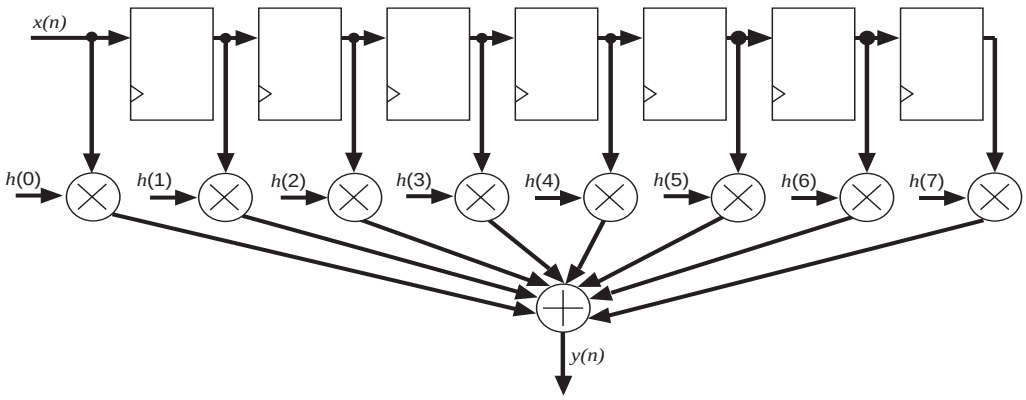
$$y(n) = \sum_{i=0}^{L-1} x(n-i)h(i)$$

where:

$x(n)$ represents the sequence of input samples
$h(n)$ represents the filter coefficients
L is the number of filter taps

A sample FIR filter with L=8 is shown in Figure 7–2.

Figure 7–2. Basic FIR Filter



This example filter in Figure 7–2 uses the input values at eight different time instants to produce an output. Hence, it is an 8-tap filter. Each register provides a unit sample delay. The delayed inputs are multiplied with their respective filter coefficients and added together to produce the output. The width of the output bus depends on the number of taps and the bit width of the input and coefficients.

Basic FIR Filter

A basic FIR filter is the simplest FIR filter type. As shown in Figure 7–2, a basic FIR filter has a single input channel and a single output channel.

Basic FIR Filter Implementation

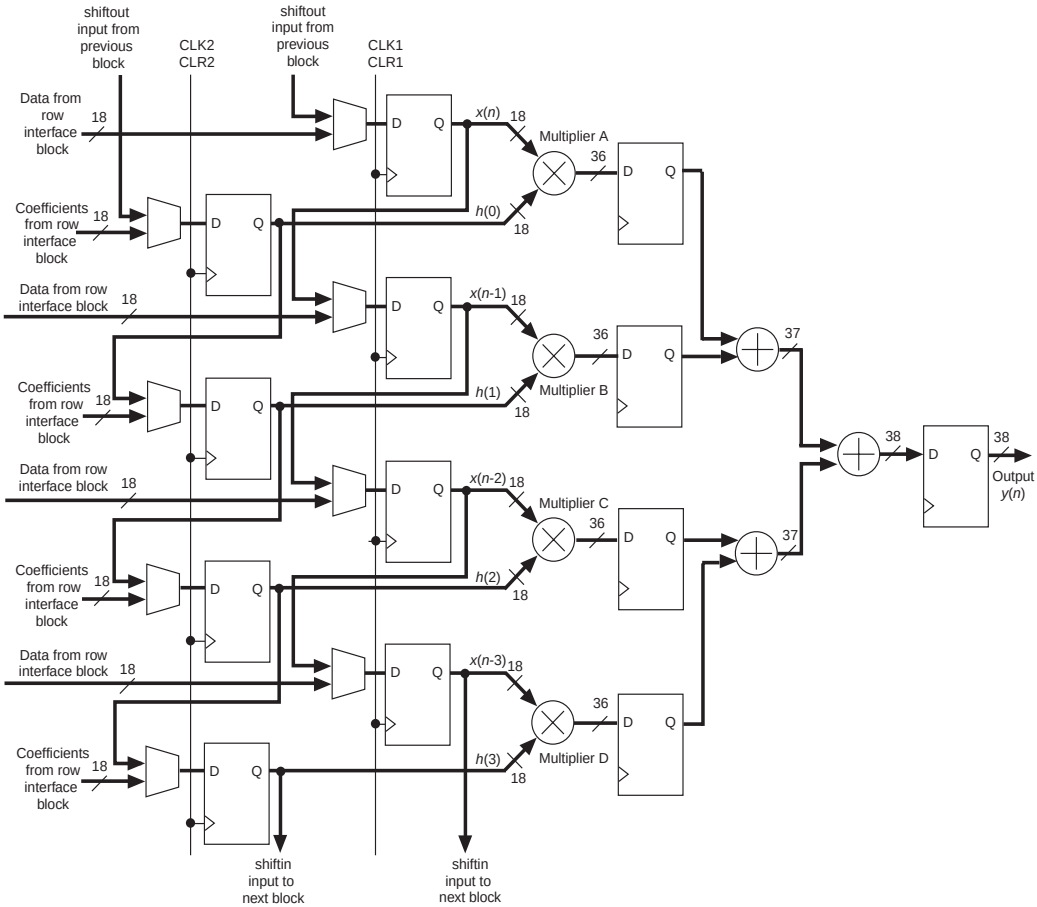
Stratix and Stratix GX devices' dedicated DSP blocks can implement basic FIR filters. Because these DSP blocks have closely integrated multipliers and adders, filters can be implemented with minimal routing resources and delays. For implementing FIR filters, the DSP blocks are configured in the four-multipliers adder mode.

f See the *DSP Blocks in Stratix & Stratix GX Devices* chapter for more information on the different modes of the DSP blocks.

This section describes the implementation of an 18-bit 8-tap FIR filter. Because Stratix and Stratix GX devices support modularity, cascading two 4-tap filters can implement an 8-tap filter. Larger FIR filters can be designed by extending this concept. Users can also increase the number of taps available per DSP block if 18 bits of resolution are not required. For example, by using only 9 bits of resolution for input samples and coefficient values, 8 multipliers are available per DSP block. Therefore, a 9-bit 8-tap filter can be implemented in a single DSP block provided an external adder is implemented in logic cells.

The four-multipliers adder mode, shown in Figure 7–3, provides four 18 × 18-bit multipliers and three adders in a single DSP block. Hence, it can implement a 4-tap filter. The data width of the input and the coefficients is 18 bits, which results in a 38-bit output for a 4-tap filter.

Figure 7–3. Hardware View of a DSP Block in Four-Multipliers Adder Mode *Notes (1). (2), (3)*



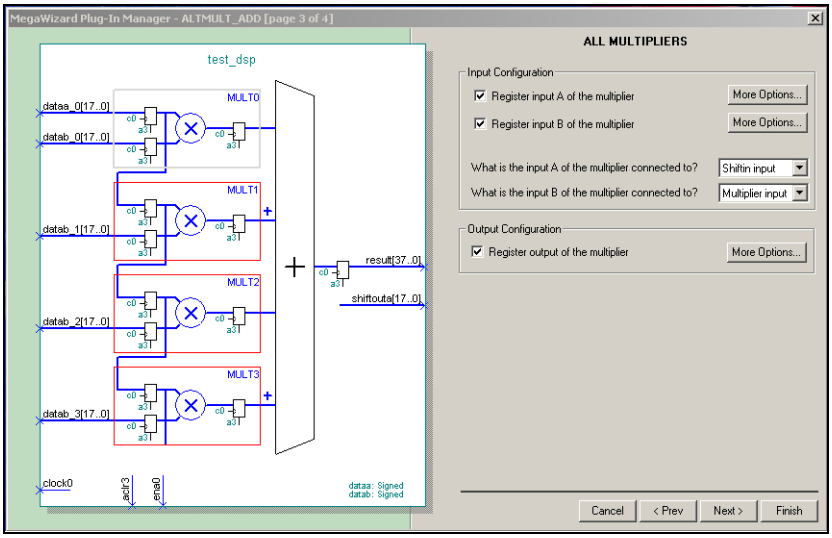
Notes to Figure 7–3:

(1) The input registers feed the multiplier blocks. These registers can increase the DSP block performance, but are optional. These registers can also function as shift registers if the dedicated shiftin / shiftout signals are used.

(2) The pipeline registers are fed by the multiplier blocks. These registers can increase the DSP block performance, but are optional.

(3) The output registers register the DSP block output. These registers can increase the DSP block performance, but are optional.

Figure 7–4. Quartus II Software View of MegaWizard Implementation of a DSP Block in Four-Multipliers Adder Mode



Each input register of the DSP block provides a shiftout output that connects to the shiftin input of the adjacent input register of the same DSP block. The registers on the boundaries of a DSP block also connect to the registers of adjacent DSP blocks through the use of shiftin / shiftout connections. These connections create register chains spanning multiple DSP blocks, which makes it easy to increase the length of FIR filters. Figure 7–5 shows two DSP blocks connected to create an 8-tap FIR filter. Filters with more taps can be implemented by connecting DSP blocks in a similar manner, provided sufficient DSP blocks are available in the device.

1 Adding the outputs of the two DSP blocks requires an external adder which can be implemented using logic cells.

The input data can be fed directly or by using the shiftout / shiftin chains, which allow a single input to shift down the register chain inside the DSP block. The input to each of the registers has a multiplexer, hence, the data can be fed either from outside the DSP block or the preceding register. This can be selected from the MegaWizard® in the Quartus® II software, as shown in Figure 7–4. The example in Figure 7–5 uses the shiftout / shiftin flip-flop chains where the multiplexers are configured to use these chains. In this example, the flip-flops inside the DSP blocks serve as the taps of the FIR filter.

When the coefficients are loaded in parallel, they can be fed directly from memory elements or any other muxing scheme. This facilitates the implementation of an adaptive (variable) filter.

Further, if the user wants to implement the shift register chains external to the DSP block, this can be done by using the `altshift_taps` megafunction. In this case, the coefficient and data shifting is done external to the DSP block. The DSP block is only used to implement the multiplications and the additions.

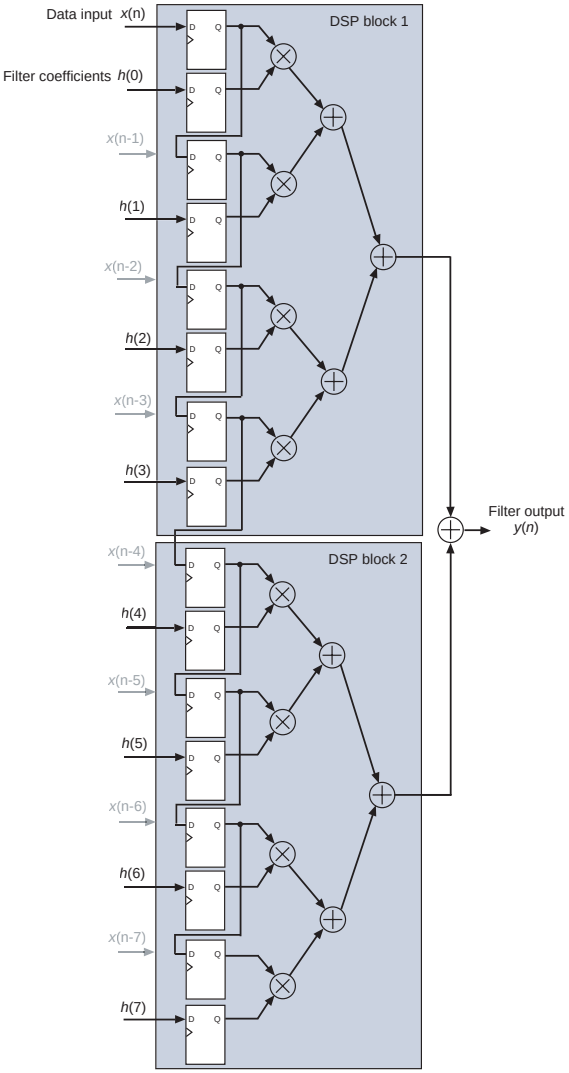
Parallel vs. Serial Implementation

The fastest implementations are fully parallel, but consume more logic resources than serial implementations. To trade-off performance for logic resources, implement a serial scheme with a specified number of taps. To facilitate this, Altera provides the FIR Compiler core through its MegaCore program. The FIR Compiler is an easy-to-use, fully-integrated graphical user interface (GUI) based FIR filter design software.

f For more information on the FIR Compiler MegaCore, visit the Altera web site at **www.altera.com** and search for “FIR compiler” in the “Intellectual Property” page.

It is important to note that the four-multipliers adder mode allows a DSP block to be configured for parallel or serial input. When it is configured for parallel input, as shown in Figure 7–6, the data input and the coefficients can be loaded directly without the need for shiftin / shiftout chains between adjacent registers in the DSP block. When the DSP block is configured for serial input, as shown in Figure 7–5, the shiftin / shiftout chains create a register cascade both within the DSP block and also between adjacent DSP blocks.

Figure 7–5. Serial Loading 18-Bit 8-Tap FIR Filter Using Two DSP Blocks
Notes (1), (2), (3)



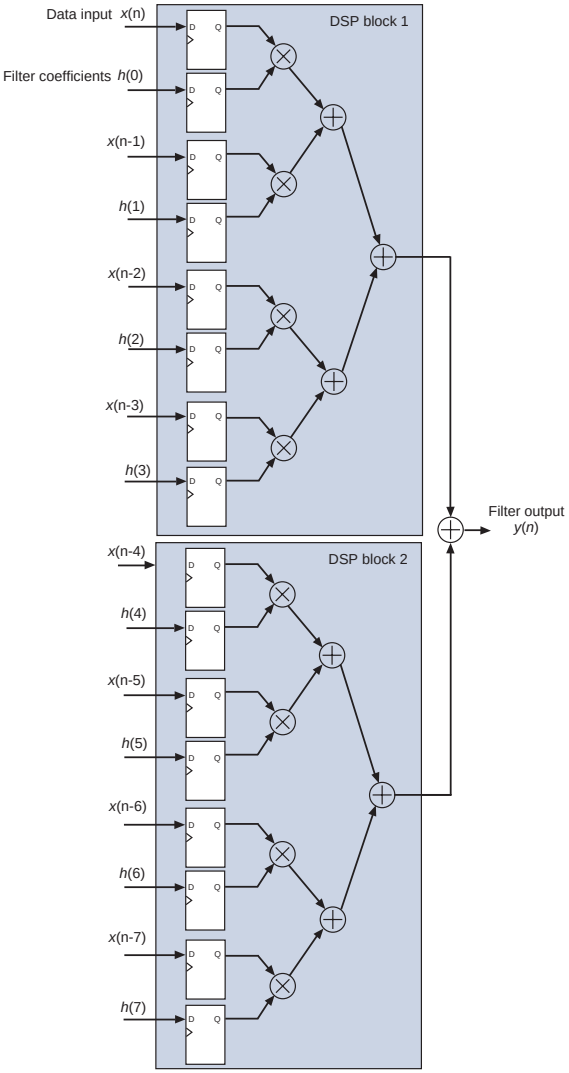
Notes to Figure 7–5:

(1) Unused ports grayed out.

(2) The indexing *x*(n-1), ..., *x*(n-7) refers to the case of parallel loading and should be ignored here. This indexing is retained in this figure for consistency with other figures in this chapter.

(3) To increase the DSP block performance, include the pipeline and output registers. See Figure 7–3 on page 7–8 for the details.

Figure 7–6. Parallel Loading 18-Bit 8-Tap FIR Filter Using Two DSP Blocks
Notes (1), (2)



Notes to Figure 7–6:

(1) The indexing $x(n\text{-}1)$, ..., $x(n\text{-}7)$ refers to the case of parallel loading.

(2) To increase the DSP block performance, include the input, pipeline, and output registers. See Figure 7–3 on page 7–8 for the details.

Basic FIR Filter Implementation Results

Table 7–6 shows the results of the serial implementation of an 18-bit 8 tap FIR filter as shown in Figure 7–5 on page 7–11

Table 7–6. Basic FIR Filter Implementation Results	
Part	EP1S10F780
Utilization	LCELL: 130/10570 (1%) DSP Block 9-bit elements: 16/48 (33%) Memory bits: 288/920448 (<1%)
Performance	247 MHz

Basic FIR Filter Design Example

Download the Basic FIR Filter (**base_fir.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

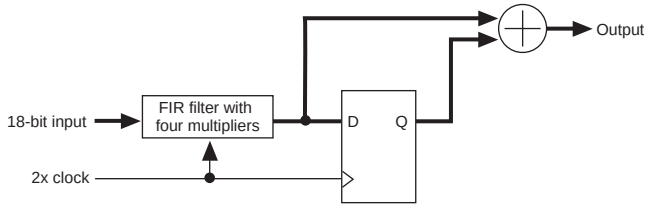
Time-Domain Multiplexed FIR Filters

A TDM FIR filter is clocked n-times faster than the sample rate in order to reuse the same hardware. Consider the 8-tap filter shown in Figure 7–2. The TDM technique can be used with a TDM factor of 2, i.e., n = 2, to implement this filter using only four multipliers, provided the filter is clocked two times faster internally.

To understand this concept, consider Figure 7–7 that shows a TDM filter with a TDM factor of 2. A 2×-multiplied clock is required to run the filter. On cycle 0 of the 2× clock, the user loads four coefficients into the four multiplier inputs. The resulting output is stored in a register. On cycle 1 of the 2× clock, the user loads the remaining four coefficients into the multiplier inputs. The output of cycle 1 is added with the output of cycle 0 to create the overall output. See the "TDM Filter Implementation" on page 7–14 section for details on the coefficient loading schedule.

The TDM implementation shown in Figure 7–7 requires only four multipliers to achieve the functionality of an 8-tap filter. Thus, TDM is a good way to save logic resources, provided the multipliers can run at n-times the clock speed. The coefficients can be stored in ROM/RAM, or any other muxing scheme.

Figure 7–7. Block Diagram of 8-Tap FIR Filter with TDM Factor of n=2



TDM Filter Implementation

TDM FIR filters are implemented in Stratix and Stratix GX devices by configuring the DSP blocks in the multiplier-adder mode. Figure 7–9 shows the implementation of an 8-tap TDM FIR filter (n=2) with 18 bits of data and coefficient inputs. Because the input data needs to be loaded into the DSP block in parallel, a shift register chain is implemented using a combination of logic cells and the altshift_taps function. This shift register is clocked with the same data sample rate (clock 1×). The filter coefficients are stored in ROM and loaded into the DSP block in parallel as well. Because the TDM factor is 2, both the ROM and DSP block are clocked with clock 2×.

Table 7–7. Operation of TDM Filter (Shown in Figure 7–9 on page 7–16)

Cycle of 2× Clock	Cycle Output	Operation	Overall Output, y(n)
0	$y_0 = x(n\text{-}1)h(1) + x(n\text{-}3)h(3) + x(n\text{-}5)h(5) + x(n\text{-}7)h(7)$	Store result	N/A
1	$y_1 = x(n)h(0) + x(n\text{-}2)h(2) + x(n\text{-}4)h(4) + x(n\text{-}6)h(6)$	Generate output	$y(n) = y_0 + y_1$
2	$y_2 = x(n)h(1) + x(n\text{-}2)h(3) + x(n\text{-}4)h(5) + x(n\text{-}6)h(7)$	Store result	N/A
3	$y_3 = x(n\text{+}1)h(0) + x(n\text{-}1)h(2) + x(n\text{-}3)h(4) + x(n\text{-}5)h(6)$	Generate output	$y(n) = y_2 + y_3$
4	$y_4 = x(n\text{+}1)h(1) + x(n\text{-}1)h(3) + x(n\text{-}3)h(5) + x(n\text{-}5)h(7)$	Store result	N/A
5	$y_5 = x(n\text{+}2)h(0) + x(n)h(2) + x(n\text{-}2)h(4) + x(n\text{-}4)h(6)$	Generate output	$y(n) = y_4 + y_5$
6	$y_6 = x(n\text{+}2)h(1) + x(n)h(3) + x(n\text{-}2)h(5) + x(n\text{-}4)h(7)$	Store result	N/A
7	$y_7 = x(n\text{+}3)h(0) + x(n\text{+}1)h(2) + x(n\text{-}1)h(4) + x(n\text{-}3)h(6)$	Generate output	$y(n) = y_6 + y_7$

Figure 7–8 and Table 7–7 show the coefficient loading schedule. For example, during cycle 0, only the flip-flops corresponding to $h(1)$, $h(3)$, $h(5)$, and $h(7)$ are enabled. This produces the temporary output, y_0, which is stored in a flip-flop outside the DSP block. During cycle 1, only the flip-

flops corresponding to *h*(0), *h*(2), *h*(4) and *h*(6) are enabled. This produces the temporary output, y_1, which is added to y_0 to produce the overall output, *y*(n). The following shows what the overall output, *y*(n), equals:

$$y(n) = y_0 + y_1$$

$$y(n) = x(0)h(0) + x(n-1)h(1) + x(n-2)h(2) + x(n-3)h(3) + x(n-4)h(4) + x(n-5)h(5) + x(n-6)h(6) + x(n-7)h(7)$$

This is identical to the output of the 8-tap filter shown in Figure 7–2. After cycle 1, this process is repeated at every cycle.

Figure 7–8. Coefficient Loading Schedule in a TDM Filter

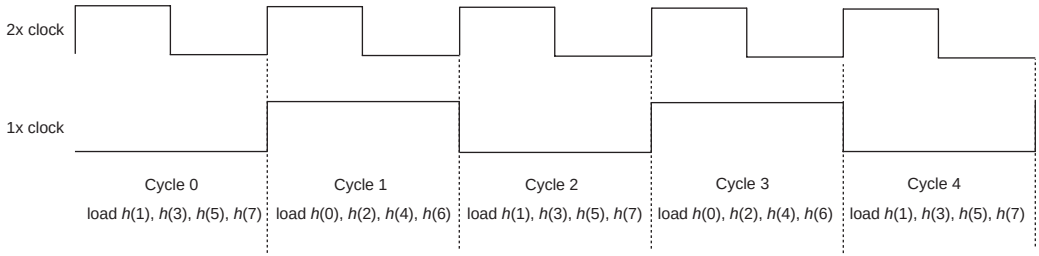
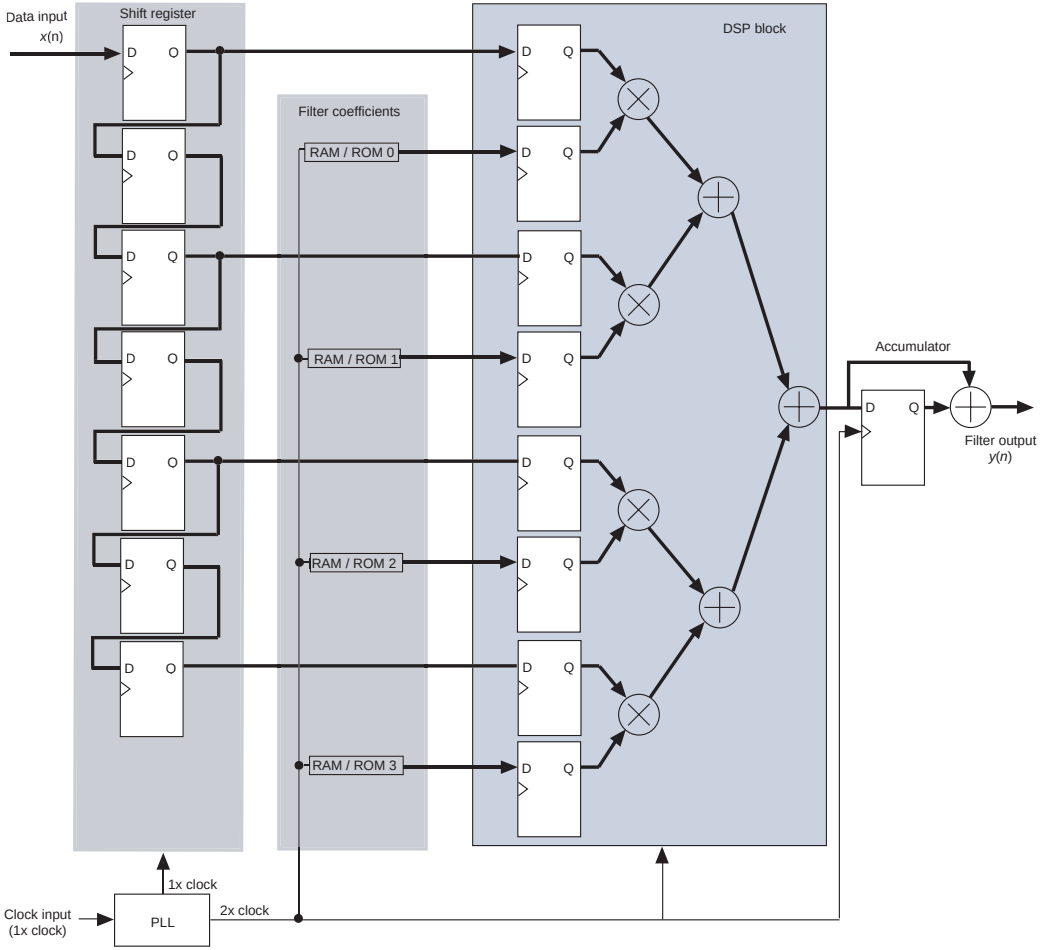


Figure 7–9. TDM FIR Filter Implementation *Note (1)*



Note to Figure 7–9:

(1) To increase the DSP block performance, include the pipeline and output registers. See Figure 7–3 on page 7–8 for details.

If the TDM factor is more than 2, then a multiply-accumulator needs to be implemented. This multiply-accumulator can be implemented using the soft logic outside the DSP block if all the multipliers of the DSP block are needed. Alternatively, the multiply-accumulator may be implemented inside the DSP block if all the multipliers of the DSP block are not needed. The accumulator needs to be zeroed at the start of each new sample input. The user also needs a way to store additional sample inputs in memory. For example, consider a sample rate of r and TDM factor of 4. Then, the

user needs a way to accept this sample data and send it at a 4r rate to the input of the DSP block. One way to do this is using a first-in-first-out (FIFO) memory with input clocked at rate r and output clocked at rate 4r. The FIFO may be implemented in the TriMatrix memory.

TDM Filter Implementation Results

Table 7–8 shows the results of the implementation of an 18-bit 8-tap TDM FIR filter as shown in Figure 7–9 on page 7–16.

Table 7–8. TDM Filter Implementation Results

Part	EP1S10F780
Utilization	Lcell: 196/10570 (1%) DSP Block 9-bit elements: 8/48 (17%) Memory bits: 360/920448 (<1%)
Performance	240 MHz *(1)*

Note to Table 7–8:

(1) This refers to the performance of the DSP blocks. The input and output rate is 120 million samples per second (MSPS), clocked in and out at 120 MHz.

TDM Filter Design Example

Download the TDM FIR Filter (**tdm_fir.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Polyphase FIR Interpolation Filters

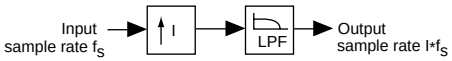
An interpolation filter can be used to increase sample rate. An interpolation filter is efficiently implemented with a polyphase FIR filter. DSP systems frequently use polyphase filters because they simplify overall system design and also reduce the number of computations per cycle required of the hardware. This section first describes interpolation filters and then how to implement them as polyphase filters in Stratix and Stratix GX devices. See the "Polyphase FIR Decimation Filters" on page 7–24 section for a discussion of decimation filters.

Interpolation Filter Basics

An interpolation filter increases the output sample rate by a factor of I through the insertion if I-1 zeros between input samples, a process known as zero padding. After the zero padding, the output samples in time domain are separated by $Ts / I = 1 / (I \times f_s)$, where T_s and f_s are the sample period and sample frequency of the original signal, respectively. Figure 7–10 shows the concept of signal interpolation.

Inserting zeros between the samples creates reflections of the original spectrum, thus, a low pass filter is needed to filter out the reflections.

Figure 7–10. Block Diagram Representation of Interpolation



To see how interpolation filters work, consider the Nyquist Sampling Theorem. This theorem states that the maximum frequency of the input to be sampled must be smaller than $f_S/2$, where f_S is the sampling frequency, to avoid aliasing. This frequency, $f_S/2$, is also known as the Nyquist frequency (F_n). Typically, before a signal is sampled using an analog to digital converter (ADC), it needs to be low pass filtered using an analog anti-aliasing filter to prevent aliasing. If the input frequency spectrum extends close to the Nyquist frequency, then the first alias is also close to the Nyquist frequency. Therefore, the low pass filter needs to be very sharp to reject this alias. A very sharp analog filter is hard to design and manufacture and could increase passband ripple, thereby compromising system performance.

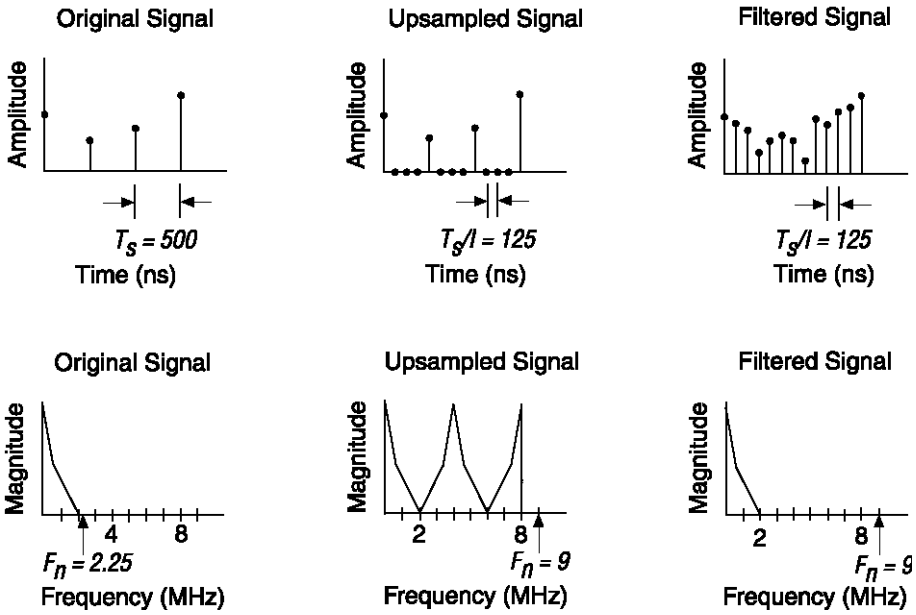
The solution is to increase the sampling rate of the ADC, so that the new Nyquist frequency is higher and the spacing between the desired signal and the alias is also higher. Zero padding as described above increase the sample rate. This process also known as upsampling (oversampling) relaxes the roll off requirements of the anti-aliasing filter. Consequently, a simpler filter achieves alias suppression. A simpler analog filter is easier to implement, does not compromise system performance, and is also easier to manufacture.

Similarly, the digital to analog converter (DAC) typically interpolates the data before the digital to analog conversion. This relaxes the requirement on the analog low pass filter at the output of the DAC.

The interpolation filter does not need to run at the oversampled (upsampled) rate of $f_S \times I$. This is because the extra sample points added are zeros, so they do not contribute to the output.

Figure 7–11 shows the time and frequency domain representation of interpolation for a specific case where the original signal spectrum is limited to 2 MHz and the interpolation factor (I) is 4. The Nyquist frequency of the upsampled signal must be greater than 8 MHz, and is chosen to be 9 MHz for this example.

Figure 7–11. Time & Frequency Domain Representations of Interpolation for I = 4



As an example, CD players use interpolation, where the nominal sample rate of audio input is 44.1 kilosamples per second. A typical implementation might have an interpolation (oversampling) factor of 4 generating 176.4 kilosamples per second of oversampled data stream.

Polyphase Interpolation Filters

A direct implementation of an interpolation filter, as shown in Figure 7–10, imposes a high computational burden. For example, if the filter is 16 taps long and a multiplication takes one cycle, then the number of computations required per cycle is 16× I. Depending on the interpolation factor (I), this number can be quite big and may not be achievable in hardware. A polyphase implementation of the low pass filter can reduce the number of computations required per cycle, often by a large factor, as will be evident later in this section.

The polyphase implementation "splits" the original filter into I polyphase filters whose impulse responses are defined by the following equation:

$$h_k(n) = h(k + nI)$$

where:

$k = 0,1, \ldots, I-1$
$n = 0,1, \ldots, P-1$
$P = L / I$ = length of polyphase filters
L = length of the filter (selected to be a multiple of I)
I = interpolation factor
$h(n)$ = original filter impulse response

This equation implies that the first polyphase filter, $h_0(n)$, has coefficients $h(0)$, $h(I)$, $h(2I)$,..., $h((P-1)I)$. The second polyphase filter, $h_1(n)$, has coefficients $h(1)$, $h(1+I)$, $h(1+2I)$, ..., $h(1+(P-1)I)$. Continuing in this way, the last polyphase filter, $hI_{-1}(n)$, has coefficients $h(I-1)$, $h((I-1)+N)$, $h((I-1)+2I)$, ..., $h((I-1)+(P-1)I)$.

An example helps in understanding the polyphase implementation of interpolation. Consider the polyphase representation of a 16-tap low pass filter with an interpolation factor of 4. Thus, the output is given below:

$$y(n) = \sum_{i=0}^{15} h(n - iI)x(i)$$

Referring back to Figure 7–11 on page 7–19, the only nonzero samples of the input are $x(0)$, $x(4)$, $x(8,)$ and $x(12)$. The first output, $y(0)$, only depends on $h(0)$, $h(4)$, $h(8)$ and $h(12)$ because $x(i)$ is zero for $i \neq 0, 4, 8, 12$. Table 7–9 shows the coefficients required to generate output samples.

Table 7–9. Decomposition of a 16-Tap Interpolating Filter into Four Polyphase Filters

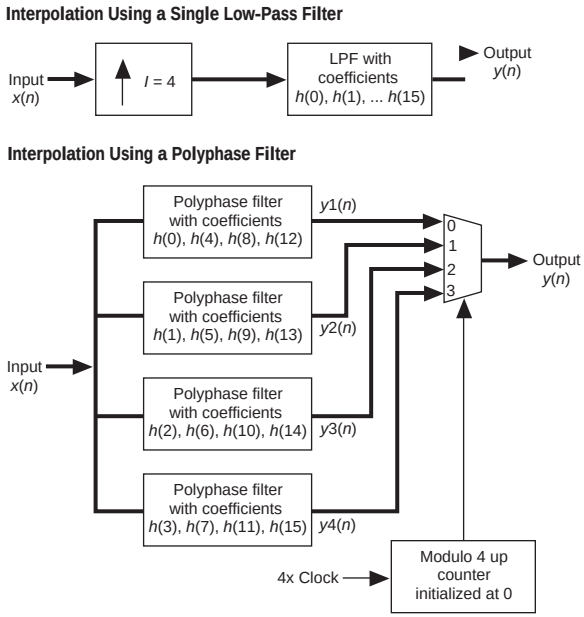
Output Sample	Coefficients Required	Polyphase Filter Impulse Response
y(0), y(4)...	h(0), h(4), h(8), h(12)	$h_0(n)$
y(1), y(5)...	h(1), h(5), h(9), h(13)	$h_1(n)$
y(2), y(6)...	h(2), h(6), h(10), h(14)	$h_2(n)$
y(3), y(7)...	h(3), h(7), h(11), h(15)	$h_3(n)$

Table 7–9 shows that this filter operation can be represented by four parallel polyphase filters. This is shown in Figure 7–12. The outputs from the filters are multiplexed to generate the overall output. The multiplexer is controlled by a counter, which counts up modulo-I starting at 0.

It is illuminating to compare the computational requirements of the direct implementation versus polyphase implementation of the low pass filter. In the direct implementation, the number of computations per cycle

required is 16 × I = 16 × 4 = 64. In the polyphase implementation, the number of computations per cycle required is 4 × 4 = 16. This is because there are four polyphase filters, each with four taps.

Figure 7–12. Polyphase Representation of I=4 Interpolation Filter



Polyphase Interpolation Filter Implementation

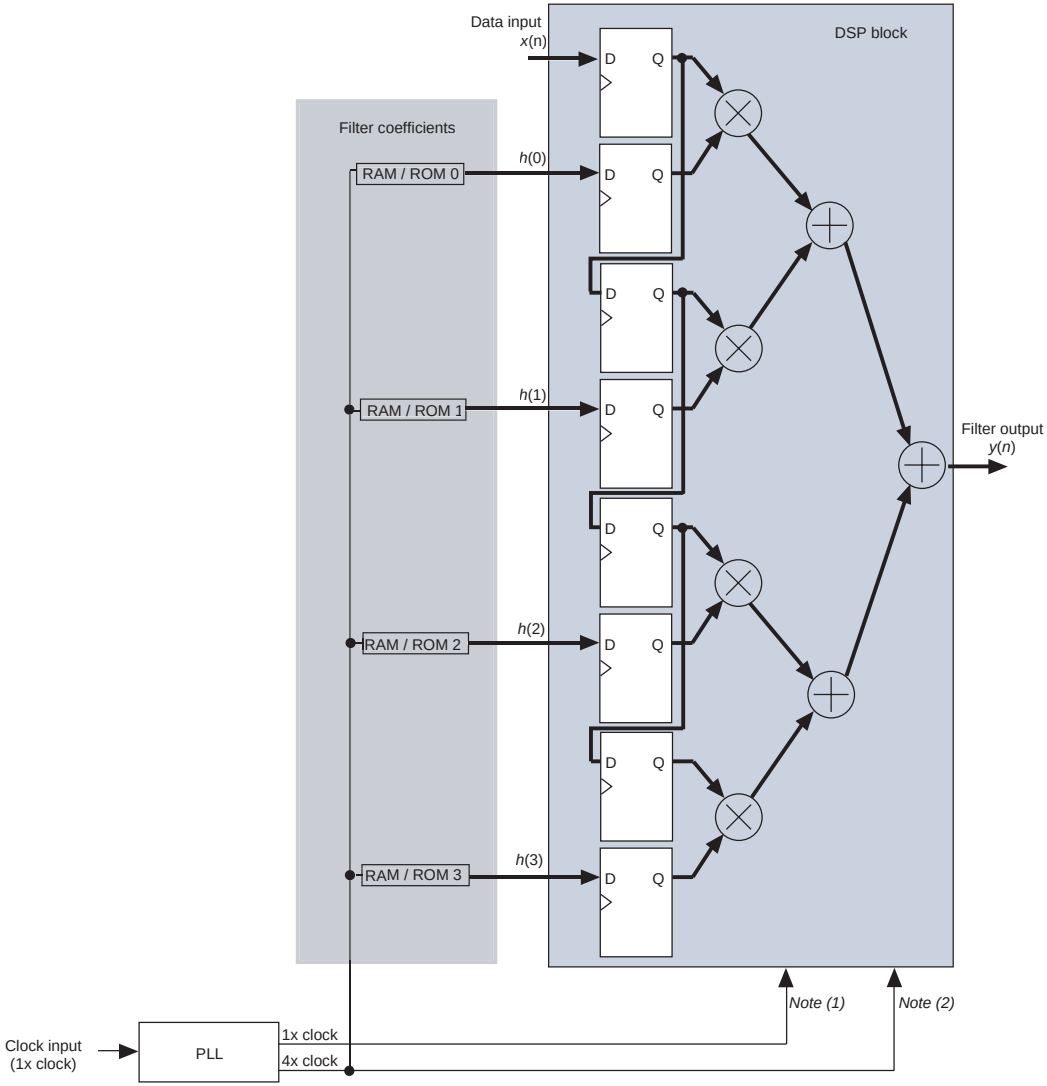
Figure 7–13 shows the Stratix or Stratix GX implementation of the polyphase interpolation filter in Figure 7–12. The four polyphase filters share the same hardware, which is a 4-tap filter. One Stratix or Stratix GX DSP block can implement one 4-tap filter with 18-bit wide data and coefficients. A multiplexer can be used to load new coefficient values on every cycle of the 4× clock. Stratix and Stratix GX phase lock loops (PLLs) can generate the 4× clock. In the first cycle of the 4× clock, the user needs to load coefficients for polyphase filter $h_0(n)$; in the second cycle of the 4×

clock, the users needs to load coefficients of the polyphase filter $h_1(n)$ and so on. Table 7–10 summarizes the coefficient loading schedule. The output, $y(n)$, is clocked using the 4× clock.

Table 7–10. Polyphase Interpolation (I=4) Filter Coefficient Loading Schedule

Cycle of 4× Clock	Coefficients to Load	Corresponding RAM/ROM
1, 5,...	h(0), h(4), h(8), h(12)	0, 1, 2, 3
2, 6,...	h(1), h(5), h(9), h(13)	0, 1, 2, 3
3, 7,...	h(2), h(6), h(10), h(14)	0, 1, 2, 3
4, 8,...	h(3), h(7), h(11), h(15)	0, 1, 2, 3

Figure 7–13. Implementation of the Polyphase Interpolation Filter (I=4) *Notes (1), (2), (3)*



Notes to Figure 7–13:

(1) The 1× clock feeds the input data shiftin register chain.

(2) The 4× clock feeds the input registers for the filter coefficients and other optional registers in the DSP block. See *Note (3)*.

(3) To increase the DSP block performance, include the pipeline, and output registers. See Figure 7–3 for the details.

Polyphase Interpolation Filter Implementation Results

Table 7–11 shows the results of the polyphase interpolation filter implementation in a Stratix device shown in Figure 7–13.

Table 7–11. Polyphase Interpolation Filter Implementation Results

Part	EP1S10F780
Utilization	Lcell: 3/10570 (<1%) DSP Block 9-bit elements: 8/48 (17%) Memory bits: 288/920448 (<1%)
Performance	240 MHz *(1)*

Note to Table 7–11:

(1) This refers to the performance of the DSP blocks, as well as the output clock rate. The input rate is 60 MSPS, clocked in at 60MHz.

Polyphase Interpolation Filter Design Example

Download the Interpolation FIR Filter (**interpolation_fir.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Polyphase FIR Decimation Filters

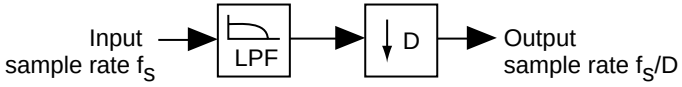
A decimation filter can be used to decrease the sample rate. A decimation filter is efficiently implemented with a polyphase FIR filter. DSP systems frequently use polyphase filters because they simplify overall system design and also reduce the number of computations per cycle required of the hardware. This section first describes decimation filters and then how to implement them as polyphase filters in Stratix devices. See the "Polyphase FIR Interpolation Filters" section for a discussion of interpolation filters.

Decimation Filter Basics

A decimation filter decreases the output sample rate by a factor of D through keeping only every D-th input sample. Consequently, the samples at the output of the decimation filter are separated by $D \times T_s = D / f_s$, where T_s and f_s are the sample period and sample frequency of the original signal, respectively. Figure 7–14 shows the concept of signal decimation.

The signal needs to be low pass filtered before downsampling can begin in order to avoid the reflections of the original spectrum from being aliased back into the output signal.

Figure 7–14. Block Diagram Representation of Decimation

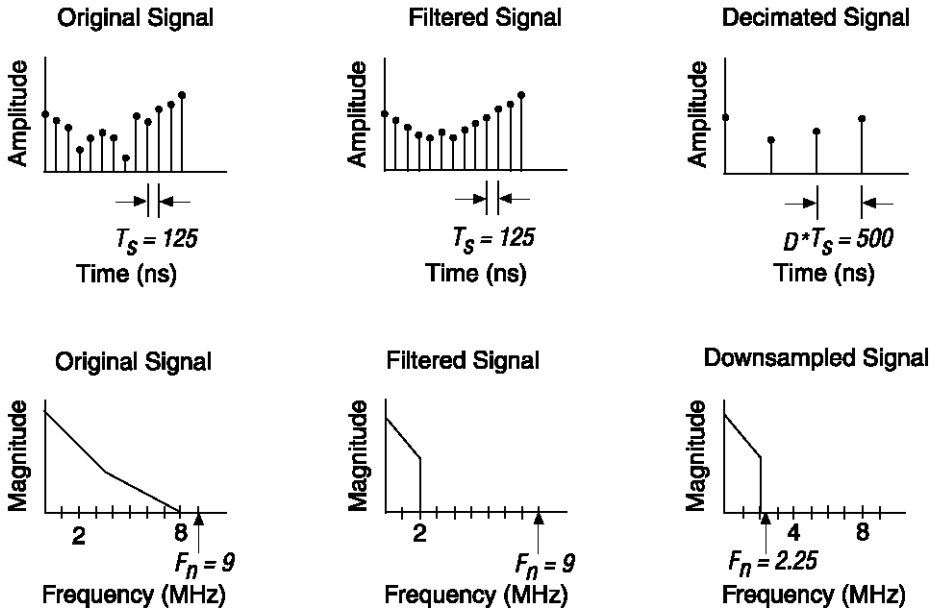


Decimation filters reverse the effect of the interpolation filters. Before the decimation process, a low pass filter is applied to the signal to attenuate noise and aliases present beyond the Nyquist frequency. The filtered signal is then applied to the decimation filter, which processes every D-th input. Therefore the values between samples D, D-1, D-2 etc. are ignored. This allows the filter to run M times slower than the input data rate.

In a typical system, after the analog to digital conversion is complete, the data needs to be filtered to remove aliases inherent in the sampled data. Further, at this point there is no need to continue to process this data at the higher sample (oversampled) rate. Therefore, a decimation FIR filter at the output of the ADC lowers the data rate to a value that can be processed digitally.

Figure 7–15 shows a specific example where a signal spread over 8 MHz is decimated by a factor of 4 to 2 MHz. The Nyquist frequency of the downsampled signal must be greater than 2 MHz, and is chosen to be 2.25 MHz in this example.

Figure 7–15. Time & Frequency Domain Representations of Decimation for D=4



Polyphase Decimation Filters

Figure 7–14 shows a direct implementation of a decimation filter, which imposes a high computational burden. For example, if the filter is 16 taps long and a multiplication takes one cycle, the number of computations required per cycle is 16× D. Depending on the decimation factor (D), this number can be quite big and may not be achievable in hardware. A polyphase implementation of the low pass filter can reduce the number of computations required, often by a large ratio, as will be evident later in this section.

The polyphase implementation "splits" the original filter into D polyphase filters with impulse responses defined by the following equation.

$$h_k(n) = h(k + nD)$$

where:

$k = 0,1, \ldots, D-1$
$n = 0,1, \ldots, P-1$
$P = L / D$ = length of polyphase filters
L is the length of the filter (selected to be a multiple of D)
D is the decimation factor
$h(n)$ is the original filter impulse response

This equation implies that the first polyphase filter, $h_0(n)$, has coefficients $h(0)$, $h(D)$, $h(2D)\ldots h((P-1)D)$. The second polyphase filter, $h_1(n)$, has coefficients $h(1)$, $h(1+D)$, $h(1+2D)$, ..., $h(1+(P-1)D)$. Continuing in this way, the last polyphase filter, $h_{D-1}(n)$ has coefficients $h(D-1)$, $h((D - 1) + D)$, $h((D - 1) + 2D)$, ..., $h((D - 1) + (P-1)D)$.

An example helps in the understanding of the polyphase implementation of decimation. Consider the polyphase representation of a 16-tap low pass filter with a decimation factor of 4. The output is given by:

$$y(n) = \sum_{i=0}^{15} h(i)x(nD - i)$$

Referring to Figure 7–15 on page 7–26, it is clear that the output, $y(n)$ is discarded for $n \neq 0, 4, 8, 12$, hence the only values of $y(n)$ that need to be computed are $y(0)$, $y(4)$, $y(8)$, $y(12)$. Table 7–12 shows which coefficients are required to generate the output samples.

Table 7–12. Decomposition of a 16-Tap Decimation Filter into Four Polyphase Filters

Output Sample *(1)*	Coefficients Required	Polyphase Filter Impulse Response
$y(0)_0, y(4)_0, \ldots$	h(0), h(4), h(8), h(12)	$h_0(n)$
$y(0)_1, y(4)_1, \ldots$	h(1), h(5), h(9), h(13)	$h_1(n)$
$y(0)_2, y(4)_2, \ldots$	h(2), h(6), h(10), h(14)	$h_2(n)$
$y(0)_3, y(4)_3, \ldots$	h(3), h(7), h(11), h(15)	$h_3(n)$

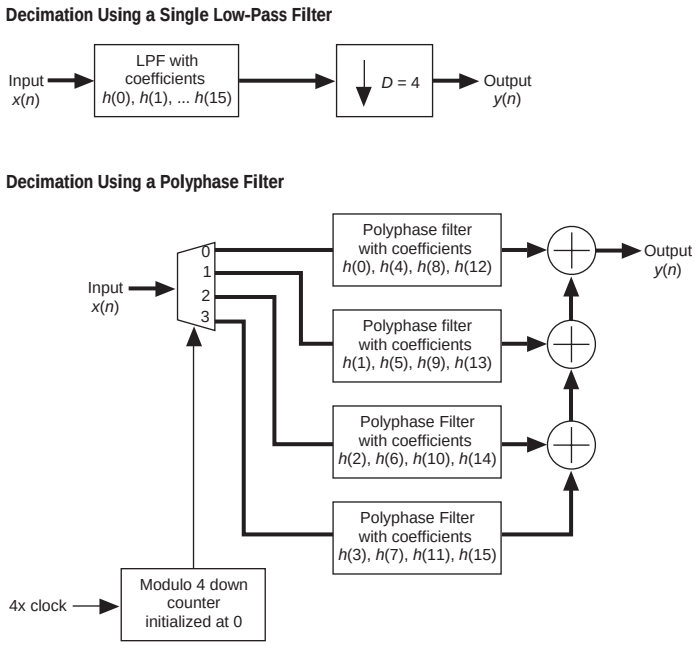
Note to Table 7–12:
(1) The output sample is the sum of the results from four polyphase filters: $y(n) = y(n)_0 + y(n)_1 + y(n)_2 + y(n)_3$.

Table 7–12 shows that the overall decimation filter operation can be represented by 4 parallel polyphase filters. Figure 7–16 shows the polyphase representation of the decimation filter. A demultiplexer at the input ensures that the input is applied to only one polyphase filter at a

time. The demultiplexer is controlled by a counter, which counts down modulo-D starting at 0. The overall output is taken by adding the outputs of all the filters.

The polyphase representation of the decimation filter also reduces the computational requirement. For the example in Figure 7–16, the direct implementation requires $16 \times D = 16 \times 4 = 64$ computations per cycle, whereas the polyphase implementation requires only $4 \times 4 \times 1 = 16$ computations per cycle. This saving in computational complexity is quite significant and is often a very convincing reason to use polyphase filters.

Figure 7–16. Polyphase Filter Representation of a D=4 Decimation Filter



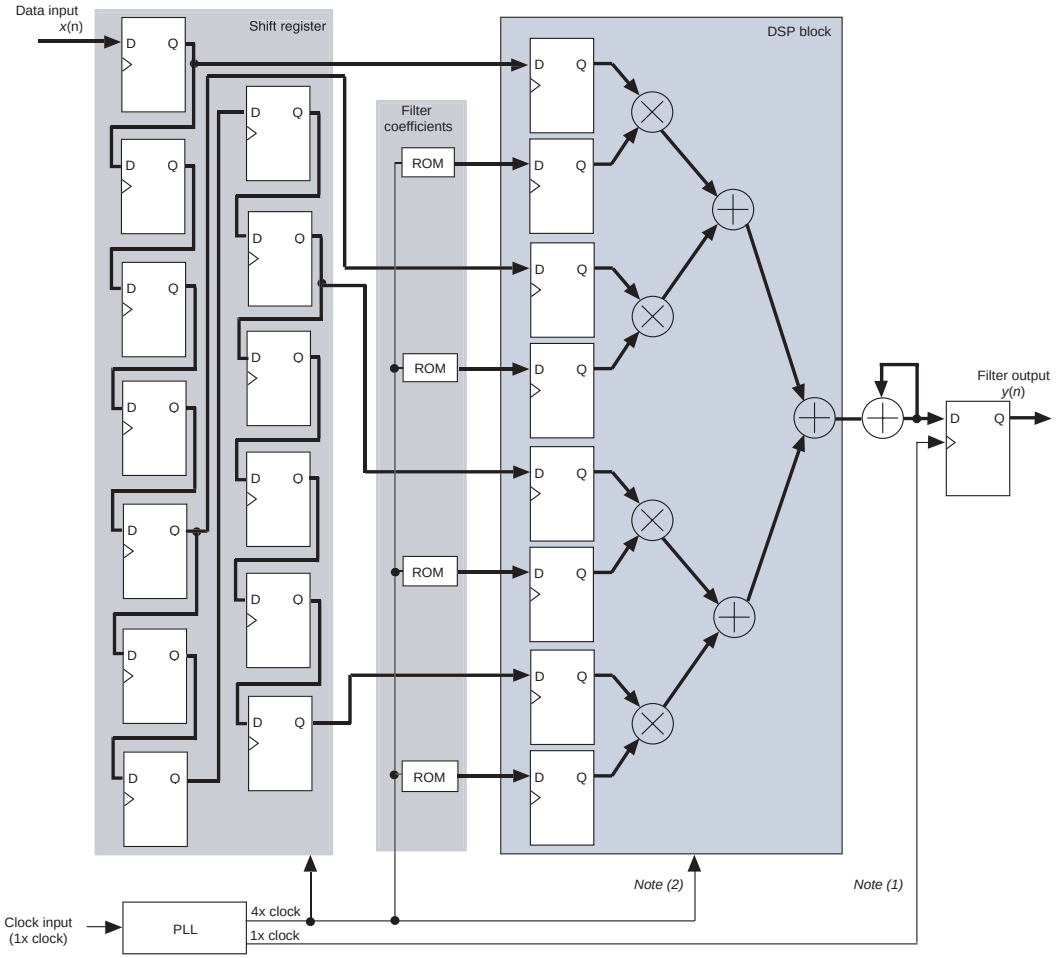
Polyphase Decimation Filter Implementation

Figure 7–17 shows the decimation polyphase filter example of Figure 7–16 as it would fit into Stratix or Stratix GX DSP blocks. The coefficients of the polyphase filters need to be cycled using the schedule shown in Table 7–13. The output *y(n)*, is clocked using the 1× clock.

Table 7–13. Coefficient Loading Schedule for Polyphase Decimation Filter (D=4)

Cycle of 4× Clock	Coefficients to Load	Corresponding RAM/ROM
1, 5,...	h(0), h(4), h(8), h(12)	0, 1, 2, 3
2, 6,...	h(3), h(7), h(11), h(15)	0, 1, 2, 3
3, 7,...	h(2), h(6), h(10), h(14)	0, 1, 2, 3
4, 8,...	h(1), h(5), h(9), h(13)	0, 1, 2, 3

Figure 7–17. Implementation of the Polyphase Decimation Filter (D=4) *Notes (1), (2), (3)*



Notes to Figure 7–17:

(1) The 1× clock feeds the register after the accumulator block.

(2) The 4× clock feeds the shift register for the data, the input registers for both the data and filter coefficients, the other optional registers in the DSP block (see *Note (3)*), and the accumulator block.

(3) To increase the DSP block performance, include the pipeline, and output registers. See Figure 7–3 on page 7–8 for the details.

Polyphase Decimation Filter Implementation Results

Table 7–14 shows the results of the polyphase decimation filter implementation in a Stratix device shown in Figure 7–17.

Table 7–14. Polyphase Decimation Filter Implementation Results

Part	EP1S10F780
Utilization	Lcell: 168/10570 (1%) DSP Block 9-bit elements: 8/48 (17%) Memory bits: 300/920448(<1%)
Performance	240 MHz *(1)*

Note to Table 7–14:

(1) This refers to the performance of the DSP blocks, as well as the input clock rate. The output rate is 60 MSPS (clocked out at 60MHz).

Polyphase Decimation Filter Design Example

Download the Decimation FIR Filter (**decimation_fir.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Complex FIR Filter

A complex FIR filter takes real and imaginary input signals and performs the filtering operation with real and imaginary filter coefficients. The output also consists of real and imaginary signals. Therefore, a complex FIR filter is similar to a regular FIR filter except for the fact that the input, output, and coefficients are all complex numbers.

One example application of complex FIR filters is equalization. Consider a Phase Shift Keying (PSK) system; a single complex channel can represent the I and Q data channels. A FIR filter with complex coefficients could then process both data channels simultaneously. The filter coefficients are chosen in order to reverse the effects of intersymbol interference (ISI) inherent in practical communication channels. This operation is called equalization. Often, the filter is adaptive, i.e. the filter coefficients can be varied as desired, to optimize performance with varying channel characteristics.

A complex variable FIR filter is a cascade of complex multiplications followed by a complex addition. Figure 7–18 shows a block diagram representation of a complex FIR filter. To compute the overall output of the FIR filter, it is first necessary to determine the output of each complex multiplier. This can be expressed as:

$$y_{real} = x_{real} \times h_{real} - x_{imag} \times h_{imag}$$
$$y_{imag} = x_{real} \times h_{imag} + h_{real} \times x_{imag}$$

where:

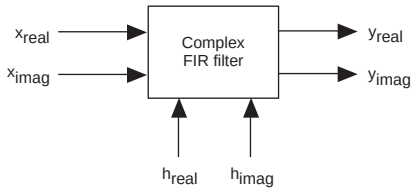
x_{real} is the real input signal
x_{imag} is the imaginary input signal
h_{real} is the real filter coefficients
h_{imag} is the imaginary filter coefficients
y_{real} is the real output signal
y_{imag} is the imaginary output signal

In complex representation, this equals:

$$y_{real} + jy_{imag} = (x_{real} + jx_{imag}) \times (h_{real} + jh_{imag})$$

The overall real channel output is obtained by adding the real channel outputs of all the multipliers. Similarly, the overall imaginary channel output is obtained by adding the imaginary channel outputs of all the multipliers.

Figure 7–18. Complex FIR Filter Block Diagram



Complex FIR Filter Implementation

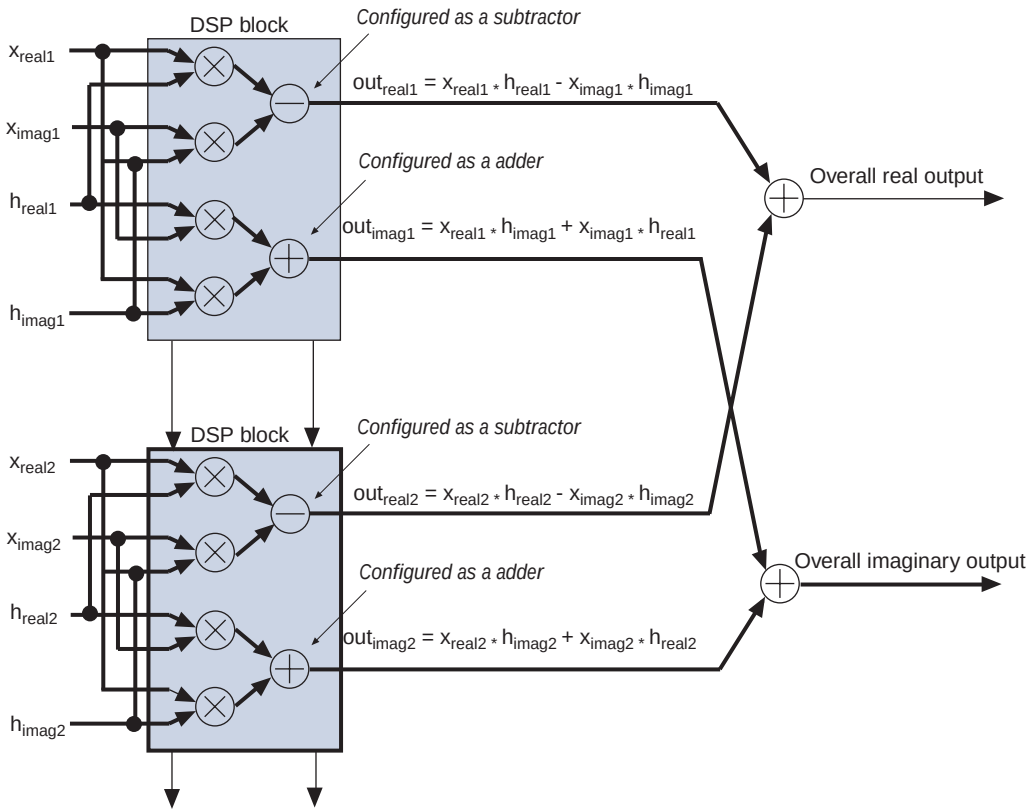
Complex filters can be easily implemented in Stratix devices with the DSP blocks configured in the two-multipliers adder mode. One DSP block can implement a 2-tap complex FIR filter with 9-bit inputs, or a single tap complex FIR filter with 18-bit inputs. DSP blocks can be cascaded to implement complex filters with more taps.

1 The two-multipliers adder mode has two adders, each adding the outputs of two multipliers. One of the adders is configured as a subtractor.

f For more information on the different modes of the DSP blocks, see the *DSP Blocks in Stratix & Stratix GX Devices* chapter.

Figure 7–19 shows an example of a 2-tap complex FIR filter design with 18-bit inputs. The real and the complex outputs of the DSP blocks are added externally to generate the overall real and imaginary output. As in the case of basic, TDM, or polyphase FIR filters, the coefficients may be loaded in series or parallel.

Figure 7–19. 2-Tap 18-Bit Complex FIR Filter Implementation



Infinite Impulse Response (IIR) Filters

Another class of digital filters are IIR filters. These are recursive filters where the current output is dependent on previous outputs. In order to maintain stability in an IIR filter, careful design consideration must be given, especially to the effects of word-length to avoid unbounded conditions. The following section discusses the general theory and applications behind IIR Filters.

IIR Filter Background

The impulse response of an IIR filter extends for an infinite amount of time because their output is based on feedback from previous outputs. The general expression for IIR filters is:

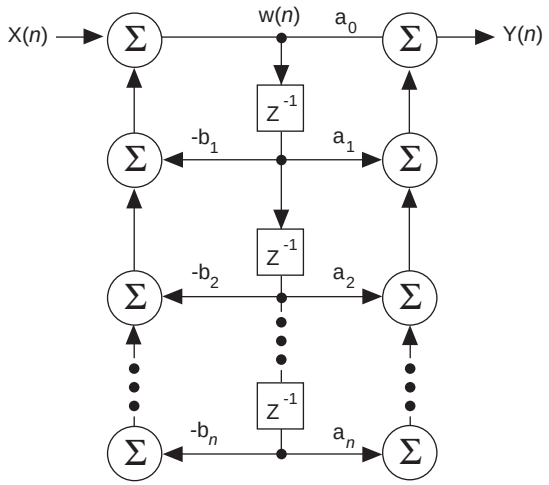
$$y(n) = \sum_{i=0}^{n} a(i)x(n-i) - \sum_{i=1}^{n} b(i)y(n-i)$$

where a_i and b_i represent the coefficients in the feed-forward path and feedback path, respectively, and n represents the filter order. These coefficients determine where the poles and zeros of the IIR filter lie. Consequently, they also determine how the filter functions (i.e., cut-off frequencies, band pass, low pass, etc.).

The feedback feature makes IIR filters useful in high data throughput applications that require low hardware usage. However, feedback also introduces complications and caution must be taken to make sure these filters are not exposed to situations in which they may become unstable. The complications include phase distortion and finite word length effects, but these can be overcome by ensuring that the filter always operates within its intended range.

Figure 7–20 shows a direct form II structure of an IIR filter.

Figure 7–20. Direct Form II Structure of an IIR Filter



The transfer function for an IIR filter is:

$$H(z) = \frac{\sum_{i=0}^{n} a_i z^{-i}}{1 + \sum_{i=1}^{n} b_i z^{-i}}$$

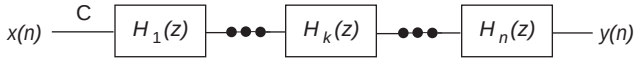
The numerator contains the zeros of the filter and the denominator contains the poles. The IIR filter structure requires a multiplication followed by an accumulation. Constructing the filter directly from the transfer function shown above may result in finite word length limitations and make the filter potentiality unstable. This becomes more critical as the filter order increases, because it only has a finite number of bits to represent the output. To prevent overflow or instability, the transfer function can be split into two or more terms representing several second order filters called biquads. These biquads can be individually scaled and cascaded, splitting the poles into multiples of two. For example, an IIR filter having ten poles should be split into five biquad sections. Doing this minimizes quantization and recursive accumulation errors.

This cascaded structure rearranges the transfer function. This is shown in the equation below, where each product term is a second order IIR filter. If n is odd, the last product term is a first order IIR filter:

$$H(z) = C \times \prod_{k=1}^{(n+1)/2} \frac{a_{0k} + a_{1k}z^{-1} + a_{2k}z^{-2}}{1 + b_{1k}z^{-1} + b_{2k}z^{-2}} = C \times \prod_{k=1}^{(n+1)/2} H_k(z)$$

Figure 7–21 shows the cascaded structure.

Figure 7–21. Cascaded IIR Filter



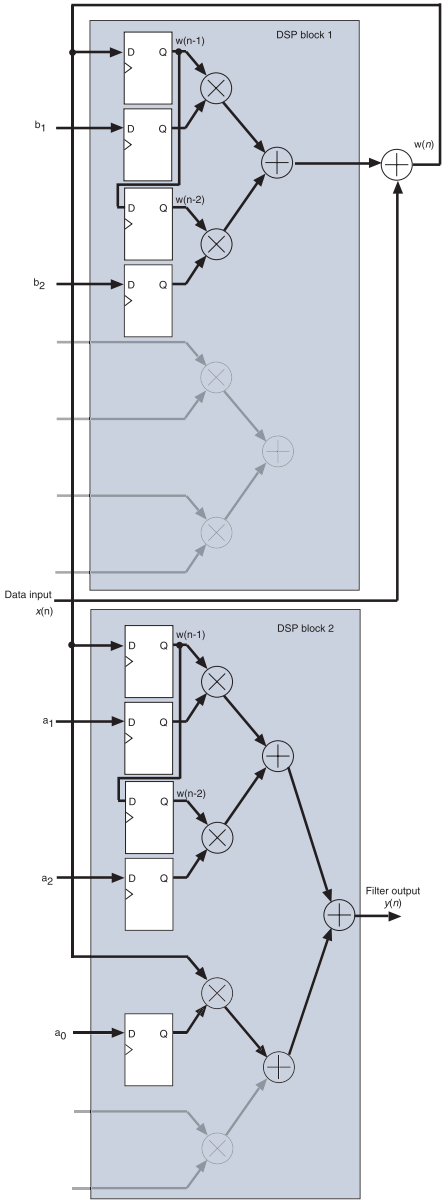
Basic IIR Filters

In this section, the basic IIR filter is implemented using cascaded second order blocks or biquads in the direct form II structure.

Basic IIR FIlter Implementation

Multiplier blocks, adders and delay elements can implement a basic IIR filter. The Stratix architecture lends itself to IIR filters because of its embedded DSP blocks, which can easily be configured to perform these operations. The `altmult_add` megafunction can be used to implement the multiplier-adder mode in the DSP blocks. Figure 7–22 shows the implementation of an individual biquad using Stratix and Stratix GX DSP blocks.

Figure 7–22. IIR Filter Biquad Note (1)

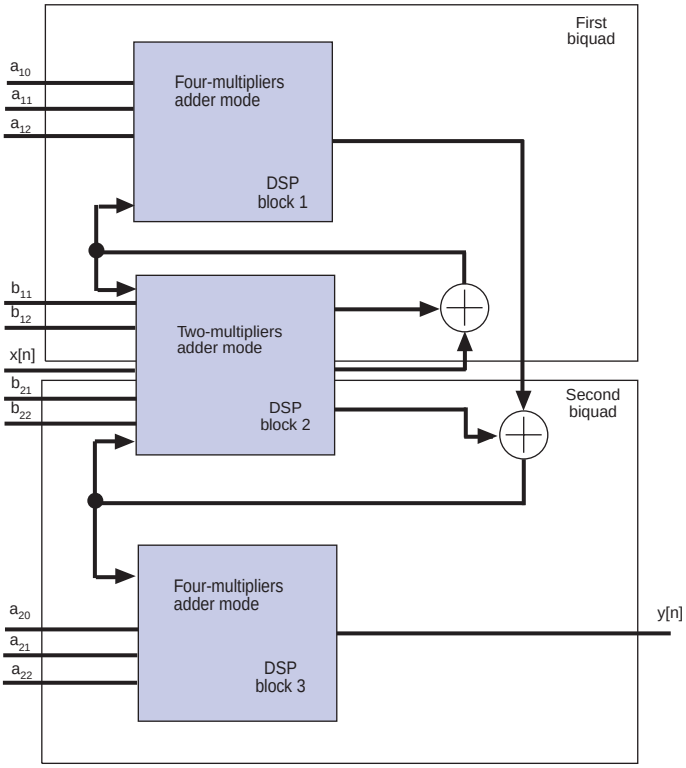


Note to Figure 7–22:

(1) Unused ports are grayed out.

The first DSP block in Figure 7–22 is configured in the two-multipliers adder mode, and the second DSP block is in the four-multipliers adder mode. For an 18-bit input to the IIR filter, each biquad requires five multipliers and five adders (two DSP blocks). One of the adders is implemented using logic elements. Cascading several biquads together can implement more complex, higher order IIR filters. It is possible to insert registers in between the biquad stages to improve the performance. Figure 7–23 shows a 4thorder IIR filter realized using two cascaded biquads in three DSP blocks.

Figure 7–23. Two Cascaded Biquads



Basic IIR Filter Implementation Results

Table 7–15 shows the results of implementing a 4th order IIR filter in a Stratix device.

Table 7–15. 4th Order IIR Filter Implementation Results

Part	EP1S10F780C5
Utilization	Lcell: 102/10570(<1%) DSP Block 9-bit elements: 24/48 (50%) Memory bits: 0/920448(0%)
Performance	73 MHz
Latency	4 clock cycles

Basic IIR Filter Design Example

Download the 4th Order IIR Filter (**iir.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Butterworth IIR Filters

Butterworth filters are the most popular version of IIR analog filters. These filters are also known as "maximally flat" because they have no passband ripple. Additionally, they have a monotonic response in both the stopband and the passband. Butterworth filters trade-off roll off steepness for their no ripple characteristic. The distinguishing Butterworth filter feature is its poles are arranged in a uniquely symmetrical manner along a circle in the *s*-plane. The expression for the Butterworth filter's magnitude-squared function is as follows:

$$\left|H_c(j\omega)\right|^2 = \frac{1}{1 + \left(\frac{j\omega}{j\omega_c}\right)^{2N}}$$

where:

ω_c is the cut-off frequency
N is the filter order

The filter's cutoff characteristics become sharper as *N* increases. If a substitution is made such that $j\omega = s$, then the following equation is derived:

$$H_c(s)H_c(-s) = \frac{1}{1 + \left(\frac{s}{j\omega_c}\right)^{2N}}$$

with poles at:

$$s_k = (-1)^{\frac{1}{2N}}(j\omega_c) \qquad \text{for } k=0,1,\ldots,2N-1$$
$$= \omega_c e^{\left(\frac{j\pi}{2N}\right)(2k+N-1)}$$

There are 2N poles on the circle with a radius of ω_c in the *s*-plane. These poles are evenly spaced at π/N intervals along the circle. The poles chosen for the implementation of the filter lie in the left half of the *s*-plane, because these generate a stable, causal filter.

Each of the impulse invariance, the bilinear, and matched *z* transforms can transform the Laplace transform of the Butterworth filter into the *z*-transform.

- Impulse invariance transforms take the inverse of the Laplace transform to obtain the impulse response, then perform a *z*-transform on the sampled impulse response. The impulse invariance method can cause some aliasing.
- The bilinear transform maps the entire $j\omega$-axis in the *s*-plane to one revolution of the unit circle in the *z*-plane. This is the most popular method because it inherently eliminates aliasing.
- The matched *z*-transform maps the poles and the zeros of the filter directly from the *s*-plane to the *z*-plane. Usually, these transforms are transparent to the user because several tools, such as MATLAB, exist for determining the coefficients of the filter. The *z*-transform generates the coefficients much like in the basic IIR filter discussed earlier.

Butterworth Filter Implementation

For digital designs, consideration must be made to optimize the IIR biquad structure so that it maps optimally into logic. Because speed is often a critical requirement, the goal is to reduce the number of operations per biquad. It is possible to reduce the number of multipliers needed in each biquad to just two.

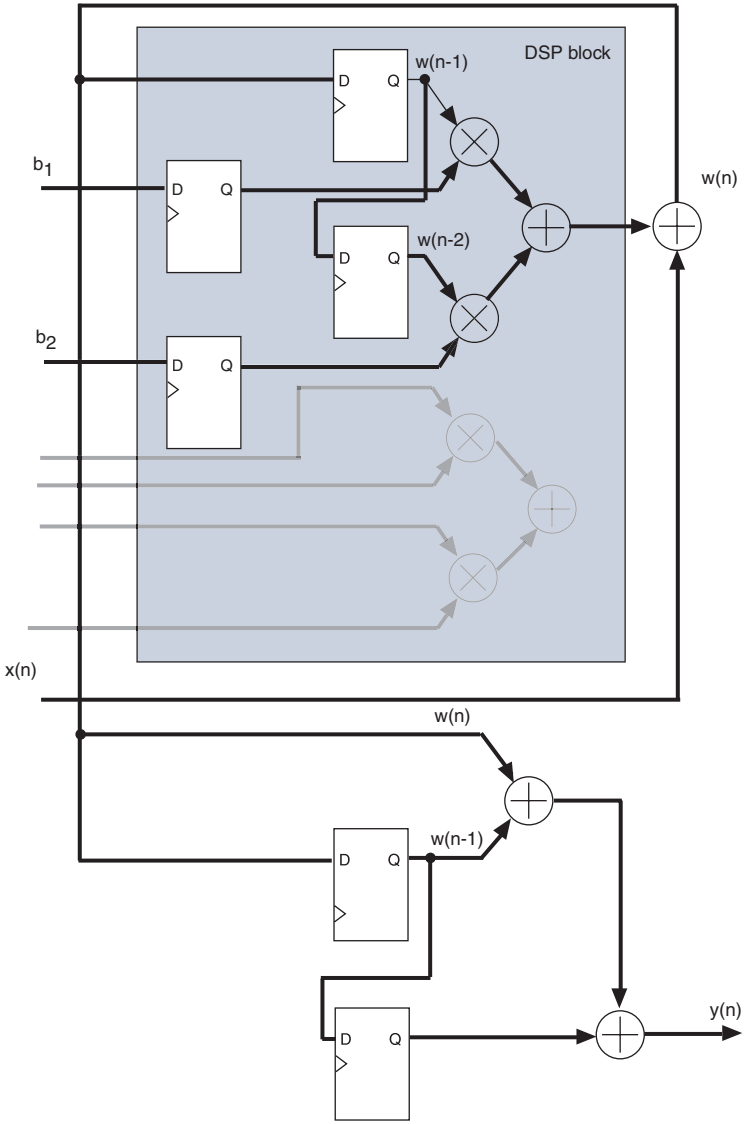
Through the use of integer feedforward multiplies, which can be implemented by combining addition, shifting, and complimenting operations, a Butterworth filter's transfer function biquad can be optimized for logic synthesis. The most efficient transformation is that of an all pole filter. This is because there is a unique relationship between the feedforward integer coefficients of the filter represented as:

$$H(z) = \frac{1 + 2z^{-1} + z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}}$$

As can be seen by this equation, the z^{-1} coefficient in the numerator (representing the feedforward path) is twice the other two operands (z^{-2} and 1). This is always the case in the transformation from the frequency to the digital domain. This represents the normalized response, which is faster and smaller to implement in hardware than real multipliers. It introduces a scaling factor as well, but this can be corrected at the end of the cascade chain through a single multiply.

Figure 7–24 shows how a Butterworth filter biquad is implemented in a Stratix or Stratix GX device.

Figure 7–24. Butterworth Filter Biquad Notes (1), (2)



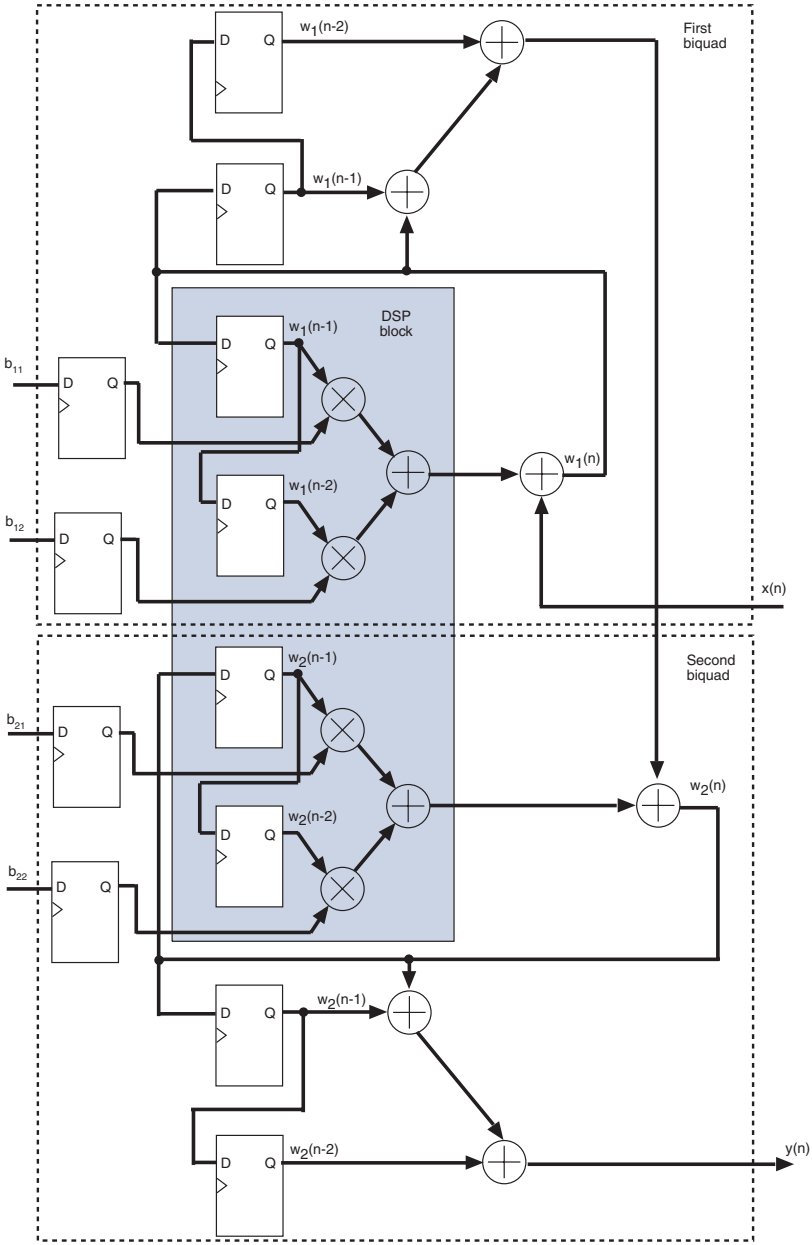
Notes to Figure 7–24:

(1) Unused ports are grayed out.

(2) The z^{-1} coefficient is a multiple of the other coefficients (z-2 and 1) in the feedforward path. This is implemented using a shift operation.

The DSP block in Figure 7–24 is configured in multiply and add mode. The three external adders are implemented in logic elements and therefore are not part of the DSP block. Therefore, for an 18-bit input, each biquad requires half a DSP block and three logic element adders. The gain factor can be compensated for at the end of the filtering stage and is not shown in this simple example. More complex, higher order Butterworth filters can be realized by cascading several biquads together, as in the IIR example. Figure 7–25 below shows a 4th order Butterworth filter using two cascaded biquads in a single DSP block.

Figure 7–25. Cascaded Butterworth Biquads *Note (1)*



Note to Figure 7–25:

(1) The gain factor is compensated for at the end of the filtering stage and is not shown in this figure.

Butterworth Filter Implementation Results

Table 7–16 shows the results of implementing a 4th order Butterworth filter as shown in Figure 7–25.

Table 7–16. 4th Order Butterworth Filter Implementation Results

Part	EP1S10F780C6
Utilization	Lcell: 251/10570(2%) DSP Block 9-bit elements: 16/48 (33%) Memory bits: 0/920448 (0%)
Performance	80 MHz
Latency	4 clock cycles

Butterworth Filter Design Example

Download the 4th Order Butterworth Filter (**butterworth.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Matrix Manipulation

DSP relies heavily on matrix manipulation. The key idea is to transform the digital signals into a format that can then be manipulated mathematically.

This section describes an example of matrix manipulation used in 2-D convolution filter, and its implementation in a Stratix device.

Background on Matrix Manipulation

A matrix can represent all digital signals. Apart from the convenience of compact notation, matrix representation also exploits the benefits of linear algebra. As with one-dimensional, discrete sequences, this advantage becomes more apparent when processing multi-dimensional signals.

In image processing, matrix manipulation is important because it requires analysis in the spatial domain. Smoothing, trend reduction, and sharpening are examples of common image processing operations, which are performed by convolution. This can also be viewed as a digital filter operation with the matrix of filter coefficients forming a convolutional kernel, or mask.

Two-Dimensional Filtering & Video Imaging

FIR filtering for video applications and image processing in general is used in many applications, including noise removal, image sharpening to feature extraction.

For noise removal, the goal is to reduce the effects of undesirable, contaminative signals that have been linearly added to the image. Applying a low pass filter or smoothing function flattens the image by reducing the rapid pixel-to-pixel variation in gray levels and, ultimately, removing noise. It also has a blurring effect usually used as a precursor for removing unwanted details before extracting certain features from the image.

Image sharpening focuses on the fine details of the image and enhances sharp transitions between the pixels. This acts as a high-pass filter that reduces broad features like the uniform background in an image and enhances compact features or details that have been blurred.

Feature extraction is a form of image analysis slightly different from image processing. The goal of image analysis in general is to extract information based on certain characteristics from the image. This is a multiple step process that includes edge detection. The easiest form of edge detection is the derivative filter, using gradient operators.

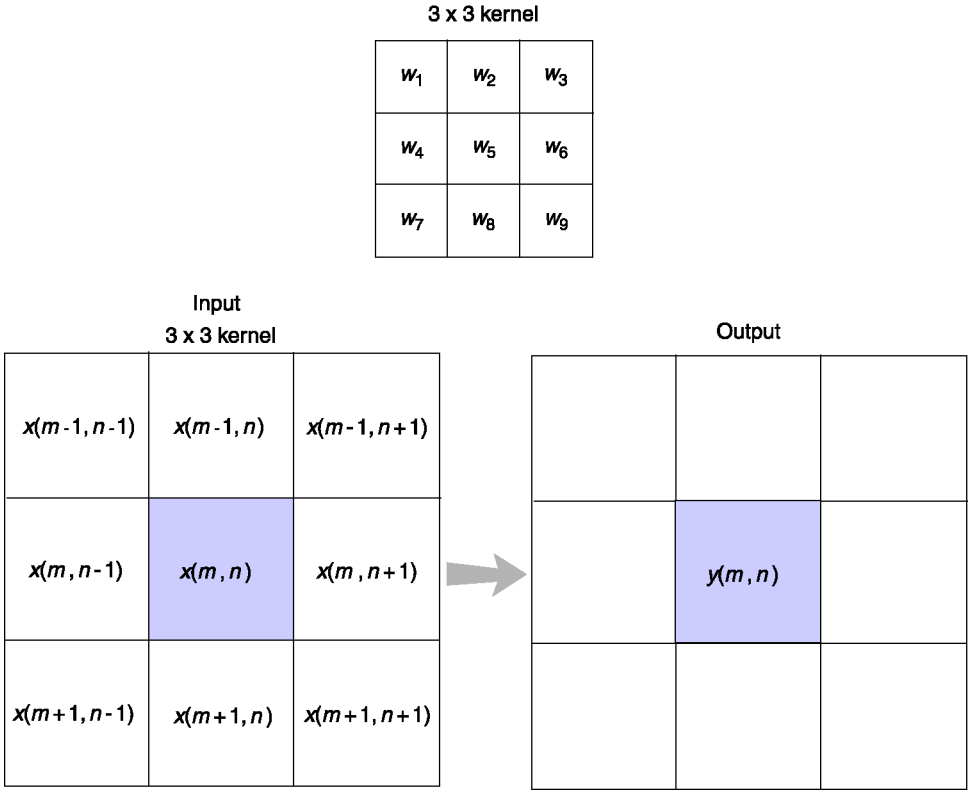
All of the operations above involve transformation of the input image. This can be presented as the convolution of the two-dimensional input image, $x(m,n)$ with the impulse response of the transform, $f(k,l)$, resulting in $y(m,n)$ which is the output image.

$$y(m, n) = f(k, l) \otimes x(m, n)$$

$$y(m, n) = \sum_{k=-N}^{N} \sum_{l=-N}^{N} f(k, l)x(m-k, n-l)$$

The $f(k,l)$ function refers to the matrix of filter coefficients. Because the matrix operation is analogous to a filter operation, the matrix itself is considered the impulse response of the filter. Depending on the type of operation, the choice of the convolutional kernel or mask, $f(k,l)$ is different. Figure 7–26 shows an example of convolving a 3×3 mask with a larger image.

Figure 7–26. Convolution Using a 3 × 3 Kernel



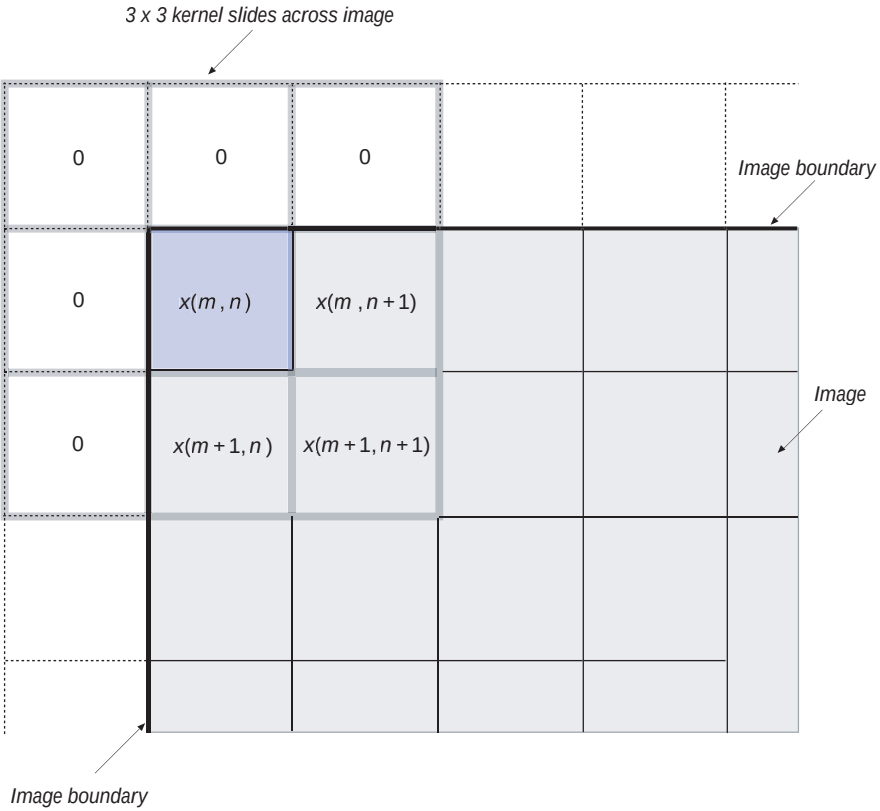
The output pixel value, $y(m,n)$ depends on the surrounding pixel values in the input image, as well as the filter weights:

$$
\begin{aligned}
y(m, n) = {} & w_1x(m-1, n-1) + w_2x(m-1, n) + w_3x(m-1, n+1) \\
& + w_4x(m, n-1) + w_5x(m, n) + w_6x(m, n+1) \\
& + w_7x(m+1, n-1) + w_8x(m+1, n) + w_9x(m+1, n+1)
\end{aligned}
$$

To complete the transformation, the kernel slides across the entire image. For pixels on the edge of the image, the convolution operation does not have a complete set of input data. To work around this problem, the pixels on the edge can be left unchanged. In some cases, it is acceptable to have an output image of reduced size. Alternatively, the matrix effect can be applied to edge pixels as if they are surrounded on the “empty” side by

black pixels, that is pixels with value zero. This is similar to padding the edges of the input image matrix with zeros and is referred to as the free boundary condition. This is shown in Figure 7–27.

Figure 7–27. Using Free Boundary Condition for Edge Pixels

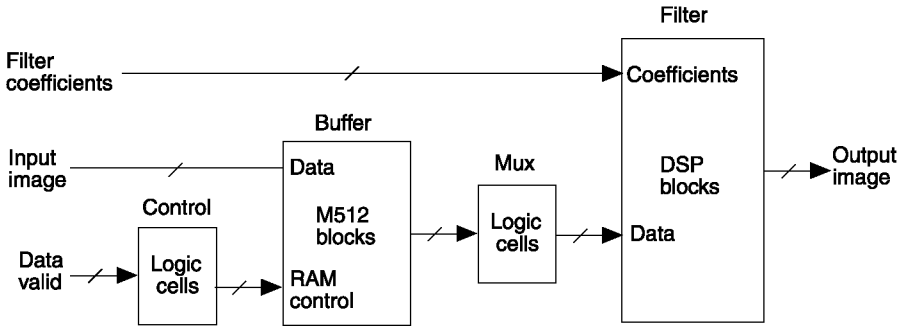


Convolution Implementation

This design example shows a 3 × 3 2-D FIR filter that takes in an 8 × 8 input image with gray pixel values ranging from 0-255 (8-bit). Data is fed in serially starting from the top left pixel, moving horizontally on a row-by-row basis. Next the data is stored in three separate RAM blocks in the buffering stage. Each M512 memory block represents a line of the image, and this is cycled through. For a 32 × 32 input image, the design needs M4K memory blocks. For larger images (640 × 480), this can be extended to M-RAM blocks or other buffering schemes. The control logic block provides the RAM control signals to interleave the data across all three

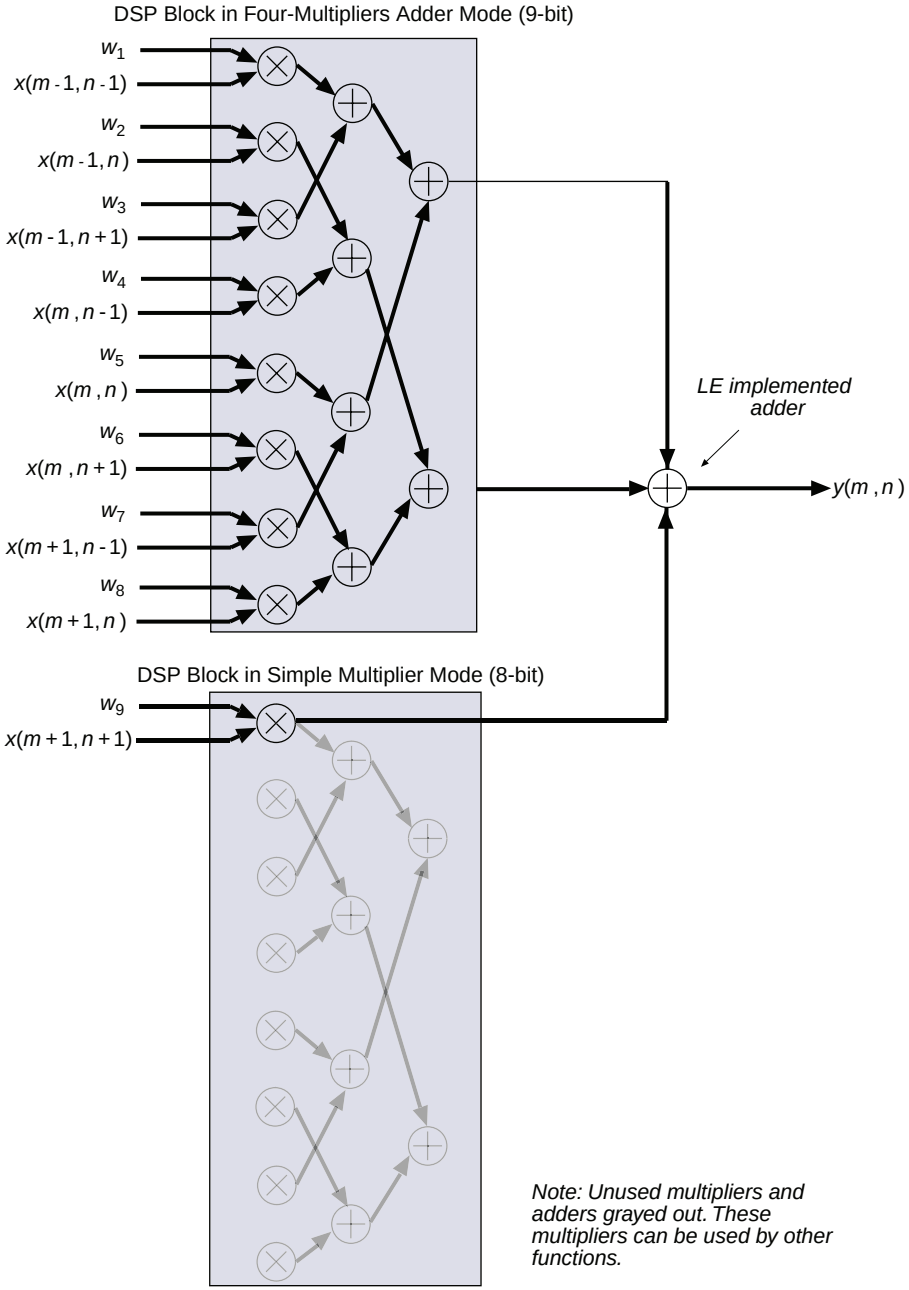
RAM blocks. The 9-bit signed filter coefficients feed directly into the filter block. As the data is shifted out from the RAM blocks, the multiplexer block checks for edge pixels and uses the free boundary condition. Figure 7–28 shows a top-level diagram of the design.

Figure 7–28. Block Diagram on Implementation of 3 × 3 Convolutional Filter for an 8 × 8 Pixel Input Image



The 3 × 3 filter block implements the nine multiply-add operations in parallel using two DSP blocks. One DSP block can implement eight of these multipliers. The second DSP block implements the ninth multiplier. The first DSP block is in the four-multipliers adder mode, and the second is in simple multiplier mode. In addition to the two DSP blocks, an external adder is required to sum the output of all nine multipliers. Figure 7–29 shows this implementation.

Figure 7–29. Implementation of 3 × 3 Convolutional Filter Block

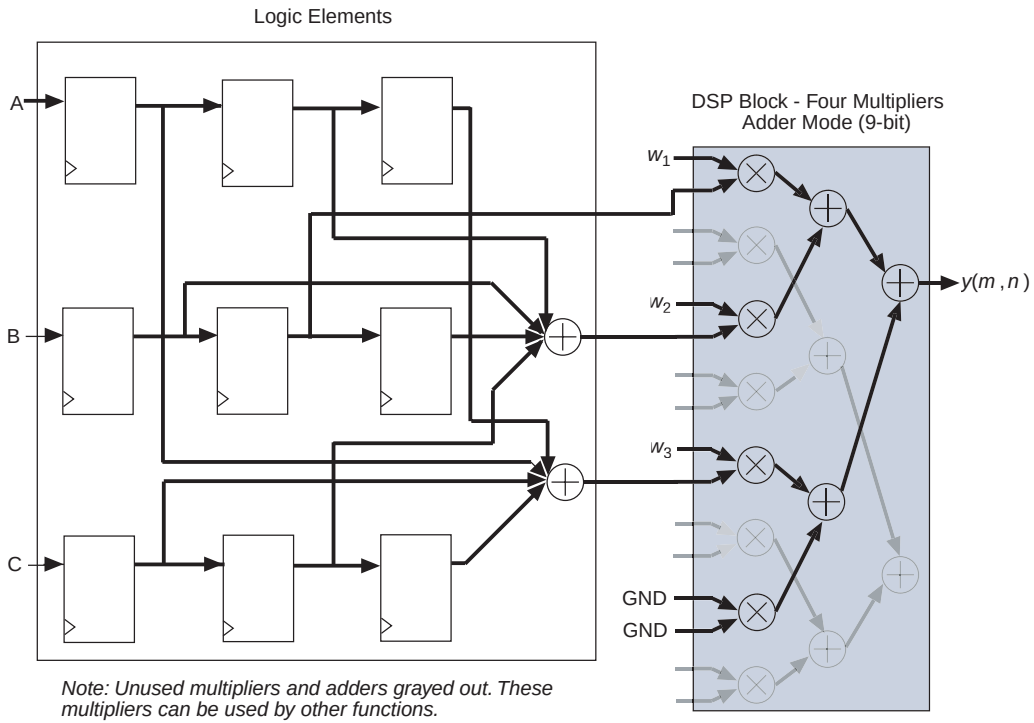


In cases where a symmetric 2-D filter is used, pixels sharing the same filter coefficients from three separate line-stores A, B, and C can be added together prior to the multiplication operation. This reduces the number of multipliers used. Referring to Figure 7–30, w_1, w_2, and w_3 are the filter coefficients. Figure 7–31 shows the implementation of this circular symmetric filter.

Figure 7–30. Symmetric 3 × 3 Kernel

w_3	w_2	w_3
w_2	w_1	w_2
w_3	w_2	w_3

Figure 7–31. Details on Implementation of Symmetric 3 × 3 Convolution Filter Block



Note: Unused multipliers and adders grayed out. These multipliers can be used by other functions.

Convolution Implementation Results

Table 7–17 shows the results of the 3 × 3 2-D FIR filter implementation in Figure 7–28.

Table 7–17. 3 × 3 2-D Convolution Filter Implementation Results

Part	EP1S10F780
Utilization	Lcell: 372/10570 (3%) DSP block 9-bit elements: 9/48 (18%) Memory bits: 768/920448 (<1%)
Performance	226 MHz
Latency	15 clock cycles

The design requires the input to be an 8 × 8 image, with 8-bit input data and 9-bit filter coefficient width. The output is an image of the same size.

Convolution Design Example

Download the 3 × 3 2-D Convolutional Filter (**two_d_fir.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Discrete Cosine Transform (DCT)

The discrete cosine transform (DCT) is widely used in video and audio compression, for example in JPEG, MPEG video, and MPEG audio. It is a form of transform coding, which is the preferred method for compression techniques. Images tend to compact their energy in the frequency domain making compression in the frequency domain much more effective. This is an important element in compressing data, where the goal is to have a high data compression rate without significant degradation in the image quality.

DCT Background

Similar to the discrete fourier transform (DFT), the DCT is a function that maps the input signal or image from the spatial to the frequency domain. It transforms the input into a linear combination of weighted basis functions. These basis functions are the frequency components of the input data.

For 1-D with input data $x(n)$ of size N, the DCT coefficients $Y(k)$ are:

$$Y(k) = \frac{\alpha(k)}{2} \sum_{n=0}^{N-1} x(n) \cos\left(\frac{(2n+1)\pi k}{2N}\right) \quad \text{for } 0 \leq k \leq N-1$$

where:

$$\alpha(k) = \sqrt{\frac{1}{N}} \quad \text{for } k = 0$$

$$\alpha(k) = \sqrt{\frac{2}{N}} \quad \text{for } 1 \leq k \leq N-1$$

For 2-D with input data $x(m,n)$ of size $N \times N$, the DCT coefficients for the output image, $Y(p,q)$ are:

$$Y(p, q) = \frac{\alpha(p)\alpha(q)}{2} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m, n) \cos\left(\frac{(2m+1)\pi p}{2N}\right) \cos\left(\frac{(2n+1)\pi q}{2N}\right)$$

where:

$$\alpha(p) = \sqrt{\frac{1}{N}} \quad \text{for } p = 0$$

$$\alpha(q) = \sqrt{\frac{1}{N}} \quad \text{for } q = 0$$

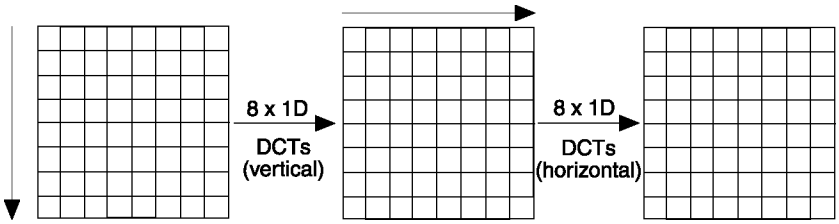
$$\alpha(p) = \sqrt{\frac{2}{N}} \quad \text{for } 1 \leq p \leq N-1$$

$$\alpha(q) = \sqrt{\frac{2}{N}} \quad \text{for } 1 \leq q \leq N-1$$

2-D DCT Algorithm

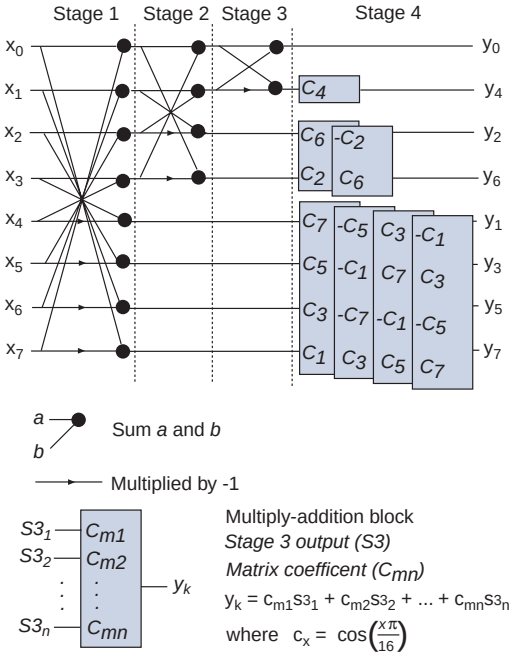
The 2-D DCT can be thought of as an extended 1-D DCT applied twice; once in the *x* direction and again in the *y* direction. Because the 2-D DCT is a separable transform, it is possible to calculate it using efficient 1-D algorithms. Figure 7–32 illustrates the concept of a separable transform.

Figure 7–32. A 2-D DCT is a Separable Transform



This section uses a standard algorithm proposed in [1]. Figure 7–33 shows the flow graph for the algorithm. This is similar to the butterfly computation of the fast fourier transform (FFT). Similar to the FFT algorithms, the DCT algorithm reduces the complexity of the calculation by decomposing the computation into successively smaller DCT components. The even coefficients (y_0, y_2, y_4, y_6) are calculated in the upper half and the odd coefficients (y_1, y_3, y_5, y_7) in the lower half. As a result of the decomposition, the output is reordered as well.

Figure 7–33. Implementing an N=8 Fast DCT



The following defines in matrix format, the 8-point 1-D DCT of Figure 7–33:

$$\left[Y_{1D}\right] = \left[x\right] \times \left[Add_1\right] \times \left[Add_2\right] \times \left[Add_3\right] \times \left[C\right]$$

where:

$[x]$ is the 1×8 input matrix

$$\left[Add_1\right] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

$$\left[Add_2\right] = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\left[Add_3\right] = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$[C] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & C_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_6 & -C_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & C_2 & C_6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & C_7 & -C_5 & C_3 & -C_1 \\ 0 & 0 & 0 & 0 & C_5 & -C_1 & C_7 & C_3 \\ 0 & 0 & 0 & 0 & C_3 & -C_7 & -C_1 & -C_5 \\ 0 & 0 & 0 & 0 & C_1 & C_3 & C_5 & C_7 \end{bmatrix}$$

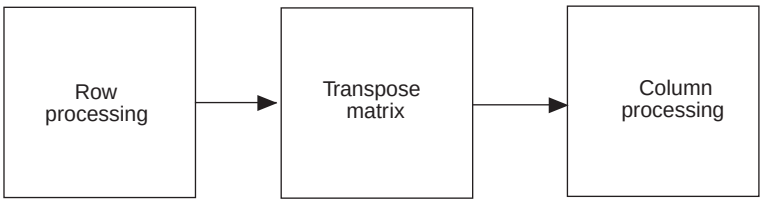
$$C_x = \cos\frac{\pi x}{16}$$

All of the additions in stages 1, 2 and 3 of Figure 7–32 appear in symmetric add and subtract pairs. The entire first stage is simply four such pairs in a very typical cross-over pattern. This pattern is repeated in stages 2 and 3. Multiplication operations are confined to stage 4 in the algorithm. This implementation is shown in more detail in the next section.

DCT Implementation

In taking advantage of the separable transform property of the DCT, the implementation can be divided into separate stages; row processing and column processing. However, some data restructuring is necessary before applying the column processing stage to the results from the row processing stage. The data buffering stage must transpose the data first. Figure 7–34 shows the different stages.

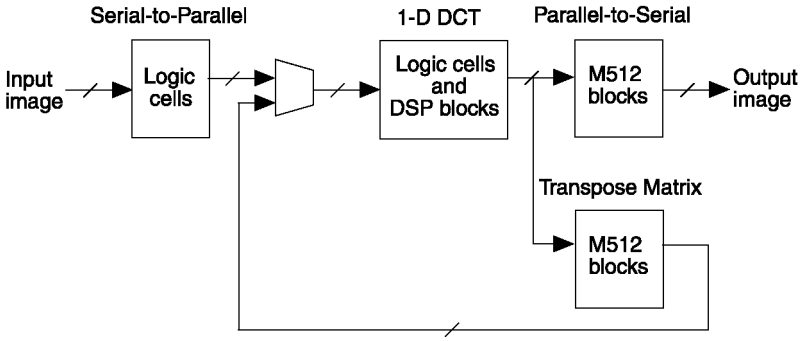
Figure 7–34. Three Separate Stages in Implementing the 2-D DCT



Because the row processing and column processing blocks share the same 1-D 8-point DCT algorithm, the hardware implementation shows this block as being shared. The DCT algorithm requires a serial-to-parallel conversion block at the input because it works on blocks of eight data

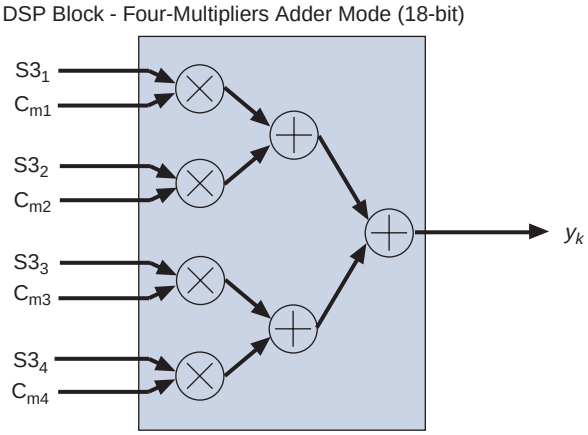
points in parallel. There is also a parallel-to-serial conversion block at the output because the column processing stage generates the output image column-by-column. In order to have the output in the same order as the input (i.e., row-by-row), this conversion is necessary. Appropriate scaling needs to be applied to the completed transform but this can be combined with the quantization stage which often follows a DCT [1]. Figure 7–35 shows a top-level block diagram of this design.

Figure 7–35. Block Diagram on Serial Implementation of 2-D DCT



The implementation of the 1-D DCT block is based on the algorithm shown in Figure 7–33. The simple addition and subtraction operations in stages 1, 2 and 3 are implemented using logic cells. The multiply and multiply-addition operations in stage 4 are implemented using DSP blocks in the Stratix device in the simple multiplier mode, two-multiplier adder mode, and the four-multiplier adder mode. An example of the multiply-addition block is shown in Figure 7–36.

Figure 7–36. Details on the Implementation of the Multiply-Addition Operation in Stage 4 of the 1-D DCT Algorithm



Note to Figure 7–36:

(1) Referring to Figure 7–33. $S3_n$ is an output from stage 3 of the DCT and C_{mn} is a matrix coefficient. $C_x = \cos(x\pi/16)$.

DCT Implementation Results

Table 7–18 shows the results of implementing a 2-D DCT with 18-bit precision, as shown in Figure 7–35.

Table 7–18. 2-D DCT Implementation Results

Part	EP1S20F780
Utilization	Lcell: 1717/18460 (9%) DSP Block 9-bit element: 18/80 (22%) Memory bits: 2816/1669248 (<1%)
Performance	165 MHz
Latency	80 clock cycles

DCT Design Example

Download the 2-D convolutional filter (**d_dct.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Arithmetic Functions

Arithmetic functions, such as trigonometric functions, including sine, cosine, magnitude and phase calculation, are important DSP elements. This section discusses the implementation of a simple vector magnitude function in a Stratix device.

Background

Complex numbers can be expressed in two parts: real and imaginary.

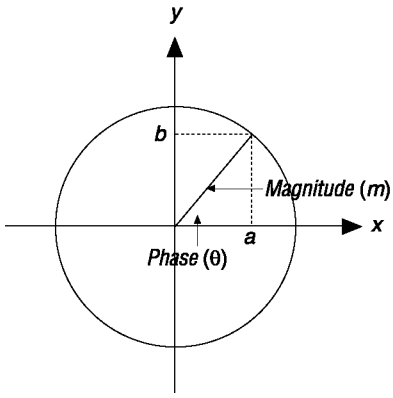
$$z = a + jb$$

where:

a is the real part
b is the imaginary part
$j^2 = -1$

In a two-dimensional plane, a vector (a,b) with reference to the origin (0,0) can also be represented as a complex number. In essence, the x-axis represents the real part, and the y-axis represents the imaginary part (see Figure 7–37).

Figure 7–37. Magnitude of Vector (a,b)



Complex numbers can be converted to phase and amplitude or magnitude representation, using a Cartesian-to-polar coordinate conversion. For a vector (a,b), the phase and magnitude representation is the following:

$$\text{Magnitude } m = \sqrt{a^2 + b^2}$$

$$\text{Phase angle } \theta = \tan^{-1}(b/a)$$

This conversion is useful in different applications, such as position control and position monitoring in robotics. It is also important to have these transformations at very high speeds to accommodate real-time processing.

Arithmetic Function Implementation

A common approach to implementing these arithmetic functions is using the coordinate rotation digital computer (CORDIC) algorithm. The CORDIC algorithm calculates the trigonometric functions of sine, cosine, magnitude, and phase using an iterative process. It is made up of a series of micro-rotations of the vector by a set of predetermined constants, which are powers of 2.

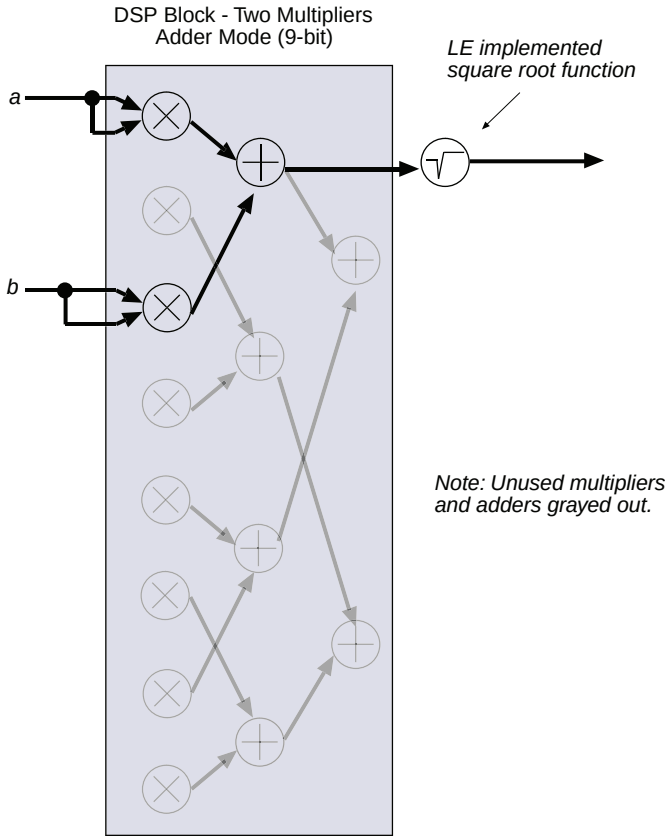
Using binary arithmetic, this algorithm essentially replaces multipliers with shift and add operations. In Stratix devices, it is possible to calculate some of these arithmetic functions directly, without having to implement the CORDIC algorithm.

This section describes a design example that calculates the magnitude of a 9-bit signed vector (a,b) using a pipelined version of the square root function available at the Altera IP Megastore. To calculate the sum of the squares of the input ($a^2 + b^2$), configure the DSP block in the two-multipliers adder mode. The square root function is implemented using an iterative algorithm similar to the long division operation. The binary numbers are paired off, and subtracted by a trial number. Depending on if the remainder is positive or negative, each bit of the square root is determined and the process is repeated. This square root function does not require memory and is implemented in logic cells only.

In this example, the input bit precision (`IN_PREC`) feeding into the square root macro is set to twenty, and the output precision (`OUT_PREC`) is set to ten. The number of precision bits is parameterizable. Also, there is a third parameter, `PIPELINE`, which controls the architecture of the square root macro. If this parameter is set to `YES`, it includes pipeline stages in the square root macro. If set to `NO`, the square root macro becomes a single-cycled combinatorial function.

Figure 7–38 shows the implementation the magnitude design.

Figure 7–38. Implementing the Vector Magnitude Function



Arithmetic Function Implementation Results

Table 7–19 shows the results of the implementation shown in Figure 7–38 with the PIPELINE parameter set to YES. Table 7–20 shows the results of the implementation shown in Figure 7–38 with the PIPELINE parameter set to NO.

Table 7–19. Vector Magnitude Function Implementation Results (PIPELINE=YES)

Part	EP1S10F780
Utilization	Lcell: 497/10570 (4%) DSP block 9-bit elements: 2/48 (4%) Memory bits: 0/920448 (0%)
Performance	194 MHz
Latency	15 clock cycles

Table 7–20. Vector Magnitude Function Implementation Results (PIPELINE=NO)

Part	EP1S10F780
Utilization	Lcell: 244/10570 (2%) DSP block 9-bit elements: 2/48 (4%) Memory bits: 0/920448 (0%)
Performance	30 MHz
Latency	3 clock cycles

Arithmetic Function Design Example

Download the Vector Magnitude Function (**magnitude.zip**) design example from the Design Examples section of the Altera web site at **www.altera.com.**

Conclusion

The DSP blocks in Stratix and Stratix GX devices are optimized to support DSP functions requiring high data throughput, such as FIR filters, IIR filters and the DCT. The DSP blocks are flexible and configurable in different operation modes based on the application's needs. The TriMatrix memory provides the data storage capability often needed in DSP applications.

The DSP blocks and TriMatrix memory in Stratix and Stratix GX devices offer performance and flexibility that translates to higher performance DSP functions.

References

See the following for more information:

- *Optimal DCT for Hardware Implementation*
  M. Langhammer. Proceedings of International Conference on Signal Processing Applications & Technology (ICSPAT) '95, October 1995

- *Digital Signal Processing: Principles, Algorithms, and Applications*
  John G. Proakis, Dimitris G. Manolakis. Prentice Hall

- *Hardware Implementation of Multirate Digital Filters*
  Tony San. Communication Systems Design, April 2000

- *AN 73: Implementing FIR Filters in FLEX Devices*

- *Efficient Logic Synthesis Techniques for IIR Filters*
  M.Langhammer. Proceedings of International Conference on Signal Processing Applications & Technology (ICSPAT) '95, October 1995

Section V. IP & Design Considerations

This section provides documentation on some of the IP functions offered by Altera® for Stratix® devices. (Also see the Intellectual Property section of the Altera web site for a complete offering of IP cores for Stratix devices.) The last chapter details design considerations for migrating from the APEX™ architecture.

This section contains the following chapters:

- Chapter 8, Implementing 10-Gigabit Ethernet Using Stratix & Stratix GX Devices
- Chapter 9, Implementing SFI-4 in Stratix & Stratix GX Devices
- Chapter 10, Transitioning APEX Designs to Stratix & Stratix GX Devices

Revision History

The table below shows the revision history for Chapters 8 through 10.

Chapter	Date/Version	Changes Made
8	July 2005, v2.0	Updated Stratix GX device information.
	September 2004, v1.2	• Table 8–2 on page 8–10: updated table, deleted Note 1, and updated Note 2. • Updated Table 8–4 on page 8–12.
	November 2003, v1.1	• Removed support for series and parallel on-chip termination.
	April 2003, v1.0	• No new changes in *Stratix Device Handbook* v2.0.
9	July 2005, v2.0	• Updated Stratix GX device information.
	September 2004, v1.1	• Table 9–2 on page 9–9: updated table, deleted Note 1, and updated Note 2. • Updated Table 9–4 on page 9–10.
	April 2003, v1.0	• No new changes in *Stratix Device Handbook* v2.0.

Chapter	Date/Version	Changes Made
10	July 2005, v3.0	• Updated Stratix GX device information.
	September 2004, v2.1	• Updated Table 10–9 on page 10–26.
	April 2004, v2.0	• Synchronous occurrences renamed pipelined. • Asynchronous occurrences renamed flow-through.
	November 2003, v1.2	• Removed support for series and parallel on-chip termination.
	October 2003, v1.1	• Updated Table 10–6.
	April 2003, v1.0	• No new changes in *Stratix Device Handbook* v2.0.

8. Implementing 10-Gigabit Ethernet Using Stratix & Stratix GX Devices

S52010-2.0

Introduction

Ethernet has evolved to meet ever-increasing bandwidth demands and is the most prevalent local-area network (LAN) communications protocol. 10-Gigabit Ethernet extends that protocol to higher bandwidth for future high-speed applications. The accelerated growth of network traffic and the resulting increase in bandwidth requirements is driving service providers and enterprise network architects towards high-speed network solutions. Potential applications for 10-Gigabit Ethernet include private campus or LAN backbones, high-speed access links between service providers and enterprises, and aggregation and transport in metropolitan area networks (MANs).

The I/O features of Stratix® and Stratix GX devices enable support for 10-Gigabit Ethernet, supporting 10-Gigabit 16-bit interface (XSBI) and 10-Gigabit medium independent interface (XGMII). Stratix GX devices can additionally support the 10-gigabit attachment unit interface (XAUI) using the embedded 3.125-Gbps transceivers. You can find more information on XAUI support in Section II, *Stratix GX Transceiver User Guide*, of the *Stratix GX Device Handbook, Volume 1*.

This chapter discusses the following topics:

- Fundamentals of 10-Gigabit Ethernet
- Description and implementation of XSBI
- Description and implementation of XGMII
- Description of XAUI
- I/O characteristics of XSBI, XGMII, and XAUI

Related Links

- 10-Gigabit Ethernet Alliance at **www.10gea.org**
- The *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* and the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*
- The *High-Speed Differential I/O Interfaces in Stratix Devices* chapter

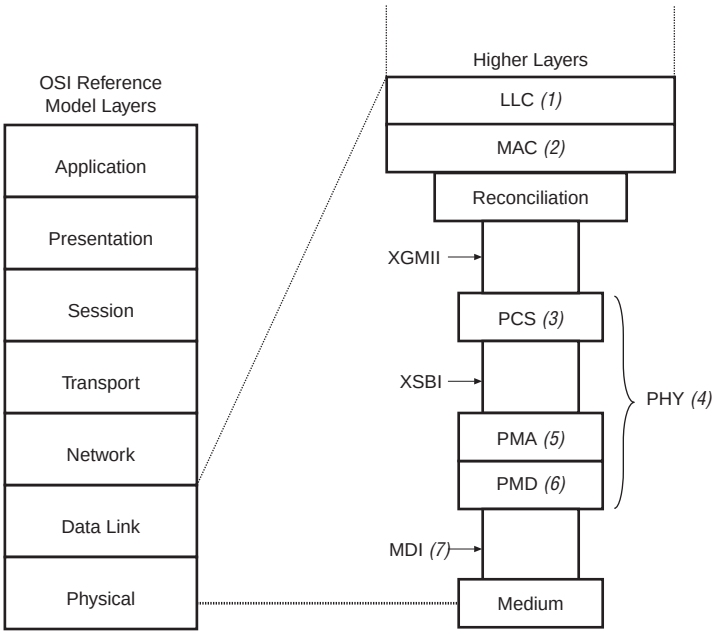
10-Gigabit Ethernet

Ethernet speed has increased to keep pace with demand, initially to 10 megabits per second (Mbps), later to 100 Mbps, and recently to 1 gigabit per second (Gbps). Ethernet is the dominant network technology in LANs, and with the advent of 10-Gigabit Ethernet, it is entering the MAN and wide area network (WAN) markets.

The purpose of the 10-Gigabit Ethernet proposed standard is to extend the operating speed to 10 Gbps defined by protocol IEEE 802.3 and include WAN applications. These additions provide a significant increase in bandwidth while maintaining maximum compatibility with current IEEE 802.3 interfaces.

Since its inception in March 1999, the 10-Gigabit Ethernet Task Force has been working on the IEEE 802.3ae Standard. Some of the information in the following sections is derived from Clauses 46, 47, 49, and 51 of the IEEE Draft P802.3ae / D3.1 document. A fully ratified standard is expected in the first half of 2002. Figure 8–1 shows the relationship of 10-Gigabit Ethernet to the Open Systems Interconnection (OSI) protocol stack.

Figure 8–1. 10-Gigabit Ethernet Protocol in Relation to OSI Protocol Stack



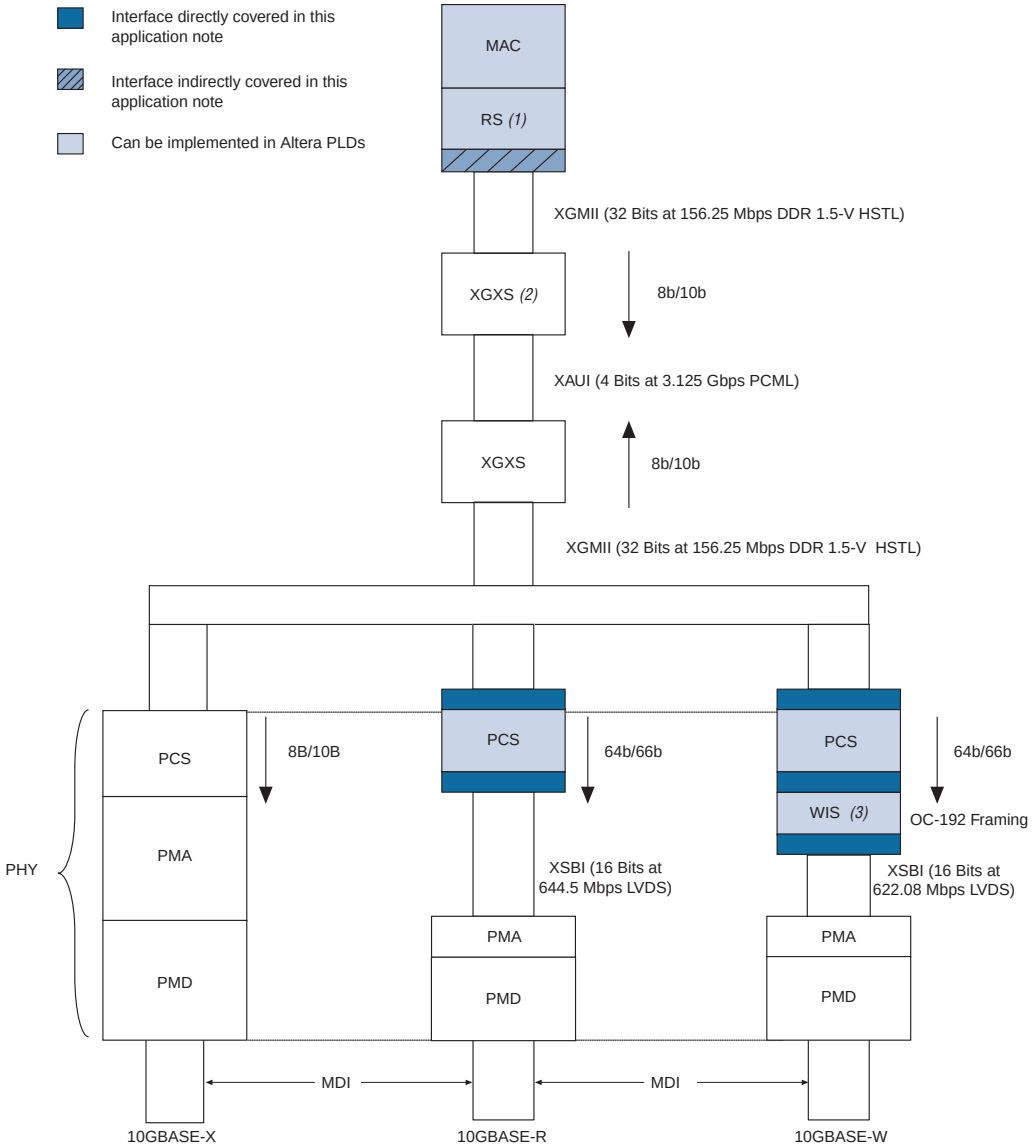
Notes to Figure 8–1:

(1) LLC: logical link controller
(2) MAC: media access controller
(3) PCS: physical coding sublayer
(4) PHY: physical layer
(5) PMA: physical medium attachment
(6) PMD: physical medium dependent
(7) MDI: medium dependent interface

The Ethernet PHY (layer 1 of the OSI model) connects the media (optical or copper) to the MAC (layer 2). The Ethernet architecture further divides the PHY (layer 1) into a PMD sublayer, a PMA sublayer, and a PCS. For example, optical transceivers are PMD sublayers. The PMA converts the data between the PMD sublayer and the PCS sublayer. The PCS is made up of coding (e.g., 8b / 10b, 64b / 66b) and serializer or multiplexing functions. Figure 8–2 shows the components of 10-Gigabit Ethernet and how Altera implements certain blocks and interfaces.

10-Gigabit Ethernet has three different implementations for the PHY: 10GBASE-X, 10GBASE-R, and 10GBASE-W. The 10GBASE-X implementation is a PHY that supports the XAUI interface. The XAUI interface used in conjunction with the XGMII extender sublayer (XGXS) allows more separation in distance between the MAC and PHY. 10GBASE-X PCS uses four lanes of 8b / 10b coded data at a rate of 3.125 Gbps. 10GBASE-X is a wide wave division multiplexing (WWDM) LAN PHY. 10GBASE-R and 10GBASE-W are serial LAN PHYs and serial WAN PHYs, respectively. Unlike 10GBASE-X, 10GBASE-R and 10GBASE-W implementations have a XSBI interface and are described in more detail in the following section.

Figure 8–2. 10-Gigabit Ethernet Block Diagram



Notes to Figure 8–2:

(1) The reconciliation sublayer (RS) interfaces the serial MAC data stream and the parallel data of XGMII.

(2) The XGMII extender sublayer (XGXS) extends the distance of XGMII when used with XUAI and provides the data conversion between XGMII and XAUI.

(3) The WAN interface sublayer (WIS) implements the OC-192 framing and scrambling functions.

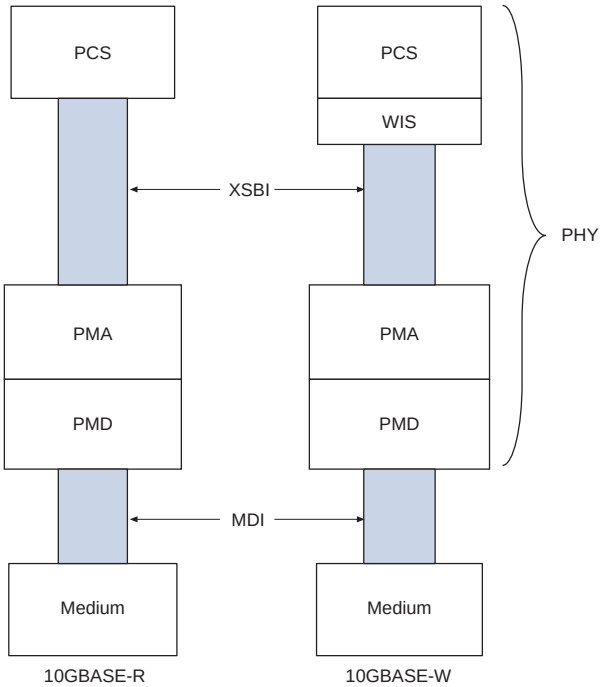
Interfaces

The following sections discuss XSBI, PCS, XGMII, and XAUI.

XSBI

One of the blocks of 10-Gigabit Ethernet is the XSBI interface. XSBI is the interface between the PCS and the PMA sublayers of the PHY layer of the OSI model. XSBI supports two types of PHY layers, LAN PHY and WAN PHY. The LAN PHY is part of 10GBASE-R, and supports existing Gigabit Ethernet applications at ten times the bandwidth. The WAN PHY is part of 10GBASE-W, and supports connections to existing and future installations of SONET/SDH circuit-switched access equipment. 10GBASE-R is a physical layer implementation that is comprised of the PCS sublayer, the PMA, and the PMD. 10GBASE-R is based upon 64b/66b data coding. 10GBASE-W is a PHY layer implementation that is comprised of the PCS sublayer, the WAN interface sublayer (WIS), the PMA, and the PMD. 10GBASE-W is based on STS-192c/SDH VC-4-64c encapsulation of 64b/66b encoded data. Figure 8–3 shows the construction of these two PHY layers.

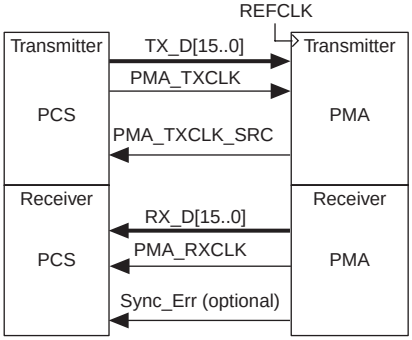
Figure 8–3. XSBI Interface for the Two PHY Layers



Functional Description

XSBI uses 16-bit LVDS data to interface between the PCS and the PMA sublayer. Figure 8–4 shows XSBI between these two sublayers.

Figure 8–4. XSBI Functional Block Diagram



On the transmitter side, the transmit data (`TX_D[15..0]`) is output by the PCS and input at the PMA using the transmitter clock (`PMA_TXCLK`), which is derived from the PMA source clock (`PMA_TXCLK_SRC`). The PMA source clock is generated from the PMA with its reference clock (`REFCLK`). On the receiver side, the receiver data (`RX_D[15..0]`) is output by the PMA and input at the PCS using the PMA-generated receiver clock (`PMA_RXCLK`). The `SYNC_ERR` optional signal is sent to the PCS by the PMA if the PMA fails to recover the clock from the serial data stream.

The ratios for these two clocks and data are dependent on the type of PHY used. Table 8–1 shows the rates for both PHY types.

Table 8–1. XSBI Clock & Data Rates for WAN & LAN PHY

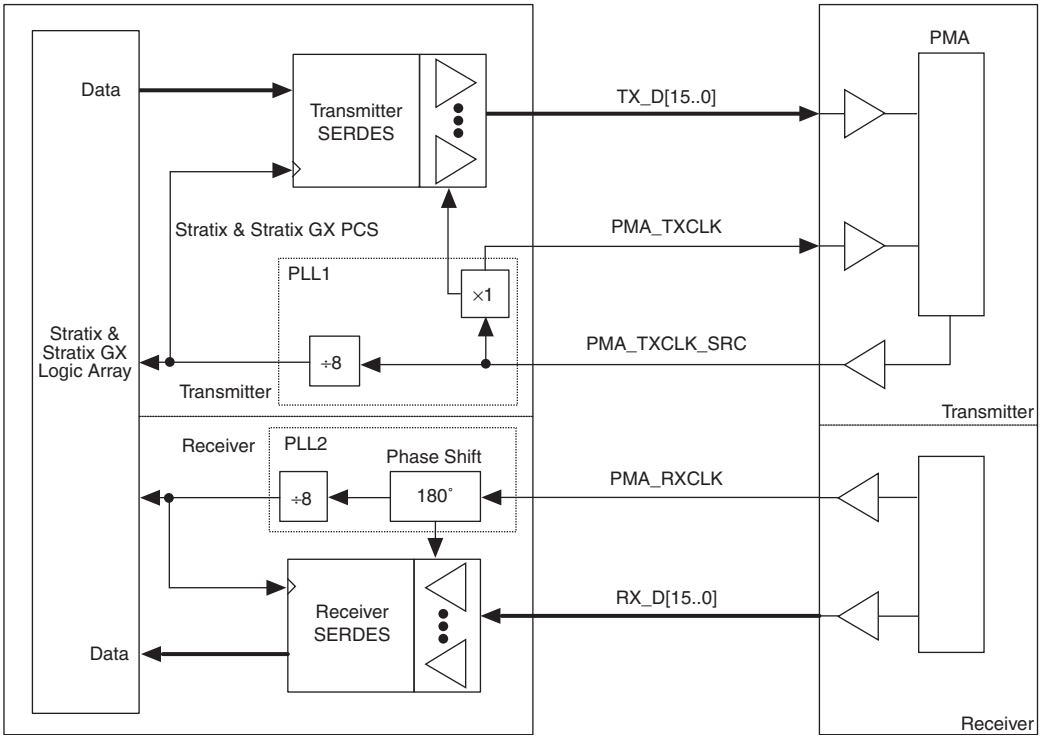
Parameter	WAN PHY	LAN PHY	Unit
`TX_D[15..0]`	622.08	644.53125	Mbps
`PMA_TXCLK`	622.08	644.53125	MHz
`PMA_TXCLK_SRC`	622.08	644.53125	MHz
`RX_D[15..0]`	622.08	644.53125	Mbps
`PMA_RXCLK`	622.08	644.53125	MHz

Implementation

The 16-bit full duplex LVDS implementation of XSBI in Stratix devices is shown in Figure 8–5.

The source-synchronous I/O implemented in Stratix GX devices optionally includes dynamic phase alignment (DPA). DPA automatically and continuously tracks fluctuations caused by system variations and self-adjusts to eliminate the phase skew between the multiplied clock and the serial data, allowing for data rates of 1 Gbps. In non DPA mode the I/O behaves similarly to that of the Stratix I/O. This document assumes that DPA is disabled. However, it is simple to implement the same system with DPA enabled to take advantage of its features. For more information on DPA, see the *Stratix GX Transceivers* chapter in the *Stratix GX Device Handbook, Volume 1*.

Figure 8–5. Stratix & Stratix GX Device XSBI Implementation



The transmit serializer / deserializer (SERDES) clock comes from the transmitter clock source (PMA_TXCLK_SRC). The receiver SERDES clock comes from the PMA receiver recovered clock (PMA_RXCLK).

Figure 8–6 shows the transmitter output of the XSBI core. Data transmitted from the PCS to the PMA starts at the core of the Stratix or Stratix GX device and travels to the Stratix or Stratix GX transmitter SERDES block. The transmitter SERDES block converts the parallel data to serial data for 16 individual channels (TX_D[15..0]). The PMA source clock (PMA_TXCLK_SRC) is used to clock out the signal data. PMA_TXCLK is generated from the same phase-locked loop (PLL) as the data, and it travels to the PMA at the same rate as the data. By using one of the data channels in the middle of the bus as the clock (in this case, the eighth channel CH8), the clock-to-data skew improves.

Figure 8–6. Stratix & Stratix GX Device XSBI Transmitter Implementation

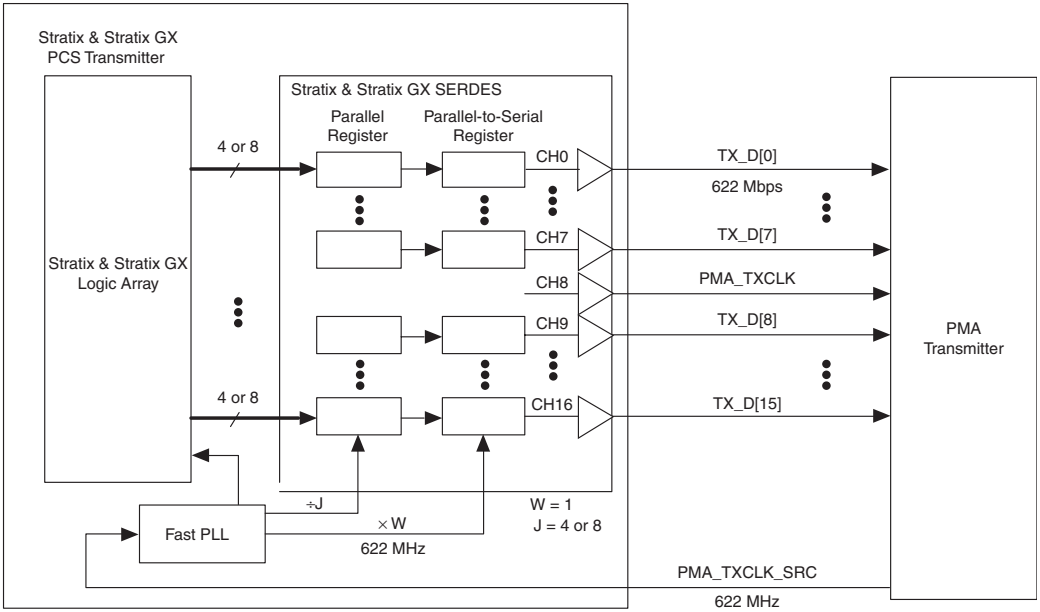
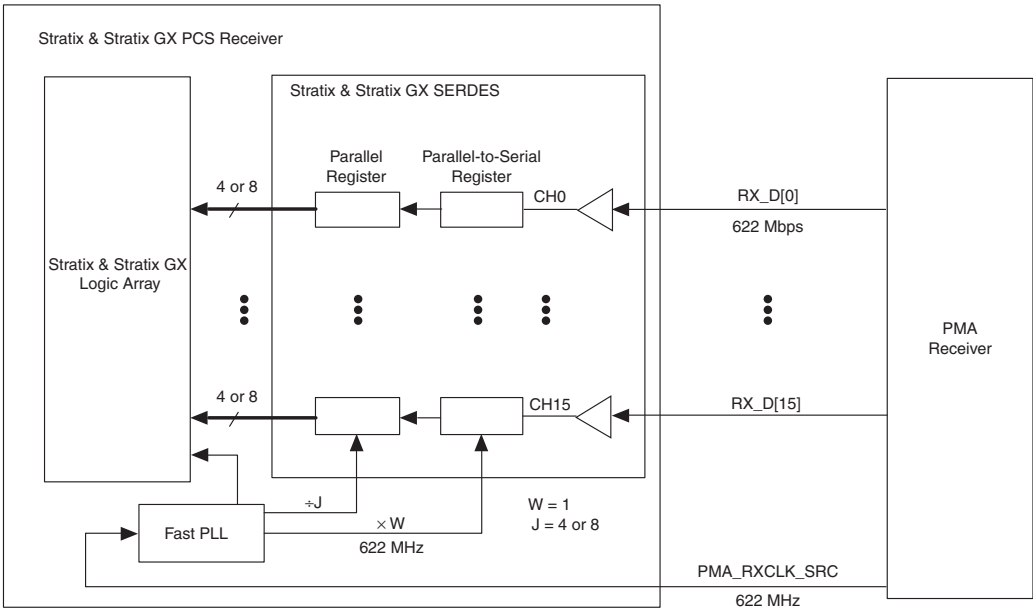


Figure 8–7 shows the receiver input of the XSBI core. From the receiver side, data (RX_D[15..0]) comes from the PMA to the Stratix or Stratix GX receiver SERDES block along with the PMA receiver clock (PMA_RXCLK). The PMA receiver clock is used to convert the serial data to parallel data. The phase shift or inversion on the PMA receiver clock is needed to capture the receiver data.

Stratix and Stratix GX devices contain up to eight fast PLLs. These PLLs provide high-speed outputs for high-speed differential I/O support as well as general- purpose clocking with multiplication and phase shifting. The fast PLL incorporates this 180° phase shift. The Stratix and Stratix GX device's data realignment feature enables you to save more logic elements (LEs). This feature provides a byte-alignment capability, which is embedded inside the SERDES. The data realignment circuitry can correct for bit misalignments by slipping data bits.

f For more information about fast PLLs, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

Figure 8–7. Stratix & Stratix GX Device XSBI Receiver Implementation



With this XSBI transmitter and receiver block implementation, each XSBI core requires two fast PLLs. The potential number of XSBI cores per device corresponds to the number of fast PLLs each Stratix or Stratix GX device contains. Tables 8–2 and 8–3 show the number of LVDS channels, the number of fast PLLs, and the number of XSBI cores that can be supported for each Stratix or Stratix GX device.

Table 8–2. Stratix Device XSBI Core Support

Stratix Device	Number of LVDS Channels (Receive/Transmit) *(1)*	Number of Fast PLLs	Number of XSBI Interfaces (Maximum)
EP1S10	44/44	4	2
EP1S20	66/66	4	2
EP1S25	78/78	4	2
EP1S30	82/82	8	4
EP1S40	90/90	8	4
EP1S60	116/116	8	4
EP1S80	152/156	8	4

Note to Table 8–2:

(1) The LVDS channels can go up to 840 Mbps for flip-chip packages and up to 624 Mbps for wire-bond packages. This number includes both high speed and low speed channels. The high speed LVDS channels can go up to 840 Mbps. The low speed LVDS channels can go up to 462 Mbps. The *High-Speed Differential I/O Support chapter* in the *Stratix Device Handbook, Volume 1,* and the device pin-outs on the web (**www.altera.com**) specify which channels are high and low speed.

Table 8–3. Stratix GX Device XSBI Core Support

Stratix GX Device	Number of LVDS Channels (Receive/Transmit) (1)	Number of Fast PLLs	Number of XSBI Interfaces (Maximum)
EP1SGX10	22/22	2	1
EP1SGX25	39/39	2	2
EP1SGX40	45/45	4	2

Note to Table 8–3:

(1) The LVDS channels can go up to 840 Mbps for flip-chip packages and up to 624 Mbps for wire-bond packages. This number includes both high speed and low speed channels. The high speed LVDS channels can go up to 840 Mbps. The low speed LVDS channels can go up to 462 Mbps. The *High-Speed Differential I/O Support chapter* in the *Stratix Device Handbook, Volume 1*, and the device pin-outs on the web (**www.altera.com**) specify which channels are high and low speed.

AC Timing Specifications

Stratix and Stratix GX devices support a PCS interface. Figures 8–8 and 8–9 and Tables 8–4 and 8–5 illustrate timing characteristics of the PCS transmitter and receiver interfaces.

Figure 8–8 shows the AC timing diagram for the Stratix and Stratix GX PCS transmitter. You can determine PCS channel-to-channel skew by adding the data invalid window before the rising edge (T_{cq_pre}) to the data invalid window after the rising edge (T_{cq_post}).

Figure 8–8. PCS Transmitter Timing Diagram

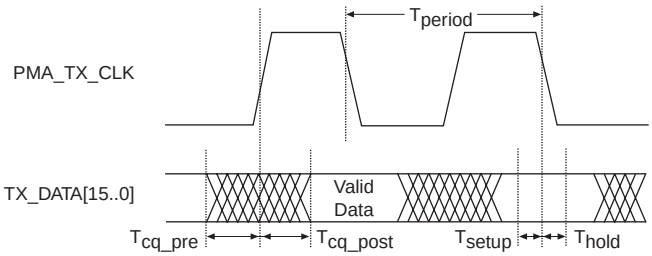


Table 8–4 lists the AC timing specifications for the PCS transmitter.

Table 8–4. PCS Transmitter Timing Specifications

Parameter	Value			Unit
	Min	Typ	Max	
PMA_TX_CLK T_{period} (WAN)		1,608		ps
PMA_TX_CLK T_{period} (LAN)		1,552		ps
Data invalid window before the rising edge (T_{cq_pre})			200	ps
Data invalid window after the rising edge (T_{cq_post})			200	ps
PMA_TX_CLK duty cycle	40		60	%
PCS transmitter channel-to-channel skew			200	ps

Figure 8–9 shows the AC timing diagram for the Stratix and Stratix GX PCS receiver interface. You can determine the PCS sampling window by adding T_{setup} to T_{hold}. Receiver skew margin (RSKM) refers to the amount of skew tolerated on the printed circuit board (PCB).

Figure 8–9. PCS Receiver Timing Diagram

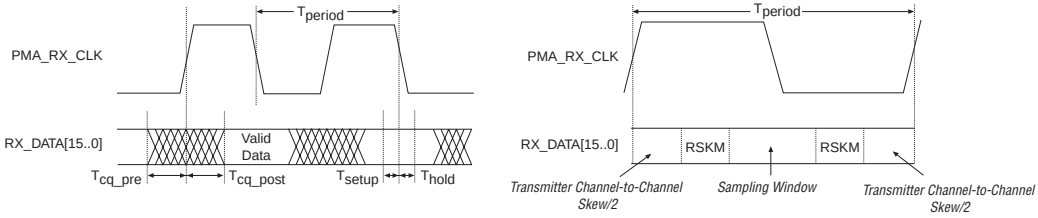


Table 8–5 lists the AC timing specifications for the PCS receiver interface.

Table 8–5. PCS Receiver Timing Specifications (Part 1 of 2)

Parameter	Value			Unit
	Min	Typ	Max	
PMA_RX_CLK T_{period} (WAN)		1,608		ps
PMA_RX_CLK T_{period} (LAN)		1,552		ps
Data invalid window before the rising edge (T_{cq_pre})			200	ps
Data invalid window after the rising edge (T_{cq_post})			200	ps

Table 8–5. PCS Receiver Timing Specifications (Part 2 of 2)

Parameter	Value			Unit
	Min	Typ	Max	
PMA_RX_CLK duty cycle	45		55	%
Data set-up time (T_{setup})	300			ps
Data hold time (T_{hold})	300			ps
PCS sampling window	600			ps
RSKM (WAN)			304	ps
RSKM (LAN)			276	ps

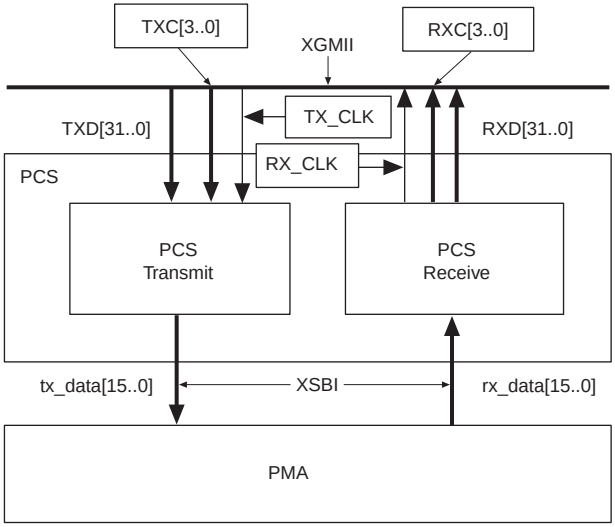
XGMII

The purpose of XGMII is to provide a simple, inexpensive, and easy to implement interconnection between the MAC sublayer and the PHY. Though XGMII is an optional interface, it is used extensively in the 10-Gigabit Ethernet standard as the basis for the specification. The conversion between the parallel data paths of XGMII and the serial MAC data stream is carried out by the reconciliation sublayer. The reconciliation sublayer maps the signal set provided at the XGMII to the physical layer signaling (PLS) service primitives provided at the MAC. XGMII supports a 10-Gbps MAC data rate.

Functional Description

The XGMII is composed of independent transmit and receive paths. Each direction uses 32 data signals, `TXD[31..0]` and `RXD[31..0]`, 4 control signals, `TXC[3..0]` and `RXC[3..0]`, and a clock `TX_CLK` and `RX_CLK`. Figure 8–10 shows the XGMII functional block diagram.

Figure 8–10. XGMII Functional Block Diagram



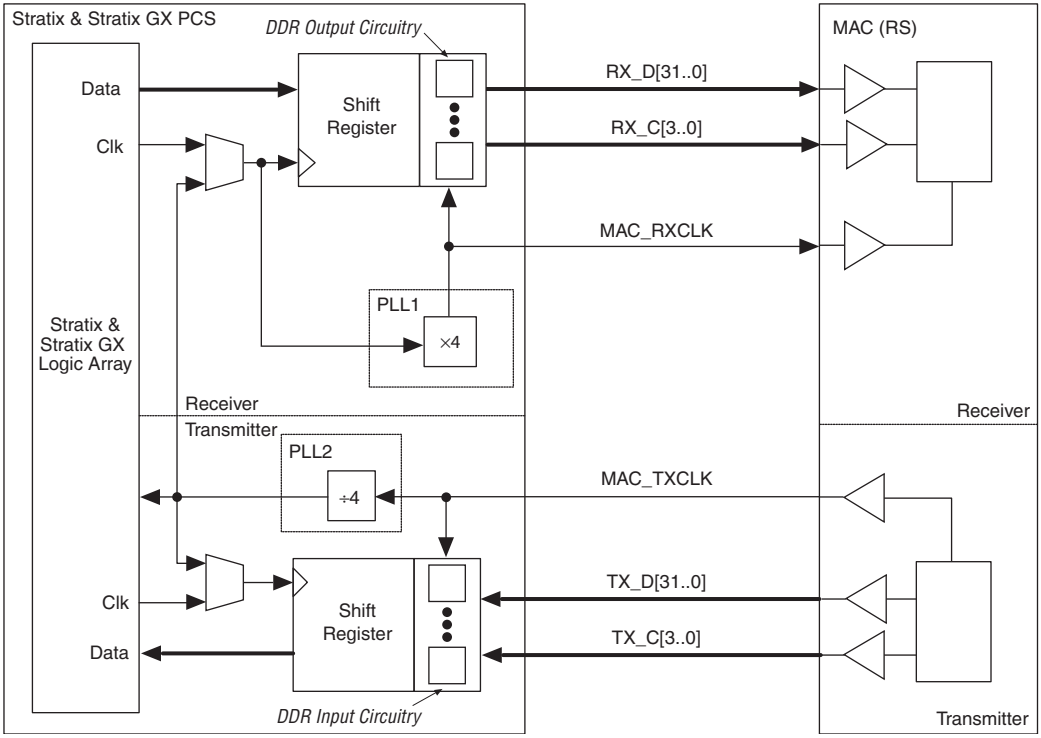
The 32 TXD and four TXC signals as well as the 32 RXD and four RXC signals are organized into four data lanes. The four lanes in each direction share a common clock (TX_CLK for transmit and RX_CLK for receive). The four lanes are used in round-robin sequence to carry an octet stream (8 bits of data per lane). The reconciliation sublayer generates continuous data or control characters on the transmit path and expects continuous data or control characters on the receive path.

Implementation

XGMII uses the 1.5-V HSTL I/O standard. Stratix and Stratix GX devices support the 1.5-V HSTL Class I and Class II I/O standard (EIA/JESD8-6). The standard requires a differential input with an external reference voltage (V_{REF}) of 0.75 V, as well as a termination voltage V_{TT} of 0.75 V, to which termination resistors are connected. The HSTL Class I standard requires a 1.5-V V_{CCIO} voltage, which is supported by Stratix and Stratix GX devices.

Figure 8–11 shows the 32-bit full-duplex 1.5-V HSTL implementation of XGMII.

Figure 8–11. Stratix & Stratix GX XGMII Implementation



For this implementation, the shift register clocks can either be generated from a divided down MAC reconciliation sublayer transmitter clock (MAC_TXCLK), or the asynchronous core clock, or both if using a FIFO buffer.

Figure 8–12 shows one channel of the output half of XGMII. Data that is transmitted from the PCS to the MAC reconciliation sublayer starts at the core of the Stratix or Stratix GX device and travels to the shift register. The shift register takes in the parallel data (even bits sent to the top register and odd bits sent to the bottom register) and serializes the data. After the data is serialized, it travels to the double data rate (DDR) output circuitry, which is clocked with the ×4 clock from the PLL. Out of the DDR output circuitry, the data drives off-chip along with the ×4 clock. This transaction creates the DDR relationship between the clock and the data output. This implementation only shows one channel, but can be duplicated to include all 32 bits of the RX_D signal and all 4 bits of the RX_C signal.

Figure 8–12. Stratix & Stratix GX XGMII Output Implementation (One Channel)

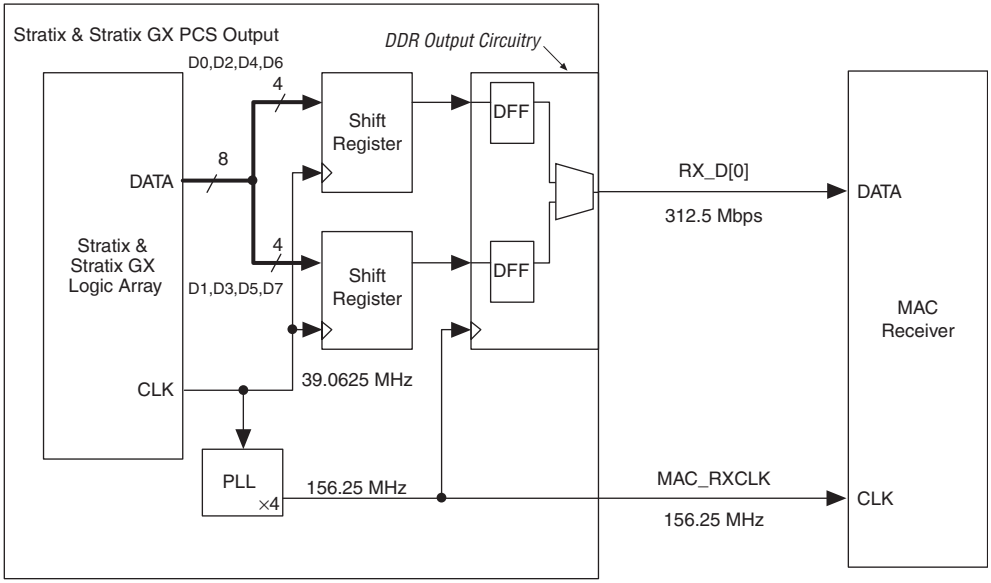
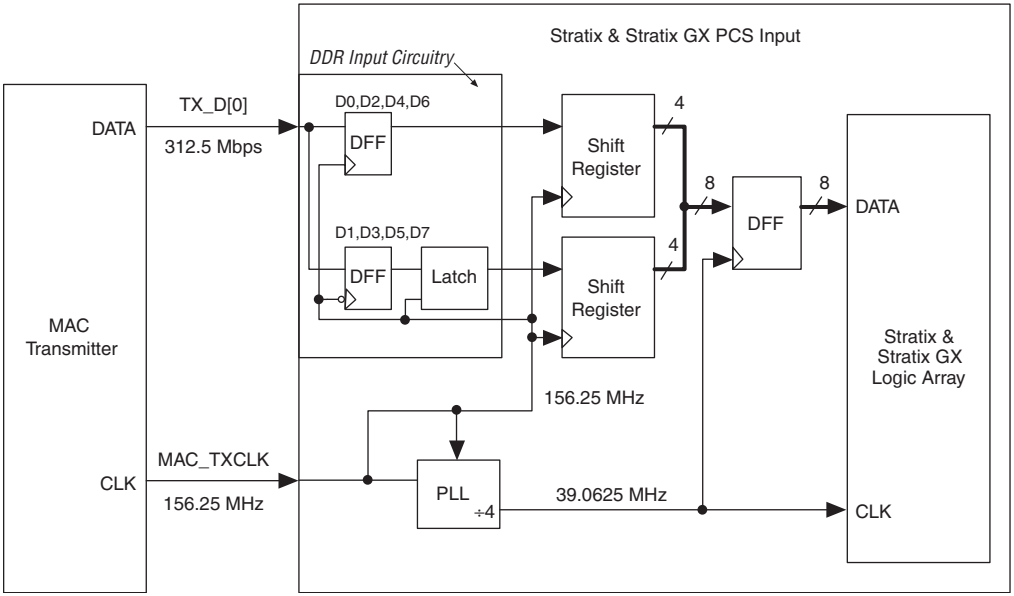


Figure 8–13 shows one channel of the input half of the XGMII interface. From the receiver side, the DDR data is captured from the MAC to the Stratix and Stratix GX PCS DDR input circuitry. The serial data is separated into two individual data streams with the even bits routed to the top register and odd bits routed to the bottom register. The DDR input circuitry produces two output data streams that go into the shift registers. From the shift registers, the data is deserialized using the clock from the MAC, combining into an 8-bit word. This parallel data goes to a register that is clocked by the divide-by-4 clock from the PLL. This data and clock go to the Stratix and Stratix GX core. This implementation shows only one channel, but can be duplicated to include all 32 bits of the TX_D signal and all 4 bits of the TX_C signal.

Figure 8–13. Stratix & Stratix GX XGMII Input Implementation (One Channel)



Stratix and Stratix GX devices contain up to four enhanced PLLs. These PLLs provide features such as clock switchover, spread-spectrum clocking, programmable bandwidth, phase and delay control, and PLL reconfiguration. Since the maximum clock rate is 156.25 MHz, you can use a fast or enhanced PLL for both the XGMII output and input blocks.

f For more information about fast PLLs, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

With this implementation for the XGMII output and input blocks, the number of XGMII cores per device corresponds to the number of PLLs each Stratix and Stratix GX device contains. Tables 8–6 and 8–7 show the number of 1.5-V HSTL I/O pins, PLLs, and XGMII cores that are supported in each Stratix and Stratix GX device. Each core requires 72 1.5-

V HSTL I/O pins for data and control and 2 clock pins for the transmitter and receiver clocks. Each XGMII core also needs two PLLs (one for each direction).

Table 8–6. Stratix XGMII Core Support

Stratix Device	Number of 1.5-V HSTL Class I I/O Pins	Number of Fast & Enhanced PLLs	Number of XGMII Interfaces
EP1S10	410	6	3
EP1S20	570	6	3
EP1S25	690	6	3
EP1S30	718	10	5
EP1S40	814	12	6
EP1S60	1,014	12	6
EP1S80	1,195	12	6

Table 8–7. Stratix GX XGMII Core Support

Stratix Device	Number of 1.5-V HSTL Class I I/O Pins	Number of Fast & Enhanced PLLs	Number of XGMII Interfaces
EP1SGX10 C, D	226	4	2
EP1SGX25 C	253	4	2
EP1SGX25 D, F	370	4	2
EP1SGX40 D, G	430	8	4

Reduced System Noise

The output buffer of each Stratix and Stratix GX device I/O pin has a programmable drive strength control for certain I/O standards. The 1.5-V HSTL Class I standard supports the minimum setting, which is the lowest drive strength that guarantees I_{OH} and I_{OL} of the standard. Using minimum settings provides signal slew rate control to reduce system noise and signal overshoot.

f For more information on I_{OH} and I_{OL} values, see *Operating Conditions* in the *DC & Switching Characteristics* chapter of the *Stratix Device Handbook, Volume 1* or *Operating Conditions* in the *DC & Switching Characteristics* chapter of the *Stratix GX Device Handbook, Volume 1*.

Timing

XGMII signals must meet the timing requirements shown in Figure 8–14. Make all XGMII timing measurements at the driver output (shown in Figure 8–14) and a capacitive load from all sources of 20 pF that are specified relative to the V_{IL}_AC(max) and V_{IH}_AC(min) thresholds.

Figure 8–14. XGMII Timing Diagram

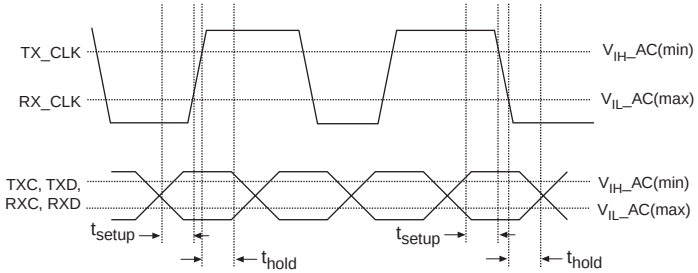


Table 8–8 shows the XGMII timing specifications.

Table 8–8. XGMII Timing Specifications Note (1)

Symbol	Driver	Receiver	Unit
T_{setup}	960	480	ps
T_{hold}	960	480	ps

Note to Table 8–8:

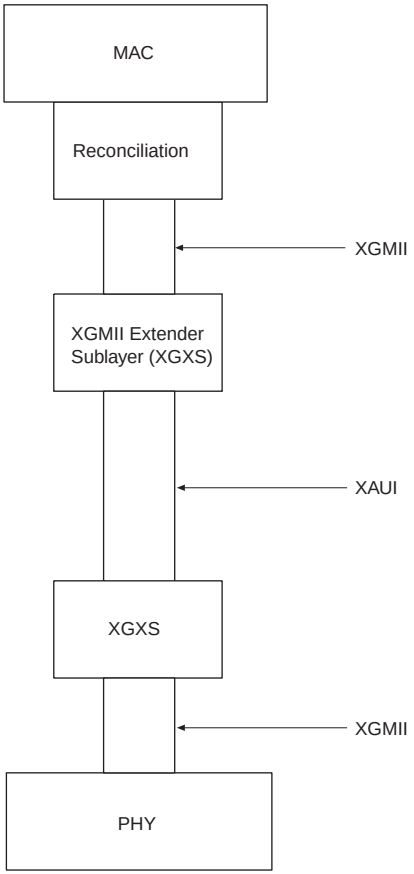
(1) The actual set-up and hold times will be made available after device characterization is complete.

Stratix and Stratix GX devices support DDR data with clock rates of up to 200 MHz, well above the XGMII clock rate of 156.25 MHz. For the HSTL Class I I/O standard, Stratix and Stratix GX device I/O drivers provide a 1.0-V/ns slew rate at the input buffer of the receiving device.

XAUI

XAUI (pronounced Zowie) is located between the XGMII at the reconciliation sublayer and the XGMII at the PHY layer. Figure 8–15 shows the location of XAUI. XAUI is designed to either extend or replace XGMII in chip-to-chip applications of most Ethernet MAC to PHY interconnects.

Figure 8–15. XAUI Location

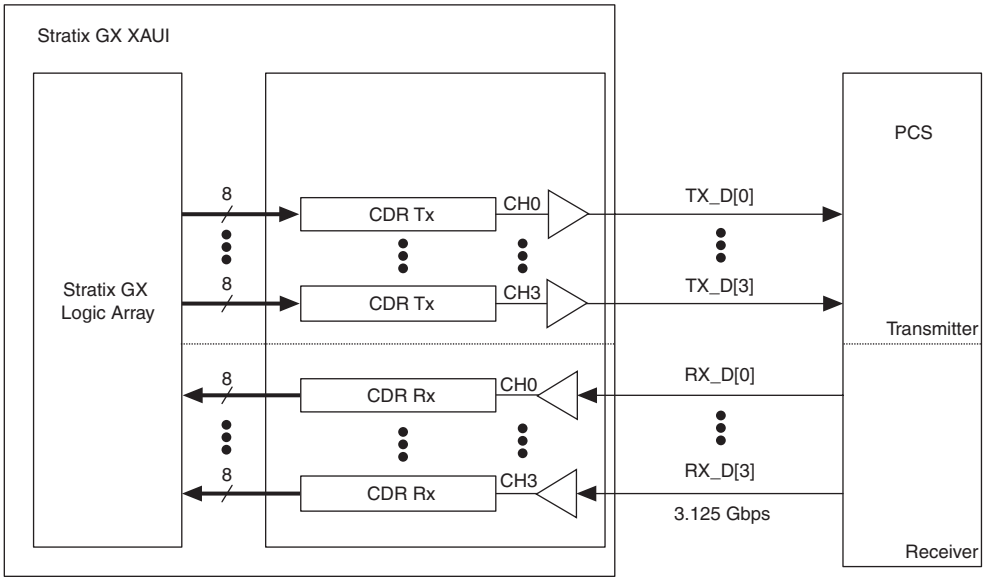


Functional Description

XAUI can replace the 32 bits of parallel data required by XGMII for transmission with just 4 lanes of serial data. XAUI uses clock data recovery (CDR) to eliminate the need for separate clock signals. 8b / 10b encoding is employed on the data stream to embed the clock in the data. The 8b / 10b protocol to encode an 8-bit word stream to 10-bit codes that results in a DC-balanced serial stream and eases the receiver synchronization. To support 10-Gigabit Ethernet, each lane must run at a speed of at least 2.5 Gbps. Using 8b / 10b encoding increases the rate for each lane to 3.125 Gbps, which will be supported in Stratix GX Gbps devices. This circuitry is supported by the embedded 3.125 Gbps transceivers within the Stratix GX architecture. You can find more

information on XAUI support in *Section II, Stratix GX Transceiver User Guide* of the *Stratix GX Device Handbook, Volume 2*. Figure 8–16 shows how XAUI is implemented.

Figure 8–16. Stratix GX XAUI Implementation



I/O Characteristics for XSBI, XGMII & XAUI

The three interfaces of 10-Gigabit Ethernet (XSBI, XGMII, and XAUI) each have different rates and I/O standards. Table 8–9 shows the characteristics for each interface.

Table 8–9. 10-Gigabit Ethernet Interfaces Characteristics

Interface	Width	Clock Rate (MHz)	Data Rate Per Channel	Clocking Scheme	I/O Type
XGMII	32	156.25	312.5 Mbps	DDR source synchronous	1.5-V HSTL
XSBI	16	644.5 or 622.08	644.5 or 622.08 Mbps	SDR source synchronous	LVDS
XAUI	4	None	3.125 Gbps	Clock data recovery (CDR)	1.5-V PCML

Software Implementation

You can use the **Assignment Organizer** in the Altera® Quartus® II software to implement the I/O standards for a particular interface. For example, set the I/O standard to LVDS for XSBI and to HSTL Class I for XGMII. You can use the MegaWizard® Plug-In Manager to create the PLLs and transmitter and receiver SERDES blocks for the XSBI implementation and PLLs and DDR input and output circuitry for the XGMII implementation. For more information on the Assignment Organizer or MegaWizard Plug-In Manager, see the Quartus II Software Help.

AC/DC Specifications

Table 8–10 lists the XSBI DC electrical characteristics, similar to Stratix and Stratix GX devices, that are based on the ANSI/TIA-644 LVDS specification.

Table 8–10. XSBI DC Specifications

Parameter	Value			Unit
	Min	Typ	Max	
Output differential voltage (V_{OD})	250		400 *(1)*	mV
Output offset voltage (V_{OS})	1,125		1,375	mV
Output Impedance, single ended	40		140	W
Change in V_{OD} between ‘0’ and ‘1’			50	mV
Change in V_{OS} between ‘0’ and ‘1’			50	mV
Input voltage range (V_I)	900		1,600	mV
Differential impedance		100		W
Input differential voltage (V_{ID})	100		600	mV
Receiver differential input impedance	70		130	W
Ground potential difference (between PCS and PMA)			50	mV
Rise and fall times (20% to 80%)	100		400	ps

Note to Table 8–10:
(1) Larger V_{OD} is possible for better signal intensity.

I/O characteristics for the 1.5-V HSTL standard for Stratix and Stratix GX devices are shown in Figure 8–17 and comply with XGMII electrical specifications available in 10-Gigabit Ethernet draft IEEE P802.3ae.

Figure 8–17. Electrical Characteristics for Stratix & Stratix GX Devices (1.5-V HSTL Class I)

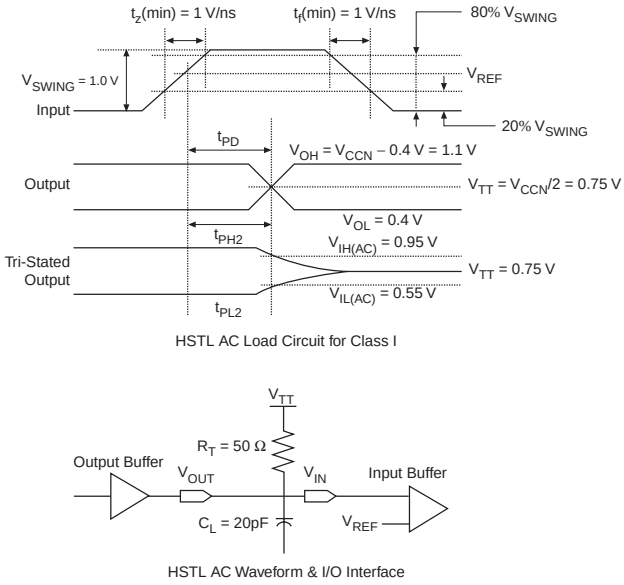


Table 8–11 lists the DC specifications for Stratix and Stratix GX devices (1.5-V HSTL Class I).

Table 8–11. DC Specifications for Stratix & Stratix GX Devices (1.5-V HSTL Class I) *Note (1)*

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		1.4	1.5	1.6	V
V_{REF}	Input reference voltage		0.68	0.75	0.9	V
V_{TT}	Termination voltage		0.7	0.75	0.8	V
V_{IH} (DC)	DC high-level input voltage		V_{REF} + 0.1			V
V_{IL} (DC)	DC low-level input voltage		–0.3		V_{REF} – 0.1	V
V_{IH} (AC)	AC high-level input voltage		V_{REF} + 0.2			V
V_{IL} (AC)	AC low-level input voltage				V_{REF} – 0.2	V
I_I	Input pin leakage current	$0 < V_{IN} < V_{CCIO}$	–10		10	µA
V_{OH}	High-level output voltage	I_{OH} = –8 mA	V_{CCIO} – 0.4			V
V_{OL}	Low-level output voltage	I_{OL} = 8 mA			0.4	V
I_O	Output leakage current (when output is high Z)	GND $\leq V_{OUT} \leq V_{CCIO}$	–10		10	µA

Note to Table 8–11:

(1) Drive strength is programmable according to values shown in the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

10-Gigabit Ethernet MAC Core

As an Altera Megafunction Partners Program (AMPP℠) member, MorethanIP provides a 10-Gigabit Ethernet MAC core for Altera customers. MorethanIP's 10-Gigabit Ethernet MAC core implements the RS, the MAC layer, and user-programmable FIFO buffers for clock and data decoupling.

Core Features

MorethanIP's 10-Gigabit Ethernet MAC core provides the following features:

- Includes automatic pause frame generation (per IEEE 802.3×31) with user-programmable pause quanta and pause-frame termination
- Includes a programmable 48-bit MAC address with a promiscuous mode option, and a programmable Ethernet frame length that supports IEEE 802.1Q VLAN-tagged frames or jumbo Ethernet frames

- Supports broadcast traffic and multi-cast address resolution with a 64-entry hash table
- Compliant with the IEEE802.3ae Draft 4.0
- Implements XGMII, allowing it to interface to XAUI through a 10-Gigabit commercial SERDES

Conclusion

10-Gigabit Ethernet takes advantage of the existing Gigabit Ethernet standard. With their rich I/O features, Stratix and Stratix GX devices support the components of 10-Gigabit Ethernet as well as XSBI and XGMII. Stratix GX devices also support XAUI. These interfaces are easily implemented using the core architecture, differential I/O capabilities, and superior PLLs of Stratix and Stratix GX devices.

9. Implementing SFI-4 in Stratix & Stratix GX Devices

S52011-2.0

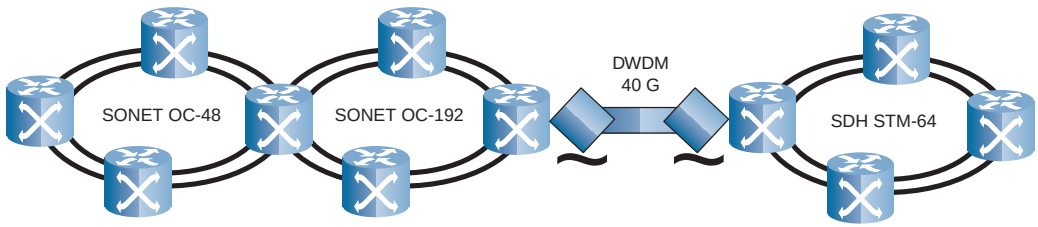
Introduction

The growth of the Internet has created huge bandwidth demands as voice, video, and data push the limits of the existing wide area network (WAN) backbones. To facilitate this bandwidth growth, speeds of OC-192 and higher are being deployed in WAN backbones (see Figure 9–1). Today's carrier backbone networks are supported by SONET/SDH transmission technology. SONET/SDH is a transmission technology for transporting optical signals at speeds ranging from 51 megabits per second (Mbps) up to 40 gigabits per second (Gbps). SONET/SDH rings make up the majority of the existing backbone infrastructure of the Internet and the public switched telephone network (PSTN).

The Optical Internetworking Forum (OIF) standard SFI-4 is a 16-bit LVDS interface used in an OC-192 SONET system to link the framer and the serializer/deserializer (SERDES). Stratix® and Stratix GX devices support the required data rates of up to 622.08 Mbps along with the one-to-one relationship required between clock frequency and data rate. The fast phase-locked loop (PLL) was designed to support the high clock frequencies and the one-to-one relationship (between clock and data rate) needed for interfaces such as XSBI and SFI-4. Support for SFI-4 extends the reach of high-density programmable logic from the backplane to the physical layer (PHY) devices.

This chapter focuses on the implementation of the interface between the SERDES and the framer.

Figure 9–1. WAN Backbone



A SONET/SDH transmission network is composed of several pieces of equipment, including terminal multiplexers, add-drop multiplexers, and repeater and digital cross-connect systems. SONET is the standard used in North America and SDH is the standard used outside North America.

The SONET/SDH specification outlines the frame format, multiplexing method, synchronization method, and optical interface between the equipment, as well as the specific optical interface.

SONET/SDH continues to play a key role in the next generation of networks for many carriers. In the core network, the carriers offer services such as telephone, dedicated leased lines, and Internet protocol (IP) data, which are continuously transmitted. The individual data channels are not transmitted on separate lines; instead, they are multiplexed into higher speeds and transmitted on SONET/SDH networks at the corresponding transmission speed.

Figure 9–2 shows a typical SONET/SDH line card. The system operates as follows:

1. The SONET/SDH line card first takes a high-speed serial optical signal and converts it into a high-speed serial electrical signal. The devices are called physical media dependent (PMD) devices.

2. The system then recovers the clock from the electrical data using a clock data recovery (CDR) unit.

3. The SERDES parallelizes the data so that it can be manipulated easily at lower clock rates.

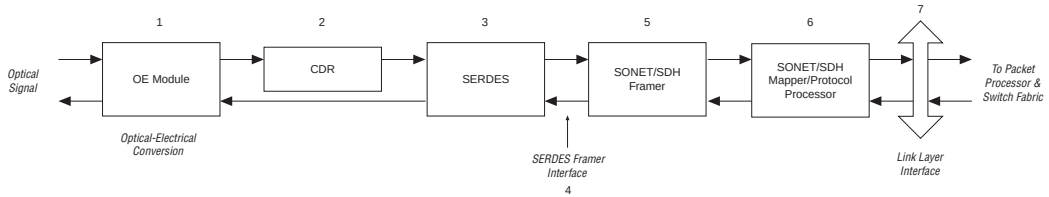
4. The interface between the SERDES and framer is called the SERDES framer interface. The interface requirements are defined by the OIF.

5. The framer identifies the beginning of the SONET/SDH frames and monitors the performance of the system.

6. The mapper following the framer maps asynchronous transfer mode (ATM) cells, IP packets, or T/E carrier signals into the SONET frame.

7. The PHY-link layer interface provides a bus interface to packet/cell processors or other link-layer devices.

Figure 9–2. SONET/SDH Line Card

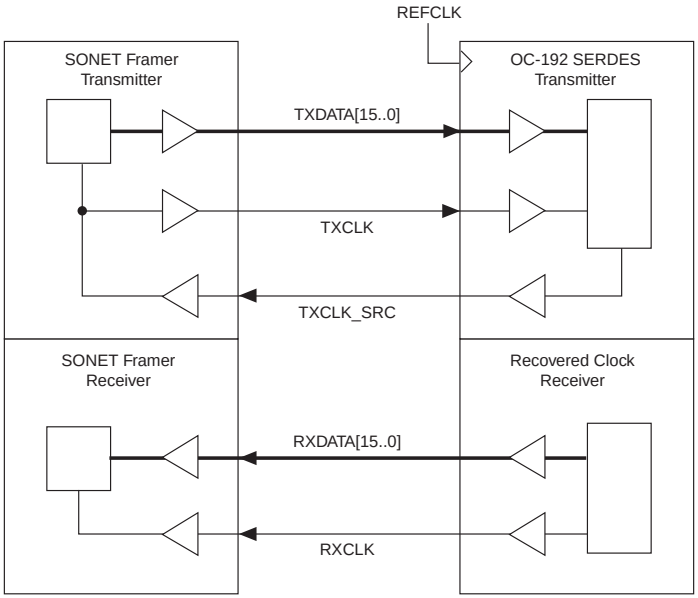


The OIF has defined the electrical interface (SFI) between the SONET/SDH framer and high-speed SERDES devices. To keep up with evolving transmission speeds and technology enhancements, different versions of electrical interfaces are defined. SFI-4 is the version of SFI that acts as an interface between an OC-192 SERDES and SONET framer, as shown in Figure 9–2. An aggregate of 9953.28 Mbps is transferred in each direction. With their differential I/O capabilities, Stratix and Stratix GX devices are ideally suited to support the framer side of the SFI-4 interface. Support for SFI-4 extends the reach of high-density programmable logic from the backplane to the PHY devices.

System Topology

The SFI-4 interface uses 16 channels of source-synchronous LVDS to interface between a SONET framer and an OC-192 SERDES. Figure 9–3 shows the SFI-4 interface.

Figure 9–3. SFI-4 Interface Signals



The framer transmits outbound data via TXDATA[15..0] and is received at the SERDES using TXCLK. TXCLK is derived from TXCLK_SRC, which is provided by the OC-192 SERDES. The framer receives incoming data on RXDATA[15..0] from the OC-192 SERDES. The data received is latched on the rising edge of RXCLK. Table 9–1 provides the data rates and clock frequencies specified by SFI-4. The modes of TXCLK are specified by the SFI-4 standard. In required mode (622 MHz clock mode or ×1 mode), TXCLK should run at 622.08 MHz. In optional mode (311 MHz clock mode or ×2 mode), TXCLK should run at 311.04 MHz.

Table 9–1. SFI-4 Interface Data Rates & Clock Frequencies

Signal	Performance
TXDATA[15..0]	622.08 Mbps
TXCLK	622.08 MHz or 311.04 MHz
TXCLK_SRC	622.08 MHz
RXDATA[15..0]	622.08 Mbps
RXCLK	622.08 MHz
REFCLK	622.08 MHz

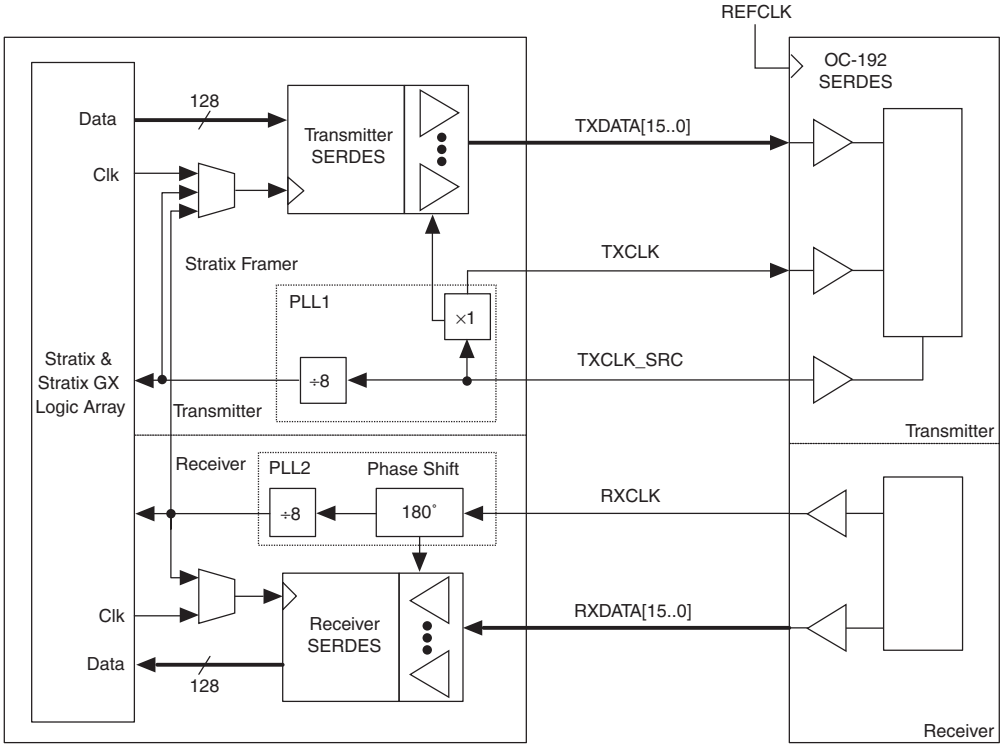
Interface Implementation in Stratix & Stratix GX Devices

The 16-bit full-duplex LVDS implementation of the framer part of the SFI-4 interface is shown in Figure 9–4. Stratix devices support source-synchronous interfacing and LVDS differential signaling up to 840 Mbps. Stratix devices have embedded SERDES circuitry for serial and parallel data conversion.

The source-synchronous I/O implemented in Stratix GX devices optionally includes dynamic phase alignment (DPA). DPA automatically and continuously tracks fluctuations caused by system variations and self-adjusts to eliminate the phase skew between the multiplied clock and the serial data, allowing for data rates of 1 Gbps. In non DPA mode the I/O behaves similarly to that of the Stratix I/O. This document assumes that DPA is disabled. However, it is simple to implement the same system with DPA enabled to take advantage of its features. For more information on DPA, see the *Stratix GX Transceivers* chapter in the *Stratix GX Device Handbook, Volume 1*.

The fast PLL enables 622.08 Mbps data transmission by transmitting and receiving a differential clock at rates of up to 645 MHz. The clocks required in the SERDES for parallel and serial data conversion can be configured from the received `RXCLK` (divided down), the `TXCLK_SRC` (divided down), or the asynchronous core clock. See Figure 9–4.

Figure 9–4. Implementation of SFI-4 Interface Using Stratix & Stratix GX Devices



f For details on differential I/O buffers, SERDES, and clock dividers using PLLs, see the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter in the *Stratix Device Handbook* or the *Stratix GX Device Handbook*.

Figure 9–5 shows the transmitter block (from Figure 9–4) of the SFI-4 framer interface implemented in Stratix and Stratix GX devices. The data starts in the logic array and goes into the Stratix and Stratix GX SERDES block. The transmitter SERDES of the framer converts the parallel data to serial data for the 16 `TXDATA` channels (`TXDATA[15..0]`). A fast PLL is used to generate `TXCLK` from `TXCLK_SRC`. The fast PLL keeps the `TXDATA` and `TXCLK` edge-aligned. A divided down (÷8) clock generated from `TXCLK_SRC` is used to convert the parallel data to serial in the transmitter SERDES. The divided down clock also clocks some of the logic in the logic array.

Figure 9–5. Framer Transmitter Interface in Stratix & Stratix GX Devices

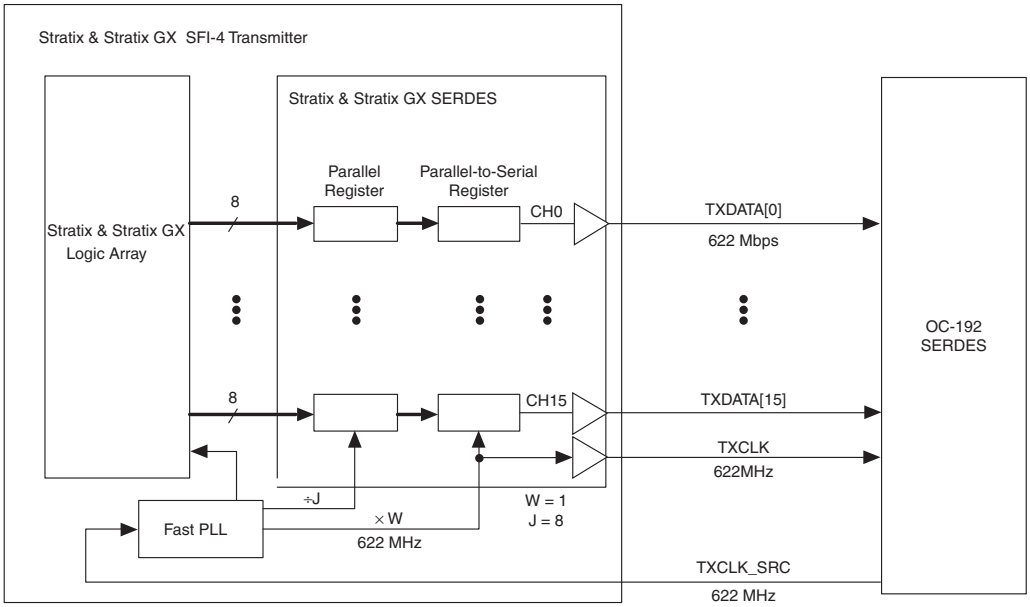
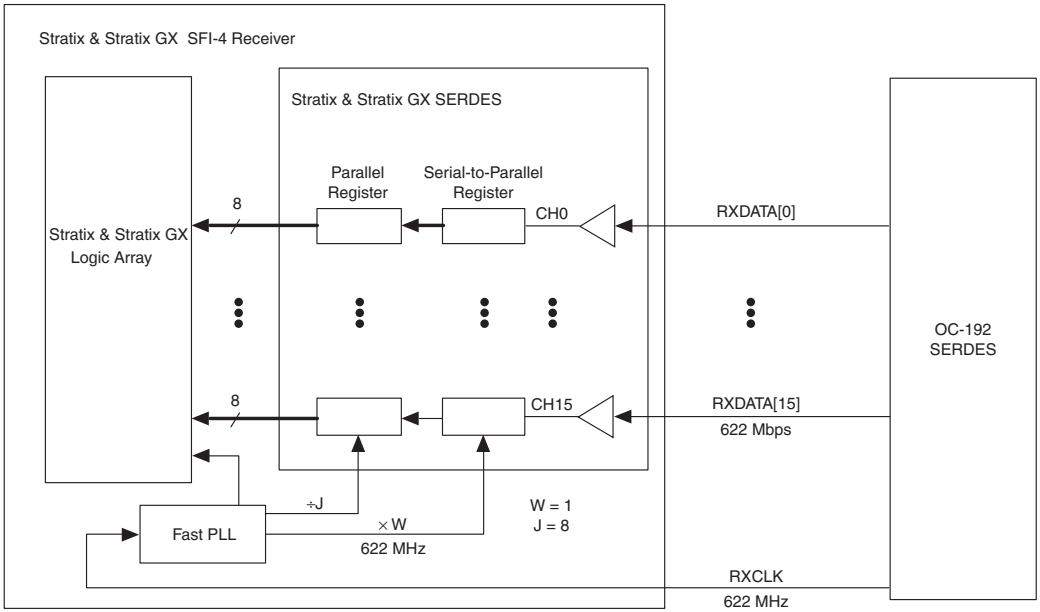


Figure 9–6 shows the receiver block (from Figure 9–4) of the SFI-4 framer interface implemented in Stratix and Stratix GX devices. `RXDATA[15..0]` is received from the OC-192 SERDES on the differential I/O pins of the Stratix or Stratix GX device. The receiver SERDES converts the high-speed serial data to parallel. You can generate the clocks required in the SERDES for parallel and serial data conversion from the received `RXCLK`. `RXCLK` is inverted (phase-shifted by 180°) to capture received data. While normal I/O operation guarantees that data is captured, it does not guarantee the parallelization boundary, which is randomly determined based on the power up of both communicating devices. The SERDES has embedded data realignment capability, which can be used to save logic elements (LEs).

Figure 9–6. Framer Receiver Interface in Stratix & Stratix GX Devices



Note to Figure 9–6:

(1) The figure shows Stratix GX DPA disabled.

f For more information on the byte-alignment feature in Stratix and Stratix GX devices, see the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter in the *Stratix Device Handbook* or the *Stratix GX Device Handbook*.

Tables 9–2 and 9–3 list the number of SFI-4 cores that can be implemented in Stratix and Stratix GX devices. See the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter in the *Stratix Device Handbook* or the *Stratix GX Device Handbook* for the package type and the maximum number of channels supported by each package.

Table 9–2. Stratix SFI-4 Core Support

Stratix Device	Number of LVDS Channels (Receiver/Transmitter) (1)	Number of PLLs	Number of SFI-4 Interfaces (Maximum)
EP1S10	44/44	4	2
EP1S20	66/66	4	2
EP1S25	78/78	4	2
EP1S30	82/82	8	4
EP1S40	90/90	8	4
EP1S60	116/116	8	4
EP1S80	152/156	8	4

Note to Table 9–2:

(1) The LVDS channels can go up to 840 Mbps (or 1 Gbps using DPA in Stratix GX devices). This number includes both high speed and low speed channels. The high speed LVDS channels can go up to 840 Mbps. The low speed LVDS channels can go up to 462 Mbps. The *High-Speed Differential I/O Support* chapters in the *Stratix Device Handbook, Volume 1* and the *Stratix GX Device Handbook, Volume 1* and the device pin-outs on the web (**www.altera.com**) specify which channels are high and low speed.

Table 9–3. Stratix GX SFI-4 Core Support

Stratix GX Device	Number of LVDS Channels (Receiver/Transmitter) (1)	Number of PLLs	Number of SFI-4 Interfaces (Maximum)
EP1SGX10	22/22	2	1
EP1SGX25	39/39	2	2
EP1SGX40	45/45	4	2

Note to Table 9–3:

(1) The LVDS channels can go up to 840 Mbps, or 1 Gbps using DPA. This number includes both high speed and low speed channels. The high speed LVDS channels can go up to 840 Mbps. The low speed LVDS channels can go up to 462 Mbps. The *High-Speed Differential I/O Support* chapter in the *Stratix Device Handbook, Volume 1* and the *Stratix GX Device Handbook, Volume 1* and the device pin-outs on the web (**www.altera.com**) specify which channels are high and low speed.

AC Timing Specifications

Figures 9–7 through 9–9 and Tables 9–4 through 9–6 illustrate the timing characteristics of SFI-4 at the framer. Stratix and Stratix GX devices support all the timing requirements needed to support transmitter and receiver functions of a SFI-4 framer; only framer-related timing specifications are applicable.

f For details on the timing specifications of LVDS I/O standards in Stratix and Stratix GX devices, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* and the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1* and the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter

Figure 9–7 shows the timing diagram for the Stratix and Stratix GX framer transmitter ×1 (622 MHz clock) mode.

Figure 9–7. Framer Transmitter ×1 (622 MHz Clock) Mode Timing Diagram

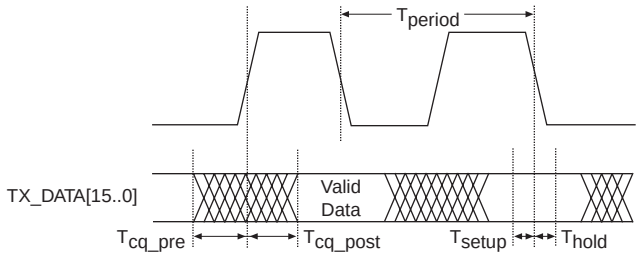


Table 9–4 lists the timing specifications for the SFI-4 framer transmitter in ×1 (622 MHz clock) mode.

Table 9–4. SFI-4 Framer Transmitter ×1 (622 MHz Clock) Mode Timing Specifications

Parameter	Value			Unit
	Min	Typ	Max	
TX_CLK (T_{period})		1,608		ps
Data invalid window before the rising edge (T_{cq_pre})			200	ps
Data invalid window after the rising edge (T_{cq_post})			200	ps
TX_CLK duty cycle	40		60	%
Framer transmitter channel-to-channel skew			200	ps

Figure 9–8 shows the timing diagram for the SFI-4 framer transmitter in ×2 (311 MHz clock) mode

Figure 9–8. Framer Transmitter ×2 (311 MHz Clock) Mode Timing Diagram

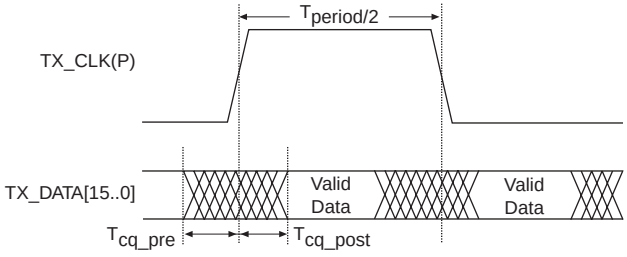


Table 9–5 lists the timing specifications for the SFI-4 framer transmitter in ×2 (311 MHz clock) mode.

Table 9–5. SFI-4 Framer Transmitter ×2 (311 MHz Clock) Mode Timing Specifications

Parameter	Value			Unit
	Min	Typ	Max	
TX_CLK (T_{period})		3,215		ps
Data invalid window before the rising edge (T_{cq_pre})			200	ps
Data invalid window after the rising edge (T_{cq_post})			200	ps
TX_CLK duty cycle	48		52	%
Framer transmitter channel-to-channel skew			200	ps

Figure 9–9 shows the timing diagram for the SFI-4 framer receiver.

Figure 9–9. Framer Receiver Timing Diagram

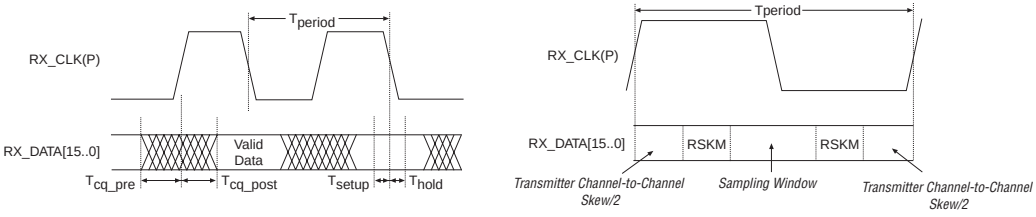


Table 9–6 lists the timing specifications for the SFI-4 framer receiver.

Table 9–6. Framer Receiver Timing Specifications

Parameter	Value			Unit
	Min	Typ	Max	
RX_CLK (T_{period})		1,608		ps
Data invalid window before the rising edge (T_{cq_pre})			200	ps
Data invalid window after the rising edge (T_{cq_post})			200	ps
RX_CLK duty cycle	45		55	%
Data set-up time (T_{setup})	300			ps
Data hold time (T_{hold})	300			ps
Framer sampling window	600			ps
Receiver skew margin (RSKM)			304	ps

Electrical Specifications

SFI-4 uses LVDS as a high-speed data transfer mechanism to implement the SFI-4 interface. Table 9–7 lists the DC electrical characteristics for the interface, which are based on the IEEE Std. 1596.3-1996 7 specification. For more information on the voltage specification of LVDS I/O standards in Stratix and Stratix GX devices, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* and the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1* and the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter.

Table 9–7. Framer LVDS DC Specifications

Parameter	Value			Unit
	Min	Typ	Max	
Output differential voltage (V_{OD})	250		600 *(1)*	mV
Output offset voltage (V_{OS})	1,125		1,375	mV
Output Impedance, single ended	40		140	W
Change in V_{OD} between ‘0’ and '1'			50	mV
Change in V_{OD} between '1' and '0'			50	mV
Input voltage range (V_I)	0		2,400	mV
Differential impedance		100		W
Input differential voltage (V_{ID})	100		600	mV
Receiver differential input impedance	70		130	W
Ground potential difference (between PCS and PMA)			50	mV
Rise and fall times (20% to 80%)	100		400	ps

Note to Table 9–7:

(1) The IEEE standard requires 400 mV. A larger swing is encouraged, but not required.

Software Implementation

The SFI-4 interface uses a 16-bit LVDS I/O interface. The Altera® Quartus® II software version 2.0 supports Stratix and Stratix GX devices, allowing you to implement LVDS I/O buffers through the Quartus II Assignment Organizer.

f For information on the Quartus II Assignment Organizer, see the Quartus II Software Help.

Conclusion

SFI-4 is the standard interface between SONET framers and optical SERDES for OC-192 interfaces. With embedded SERDES and fast PLLs, Stratix and Stratix GX devices can easily support the SFI-4 framer interface, enabling
10-Gbps (OC-192) data transfer rates. Stratix and Stratix GX I/O supports the required data rates of up to 622.08 Mbps. Stratix and Stratix GX fast PLLs are designed to support the high clock frequencies and one-to-one relationship needed for interfaces such as XSBI and SFI-4. Stratix and Stratix GX devices can support multiple SFI-4 functions on one device.

10. Transitioning APEX Designs to Stratix & Stratix GX Devices

S52012-3.0

Introduction

Stratix® and Stratix GX devices are Altera's next-generation, system-on-a-programmable-chip (SOPC) solution. Stratix and Stratix GX devices simplify the block-based design methodology and bridge the gap between system bandwidth requirements and programmable logic performance.

This chapter highlights the new features in the Stratix and Stratix GX devices and provides assistance when transitioning designs from APEX™ II or APEX 20K devices to the Stratix or Stratix GX architecture. You should be familiar with the APEX II or APEX 20K architecture and available device features before using this chapter. Use this chapter in conjunction with the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

General Architecture

Stratix and Stratix GX devices offer many new features and architectural enhancements. Enhanced logic elements (LEs) and the MultiTrack™ interconnect structure offer reduced resource utilization and considerable design performance improvement. The MultiTrack interconnect uses DirectDrive™ technology to ensure the availability of deterministic routing resources for any design block, regardless of its placement within the device.

All architectural changes between Stratix and Stratix GX and APEX II or APEX 20K devices described in this section do not require any design changes. However, you must resynthesize your design and recompile in the Quartus® II software to target Stratix and Stratix GX devices.

Logic Elements

Stratix and Stratix GX device LEs include several new, advanced features that improve design performance and reduce logic resource consumption (see Table 10–1). The Quartus II software automatically uses these new LE features to improve device utilization.

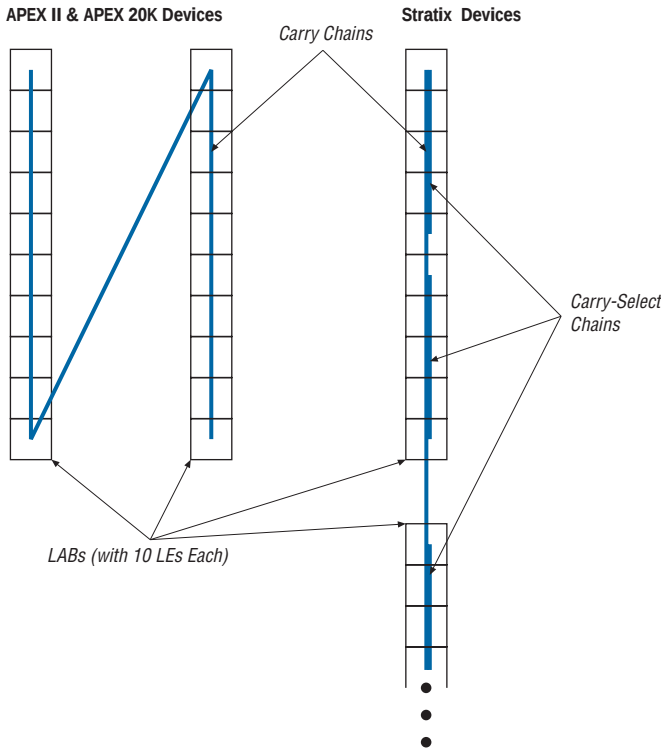
Table 10–1. Stratix & Stratix GX LE Features

Feature	Function	Benefit
Register chain interconnects	Direct path between the register output of an LE and the register input of an adjacent LE within the same logic array block (LAB)	■ Conserves LE resources ■ Provides fast shift register implementation ■ Saves local interconnect routing resources within an LAB
Look-up table (LUT) chain interconnects	Direct path between the combinatorial output of an LE and the fast LUT input of an adjacent LE within the same LAB	■ Allows LUTs within the same LAB to cascade together for high-speed wide fan-in functions, such as wide XOR operations ■ Bypasses local interconnect for faster performance
Register-to-LUT feedback path	Allows the register output to feed back into the LUT of the same LE, such that the register is packed with its own fan-out LUT	■ Enhanced register packing mode ■ Uses resources more efficiently
Dynamic arithmetic mode	Uses one set of LEs for implementing both an adder and subtractor	■ Improves performance for functions that switch between addition and subtraction frequently, such as correlators
Carry-select chain	Calculates outputs for a possible carry-in of 1 or 0 in parallel	■ Gives immediate access to result for both a carry-in of 1 or 0 ■ Increases speed of carry functions for high-speed operations, such as counters, adders, and comparators
Asynchronous clear and asynchronous preset function	Supports direct asynchronous clear and preset functions	■ Conserves LE resources ■ Does not require additional logic resources to implement NOT-gate push-back

In addition to the new LE features described in Table 10–1, there are enhancements to the chains that connect LEs together. Carry chains are implemented vertically in Stratix and Stratix GX devices, instead of horizontally as in APEX II and APEX 20K devices, and continue across rows, instead of across columns, as shown in Figure 10–1. Also note that the Stratix and Stratix GX architectures do not support the cascade primitive. Therefore, the Quartus II Compiler automatically converts

cascade primitives in APEX II and APEX 20K designs to a wire primitive when compiled for Stratix and Stratix GX devices. These architectural changes are transparent to the user and do not require design changes.

Figure 10–1. Carry Chain Implementation in APEX II & APEX 20K Devices vs. Stratix & Stratix GX Devices



MultiTrack Interconnect

Stratix and Stratix GX devices use the MultiTrack interconnect structure to provide a high-speed connection between logic resources using performance-optimized routing channels of different lengths. This feature maximizes overall design performance by placing critical paths on routing lines with greater speed, resulting in minimal propagation delay.

Stratix and Stratix GX device MultiTrack interconnect resources are described in Table 10–2.

Table 10–2. Stratix & Stratix GX Device MultiTrack Interconnect Resources

Routing Type	Interconnect	Span
Row	Direct link	Adjacent LABs and/or blocks
Row	R4	Four LAB units horizontally
Row	R8	Eight LAB units horizontally
Row	R24	Horizontal routing across the width of the device
Column	C4	Four LAB units vertically
Column	C8	Eight LAB units vertically
Column	C16	Vertical routing across the length of the device

Direct link routing saves row routing resources while providing fast communication paths between resource blocks. Direct link interconnects allow an LAB, digital signal processing (DSP) block, or TriMatrix™ memory block to drive data into the local interconnect of its left and right neighbors. LABs, DSP blocks, and TriMatrix memory blocks can also use direct link interconnects to drive data back into themselves for feedback.

The Quartus II software automatically uses these routing resources to enhance design performance.

f For more information about LE architecture and the MultiTrack interconnect structure in Stratix and Stratix GX devices, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

DirectDrive Technology

When using APEX II or APE 20K devices, you must place critical paths in the same MegaLAB™ column to improve performance. Additionally, you should place critical paths in the same MegaLAB structure for optimal performance. However, this restriction does not exist in Stratix and Stratix GX devices because they do not contain MegaLAB structures. With the new DirectDrive™ technology in Stratix and Stratix GX devices, the actual distance between the source and destination of a path is the most important criteria for meeting timing performance. DirectDrive technology ensures that the same routing resources are available to each design block, regardless of its location in the device.

Architectural Element Names

The architectural element naming system within Stratix and Stratix GX devices differs from the row-column coordinate system (for example, LC1_A2, LAB_B1) used in previous Altera device families. Stratix and Stratix GX devices uses a new naming system based on the X-Y coordinate system, (X, Y). A number (N) designates the location within the block where the logic resides, such as LEs within an LAB. Because the Stratix and Stratix GX architectures are column-based, this naming simplifies location assignments. Stratix and Stratix GX architectural blocks include:

- LAB: logic array block
- DSP: digital signal processing block
- DSPOUT: adder / subtractor / accumulator or summation block of the DSP block
- M512: 512-bit memory block
- M4K: 4-Kbit memory block
- M-RAM: 512-Kbit memory block

Elements within architectural blocks include:

- LE: logic element
- IOC: I / O element
- PLL: phase-locked loop
- DSPMULT: DSP block multiplier
- SERDESTX: transmitter serializer / deserializer
- SERDESRX: receiver serializer / deserializer

Table 10–3 highlights the new location syntax used for Stratix and Stratix GX devices.

Table 10–3. Stratix & Stratix GX Location Assignment Syntax

Architectural Elements	Element Name	Location Syntax	Example of Location Syntax	
			Location	Description
Blocks	LAB, DSP, DSPOUT, M512, M4K, M-RAM	<element_name>_X<number>_Y<number>	LAB_X1_Y1	Designates the LAB in row 1, column 1
Logic	LE, IOC, PLL, DSPMULT, SERDESTX, SERDESRX	<element_name>_X<number>_Y<number>_N<number>	LC_X1_Y1_N0	Designates the first LE, N0, in the LAB located in row 1, column 1
Pins *(1)*	I/O pins	pin_<pin_label>	pin_5	Pin 5

Note to Table 10–3:

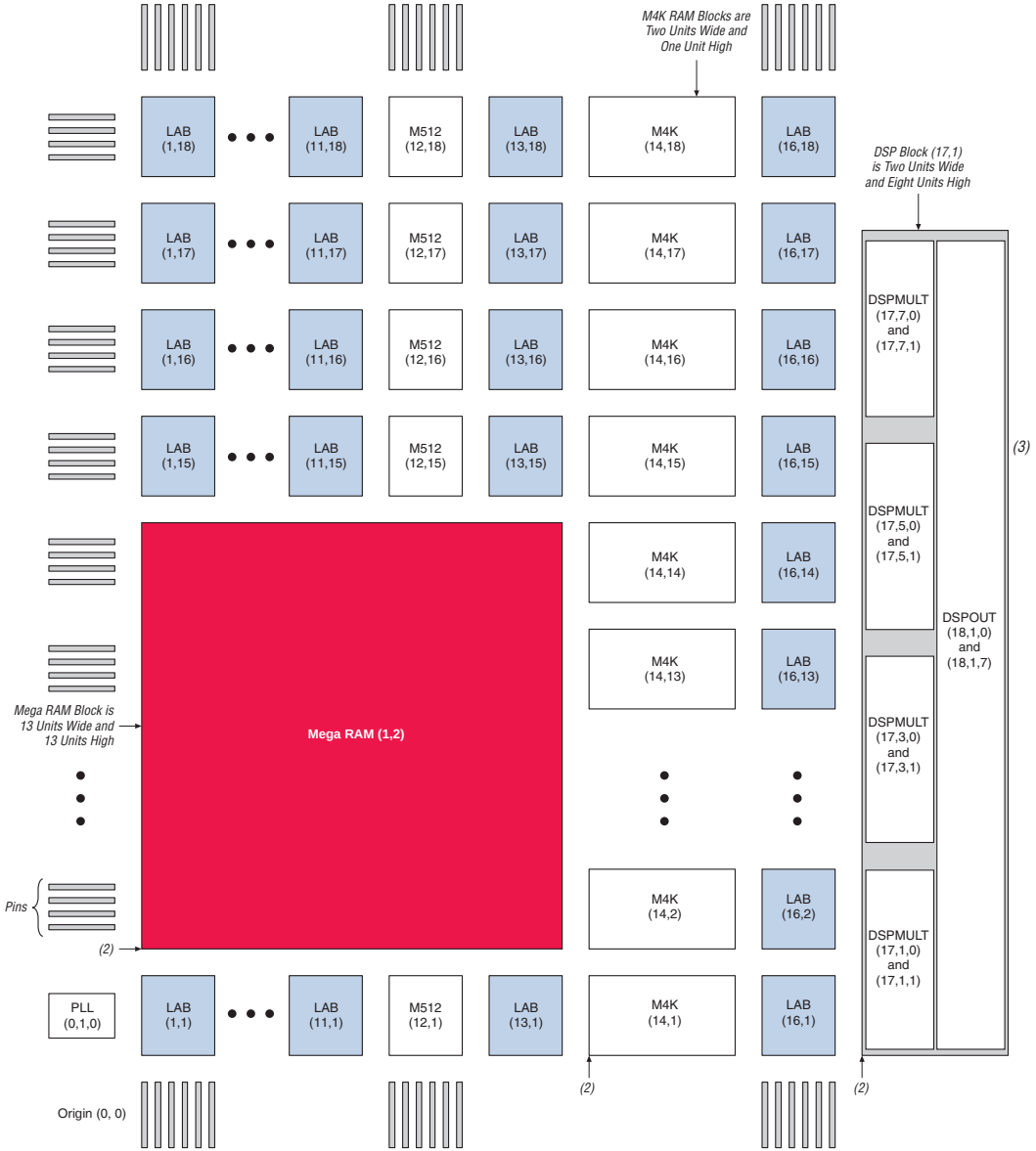
(1) You can make assignments to I/O pads using IOC_X<*number*>_Y<*number*>_N<*number*>.

Use the following guidelines with the new naming system:

- The anchor point, or origin, in Stratix and Stratix GX devices is in the bottom-left corner, instead of the top-left corner as in APEX II and APEX 20K devices.
- The anchor point, or origin, of a large block element (e.g., a M-RAM or DSP block) is also the bottom-left corner.
- All numbers are zero-based, meaning the origin at the bottom-left of the device is X0, Y0.
- The I/O pins constitute the first and last rows and columns in the X-Y coordinates. Therefore, the bottom row of pins resides in X<*number*>, Y0, and the first left column of pins resides in X0, Y<*number*>.
- The sub-location of elements, N, numbering begins at the top. Therefore, the LEs in an LAB are still numbered from top to bottom, but start at zero.

Figure 10–2 show the Stratix and Stratix GX architectural element numbering convention. Figure 10–3 displays the floorplan view in the Quartus II software.

Figure 10–2. Stratix & Stratix GX Architectural Elements *Note (1)*



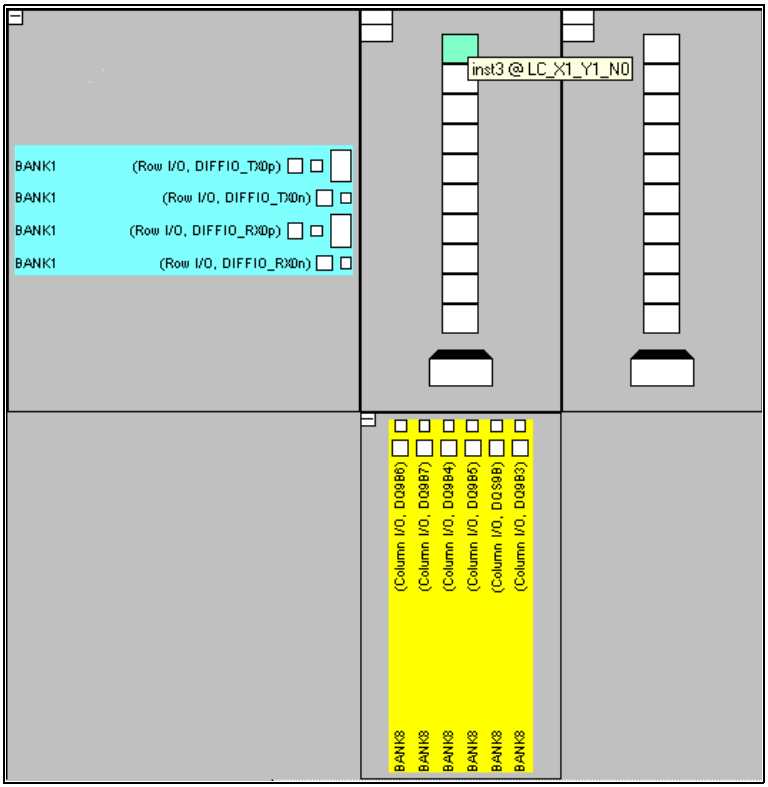
Notes to Figure 10–2:

(1) Figure 10–2 shows part of a Stratix and Stratix GX device.

(2) Large block elements use their lower-left corner for the coordinate location.

(3) The Stratix GX architectural elements include transceiver blocks on the right side of the device.

Figure 10–3. LE Numbering as Shown in the Quartus II Software



TriMatrix Memory

TriMatrix memory has three different sizes of memory blocks, each optimized for a different purpose or application. M512 blocks consist of 512 bits plus parity (576 bits), M4K blocks consist of 4K bits plus parity (4,608 bits), and M-RAM blocks consist of 512K bits plus parity (589,824 bits). This new structure differs from APEX II and APEX 20K devices, which feature uniformly sized embedded system blocks (ESBs) either 4 Kbits (APEX II devices) or 2 Kbits (APEX 20K devices) large. Stratix and Stratix GX TriMatrix memory blocks give you advanced control of each memory block, with features such as byte enables, parity bit storage, and shift-register mode, as well as mixed-port width support and true dual-port mode operation.

Table 10–4 compares TriMatrix memory with ESBs.

Table 10–4. Stratix & Stratix GX TriMatrix Memory Blocks vs. APEX II & APEX 20K ESBs

Features	Stratix & Stratix GX			APEX II ESB	APEX 20K ESB
	M512 RAM	M4K RAM	M-RAM		
Size (bits)	576	4,608	589,824	4,096	2,048
Parity bits	Yes	Yes	Yes	No	No
Byte enable	No	Yes	Yes	No	No
True dual-port mode	No	Yes Includes support for mixed width	Yes Includes support for mixed width	Yes Includes support for mixed width	No
Embedded shift register	Yes	Yes	No	No	No
Dedicated content-addressable memory (CAM) support	No	No	No	Yes	Yes
Pre-loadable initialization with a **.mif** *(1)*	Yes	Yes	No	Yes	Yes
Packed mode *(2)*	No	Yes	No	Yes	Yes
Feed-through behavior	Rising edge	Rising edge	Rising edge	Falling edge	Falling edge
Output power-up condition	Powers up cleared even if using a **.mif** *(1)*	Powers up cleared even if using a **.mif** *(1)*	Powers up with unknown state	Powers up cleared or to initialized value, if using a **.mif** *(1)*	Powers up cleared or to initialized value, if using a **.mif** *(1)*

Notes to Table 10–4:

(1) **.mif**: Memory Initialization File.

(2) Packed mode refers to combining two single-port RAM blocks into a single RAM block that is placed into true dual-port mode.

Stratix and Stratix GX TriMatrix memory blocks only support pipelined mode, while APEX II and APEX 20K ESBs support both pipelined and flow-through modes. Since all TriMatrix memory blocks can be pipelined, all input data and address lines are registered, while outputs can be either registered or combinatorial. You can use Stratix and Stratix GX memory block registers to implement input and output registers without utilizing additional resources. You can compile designs containing pipelined memory blocks (inputs registered) for Stratix and Stratix GX devices without any modifications. However, if an APEX II or

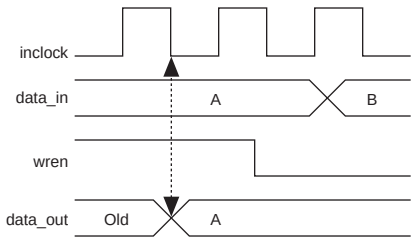
APEX 20K design contains flow-through memory, you must modify the memory modules to target the Stratix and Stratix GX architectures (see "Memory Megafunctions" on page 10–12 for more information).

f For more information about TriMatrix memory and converting flow-through memory modules to pipelined, see the *TriMatrix Embedded Memory Blocks in Stratix & Stratix GX Devices* chapter in the *Stratix GX Device Handbook* and *AN 210: Converting Memory from Asynchronous to Synchronous for Stratix & Stratix GX Designs*.

Same-Port Read-During-Write Mode

In same-port read-during-write mode, the RAM block can be in single-port, simple dual-port, or true dual-port mode. One port from the RAM block both reads and writes to the same address location using the same clock. When APEX II or APEX 20K devices perform a same-port read-during-write operation, the new data is available on the falling edge of the clock cycle on which it was written, as shown in Figure 10–4. When Stratix and Stratix GX devices perform a same-port read-during-write operation, the new data is available on the rising edge of the same clock cycle on which it was written, as shown in Figure 10–5. This holds true for all TriMatrix memory blocks.

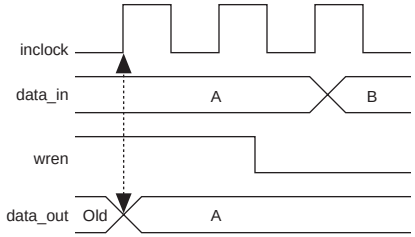
Figure 10–4. Falling Edge Feed-Through Behavior (APEX II & APEX 20K Devices) *Note (1)*



Note to Figure 10–4:

(1) Figures 10–4 and 10–5 assume that the address stays constant throughout and that the outputs are not registered.

Figure 10–5. Rising Edge Feed-Through Behavior (Stratix & Stratix GX Devices) *Note (1)*



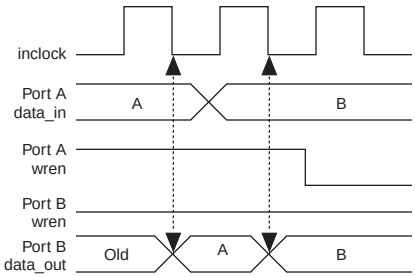
Note to Figure 10–5:

(1) Figures 10–4 and 10–5 assume that the address stays constant throughout and that the outputs are not registered.

Mixed-Port Read-During-Write Mode

Mixed-port read-during-write mode occurs when a RAM block in simple or true dual-port mode has one port reading and the other port writing to the same address location using the same clock. In APEX II and APEX 20K designs, the ESB outputs the old data in the first half of the clock cycle and the new data in the second half of the clock cycle, as indicated by Figure 10–6.

Figure 10–6. Mixed-Port Feed-Through Behavior (APEX II & APEX 20K Devices) *Note (1)*



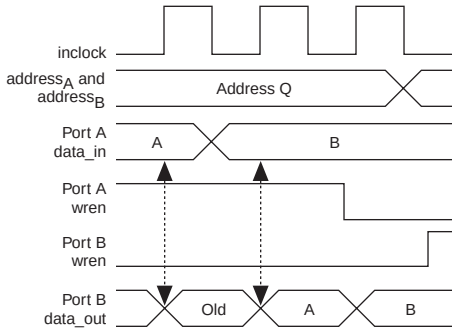
Note to Figure 10–6:

(1) Figure 10–6 assumes that outputs are not registered.

Stratix and Stratix GX device RAM outputs the new data on the rising edge of the clock cycle immediately after the data was written. When you use Stratix and Stratix GX M512 and M4K blocks, you can choose whether to output the old data at the targeted address or output a don't care value during the clock cycle when the new data is written. M-RAM blocks

always output a don't care value. Figures 10–7 and 10–8 show the feed-through behavior of the mixed-port mode. You can use the `altsyncram` megafunction to set the output behavior during mixed-port read-during-write mode.

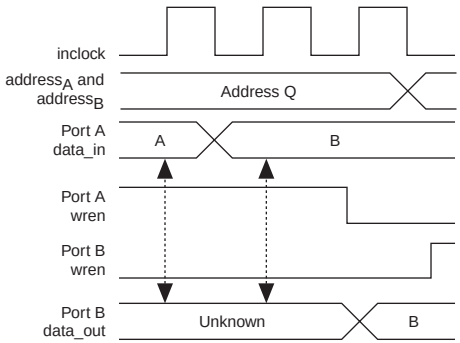
Figure 10–7. Mixed-Port Feed-Through Behavior (OLD_DATA) *Note (1)*



Note to Figure 10–7:

(1) Figures 10–7 and 10–8 assume that the address stays constant throughout and that the outputs are not registered.

Figure 10–8. Mixed-Port Feed-Through Behavior (DONT_CARE) *Note (1)*



Note to Figure 10–8:

(1) Figures 10–7 and 10–8 assume that the address stays constant throughout and that the outputs are not registered.

Memory Megafunctions

To convert RAM and ROM originally targeting the APEX II or APEX 20K architecture to Stratix or Stratix GX memory, specify Stratix or Stratix GX as the target family in the MegaWizard Plug-In Manager. The software

updates the memory module for the Stratix or Stratix GX architecture and instantiates the new synchronous memory megafunction, `altsyncram`, which supports both RAM and ROM blocks in the Stratix and Stratix GX architectures.

FIFO Conditions

First-in first-out (FIFO) functionality is slightly different in Stratix and Stratix GX devices compared to APEX II and APEX 20K devices. Stratix and Stratix GX devices do not support simultaneous reads and writes from an empty FIFO buffer. Also, Stratix and Stratix GX devices do not support the `lpm_showahead` parameter when targeting a FIFO buffer because the TriMatrix memory blocks are synchronous. The `lpm_showahead` parameter for APEX II and APEX 20K devices puts the FIFO buffer in "read-acknowledge" mode so the first data written into the FIFO buffer immediately flows through to the output. Other than these two differences, all APEX II and APEX 20K FIFO functions are fully compatible with the Stratix and Stratix GX architectures.

Design Migration Mode in Quartus II Software

The Quartus II software features a migration mode for simplifying the process of converting APEX II and APEX 20K memory functions to the Stratix or Stratix GX architecture. If the design can use the Stratix or Stratix GX `altsyncram` megafunction as a replacement for a previous APEX II or APEX 20K memory function while maintaining functionally similar behavior, the Quartus II software automatically converts the memory. The software produces a warning message during compilation reminding you to verify that the design migrated correctly.

For memory blocks with all inputs registered, the existing megafunction is converted to the new `altsyncram` megafunction. The software generates a warning when the `altsyncram` megafunction is incompatible. For example, a RAM block with all inputs registered except the read enable compiles with a warning message indicating that the read-enable port is registered.

You can suppress warning messages for the entire project or for individual memory blocks by setting the `SUPPRESS_MEMORY_CONVERSION_WARNINGS` parameter to "on" as a global parameter by selecting **Assignment Organizer** (Tools menu). In the **Assignment Organizer** window, click **Parameters** in the **Assignment Categories** box. Type `SUPPRESS_MEMORY_CONVERSION_WARNINGS` in the **Assignment Name** box and type `ON` in the **Assignment Setting** box. To suppress these warning messages on a per-memory-instance basis, set the `SUPPRESS_MEMORY_CONVERSION_WARNINGS` parameter in the Assignment Organizer to "on" for the memory instance.

If the functionality of the APEX II or APEX 20K memory megafunction differs from the `altsyncram` functionality and at least one clock feeds the memory megafunction, the Quartus II software converts the APEX II or APEX 20K memory megafunction to the Stratix or Stratix GX `altsyncram` megafunction. This conversion is useful for an initial evaluation of how a design might perform in Stratix or Stratix GX devices and should only be used for evaluation purposes. During this process, the Quartus II software generates a warning that the conversion may be functionally incorrect and timing results may not be accurate. Since the functionality may be incorrect and the compilation is only intended for comparative purposes, the Quartus II software does not generate a programming file. A functionally correct conversion requires manually instantiating the `altsyncram` megafunction and may require additional design changes.

If the previous memory function does not have a clock (fully asynchronous), the fitting-evaluation conversion results in an error message during compilation and does not successfully convert the design.

f See *AN 210: Converting Memory from Asynchronous to Synchronous for Stratix & Stratix GX Designs* for more information.

Table 10–5 summarizes the possible scenarios when using design migration mode and the resulting behavior of the Quartus II software.

The most common cases where design-migration mode may have difficulty converting the existing design are when:

- A port is reading from an address that is being written to by another port (mixed-port read-during-write mode). If both ports are using the same clock, the read port in Stratix and Stratix GX devices do not see the new data until the next clock cycle, after the new data was written.

- There are differences in power-up behavior between APEX II, APEX 20K, and Stratix and Stratix GX devices. You should manually account for these differences to maintain desired operation of the system.

Table 10–5. Migration Mode Summary

Memory Configuration	Conditions	Possible Instantiated Megafunctions	Quartus II Warning Message(s)	Programming File Generated

Single-port	All inputs are registered.	`altram`
		`altrom`
		`lpm_ram_dq`
		`lpm_ram_io`
		`lpm_rom`
			Power-up differences. *(1)*	Yes

Multi-port (two-, three-, or four-port functions)	All inputs are registered.	`altdpram`
		`lpm_ram_dp`
		`altqpram`
		`alt3pram`
			Power-up differences.
			Mixed-port read- during-write. *(1)*
				Yes

Dual-port	Read-enable ports are unregistered.
	Other inputs registered.
		`altdpram`
		`lpm_ram_dp`
		`altqpram`
		`alt3pram`
			Power-up differences.
			Mixed-port read- during-write.
			Read enable will be registered. *(1)*
				Yes

Dual-port	Any other unregistered port except read-enable ports.
	Clock available.
		`altdpram`
		`lpm_ram_dp`
		`altqpram`
		`alt3pram`
			Compile for fitting- evaluation purposes.	No

Single-port	At least one registered input.
	Clock available.
		`altram`
		`lpm_ram_dq`
		`lpm_ram_io`
			Compile for fitting- evaluation purposes.	No

No clock	No clock.	`altram`
		`altrom`
		`altdpram`
		`altqpram`
		`alt3pram`
		`altdpram`
		`lpm_ram_dq`
		`lpm_ram_io`
		`lpm_rom`
		`lpm_ram_dp`
		`lpm_ram_dp`
			Error – no conversion possible.	No

Note to Table 10–5:

(1) If the `SUPPRESS_MEMORY_COUNVERSION_WARNINGS` parameter is turned on, the Quartus II software does not issue these warnings.

DSP Block

Stratix and Stratix GX device DSP blocks outperform LE-based implementations for common DSP functions. Each DSP block contains several multipliers that can be configured for widths of 9, 18, or 36 bits. Depending on the mode of operation, these multipliers can optionally feed an adder / subtractor / accumulator or summation unit.

You can configure the DSP block's input registers to efficiently implement shift registers for serial input sharing, eliminating the need for external shift registers in LEs. You can add pipeline registers to the DSP block for accelerated operation. Registers are available at the input and output of the multiplier, and at the output of the adder / subtractor / accumulator or summation block.

DSP blocks have four modes of operation:

- Simple multiplier mode
- Multiply-accumulator mode
- Two-multipliers adder mode
- Four-multipliers adder mode

Associated megafunctions are available in the Quartus II software to implement each mode of the DSP block.

DSP Block Megafunctions

You can use the `lpm_mult` megafunction to configure the DSP block for simple multiplier mode. You can set the `lpm_mult` **Multiplier Implementation** option in the MegaWizard Plug-In Manager to either use the default implementation, ESBs, or the DSP blocks. If you select the **Use Default** option, the compiler first attempts to place the multiplier in the DSP blocks. However, under certain conditions, the compiler may implement the multiplier in LEs. The placement depends on factors such as DSP block resource consumption, the width of the multiplier, whether an operand is a constant, and other options chosen for the megafunction.

Stratix and Stratix GX devices do not support the **Use ESBs** option. If you select this option, the Quartus II software tries to place the multiplier in unused DSP blocks.

You can recompile APEX II or APEX 20K designs using the `lpm_mult` megafunction for Stratix and Stratix GX devices in the Quartus II software without changing the megafunction. This makes converting `lpm_mult` megafunction designs to Stratix or Stratix GX devices straightforward.

APEX II and APEX 20K designs use pipeline stages to improve registered performance of LE-based multipliers at the expense of latency. However, you may not need to use pipeline stages when targeting Stratix and Stratix GX high-speed DSP blocks. The DSP blocks offer three sets of dedicated pipeline registers. Therefore, Altera recommends that you reduce the number of pipeline stages to three or fewer and implement them in the DSP blocks. Additional pipeline stages are implemented in LEs, which add latency without providing any performance benefit.

For example, you can configure a DSP block for 36 × 36-bit multiplication using the `lpm_mult` megafunction. If you specify two pipeline stages, the software uses the DSP block input and pipeline registers. If you specify three pipeline stages, the software places the third pipeline stage in the DSP block output registers. This design yields the same performance with three pipeline stages because the critical path for a 36 × 36-bit operation is within the multiplier. With four or more pipeline stages, the device inefficiently uses LE resources for the additional pipeline stages. Therefore, if multiplier modules in APEX II or APEX 20K designs are converted to Stratix or Stratix GX designs and do not require the same number of pipeline stages, the surrounding circuitry must be modified to preserve the original functionality of the design.

A design with multipliers feeding an accumulator can use the `altmult_accum (MAC)` megafunction to set the DSP block in multiply-accumulator mode. If the APEX II or APEX 20K design already uses LE-based multipliers feeding an accumulator, the Quartus II software does not automatically instantiate the new `altmult_accum (MAC)` megafunction. Therefore, you should use the MegaWizard Plug-In Manager to instantiate the `altmult_accum (MAC)` megafunction. You can also use LeonardoSpectrum™ or Synplify synthesis tools, which have DSP block inference support, to instantiate the megafunction.

Designs that use multipliers feeding into adders can instantiate the new `altmult_add` megafunction to configure the DSP blocks for two-multipliers adder or four-multipliers adder mode. You can also use the `altmult_add` megafunction for stand-alone multipliers to take advantage of the DSP blocks features such as dynamic sign control of the inputs and the input shift register connections. These features are not accessible through the `lpm_mult` megafunction. If your APEX II or APEX 20K designs already use multipliers feeding an adder / subtractor, the Quartus II software does not automatically infer the new `altmult_add` megafunction. Therefore, you should step through the MegaWizard Plug-In Manager to instantiate the new `altmult_add` megafunction or use LeonardoSpectrum or Synplify synthesis tools, which have DSP block inference support.

Furthermore, the `altmult_add` and `altmult_accum (MAC)` megafunctions are only available for Stratix and Stratix GX devices because these megafunctions target Stratix and Stratix GX DSP blocks, which are not available in other device families. If you attempt to use these megafunctions in designs that target other Altera device families, the Quartus II software reports an error message. Use `lpm_mult` and an `lpm_add_sub` or an `altaccumulate` megafunction for similar functionality in other device families.

If you use a third-party synthesis tool, you may be able to avoid the megafunction conversion process. LeonardoSpectrum and Synplify provide inference support for `lpm_mult`, `altmult_add`, and `altmult_accum (MAC)` to use the DSP blocks.

If your design does not require you to implement all the multipliers in DSP blocks, you must manually set a global parameter or a parameter for each instance to force the tool to implement the `lpm_mult` megafunction in LEs. Depending on the synthesis tools, inference of DSP blocks is handled differently.

f For more information about using DSP blocks in Stratix and Stratix GX devices, see the *DSP Blocks in Stratix & Stratix GX Devices* chapter of the *Stratix Device Handbook*.

PLLs & Clock Networks

Stratix and Stratix GX devices provide exceptional clock management with a hierarchical clock network and up to four enhanced phase-locked loops (PLLs) and eight fast PLLs versus the four general-purpose PLLs and four True-LVDS™ PLLs in APEX II devices. By providing superior clock interfacing, numerous advanced clocking features, and significant enhancements over APEX II and APEX 20K PLLs, the Stratix and Stratix GX device PLLs increase system performance and bandwidth.

Clock Networks

There are 16 global clock networks available throughout each Stratix or Stratix GX device as well as two fast regional and four regional clock networks per device quadrant, resulting in up to 40 unique clock networks per device. The increased number of dedicated clock resources available in Stratix and Stratix GX devices eliminate the need to use general-purpose I/O pins as clock inputs.

Stratix EP1S25 and smaller devices have 16 dedicated clock pins and EP1S30 and larger devices have four additional clock pins to feed various clocking networks. In comparison, APEX II devices have eight dedicated clock pins and APEX 20KE and APEX 20KC devices have four dedicated clock pins.

The dedicated clock pins in Stratix and Stratix GX devices can feed the PLL clock inputs, the global clock networks, and the regional clock networks. PLL outputs and internally-generated signals can also drive the global clock network. These global clocks are available throughout the entire device to clock all device resources.

Stratix and Stratix GX devices are divided into four quadrants, each equipped with four regional clock networks. The regional clock network can be fed by either the dedicated clock pins or the PLL outputs within its device quadrant. The regional clock network can only feed device resources within its particular device quadrant.

Each Stratix and Stratix GX device provides eight dedicated fast clock I/O pins `FCLK[7..0]` versus four dedicated fast I/O pins in APEX II and APEX 20K devices. The fast regional clock network can be fed by these dedicated `FCLK[7..0]` pins or by the I/O interconnect. The I/O interconnect allows internal logic or any I/O pin to drive the fast regional clock network. The fast regional clock network is available for general-purpose clocking as well as high fan-out control signals such as clear, preset, enable, `TRDY` and `IRDY` for PCI applications, or bidirectional or output pins.

EP1S25 and smaller devices have eight fast regional clock networks, two per device quadrant. The quadrants in EP1S30 and larger devices are divided in half, and each half-quadrant can be clocked by one of the eight fast regional networks. Additionally, each fast regional clock network can drive its neighboring half-quadrant (within the same device quadrant).

PLLs

Table 10–6 highlights Stratix and Stratix GX PLL enhancements to existing APEX II, APEX 20KE and APEX 20KC PLL features.

Table 10–6. Stratix & Stratix GX PLL vs. APEX II, APEX 20KE & APEX 20KC PLL Features (Part 1 of 2)

Feature	Stratix & Stratix GX		APEX II PLLs	APEX 20KE & APEX 20KC PLLs
	Enhanced PLLs	Fast PLLs		
Number of PLLs	Two (EP1S30 and smaller devices); four (EP1S40 and larger devices) *(9)*	Four (EP1S25 and smaller devices); eight (EP1S30 and larger devices) *(10)*	Four general-purpose PLLs and four LVDS PLLs	Up to four general-purpose PLLs. Up to two LVDS PLLs. *(1)*
Minimum input frequency	3 MHz	15 MHz	1.5 MHz	1.5 MHz
Maximum input frequency	250 to 582 MHz *(2)*	644.5 MHz *(11)*	420 MHz	420 MHz

Table 10–6. Stratix & Stratix GX PLL vs. APEX II, APEX 20KE & APEX 20KC PLL Features (Part 2 of 2)

Feature	Stratix & Stratix GX		APEX II PLLs	APEX 20KE & APEX 20KC PLLs
	Enhanced PLLs	Fast PLLs		
Internal clock outputs per PLL	6	3 *(3)*	2	2
External clock outputs per PLL	Four differential/eight singled-ended or one single-ended *(4)*	Yes *(5)*	1	1
Phase Shift	Down to 160-ps increments *(6)*	Down to 125-ps increments *(6)*	500-ps to 1-ns resolution	0.4- to 1-ns resolution
Time shift	250-ps increments for ± 3 ns *(7)*	No	No	No
M counter values	1 to 512	1 to 32	1 to 160	2 to 160
N counter values	1 to 512	N/A	1 to 16	1 to 16
PLL clock input sharing	No	Yes	Yes	Yes
T1/E1 rate conversion *(8)*	No	No	Yes	Yes

Notes to Table 10–6:

(1) EP20K200E and smaller devices only have two general-purpose PLLs. EP20K400E and larger devices have two LVDS PLLs and four general-purpose PLLs. For more information, see *AN 115: Using the ClockLock & ClockBoost PLL Features in APEX Devices*.

(2) The maximum input frequency for Stratix and Stratix GX enhanced PLLs depends on the I/O standard used with that input clock pin. For more information, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1*.

(3) Fast PLLs 1, 2, 3, and 4 have three internal clock output ports per PLL. Fast PLLs 7, 8, 9, and 10 have two internal clock output ports per PLL.

(4) Every Stratix device has two enhanced PLLs with eight single-ended or four differential outputs each. Two additional enhanced PLLs in EP1S80, EP1S60, and EP1S40 devices each have one single-ended output.

(5) Any I/O pin can be driven by the fast PLL global or regional outputs as an external clock output pin.

(6) The smallest phase shift unit is determined by the voltage-controlled oscillator (VCO) period divided by 8.

(7) There is a maximum of 3 ns between any two PLL clock outputs.

(8) The T1 clock frequency is 1.544 MHz and the E1 clock frequency is 2.048 MHz, which violates the minimum clock input frequency requirement of the Stratix PLL.

(9) Stratix GX EP1SGX10 and EP1SGX25 contain two. EP1SGX40 contains four.

(10) Stratix GX EP1SGX10 and EP1SGX25 contain two. EP1SGX40 contains four.

(11) Stratix GX supports clock rates of 1 Gbps using DPA.

Enhanced PLLs

Stratix and Stratix GX devices provide up to four enhanced PLLs with advanced PLL features. In addition to the feature changes mentioned in Table 10–6, Stratix and Stratix GX device PLLs include many new,

advanced features to improve system timing management and performance. Table 10–7 shows some of the new features available in Stratix and Stratix GX enhanced PLLs.

Table 10–7. Stratix & Stratix GX Enhanced PLL Features

Feature	Description
Programmable duty cycle *(1)*	Allows variable duty cycle for each PLL clock output.
PLL clock outputs can feed logic array *(1)*	Allows the PLL clock outputs to feed data ports of registers or combinatorial logic.
PLL locked output can feed the logic array *(1)*	Allows the PLL locked port to feed data ports of registers or combinatorial logic.
Multiplication allowed in zero-delay buffer mode or external feedback mode	The PLL clock outputs can be a multiplied or divided down ratio of the PLL input clock.
Programmable phase shift allowed in zero-delay buffer mode or external feedback mode *(2)*	The PLL clock outputs can be phase shifted. The phase shift is relative to the PLL clock output.
Phase frequency detector (PFD) disable	Allows the VCO to operate at its last set control voltage and frequency with some long term drift.
Clock output disable *(3)*	PLL maintains lock with output clocks disabled. *(4)*
Programmable lock detect & gated lock	Holds the lock signal low for a programmable number of input clock cycles.
Dynamic clock switchover	Enables the PLL to switch between two reference input clocks, either for clock redundancy or dual-clock domain applications.
PLL reconfiguration	Allows the counters and delay elements within the PLL to be reconfigured in real-time without reloading a programmer object file (**.pof**).
Programmable bandwidth	Provides advanced control of the PLL bandwidth by using the programmable control of the PLL loop characteristics.
Spread spectrum	Modulates the target frequency over a frequency range to reduce electromagnetic interference (EMI) emissions.

Notes to Table 10–7:

(1) These features are also available in fast PLLs.

(2) In addition to the delay chains at each counter, you can specify the programmable phase shift for each PLL output at fine and coarse levels.

(3) Each PLL clock output has an associated clock enable signal.

(4) If the PLL is used in external feedback mode, the PLL will need to relock.

Fast PLLs

Stratix and Stratix GX fast PLLs are similar to the APEX II True-LVDS PLLs in that the *W* setting, which governs the relationship between the clock input and the data rate, and the *J* setting, which controls the width

of the high-speed differential I/O data bus, do not have to be equal. Additionally, Stratix and Stratix GX fast PLLs offer up to three clock outputs, two multiplied high-speed PLL clocks to drive the serializer/deserializer (SERDES) block and/or an external pin, and a low-speed clock to drive the logic array. You can use fast PLLs for both high-speed interfacing and for general-purpose PLL applications.

Table 10–8 shows the differences between Stratix and Stratix GX fast PLLs and APEX II and APEX 20K True-LVDS PLLs.

Table 10–8. Stratix & Stratix GX Fast PLL vs. APEX II & APEX 20K True-LVDS PLL

Feature	Stratix & Stratix GX	APEX II	APEX 20KE APEX 20KC
Number of fast PLLs or True-LVDS PLLs *(1)*	Four (EP1S25 and smaller devices) fast PLLs Eight (EP1S30 and larger devices) fast PLLs *(4)*	Four True-LVDS PLLs	Two True-LVDS PLLs *(2)*
Number of channels per transmitter/receiver block	20	18	18
VCO frequency	300 to 840 MHz *(5)*	200 MHz to 1GHz	200 to 840 MHz
Minimum input frequency *M* = 4, 5, 6	300 – *M* MHz	50 MHz	50 MHz *M* = 4 *(3)*
Minimum input frequency *M* = 7, 8, 9, 10	300 – *M* MHz	30 MHz	30 MHz *M* = 7, 8 *(3)*

Notes to Table 10–8:

(1) You can also use Stratix and Stratix GX device fast PLLs for general-purpose PLL applications.
(2) EP20K400E and larger devices have two True-LVDS PLLs.
(3) In APEX 20KE and APEX 20KC devices, *M* = 4, 7, or 8.
(4) Stratix GX EP1SGX10 and EP1SGX25 contain two. EP1SGX10 contains four.
(5) Stratix GX supports a frequency range of 300–1000 MHz (using DPA).

The Stratix and Stratix GX fast PLL VCO frequency range is 300 to 840 MHz, and the APEX II True-LVDS PLL VCO frequency range is 200 MHz to 1 GHz. Therefore, you must update designs that use a data rate of less than 300 megabits per second (Mbps) to use the enhanced PLLs and M512 RAM blocks in SERDES bypass mode. Additionally, you must update designs that use a data rate faster than 840 Mbps.

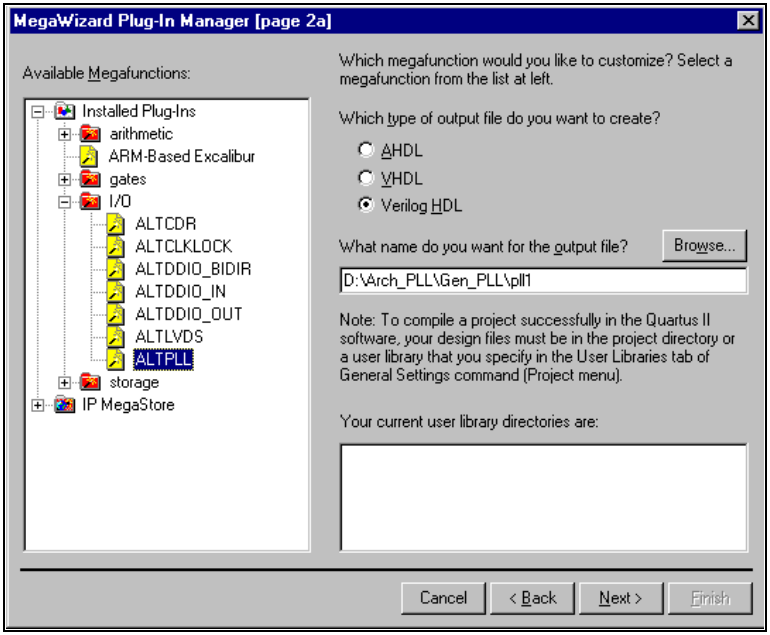
altpll Megafunction

Altera recommends that you replace instances of the `altclklock` megafunction with the `altpll` megafunction to take advantage of new Stratix and Stratix GX PLL features. Although in most cases you can retarget your APEX II or APEX 20K design to a Stratix or Stratix GX

device with the `altclklock` megafunction, there are specific cases where you must use the `altpll` megafunction, as explained in this section.

In the MegaWizard Plug-In Manager, select the `altpll` megafunction in the I/O directory from the **Available Megafunctions** box (see Figure 10–9). The `altclklock` megafunction is also available from the Quartus II software for backward compatibility, but instantiates the new `altpll` megafunction when targeting Stratix or Stratix GX devices. The Quartus II Compiler automatically selects whether the `altpll` module uses either an enhanced PLL or a fast PLL based on the design's PLL needs and the feature requirements of each PLL.

Figure 10–9. altpll Megafunction Selection in the MegaWizard Plug-In Manager



You can compile APEX II, APEX 20KE, and APEX 20KC designs using the `altclklock` megafunction in normal mode for Stratix and Stratix GX devices without updating the megafunction. However, you should replace the `altclklock` megafunction with the `altpll` megafunction. If the Quartus II software cannot implement the requested clock multiplication and division of the PLL, the compiler reports an error message with the appropriate reason stated.

APEX II, APEX 20KE, and APEX 20KC devices have only one external clock output available per PLL. Therefore, when retargeting an APEX II, APEX 20KE, or APEX 20KC design that uses PLLs in zero delay buffer mode or external feedback mode to a Stratix or Stratix GX device, you should replace instances of the `altclklock` megafunction. If an APEX II, APEX 20KE, or APEX 20KC `altclklock` module only uses one PLL clock output (internal or external) and is compiled to target a Stratix or Stratix GX device, the design compiles successfully with a warning that the design uses the Stratix or Stratix GX PLL external clock output, `extclk0`. However, if the APEX II, APEX 20KE, or APEX 20KC PLL has more than one PLL clock output, you must replace instances of the `altclklock` megafunction with the `altpll` megafunction because the Quartus II Compiler does not know which PLL clock output is fed to an external output pin or fed back to the Stratix or Stratix GX device `fbin` pin. For example, if an APEX II, APEX 20KE, or APEX 20KC design with an `altclklock` megafunction uses the `clock0` output port to feed the external clock output pin and the `clock1` output port to feed the internal logic array, the Quartus II software generates an error during compilation and you must use the MegaWizard Plug-In Manager to instantiate the `altpll` megafunction. By using the `altpll` megafunction, you can choose which of the four external clock outputs to use and take advantage of the new Stratix and Stratix GX PLL features now available in the zero delay buffer mode or external feedback mode.

Timing Analysis

When the Quartus II software performs a timing analysis for APEX II, APEX 20KE, or APEX 20KC designs, PLL clock settings override the project clock settings. However, during timing analysis for Stratix and Stratix GX designs using PLLs, the project clock settings override the PLL input clock frequency and duty cycle settings. The MegaWizard Plug-In Manager does not use the project clock settings to determine the `altpll` parameters. This saves time with designs that use features such as clock switchover or PLL reconfiguration because the Quartus II software can perform a timing analysis without recompiling the design. It is important to note the following:

- A warning during compilation reports that the project clock settings overrides the PLL clock settings.
- The project clock setting overrides the PLL clock settings for timing-driven compilation.
- The compiler will check the lock frequency range of the PLL. If the frequency specified in the project clock settings is outside the lock frequency range, the PLL clock settings will not be overridden.
- Performing a timing analysis without recompiling your design does not change the programming files. You must recompile your design to update the programming files.

- A **Default Required f_{MAX}** setting does not override the PLL clock settings. Only individual clock settings override the PLL clock settings.

Therefore, you can enter different project clock settings corresponding to new PLL settings and accelerate timing analysis by eliminating a full compilation cycle.

f For more information about using Stratix and Stratix GX PLLs, see the *General-Purpose PLLs in Stratix & Stratix GX Devices* chapter.

I/O Structure

The Stratix and Stratix GX I/O element (IOE) architecture is similar to the APEX II architecture, with a total of six registers and a latch in each IOE. The registers are organized in three sets: two output registers to drive a single or double-data rate (DDR) output path, two input registers and a latch to support a single or DDR input path, and two output enable registers to enhance clock-to-output enable timing or for DDR SDRAM interfacing. A new synchronous reset signal is available to each of the three sets of registers for preset or clear, or neither. In addition to the advanced IOE architecture, the Stratix and Stratix GX IOE features dedicated circuitry for external RAM interfacing, new I/O standards, differential on-chip termination, and high-speed differential I/O standard support.

External RAM Interfacing

The advanced Stratix and Stratix GX IOE architecture includes dedicated circuitry to interface with external RAM. This circuitry provides enhanced support for external high-speed memory devices such as DDR SDRAM and FCRAM. The DDR SDRAM interface uses a bidirectional signal, DQS, to clock data, DQ, at both the transmitting and receiving device. Stratix and Stratix GX devices transmit the DQS signal with the DQ data signals to minimize clock to data skew.

Stratix and Stratix GX devices include groups of programmable DQS and DQ pins, in the top and bottom I/O banks of the device. Each group consists of a DQS pin that supports a fixed number of DQ pins. The number of DQ pins depends on the DQ bus mode. When using the external RAM interfacing circuitry, the DQS pin drives a dedicated clock network that feeds the DQ pins residing in that bank. The Stratix and Stratix GX IOE has programmable delay chains that can phase shift the DQS signal by 90° or 72° to ensure data is sampled at the appropriate point in time. Therefore, the Stratix and Stratix GX devices make full use of the IOEs, and remove the need to build the input data path in the logic array. You can make these I/O assignments in the Quartus II Assignment Organizer.

f For more information on external RAM interfacing, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* or the *Stratix GX Device Family Data Sheet* in the *Stratix GX Device Family Handbook, Volume 1*.

I/O Standard Support

The Stratix and Stratix GX devices support all of the I/O standards that APEX II and APEX 20K devices support, including high-speed differential I/O standards such as LVDS, LVPECL, PCML, and HyperTransport™ technology, differential HSTL on input and output clocks, and differential SSTL on output clocks. Stratix and Stratix GX devices also introduce support for SSTL-18 Class I & II. Similar to APEX II devices, Stratix and Stratix GX devices only support certain I/O standards in designated I/O banks. In addition, `vref` pins are dedicated pins in Stratix and Stratix GX devices and now support up to 40 input pins.

f For more information about I/O standard support in Stratix and Stratix GX devices, see the *Selectable I/O Standards in Stratix & Stratix GX Devices* chapter.

High-Speed Differential I/O Standards

Stratix and Stratix GX devices support high-speed differential interfaces at speeds up to 840 Mbps using high-speed PLLs that drive a dedicated clock network to the SERDES. Each fast PLL can drive up to 20 high-speed channels. Stratix and Stratix GX devices use enhanced PLLs and M512 RAM blocks to provide up to 420 Mbps performance for SERDES bypass clock interfacing. There is no restriction on the number of channels that can be clocked using this scenario.

Stratix and Stratix GX devices have a different number of differential channels than APEX II devices. Tables 10–9 and 10–10 highlight the number of differential channels supported in Stratix and Stratix GX devices.

Table 10–9. Number of Dedicated DIfferential Channels in Stratix Devices (Part 1 of 2) *Note (1)*

Device	Pin Count	Number of Receiver Channels	Number of Transmitter Channels
EP1S10	672	36	36
	780	44	44

Table 10–9. Number of Dedicated DIfferential Channels in Stratix Devices (Part 2 of 2) *Note (1)*

Device	Pin Count	Number of Receiver Channels	Number of Transmitter Channels
EP1S20	672	50	48
	780	66	66
EP1S25	672	58	56
	780	66	70
	1,020	78	78
EP1S30	780	66	70
	956	80	80
	1,020	80	80
		2	2
EP1S40	956	80	80
	1,020	80	80
		10	10
	1,508	80	80
		10	10
EP1S60	956	80	80
	1,020	80	80
		10	12
	1,508	80	80
		36	36
EP1S80	956	80	80
		0	40
	1,508	80	80
		56	72

Note to Table 10–9:

(1) For information on channel speeds, see the *Stratix Device Family Data Sheet* section of the *Stratix Device Handbook, Volume 1* and the *High-Speed Differential I/O Interfaces* chapter in the *Stratix Device Handbook, Volume 2*.

Table 10–10. Number of Dedicated DIfferential Channels in Stratix GX Devices *Note (1)*

Device	Pin Count	Number of Transceivers	Number of Source-Synchronous Channels
EP1SGX10 C	672	4	22
EP1SGX10 D	672	8	22
EP1SGX25 C	672	4	39
EP1SGX25 D	672/1,020	8	39
EP1SGX25 F	1,020	16	39
EP1SGX40 D	1,020	8	45
EP1SGX40 G	1,020	20	45

Note to Table 10–10:

(1) For information on channel speeds, see the *Stratix GX Device Family Data Sheet* section of the *Stratix GX Device Handbook, Volume 1* and the *High-Speed Source-Synchronous Differential I/O Interfaces in Stratix GX Devices* chapter of the *Stratix GX Device Handbook, Volume 2*.

The differential I/O within Stratix GX also provides dynamic phase alignment (DPA). DPA enables the differential I/O to operate up to 1 Gbps per channel. DPA automatically and continuously tracks fluctuations caused by system variations and self-adjusts to eliminate the phase skew between the multiplied clock and the serial data. The block contains a dynamic phase selector for phase detection and selection, a SERDES, a synchronizer, and a data realigner circuit. You can bypass the dynamic phase aligner without affecting the basic source-synchronous operation of the channel by using a separate deserializer.

If you compile an APEX II LVDS design that uses clock-data synchronization (CDS) for a Stratix or Stratix GX device, the Quartus II software issues a warning during compilation that Stratix and Stratix GX devices do not support CDS.

Stratix and Stratix GX devices offer a flexible solution using new byte realignment circuitry to correct for byte misalignment by shifting, or slipping, data bits. Stratix and Stratix GX devices activate the byte realignment circuitry when an external pin (`rx_data_align`) or an internal custom-made state machine asserts the `SYNC` node high.

APEX II, APEX 20KE, and APEX 20KCdevices have a dedicated transmitter clock output pin (`LVDSTXOUTCLK`). In Stratix and Stratix GX devices, a transmitter `dataout` channel with an LVDS clock (fast clock) generates the transmitter clock output. Therefore, you can drive any

channel as an output clock to an I/O pin, not just dedicated clock output pins. This solution offers better versatility to address various applications that require more complex clocking schemes.

f For more information on differential I/O support, data realignment, and the transmitter clock output in Stratix and Stratix GX devices, see the *High-Speed Differential I/O Interfaces in Stratix Devices* chapter.

altlvds Megafunction

To take full advantage of the high-speed differential I/O standards available in Stratix and Stratix GX devices, you should update each instance of the `altlvds` megafunction in APEX II, APEX 20KE, and APEX 20KC designs. In the MegaWizard Plug-In Manager, choose the `altlvds` megafunction, select Stratix or Stratix GX as the target device family, update the megafunction, and recompile your design.

The `altlvds` megafunction supports new Stratix and Stratix GX parameters that are not available for APEX II, APEX 20KE, and APEX 20KC devices. Tables 10–11 and 10–12 describe the new parameters for the LVDS receiver and LVDS transmitter, respectively.

Table 10–11. New altlvds Parameters for Stratix LVDS Receiver *Note (1)*

Parameter	Function
`input_data_rate` *(2)*	Specifies the data rate in Mbps. This parameter replaces the multiplication factor *W*.
`inclock_data_alignment`	Indicates the alignment of `rx_inclk` and `rx_in` data.
`rx_data_align`	Drives the data alignment port of the fast PLL and enables byte realignment circuitry.
`registered_data_align_input`	Registers the `rx_data_align` input port to be clocked by `rx_outclock`.
`common_rx_tx_pll` *(3)*	Indicates the fast PLL can be shared between receiver and transmitter applications.

Table 10–12. New altlvds Parameters for Stratix LVDS Transmitter (Part 1 of 2) *Note (1)*

Parameter	Function
`output_data_rate` *(2)*	Specifies the data rate in Mbps. This parameter replaces the multiplication factor *W*.
`inclock_data_alignment`	Indicates the alignment of `tx_inclk` and `tx_in` data.
`outclock_alignment`	Specifies the alignment of `tx_outclock` and `tx_out` data.

Table 10–12. New altlvds Parameters for Stratix LVDS Transmitter (Part 2 of 2) *Note (1)*	
Parameter	**Function**
`registered_input`	Specifies the clock source for the input synchronization registers, which can be either `tx_inclock` or `tx_coreclock`. Used only when the **Registered Inputs** option is selected.
`common_rx_tx_pll` *(3)*	Indicates the fast PLL can be shared between receiver and transmitter applications.

Notes to Tables 10–11 and 10–12:

(1) You can specify these parameters in the MegaWizard Plug-In Manager.

(2) You must specify a data rate in the MegaWizard Plug-In Manager instead of a *W* factor.

(3) The same fast PLL can be used to clock both the receiver and transmitter only if both are running at the same frequency.

Above the standard I/O offered by APEX II, APEX 20K, and Stratix devices, Stratix GX devices provide up to 20 3.175 Gbps transceivers. The transceivers provide high-speed serial links for chip-to-chip, backplane, and line-side connectivity and support a number of the emerging high-speed protocols. You can find more information in the *Stratix GX Family Data Sheet* in the *Stratix GX Family Handbook, Volume 1*.

Configuration

The Stratix and Stratix GX devices supports all current configuration schemes, including the use of enhanced configuration devices, passive serial (PS), passive parallel asynchronous (PPA), fast passive parallel (FPP), and JTAG. Stratix and Stratix GX devices also provide a number of new configuration enhancements that you can take advantage of when migrating APEX II and APEX 20K designs to Stratix and Stratix GX devices.

Configuration Speed & Schemes

You can configure Stratix and Stratix GX devices at a maximum clock speed of 100 MHz, which is faster than the 66-MHz and 33-MHz maximum configuration speeds for APEX II and APEX 20K devices, respectively. Similar to APEX II devices, you can use 8-bit parallel data to configure Stratix and Stratix GX devices (the target device can receive byte-wide configuration data on each clock cycle) significantly speeding up configuration times.

You can select a configuration scheme based on how the `MSEL` pins are driven. Stratix and Stratix GX devices have three `MSEL` pins (APEX II and APEX 20K devices have two `MSEL` pins) for determining the configuration scheme.

f For more information about Stratix and Stratix GX configuration schemes, see the *Configuring Stratix & Stratix GX Devices* chapter.

Remote Update Configuration

The APEX 20K device family introduced the concept of remote update configuration, where you could send the APEX 20K device new configuration files from a remote source and the device would store the files in flash memory and reconfigure itself with the new configuration data. The Stratix and Stratix GX devices enhance support for remote update configuration with new, dedicated circuitry to handle and recover from errors. If an error occurs either during device configuration or in user mode, this new circuitry reconfigures the Stratix or Stratix GX device to a known state. Additionally, the Stratix and Stratix GX devices have a user watchdog timer to ensure the application configuration data executes successfully during user mode. User logic must continually reset this watchdog timer in order to validate that the application configuration data is functioning properly.

f For more information about how to use the remote and local update modes, see the *Remote System Configuration with Stratix & Stratix GX Devices* chapter.

JTAG Instruction Support

Stratix and Stratix GX devices support two new JTAG instructions, `PULSE_NCONFIG` and `CONFIG_IO`. The `PULSE_NCONFIG` instruction emulates pulsing the `nCONFIG` signal low to trigger reconfiguration, while the actual `nCONFIG` pin on the device is unaffected. The `CONFIG_IO` instruction allows you to use the JTAG chain to configure I/O standards for all pins. Because this instruction interrupts device configuration, you should reconfigure the Stratix or Stratix GX device after you finish JTAG testing to ensure proper device operation. Table 10–13 compares JTAG instruction support in Stratix and Stratix GX devices versus APEX II and APEX 20K devices. For further information about the supported JTAG instructions, see the appropriate device family data sheet.

Table 10–13. JTAG Instruction Support (Part 1 of 2)

JTAG Instruction	Stratix	APEX II	APEX 20K
SAMPLE/PRELOAD	v	v	v
EXTEST	v	v	v
BYPASS	v	v	v
USERCODE	v	v	v

Table 10–13. JTAG Instruction Support (Part 2 of 2)

JTAG Instruction	Stratix	APEX II	APEX 20K
IDCODE	v	v	v
ICR Instructions	v	v	v
SignalTap™ II Instructions	v	v	v
HIGHZ	v	v	
CLAMP	v	v	
PULSE_NCONFIG	v		
CONFIG_IO	v		

Conclusion

The Stratix and Stratix GX devices extend the advanced features available in the APEX II and APEX 20K device families to deliver a complete system-on-a-programmable-chip (SOPC) solution. By following these guidelines, you can easily transition current APEX II and APEX 20K designs to take advantage of the new features available in Stratix and Stratix GX devices.

Section VI. System Configuration & Upgrades

This section describes configuration and remote system upgrade. This section also provides configuration information for all of the supported configuration schemes for Stratix® devices. These configuration schemes use either a microprocessor, configuration device, or download cable. There is detailed information on how to design with Altera® enhanced configuration devices which includes information on how to manage multiple configuration files and access the on-chip FLASH memory space. The last chapter shows you how to perform remote and local upgrades for your designs.

This section contains the following chapters:

- Chapter 11, Configuring Stratix & Stratix GX Devices
- Chapter 12, Remote System Configuration with Stratix & Stratix GX Devices

f For information on Altera enhanced configuration devices, see the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet* chapter in the *Configuration Handbook, Volume 2*.

Revision History

The table below shows the revision history for Chapters 11 through 12.

Chapter	Date/Version	Changes Made
11	July 2005, v3.2	• Updated "PORSEL Pins" and "nIO_PULLUP Pins" sections. • Updated "FPP Configuration Using an Enhanced Configuration Device" section. • Updated "PPA Configuration" section.
	September 2004, v3.1	• Corrected spelling error.
	April 2004, v3.0	• In the "PORSEL Pins" section and the "nIO_PULLUP Pins" section, several pull-down resistors were changed to pull-up resistors. • Updated notes in Figure 11–3. • Two vertical V_{CC} lines removed in Figures 11–12 to 11–14. • Three paragraphs added regarding the `CONF_DONE` and `INIT_DONE` pins on page 13-18. • Value in Note 1 changed in Tables 11–8 and 11–9. • Deleted reference to AS in Table 11–15 because Stratix does not support AS mode. • Text added before callout of Figure 11–7.
	July 2003, v2.0	• Updated Remote/local update PPA typical use description on page 11-1. • Updated VCCSEL Pins section on page 11-3. • Updated figures to use 10k resistors throughout for configuration control signals. • Updated text on page 11-23 to tell how to connect a microprocessor to nSTATUS. • Figure 11–19, Note 3. • Updated Table 11–12. • Added Note 6 to Figure 11–21 and the text below the figure describing the `nCE` pin. • Updated definitions for CLKUSR, and JTAG pins in Table 11–16.
12	September 2004, v3.1	• Editorial corrections.
	April 2004, v3.0	• The input file in Figure 12–22 was changed to remote_update_initial_pgm.pdf. • Title in Figure 12–23 was changed from **Local**... to **Remote** Update Partial Programming File Generation. • Rearranged the "Quartus II Software Support" section.
	July 2003, v2.0	• Added altremote_update Megafunction section on pages 12-18 to 12-21.

11. Configuring Stratix & Stratix GX Devices

S52013-3.2

Introduction

You can configure Stratix® and Stratix GX devices using one of several configuration schemes. All configuration schemes use either a microprocessor, configuration device, or a download cable. See Table 11–1.

Table 11–1. Stratix & Stratix GX Device Configuration Schemes

Configuration Scheme	Typical Use
Fast passive parallel (FPP)	Configuration with a parallel synchronous configuration device or microprocessor interface where eight bits of configuration data are loaded on every clock cycle.
Passive serial (PS)	Configuration with a serial synchronous microprocessor interface or the MasterBlaster™ communications cable, USB Blaster, ByteBlaster™ II, or ByteBlasterMV parallel port download cable.
Passive parallel asynchronous (PPA)	Configuration with a parallel asynchronous microprocessor interface. In this scheme, the microprocessor treats the target device as memory.
Remote/local update FPP	Configuration using a Nios™ (16-bit ISA) and Nios® II (32-bit ISA) or other embedded processor. Allows you to update the Stratix or Stratix GX device configuration remotely using the FPP scheme to load data.
Remote/local update PS	Passive serial synchronous configuration using a Nios or other embedded processor. Allows you to update the Stratix or Stratix GX device configuration remotely using the PS scheme to load data.
Remote/local update PPA	Passive parallel asynchronous configuration using a Nios or other embedded processor. In this scheme, the Nios microprocessor treats the target device as memory. Allows you to update the Stratix or Stratix GX device configuration remotely using the PPA scheme to load data.
Joint Test Action Group (JTAG)	Configuration through the IEEE Std. 1149.1 JTAG pins. You can perform JTAG configuration with either a download cable or an embedded device. Ability to use SignalTap® II Embedded Logic Analyzer.

This chapter discusses how to configure one or more Stratix or Stratix GX devices. It should be used together with the following documents:

- *MasterBlaster Serial/USB Communications Cable Data Sheet*
- *USB Blaster USB Port Download Cable Development Tools Data Sheet*
- *ByteBlaster II Parallel Port Download Cable Data Sheet*
- *ByteBlasterMV Parallel Port Download Cable Data Sheets*
- *Configuration Devices for SRAM-Based LUT Devices Data Sheet*
- *Enhanced Configuration Devices (EPC4, EPC8, & EPC16) Data Sheet*

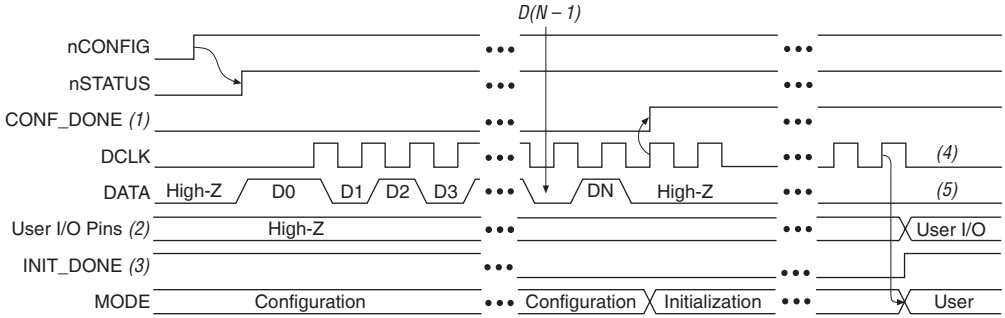
- The *Remote System Configuration with Stratix & Stratix GX Devices* chapter

f For more information on setting device configuration options or generating configuration files, see the *Software Setting* chapter in *Volume 2* of the *Configuration Handbook*.

Device Configuration Overview

During device operation, the FPGA stores configuration data in SRAM cells. Because SRAM memory is volatile, you must load the SRAM cells with the configuration data each time the device powers up. After configuration, the device must initialize its registers and I/O pins. After initialization, the device enters user mode. Figure 11–1 shows the state of the device during the configuration, initialization, and user mode.

Figure 11–1. Stratix & Stratix GX Configuration Cycle



Notes to Figure 11–1:

(1) During initial power up and configuration, `CONF_DONE` is low. After configuration, `CONF_DONE` goes high. If the device is reconfigured, `CONF_DONE` goes low after `nCONFIG` is driven low.

(2) User I/O pins are tri-stated during configuration. Stratix and Stratix GX devices also have a weak pull-up resistor on I/O pins during configuration that are enabled by `nIO_PULLUP`. After initialization, the user I/O pins perform the function assigned in the user's design.

(3) If the `INIT_DONE` pin is used, it will be high because of an external 10 kΩ resistor pull-up when `nCONFIG` is low and during the beginning of configuration. Once the option bit to enable `INIT_DONE` is programmed into the device (during the first frame of configuration data), the `INIT_DONE` pin will go low.

(4) `DCLK` should not be left floating. It should be driven high or low.

(5) `DATA0` should not be left floating. It should be driven high or low.

You can load the configuration data for the Stratix or Stratix GX device using a passive configuration scheme. When using any passive configuration scheme, the Stratix or Stratix GX device is incorporated into a system with an intelligent host, such as a microprocessor, that controls the configuration process. The host supplies configuration data from a storage device (e.g., a hard disk, RAM, or other system memory). When using passive configuration, you can change the target device's

functionality while the system is in operation by reconfiguring the device. You can also perform in-field upgrades by distributing a new programming file to system users.

The following sections describe the MSEL[2..0], VCCSEL, PORSEL, and nIO_PULLUP pins used in Stratix and Stratix GX device configuration.

MSEL[2..0] Pins

You can select a Stratix or Stratix GX device configuration scheme by driving its MSEL2, MSEL1, and MSEL0 pins either high or low, as shown in Table 11–2.

Table 11–2. Stratix & Stratix GX Device Configuration Schemes

Description	MSEL2	MSEL1	MSEL0
FPP configuration	0	0	0
PPA configuration	0	0	1
PS configuration	0	1	0
Remote/local update FPP *(1)*	1	0	0
Remote/local update PPA *(1)*	1	0	1
Remote/local update PS *(1)*	1	1	0
JTAG-based configuration *(3)*	*(2)*	*(2)*	*(2)*

Notes to Table 11–2:

(1) These schemes require that you drive a secondary pin RUnLU to specify whether to perform a remote update or local update.

(2) Do not leave MSEL pins floating. Connect them to V_{CCIO} or GND. These pins support the non-JTAG configuration scheme used in production. If only JTAG configuration is used you should connect the MSEL pins to ground.

(3) JTAG-based configuration takes precedence over other configuration schemes, which means the MSEL pins are ignored.

The MSEL[] pins can be tied to V_{CCIO} of the I/O bank they reside in or ground.

V_{CCSEL} Pins

You can configure Stratix and Stratix GX devices using the 3.3-, 2.5-, 1.8-, or 1.5-V LVTTL I/O standard on configuration and JTAG input pins. VCCSEL is a dedicated input on Stratix and Stratix GX devices that selects between 3.3-V/2.5-V input buffers and 1.8-V/1.5-V input buffers for dedicated configuration input pins. A logic low supports 3.3-V/2.5-V signaling, and a logic high supports 1.8-V/1.5-V signaling. A logic high can also support 3.3-V/2.5-V signaling. VCCSEL affects the configuration

related I/O banks (3, 4, 7, and 8) where the following pins reside: TDI, TMS, TCK, TRST, MSEL0, MSEL1, MSEL2, nCONFIG, nCE, DCLK, PLL_ENA, CONF_DONE, nSTATUS. The VCCSEL pin can be pulled to 1.5, 1.8, 2.5, or 3.3-V for a logic high level. There is an internal 2.5-kΩ pull-down resistor on VCCSEL. Therefore, if you are using a pull-up resister to pull up this signal, you need to use a 1-kΩ resistor.

VCCSEL also sets the power-on-reset (POR) trip point for all the configuration related I/O banks (3, 4, 7, and 8), ensuring that these I/O banks have powered up to the appropriate voltage levels before configuration begins. Upon power-up, the FPGA does not release nSTATUS until V_{CCINT} and all of the V_{CCIO}s of the configuration I/O banks are above their POR trip points. If you set VCCSEL to ground (logic low), this sets the POR trip point for all configuration I/O banks to a voltage consistent with 3.3-V/2.5-V signaling. When VCCSEL = 0, the POR trip point for these I/O banks may be as high as 1.8 V. If V_{CCIO} of any of the configuration banks is set to 1.8 or 1.5 V, the voltage supplied to this I/O bank(s) may never reach the POR trip point, which will not allow the FPGA to begin configuration.

1 If the V_{CCIO} of I/O banks 3, 4, 7, or 8 is set to 1.5 or 1.8 V and the configuration signals used require 3.3-V or 2.5-V signaling you should set VCCSEL to V_{CC} (logic high) in order to lower the POR trip point to enable successful configuration.

Table 11–3 shows how you should set the VCCSEL depending on the V_{CCIO} setting of the configuration I/O banks and your configuration input signaling voltages.

Table 11–3. VCCSEL Setting

V_{CCIO} (banks 3,4,7,8)	Configuration Input Signaling Voltage	V_{CCSEL}
3.3-V/2.5-V	3.3-V/2.5-V	GND
1.8-V/1.5-V	3.3-V/2.5-V/1.8-V/1.5-V	VCC
3.3-V/2.5-V	1.8-V/1.5-V	Not Supported

The VCCSEL signal does not control any of the dual-purpose pins, including the dual-purpose configuration pins, such as the DATA[7..0] and PPA pins (nWS, nRS, CS, nCS, and RDYnBSY). During configuration, these dual-purpose pins drive out voltage levels corresponding to the V_{CCIO} supply voltage that powers the I/O bank containing the pin. After configuration, the dual-purpose pins inherit the I/O standards specified in the design.

PORSEL Pins

PORSEL is a dedicated input pin used to select POR delay times of 2 ms or 100 ms during power-up. When the PORSEL pin is connected to ground, the POR time is 100 ms; when the PORSEL pin is connected to VCC, the POR time is 2 ms. There is an internal 2.5-kΩ pull-down resistor on PORSEL. Therefore if you are using a pull-up resistor to pull up this signal, you need to use a 1-kΩ resistor.

When using enhanced configuration devices to configure Stratix devices, make sure that the PORSEL setting of the Stratix device is the same or faster than the PORSEL setting of the enhanced configuration device. If the FPGA is not powered up after the enhanced configuration device exits POR, the CONF_DONE signal will be high since the pull-up resistor is pulling this signal high. When the enhanced configuration device exits POR, OE of the enhanced configuration device is released and pulled high by a pull-up resistor. Since the enhanced configuration device sees its nCS / CONF_DONE signal also high, it enters a test mode. Therefore, you must ensure the FPGA powers up before the enhanced configuration device exits POR.

For more margin, the 100-ms setting can be selected when using an enhanced configuration device to allow the Stratix FPGA to power-up before configuration is attempted (see Table 11–4).

Table 11–4. PORSEL Settings

PORSEL Settings	POR Time (ms)
GND	100
V_{CC}	2

nIO_PULLUP Pins

The nIO_PULLUP pin enables a built-in weak pull-up resistor to pull all user I / O pins to VCCIO before and during device configuration. If nIO_PULLUP is connected to V_{CC} during configuration, the weak pull-ups on all user I / O pins and all dual-purpose pins are disabled. If connected to ground, the pull-ups are enabled during configuration. The nIO_PULLUP pin can be pulled to 1.5, 1.8, 2.5, or 3.3-V for a logic level high. There is an internal 2.5-kΩ pull-down resistor on nIO_PULLUP. Therefore, if you are using a pull-up resistor to pull up this signal, you need to use a 1-kΩ resistor.

TDO & nCEO Pins

TDO and nCEO pins drive out the same voltage levels as the V_{CCIO} that powers the I/O bank where the pin resides. You must select the V_{CCIO} supply for the bank containing TDO accordingly. For example, when using the ByteBlasterMV cable, the V_{CCIO} for the bank containing TDO must be powered up at 3.3-V. The current strength for TDO is 12 mA.

Configuration File Size

Tables 11–5 and 11–6 summarize the approximate configuration file size required for each Stratix and Stratix GX device. To calculate the amount of storage space required for multi-device configurations, add the file size of each device together.

Table 11–5. Stratix Configuration File Sizes

Device	Raw Binary File (.rbf) Size (Bits)
EP1S10	3,534,640
EP1S20	5,904,832
EP1S25	7,894,144
EP1S30	10,379,368
EP1S40	12,389,632
EP1S60	17,543,968
EP1S80	23,834,032

Table 11–6. Stratix GX Configuration File Sizes

Device	Raw Binary File Size (Bits)
EP1SGX10C	3,579,928
EP1SGX10D	3,579,928
EP1SGX25C	7,951,248
EP1SGX25D	7,951,248
EP1SGX25F	7,951,248
EP1SGX40D	12,531,440
EP1SGX40G	12,531,440

You should only use the numbers in Tables 11–5 and 11–6 to estimate the file size before design compilation. The exact file size may vary because different Altera® Quartus® II software versions may add a slightly

different number of padding bits during programming. However, for any specific version of the Quartus II software, any design targeted for the same device has the same configuration file size.

Altera Configuration Devices

The Altera enhanced configuration devices (EPC16, EPC8, and EPC4 devices) support a single-device configuration solution for high-density FPGAs and can be used in the FPP and PS configuration schemes. They are ISP-capable through its JTAG interface. The enhanced configuration devices are divided into two major blocks, the controller and the flash memory.

f For information on enhanced configuration devices, see the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet* and the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook*.

The EPC2 and EPC1 configuration devices provide configuration support for the PS configuration scheme. The EPC2 device is ISP-capable through its JTAG interface. The EPC2 and EPC1 can be cascaded to hold large configuration files.

f For more information on EPC2, EPC1, and EPC1441 configuration devices, see the *Configuration Devices for SRAM-Based LUT Devices Data Sheet*.

Configuration Schemes

This section describes how to configure Stratix and Stratix GX devices with the following configuration schemes:

- PS Configuration with Configuration Devices
- PS Configuration with a Download Cable
- PS Configuration with a Microprocessor
- FPP Configuration
- PPA Configuration
- JTAG Programming & Configuration
- JTAG Programming & Configuration of Multiple Devices

PS Configuration

PS configuration of Stratix and Stratix GX devices can be performed using an intelligent host, such as a MAX® device, microprocessor with flash memory, an Altera configuration device, or a download cable. In the PS scheme, an external host (MAX device, embedded processor, configuration device, or host PC) controls configuration. Configuration data is clocked into the target Stratix devices via the `DATA0` pin at each rising edge of `DCLK`.

PS Configuration with Configuration Devices

The configuration device scheme uses an Altera configuration device to supply data to the Stratix or Stratix GX device in a serial bitstream (see Figure 11–3).

In the configuration device scheme, nCONFIG is usually tied to V_{CC} (when using EPC16, EPC8, EPC4, or EPC2 devices, nCONFIG may be connected to nINIT_CONF). Upon device power-up, the target Stratix or Stratix GX device senses the low-to-high transition on nCONFIG and initiates configuration. The target device then drives the open-drain CONF_DONE pin low, which in-turn drives the configuration device's nCS pin low. When exiting power-on reset (POR), both the target and configuration device release the open-drain nSTATUS pin.

Before configuration begins, the configuration device goes through a POR delay of up to 200 ms to allow the power supply to stabilize (power the Stratix or Stratix GX device before or during the POR time of the configuration device). This POR delay has a maximum of 200 ms for EPC2 devices. For enhanced configuration devices, you can select between 2 ms and 100 ms by connecting PORSEL pin to VCC or GND, accordingly. During this time, the configuration device drives its OE pin low. This low signal delays configuration because the OE pin is connected to the target device's nSTATUS pin. When the target and configuration devices complete POR, they release nSTATUS, which is then pulled high by a pull-up resistor.

When configuring multiple devices, configuration does not begin until all devices release their OE or nSTATUS pins. When all devices are ready, the configuration device clocks data out serially to the target devices using an internal oscillator.

After successful configuration, the Stratix FPGA starts initialization using the 10-MHz internal oscillator as the reference clock. After initialization, this internal oscillator is turned off. The CONF_DONE pin is released by the target device and then pulled high by a pull-up resistor. When initialization is complete, the FPGA enters user mode. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize.

If an error occurs during configuration, the target device drives its nSTATUS pin low, resetting itself internally and resetting the configuration device. If the **Auto-Restart Configuration on Frame Error** option—available in the Quartus II **Global Device Options** dialog box (Assign menu)—is turned on, the device reconfigures automatically if an error occurs. To find this option, choose **Compiler Settings** (Processing menu), then click on the **Chips & Devices** tab.

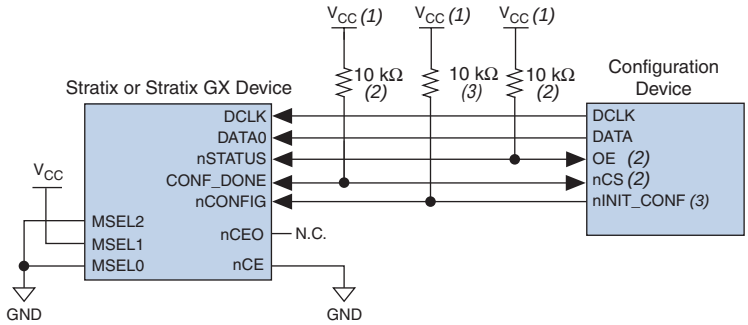
If this option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low to restart configuration. The external system can pulse nCONFIG if it is under system control rather than tied to V_{CC}. When configuration is complete, the target device releases CONF_DONE, which disables the configuration device by driving nCS high. The configuration device drives DCLK low before and after configuration.

In addition, if the configuration device sends all of its data and then detects that CONF_DONE has not gone high, it recognizes that the target device has not configured successfully. In this case, the configuration device pulses its OE pin low for a few microseconds, driving the target device's nSTATUS pin low. If the **Auto-Restart Configuration on Frame Error** option is set in the software, the target device resets and then pulses its nSTATUS pin low. When nSTATUS returns high, the configuration device reconfigures the target device. When configuration is complete, the configuration device drives DCLK low.

Do not pull CONF_DONE low to delay initialization. Instead, use the Quartus II software's **Enable User-Supplied Start-Up Clock (CLKUSR)** option to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain initialize together. When CONF_DONE is driven low after device configuration, the configuration device recognizes that the target device has not configured successfully.

Figure 11–2 shows how to configure one Stratix or Stratix GX device with one configuration device.

Figure 11–2. Single Device Configuration Circuit



Notes to Figure 11–2:

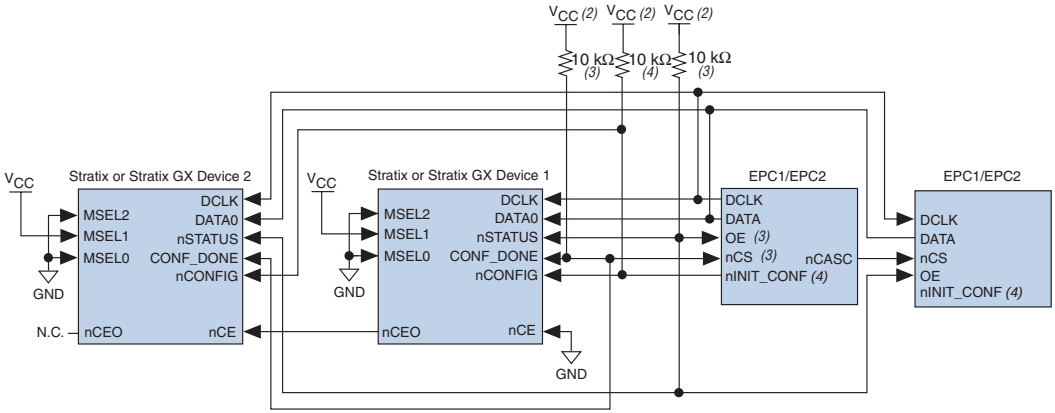
(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The enhanced configuration devices and EPC2 devices have internal programmable pull-ups on `OE` and `nCS`. You should only use the internal pull-ups of the configuration device if the `nSTATUS` and `CONF_DONE` signals are pulled up to 3.3 V or 2.5 V (not 1.8 V or 1.5 V). If external pull-ups are used, they should be 10 kΩ

(3) The `nINIT_CONF` pin is available on EPC16, EPC8, EPC4, and EPC2 devices. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} through a resistor. he `nINIT_CONF` pin has an internal pull-up resistor that is always active in EPC16, EPC8, EPC4, and EPC2 devices. These devices do not need an external pull-up resistor on the `nINIT_CONF` pin.

Figure 11–3 shows how to configure multiple Stratix and Stratix GX devices with multiple EPC2 or EPC1 configuration devices.

Figure 11–3. Multi-Device Configuration Circuit *Note (1)*



Notes to Figure 11–3:

(1) When performing multi-device active serial configuration, you must generate the configuration device programmer object file (**.pof**) from each project's SOF. You can combine multiple SOFs using the Quartus II software through the **Device & Pin Option** dialog box. For more information on how to create configuration and programming files, see the *Software Settings* section in the *Configuration Handbook, Volume 2*.

(2) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(3) The enhanced configuration devices and EPC2 devices have internal programmable pull-ups on `OE` and `nCS`. You should only use the internal pull-ups of the configuration device if the `nSTATUS` and `CONF_DONE` signals are pulled up to 3.3 V or 2.5 V (not 1.8 V or 1.5 V). If external pull-ups are used, they should be 10 kΩ

(4) The `nINIT_CONF` pin is available on EPC16, EPC8, EPC4, and EPC2 devices. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} through a resistor. The `nINIT_CONF` pin has an internal pull-up resistor that is always active in EPC16, EPC8, EPC4, and EPC2 devices. These devices do not need an external pull-up resistor on the `nINIT_CONF` pin.

After the first Stratix or Stratix GX device completes configuration during multi-device configuration, its `nCEO` pin activates the second device's `nCE` pin, prompting the second device to begin configuration. Because all device `CONF_DONE` pins are tied together, all devices initialize and enter user mode at the same time.

In addition, all `nSTATUS` pins are tied together; thus, if any device (including the configuration devices) detects an error, configuration stops for the entire chain. Also, if the first configuration device does not detect `CONF_DONE` going high at the end of configuration, it resets the chain by pulsing its `OE` pin low for a few microseconds. This low pulse drives the `OE` pin low on the second configuration device and drives `nSTATUS` low on all Stratix and Stratix GX devices, causing them to enter an error state.

If the **Auto-Restart Configuration on Frame Error** option is turned on in the software, the Stratix or Stratix GX device releases its `nSTATUS` pins after a reset time-out period. When the `nSTATUS` pins are released and pulled high, the configuration devices reconfigure the chain. If the **Auto-**

Restart Configuration on Frame Error option is not turned on, the Stratix or Stratix GX devices drive nSTATUS low until they are reset with a low pulse on nCONFIG.

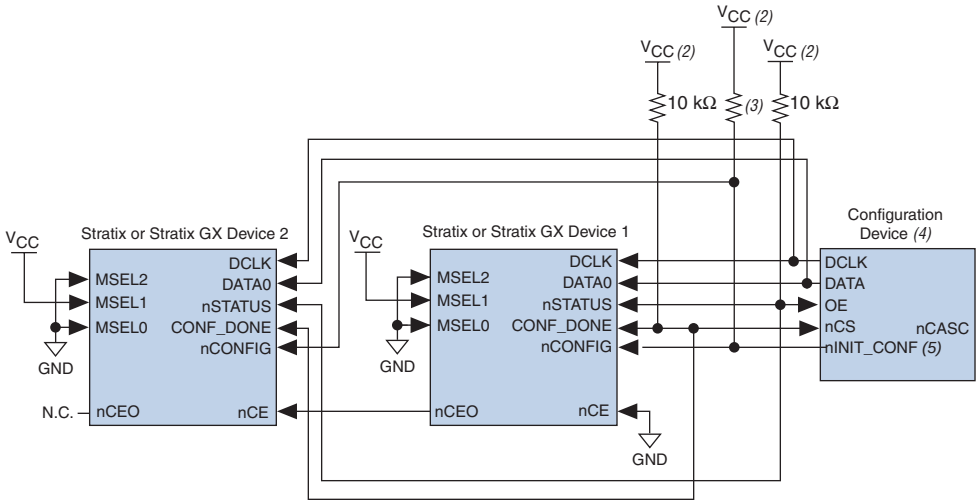
You can also cascade several EPC2/EPC1 configuration devices to configure multiple Stratix and Stratix GX devices. When all data from the first configuration device is sent, it drives nCASC low, which in turn drives nCS on the subsequent configuration device. Because a configuration device requires less than one clock cycle to activate a subsequent configuration device, the data stream is uninterrupted.

1 You cannot cascade enhanced (EPC16, EPC8, and EPC4) configuration devices.

You can use a single configuration chain to configure multiple Stratix and Stratix GX devices. In this scheme, the nCEO pin of the first device is connected to the nCE pin of the second device in the chain. If there are additional devices, connect the nCE pin of the next device to the nCEO pin of the previous device. To configure properly, all of the device CONF_DONE and nSTATUS pins must be tied together.

Figure 11–4 shows an example of configuring multiple Stratix and Stratix GX devices using a configuration device.

Figure 11–4. Configuring Multiple Stratix & Stratix GX Devices with A Single Configuration Device *Note (1)*



Notes to Figure 11–4:

(1) When performing multi-device active serial configuration, you must generate the configuration device programmer object file (**.pof**) from each project's SOF. You can combine multiple SOFs using the Quartus II software through the **Device & Pin Option** dialog box. For more information on how to create configuration and programming files, see the *Software Settings* section in the *Configuration Handbook, Volume 2*.

(2) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(3) The enhanced configuration devices and EPC2 devices have internal programmable pull-ups on `OE` and `nCS`. You should only use the internal pull-ups of the configuration device if the `nSTATUS` and `CONF_DONE` signals are pulled up to 3.3 V or 2.5 V (not 1.8 V or 1.5 V). If external pull-ups are used, they should be 10 kΩ

(4) EPC16, EPC8, and EPC4 configuration devices cannot be cascaded.

(5) The `nINIT_CONF` pin is available on EPC16, EPC8, EPC4, and EPC2 devices. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} through a resistor. The `nINIT_CONF` pin has an internal pull-up resistor that is always active in EPC16, EPC8, EPC4, and EPC2 devices. These devices do not need an external pull-up resistor on the `nINIT_CONF` pin.

Table 11–7 shows the status of the device DATA pins during and after configuration.

Table 11–7. DATA Pin Status Before & After Configuration

Pins	Stratix or Stratix GX Device	
	During	After
DATA0 *(1)*	Used for configuration	User defined
DATA[7..1] *(2)*	Used in some configuration modes	User defined
I/O Pins	Tri-state	User defined

Notes to Table 11–7:

(1) The status shown is for configuration with a configuration device.

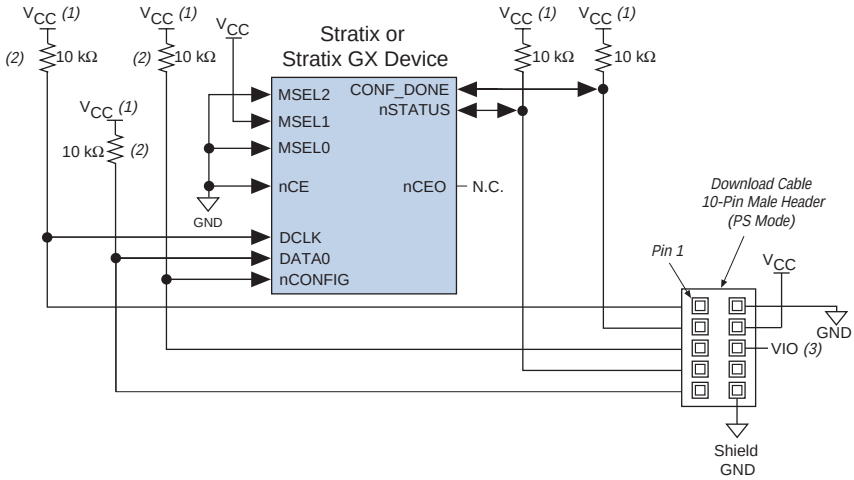
(2) The function of these pins depends upon the settings specified in the Quartus II software using the **Device & Pin Option** dialog box (see the *Software Settings* section in the *Configuration Handbook, Volume 2*, and the Quartus II Help software for more information).

PS Configuration with a Download Cable

In PS configuration with a download cable, an intelligent host transfers data from a storage device to the Stratix or Stratix GX device through the MasterBlaster, USB-Blaster, ByteBlaster II or ByteBlasterMV cable. To initiate configuration in this scheme, the download cable generates a low-to-high transition on the nCONFIG pin. The programming hardware then places the configuration data one bit at a time on the device's DATA0 pin. The data is clocked into the target device until CONF_DONE goes high. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize.

When using programming hardware for the Stratix or Stratix GX device, turning on the **Auto-Restart Configuration on Frame Error** option does not affect the configuration cycle because the Quartus II software must restart configuration when an error occurs. Additionally, the **Enable User-Supplied Start-Up Clock (CLKUSR)** option has no affect on the device initialization since this option is disabled in the SOF when programming the FPGA using the Quartus II software programmer and a download cable. Therefore, if you turn on the **CLKUSR** option, you do not need to provide a clock on CLKUSR when you are configuring the FPGA with the Quartus II programmer and a download cable.
Figure 11–5 shows PS configuration for the Stratix or Stratix GX device using a MasterBlaster, USB-Blaster, ByteBLaster II or ByteBlasterMV cable.

Figure 11–5. PS Configuration Circuit with a Download Cable



Notes to Figure 11–5:

(1) You should connect the pull-up resistor to the same supply voltage as the MasterBlaster (VIO pin) or ByteBlasterMV cable.

(2) The pull-up resistors on the DATA0 and DCLK pins are only needed if the download cable is the only configuration scheme used on the board. This is to ensure that the DATA0 and DCLK pins are not left floating after configuration. For example, if the design also uses a configuration device, the pull-up resistors on the DATA0 and DCLK pins are not necessary.

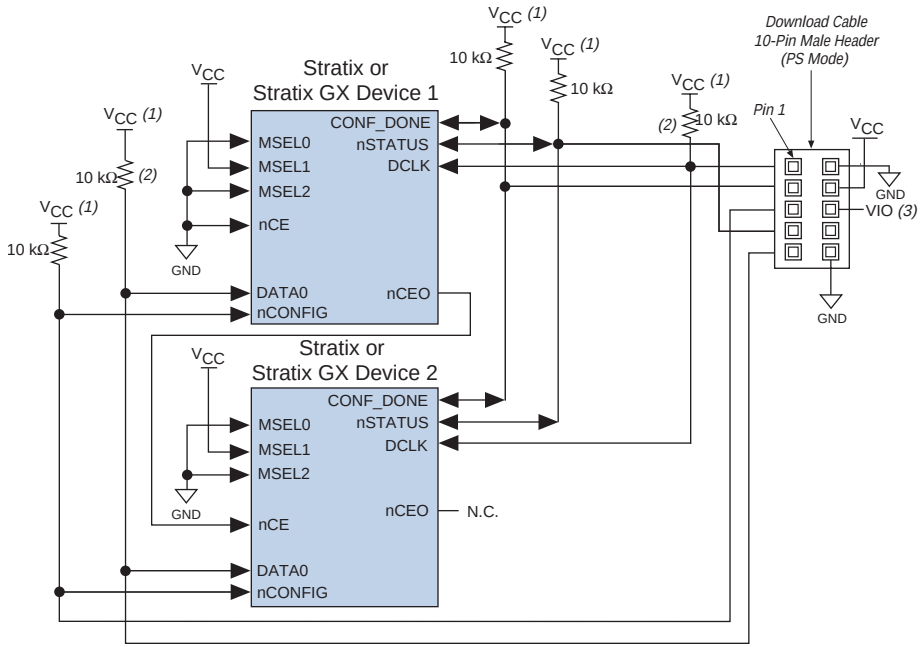
(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. This pin is a no-connect pin for the ByteBlasterMV header.

You can use programming hardware to configure multiple Stratix and Stratix GX devices by connecting each device's nCEO pin to the subsequent device's nCE pin. All other configuration pins are connected to each device in the chain.

Because all CONF_DONE pins are tied together, all devices in the chain initialize and enter user mode at the same time. In addition, because the nSTATUS pins are tied together, the entire chain halts configuration if any device detects an error. In this situation, the Quartus II software must restart configuration; the **Auto-Restart Configuration on Frame Error** option does not affect the configuration cycle.

Figure 11–6 shows how to configure multiple Stratix and Stratix GX devices with a MasterBlaster or ByteBlasterMV cable.

Figure 11–6. Multi-Device PS Configuration with a Download Cable



Notes to Figure 11–6:

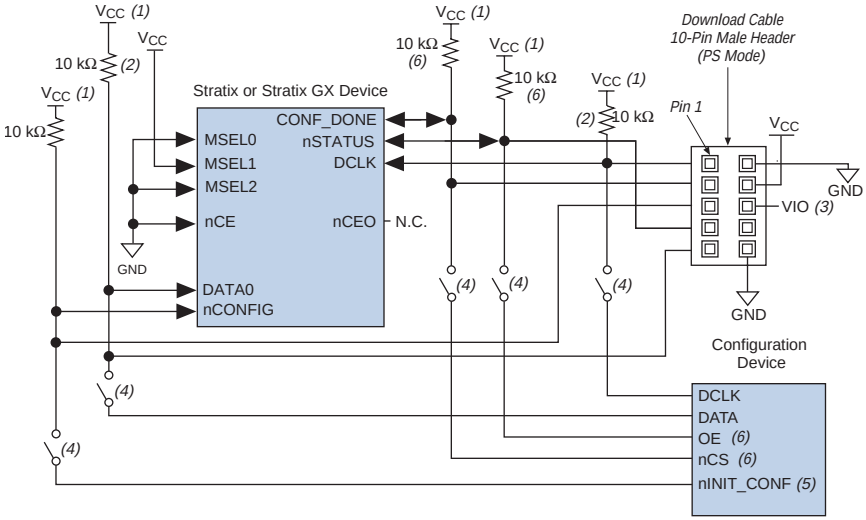
(1) You should connect the pull-up resistor to the same supply voltage as the MasterBlaster (`VIO` pin) or ByteBlasterMV cable.

(2) The pull-up resistors on the `DATA0` and `DCLK` pins are only needed if the download cable is the only configuration scheme used on the board. This is to ensure that the `DATA0` and `DCLK` pins are not left floating after configuration. For example, if the design also uses a configuration device, the pull-up resistors on the `DATA0` and `DCLK` pins are not necessary.

(3) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. See the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

If you are using a download cable to configure device(s) on a board that also has configuration devices, you should electrically isolate the configuration devices from the target device(s) and cable. One way to isolate the configuration devices is to add logic, such as a multiplexer, that can select between the configuration devices and the cable. The multiplexer device should allow bidirectional transfers on the `nSTATUS` and `CONF_DONE` signals. Another option is to add switches to the five common signals (`CONF_DONE`, `nSTATUS`, `DCLK`, `nCONFIG`, and `DATA0`) between the cable and the configuration devices. The last option is to remove the configuration devices from the board when configuring with the cable. Figure 11–7 shows a combination of a configuration device and a download cable to configure a Stratix or Stratix GX device.

Figure 11–7. Configuring with a Combined PS & Configuration Device Scheme



Notes to Figure 11–7:

(1) You should connect the pull-up resistor to the same supply voltage as the configuration device.

(2) The pull-up resistors on the `DATA0` and `DCLK` pins are only needed if the download cable is the only configuration scheme used on the board. This is to ensure that the `DATA0` and `DCLK` pins are not left floating after configuration. For example, if the design also uses a configuration device, the pull-up resistors on the `DATA0` and `DCLK` pins are not necessary.

(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} should match the target device's V_{CCIO}. This is a no-connect pin for the ByteBlasterMV header.

(4) You should not attempt configuration with a download cable while a configuration device is connected to a Stratix or Stratix GX device. Instead, you should either remove the configuration device from its socket when using the download cable or place a switch on the five common signals between the download cable and the configuration device. Remove the download cable when configuring with a configuration device.

(5) If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor.

(6) If external pull-ups are used on `CONF_DONE` and `nSTATUS` pins, they should always be 10 kΩ resistors. You can use the internal pull-ups of the configuration device only if the `CONF_DONE` and `nSTATUS` signals are pulled-up to 3.3 V or 2.5 V (not 1.8 V or 1.5 V).

f For more information on how to use the MasterBlaster or ByteBlasterMV cables, see the following documents:

- *USB-Blaster USB Port Download Cable Data Sheet*
- *MasterBlaster Serial/USB Communications Cable Data Sheet*
- *ByteBlasterMV Parallel Port Download Cable Data Sheet*
- *ByteBlaster II Parallel Port Download Cable Data Sheet*

PS Configuration with a Microprocessor

In PS configuration with a microprocessor, a microprocessor transfers data from a storage device to the target Stratix or Stratix GX device. To initiate configuration in this scheme, the microprocessor must generate a low-to-high transition on the nCONFIG pin and the target device must release nSTATUS. The microprocessor or programming hardware then places the configuration data one bit at a time on the DATA0 pin of the Stratix or Stratix GX device. The least significant bit (LSB) of each data byte must be presented first. Data is clocked continuously into the target device until CONF_DONE goes high.

After all configuration data is sent to the Stratix or Stratix GX device, the CONF_DONE pin goes high to show successful configuration and the start of initialization. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize. Initialization, by default, uses an internal oscillator, which runs at 10 MHz. After initialization, this internal oscillator is turned off. If you are using the **clkusr** option, after all data is transferred clkusr must be clocked an additional 136 times for the Stratix or Stratix GX device to initialize properly. Driving DCLK to the device after configuration is complete does not affect device operation.

Handshaking signals are not used in PS configuration modes. Therefore, the configuration clock speed must be below the specified frequency to ensure correct configuration. No maximum DCLK period exists. You can pause configuration by halting DCLK for an indefinite amount of time.

If the target device detects an error during configuration, it drives its nSTATUS pin low to alert the microprocessor. The microprocessor can then pulse nCONFIG low to restart the configuration process. Alternatively, if the **Auto-Restart Configuration on Frame Error** option is turned on in the Quartus II software, the target device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the target device without needing to pulse nCONFIG low.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. If the microprocessor sends all data and the initialization clock starts but CONF_DONE and INIT_DONE have not gone high, it must reconfigure the target device. By default the INIT_DONE output is disabled. You can enable the INIT_DONE output by turning on **Enable INIT_DONE output** option in the Quartus II software.

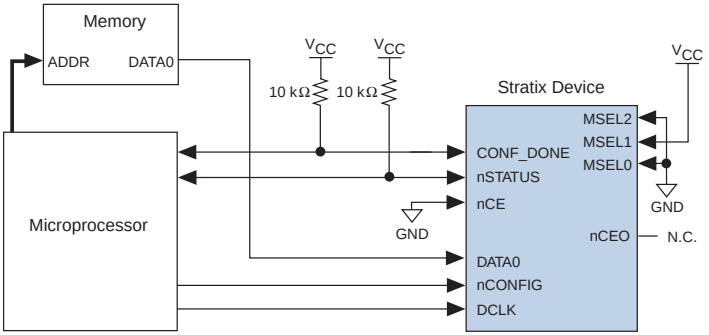
If you do not turn on the **Enable INIT_DONE output** option in the Quartus II software, you are advised to wait for the maximum value of t_{CD2UM} (see Table 11–8) after the CONF_DONE signal goes high to ensure the device has been initialized properly and that it has entered user mode.

During configuration and initialization, and before the device enters user mode, the microprocessor must not drive the CONF_DONE signal low.

1 If the optional CLKUSR pin is used and nCONFIG is pulled low to restart configuration during device initialization, you need to ensure CLKUSR continues toggling during the time nSTATUS is low (maximum of 40 µs).

Figure 11–8 shows the circuit for PS configuration with a microprocessor.

Figure 11–8. PS Configuration Circuit with Microprocessor



PS Configuration Timing

Figure 11–9 shows the PS configuration timing waveform for Stratix and Stratix GX devices. Table 11–8 shows the PS timing parameters for Stratix and Stratix GX devices.

Table 11–8. PS Timing Parameters for Stratix & Stratix GX Devices

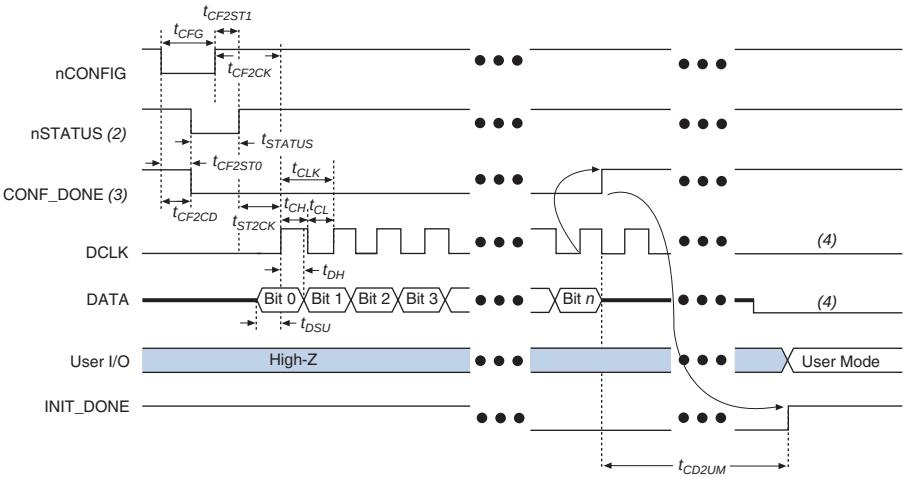
Symbol	Parameter	Min	Max	Units
t_{CF2CD}	nCONFIG low to CONF_DONE low		800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		800	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		40 *(2)*	µs
t_{CFG}	nCONFIG low pulse width	40		µs
t_{STATUS}	nSTATUS low pulse width	10	40 *(2)*	µs
t_{CF2CK}	nCONFIG high to first rising edge on DCLK	40		µs
t_{ST2CK}	nSTATUS high to first rising edge on DCLK	1		µs
t_{DSU}	Data setup time before rising edge on DCLK	7		ns
t_{DH}	Data hold time after rising edge on DCLK	0		ns
t_{CH}	DCLK high time	4		ns
t_{CL}	DCLK low time	4		ns
t_{CLK}	DCLK period	10		ns
f_{MAX}	DCLK maximum frequency		100	MHz
t_{CD2UM}	CONF_DONE high to user mode *(1)*	6	20	µs

Notes to Table 11–8:

(1) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR, multiply the clock period by 136 to obtain this value.

(2) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.

Figure 11–9. PS Timing Waveform for Stratix & Stratix GX Devices Note (1)



Notes to Figure 11–9:

(1) The beginning of this waveform shows the device in user-mode. In user-mode, `nCONFIG`, `nSTATUS`, and `CONF_DONE` are at logic high levels. When `nCONFIG` is pulled low, a reconfiguration cycle begins.

(2) Upon power-up, the Stratix II device holds `nSTATUS` low for the time of the POR delay.

(3) Upon power-up, before and during configuration, `CONF_DONE` is low.

(4) `DCLK` should not be left floating after configuration. It should be driven high or low, whichever is convenient. `DATA[]` is available as user I/Os after configuration and the state of these pins depends on the dual-purpose pin settings.

FPP Configuration

Parallel configuration of Stratix and Stratix GX devices meets the continuously increasing demand for faster configuration times. Stratix and Stratix GX devices can receive byte-wide configuration data per clock cycle, and guarantee a configuration time of less than 100 ms with a 100-MHz configuration clock. Stratix and Stratix GX devices support programming data bandwidth up to 800 megabits per second (Mbps) in this mode. You can use parallel configuration with an EPC16, EPC8, or EPC4 device, or a microprocessor.

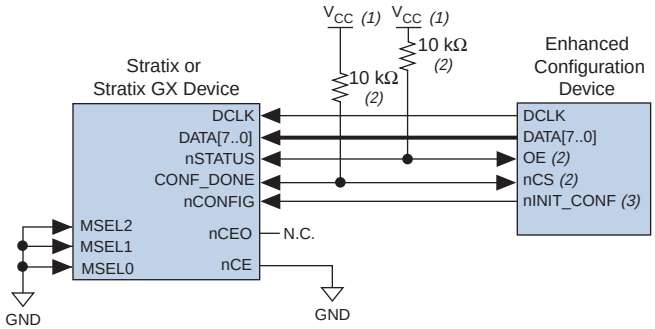
This section discusses the following schemes for FPP configuration in Stratix and Stratix GX devices:

- FPP Configuration Using an Enhanced Configuration Device
- FPP Configuration Using a Microprocessor

FPP Configuration Using an Enhanced Configuration Device

When using FPP with an enhanced configuration device, it supplies data in a byte-wide fashion to the Stratix or Stratix GX device every DCLK cycle. See Figure 11–10.

Figure 11–10. FPP Configuration Using Enhanced Configuration Devices



Notes to Figure 11–10:

(1) The pull-up resistors should be connected to the same supply voltage as the configuration device.

(2) The enhanced configuration devices and EPC2 devices have internal programmable pull-ups on OE and nCS. You should only use the internal pull-ups of the configuration device if the nSTATUS and CONF_DONE signals are pulled up to 3.3 V or 2.5 V (not 1.8 V or 1.5 V). If external pull-ups are used, they should be 10 kΩ

(3) The nINIT_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices. If nINIT_CONF is not used, nCONFIG must be pulled to V_{CC} through a resistor. The nINIT_CONF pin has an internal pull-up resistor that is always active in EPC16, EPC8, EPC4, and EPC2 devices. These devices do not need an external pull-up resistor on the nINIT_CONF pin.

In the enhanced configuration device scheme, nCONFIG is tied to nINIT_CONF. On power up, the target Stratix or Stratix GX device senses the low-to-high transition on nCONFIG and initiates configuration. The target Stratix or Stratix GX device then drives the open-drain CONF_DONE pin low, which in-turn drives the enhanced configuration device's nCS pin low.

Before configuration starts, there is a 2-ms POR delay if the PORSEL pin is connected to V_{CC} in the enhanced configuration device. If the PORSEL pin is connected to ground, the POR delay is 100 ms. When each device determines that its power is stable, it releases its nSTATUS or OE pin. Because the enhanced configuration device's OE pin is connected to the target Stratix or Stratix GX device's nSTATUS pin, configuration is delayed until both the nSTATUS and OE pins are released by each device. The nSTATUS and OE pins are pulled up by a resistor on their respective

devices once they are released. When configuring multiple devices, connect the nSTATUS pins together to ensure configuration only happens when all devices release their OE or nSTATUS pins. The enhanced configuration device then clocks data out in parallel to the Stratix or Stratix GX device using a 66-MHz internal oscillator, or drives it to the Stratix or Stratix GX device through the EXTCLK pin.

If there is an error during configuration, the Stratix or Stratix GX device drives the nSTATUS pin low, resetting itself internally and resetting the enhanced configuration device. The Quartus II software provides an **Auto-restart configuration after error** option that automatically initiates the reconfiguration whenever an error occurs. See the *Software Settings* chapter in Volume 2 of the *Configuration Handbook* for information on how to turn this option on or off.

If this option is turned off, you must set monitor nSTATUS to check for errors. To initiate reconfiguration, pulse nCONFIG low. The external system can pulse nCONFIG if it is under system control rather than tied to V_{CC}. Therefore, nCONFIG must be connected to nINIT_CONF if you want to reprogram the Stratix or Stratix GX device on the fly.

When configuration is complete, the Stratix or Stratix GX device releases the CONF_DONE pin, which is then pulled up by a resistor. This action disables the EPC16, EPC8, or EPC4 enhanced configuration device as nCS is driven high. Initialization, by default, uses an internal oscillator, which runs at 10 MHz. After initialization, this internal oscillator is turned off. When initialization is complete, the Stratix or Stratix GX device enters user mode. The enhanced configuration device drives DCLK low before and after configuration.

1 CONF_DONE goes high one byte early in parallel synchronous (FPP) and asynchronous (PPA) modes using a microprocessor with **.rbf**, **.hex**, and **.ttf** file formats. This does not apply to FPP mode for enhanced configuration devices using **.pof** file format. This also does not apply to serial modes.

If, after sending out all of its data, the enhanced configuration device does not detect CONF_DONE going high, it recognizes that the Stratix or Stratix GX device has not configured successfully. The enhanced configuration device pulses its OE pin low for a few microseconds, driving the nSTATUS pin on the Stratix or Stratix GX device low. If the **Auto-restart configuration after error** option is on, the Stratix or Stratix GX device resets and then pulses its nSTATUS low. When nSTATUS returns high, reconfiguration is restarted (see Figure 11–11 on page 11–25).

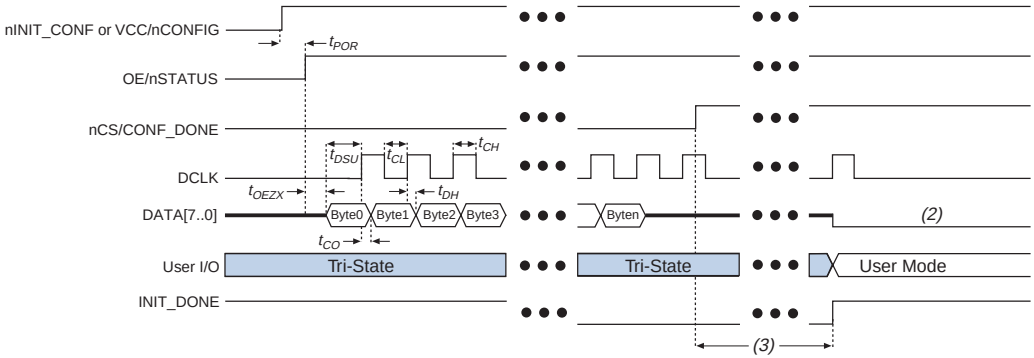
Do not drive CONF_DONE low after device configuration to delay initialization. Instead, use the **Enable User-Supplied Start-Up Clock (CLKUSR)** option in the **Device & Pin Options** dialog box. You can use this option to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain initialize together.

After the first Stratix or Stratix GX device completes configuration during multi-device configuration, its nCEO pin activates the second Stratix or Stratix GX device's nCE pin, prompting the second device to begin configuration. Because CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time. Because nSTATUS pins are tied together, configuration stops for the whole chain if any device (including enhanced configuration devices) detects an error. Also, if the enhanced configuration device does not detect a high on CONF_DONE at the end of configuration, it pulses its OE low for a few microseconds to reset the chain. The low OE pulse drives nSTATUS low on all Stratix and Stratix GX devices, causing them to enter an error state. This state is similar to a Stratix or Stratix GX device detecting an error.

If the **Auto-restart configuration after error** option is on, the Stratix and Stratix GX devices release their nSTATUS pins after a reset time-out period. When the nSTATUS pins are released and pulled high, the configuration device reconfigures the chain. If the **Auto-restart configuration after error** option is off, nSTATUS stays low until the Stratix and Stratix GX devices are reset with a low pulse on nCONFIG.

Figure 11–11 shows the FPP configuration with a configuration device timing waveform for Stratix and Stratix GX devices.

Figure 11–11. FPP Configuration with a Configuration Device Timing Waveform *Note (1)*



Notes to Figure 11–11:

(1) For timing information, see the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet*.

(2) The configuration device drives DATA high after configuration.

(3) Stratix and Stratix GX devices enter user mode 136 clock cycles after CONF_DONE goes high.

FPP Configuration Using a Microprocessor

When using a microprocessor for parallel configuration, the microprocessor transfers data from a storage device to the Stratix or Stratix GX device through configuration hardware. To initiate configuration, the microprocessor needs to generate a low-to-high transition on the nCONFIG pin and the Stratix or Stratix GX device must release nSTATUS. The microprocessor then places the configuration data to the DATA[7..0] pins of the Stratix or Stratix GX device. Data is clocked continuously into the Stratix or Stratix GX device until CONF_DONE goes high.

The configuration clock (DCLK) speed must be below the specified frequency to ensure correct configuration. No maximum DCLK period exists. You can pause configuration by halting DCLK for an indefinite amount of time.

After all configuration data is sent to the Stratix or Stratix GX device, the CONF_DONE pin goes high to show successful configuration and the start of initialization. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize. Initialization, by default, uses an internal oscillator, which runs at 10 MHz. After initialization, this internal oscillator is turned off. If you are using the **clkusr** option, after all data is transferred clkusr must be clocked an additional 136 times for the Stratix or Stratix GX device to initialize properly. Driving DCLK to the device after configuration is complete does not affect device operation. By

default, the INIT_DONE output is disabled. You can enable the INIT_DONE output by turning on the **Enable INIT_DONE output** option in the Quartus II software.

If you do not turn on the **Enable INIT_DONE output** option in the Quartus II software, you are advised to wait for maximum value of t_{CD2UM} (see Table 11–9) after the CONF_DONE signal goes high to ensure the device has been initialized properly and that it has entered user mode.

During configuration and initialization and before the device enters user mode, the microprocessor must not drive the CONF_DONE signal low.

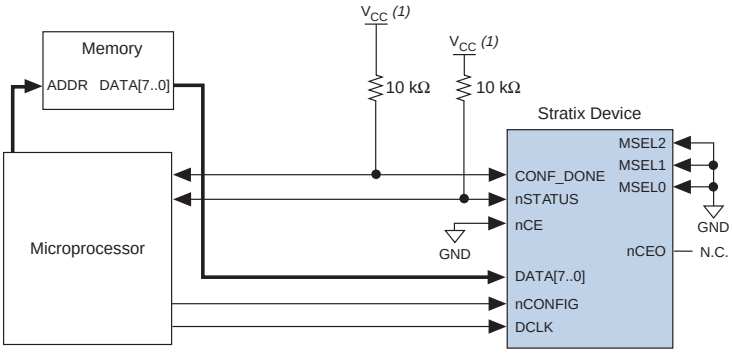
1 If the optional CLKUSR pin is used and nCONFIG is pulled low to restart configuration during device initialization, you need to ensure CLKUSR continues toggling during the time nSTATUS is low (maximum of 40 µs).

If the Stratix or Stratix GX device detects an error during configuration, it drives nSTATUS low to alert the microprocessor. The pin on the microprocessor connected to nSTATUS must be an input. The microprocessor can then pulse nCONFIG low to restart the configuration error. With the **Auto-restart configuration after error** option on, the Stratix or Stratix GX device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the Stratix or Stratix GX device without pulsing nCONFIG low.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. If the microprocessor sends all the data and the initialization clock starts but CONF_DONE and INIT_DONE have not gone high, it must reconfigure the Stratix or Stratix GX device. After waiting the specified 136 DCLK cycles, the microprocessor should restart configuration by pulsing nCONFIG low.

Figure 11–12 shows the circuit for Stratix and Stratix GX parallel configuration using a microprocessor.

Figure 11–12. Parallel Configuration Using a Microprocessor



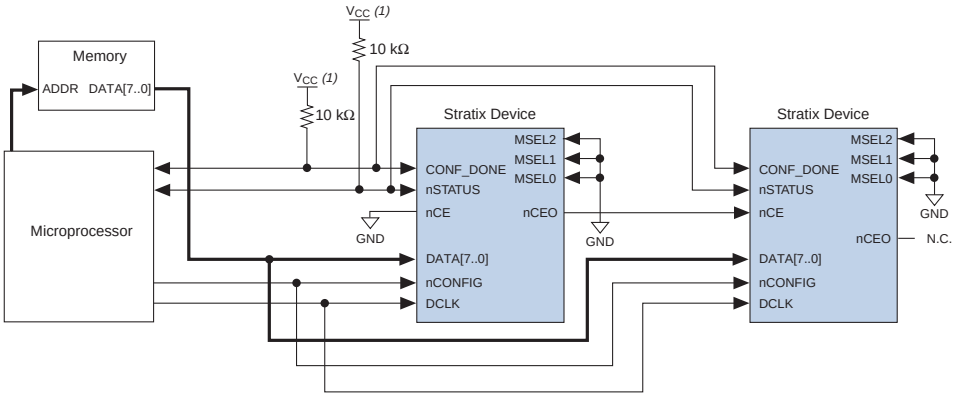
Note to Figure 11–12:

(1) The pull-up resistors should be connected to any V_{CC} that meets the Stratix high-level input voltage (V_{IH}) specification.

For multi-device parallel configuration with a microprocessor, the nCEO pin of the first Stratix or Stratix GX device is cascaded to the second device's nCE pin. The second device in the chain begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the microprocessor. Because the CONF_DONE pins of the devices are connected together, all devices initialize and enter user mode at the same time.

Because the nSTATUS pins are also tied together, if any of the devices detects an error, the entire chain halts configuration and drives nSTATUS low. The microprocessor can then pulse nCONFIG low to restart configuration. If the **Auto-restart configuration after error** option is on, the Stratix and Stratix GX devices release nSTATUS after a reset time-out period. The microprocessor can then reconfigure the devices once nSTATUS is released. Figure 11–13 shows multi-device configuration using a microprocessor. Figure 11–14 shows multi-device configuration when both Stratix and Stratix GX devices are receiving the same data. In this case, the microprocessor sends the data to both devices simultaneously, and the devices configure simultaneously.

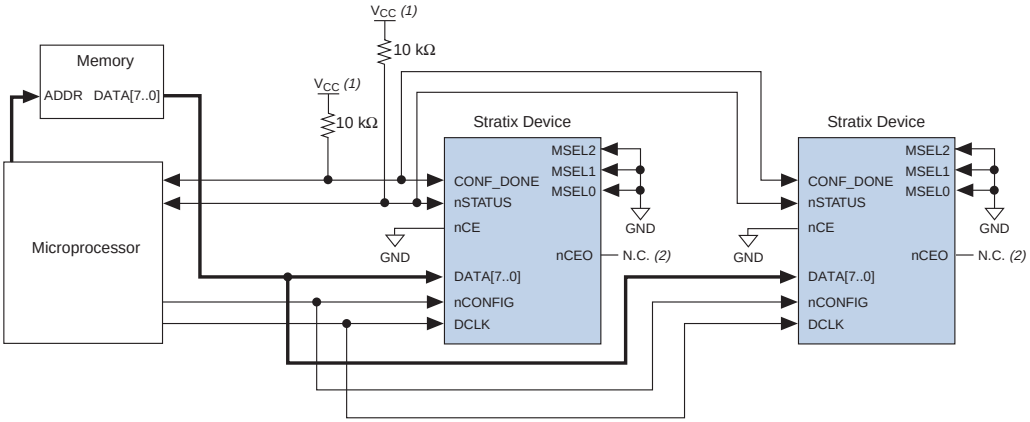
Figure 11–13. Parallel Data Transfer in Serial Configuration with a Microprocessor



Note to Figure 11–13:

(1) You should connect the pull-up resistors to any V_{CC} that meets the Stratix high-level input voltage (V_{IH}) specification.

Figure 11–14. Multiple Device Parallel Configuration with the Same Data Using a Microprocessor



Notes to Figure 11–14:

(1) You should connect the pull-up resistors to any V_{CC} that meets the Stratix high-level input voltage (V_{IH}) specification.

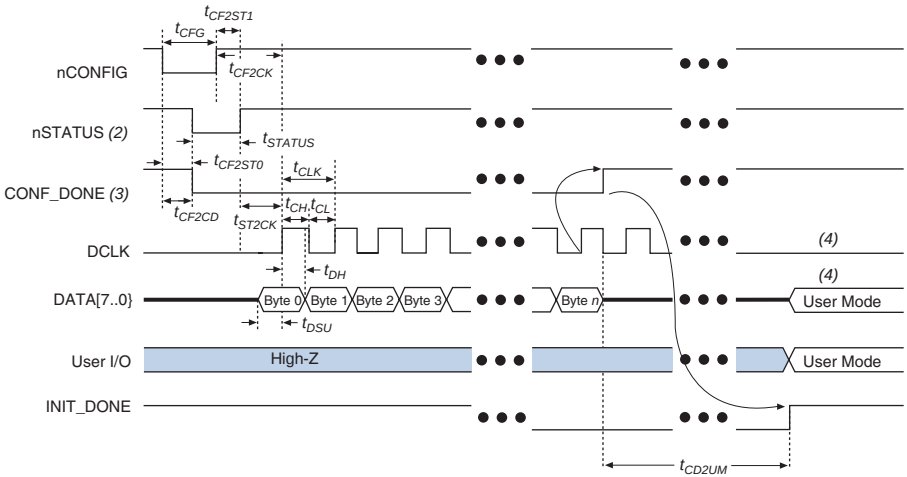
(2) The nCEO pins are left unconnected when configuring the same data into multiple Stratix or Stratix GX devices.

f For more information on configuring multiple Altera devices in the same configuration chain, see the *Configuring Mixed Altera FPGA Chains* chapter in the *Configuration Handbook, Volume 2*.

FPP Configuration Timing

Figure 11–15 shows FPP timing waveforms for configuring a Stratix or Stratix GX device in FPP mode. Table 11–9 shows the FPP timing parameters for Stratix or Stratix GX devices.

Figure 11–15. Timing Waveform for Configuring Devices in FPP Mode *Note (1)*



Notes to Figure 11–15:

(1) The beginning of this waveform shows the device in user-mode. In user-mode, `nCONFIG`, `nSTATUS`, and `CONF_DONE` are at logic high levels. When `nCONFIG` is pulled low, a reconfiguration cycle begins.

(2) Upon power-up, the Stratix II device holds `nSTATUS` low for the time of the POR delay.

(3) Upon power-up, before and during configuration, `CONF_DONE` is low.

(4) `DCLK` should not be left floating after configuration. It should be driven high or low, whichever is convenient. `DATA[]` is available as user I/Os after configuration and the state of these pins depends on the dual-purpose pin settings.

Table 11–9. FPP Timing Parameters for Stratix & Stratix GX Devices (Part 1 of 2)

Symbol	Parameter	Min	Max	Units
t_{CF2CK}	`nCONFIG` high to first rising edge on `DCLK`	40		µs
t_{DSU}	Data setup time before rising edge on `DCLK`	7		ns
t_{DH}	Data hold time after rising edge on `DCLK`	0		ns
t_{CFG}	`nCONFIG` low pulse width	40		µs
t_{CH}	`DCLK` high time	4		ns
t_{CL}	`DCLK` low time	4		ns
t_{CLK}	`DCLK` period	10		ns

Table 11–9. FPP Timing Parameters for Stratix & Stratix GX Devices (Part 2 of 2)

Symbol	Parameter	Min	Max	Units
f_{MAX}	DCLK frequency		100	MHz
t_{CD2UM}	CONF_DONE high to user mode *(1)*	6	20	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		800	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		40 *(2)*	µs
t_{STATUS}	nSTATUS low pulse width	10	40 *(2)*	µs
t_{ST2CK}	nSTATUS high to firstrising edge of DCLK	1		µs

Notes to Table 11–9:

(1) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR, multiply the clock period by 136 to obtain this value.

(2) This value is obtainable if users do not delay configuration by extending the nSTATUS low pulse width.

PPA Configuration

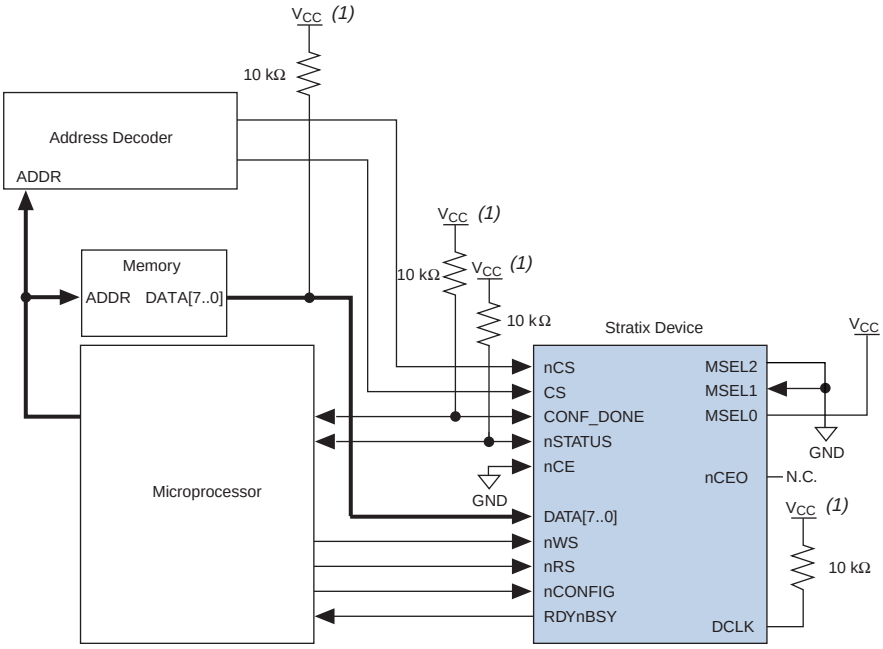
In PPA schemes, a microprocessor drives data to the Stratix or Stratix GX device through a download cable. When using a PPA scheme, use a 1-kΩ pull-up resistor to pull the DCLK pin high to prevent unused configuration pins from floating.

To begin configuration, the microprocessor drives nCONFIG high and then asserts the target device's nCS pin low and CS pin high. Next, the microprocessor places an 8-bit configuration word on the target device's data inputs and pulses nWS low. On the rising edge of nWS, the target device latches a byte of configuration data and then drives its RDYnBSY signal low, indicating that it is processing the byte of configuration data. The microprocessor then performs other system functions while the Stratix or Stratix GX device is processing the byte of configuration data.

Next, the microprocessor checks nSTATUS and CONF_DONE. If nSTATUS is high and CONF_DONE is low, the microprocessor sends the next data byte. If nSTATUS is low, the device is signaling an error and the microprocessor should restart configuration. However, if nSTATUS is high and all the configuration data is received, the device is ready for initialization. At the beginning of initialization, CONF_DONE goes high to indicate that configuration is complete. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize. Initialization, by default, uses an internal oscillator, which runs at 10 MHz. After initialization, this internal oscillator is turned off. When initialization is complete, the Stratix or Stratix GX device enters user mode.

Figure 11–16 shows the PPA configuration circuit. An optional address decoder controls the device's nCS and CS pins. This decoder allows the microprocessor to select the Stratix or Stratix GX device by accessing a particular address, simplifying the configuration process.

Figure 11–16. PPA Configuration Circuit



Note to Figure 11–16:

(1) The pull-up resistor should be connected to the same supply voltage as the Stratix or Stratix GX device.

The device's nCS or CS pins can be toggled during PPA configuration if the design meets the specifications for t_{CSSU}, t_{WSP}, and t_{CSH} given in Table 11–10 on page 11–36. The microprocessor can also directly control the nCS and CS signals. You can tie one of the nCS or CS signals to its active state (i.e., nCS may be tied low) and toggle the other signal to control configuration.

Stratix and Stratix GX devices can serialize data internally without the microprocessor. When the Stratix or Stratix GX device is ready for the next byte of configuration data, it drives RDYnBSY high. If the microprocessor senses a high signal when it polls RDYnBSY, the microprocessor strobes the next byte of configuration data into the device. Alternatively, the nRS signal can be strobed, causing the RDYnBSY signal to appear on DATA7. Because RDYnBSY does not need to

be monitored, reading the state of the configuration data by strobing nRS low saves a system I/O port. Do not drive data onto the data bus while nRS is low because it causes contention on DATA7. If the nRS pin is not used to monitor configuration, you should tie it high. To simplify configuration, the microprocessor can wait for the total time of t_{BUSY}(max) + t_{RDY2WS} + t_{W2SB} before sending the next data bit.

After configuration, the nCS, CS, nRS, nWS, and RDYnBSY pins act as user I/O pins. However, if the PPA scheme is chosen in the Quartus II software, these I/O pins are tri-stated by default in user mode and should be driven by the microprocessor. To change the default settings in the Quartus II software, select **Device & Pin Option** (Compiler Setting menu).

If the Stratix or Stratix GX device detects an error during configuration, it drives nSTATUS low to alert the microprocessor. The microprocessor can then pulse nCONFIG low to restart the configuration process. Alternatively, if the **Auto-Restart Configuration on Frame Error** option is turned on, the Stratix or Stratix GX device releases nSTATUS after a reset time-out period. After nSTATUS is released, the microprocessor can reconfigure the Stratix or Stratix GX device. At this point, the microprocessor does not need to pulse nCONFIG low.

The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The microprocessor must monitor the nSTATUS pin to detect errors and the CONF_DONE pin to determine when programming completes (CONF_DONE goes high one byte early in parallel mode). If the microprocessor sends all configuration data and starts initialization but CONF_DONE is not asserted, the microprocessor must reconfigure the Stratix or Stratix GX device.

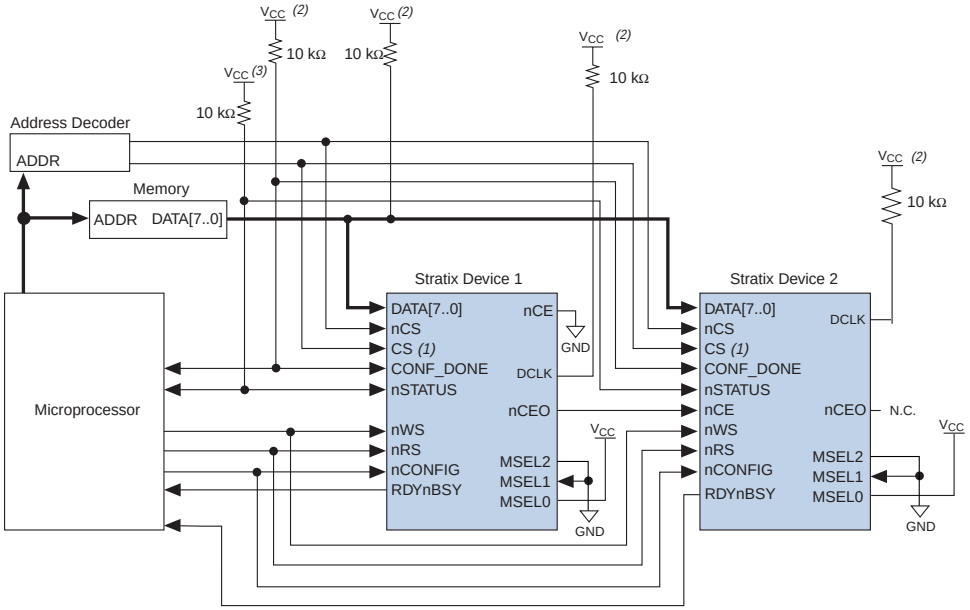
By default, the INIT_DONE is disabled. You can enable the INIT_DONE output by turning on the **Enable INIT_DONE output** option in the Quartus II software. If you do not turn on the **Enable INIT_DONE output** option in the Quartus II software, you are advised to wait for the maximum value of t_{CD2UM} (see Table 11–10) after the CONF_DONE signal goes high to ensure the device has been initialized properly and that it has entered user mode.

During configuration and initialization, and before the device enters user mode, the microprocessor must not drive the CONF_DONE signal low.

1 If the optional CLKUSR pin is used and nCONFIG is pulled low to restart configuration during device initialization, you need to ensure that CLKUSR continues toggling during the time nSTATUS is low (maximum of 40 µs).

You can also use PPA mode to configure multiple Stratix and Stratix GX devices. Multi-device PPA configuration is similar to single-device PPA configuration, except that the Stratix and Stratix GX devices are cascaded. After you configure the first Stratix or Stratix GX device, nCEO is asserted, which asserts the nCE pin on the second device, initiating configuration. Because the second Stratix or Stratix GX device begins configuration within one write cycle of the first device, the transfer of data destinations is transparent to the microprocessor. All Stratix and Stratix GX device CONF_DONE pins are tied together; therefore, all devices initialize and enter user mode at the same time. See Figure 11–17.

Figure 11–17. PPA Multi-Device Configuration Circuit



Notes to Figure 11–17:

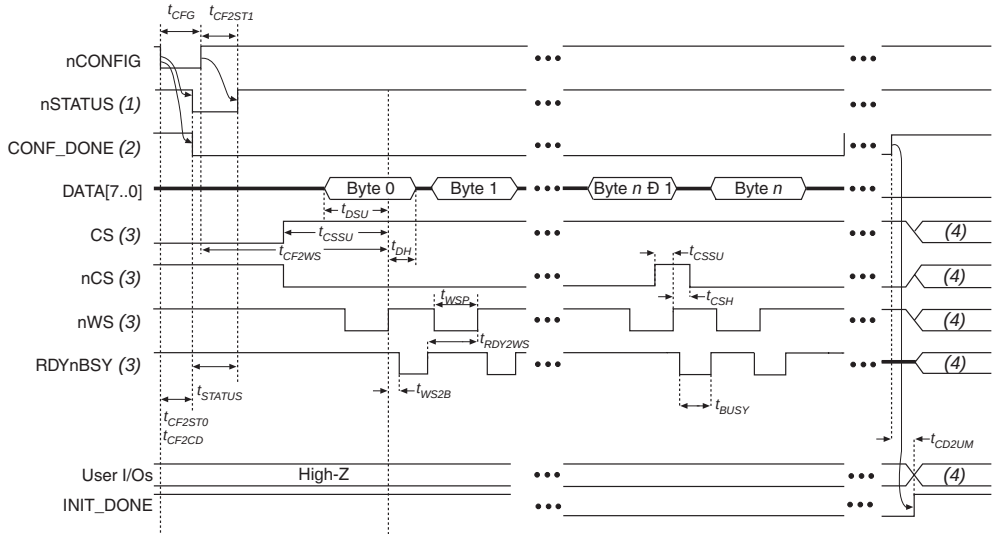
(1) If not used, you can connect the CS pin to V_{CC} directly. If not used, the nCS pin can be connected to GND directly.

(2) Connect the pull-up resistor to the same supply voltage as the Stratix or Stratix GX device.

PPA Configuration Timing

Figure 11–18 shows the Stratix and Stratix GX device timing waveforms for PPA configuration.

Figure 11–18. PPA Timing Waveforms for Stratix & Stratix GX Devices



Notes to Figure 11–18:

(1) Upon power-up, nSTATUS is held low for the time of the POR delay.

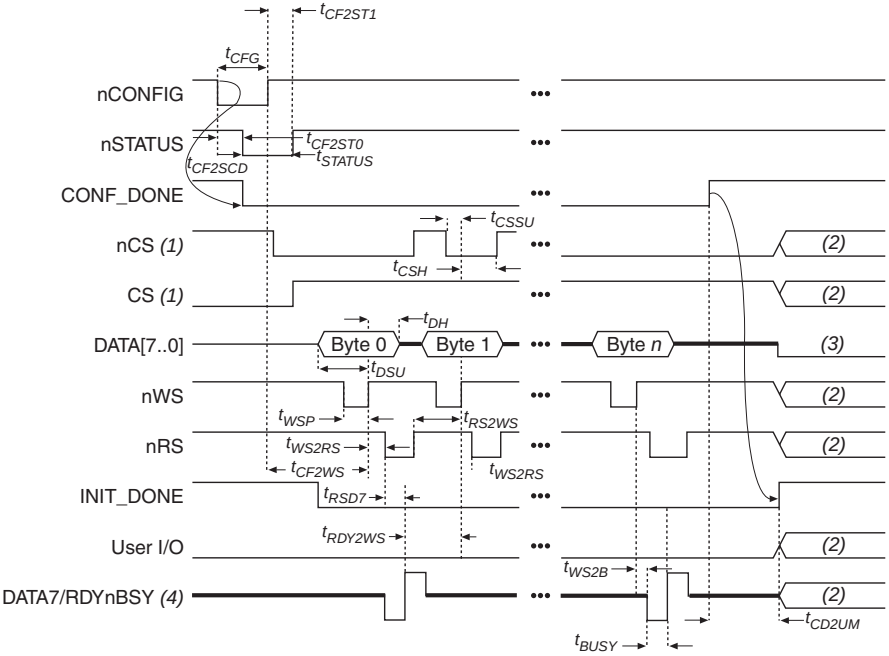
(2) Upon power-up, before and during configuration, CONF_DONE is low.

(3) After configuration, the state of CS, nCS, nWS, and RDYnBSY depends on the design programmed into the Stratix or Stratix GX device.

(4) Device I/O pins are in user mode.

Figure 11–19 shows the Stratix and Stratix GX timing waveforms when using strobed nRS and nWS signals.

Figure 11–19. PPA Timing Waveforms Using Strobed nRS & nWS Signals



Notes to Figure 11–19:

(1) The user can toggle nCS or CS during configuration if the design meets the specification for t_{CSSU}, t_{WSP}, and t_{CSH}.

(2) Device I/O pins are in user mode.

(3) The DATA[7..0] pins are available as user I/Os after configuration and the state of theses pins depends on the dual-purpose pin settings. Do not leave DATA[7..0] floating. If these pins are not used in user-mode, you should drive them high or low, whichever is more convenient.

(4) DATA7 is a bidirectional pin. It represents an input for data input, but represents an output to show the status of RDYnBSY.

Table 11–10 defines the Stratix and Stratix GX timing parameters for PPA configuration

Table 11–10. PPA Timing Parameters for Stratix & Stratix GX Devices

Symbol	Parameter	Min	Max	Units
t_{CF2WS}	nCONFIG high to first rising edge on nWS	40		µs
t_{DSU}	Data setup time before rising edge on nWS	10		ns
t_{DH}	Data hold time after rising edge on nWS	0		ns
t_{CSSU}	Chip select setup time before rising edge on nWS	10		ns
t_{CSH}	Chip select hold time after rising edge on nWS	0		ns
t_{WSP}	nWS low pulse width	15		ns
t_{CFG}	nCONFIG low pulse width	40		µs
t_{WS2B}	nWS rising edge to RDYnBSY low		20	ns
t_{BUSY}	RDYnBSY low pulse width	7	45	ns
t_{RDY2WS}	RDYnBSY rising edge to nWS rising edge	15		ns
t_{WS2RS}	nWS rising edge to nRS falling edge	15		ns
t_{RS2WS}	nRS rising edge to nWS rising edge	15		ns
t_{RSD7}	nRS falling edge to DATA7 valid with RDYnBSY signal		20	ns
t_{CD2UM}	CONF_DONE high to user mode *(1)*	6	20	µs
t_{STATUS}	nSTATUS low pulse width	10	40 *(2)*	µs
t_{CF2CD}	nCONFIG low to CONF_DONE low		800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		800	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high		40 *(2)*	µs

Notes to Table 11–10:

(1) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device. If the clock source is CLKUSR, multiply the clock period by 136 to obtain this value.

(2) This value is obtained if you do not delay configuration by extending the nstatus to low pulse width.

f For information on how to create configuration and programming files for this configuration scheme, see the *Software Settings* section in the *Configuration Handbook, Volume 2*.

JTAG Programming & Configuration

The JTAG has developed a specification for boundary-scan testing. This boundary-scan test (BST) architecture offers the capability to efficiently test components on printed circuit boards (PCBs) with tight lead spacing. The BST architecture can test pin connections without using physical test

probes and capture functional data while a device is operating normally. You can also use the JTAG circuitry to shift configuration data into the device.

f For more information on JTAG boundary-scan testing, see *AN 39: IEEE 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*.

To use the SignalTap® II embedded logic analyzer, you need to connect the JTAG pins of your Stratix device to a download cable header on your PCB.

f For more information on SignalTap II, see the *Design Debugging Using SignalTap II Embedded Logic Analyzer* chapter in the *Quartus II Handbook, Volume 2*.

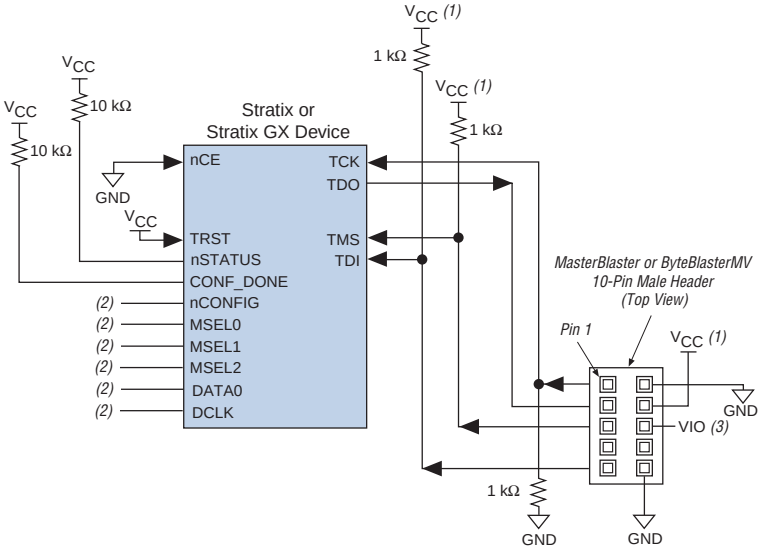
A device operating in JTAG mode uses four required pins, TDI, TDO, TMS, and TCK, and one optional pin, TRST. The four JTAG input pins (TDI, TMS, TCK and TRST) have weak, internal pull-up resistors, whose values range from 20 to 40 kΩ All other pins are tri-stated during JTAG configuration. Do not begin JTAG configuration until all other configuration is complete. Table 11–11 shows each JTAG pin's function.

Table 11–11. JTAG Pin Descriptions

Pin	Description	Function
TDI	Test data input	Serial input pin for instructions as well as test and programming data. Data is shifted in on the rising edge of TCK. The VCCSEL pin controls the input buffer selection.
TDO	Test data output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of TCK. The pin is tri-stated if data is not being shifted out of the device. The high level output voltage is determined by VCCIO.
TMS	Test mode select	Input pin that provides the control signal to determine the transitions of the Test Access Port (TAP) controller state machine. Transitions within the state machine occur on the rising edge of TCK. Therefore, TMS must be set up before the rising edge of TCK. TMS is evaluated on the rising edge of TCK. The VCCSEL pin controls the input buffer selection.
TCK	Test clock input	The clock input to the BST circuitry. Some operations occur at the rising edge, while others occur at the falling edge. The VCCSEL pin controls the input buffer selection.
TRST	Test reset input (optional)	Active-low input to asynchronously reset the boundary-scan circuit. The TRST pin is optional according to IEEE Std. 1149.1. The VCCSEL pin controls the input buffer selection.

During JTAG configuration, data is downloaded to the device on the PCB through the MasterBlaster or ByteBlasterMV header. Configuring devices through a cable is similar to programming devices in-system. One difference is to connect the TRST pin to V_{CC} to ensure that the TAP controller is not reset. See Figure 11–20.

Figure 11–20. JTAG Configuration of a Single Device



Notes to Figure 11–20:

(1) You should connect the pull-up resistor to the same supply voltage as the download cable.

(2) You should connect the nCONFIG, MSEL0, and MSEL1 pins to support a non-JTAG configuration scheme. If you only use JTAG configuration, connect nCONFIG to V_{CC}, and MSEL0, MSEL1, and MSEL2 to ground. Pull DATA0 and DCLK to high or low.

(3) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. See the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

To configure a single device in a JTAG chain, the programming software places all other devices in BYPASS mode. In BYPASS mode, devices pass programming data from the TDI pin to the TDO pin through a single bypass register without being affected internally. This scheme enables the programming software to program or verify the target device. Configuration data driven into the device appears on the TDO pin one clock cycle later.

Stratix and Stratix GX devices have dedicated JTAG pins. You can perform JTAG testing on Stratix and Stratix GX devices before and after, but not during configuration. The chip-wide reset and output enable pins on Stratix and Stratix GX devices do not affect JTAG boundary-scan or programming operations. Toggling these pins does not affect JTAG operations (other than the usual boundary-scan operation).

When designing a board for JTAG configuration of Stratix and Stratix GX devices, you should consider the regular configuration pins. Table 11–12 shows how you should connect these pins during JTAG configuration.

Table 11–12. Dedicated Configuration Pin Connections During JTAG Configuration

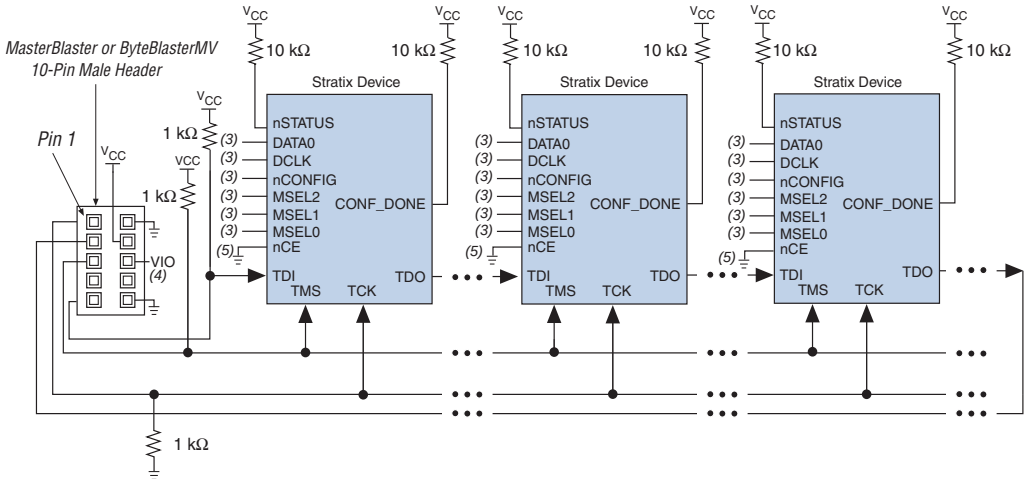
Signal	Description
nCE	On all Stratix and Stratix GX devices in the chain, nCE should be driven low by connecting it to ground, pulling it low via a resistor, or driving it by some control circuitry. For devices that are also in multi-device PS, FPP or PPA configuration chains, the nCE pins should be connected to GND during JTAG configuration or JTAG configured in the same order as the configuration chain.
nCEO	On all Stratix and Stratix GX devices in the chain, nCEO can be left floating or connected to the nCE of the next device. See nCE pin description above.
MSEL	These pins must not be left floating. These pins support whichever non-JTAG configuration is used in production. If only JTAG configuration is used, you should tie both pins to ground.
nCONFIG	nCONFIG must be driven high through the JTAG programming process. Driven high by connecting to V_{CC}, pulling high via a resistor, or driven by some control circuitry.
nSTATUS	Pull to V_{CC} via a 10-kΩ resistor. When configuring multiple devices in the same JTAG chain, each nSTATUS pin should be pulled up to V_{CC} individually. nSTATUS pulling low in the middle of JTAG configuration indicates that an error has occurred.
CONF_DONE	Pull to V_{CC} via a 10-kΩ resistor. When configuring multiple devices in the same JTAG chain, each CONF_DONE pin should be pulled up to V_{CC} individually. CONF_DONE going high at the end of JTAG configuration indicates successful configuration.
DCLK	Should not be left floating. Drive low or high, whichever is more convenient on your board.
DATA0	Should not be left floating. Drive low or high, whichever is more convenient on your board.

JTAG Programming & Configuration of Multiple Devices

When programming a JTAG device chain, one JTAG-compatible header, such as the ByteBlasterMV header, is connected to several devices. The number of devices in the JTAG chain is limited only by the drive capacity of the download cable. However, when more than five devices are connected in a JTAG chain, Altera recommends buffering the TCK, TDI, and TMS pins with an on-board buffer.

JTAG-chain device programming is ideal when the PCB contains multiple devices, or when testing the PCB using JTAG BST circuitry. Figure 11–21 shows multi-device JTAG configuration.

***Figure 11–21. Multi-Device JTAG Configuration** Notes (1), (2)*



Notes to Figure 11–21:

(1) Stratix, Stratix GX, APEX™ II, APEX 20K, Mercury™, ACEX® 1K, and FLEX® 10K devices can be placed within the same JTAG chain for device programming and configuration.

(2) For more information on all configuration pins connected in this mode, see Table 11–11 on page 11–37.

(3) Connect the `nCONFIG`, `MSEL0`, `MSEL1`, and `MSEL2` pins to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect `nCONFIG` to V_{CC}, and `MSEL0`, `MSEL1`, and `MSEL2` to ground. Pull `DATA0` and `DCLK` to either high or low.

(4) V_{IO} is a reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. See the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value.

(5) `nCE` must be connected to GND or driven low for successful JTAG configuration.

The `nCE` pin must be connected to `GND` or driven low during JTAG configuration. In multi-device PS, FPP and PPA configuration chains, the first device's `nCE` pin is connected to `GND` while its `nCEO` pin is connected to `nCE` of the next device in the chain. The last device's `nCE` input comes from the previous device, while its `nCEO` pin is left floating. After the first device completes configuration in a multi-device configuration chain, its `nCEO` pin drives low to activate the second device's `nCE` pin, which prompts the second device to begin configuration. Therefore, if these devices are also in a JTAG chain, you should make sure the `nCE` pins are connected to `GND` during JTAG configuration or that the devices are JTAG configured in the same order as the configuration chain. As long as the devices are JTAG configured in the same order as the multi-device configuration chain, the `nCEO` of the previous device drives `nCE` of the next device low when it has successfully been JTAG configured.

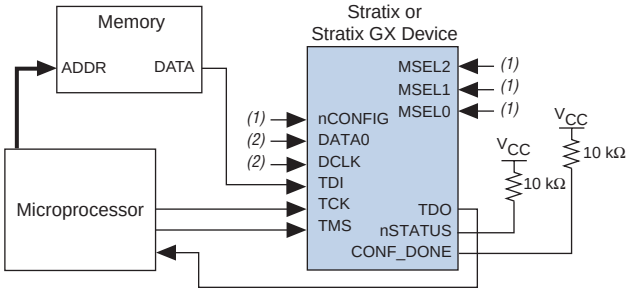
The Quartus II software verifies successful JTAG configuration upon completion. The software checks the state of CONF_DONE through the JTAG port. If CONF_DONE is not in the correct state, the Quartus II software indicates that configuration has failed. If CONF_DONE is in the correct state, the software indicates that configuration was successful.

1 If VCCIO is tied to 3.3 V, both the I/O pins and JTAG TDO port drive at 3.3-V levels.

Do not attempt JTAG and non-JTAG configuration simultaneously. When configuring through JTAG, allow any non-JTAG configuration to complete first.

Figure 11–22 shows the JTAG configuration of a Stratix or Stratix GX device with a microprocessor.

Figure 11–22. JTAG Configuration of Stratix & Stratix GX Devices with a Microprocessor



Notes to Figure 11–22:

(1) Connect the nCONFIG, MSEL2, MSEL1, and MSEL0 pins to support a non-JTAG configuration scheme. If your design only uses JTAG configuration, connect the nCONFIG pin to V_{CC} and the MSEL2, MSEL1, and MSEL0 pins to ground.

(2) Pull DATA0 and DCLK to either high or low.

Configuration with JRunner Software Driver

JRunner is a software driver that allows you to configure Altera FPGAs through the ByteBlasterMV download cable in JTAG mode. The programming input file supported is in Raw Binary File (**.rbf**) format. JRunner also requires a Chain Description File (**.cdf**) generated by the Quartus II software. JRunner is targeted for embedded JTAG configuration. The source code has been developed for the Windows NT operating system. You can customize the code to make it run on other platforms.

f For more information on the JRunner software driver, see the *JRunner Software Driver: An Embedded Solution to the JTAG Configuration White Paper* and zip file.

Jam STAPL Programming & Test Language

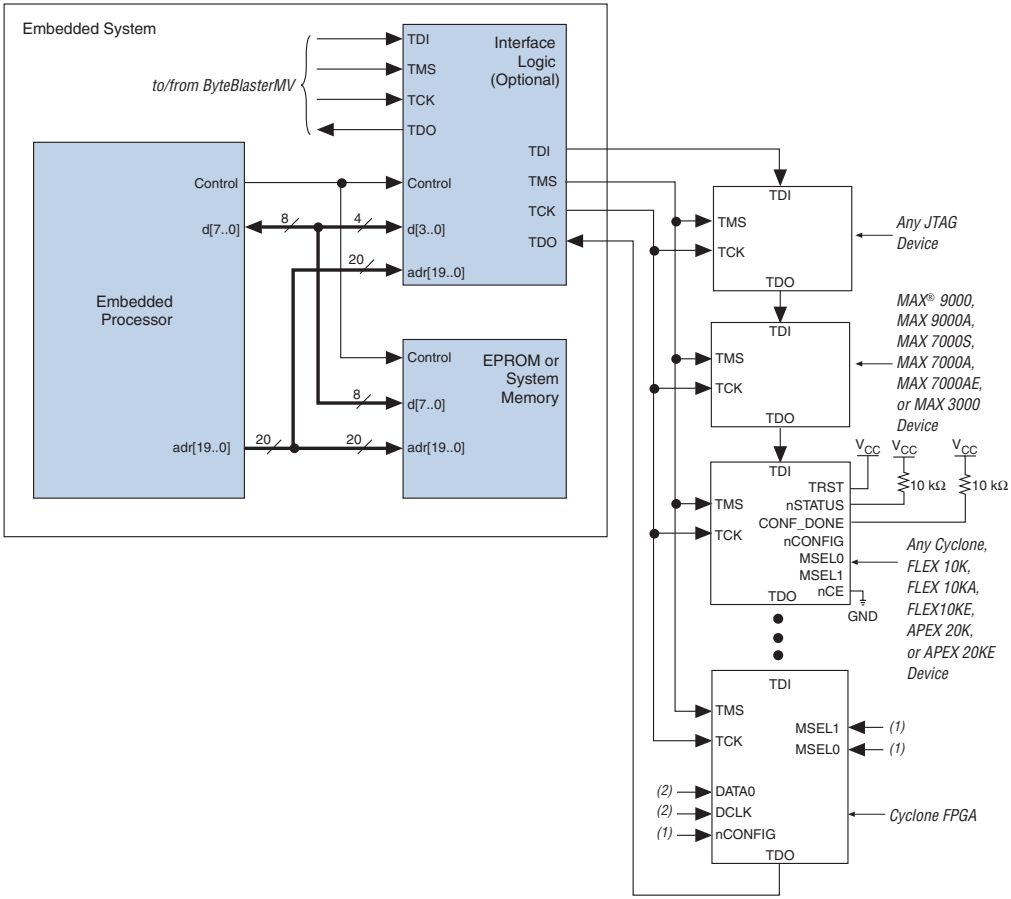
The Jam™ Standard Test and Programming Language (STAPL), JEDEC standard JESD-71, is a standard file format for in-system programmability (ISP) purposes. Jam STAPL supports programming or configuration of programmable devices and testing of electronic systems, using the IEEE 1149.1 JTAG interface. Jam STAPL is a freely licensed open standard.

Connecting the JTAG Chain to the Embedded Processor

There are two ways to connect the JTAG chain to the embedded processor. The most straightforward method is to connect the embedded processor directly to the JTAG chain. In this method, four of the processor pins are dedicated to the JTAG interface, saving board space but reducing the number of available embedded processor pins.

Figure 11–23 illustrates the second method, which is to connect the JTAG chain to an existing bus through an interface PLD. In this method, the JTAG chain becomes an address on the existing bus. The processor then reads from or writes to the address representing the JTAG chain.

Figure 11–23. Embedded System Block Diagram



Notes to Figure 11–23:

(1) Connect the `nCONFIG`, `MSEL2`, `MSEL1`, and `MSEL0` pins to support a non-JTAG configuration scheme. If your design only uses JTAG configuration, connect the `nCONFIG` pin to V_{CC} and the `MSEL2`, `MSEL1`, and `MSEL0` pins to ground.

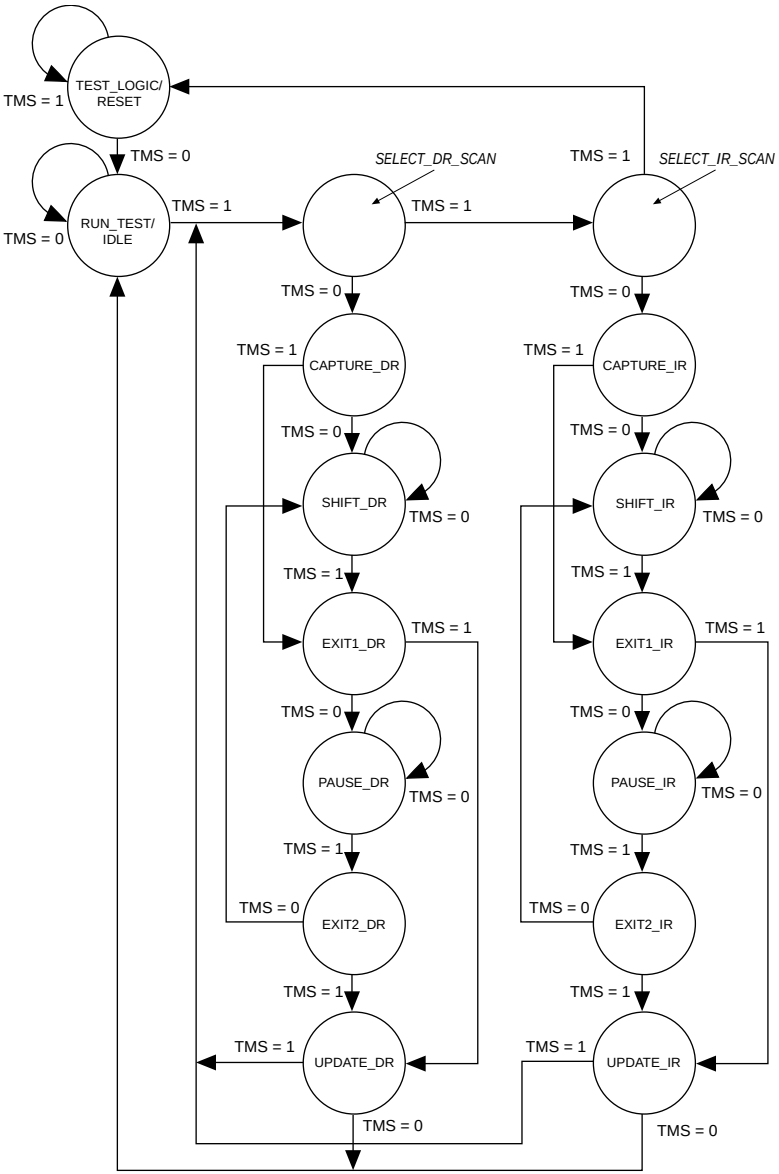
(2) Pull `DATA0` and `DCLK` to either high or low.

Both JTAG connection methods should include space for the MasterBlaster or ByteBlasterMV header connection. The header is useful during prototyping because it allows you to verify or modify the Stratix or Stratix GX device's contents. During production, you can remove the header to save cost.

Program Flow

The Jam Player provides an interface for manipulating the IEEE Std. 1149.1 JTAG TAP state machine. The TAP controller is a 16-state state machine that is clocked on the rising edge of TCK, and uses the TMS pin to control JTAG operation in a device. Figure 11–24 shows the flow of an IEEE Std. 1149.1 TAP controller state machine.

Figure 11–24. JTAG TAP Controller State Machine



While the Jam Player provides a driver that manipulates the TAP controller, the Jam Byte-Code File (**.jbc**) provides the high-level intelligence needed to program a given device. All Jam instructions that

send JTAG data to the device involve moving the TAP controller through either the data register leg or the instruction register leg of the state machine. For example, loading a JTAG instruction involves moving the TAP controller to the SHIFT_IR state and shifting the instruction into the instruction register through the TDI pin. Next, the TAP controller is moved to the RUN_TEST/IDLE state where a delay is implemented to allow the instruction time to be latched. This process is identical for data register scans, except that the data register leg of the state machine is traversed.

The high-level Jam instructions are the DRSCAN instruction for scanning the JTAG data register, the IRSCAN instruction for scanning the instruction register, and the WAIT command that causes the state machine to sit idle for a specified period of time. Each leg of the TAP controller is scanned repeatedly, according to instructions in the JBC file, until all of the target devices are programmed.

Figure 11–25 illustrates the functional behavior of the Jam Player when it parses the JBC file. When the Jam Player encounters a DRSCAN, IRSCAN, or WAIT instruction, it generates the proper data on TCK, TMS, and TDI to complete the instruction. The flow diagram shows branches for the DRSCAN, IRSCAN, and WAIT instructions. Although the Jam Player supports other instructions, they are omitted from the flow diagram for simplicity.

Figure 11–25. Jam Player Flow Diagram (Part 1 of 2)

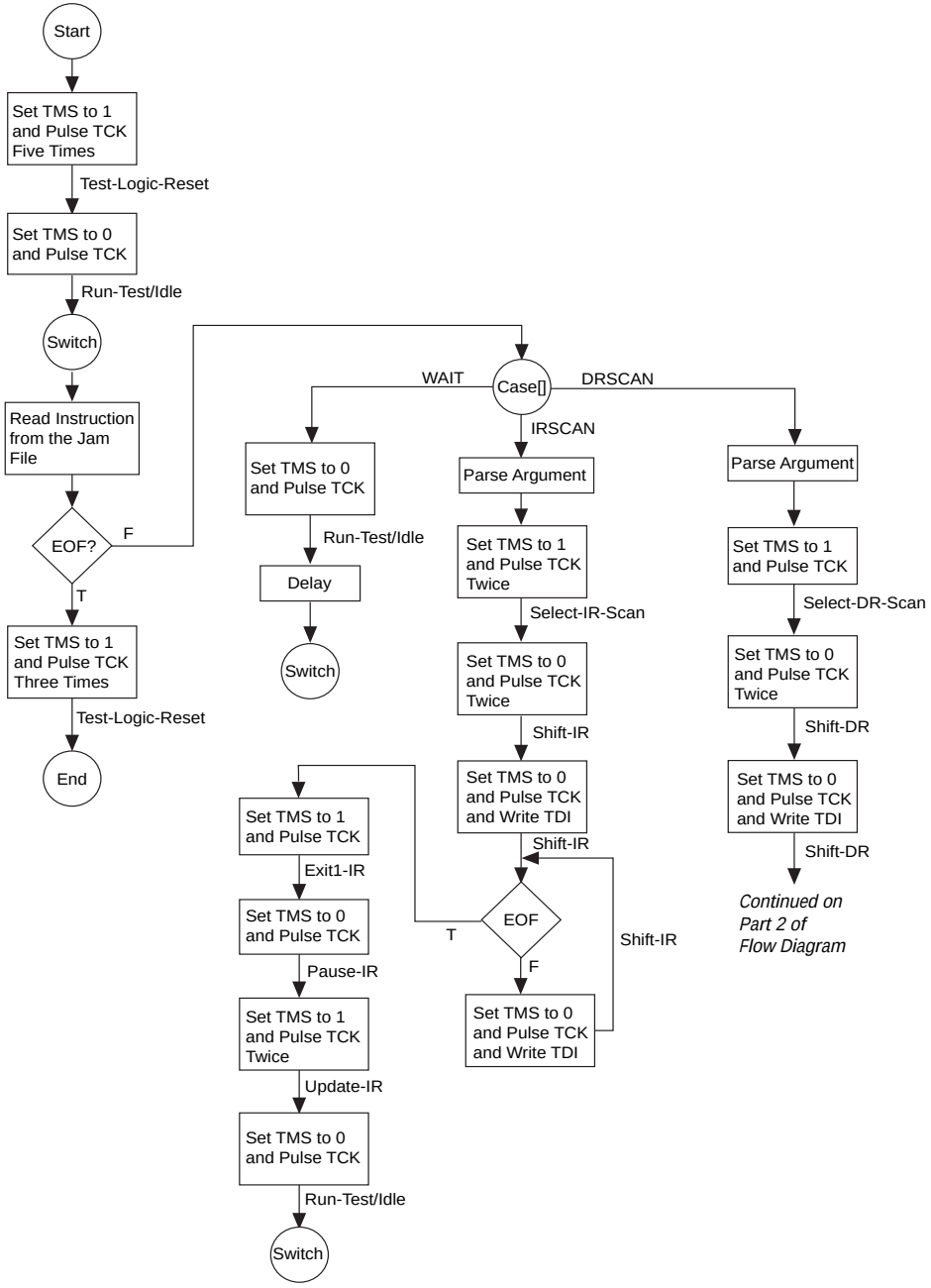
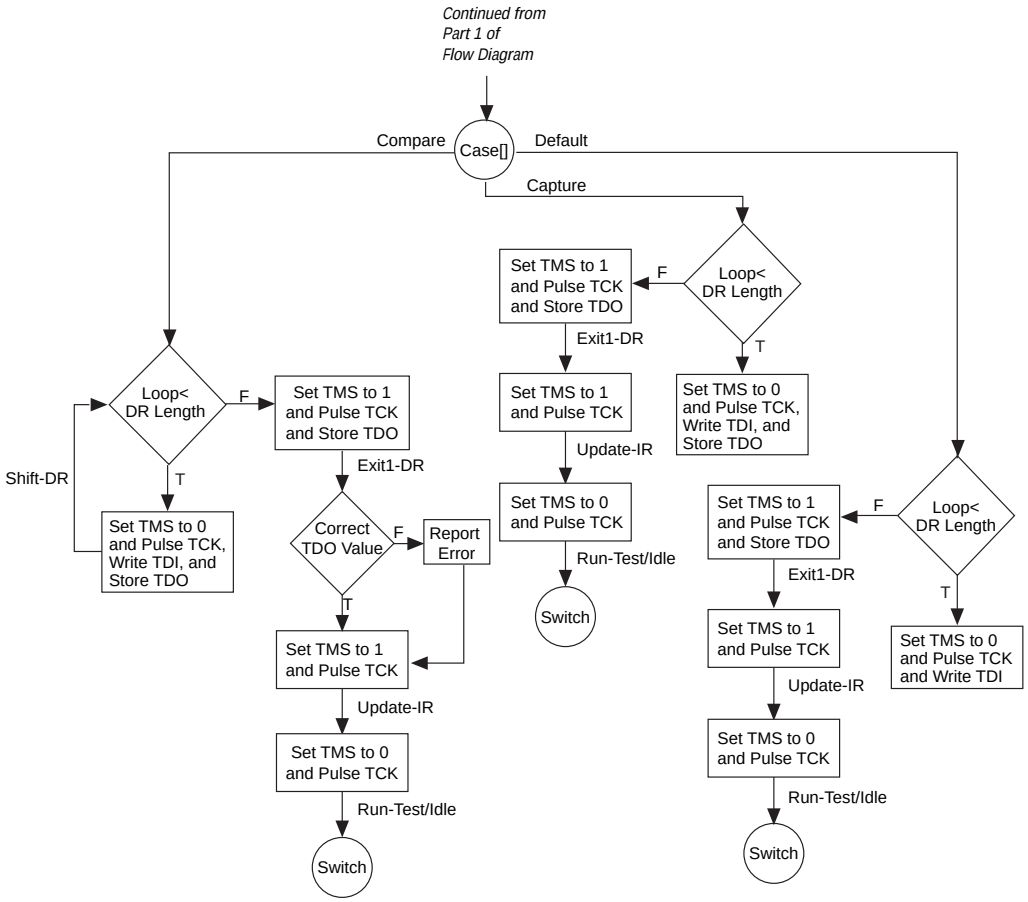


Figure 11–26. Jam Player Flow Diagram (Part 2 of 2)



Execution of a Jam program starts at the beginning of the program. The program flow is controlled using `GOTO`, `CALL/RETURN`, and `FOR/NEXT` structures. The `GOTO` and `CALL` statements see labels that are symbolic names for program statements located elsewhere in the Jam program. The language itself enforces almost no constraints on the organizational structure or control flow of a program.

1 The Jam language does not support linking multiple Jam programs together or including the contents of another file into a Jam program.

Jam Instructions

Each Jam statement begins with one of the instruction names listed in Table 11–13. The instruction names, including the names of the optional instructions, are reserved keywords that you cannot use as variable or label identifiers in a Jam program.

Table 11–13. Instruction Names

BOOLEAN	INTEGER	PREIR
CALL	IRSCAN	PRINT
CRC	IRSTOP	PUSH
DRSCAN	LET	RETURN
DRSTOP	NEXT	STATE
EXIT	NOTE	WAIT
EXPORT	POP	VECTOR *(1)*
FOR	POSTDR	VMAP *(1)*
GOTO	POSTIR	–
IF	PREDR	–

Note to Table 11–13:
(1) This instruction name is an optional language extension.

Table 11–14 shows the state names that are reserved keywords in the Jam language. These keywords correspond to the state names specified in the IEEE Std. 1149.1 JTAG specification.

Table 11–14. Reserved Keywords (Part 1 of 2)

IEEE Std. 1149.1 JTAG State Names	Jam Reserved State Names
Test-Logic-Reset	RESET
Run-Test-Idle	IDLE
Select-DR-Scan	DRSELECT
Capture-DR	DRCAPTURE
Shift-DR	DRSHIFT
Exit1-DR	DREXIT1
Pause-DR	DRPAUSE
Exit2-DR	DREXIT2
Update-DR	DRUPDATE
Select-IR-Scan	IRSELECT

Table 11–14. Reserved Keywords (Part 2 of 2)

IEEE Std. 1149.1 JTAG State Names	Jam Reserved State Names
Capture-IR	IRCAPTURE
Shift-IR	IRSHIFT
Exit1-IR	IREXIT1
Pause-IR	IRPAUSE
Exit2-IR	IREXIT2
Update-IR	IRUPDATE

Example Jam File that Reads the IDCODE

Figure 11–27 illustrates the flexibility and utility of the Jam STAPL. The example reads the IDCODE out of a single device in a JTAG chain.

1 The array variable, I_IDCODE, is initialized with the IDCODE instruction bits ordered the LSB first (on the left) to most significant bit (MSB) (on the right). This order is important because the array field in the IRSCAN instruction is always interpreted, and sent, MSB to LSB.

Figure 11–27. Example Jam File Reading IDCODE

```
BOOLEAN read_data[32];
BOOLEAN I_IDCODE[10] = BIN 1001101000; ‘assumed
BOOLEAN ONES_DATA[32] = HEX FFFFFFFF;
INTEGER i;
‘Set up stop state for IRSCAN
IRSTOP IRPAUSE;
‘Initialize device
STATE RESET;
IRSCAN 10, I_IDCODE[0..9]; ‘LOAD IDCODE INSTRUCTION
STATE IDLE;
WAIT 5 USEC, 3 CYCLES;
DRSCAN 32, ONES_DATA[0..31], CAPTURE
read_data[0..31];
‘CAPTURE IDCODE
PRINT “IDCODE:”;
FOR i=0 to 31;
PRINT read_data[i];
NEXT i;
EXIT 0;
```

Configuring Using the MicroBlaster Driver

The MicroBlaster™ software driver allows you to configure Altera devices in an embedded environment using PS or FPP mode. The MicroBlaster software driver supports a Raw Binary File (**.rbf**) programming input file. The source code is developed for the Windows NT operating system, although you can customize it to run on other operating systems. For more information on the MicroBlaster software driver, go to the Altera web site (**www.altera.com**).

Device Configuration Pins

The following tables describe the connections and functionality of all the configuration related pins on the Stratix or Stratix GX device. Table 11–15 describes the dedicated configuration pins, which are required to be connected properly on your board for successful configuration. Some of these pins may not be required for your configuration schemes.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 1 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
VCCSEL	N/A	All	Input	Dedicated input that selects which input buffer is used on the configuration input pins; nCONFIG, DCLK, RUnLU, nCE, nWS, nRS, CS, nCS and CLKUSR. The VCCSEL input buffer is powered by V_{CCINT} and has an internal 2.5 kΩ pull-down resistor that is always active. A logic high (1.5-V, 1.8-V, 2.5-V, 3.3-V) selects the 1.8-V/1.5-V input buffer, and a logic low selects the 3.3-V/2.5-V input buffer. See the "V_{CCSEL} Pins" section for more details.
PORSEL	N/A	All	Input	Dedicated input which selects between a POR time of 2 ms or 100 ms. A logic high (1.5-V, 1.8-V, 2.5-V, 3.3-V) selects a POR time of about 2 ms and a logic low selects POR time of about 100 ms. The PORSEL input buffer is powered by V_{CCINT} and has an internal 2.5 kΩ pull-down resistor that is always active.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 2 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nIO_PULLUP	N/A	All	Input	Dedicated input that chooses whether the internal pull-ups on the user I/Os and dual-purpose I/Os (DATA[7..0], nWS, nRS, RDYnBSY, nCS, CS, RUnLU, PGM[], CLKUSR, INIT_DONE, DEV_OE, DEV_CLR) are on or off before and during configuration. A logic high (1.5-V, 1.8-V, 2.5-V, 3.3-V) turns off the weak internal pull-ups, while a logic low turns them on. The nIO_PULLUP input buffer is powered by V_{CCINT} and has an internal 2.5 kΩ pull-down resistor that is always active.
MSEL[2..0]	N/A	All	Input	3-bit configuration input that sets the Stratix or Stratix GX device configuration scheme. See Table 11–2 for the appropriate connections. These pins can be connected to V_{CCIO} of the I/O bank they reside in or ground. This pin uses Schmitt trigger input buffers.
nCONFIG	N/A	All	Input	Configuration control input. Pulling this pin low during user-mode causes the FPGA to lose its configuration data, enter a reset state, tri-state all I/O pins. Returning this pin to a logic high level initiates a reconfiguration. If your configuration scheme uses an enhanced configuration device or EPC2 device, nCONFIG can be tied directly to V_{CC} or to the configuration device's nINIT_CONF pin. This pin uses Schmitt trigger input buffers.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 3 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nSTATUS	N/A	All	Bidirectional open-drain	The device drives nSTATUS low immediately after power-up and releases it after the POR time. Status output. If an error occurs during configuration, nSTATUS is pulled low by the target device. Status input. If an external source drives the nSTATUS pin low during configuration or initialization, the target device enters an error state. Driving nSTATUS low after configuration and initialization does not affect the configured device. If a configuration device is used, driving nSTATUS low causes the configuration device to attempt to configure the FPGA, but since the FPGA ignores transitions on nSTATUS in user-mode, the FPGA does not reconfigure. To initiate a reconfiguration, nCONFIG must be pulled low. The enhanced configuration devices' and EPC2 devices' OE and nCS pins have optional internal programmable pull-up resistors. If internal pull-up resistors on the enhanced configuration device are used, external 10-kΩ pull-up resistors should not be used on these pins. When using EPC2 devices, only external 10-kΩ pull-up resistors should be used. This pin uses Schmitt trigger input buffers.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 4 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
CONF_DONE	N/A	All	Bidirectional open-drain	Status output. The target FPGA drives the CONF_DONE pin low before and during configuration. Once all configuration data is received without error and the initialization cycle starts, the target device releases CONF_DONE. Status input. After all data is received and CONF_DONE goes high, the target device initializes and enters user mode. The CONF_DONE pin must have an external 10-kΩ pull-up resistor in order for the device to initialize. Driving CONF_DONE low after configuration and initialization does not affect the configured device. The enhanced configuration devices' and EPC2 devices' OE and nCS pins have optional internal programmable pull-up resistors. If internal pull-up resistors on the enhanced configuration device are used, external 10-kΩ pull-up resistors should not be used on these pins. When using EPC2 devices, only external 10-kΩ pull-up resistors should be used. This pin uses Schmitt trigger input buffers.
nCE	N/A	All	Input	Active-low chip enable. The nCE pin activates the device with a low signal to allow configuration. The nCE pin must be held low during configuration, initialization, and user mode. In single device configuration, it should be tied low. In multi-device configuration, nCE of the first device is tied low while its nCEO pin is connected to nCE of the next device in the chain. The nCE pin must also be held low for successful JTAG programming of the FPGA. This pin uses Schmitt trigger input buffers.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 5 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nCEO	N/A	All Multi-Device Schemes	Output	Output that drives low when device configuration is complete. In single device configuration, this pin is left floating. In multi-device configuration, this pin feeds the next device's nCE pin. The nCEO of the last device in the chain is left floating. The voltage levels driven out by this pin are dependent on the V_{CCIO} of the I/O bank it resides in.
DCLK	N/A	Synchronous configuration schemes (PS, FPP)	Input (PS, FPP)	In PS and FPP configuration, DCLK is the clock input used to clock data from an external source into the target device. Data is latched into the FPGA on the rising edge of DCLK. In PPA mode, DCLK should be tied high to V_{CC} to prevent this pin from floating. After configuration, this pin is tri-stated. In schemes that use a configuration device, DCLK is driven low after configuration is done. In schemes that use a control host, DCLK should be driven either high or low, whichever is more convenient. Toggling this pin after configuration does not affect the configured device. This pin uses Schmitt trigger input buffers.
DATA0	I/O	PS, FPP, PPA	Input	Data input. In serial configuration modes, bit-wide configuration data is presented to the target device on the DATA0 pin. The V_{IH} and V_{IL} levels for this pin are dependent on the V_{CCIO} of the I/O bank that it resides in. After configuration, DATA0 is available as a user I/O and the state of this pin depends on the **Dual-Purpose Pin** settings. After configuration, EPC1 and EPC1441 devices tri-state this pin, while enhanced configuration and EPC2 devices drive this pin high.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 6 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
DATA[7..1]	I/O	Parallel configuration schemes (FPP and PPA)	Inputs	Data inputs. Byte-wide configuration data is presented to the target device on DATA[7..0]. The V_{IH} and V_{IL} levels for these pins are dependent on the V_{CCIO} of the I/O banks that they reside in. In serial configuration schemes, they function as user I/Os during configuration, which means they are tri-stated. After PPA or FPP configuration, DATA[7..1] are available as a user I/Os and the state of these pin depends on the **Dual-Purpose Pin** settings.
DATA7	I/O	PPA	Bidirectional	In the PPA configuration scheme, the DATA7 pin presents the RDYnBSY signal after the nRS signal has been strobed low. The V_{IL} and V_{IL} levels for this pin are dependent on the V_{CCIO} of the I/O bank that it resides in. In serial configuration schemes, it functions as a user I/O during configuration, which means it is tri-stated. After PPA configuration, DATA7 is available as a user I/O and the state of this pin depends on the **Dual-Purpose Pin** settings.
nWS	I/O	PPA	Input	Write strobe input. A low-to-high transition causes the device to latch a byte of data on the DATA[7..0] pins. In non-PPA schemes, it functions as a user I/O during configuration, which means it is tri-stated. After PPA configuration, nWS is available as a user I/O and the state of this pin depends on the **Dual-Purpose Pin** settings.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 7 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nRS	I/O	PPA	Input	Read strobe input. A low input directs the device to drive the RDYnBSY signal on the DATA7 pin. If the nRS pin is not used in PPA mode, it should be tied high. In non-PPA schemes, it functions as a user I/O during configuration, which means it is tri-stated. After PPA configuration, nRS is available as a user I/O and the state of this pin depends on the **Dual-Purpose Pin** settings.
RDYnBSY	I/O	PPA	Output	Ready output. A high output indicates that the target device is ready to accept another data byte. A low output indicates that the target device is busy and not ready to receive another data byte. In PPA configuration schemes, this pin drives out high after power-up, before configuration and after configuration before entering user-mode. In non-PPA schemes, it functions as a user I/O during configuration, which means it is tri-stated. After PPA configuration, RDYnBSY is available as a user I/O and the state of this pin depends on the **Dual-Purpose Pin** settings.

Table 11–15. Dedicated Configuration Pins on the Stratix or Stratix GX Device (Part 8 of 8)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
nCS/CS	I/O	PPA	Input	Chip-select inputs. A low on nCS and a high on CS select the target device for configuration. The nCS and CS pins must be held active during configuration and initialization. During the PPA configuration mode, it is only required to use either the nCS or CS pin. Therefore, if only one chip-select input is used, the other must be tied to the active state. For example, nCS can be tied to GND while CS is toggled to control configuration.In non-PPA schemes, it functions as a user I/O during configuration, which means it is tri-stated. After PPA configuration, nCS and CS are available as a user I/Os and the state of these pins depends on the **Dual-Purpose Pin** settings.
RUnLU	N/A if using Remote Configuration; I/O if not	Remote Configuration in FPP, PS or PPA	Input	Input that selects between remote update and local update. A logic high (1.5-V, 1.8-V, 2.5-V, 3.3-V) selects remote update and a logic low selects local update. When not using remote update or local update configuration modes, this pins is available as general-purpose user I/O pin.
PGM[2..0]	N/A if using Remote Configuration; I/O if not using	Remote Configuration in FPP, PS or PPA	Input	These output pins select one of eight pages in the memory (either flash or enhanced configuration device) when using a remote configuration mode. When not using remote update or local update configuration modes, these pins are available as general-purpose user I/O pins.

Table 11–16 describes the optional configuration pins. If these optional configuration pins are not enabled in the Quartus II software, they are available as general-purpose user I/O pins. Therefore during configuration, these pins function as user I/O pins and are tri-stated with weak pull-ups.

Table 11–16. Optional Configuration Pins

Pin Name	User Mode	Pin Type	Description
CLKUSR	N/A if option is on. I/O if option is off.	Input	Optional user-supplied clock input. Synchronizes the initialization of one or more devices. This pin is enabled by turning on the **Enable user-supplied start-up clock (CLKUSR)** option in the Quartus II software.
INIT_DONE	N/A if option is on. I/O if option is off.	Output open-drain	Status pin. Can be used to indicate when the device has initialized and is in user mode. When nCONFIG is low and during the beginning of configuration, the INIT_DONE pin is tri-stated and pulled high due to an external 10-kΩ pull-up. Once the option bit to enable INIT_DONE is programmed into the device (during the first frame of configuration data), the INIT_DONE pin goes low. When initialization is complete, the INIT_DONE pin is released and pulled high and the FPGA enters user mode. Thus, the monitoring circuitry must be able to detect a low-to-high transition. This pin is enabled by turning on the **Enable INIT_DONE output** option in the Quartus II software.
DEV_OE	N/A if option is on. I/O if option is off.	Input	Optional pin that allows the user to override all tri-states on the device. When this pin is driven low, all I/Os are tri-stated. When this pin is driven high, all I/Os behave as programmed. This pin is enabled by turning on the **Enable device-wide output enable (DEV_OE)** option in the Quartus II software.
DEV_CLRn	N/A if option is on. I/O if option is off.	Input	Optional pin that allows you to override all clears on all device registers. When this pin is driven low, all registers are cleared. When this pin is driven high, all registers behave as programmed. This pin is enabled by turning on the **Enable device-wide reset (DEV_CLRn)** option in the Quartus II software.

Table 11–17 describes the dedicated JTAG pins. JTAG pins must be kept stable before and during configuration to prevent accidental loading of JTAG instructions. If you plan to use the SignalTap II Embedded Logic Analyzer, you will need to connect the JTAG pins of your device to a JTAG header on your board.

Table 11–17. Dedicated JTAG pins

Pin Name	User Mode	Pin Type	Description
TDI	N/A	Input	Serial input pin for instructions as well as test and programming data. Data is shifted in on the rising edge of TCK. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V_{CC}. This pin uses Schmitt trigger input buffers.
TDO	N/A	Output	Serial data output pin for instructions as well as test and programming data. Data is shifted out on the falling edge of TCK. The pin is tri-stated if data is not being shifted out of the device. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by leaving this pin unconnected.
TMS	N/A	Input	Input pin that provides the control signal to determine the transitions of the TAP controller state machine. Transitions within the state machine occur on the rising edge of TCK. Therefore, TMS must be set up before the rising edge of TCK. TMS is evaluated on the rising edge of TCK. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V_{CC}. This pin uses Schmitt trigger input buffers.
TCK	N/A	Input	The clock input to the BST circuitry. Some operations occur at the rising edge, while others occur at the falling edge. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to GND. This pin uses Schmitt trigger input buffers.
TRST	N/A	Input	Active-low input to asynchronously reset the boundary-scan circuit. The TRST pin is optional according to IEEE Std. 1149.1. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to GND. This pin uses Schmitt trigger input buffers.

12. Remote System Configuration with Stratix & Stratix GX Devices

S52015-3.1

Introduction

Altera® Stratix® and Stratix GX devices are the first programmable logic devices (PLDs) featuring dedicated support for remote system configuration. Using remote system configuration, a Stratix or Stratix GX device can receive new configuration data from a remote source, update the flash memory content (through enhanced configuration devices or any other storage device), and then reconfigure itself with the new data.

Like all Altera SRAM-based devices, Stratix and Stratix GX devices support standard configuration modes such as passive serial (PS), fast passive parallel (FPP), and passive parallel asynchronous (PPA). You can use the standard configuration modes with remote system configuration.

This chapter discusses remote system configuration of Stratix and Stratix GX devices, and how to interface them with enhanced configuration devices to enable this capability. This document also explains some related remote system configuration topics, such as the watchdog timer, remote system configuration registers, and factory or application configurations files. The Quartus® II software (version 2.1 and later) supports remote system configuration.

Remote Configuration Operation

Remote system configuration has three major parts:

- The Stratix or Stratix GX device receives updated or new data from a remote source over a network (or through any other source that can transfer data). You can implement a Nios™ (16-bit ISA) or Nios® II (32-bit ISA) embedded processor within either a Stratix or Stratix GX device or an external processor to control the read and write functions of configuration files from the remote source to the memory device.
- The new or updated information is stored into the memory device, which can be an enhanced configuration device, industry-standard flash memory device, or any other storage device (see Figure 12–2).
- The Stratix or Stratix GX device updates itself with the new data from the memory.

Figure 12–1 shows the concept of remote system configuration in Stratix and Stratix GX devices.

Figure 12–1. Remote System Configuration with Stratix & Stratix GX Devices

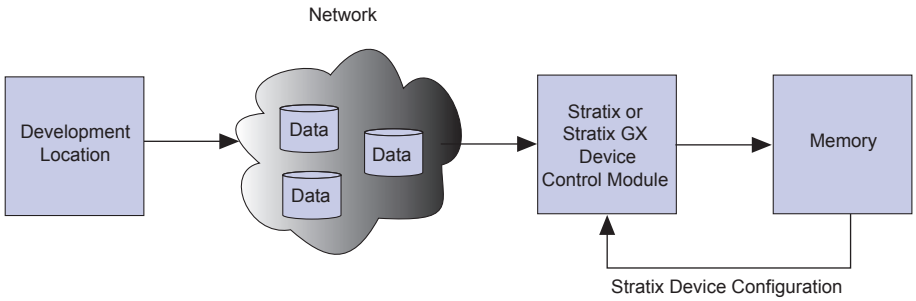
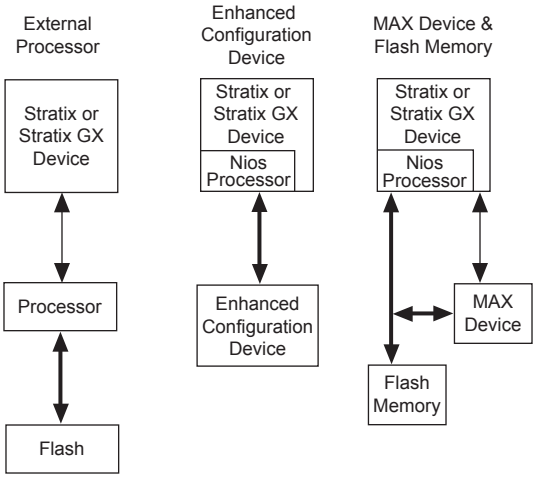


Figure 12–2. Different Options for Remote System Configuration



Remote System Configuration Modes

Stratix and Stratix GX device remote system configuration has two modes: remote configuration mode and local configuration mode. Table 12–1 shows the pin selection settings for each configuration mode.

Table 12–1. Standard, Remote & Local Configuration Options Note (1)

RUnLU (2)	MSEL[2] (3)	MSEL[1..0]	System Configuration Mode	Configuration Mode
–	0	00	Standard	FPP
–	0	01	Standard	PPA
–	0	10	Standard	PS
1	1	00	Remote	FPP
1	1	01	Remote	PPA
1	1	10	Remote	PS
0	1	00	Local	FPP
0	1	01	Local	PPA
0	1	10	Local	PS

Notes to Table 12–1:

(1) For detailed information on standard PS, FPP, and PPA models, see the *Configuring Stratix & Stratix GX Devices* chapter of the *Stratix Device Handbook, Volume 2*.

(2) In Stratix and Stratix GX devices, the RUnLU (remote update / local update) pin, selects between local or remote configuration mode.

(3) The MSEL[2] select mode selects between standard or remote system configuration mode.

Remote Configuration Mode

Using remote configuration mode, you can manage up to seven different application configurations for Stratix and Stratix GX devices. The seven-configuration-file limit is due to the number of pages that the PGM[] pins in the Stratix or Stratix GX device and enhanced configuration devices can select.

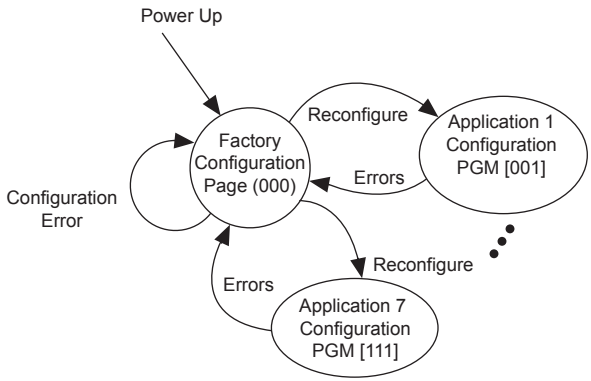
1 If more than seven files are sent to a system using remote configuration mode, previous files are overwritten.

Stratix and Stratix GX devices support remote configuration mode for PS, FPP, and PPA modes. Specify remote configuration mode by setting the MSEL2 and RUnLU pins to high. (See Table 12–1).

On power-up in remote configuration mode, the Stratix or Stratix GX device loads the user-specified factory configuration file, located in the default page address 000 in the enhanced configuration device. After the device configures, the remote configuration control register points to the

page address of the application configuration that should be loaded into the Stratix or Stratix GX device. If an error occurs during user mode of an application configuration, the device reloads the default factory configuration page. Figure 12–3 shows a diagram of remote configuration mode.

Figure 12–3. Remote Configuration Mode



Local Configuration Mode

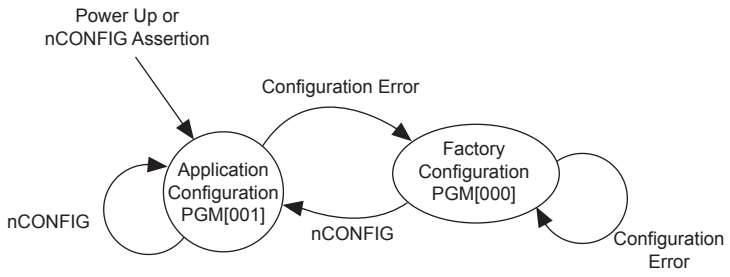
Local configuration mode—a simplified version of remote configuration mode—is suitable for systems that load an application immediately upon power-up. In this mode you can only use one application configuration, which you can update either remotely or locally.

In local configuration mode, upon power-up, or when `nCONFIG` is asserted, the Stratix or Stratix GX device loads the application configuration immediately. Factory configuration loads only if an error occurs during the application configuration's user mode. If you use an enhanced configuration device, page address `001` is the location for the application configuration data, and page address `000` is the location for the factory configuration data.

If the configuration data at page address `001` does not load correctly due to cyclic redundancy code (CRC) failure, or it times-out of the enhanced configuration device, or the external processor times-out, then the factory configuration located at the default page (page address `000`) loads into the Stratix or Stratix GX device.

In local configuration mode (shown in Figure 12–4), the user watchdog timer is disabled. For more information on the watchdog timer, see "Watchdog Timer" on page 12–7.

Figure 12–4. Local Configuration Mode



In local configuration mode, one application configuration is available to the device. For remote or local configuration mode selection, see Table 12–1.

Remote System Configuration Components

The following components are used in Stratix and Stratix GX devices to support remote and local configuration modes:

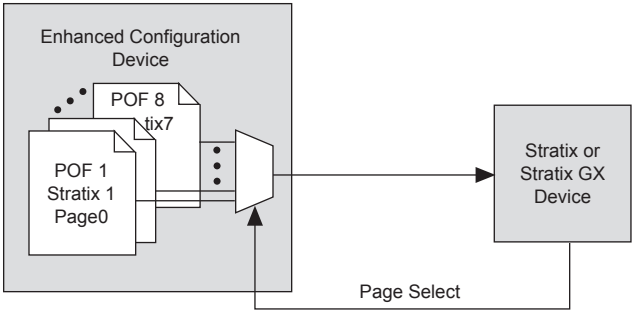
- Page mode feature
- Factory configuration
- Application configuration
- Watchdog timer
- Remote update sub-block
- Remote configuration registers

A description of each component follows.

Page Mode Feature

The page mode feature enables Stratix and Stratix GX devices to select a location to read back data for configuration. The enhanced configuration device can receive and store up to eight different configuration files (one factory and seven application files). Selection of pages to read from is performed through the `PGM[2..0]` pins on the Stratix or Stratix GX device and enhanced configuration devices. These pins in the Stratix or Stratix GX device can be designated user I/O pins during standard configuration mode, but in remote system configuration mode, they are dedicated output pins. Figure 12–5 shows the page mode feature in Stratix or Stratix GX devices and enhanced configuration devices.

Figure 12–5. Page Mode Feature in Stratix or Stratix GX Devices & Enhanced Configuration Devices



Upon power-up in remote configuration mode, the factory configuration (see description below) selects the user-specified page address through the Stratix or Stratix GX `PGM[2..0]` output pins. These pins drive the `PGM[2..0]` input pins of the enhanced configuration device and select the requested page in the memory.

If an intelligent host is used instead of an enhanced configuration device, you should create logic in the intelligent host to support page mode settings similar to that in enhanced configuration devices.

Factory Configuration

Factory configuration is the default configuration data setup. In enhanced configuration devices, this default page address is `000`. Factory configuration data is written into the memory device only once by the system manufacturer and should not be remotely updated or altered. In remote configuration mode, the factory configuration loads into the Stratix or Stratix GX device upon power-up.

The factory configuration specifications are as follows:

- Receives new configuration data and writes it to the enhanced configuration or other memory devices
- Determines the page address for the next application configuration that should be loaded to the Stratix or Stratix GX device
- Upon an error in the application configuration, the system reverts to the factory configuration
- Determines the reason for any application configuration error
- Determines whether to enable or disable the user watchdog timer for application configurations

- Determines the user watchdog timer's settings if the timer is enabled (remote configuration mode)
- If the user watchdog timer is not reset after a predetermined amount of time, it times-out and the system loads the factory configuration data back to the Stratix or Stratix GX device

If a system encounters an error while loading application configuration data, or if the device re-configures due to `nCONFIG` assertion, the Stratix or Stratix GX device loads the factory configuration. The remote system configuration register determines the reason for factory re-configuration. Based on this information, the factory configuration determines which application configuration needs to be loaded.

Application Configuration

The application configuration is the configuration data received from the remote source and updated into different locations or pages of the memory storage device (excluding the factory default page).

Watchdog Timer

A watchdog timer is a circuit that determines whether another mechanism functions properly. The watchdog timer functions like a time-delay relay that remains in the reset state while an application runs properly. This action periodically sends a reset command from the working application to the watchdog timer. Stratix and Stratix GX devices are equipped with a built-in watchdog timer for remote system configuration.

A user watchdog timer prevents a faulty application configuration from indefinitely stalling the Stratix or Stratix GX device. The timer functions as a counter that counts down from an initial value, which is loaded into the device from the factory configuration. This is a 29-bit counter, but you use only the upper 12 bits to set the value for the watchdog timer. You specify the counter value according to your design needs.

The timer begins counting once the Stratix or Stratix GX device goes into user mode. If the application configuration does not reset the user watchdog timer after the specified time, the timer times-out. At this point, the Stratix or Stratix GX device is re-configured by loading the factory configuration and resetting the user watchdog timer.

1 The watchdog timer is disabled in local configuration mode.

Remote Update Sub-Block

The remote update sub-block is responsible for administrating the remote configuration feature. This sub-block, which is controlled by a remote configuration state machine, generates the control signals required to control different remote configuration registers.

Remote Configuration Registers

Remote configuration registers are a series of registers required to keep track of page addresses and the cause of configuration errors. Table 12–2 gives descriptions of the registers' functions. You can control both the update and shift registers; the status and control registers are controlled by internal logic, but can be read via the shift register.

Table 12–2. Remote Configuration Registers

Register	Description
Control register	This register contains the current page address, the watchdog timer setting, and one bit specifying if the current configuration is a factory or application configuration. During a capture in an application configuration, this register is read into the shift register.
Update register	This register contains the same data as the control register, except that it is updated by the factory configuration. The factory configuration updates the register with the values to be used in the control register on the next re-configuration. During capture in a factory configuration, this register is read into the shift register.
Shift register	This register is accessible by the core logic and allows the update, status, and control registers to be written and sampled by the user logic. The update register can only be updated by the factory configuration in remote configuration mode.
Status register	This register is written into by the remote configuration block on every re-configuration to record the cause of the re-configuration. This information is used by factory configuration to determine the appropriate action following a re-configuration.

Figure 12–6 shows the control, update, shift, and status registers and the data path used to control remote system configuration.

Figure 12–6. Remote Configuration Registers & Related Data Path

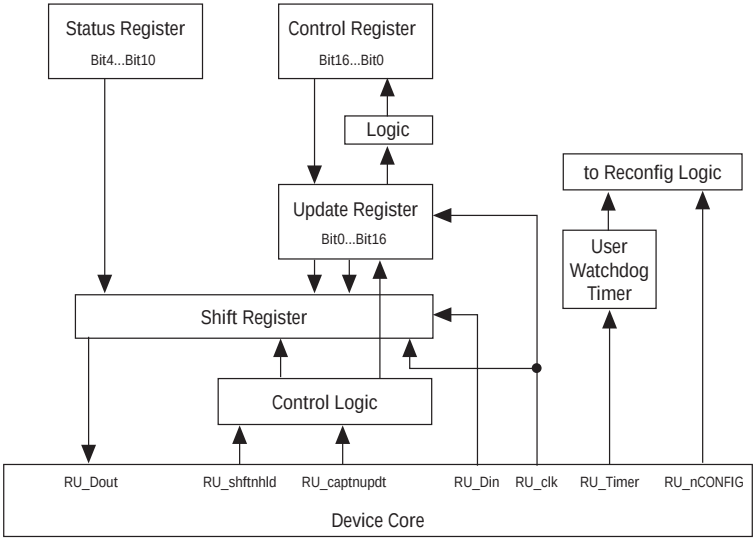


Table 12–3 describes the user configuration signals that are driven to/from the device logic array. The remote configuration logic has one input signal to the device logic array and six output signals from the device logic array.

Table 12–3. User Configuration Signals To/From Device Core (Part 1 of 2)

Signal Name	To/From Device Core	Description
`RU_Timer`	Output from the core to the remote update block	Request from the application to reset the user watchdog timer with its initial count. A falling edge of this signal triggers a reset of the user watchdog timer.
`RU_nCONFIG`	Output from the core to the remote update block	When driven low, this signal triggers the device to reconfigure. If requested by the factory configuration, the application configuration specified in the remote update control register is loaded. If requested by the application configuration, the factory configuration is loaded.
`RU_Clk`	Output from the core to the remote update block	Clocks the remote configuration shift register so that the contents of the status and control registers can be read out, and the contents of update register can be loaded. The shift register latches data on the rising edge of the `RU_Clk`.

Table 12–3. User Configuration Signals To/From Device Core (Part 2 of 2)

Signal Name	To/From Device Core	Description
RU_shftnhld	Output from the core to the remote update block	If its value is “1”, the remote configuration shift register shifts data on the rising edge of RU_Clk. It its value is “0” and RU_captnupdt is “0”, the shift register updates the update register. If its value is “0”, and RU_captnupdt is “1”, the shift register captures the status register and either the control or update register (depending on whether the configuration is factory or application).
RU_captnupdt	Output from the core to the remote update block	When RU_captnupdt is at value “1” and RU_shftnhld is at value”0”, the system specifies that the remote configuration shift register should be written with the content of the status register and either the update register (in a factory configuration) or the control register (in an application configuration). This shift register is loaded on the rising edge of RU_Clk. When RU_captnupdt is at value “0” and RU_shftnhld is at value”0”, the system specifies that the remote configuration update register should be written with the content of the shift register in a factory configuration. The update register is loaded on the rising edge of RU_Clk. This pin is enabled only for factory configuration in remote configuration mode (it is disabled for the application configuration in remote configuration or for local configuration modes). If RU_shftnhld is at value “1”, RU_captnupdt has no function.
RU_Din	Output from the core to the remote update block	Data to be written into the remote configuration shift register on the rising edge of RU_Clk. To load into the shift register, RU_shftnhld must be asserted.
RU_Dout	Input to the core from the remote update block	Output of the remote configuration shift register to be read by core logic. New data arrives on each rising edge of RU_Clk.

All of the seven device core signals (see Figure 12–6), are enabled for both remote and local configuration for both factory and application configuration, except RU_Timer and RU_captnupdt. Figure 12–7 and Table 12–4 specify the content of control register upon power-on reset (POR).

The difference between local configuration and remote configuration is how the control register is updated during a re-configuration and which core signals are enabled.

Figure 12–7. Remote System Configuration Control Register

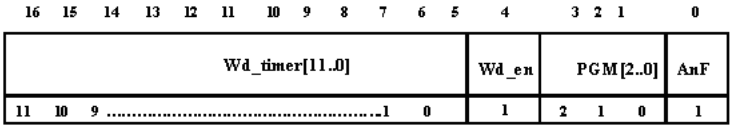


Table 12–4 shows the content of the control register upon POR.

Table 12–4. Control Register Contents

Parameter	Definition	POR Reset Value	Comment
AnF	Current configuration is factory or applications	1 bit ‘1’	Applications
		1 bit ‘0’	Factory
PGM[2..0]	Page mode selection	3 bits ‘001’	Local configuration
		3 bits ‘000’	Remote configuration
Wd_en	User watchdog timer enable	1 bit ‘0’	–
Wd_timer [11..0]	User watchdog timer time-out value	12 bits ‘0’	High order bits of 29 bit counter

The status register specifies the reason why re-configuration has occurred and determines if the re-configuration was due to a CRC error, nSTATUS pulled low due to an error, the device core caused an error, nCONFIG was reset, or the watchdog timer timed-out. Figure 12–8 and Table 12–5 specify the content of the status register.

Figure 12–8. Remote System Configuration Status Register

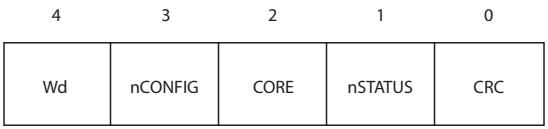


Table 12–5 shows the content of the status register upon POR.

Table 12–5. Status Register Contents

Parameter	Definition	POR Reset Value
CRC (from configuration)	CRC caused re-configuration	1 bit ‘0’
nSTATUS	nSTATUS caused re-configuration	1 bit ‘0’
CORE *(1)*	Device core caused re-configuration	1 bit ‘0’
nCONFIG	NCONFIG caused re-configuration	1 bit ‘0’
Wd	Watchdog Timer caused re-configuration	1 bit ‘0’

Note to Table 12–5:

(1) Core re-configuration enforces the system to load the application configuration data into the Stratix or Stratix GX device. This occurs after factory configuration specifies the appropriate application configuration data.

Quartus II Software Support

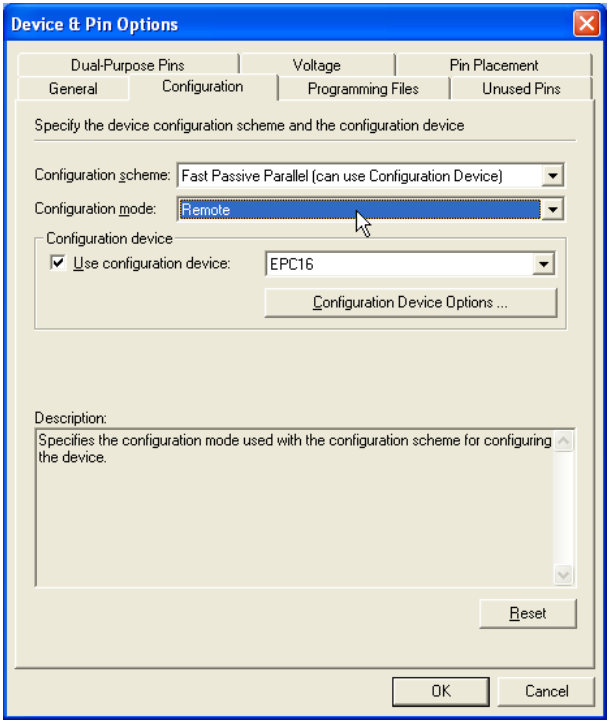
The Quartus II software supports implementation of both remote and local configuration modes in your Stratix or Stratix II device. To include the remote or local configuration feature to your design, select remote or local as the configuration mode under the **Device & Pin Options** compiler settings (prior to compilation). This selection reserves the dual-purpose RUnLU and PGM[2:0] pins for use as dedicated inputs in remote / local configuration modes.

To set the configuration mode as remote or local, follow these steps (See Figure 12–9):

1. Open the **Device & Pin Options** settings window under the **Assignments** menu.
2. Select **Device & Pin Options** dialog box. The **Device & Pin Options** dialog box is displayed.
3. Click the **Configuration** tab.
4. In the **Configuration mode** list, select **Remote** or **Local**.

The Standard mode selection disables the remote system configuration feature. In addition to the mode selection, you can specify the configuration scheme and configuration device (if any) used by your setup.

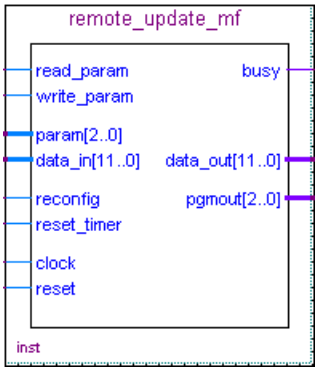
Figure 12–9. Device & Pin Options Dialog Box



Additionally, the remote configuration mode requires you to either instantiate the `altremote_update` megafunction or the WYSIWYG (what-you-see-is-what-you-get) atom into your design. Without this atom or megafunction, you are not be able to access the dedicated remote configuration circuitry or registers within the Stratix or Stratix GX device. See Figure 12–10 for a symbol of the `altremote_update` megafunction.

The local configuration mode, however, can be enabled with only the device **Configuration Options** compiler setting.

Figure 12–10. altremote_update Megafunction Symbol



altremote_update Megafunction

A remote update megafunction, `altremote_update`, is provided in the Quartus II software to provide a memory-like interface to allow for easy control of the remote update parameters. Tables 12–6 and 12–7 describe the input and output ports available on the `altremote_update` megafunction. Table 12–8 shows the `param[2..0]` bit settings.

Table 12–6. Input Ports of the altremote_update Megafunction (Part 1 of 2)

Port Name	Required	Source	Description
clock	Y	Logic Array	Clock input to the `altremote_update` block. All operations are performed with respects to the rising edge of this clock.
reset	Y	Logic Array	Asynchronous reset, which is used to initialize the remote update block. To ensure proper operation, the remote update block must be reset before first accessing the remote update block. This signal is not affected by the busy signal and will reset the remote update block even if busy is logic high. This means that if the reset signal is driven logic high during writing of a parameter, the parameter will not be properly written to the remote update block.
reconfig	Y	Logic Array	When driven logic high, reconfiguration of the device is initiated using the current parameter settings in the remote update block. If busy is asserted, this signal is ignored. This is to ensure all parameters are completely written before reconfiguration begins.
reset_timer	N	Logic Array	This signal is required if you are using the watchdog timer feature. A logic high resets the internal watchdog timer. This signal is not affected by the busy signal and can reset the timer even when the remote update block is busy. If this port is left connected, the default value is 0.

Table 12–6. Input Ports of the altremote_update Megafunction (Part 2 of 2)

Port Name	Required	Source	Description
read_param	N	Logic Array	Once `read_param` is sampled as a logic high, the busy signal is asserted. While the parameter is being read, the busy signal remains asserted, and inputs on `param[]` are ignored. Once the busy signal is deactivated, the next parameter can be read. If this port is left unconnected, the default value is 0.
write_param	N	Logic Array	This signal is required if you intend on writing parameters to the remote update block. When driven logic high, the parameter specified on the `param[]` port should be written to the remote update block with the value on `data_in[]`. The number of valid bits on `data_in[]` is dependent on the parameter type. This signal is sampled on the rising edge of clock and should only be asserted for one clock cycle to prevent the parameter from being re-read on subsequent clock cycles. Once `write_param` is sampled as a logic high, the busy signal is asserted. While the parameter is being written, the busy signal remains asserted, and inputs on `param[]` and `data_in[]` are ignored. Once the busy signal is deactivated, the next parameter can be written. This signal is only valid when the `Current_Configuration` parameter is factory since parameters cannot be written in application configurations. If this port is left unconnected, the default value is 0.
param[2..0]	N	Logic Array	3-bit bus that selects which parameter should be read or written. If this port is left unconnected, the default value is 0.
data_in[11..0]	N	Logic Array	This signal is required if you intend on writing parameters to the remote update block 12-bit bus used when writing parameters, which specifies the parameter value. The parameter value is requested using the `param[]` input and by driving the `write_param` signal logic high, at which point the busy signal goes logic high and the value of the parameter is captured from this bus. For some parameters, not all 12-bits will be used in which case only the least significant bits will be used. This port is ignored if the `Current_Configuration` parameter is set to an application configuration since writing of parameters is only allowed in the factory configuration. If this port is left unconnected, the default values is 0.

Note to Table 12–6:

(1) Logic array source means that you can drive the port from internal logic or any general-purpose I/O pin.

Table 12–7. Output Ports of the altremote_update Megafunction

Port Name	Required	Destination	Description
busy	Y	Logic Array	When this signal is a logic high, the remote update block is busy either reading or writing a parameter. When the remote update block is busy, it ignores its `data_in[]`, `param[]`, and `reconfig` inputs. This signal will go high when `read_param` or `write_param` is asserted and will remain asserted until the operation is complete.
pgm_out[2..0]	Y	PGM[2..0] pins	3-bit bus that specifies the page pointer of the configuration data to be loaded when the device is reconfigured. This port must be connected to the `PGM[]` output pins, which should be connected to the external configuration device
data_out[11..0]	N	Logic Array	12-bit bus used when reading parameters, which reads out the parameter value. The parameter value is requested using the `param[]` input and by driving the `read_param` signal logic high, at which point the busy signal will go logic high. When the busy signal goes low, the value of the parameter will be driven out on this bus. The `data_out[]` port is only valid after a read_param has been issued and once the busy signal is de-asserted. At any other time, its output values are invalid. For example, even though the `data_out[]` port may toggle during a writing of a parameter, these values are not a valid representation of what was actually written to the remote update block. For some parameters, not all 12-bits will be used in which case only the least significant bits will be used.

Note to Table 12–7:

(1) Logic array destination means that you can drive the port to internal logic or any general-purpose I/O pin.

Table 12–8. Parameter Settings for the altremote_update Megafunction (Part 1 of 2)

Selected Parameter	param[2..0] bit setting	width of parameter value	POR Reset Value	Description
Status Register Contents	000	5	5 bit '0	Specifies the reason for re-configuration, which could be caused by a CRC error during configuration, `nSTATUS` being pulled low due to an error, the device core caused an error, `nCONFIG` pulled low, or the watchdog timer timed-out. This parameter can only be read.
Watchdog Timeout Value	010	12	12 bits '0	User watchdog timer time-out value. Writing of this parameter is only allowed when in the factory configuration.
Watchdog Enable	011	1	1 bit '0	User watchdog timer enable. Writing of this parameter is only allowed when in the factory configuration

Table 12–8. Parameter Settings for the altremote_update Megafunction (Part 2 of 2)

Selected Parameter	param[2..0] bit setting	width of parameter value	POR Reset Value	Description
Page select	100	3	3 bit '001' - Local configuration 3 bit '000' - Remote configuration	Page mode selection. Writing of this parameter is only allowed when in the factory configuration.
Current configuration (AnF)	101	1	1 bit '0' - Factory 1 bit '1' - Application	Specifies whether the current configuration is factory or and application configuration. This parameter can only be read.
Illegal values	001			
	110			
	111			

Remote Update WYSIWYG ATOM

An alternative to using the `altremote_update` megafunction is to directly instantiate the remote update WYSIWYG atom. This atom should be included in the factory configuration and any application configuration image to access the remote configuration shift registers.

When implementing the atom, you should consider following:

1. Only one atom can be used in the circuit; more than one gives a no-fit.
2. All signals for the cell must be connected. The clock port (`CLK`) must be connected to a live cell. The others can be constant V_{CC} or GND.
3. The `pgmout` port must be connected and must feed `PGM[2.0]` output pins (it cannot be connected to anything else but output pins).
4. The Quartus II software reserves `RUnLU` as an input pin, and you must connect it to V_{CC}.

The Stratix and Stratix GX remote update atom ports are:

Stratix_rublock *<rublock_name>*
(
.clk(*<clock source>***),**
.shiftnld(*<shiftnld source>***),**
.captnupdt(*<shiftnld source>***),**
.regin(<*regin input source from the core>***),**
.rsttimer(*<input signal to reset the watchdog timer>***),**
.config(*<input signal to initiate configuration>***),**
.regout(*<data output destination to core>***),**
.pgmout(<*program output destinations to pins>***)**

Table 12–9 shows the remote update block input and output port names and descriptions.

Table 12–9. Remote Update Block Input & Output Ports

Ports	Definition
<rublock_name>	The unique identifier for the instance. This identifier name can be anything as long as it is legal for the given description language (i.e., Verilog, VHDL, AHDL, etc.). This field is required.
.clk(*<clock source>*)	Designates the clock input of this cell. All operation is with respect to the rising edge of this clock. This field is required.
.shiftnld(*<shiftnld source>*)	An input into the remote configuration block. When **.shiftnld** = `1`, the data shifts from the internal shift registers to the `regout` port at each rising edge of `clk`, and the data also shifts into the internal shift registers from `regin` port. This field is required.
.captnupdt(*<shiftnld source>*)	An input into the remote configuration block. This controls the protocol of when to read the configuration mode or when to write into the registers that control the configuration. This field is required.
.regin(*<regin input source from the core>*)	An input into the configuration block for all data loading into the core. The data shifts into the internal registers at the rising edge of `clk`. This field is required.
.rsttimer(*<input signal to reset the watchdog timer>*)	An input into the watchdog timer of the remote update block. When this is high, it resets the watchdog timer. This field is required.
.config(*<input signal to initiate configuration>*)	An input into the configuration section of the remote update block. When this signal goes high, the part initiates a re-configuration. This field is required.
.regout(*<data output destination to core>*)	A 1-bit output, which is the output of the internal shift register, and updated every rising edge of `clk`. The data coming out depends on the control signals. This field is required.
.pgmout(*<program output destinations to pins>*)	A 3-bit bus. It should always be connected only to output pins (not `bidir` pins). This bus gives the page address (`000` to `111`) of the configuration data to be loaded when the device is getting configured. This field is required.

f For more information on the control signals for the remote block, see Table 12–3 on page 12–9.

Using Enhanced Configuration Devices

This section describes remote system configuration of Stratix and Stratix GX devices with the Nios embedded processor using enhanced configuration devices. Enhanced configuration devices are composed of a standard flash memory and a controller. The flash memory stores configuration data, and the controller reads and writes to the flash memory.

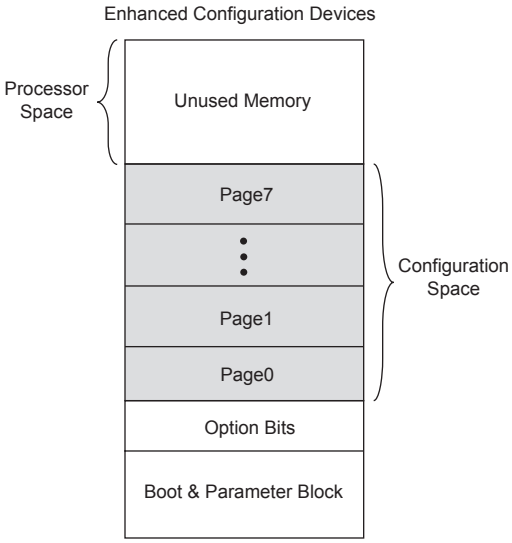
In remote system configuration, only PS and FPP modes are supported using an enhanced configuration device. A Stratix or Stratix GX device running a Nios embedded processor can receive data from a remote source through a network or any other appropriate media. A specific page of the enhanced configuration device stores the received data.

This scheme uses the page mode option in Stratix and Strati GX devices. Up to eight pages can be stored in each enhanced configuration device, each of which can store a configuration file.

In enhanced configuration devices, a page is a section of the flash memory space. Its boundary is determined by the Quartus II software (the page size is programmable). In the software, you can specify which configuration file should be stored in which page within the flash memory. To access the configuration file on each page, set the three input pins (`PGM[2..0]`), which provide access to all eight pages. Because the `PGM[2..0]` pins of an enhanced configuration device connect to the same pins of the Stratix or Stratix GX device, the Stratix or Stratix GX device selects one of the eight memory pages as a target location to read from. Figure 12–11 shows the allocation of different pages in the enhanced configuration device.

f For more information on enhanced configuration devices, see the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet* and the *Altera Enhanced Configuration Devices* chapter.

Figure 12–11. Memory Map in Enhanced Configuration Device

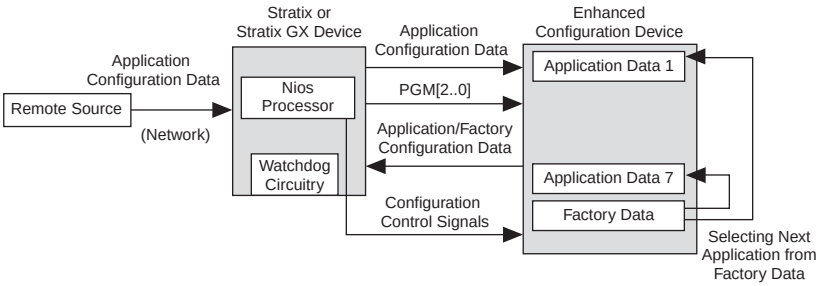


When the Stratix or Stratix GX device powers-up in remote configuration mode, the devices loads configuration data located at page address `000`. You should always load the factory default configuration data at this location and make sure this information is not altered.

The factory configuration contains information to determine the next application configuration to load into the Stratix or Stratix GX device. When the Stratix or Stratix GX device successfully loads the application configuration from the page selected by the `PGM[2..0]` pins, it enters user mode.

In user mode, the Nios embedded processor (or any other logic) assists the Stratix or Stratix GX device in detecting remote system configuration information. In remote system configuration, the Nios embedded processor receives the incoming data from the remote source via the network, writes it to the ECP16 enhanced configuration device, and then initiates loading of the factory configuration into the Stratix or Stratix GX device. Factory configuration reads the remote configuration status register and determines the appropriate application configuration to load into the Stratix or Stratix GX device. Figure 12–12 shows the remote system configuration.

Figure 12–12. Remote System Configuration Using Enhanced Configuration Devices



The user watchdog timer in Stratix and Stratix GX devices ensures that an application configuration has loaded successfully and checks if the application configuration is operating correctly in user mode. The watchdog timer must be continually reset by the user logic. If an error occurs while the application configuration loads, or if the watchdog timer times-out during user mode, the factory configuration is reloaded to prevent the system from halting in an erroneous state. Figure 12–3 on page 12–4 illustrates the remote configuration mode.

Upon power-up in local configuration scheme, the application configuration at page 001 (`PGM[001]` of the enhanced configuration device) loads into the Stratix or Stratix GX device. This application can be remotely or locally updated. If an error occurs during loading of the configuration data, the factory configuration loads automatically (see Figure 12–4 on page 12–5). The rest is identical to remote configuration mode.

Local Update Programming File Generation

This section describes the programming file generation process for performing remote system upgrades. The Quartus II convert programming files (CPF) utility generates the initial and partial programming files for configuration memory within the enhanced configuration devices.

The two pages that local configuration mode uses are a factory configuration stored at page 000, and an application configuration stored at page 001. The factory configuration cannot be updated after initial production programming. However, the application configuration can be erased and reprogrammed after initial system deployment.

In local update mode, you would first create the initial programming file with the factory configuration image and a version of the application configuration. Subsequently, you can generate partial programming files to update the application configuration (stored in page 001). Quartus II CPF can create partial programming files in **.hex** (Hexadecimal file), JAM, **.jbc** (JAM Byte-Code File), and POF formats.

In addition to the two configuration pages, user data or processor code can also be pre-programmed in the bottom boot and main data areas of the enhanced configuration device memory. The CPF utility accepts a HEX input file for the bottom and main data areas, and includes this data in the POF output file. However, this is only supported for initial programming file generation. Partial programming file generation for updating user HEX data is not supported, but can be performed using the enhanced configuration device external flash interface.

Initial Programming File Generation

The initial programming file includes configuration data for both factory and application configuration pages. The enhanced configuration device option's bits are always located between byte addresses `0x00010000` and `0x0001003F`. Also, page 0 always starts at `0x00010040` while its end address is dependent on the size of the factory configuration data.

The two memory allocation options that exist for the application configuration are auto addressing and block addressing. In auto addressing mode, Quartus II automatically allocates memory for the application configuration. All the configuration memory sectors that are not used by the page 0 factory configuration are allocated for page 1. The memory allocated is maximized to allow future versions of the application configuration to grow and have bigger configuration files (when the compression feature is enabled). Processor or user data storage (HEX input file) is only supported by the bottom boot area in auto addressing mode.

The following steps and screen shot (see Figure 12–13) describe initial programming file generation with auto addressing mode.

1. Open the **Convert Programming Files** window from the **File** menu.

2. Select **Programmer Object File (*.pof)** from the drop-down list titled **Programming File Type**.

3. Select the enhanced configuration device used (EPC4, EPC8, EPC16), and the mode used (1-bit Passive Serial or Fast Passive Parallel). Only during the initial programming file generation can you specify the **Options**, **Configuration Device**, or **Mode** settings. While generating the partial programming file, all of these settings are grayed out and inaccessible.

4. In the **Input files to convert** box, highlight **SOF Data** at **Page 0** and click **Add File**. Select input SOF file(s) for this configuration page and insert them.

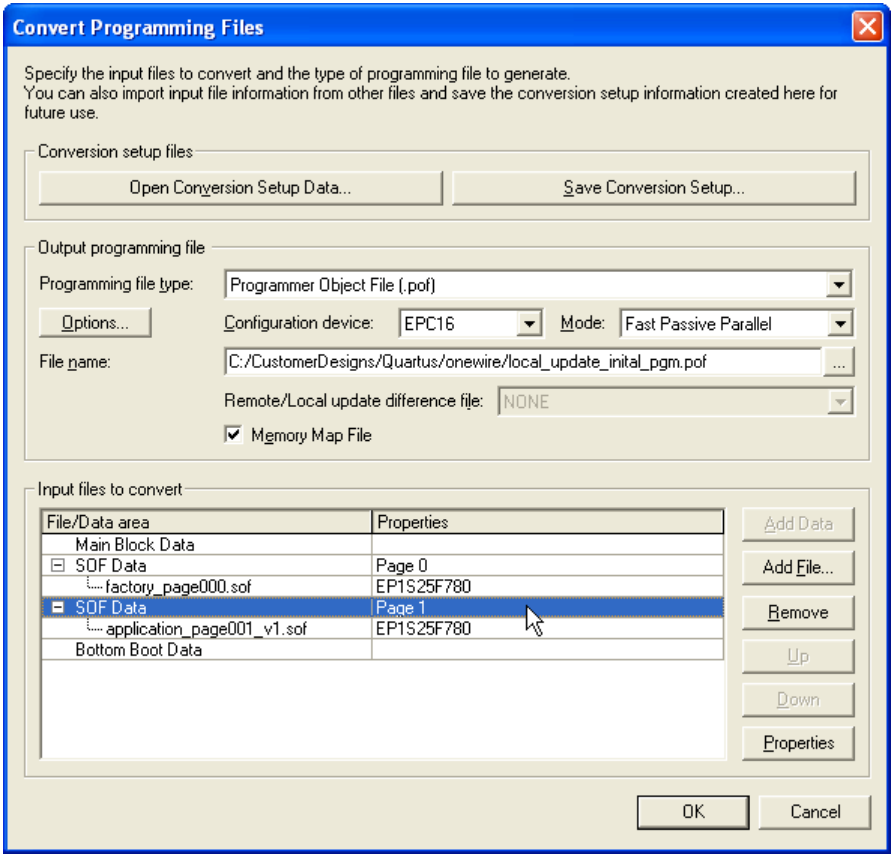
5. Repeat Step 4 for the **Page 1** application configuration page.

6. Check the **Memory Map File** box to generate a memory map output file that specifies the start/end addresses of each configuration page and user data blocks.

7. Save the CPF setup (optionally), by selecting **Save Conversion Setup**... and specifying a name for the **.cof** output file.

8. Click **OK** to generate initial programming and memory map files.

Figure 12–13. CPF Setup for Initial Programming File (Auto Addressing)



A sample memory map output file for the preceding setup is shown below. Configuration option bits and page 0 data occupy main flash sectors 0 through 4. See the *Sharp LHF16J06 Flash memory used in EPC16 devices* Data Sheet at **www.altera.com** to correlate memory addresses to the EPC16 flash sectors. In auto addressing mode, page 1 allocates all unused flash sectors. For this example, this unused area includes main sectors 5 through 30, and all of the bottom boot sectors. While this large portion of memory is allocated for page 1, the real application configuration data is top justified within this region with filler 1'b1 bits in lower memory addresses. Notice that the page 1 configuration data

wraps around the top of the memory and fills up the bottom boot area. The wrap around does not occur if the bottom boot area is used for processor / user HEX data file storage.

Block	Start Address	End Address
OPTION BITS	0x00010000	0x0001003F
PAGE 0	0x00010040	0x00054CC8
PAGE 1	0x001CB372	0x0000FFFD wrapped around

The block addressing mode allows better control of flash memory allocation. You can allocate a specific flash memory region for each application configuration page. This allocation is done by specifying a block starting and block ending address. While selecting the size of the region, you should account for growth in compressed configuration bitstream sizes due to design changes and additions. In local update mode, all configuration data is top justified within this allotted memory. In other words, the last byte of configuration data is stored such that it coincides with the highest byte address location within the allotted space. Lower unused memory address locations within the allotted region are filled with 1's. These filler bits are transmitted during a configuration cycle using page 1, but are ignored by the Stratix device. The memory map output file provides the exact byte address where real configuration data for page 1 begins. Note that any partial update of page 1 should erase all allotted flash sectors before storing new configuration data.

In the block addressing mode, HEX input files can be optionally added to the bottom boot and main flash data areas (one HEX file per area is allowed). The HEX file can be stored with relative addressing or absolute addressing. For more information on relative and absolute addressing, see the *Using Altera Enhanced Configuration Devices* chapter of the *Configuration Handbook*.

Figures 12–14 and 12–15, and the following steps illustrate generating an initial programming file with block addressing for local update mode. This example also illustrates preloading user HEX data into bottom boot and main flash sectors.

1. Open the **Convert Programming Files** window from the **File** menu.

2. Select **Programmer Object File (.pof)** from the drop-down list titled **Programming file type**.

3. Select the enhanced configuration device (EPC4, EPC8, EPC16), and the mode used (1-bit Passive Serial or Fast Passive Parallel). Only during the initial programming file generation can you specify the **Options**, **Configuration device**, or **Mode** settings. While generating the partial programming file, all of these settings are grayed out and inaccessible.

4. In the **Input files to convert** box, highlight **SOF Data** at **Page 0** and click **Add File**. Select input SOF file(s) for this configuration page and insert them.

5. Repeat Step 4 for the Page 1 application configuration page.

6. For enabling block addressing, select the **SOF Data** entry for **Page 1**, and click **Properties**. This opens the **SOF Data Properties** dialog box (see Figure 12–15).

7. Pick **Block** from the **Address Mode** drop down selection, and enter 32-bit Hexadecimal byte address for block **Starting Address** and **Ending Address**. Note that for partial programming support, the block start and end addresses should be aligned to a flash sector boundary. This prevents two configuration pages from overlapping within the same flash boundary. See the flash memory datasheet for data sector boundary information. Click **OK** to save SOF data properties.

8. Check the **Memory Map File** box to generate a memory map output file that specifies the start/end addresses of each configuration page and user data blocks.

9. Save the CPF setup (optionally), by selecting **Save Conversion Setup**… and specifying a name for the COF output file.

10. Click **OK** to generate initial programming and memory map files.

Figure 12–14. CPF Setup for Initial Programming File Generation (Block Addressing)

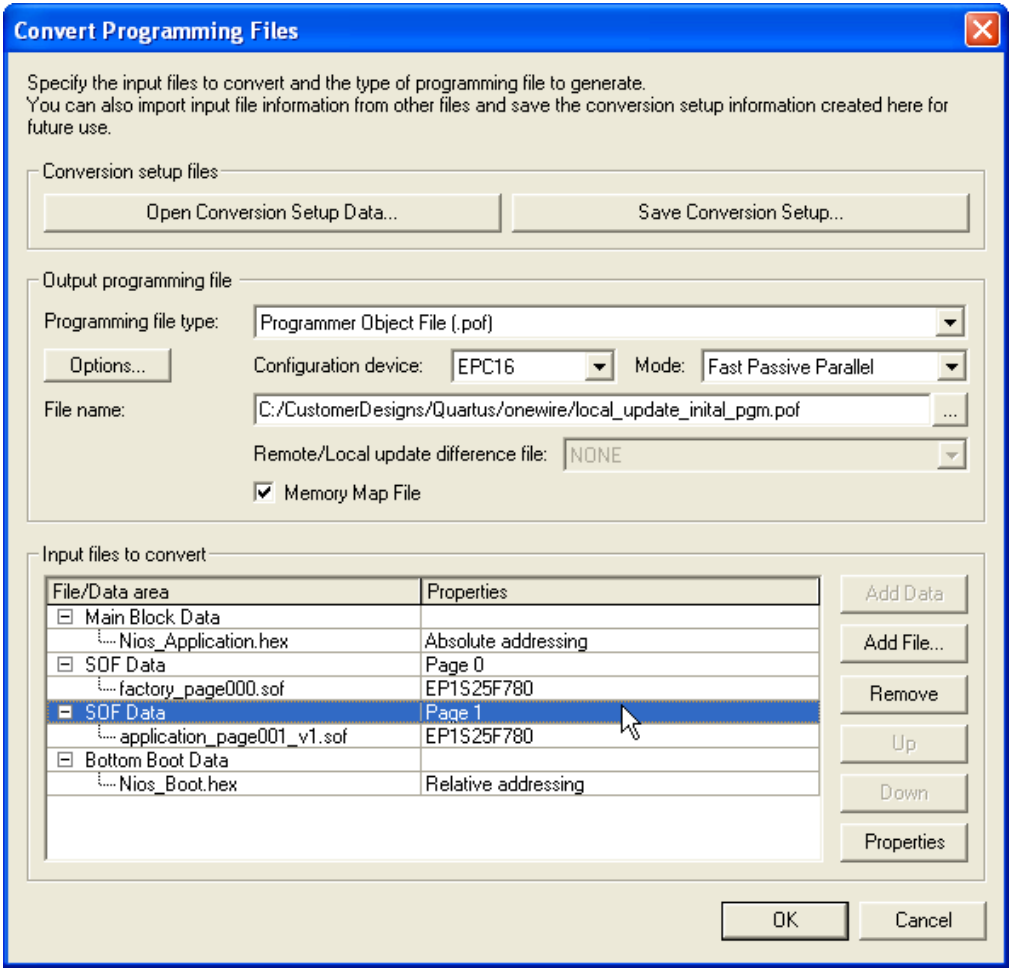
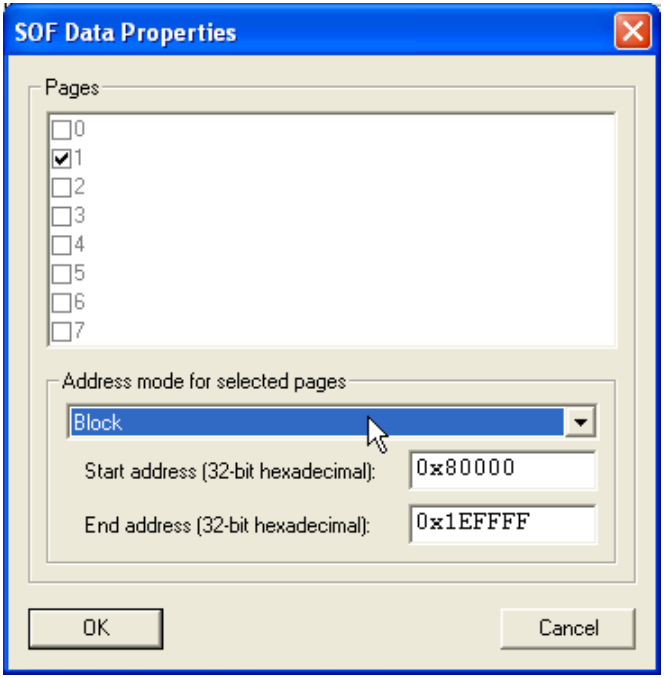


Figure 12–15. Specifying Block Addresses for Application Configuration



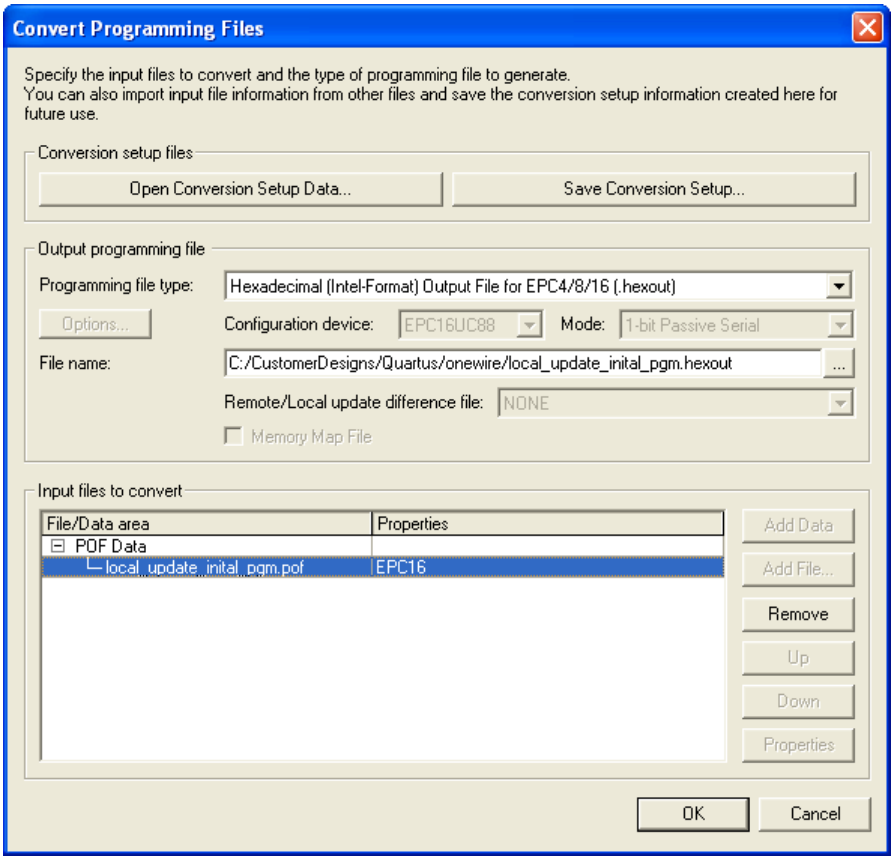
A sample memory map output file for the preceding example is shown below. Note that the allocated memory for page 1 is between `0x00080000` and `0x001EFFFF`, while the actual region used by the current application configuration bitstream is between `0x001AB36C` and `0x001EFFF7`. The configuration data is top justified within the allocated SOF data region.

Block	Start Address	End Address
BOTTOM BOOT	0x00000000	0x000001FF
OPTION BITS	0x00010000	0x0001003F
PAGE 0	0x00010040	0x00054CC8
PAGE 1	0x001AB36C	0x001EFFF7
TOP BOOT/MAIN	0x001F0000	0x001F01FF

Also note that the HEX data stored in the main data area uses absolute addressing. If relative addressing were to be used, the main data contents would be justified with the top (higher address locations) of the memory.

The initial programming file (POF) can be converted to an Intel Hexadecimal format file (*.HEXOUT) using the Quartus II CPF utility. See Figure 12–16.

Figure 12–16. Converting POF Programming File to Intel HEX Format



Partial Programming File Generation

The enhanced Quartus II CPF utility allows an existing application configuration page to be replaced with new data. Partial programming files are generated to perform such configuration data updates.

In order to generate a partial programming file, you have to input the initial programming file (POF) and new configuration data (SOF) to the Quartus II CPF utility. In addition, you have to specify the addressing mode (auto or manual) that was used during initial POF creation. And if

block addressing was used, you should specify the block start and end addresses. With this information, Quartus II ensures that the partial programming file only updates the flash region containing the application configuration. The factory configuration (page 0) and configuration option bits are left unaltered during this process.

Figure 12–17 and the following steps illustrate generation of a partial programming file:

1. Open the **Convert Programming Files** window from the **File** menu.
2. Select **Programmer Object File for Local Update (.pof)** from the drop-down list titled **Programming file type**, and specify an output **File name**.
3. In the **Input files to convert box**, highlight **POF Data** and click **Add File**. Select the initial programming POF file for this design and insert it.
4. In the **Input files to convert box**, highlight **SOF Data** and click **Add File.** Select the new application configuration bitstream (SOF) and insert it.
5. When using block addressing, select the **SOF Data** entry for **Page 1**, and click **Properties**. This opens the **SOF Data Properties** dialog box (see Figure 12–18).
6. Pick **Block** from the **Address Mode** drop down selection, and enter 32-bit Hexadecimal byte address for block **Starting Address** and **Ending Address**. These addresses should be identical to those used to generate the initial programming file. Click **OK** to save SOF data properties.
7. Check the **Memory Map File** box to generate a memory map output file that specifies the start/end addresses of the new application configuration data in page 1.
8. Pick a local update difference file from the **Remote/Local Update Difference File** drop-down menu. You can select between an Intel HEX, JAM, JBC, and POF output file types. The output file name is the same as the POF output file name with a _dif suffix.
9. Save the CPF setup (optionally), by selecting **Save Conversion Setup…** and specifying a name for the COF output file.
10. Click **OK** to generate initial programming and memory map files.

Figure 12–17. Local Update Partial Programming File Generation

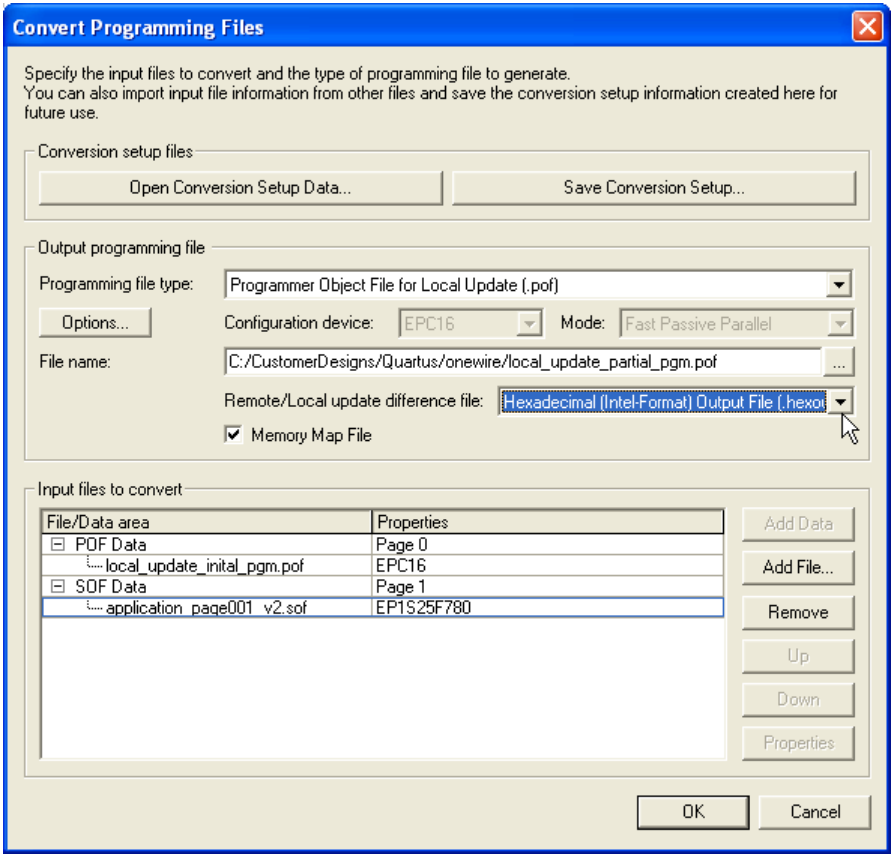
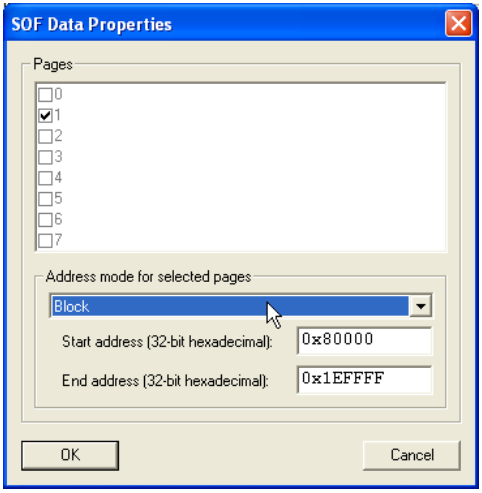


Figure 12–18. Specifying Block Addresses for Application Configuration



Remote Update Programming File Generation

This section describes the programming file generation process for performing remote system upgrades. The Quartus II CPF utility generates the initial and partial programming files for configuration memory within the enhanced configuration devices.

Remote configuration mode uses a factory configuration stored at page 0, and up to seven application configurations stored at pages 1 through 7. The factory configuration cannot be updated after initial production programming. However, the most recent application configuration can be erased and reprogrammed after initial system deployment. Alternatively, a new application configuration can be added provided adequate configuration memory availability.

In remote update mode, you would first create the initial programming file with the factory configuration image and the application configuration(s). Subsequently, you can generate partial programming files to update the most recent application configuration or add a new application configuration. Quartus II CPF can create partial programming files in HEX, JAM, JBC, and POF formats.

In addition to the configuration pages, user data or processor code can also be pre-programmed in the bottom boot and main data areas of the enhanced configuration device memory. The CPF utility accepts a HEX input file for the bottom and main data areas, and includes this data in the

POF output file. However, this is only supported for initial programming file generation. Partial programming file generation for updating user HEX data is not supported, but can be performed using the enhanced configuration device external flash interface.

Initial Programming File Generation

The initial programming file includes configuration data for both factory and application configuration pages. The enhanced configuration device option's bits are always located between byte addresses `0x00010000` and `0x0001003F`. Also, page 0 always starts at `0x00010040` while its end address is dependent on the size of the factory configuration data.

Two memory allocation options exist for application configurations: auto addressing and block addressing. In auto addressing mode, Quartus II packs all application configurations as close together as possible. This maximizes the number of application configurations that can be stored in memory. However, when auto addressing is used you cannot update existing application configurations. Only new application configurations can be added to the memory.

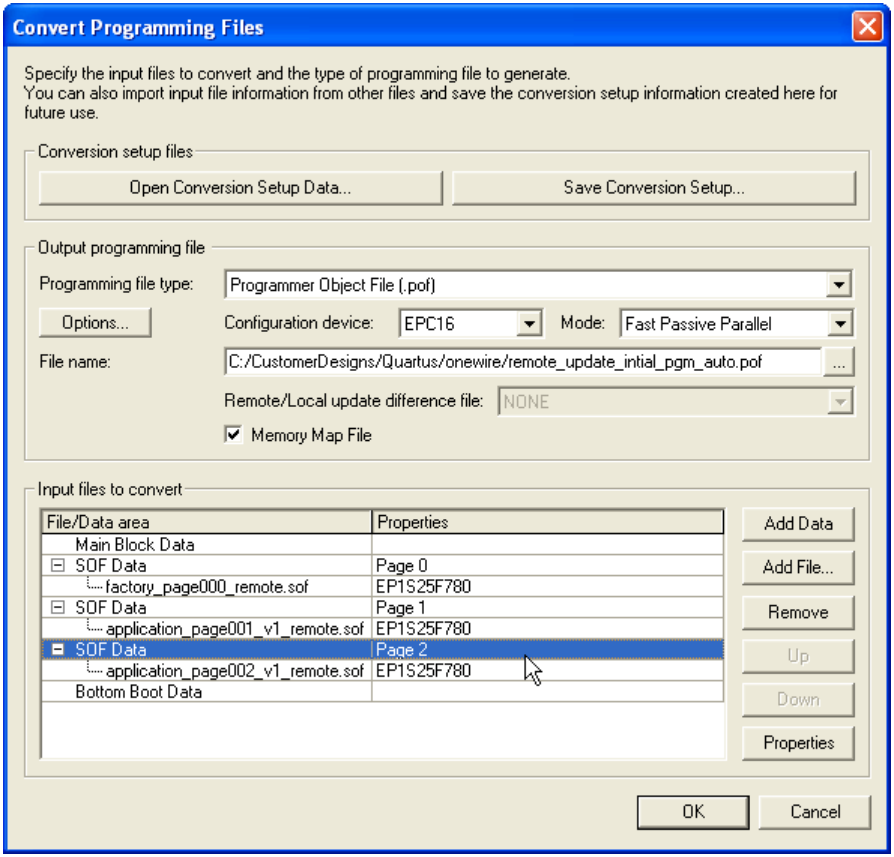
The following steps and screen shot (see Figure 12–19) describe initial programming file generation with auto addressing mode.

1. Open the **Convert Programming Files** window from the **File** menu.
2. Select **Programmer Object File (*.pof)** from the drop-down list titled **Programming file type**.
3. Select the enhanced configuration device used (EPC4, EPC8, EPC16), and the mode used (1-bit Passive Serial or Fast Passive Parallel). Only during the initial programming file generation can you specify the **Options**, **Configuration device**, or **Mode** settings. While generating the partial programming file, all of these settings are grayed out and inaccessible.
4. In the **Input files to convert box**, highlight **SOF Data** at **Page 0** and click **Add File**. Select input SOF file(s) for this configuration page and insert them.
5. Repeat Step 4 for all application configurations (up to 7 maximum).
6. Check the **Memory Map File** box to generate a memory map output file that specifies the start/end addresses of each configuration page and user data blocks.

7. Save the CPF setup (optionally), by selecting **Save Conversion Setup...** and specifying a name for the COF output file.

8. Click **OK** to generate initial programming and memory map files.

Figure 12–19. CPF Setup for Initial Programming File Generation (Auto Addressing)



A sample memory map output file for the preceding setup is shown below. Notice all configuration pages are packed such that two pages can share a flash data sector. This disallows partial programming of application configurations in auto addressing mode.

Block	Start Address	End Address
OPTION BITS	0x00010000	0x0001003F
PAGE 0	0x00010040	0x00054EFA
PAGE 1	0x00054EFC	0x00099DB6
PAGE 2	0x00099DB8	0x000DEC72

f See the *Sharp LHF16J06 Data Sheet Flash memory used in EPC16 devices* at **www.altera.com** to correlate memory addresses to the EPC16 flash sectors.

The block addressing mode allows better control of flash memory allocation. You can allocate a specific flash memory region for each application configuration page. This allocation is done by specifying a block starting and block ending address. While selecting the size of the region, you should account for growth in compressed configuration bitstream sizes due to design changes and additions. In remote update mode, all configuration data is top justified within this allotted memory. In other words, the last byte of configuration data is stored such that it coincides with the highest byte address location within the allotted space. Lower unused memory address locations within the allotted region are filled with 1's. These filler bits are transmitted during the application configuration cycle, but are ignored by the Stratix device. The memory map output file provides the exact byte address where real application configuration data for each page begins. Note that any partial update of the most recent application configuration should erase all allotted flash sectors for that page before storing new configuration data.

In the block addressing mode, HEX input files can be optionally added to the bottom boot and main flash data areas (one HEX file per area is allowed). The HEX file can be stored with relative addressing or absolute addressing. For more information on relative and absolute addressing, see the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet* chapter of the *Configuration Handbook, Volume 2*.

Figures 12–20 and 12–21, and the following steps illustrate generating an initial programming file with block addressing for remote update mode. This example also illustrates preloading user HEX data into bottom boot and main flash sectors.

1. Open the **Convert Programming Files** window from the **File** menu.

2. Select **Programmer Object File (*.pof)** from the drop-down list titled **Programming file type**.

3. Select the enhanced configuration device used (EPC4, EPC8, EPC16), and the mode used (1-bit Passive Serial or Fast Passive Parallel). Only during the initial programming file generation can you specify the **Options**, **Configuration device**, or **Mode** settings. While generating the partial programming file, all of these settings are grayed out and inaccessible.

4. In the **Input** files to convert box, highlight **SOF Data** at **Page 0** and click **Add File**. Select input SOF file(s) for this configuration page and insert them.

5. Repeat Step 4 for all the application configuration pages (pages 1 and 2 in this example).

6. For enabling block addressing, select the **SOF Data** entry for **Page 1**, and click **Properties**. This opens the **SOF Data Properties** dialog box (see Figure 12–21).

7. Pick **Block** from the **Address Mode** drop down selection, and enter 32-bit Hexadecimal byte address for block **Starting Address** and **Ending Address**. Note that for partial programming support, the block start and end addresses should be aligned to a flash sector boundary. This prevents two configuration pages from overlapping within the same flash boundary. See the flash memory datasheet for data sector boundary information. Click **OK** to save SOF data properties.

8. Check the **Memory Map File** box to generate a memory map output file that specifies the start/end addresses of each configuration page and user data blocks.

9. Save the CPF setup (optionally), by selecting **Save Conversion Setup…** and specifying a name for the COF output file.

10. Click **OK** to generate initial programming and memory map files.

Figure 12–20. CPF Setup for Initial Programming File Generation (Block Addressing)

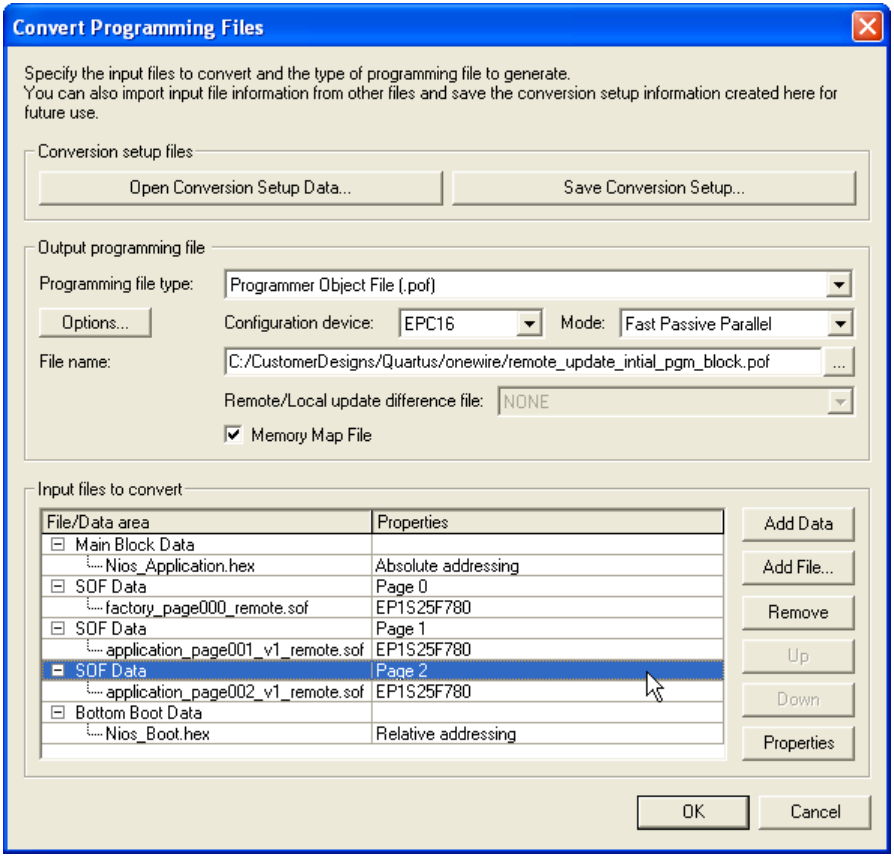
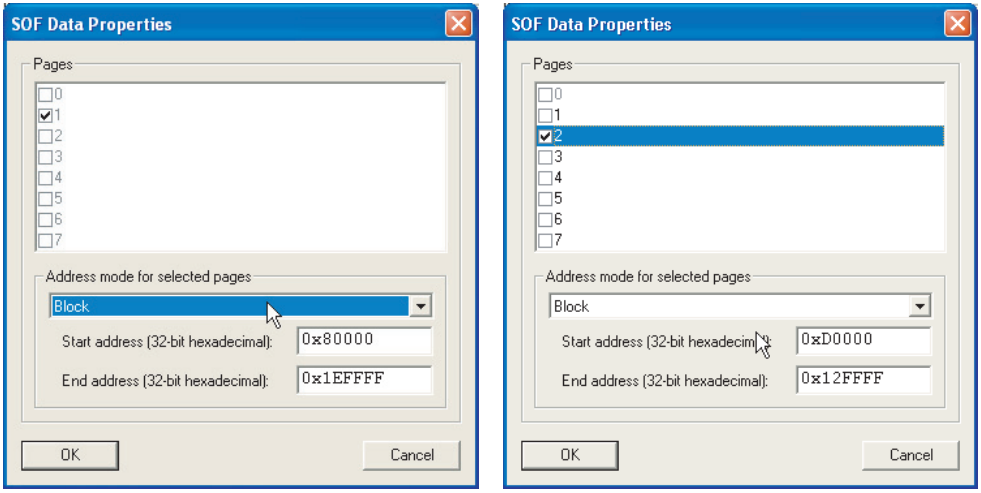


Figure 12–21. Specifying Block Addresses for an Application Configuration



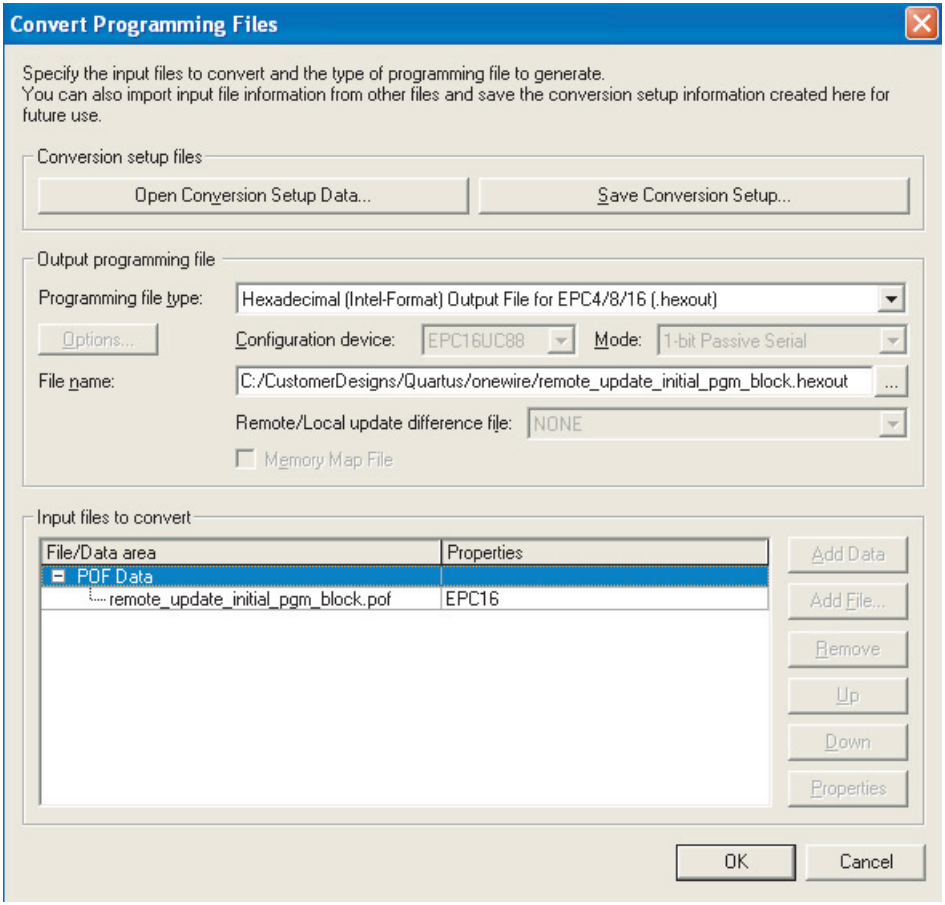
A sample memory map output file for the preceding example is shown below. Note that the allocated memory for page 1 is between `0x00070000` and `0x000BFFFF`, while the actual region used by the current application configuration bitstream is between `0x0007B144` and `0x000BFFFF`. The configuration data is top justified within the allocated SOF data region. Similarly, the allocated memory for page 2 is between `0x000D0000` and `0x0012FFFF`, while the actual region used by the application configuration is between `0x000EB13E` and `0x0012FFF9`.

Block	Start Address	End Address
BOTTOM BOOT	0x00000000	0x000001FF
OPTION BITS	0x00010000	0x0001003F
PAGE 0	0x00010040	0x00054EFA
PAGE 1	0x0007B144	0x000BFFFF
PAGE 2	0x000EB13E	0x0012FFF9
TOP BOOT/MAIN	0x001F0000	0x001F01FF

Also note that the HEX data stored in the main data area uses absolute addressing. If relative addressing were to be used, the main data contents would be justified with the top (higher address locations) of the memory.

The initial POF can be converted to an Intel Hexadecimal format file (*.HEXOUT) using the Quartus II CPF utility. See Figure 12–22.

Figure 12–22. Converting POF Programming File to Intel HEX Format



Partial Programming File Generation

In remote update mode, the Quartus II CPF utility allows an existing application configuration page to be replaced with new data, or a new application configuration to be added. Partial programming files are generated to perform such configuration data updates.

In order to generate a partial programming file, you have to input the initial POF and new configuration data (SOF) to the Quartus II CPF utility. In addition, you have to specify the addressing mode (auto or manual) that was used during initial POF creation. And if block addressing was used, you should specify the block start and end

addresses. With this information, Quartus II ensures that the partial POF only updates the flash region containing the application configuration. The factory configuration (page 0) and configuration option bits are left unaltered during this process. The only exception is when a new application configuration is added, the configuration options bits are updated to include start/end addresses for the new page. All existing page addresses and other configuration options bits remain unchanged.

Figure 12–23 and the following steps illustrate generation of a partial programming file to replace the most recent application configuration. In this example, the initial programming file contained one factory and two application configurations. Hence, the page 2 application configuration is being updated with new data.

1. Open the **Convert Programming Files** window from the **File** menu.

2. Select **Programmer Object File for Remote Update (*.pof)** from the drop-down list titled **Programming file type**, and specify an output file name.

3. In the **Input files to convert** box, highlight **POF Data** and click **Add File**. Select the initial programming POF file for this design and insert it.

4. In the **Input files to convert box**, highlight **SOF Data** and click **Add File**. Select the new application configuration bitstream (SOF) and insert it.

5. When using block addressing, select the **SOF Data** entry for **Page 2**, and click **Properties**. This opens the **SOF Data Properties** dialog box (see Figure 12–24 on page 12–42).

6. Pick **Block** from the **Address Mode** drop down selection, and enter 32-bit Hexadecimal byte address for block **Starting Address** and **Ending Address**. These addresses should be identical to those used to generate the initial programming file. Click **OK** to save SOF data properties.

7. Check the **Memory Map File** box to generate a memory map output file that specifies the start/end addresses of the new application configuration data in page 1.

8. Pick a remote update difference file from the **Remote/Local Update Difference File** drop-down menu. You can select between an Intel HEX, JAM, JBC, and POF output file types. The output file name is the same as the POF output file name with a _dif suffix.

9. Save the CPF setup (optionally), by selecting **Save Conversion Setup...** and specifying a name for the COF output file.

10. Click **OK** to generate initial programming and memory map files.

Figure 12–23. Remote Update Partial Programming File Generation

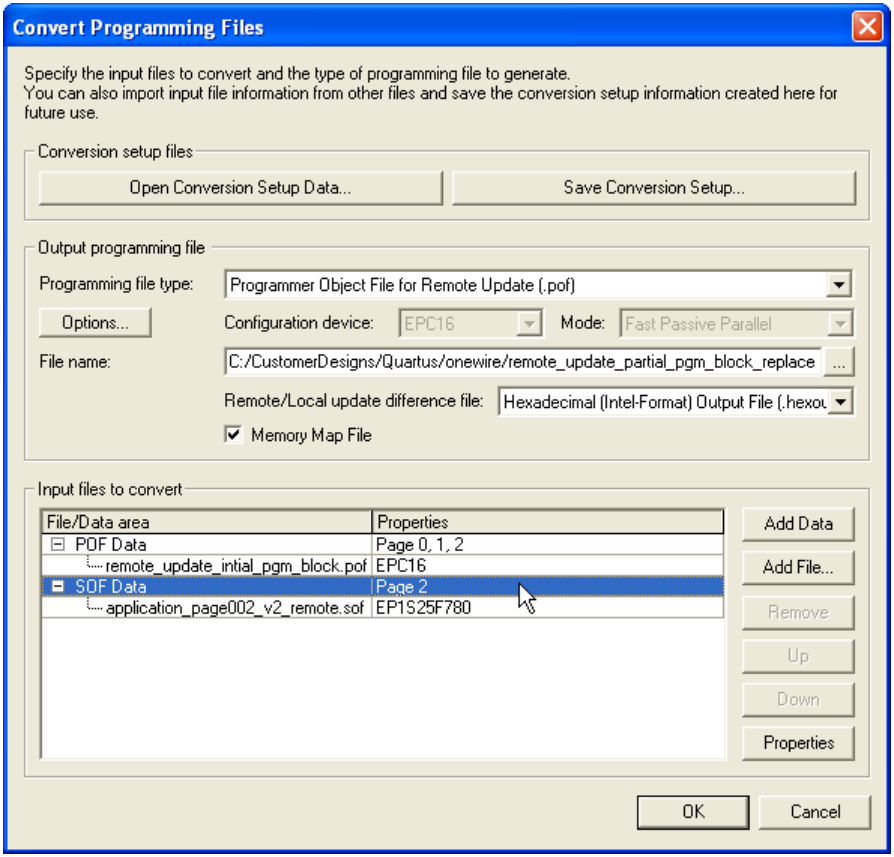
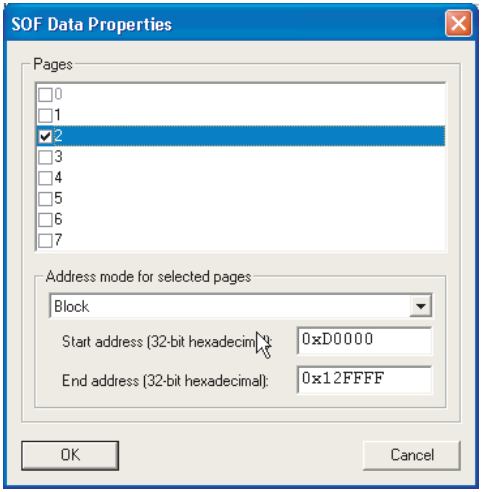


Figure 12–24. Specifying Block Addresses for Application Configuration



For adding a new application configuration, follow the steps listed above with one modification. In Step 5, select **SOF Data** and click on **Properties**. In the **SOF Data Properties** dialog box, select a new page (for example, page 3) and specify the addressing mode information. Continue with steps 7 through 10. When a new page is added, the memory map output file lists the start/end addresses for this page. A sample is shown below:

Block	Start Address	End Address
OPTION BITS	0x00010000	0x0001003F
PAGE 3	0x0012FFFA	0x00174EB4

Combining MAX Devices & Flash Memory

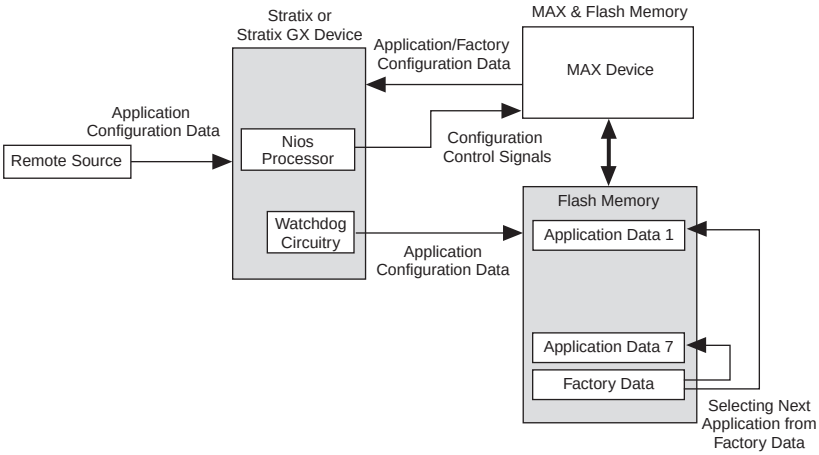
This section describes remote system configuration with the Stratix or Stratix GX device and the Nios embedded processor, using a combination of MAX® devices and flash memory.

You can use MAX 3000 or MAX 7000 devices and an industry-standard flash memory device instead of enhanced configuration devices. In this scheme, flash memory stores configuration data, and the MAX device controls reading and writing to the flash memory, keeping track of address locations.

The MAX device determines which address location and at what length to store configuration data in flash memory. The Nios embedded processor, running in the Stratix or Stratix GX device, receives the

incoming data from the remote source and writes it to the address location in flash memory. The Nios embedded processor initiates loading of factory configuration into the Stratix or Stratix GX device. Figure 12–25 shows remote system configuration using a MAX device and flash memory combination.

Figure 12–25. Remote System Configuration Using a MAX Device & Flash Memory



You can use both remote and local configuration modes in this scheme. You should specify a default page for factory configuration and make sure it is not altered or removed at any time. In remote system configuration mode, PS, FPP, and PPA modes are supported when configuring with MAX and flash devices.

Using an External Processor

This section describes remote system configuration with Stratix or Stratix GX devices and the Nios embedded processor, using an external processor and flash memory devices.

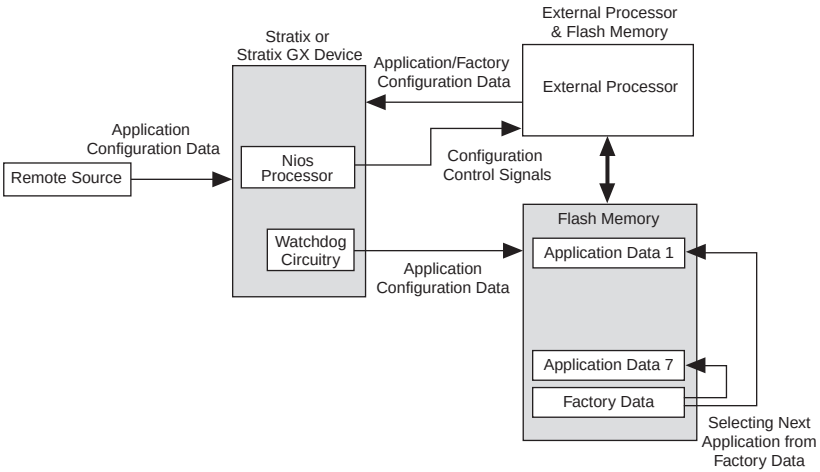
In this scheme, the external processor and flash memory device replace the enhanced configuration device. Flash memory stores configuration data, and the processor controls reading and writing to the flash memory and also keeps track of the address location. This type of remote system configuration supports PS, FPP, and PPA modes.

The processor determines at which address which length to store the configuration data in flash memory. The Nios embedded processor receives the incoming data from a remote source and writes it to the address location in the flash memory, and then initiates loading of factory

configuration data into the Stratix or Stratix GX device. Figure 12–26 shows the remote system configuration using a Nios embedded processor and flash memory.

You can use both remote and local configuration modes in this scheme. You should specify a default page for factory configuration and make sure it is not altered or removed at any time.

Figure 12–26. Remote System Configuration Using External Processor & Flash Memory



Conclusion

Stratix and Stratix GX devices are the first PLDs with dedicated support for remote system configuration. By allowing real-time system upgrades from a remote source, you can use Stratix and Stratix GX devices in a variety of applications that require automatic configuration updates. With the built-in watchdog timer circuitry, Stratix and Stratix GX devices avoid incorrect or erroneous states. Using Stratix and Stratix GX devices with remote system configuration enhances design flexibility and reduces time to market.

Section VII. PCB Layout Guidelines

This section provides information for board layout designers to successfully layout their boards for Stratix® devices. This section contains the required PCB layout guidelines and package specifications.

This section contains the following chapters:

- Chapter 13, Package Information for Stratix Devices
- Chapter 14, Designing with 1.5-V Devices

Revision History

The table below shows the revision history for Chapters 13 and 14.

Chapter	Date/Version	Changes Made
13	July 2005, v3.0	Updated packaging information.
	September 2004, v2.1	• Changed from Chapter 8, Volume 3 to Chapter 13, Volume 2. • Corrected spelling error.
	April 2003, v1.0	• No new changes in Stratix Device Handbook v2.0.
14	January 2005, v1.2	• This chapter was formerly chapter 15.
	September 2004, v1.1	• Changed from Chapter 10, Volume 3 to Chapter 15, Volume 2. • Corrected spelling error.
	April 2003, v1.0	• No new changes in Stratix Device Handbook v2.0.

13. Package Information for Stratix Devices

S53008-3.0

Introduction

This data sheet provides package information for Altera® devices. It includes these sections:

Section	Page
Device & Package Cross Reference	13–1
Thermal Resistance	13–2
Package Outlines	13–3

In this data sheet, packages are listed in order of ascending pin count.

Device & Package Cross Reference

Table 13–1 shows which Altera Stratix® devices are available in BGA, FineLine BGA and Ultra FineLine BGA packages.

Table 13–1. Stratix Devices in BGA, FineLine BGA & Ultra FineLine BGA Packages (Part 1 of 2)

Device	Package	Pins
EP1S10	Flip-chip FineLine BGA	484
	BGA	672
	FineLine BGA	672
	Flip-chip FineLine BGA	780
EP1S20	Flip-chip FineLine BGA	484
	BGA	672
	FineLine BGA	672
	Flip-chip FineLine BGA	780
EP1S25	BGA	672
	FineLine BGA	672
	Flip-chip FineLine BGA	780
	Flip-chip FineLine BGA	1,020
EP1S30	Flip-chip FineLine BGA	780
	Flip-chip BGA	956
	Flip-chip FineLine BGA	1,020

Table 13–1. Stratix Devices in BGA, FineLine BGA & Ultra FineLine BGA Packages (Part 2 of 2)

Device	Package	Pins
EP1S40	Flip-chip FineLine BGA	780
	Flip-chip BGA	956
	Flip-chip FineLine BGA	1,020
	Flip-chip FineLine BGA	1,508
EP1S60	Flip-chip BGA	956
	Flip-chip FineLine BGA	1,020
	Flip-chip FineLine BGA	1,508
EP1S80	Flip-chip BGA	956
	Flip-chip FineLine BGA	1,020
	Flip-chip FineLine BGA	1,508

Thermal Resistance

Table 13–2 provides θ_{JA} (junction-to-ambient thermal resistance) and θ_{JC} (junction-to-case thermal resistance) values for Altera Stratix devices.

Table 13–2. Thermal Resistance of Stratix Devices (Part 1 of 2)

Device	Pin Count	Package	θ_{JC} (° C/W)	θ_{JA} (° C/W) Still Air	θ_{JA} (° C/W) 100 ft./min.	θ_{JA} (° C/W) 200 ft./min.	θ_{JA} (° C/W) 400 ft./min.
EP1S10	484	FineLine BGA	0.38	11.9	9.8	8.4	7.2
	672	BGA	3.2	16.8	13.7	11.9	10.5
	672	FineLine BGA	3.4	17.2	14	12.2	10.8
	780	FineLine BGA	0.43	10.9	8.8	7.4	6.3
EP1S20	484	FineLine BGA	0.30	11.8	9.7	8.3	7.1
	672	BGA	2.5	15.5	12.4	10.7	9.3
	672	FineLine BGA	2.7	16	12.8	11	9.6
	780	FineLine BGA	0.31	10.7	8.6	7.2	6.1

Table 13–2. Thermal Resistance of Stratix Devices (Part 2 of 2)

Device	Pin Count	Package	θ_{JC} (° C/W)	θ_{JA} (° C/W) Still Air	θ_{JA} (° C/W) 100 ft./min.	θ_{JA} (° C/W) 200 ft./min.	θ_{JA} (° C/W) 400 ft./min.
EP1S25	672	BGA	2.2	14.8	11.7	10.0	8.7
	672	FineLine BGA	2.3	15.3	12	10.4	9
	780	FineLine BGA	0.25	10.5	8.5	7.1	6.0
	1020	FineLine BGA	0.25	10.0	8.0	6.6	5.5
EP1S30	780	FineLine BGA	0.2	10.4	8.4	7.0	5.9
	956	BGA	0.2	9.1	7.1	5.8	4.8
	1020	FineLine BGA	0.2	9.9	7.9	6.5	5.4
EP1S40	780	FineLine BGA	0.17	10.4	8.3	6.9	5.8
	956	BGA	0.18	9.0	7.0	5.7	4.7
	1020	FineLine BGA	0.17	9.8	7.8	6.4	5.3
	1508	FineLine BGA	0.18	9.1	7.1	5.8	4.7
EP1S60	956	BGA	0.13	8.9	6.9	5.6	4.6
	1020	FineLine BGA	0.13	9.7	7.7	6.3	5.2
	1508	FineLine BGA	0.13	8.9	7.0	5.6	4.6
EP1S80	956	BGA	0.1	8.8	6.8	5.5	4.5
	1020	FineLine BGA	0.1	9.6	7.6	6.2	5.1
	1508	FineLine BGA	0.1	8.8	6.9	5.5	4.5

Package Outlines

The package outlines on the following pages are listed in order of ascending pin count. Altera package outlines meet the requirements of *JEDEC Publication No. 95.*

484-Pin FineLine BGA - Flip Chip

- All dimensions and tolerances conform to ASME Y14.5M – 1994.
- Controlling dimension is in millimeters.
- Pin A1 may be indicated by an ID dot, or a special feature, in its proximity on the package surface.

Tables 13–3 and 13–4 show the package information and package outline figure references, respectively, for the 484-pin FineLine BGA packaging.

Table 13–3. 484-Pin FineLine BGA Package Information

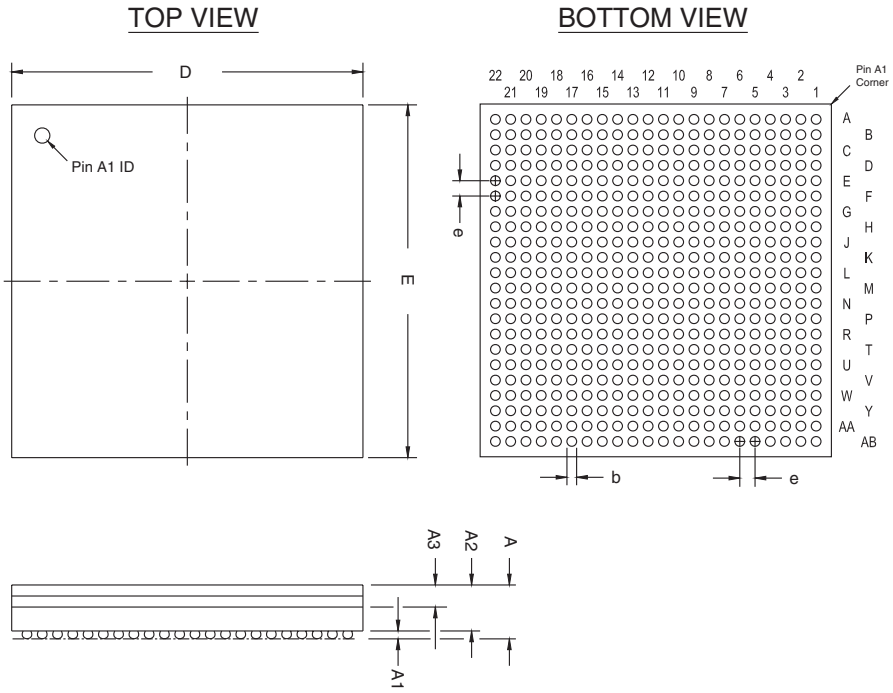
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Substrate material	BT
Solder ball composition	Regular: 63Sn:37Pb (Typ.) Pb-free: Sn:3Ag:0.5Cu (Typ.)
JEDEC outline reference	MS-034 variation: AAJ-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	5.8 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 13–4. 484-Pin FineLine BGA Package Outline Dimensions

Symbol	Millimeter		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	23.00 BSC		
E	23.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

Figure 13–1 shows a package outline for the 484-pin FineLine BGA packaging.

Figure 13–1. 484-Pin FineLine BGA Package Outline



672-Pin FineLine BGA - Flip Chip

- All dimensions and tolerances conform to ASME Y14.5M - 1994.
- Controlling dimension is in millimeters.
- Pin A1 may be indicated by an ID dot, or a special feature, in its proximity on package surface.

Tables 13–5 and 13–6 show the package information and package outline figure references, respectively, for the 672-pin FineLine BGA packaging.

Table 13–5. 672-Pin FineLine BGA Package Information

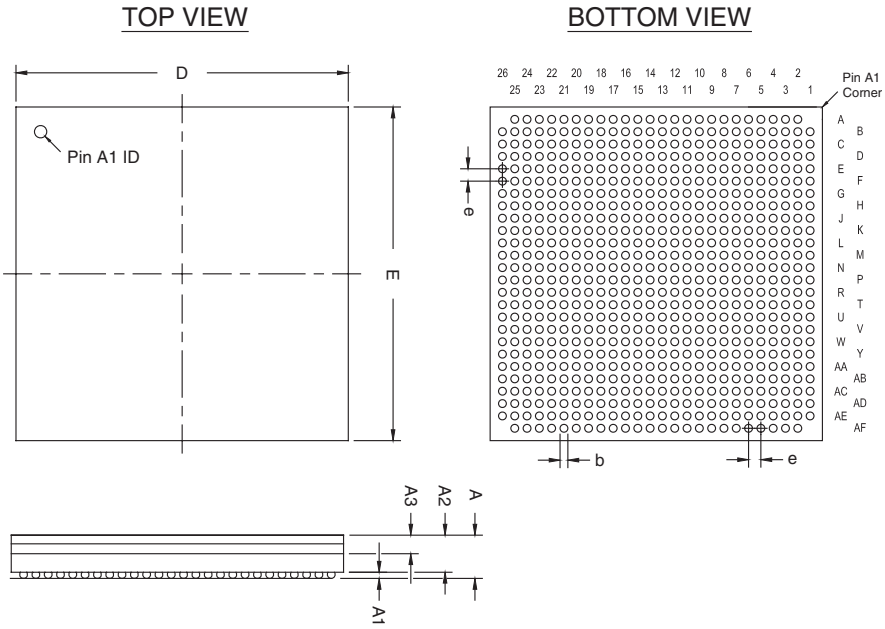
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Substrate material	BT
Solder ball composition	Regular: 63Sn:37Pb (Typ.) Pb-free: Sn:3Ag:0.5Cu (Typ.)
JEDEC Outline Reference	MS-034 Variation: AAL-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	7.7 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 13–6. 672-Pin FineLine BGA Package Outline Dimensions

Symbol	Millimeters		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	27.00 BSC		
E	27.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

Figure 13–2 shows a package outline for the 672-pin FineLine BGA packaging.

Figure 13–2. 672-Pin FineLine BGA Package Outline



780-Pin FineLine BGA - Flip Chip

- All dimensions and tolerances conform to ASME Y14.5M - 1994.
- Controlling dimension is in millimeters.
- Pin A1 may be indicated by an ID dot, or a special feature, in its proximity on package surface.

Tables 13–7 and 13–8 show the package information and package outline figure references, respectively, for the 780-pin FineLine BGA packaging.

Table 13–7. 780-Pin FineLine BGA Package Information

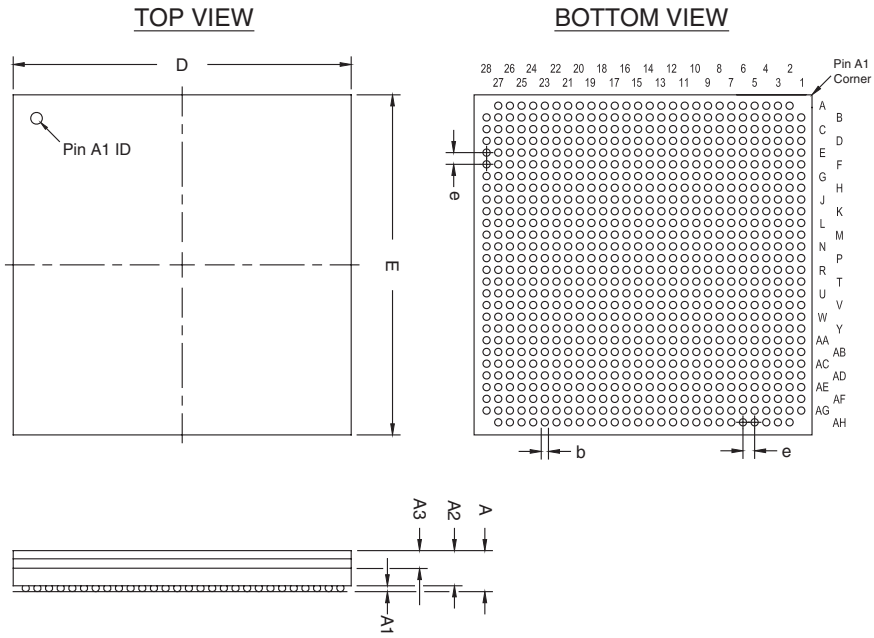
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Substrate material	BT
Solder ball composition	Regular: 63Sn:37Pb (Typ.) Pb-free: Sn:3Ag:0.5Cu (Typ.)
JEDEC outline reference	MS-034 variation: AAM-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	8.9 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 13–8. 780-Pin FineLine BGA Package Outline Dimensions

Symbol	Millimeters		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	29.00 BSC		
E	29.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

Figure 13–3 shows a package outline for the 780-pin FineLine BGA packaging.

Figure 13–3. 780-Pin FineLine BGA Package Outline



956-Pin Ball Grid Array (BGA) - Flip Chip

- All dimensions and tolerances conform to ASME Y14.5M - 1994.
- Controlling dimension is in millimeters.
- Pin A1 may be indicated by an ID dot, or a special feature, in its proximity on package surface.

Tables 13–9 and 13–10 show the package information and package outline figure references, respectively, for the 956-pin BGA packaging.

Table 13–9. 956-Pin BGA Package Information

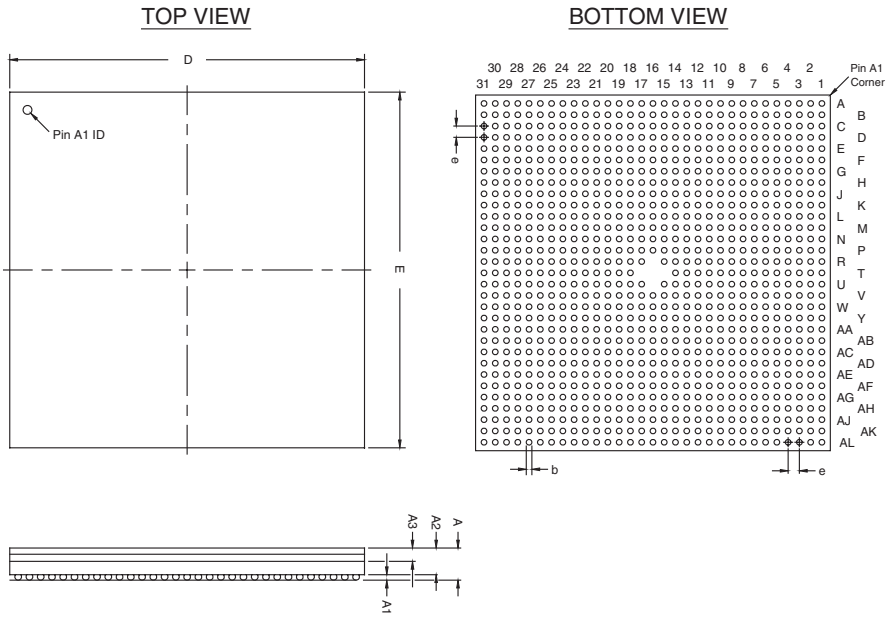
Description	Specification
Ordering code reference	B
Package acronym	BGA
Substrate material	BT
Solder ball composition	Regular: 63Sn:37Pb (Typ.) Pb-free: Sn:3Ag:0.5Cu (Typ.)
JEDEC outline reference	MS-034 Variation: BAU-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	14.6 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 13–10. 956-Pin BGA Package Outline Dimensions

Symbol	Millimeters		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	40.00 BSC		
E	40.00 BSC		
b	0.60	0.75	0.90
e	1.27 BSC		

Figure 13–4 shows a package outline for the 956-pin BGA packaging.

Figure 13–4. 956-Pin BGA Package Outline



1,020-Pin FineLine BGA - Flip Chip

- All dimensions and tolerances conform to ASME Y14.5M - 1994.
- Controlling dimension is in millimeters.
- Pin A1 may be indicated by an ID dot, or a special feature, in its proximity on package surface.

Tables 13–11 and 13–12 show the package information and package outline figure references, respectively, for the 1,020-pin FineLine BGA packaging.

Table 13–11. 1,020 FineLine BGA Package Information

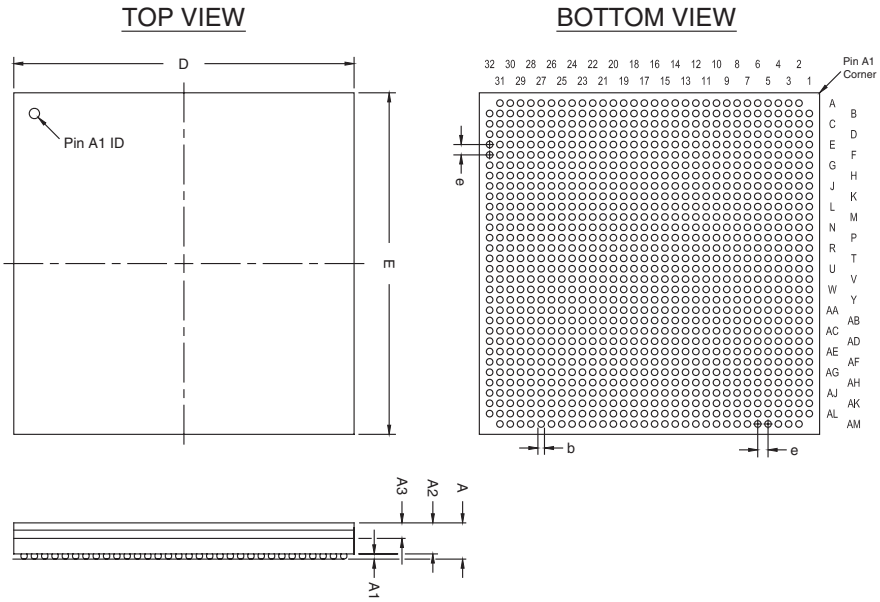
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Substrate material	BT
Solder ball composition	Regular: 63Sn:37Pb (Typ.) Pb-free: Sn:3Ag:0.5Cu (Typ.)
JEDEC outline reference	MS-034 variation: AAP-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	11.5 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 13–12. 1,020-Pin FineLine BGA Package Outline Dimensions

Symbol	Millimeters		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	33.00 BSC		
E	33.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

Figure 13–5 shows a package outline for the 1,020-pin FineLine BGA packaging.

Figure 13–5. 1,020-Pin FineLine BGA Package Outline



1,508-Pin FineLine BGA - Flip Chip

- All dimensions and tolerances conform to ASME Y14.5M - 1994.
- Controlling dimension is in millimeters.
- Pin A1 may be indicated by an ID dot, or a special feature, in its proximity on package surface.

Tables 13–13 and 13–14 show the package information and package outline figure references, respectively, for the 1,508-pin FineLine BGA packaging.

Table 13–13. 1,508-Pin FineLine BGA Package Information

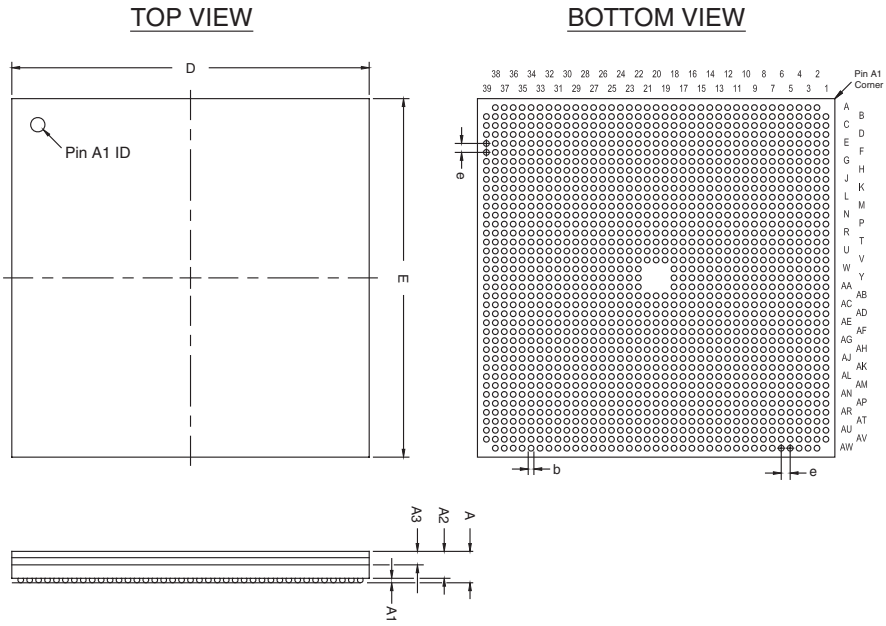
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Substrate material	BT
Solder ball composition	Regular: 63Sn:37Pb (Typ.) Pb-free: Sn:3Ag:0.5Cu (Typ.)
JEDEC outline reference	MS-034 Variation: AAU-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	14.6 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 13–14. 1,508-Pin FineLine BGA Package Outline Dimensions

Symbol	Millimeters		
	Min.	Nom.	Max.
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D	40.00 BSC		
E	40.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

Figure 13–6 shows a package outline for the 1,508-pin FineLine BGA packaging.

Figure 13–6. 1,508-Pin FineLine BGA Package Outline



14. Designing with 1.5-V Devices

C51012-1.1

Introduction

The Cyclone™ FPGA family provides the best solution for high-volume, cost-sensitive applications. Stratix® and Cyclone devices are fabricated on a leading-edge 1.5-V, 0.13-µm, all-layer copper SRAM process.

Using a 1.5-V operating voltage provides the following advantages:

- Lower power consumption compared to 2.5-V or 3.3-V devices.
- Lower operating temperature.
- Less need for fans and other temperature-control elements.

Since many existing designs are based on 5.0-V, 3.3-V and 2.5-V power supplies, a voltage regulator may be required to lower the voltage supply level to 1.5-V. This document provides guidelines for designing with Stratix and Cyclone devices in mixed-voltage and single-voltage systems and provides examples using voltage regulators. This document also includes information on:

- Power Sequencing & Hot Socketing
- Using MultiVolt I/O Pins
- Voltage Regulators
- 1.5-V Regulator Application Examples
- Board Layout

Power Sequencing & Hot Socketing

Because 1.5-V Cyclone FPGAs can be used in a mixed-voltage environment, they have been designed specifically to tolerate any possible power-up sequence. Therefore, the V_{CCIO} and V_{CCINT} power supplies may be powered in any order.

You can drive signals into Cyclone FPGAs before and during power up without damaging the device. In addition, Cyclone FPGAs do not drive out during power up since they are tri-stated during power up. Once the device reaches operating conditions and is configured, Cyclone FPGAs operate as specified by the user.

f See the *Stratix FPGA Family Data Sheet* and the *Cyclone FPGA Family Data Sheet* for more information.

Using MultiVolt I/O Pins

Cyclone FPGAs require a 1.5-V V_{CCINT} and a 3.3-V, 2.5-V, 1.8-V, or 1.5-V I/O supply voltage level (V_{CCIO}). All pins, including dedicated inputs, clock, I/O, and JTAG pins, are 3.3-V tolerant before and after V_{CCINT} and V_{CCIO} are powered.

When V_{CCIO} is connected to 1.5-V, the output is compatible with 1.5-V logic levels. The output pins can be made 1.8-V, 2.5-V, or 3.3-V compatible by using open-drain outputs pulled up with external resistors. You can use external resistors to pull open-drain outputs up with a 1.8-V, 2.5-V, or 3.3-V V_{CCIO}. Table 14–1 summarizes Cyclone MultiVolt I/O support.

Table 14–1. Cyclone MultiVolt I/O Support *Note (1)*

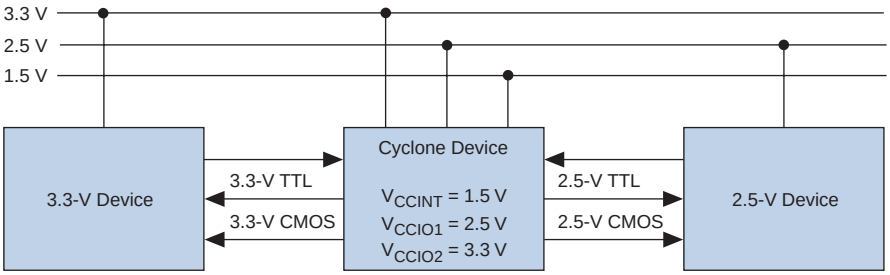
V_{CCIO} (V)	Input Signal					Output Signal				
	1.5-V	1.8-V	2.5-V	3.3-V	5.0-V	1.5-V	1.8-V	2.5-V	3.3-V	5.0-V
1.5-V	v	v	v *(2)*	v *(2)*		v				
1.8-V		v	v	v		v *(3)*	v			
2.5-V			v	v		v *(5)*	v *(5)*	v		
3.3-V			v *(4)*	v	v *(6)*	v *(7)*	v *(7)*	v *(7)*	v	v *(8)*

Notes to Table 14–1:

(1) The PCI clamping diode must be disabled to drive an input with voltages higher than V_{CCIO}.
(2) When V_{CCIO} = 1.5-V and a 2.5-V or 3.3-V input signal feeds an input pin, higher pin leakage current is expected.
(3) When V_{CCIO} = 1.8-V, a Cyclone device can drive a 1.5-V device with 1.8-V tolerant inputs.
(4) When V_{CCIO} = 3.3-V and a 2.5-V input signal feeds an input pin, the V_{CCIO} supply current will be slightly larger than expected.
(5) When V_{CCIO} = 2.5-V, a Cyclone device can drive a 1.5-V or 1.8-V device with 2.5-V tolerant inputs.
(6) Cyclone devices can be 5.0-V tolerant with the use of an external resistor and the internal PCI clamp diode.
(7) When V_{CCIO} = 3.3-V, a Cyclone device can drive a 1.5-V, 1.8-V, or 2.5-V device with 3.3-V tolerant inputs.
(8) When V_{CCIO} = 3.3-V, a Cyclone device can drive a device with 5.0-V LVTTL inputs but not 5.0-V LVCMOS inputs.

Figure 14–1 shows how Cyclone FPGAs interface with 3.3--V and 2.5-V devices while operating with a 1.5-V V_{CCINT} to increase performance and save power.

Figure 14–1. Cyclone FPGAs Interface with 3.3-V & 2.5-V Devices



Voltage Regulators

This section explains how to generate a 1.5-V supply from another system supply. Supplying power to the 1.5-V logic array and/or I/O pins requires a 5.0-V- or 3.3-V-to-1.5-V voltage regulator. A linear regulator is ideal for low-power applications because it minimizes device count and has acceptable efficiency for most applications. A switching voltage regulator provides optimal efficiency. Switching regulators are ideal for high-power applications because of their high efficiency.

This section will help you decide which regulator to use in your system, and how to implement the regulator in your design. There are several companies that provide voltage regulators for low-voltage devices, such as Linear Technology Corporation, Maxim Integrated Products, Intersil Corporation (Elantec), and National Semiconductor Corporation.

Table 14–2 shows the terminology and specifications commonly encountered with voltage regulators. Symbols are shown in parentheses. If the symbols are different for linear and switching regulators, the linear regulator symbol is listed first.

Table 14–2. Voltage Regulator Specifications & Terminology (Part 1 of 2)

Specification/Terminology	Description
Input voltage range (V_{IN},V_{CC})	Minimum and maximum input voltages define the input voltage range, which is determined by the regulator process voltage capabilities.
Line regulation (line regulation, V_{OUT})	Line regulation is the variation of the output voltage (V_{OUT}) with changes in the input voltage (V_{IN}). Error amplifier gain, pass transistor gain, and output impedance all influence line regulation. Higher gain results in better regulation. Board layout and regulator pin-outs are also important because stray resistance can introduce errors.
Load regulation (load regulation, V_{OUT})	Load regulation is a variation in the output voltage caused by changes in the input supply current. Linear Technology regulators are designed to minimize load regulation, which is affected by error amplifier gain, pass transistor gain, and output impedance.
Output voltage selection	Output voltage selection is adjustable by resistor voltage divider networks, connected to the error amplifier input, that control the output voltage. There are multiple output regulators that create 5.0-, 3.3-, 2.5-, 1.8- and 1.5-V supplies.
Quiescent current	Quiescent current is the supply current during no-load or quiescent state. This current is sometimes used as a general term for a supply current used by the regulator.
Dropout voltage	Dropout voltage is the difference between the input and output voltages when the input is low enough to cause the output to drop out of regulation. The dropout voltage should be as low as possible for better efficiency.
Current limiting	Voltage regulators are designed to limit the amount of output current in the event of a failing load. A short in the load causes the output current and voltage to decrease. This event cuts power dissipation in the regulator during a short circuit.
Thermal overload protection	This feature limits power dissipation if the regulator overheats. When a specified temperature is reached, the regulator turns off the output drive transistors, allowing the regulator to cool. Normal operation resumes once the regulator reaches a normal operating temperature.
Reverse current protection	If the input power supply fails, large output capacitors can cause a substantial reverse current to flow backward through the regulator, potentially causing damage. To prevent damage, protection diodes in the regulator create a path for the current to flow from V_{OUT} to V_{IN}.
Stability	The dominant pole placed by the output capacitor influences stability. Voltage regulator vendors can assist you in output capacitor selection for regulator designs that differ from what is offered.

Table 14–2. Voltage Regulator Specifications & Terminology (Part 2 of 2)

Specification/Terminology	Description
Minimum load requirements	A minimum load from the voltage divider network is required for good regulation, which also serves as the ground for the regulator's current path.
Efficiency	Efficiency is the division of the output power by the input power. Each regulator model has a specific efficiency value. The higher the efficiency value, the better the regulator.

Linear Voltage Regulators

Linear voltage regulators generate a regulated output from a larger input voltage using current pass elements in a linear mode. There are two types of linear regulators available: one using a series pass element and another using a shunt element (e.g., a zener diode). Altera recommends using series linear regulators because shunt regulators are less efficient.

Series linear regulators use a series pass element (i.e., a bipolar transistor or MOSFET) controlled by a feedback error amplifier (see Figure 14–2) to regulate the output voltage by comparing the output to a reference voltage. The error amplifier drives the transistor further on or off continuously to control the flow of current needed to sustain a steady voltage level across the load.

Figure 14–2. Series Linear Regulator

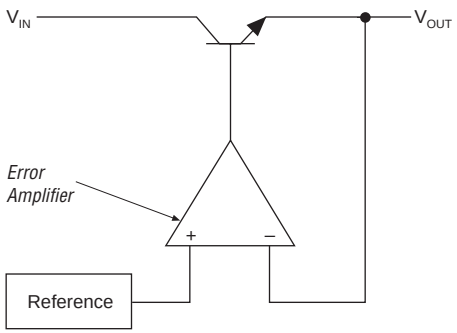


Table 14–3 shows the advantages and disadvantages of linear regulators compared to switching regulators.

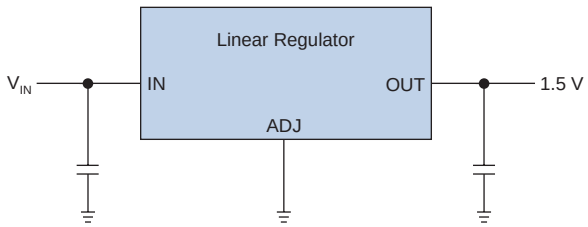
Table 14–3. Linear Regulator Advantages & Disadvantages

Advantages	Disadvantages
Requires few supporting components Low cost Requires less board space Quick transient response Better noise and drift characteristics No electromagnetic interference (EMI) radiation from the switching components Tighter regulation	Less efficient (typically 60%) Higher power dissipation Larger heat sink requirements

You can minimize the difference between the input and output voltages to improve the efficiency of linear regulators. The dropout voltage is the minimum allowable difference between the regulator's input and output voltage.

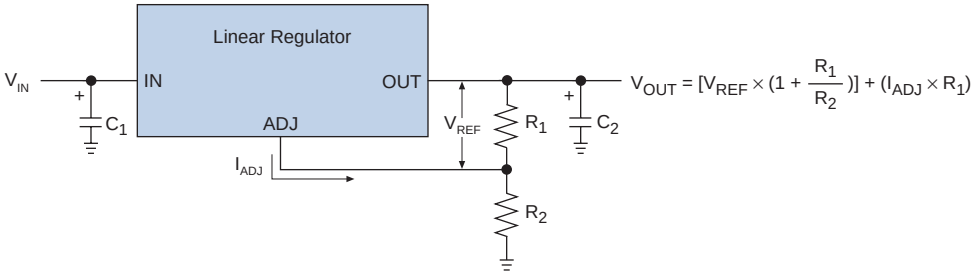
Linear regulators are available with fixed, variable, single, or multiple outputs. Multiple-output regulators can generate multiple outputs (e.g., 1.5- and 3.3-V outputs). If the board only has a 5.0-V power voltage supply, you should use multiple-output regulators. The logic array requires a 1.5-V power supply, and a 3.3-V power supply is required to interface with 3.3- and 5.0-V devices. However, fixed-output regulators have fewer supporting components, reducing board space and cost. Figure 14–3 shows an example of a three-terminal, fixed-output linear regulator.

Figure 14–3. Three-Terminal, Fixed-Output Linear Regulator



Adjustable-output regulators contain a voltage divider network that controls the regulator's output. Figure 14–4 shows how you can also use a three-terminal linear regulator in an adjustable-output configuration.

Figure 14–4. Adjustable-Output Linear Regulator



Switching Voltage Regulators

Step-down switching regulators can provide 3.3-V-to-1.5-V conversion with up to 95% efficiencies. This high efficiency comes from minimizing quiescent current, using a low-resistance power MOSFET switch, and, in higher-current applications, using a synchronous switch to reduce diode losses.

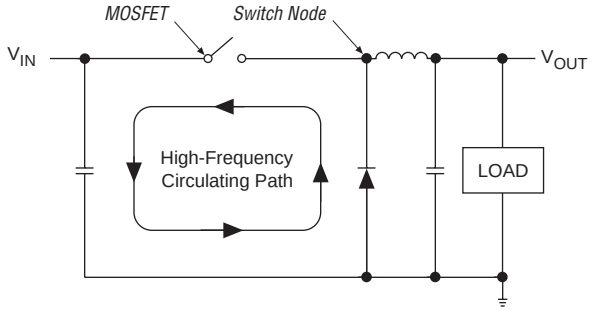
Switching regulators supply power by pulsing the output voltage and current to the load. Table 14–4 shows the advantages and disadvantages of switching regulators compared to linear regulators. For more information on switching regulators, see *Application Note 35: Step Down Switching Regulators* from Linear Technology.

Table 14–4. Switching Regulator Advantages & Disadvantages

Advantages	Disadvantages
Highly efficient (typically >80%) Reduced power dissipation Smaller heat sink requirements Wider input voltage range High power density	Generates EMI Complex to design Requires 15 or more supporting components Higher cost Requires more board space

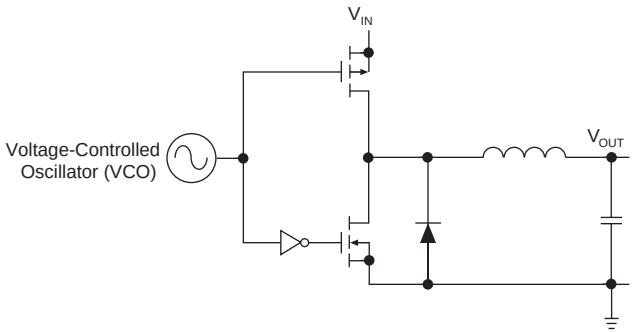
There are two types of switching regulators, asynchronous and synchronous. Asynchronous switching regulators have one field effect transistor (FET) and a diode to provide the current path while the FET is off (see Figure 14–5).

Figure 14–5. Asynchronous Switching Regulator



Synchronous switching regulators have a voltage- or current-controlled oscillator that controls the on and off time of the two MOSFET devices that supply the current to the circuit (see Figure 14–6).

Figure 14–6. Voltage-Controlled Synchronous Switching Regulator



Maximum Output Current

Select an external MOSFET switching transistor (optional) based on the maximum output current that it can supply. Use a MOSFET with a low on-resistance and a voltage rating high enough to avoid avalanche breakdown. For gate-drive voltages less than 9-V, use a logic-level MOSFET. A logic-level MOSFET is only required for topologies with a controller IC and an external MOSFET.

Selecting Voltage Regulators

Your design requirements determine which voltage regulator you need. The key to selecting a voltage regulator is understanding the regulator parameters and how they relate to the design.

The following checklist can help you select the proper regulator for your design:

- Do you require a 3.3-V, 2.5-V, and 1.5-V output (V_{OUT})?
- What precision is required on the regulated 1.5-V supplies (line and load regulation)?
- What supply voltages (V_{IN} or V_{CC}) are available on the board?
- What voltage variance (input voltage range) is expected on V_{IN} or V_{CC}?
- What is the maximum I_{CC} (I_{OUT}) required by your Altera® device?
- What is the maximum current surge ($I_{OUT(MAX)}$) that the regulator will need to supply instantaneously?

Choose a Regulator Type

If required, select either a linear, asynchronous switching, or synchronous switching regulator based on your output current, regulator efficiency, cost, and board-space requirements. DC-to-DC converters have output current capabilities from 1 to 8 A. You can use a controller with an external MOSFET rated for higher current for higher-output-current applications.

Calculate the Maximum Input Current

Use the following equation to estimate the maximum input current based on the output power requirements at the maximum input voltage:

$$I_{IN,DC(MAX)} = \frac{V_{OUT} \times I_{OUT(MAX)}}{\eta \times V_{IN(MAX)}}$$

Where η is nominal efficiency: typically 90% for switching regulators, 60% for linear 2.5-V-to-1.5-V conversion, 45% for linear 3.3-V-to-1.5-V conversion, and 30% for linear 5.0-V-to-1.5-V conversion.

Once you identify the design requirements, select the voltage regulator that is best for your design. Tables 14–5 and 14–6 list a few Linear Technology and Elantec regulators available at the time this document

was published. There may be more regulators to choose from depending on your design specification. Contact a regulator manufacturer for availability.

Table 14–5. Linear Technology 1.5-V Output Voltage Regulators

Voltage Regulator	Regulator Type	Total Number of Components	V_{IN} (V)	I_{OUT} (A)	Special Features
LT1573	Linear	10	2.5 or 3.3 *(1)*	6	–
LT1083	Linear	5	5.0	7.5	–
LT1084	Linear	5	5.0	5	–
LT1085	Linear	5	5.0	3	Inexpensive solution
LTC1649	Switching	22	3.3	15	Selectable output
LTC1775	Switching	17	5.0	5	–

Note to Table 14–5:

(1) A 3.3-V V_{IN} requires a 3.3-V supply to the regulator's input and 2.5-V supply to bias the transistors.

Table 14–6. Elantec 1.5-V Output Voltage Regulators

Voltage Regulator	Regulator Type	Total Number of Components	V_{IN} (V)	I_{OUT} (A)	Special Features
EL7551C	Switching	11	5.0	1	–
EL7564CM	Switching	13	5.0	4	–
EL7556BC	Switching	21	5.0	6	–
EL7562CM	Switching	17	3.3 or 5.5	2	–
EL7563CM	Switching	19	3.3	4	–

Voltage Divider Network

Design a voltage divider network if you are using an adjustable output regulator. Follow the controller or converter IC's instructions to adjust the output voltage.

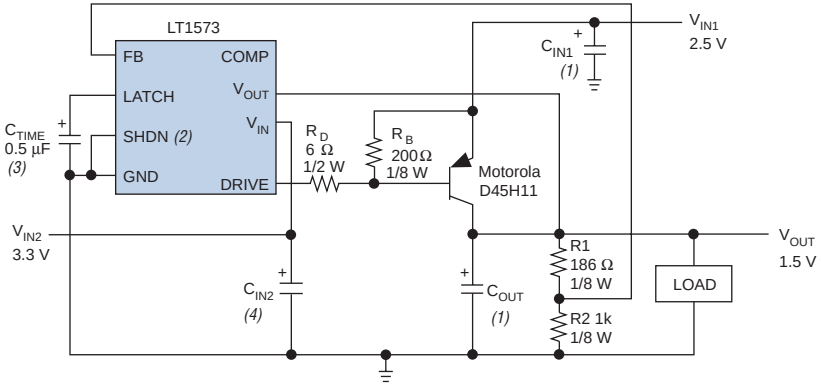
1.5-V Regulator Circuits

This section contains the circuit diagrams for the voltage regulators discussed in this chapter. You can use the voltage regulators in this section to generate a 1.5-V power supply. See the voltage regulator data sheet to find detailed specifications. If you require further information that is not shown in the data sheet, contact the regulator's vendor.

Figures 14–7 through 14–12 show the circuit diagrams of Linear Technology voltage regulators listed in Table 14–5.

The LT1573 linear voltage regulator converts 2.5-V to 1.5-V with an output current of 6A (see Figure 14–7).

Figure 14–7. LT1573: 2.5-V-to-1.5-V/6.0-A Linear Voltage Regulator



Notes to Figure 14–7:

(1) C_{IN1} and C_{OUT} are AVX 100-µF / 10-V surface-mount tantalum capacitors.

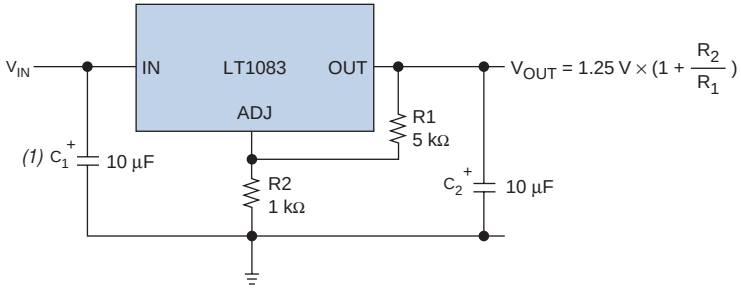
(2) Use SHDN (active high) to shut down the regulator.

(3) C_{TIME} is a 0.5-µF capacitor for 100-ms time out at room temperature.

(4) C_{IN2} is an AVX 15-µF / 10-V surface-mount tantalum capacitor.

Use adjustable 5.0- to 1.5-V regulators (shown in Figures 14–8 through 14–10) for 3.0- to 7.5-A low-cost, low-device-count, board-space-efficient solutions.

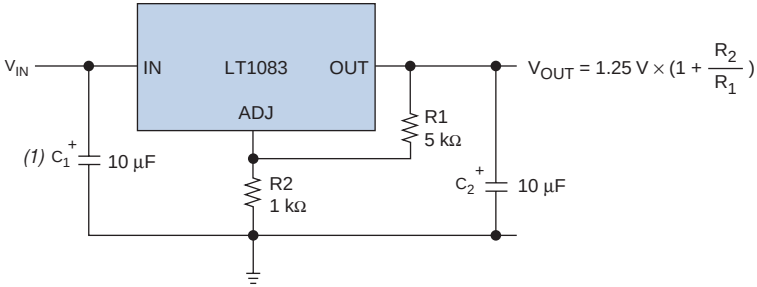
Figure 14–8. LT1083: 5.0-V-to-1.5-V/7.5-A Linear Voltage Regulator



Note to Figure 14–8:

(1) This capacitor is necessary to maintain the voltage level at the input regulator. There could be a voltage drop at the input if the voltage supply is too far away.

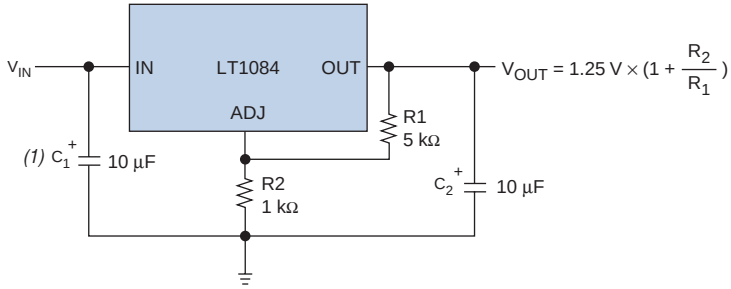
Figure 14–9. LT1084: 5.0-V-to-1.5-V/5.0-A Linear Voltage Regulator



Note to Figure 14–9:

(1) This capacitor is necessary to maintain the voltage level at the input regulator. There could be a voltage drop at the input if the voltage supply is too far away.

Figure 14–10. LT1085: 5.0-V-to-1.5-V/3-A Linear Voltage Regulator

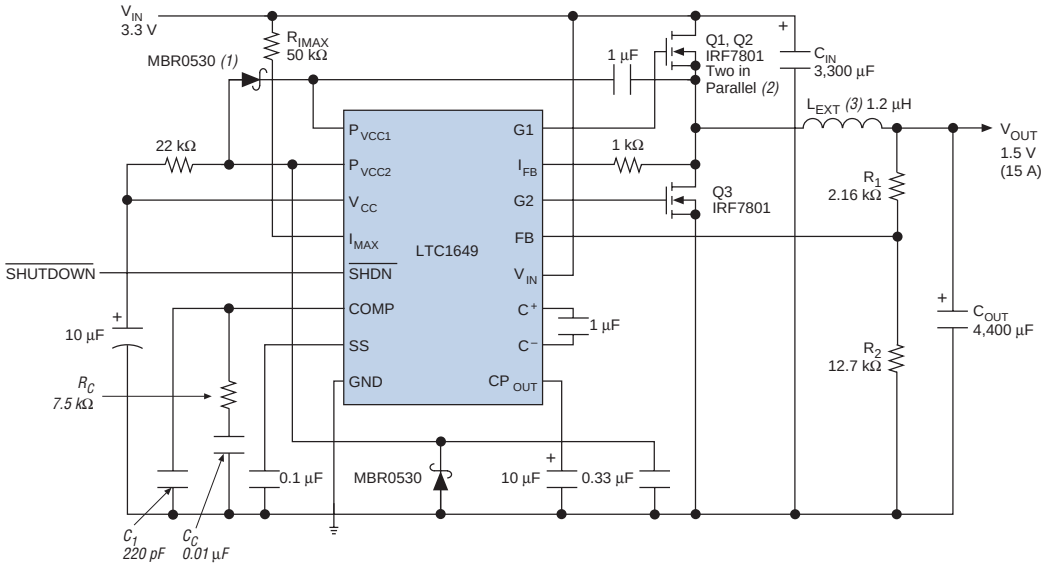


Note to Figure 14–10:

(1) This capacitor is necessary to maintain the voltage level at the input regulator. There could be a voltage drop at the input if the voltage supply is too far away.

Figure 14–11 shows a high-efficiency switching regulator circuit diagram. A selectable resistor network controls the output voltage. The resistor values in Figure 14–11 are selected for 1.5-V output operation.

Figure 14–11. LT1649: 3.3-V-to-1.5-V/15-A Asynchronous Switching Regulator

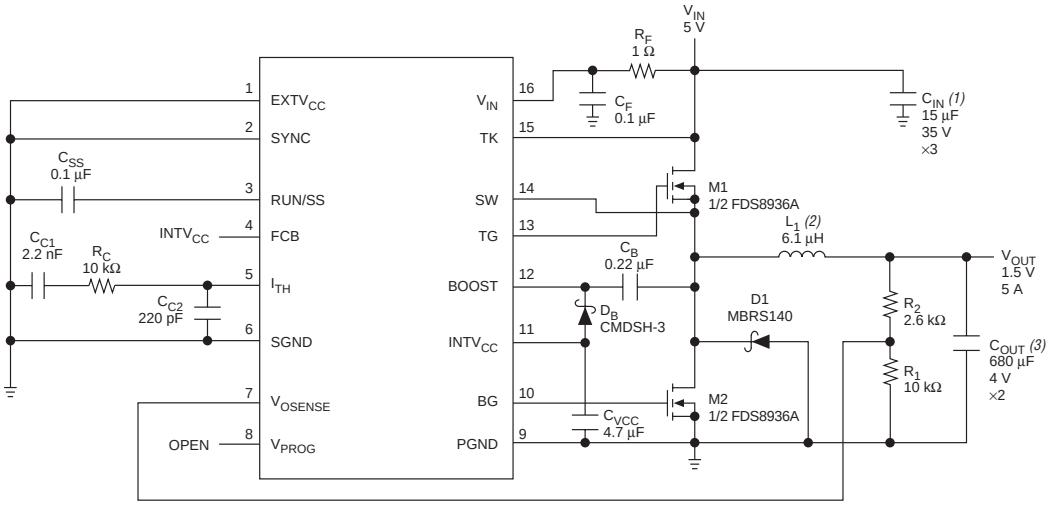


Notes to Figure 14–11:

(1) MBR0530 is a Motorola device.
(2) IRF7801 is a International Rectifier device.
(3) See the Panasonic 12TS-1R2HL device.

Figure 14–12 shows synchronous switching regulator with adjustable outputs.

Figure 14–12. LTC1775: 5.0-V-to-1.5-V/5-A Synchronous Switching Regulator



Notes to Figure 14–12:

(1) This is a KEMETT495X156M035AS capacitor.
(2) This is a Sumida CDRH127-6R1 inductor.
(3) This is a KEMETT510X687K004AS capacitor.

Figures 14–13 through 14–17 show the circuit diagrams of Elantec voltage regulators listed in Table 14–6.

Figures 14–13 through 14–15 show the switching regulator that converts 5.0-V to 1.5-V with different output current.

Figure 14–13. EL7551C: 5.0-V-to-1.5-V/1-A Synchronous Switching Regulator

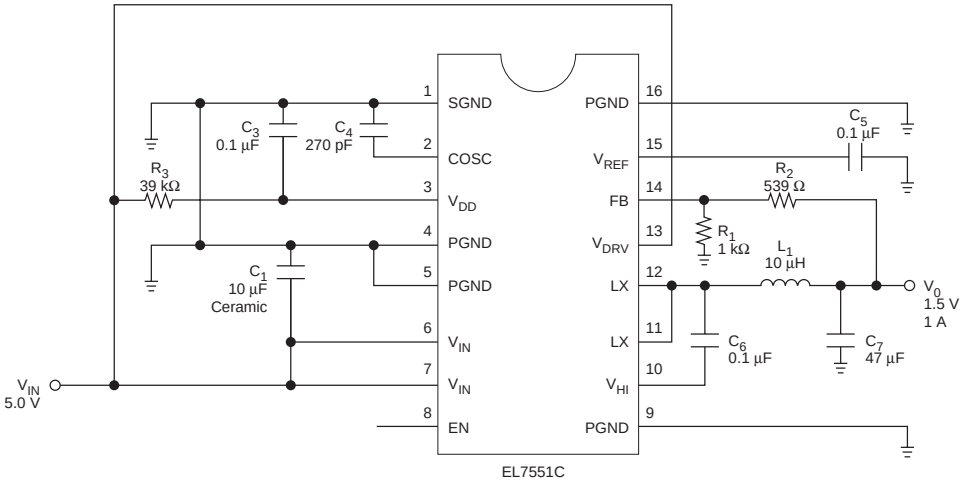


Figure 14–14. EL7564CM: 5.0-V-to-1.5-V/4-A Synchronous Switching Regulator

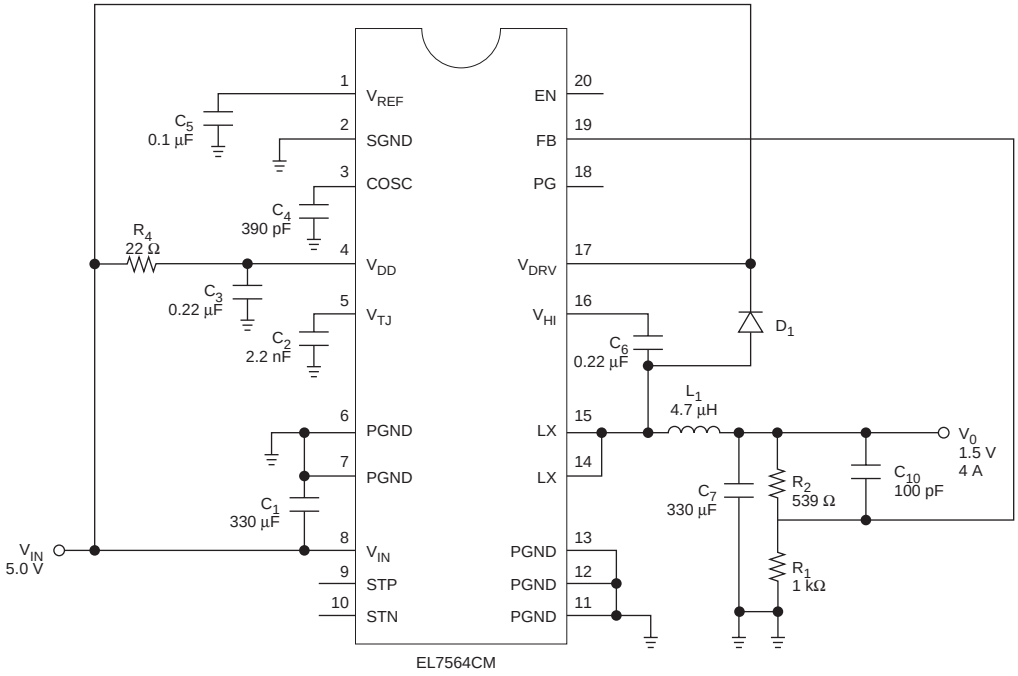
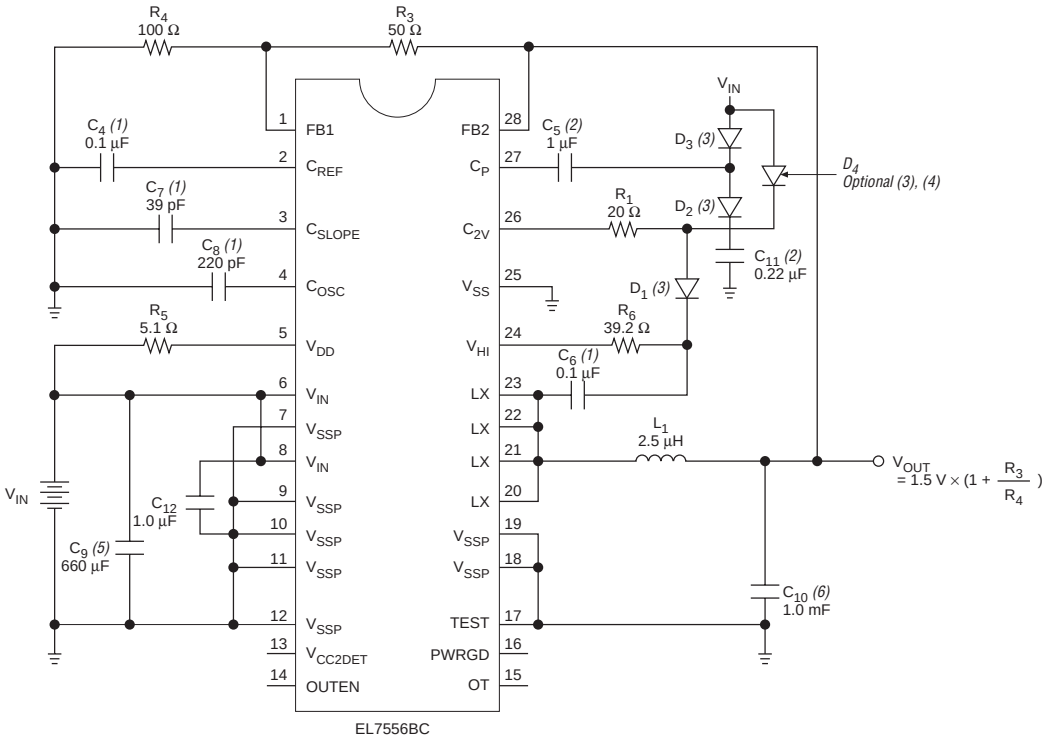


Figure 14–15. EL7556BC: 5.0-V-to-1.5-V/6-A Synchronous Switching Regulator



Notes to Figures 14–13 to 14–15:

(1) These capacitors are ceramic capacitors.
(2) These capacitors are ceramic or tantalum capacitor.
(3) These are BAT54S fast diodes.
(4) D4 is only required for EL7556ACM.
(5) This is a Sprague 293D337X96R3 2X330µF capacitor.
(6) This is a Sprague 293D337X96R3 3X330µF capacitor.

Figures 14–16 and 14–17 show the switching regulator that converts 3.3 V to 1.5 V with different output currents.

Figure 14–16. EL7562CM: 3.3-V to 1.5-V/2-A Synchronous Switching Regulator

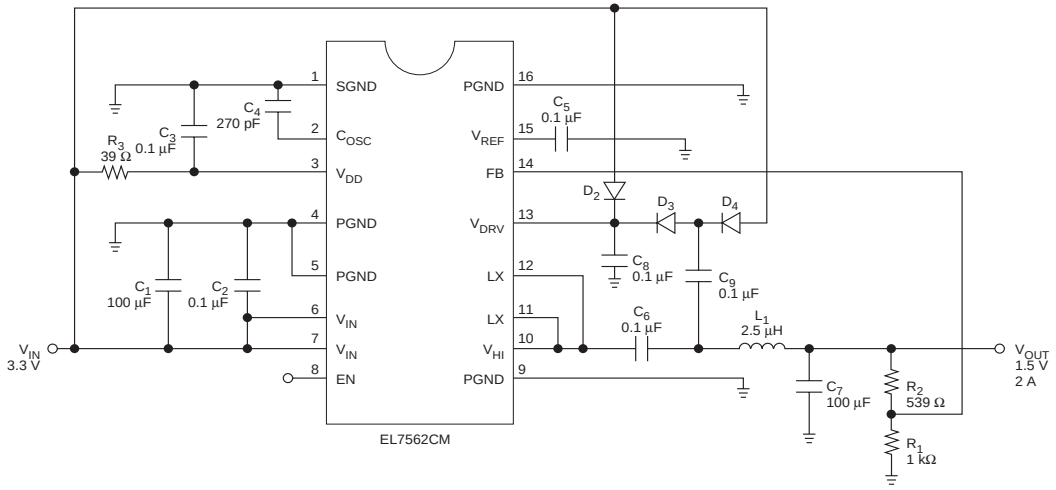
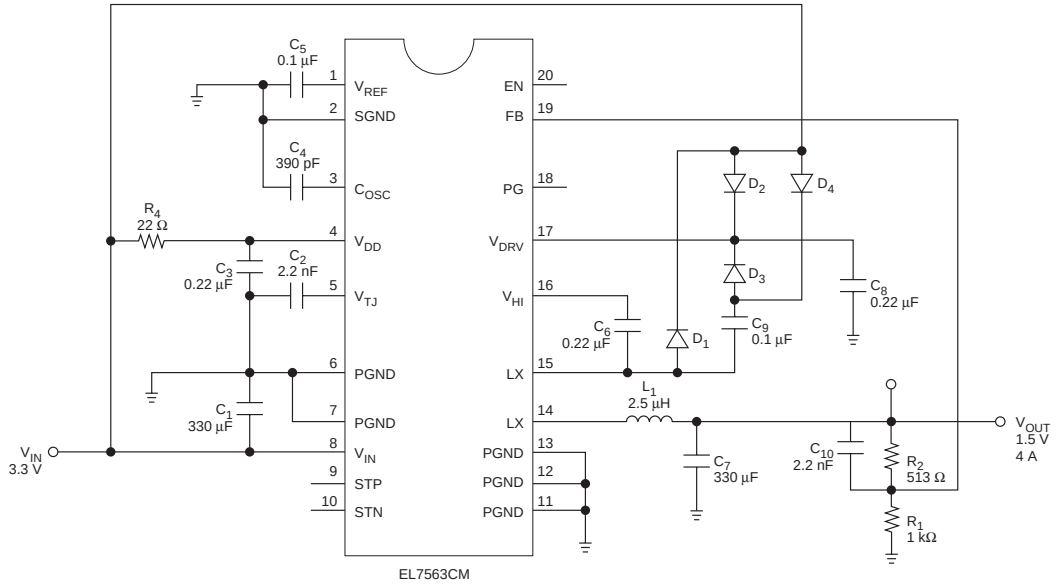


Figure 14–17. EL7563CM: 3.3-V to 1.5-V/4-A Synchronous Switching Regulator



1.5-V Regulator Application Examples

The following sections show the process used to select a voltage regulator for three sample designs. The regulator selection is based on the amount of power that the Cyclone device consumes. There are 14 variables to consider when selecting a voltage regulator. The following variables apply to Cyclone device power consumption:

- f_{MAX}
- Output and bidirectional pins
- Average toggle rate for I/O pins (tog_{IO})
- Average toggle rate for logic elements (LEs) (tog_{LC})
- User-mode I_{CC} consumption
- Maximum power-up I_{CCINT} requirement
- Utilization
- V_{CCIO} supply level
- V_{CCINT} supply level

The following variables apply to the voltage regulator:

- Output voltage precision requirement
- Supply voltage on the board
- Voltage supply output current
- Variance of board supply
- Efficiency

Different designs have different power consumptions based on the variables listed. Once you calculate the Cyclone device's power consumption, you must consider how much current the Cyclone device needs. You can use the Cyclone power calculator (available at **www.altera.com**) or the PowerGauge™ tool in the Quartus II software to determine the current needs. Also check the maximum power-up current requirement listed in the Power Consumption section of the Cyclone FPGA Family Data Sheet because the power-up current requirement may exceed the user-mode current consumption for a specific design.

Once you determine the minimum current the Cyclone device requires, you must select a voltage regulator that can generate the desired output current with the voltage and current supply that is available on the board using the variables listed in this section. An example is shown to illustrate the voltage regulator selection process.

Synchronous Switching Regulator Example

This example shows a worst-case scenario for power consumption where the design uses all the LEs and RAM. Table 14–7 shows the design requirements for 1.5-V design using a Cyclone EP1C12 FPGA.

Table 14–7. Design Requirements for the Example EP1C12F324C

Design Requirement	Value
Output voltage precision requirement	±5%
Supply voltages available on the board	3.3 V
Voltage supply output current available for this section ($I_{IN, DC(MAX)}$)	2 A
Variance of board supply (V_{IN})	±5%
f_{MAX}	150 MHz
Average tog_{IO}	12.5%
Average tog_{LC}	12.5%
Utilization	100%
Output and bidirectional pins	125
V_{CCIO} supply level	3.3 V
V_{CCINT} supply level	1.5 V
Efficiency	≥90%

Table 14–8 uses the checklist on page 14–9 to help select the appropriate voltage regulator.

Table 14–8. Voltage Regulator Selection Process for EP1C12F324C Design (Part 1 of 2)

Output voltage requirements	V_{OUT} = 1.5 V
Supply voltages	V_{IN} OR V_{CC} = 3.3 V
Supply variance from Linear Technology data sheet	Supply variance = ±5%
Estimated I_{CCINT} Use Cyclone Power Calculator	I_{CCINT} = 620 mA
Estimated I_{CCIO} if regulator powers V_{CCIO} Use Cyclone Power Calculator (not applicable in this example because V_{CCIO} = 3.3 V)	I_{CCIO} = N/A
Total user-mode current consumption $I_{CC} = I_{CCINT} + I_{CCIO}$	I_{CC} = 620 mA

Table 14–8. Voltage Regulator Selection Process for EP1C12F324C Design (Part 2 of 2)

EP1C12 maximum power-up current requirement See Power Consumption section of the Cyclone FPGA Family Data Sheet for other densities	$I_{PUC(MAX)}$ = 900 mA
Maximum output current required Compare I_{CC} with $I_{PUC(MAX)}$	$I_{OUT(MAX)}$ = 900 mA
Voltage regulator selection See *Linear Technology LTC 1649 data sheet* See *Intersil (Elantec) EL7562C data sheet*	LTC1649 $I_{OUT(MAX)}$ = 15 A EL7562C $I_{OUT(MAX)}$ = 2 A
LTC1649	
Nominal efficiency (η)	Nominal efficiency (η) = > 90%
Line and load regulation Line regulation + load regulation = (0.17 mV + 7 mV)/ 1.5 V × 100%	Line and Load Regulation = 0.478% < 5%
Minimum input voltage ($V_{IN(MIN)}$) ($V_{IN(MIN)}$) = $V_{IN}(1 – \Delta V_{IN})$ = 3.3V(1 – 0.05)	($V_{IN(MIN)}$) = 3.135 V
Maximum input current $I_{IN, DC(MAX)} = (V_{OUT} \times I_{OUT(MAX)})/(\eta \times V_{IN(MIN)})$	$I_{IN, DC(MAX)}$ = 478 mA < 2 A
EL7562C	
Nominal efficiency (η)	Nominal efficiency (η) = > 95%
Line and load regulation Line regulation + load regulation = (0.17 mV + 7 mV)/ 1.5 V × 100%	Line and Load Regulation = 0.5% < 5%
Minimum input voltage ($V_{IN(MIN)}$) ($V_{IN(MIN)}$) = $V_{IN}(1 – \Delta V_{IN})$ = 3.3V(1 – 0.05)	($V_{IN(MIN)}$) = 3.135 V
Maximum input current $I_{IN, DC(MAX)} = (V_{OUT} \times I_{OUT(MAX)})/(\eta \times V_{IN(MIN)})$	$I_{IN, DC(MAX)}$ = 453 mA < 2 A

Board Layout

Laying out a printed circuit board (PCB) properly is extremely important in high-frequency (≥100 kHz) switching regulator designs. A poor PCB layout results in increased EMI and ground bounce, which affects the reliability of the voltage regulator by obscuring important voltage and current feedback signals. Altera recommends using Gerber files —pre-designed layout files—supplied by the regulator vendor for your board layout.

If you cannot use the supplied layout files, contact the voltage regulator vendor for help on re-designing the board to fit your design requirements while maintaining the proper functionality.

Altera recommends that you use separate layers for signals, the ground plane, and voltage supply planes. You can support separate layers by using multi-layer PCBs, assuming you are using two signal layers.

Figure 14–18 shows how to use regulators to generate 1.5-V and 2.5-V power supplies if the system needs two power supply systems. One regulator is used for each power supply.

Figure 14–18. Two Regulator Solution for Systems that Require 5.0-V, 2.5-V & 1.5-V Supply Levels

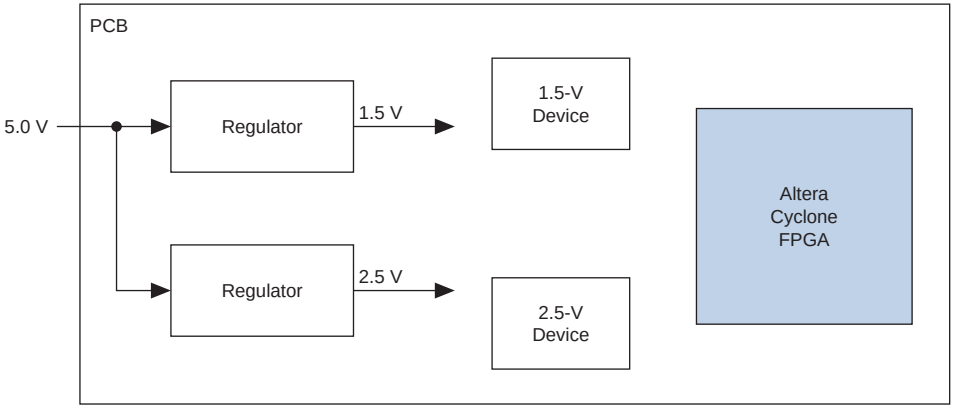
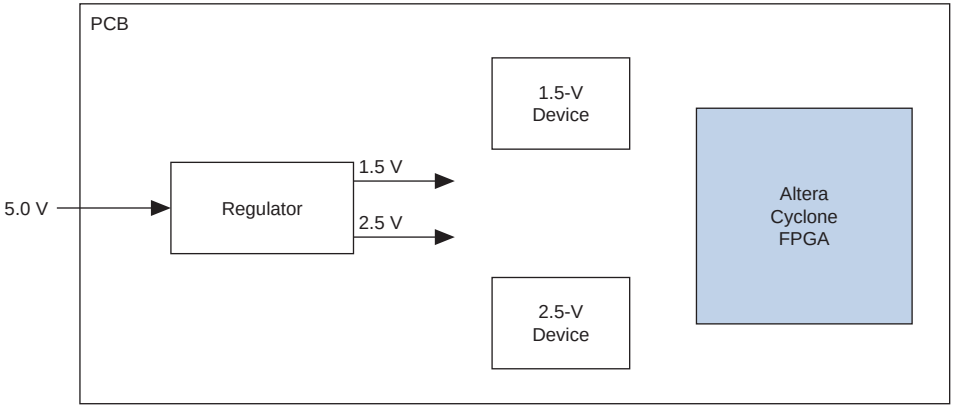


Figure 14–19 shows how to use a single regulator to generate two different power supplies (1.5-V and 2.5-V). The use of a single regulator to generate 1.5-V and 2.5-V supplies from the 5.0-V power supply can minimize the board size and thus save cost.

Figure 14–19. Single Regulator Solution for Systems that Require 5.0-V, 2.5-V & 1.5-V Supply Levels



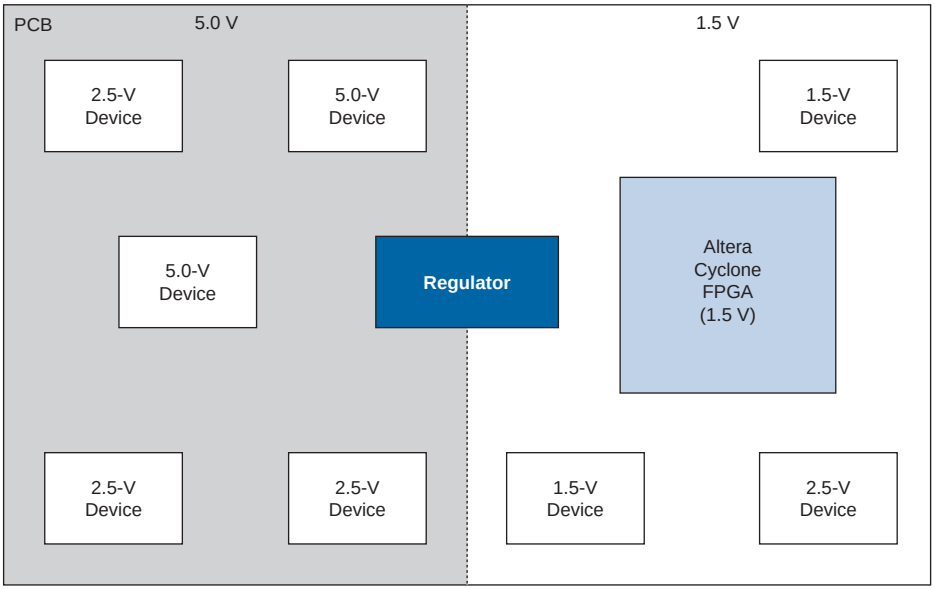
Split-Plane Method

The split-plane design method reduces the number of planes required by placing two power supply planes in one plane (see Figure 14–20). For example, the layout for this method can be structured as follows:

- One 2.5-V plane, covering the entire board
- One plane split between 5.0-V and 1.5-V

This technique assumes that the majority of devices are 2.5-V. To support MultiVolt I/O, Altera devices must have access to 1.5-V and 2.5-V planes.

Figure 14–20. Split Board Layout for 2.5-V Systems With 5.0-V & 1.5-V Devices



Conclusion

With the proliferation of multiple voltage levels in systems, it is important to design a voltage system that can support a low-power device like Cyclone devices. Designers must consider key elements of the PCB, such as power supplies, regulators, power consumption, and board layout when successfully designing a system that incorporates the low-voltage Cyclone family of devices.

References

Linear Technology Corporation. *Application Note 35 (Step Down Switching Regulators)*. Milpitas: Linear Technology Corporation, 1989.

Linear Technology Corporation. *LT1573 Data Sheet (Low Dropout Regulator Driver)*. Milpitas: Linear Technology Corporation, 1997.

Linear Technology Corporation. *LT1083/LT1084/LT1085 Data Sheet (7.5 A, 5 A, 3 A Low Dropout Positive Adjustable Regulators)*. Milpitas: Linear Technology Corporation, 1994.

Linear Technology Corporation. *LTC1649 Data Sheet (3.3V Input High Power Step-Down Switching Regulator Controller)*. Milpitas: Linear Technology Corporation, 1998.

Linear Technology Corporation. *LTC1775 Data Sheet (High Power No Rsense Current Mode Synchronous Step-Down Switching Regulator)*. Milpitas: Linear Technology Corporation, 1999.

Intersil Corporation. *EL7551C Data Sheet (Monolithic 1 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.

Intersil Corporation. *EL7564C Data Sheet (Monolithic 4 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.

Intersil Corporation. *EL7556BC Data Sheet (Integrated Adjustable 6 Amp Synchronous Switcher)*. Milpitas: Intersil Corporation, 2001.

Intersil Corporation. *EL7562C Data Sheet (Monolithic 2 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.

Intersil Corporation. *EL7563C Data Sheet (Monolithic 4 Amp DC:DC Step-Down Regulator)*. Milpitas: Intersil Corporation, 2002.