Stratix II Device Handbook, Volume 2

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

SII5V2-2.2

I.S. EN ISO 9001

Contents

Section II. Memory

Chapter 2. TriMatrix Embedded Memory Blocks in Stratix II Devices

Chapter Revision Dates

The chapters in this book, *Stratix II Device Handbook, Volume 2*, were revised on the following dates. Where chapters or groups of chapters are available separately, part numbers are listed.

Chapter 1. PLLs in Stratix II Devices
Revised: *March 2005*
Part number: *SII52001-2.2*

Chapter 2. TriMatrix Embedded Memory Blocks in Stratix II Devices
Revised: *March 2005*
Part number: *SII52002-2.1*

Chapter 3. External Memory Interfaces
Revised: *March 2005*
Part number: *SII52003-2.2*

Chapter 4. Selectable I/O Standards in Stratix II Devices
Revised: *January 2005*
Part number: *SII52004-2.0*

Chapter 5. High-Speed Differential I/O Interfaces with DPA in Stratix II Devices
Revised: *January 2005*
Part number: *SII52005-2.0*

Chapter 6. DSP Blocks in Stratix II Devices
Revised: *July 2004*
Part number: *SII52006-1.1*

Chapter 7. Configuring Stratix II Devices
Revised: *January 2005*
Part number: *SII52007-2.1*

Chapter 8. Remote System Upgrades with Stratix II Devices
Revised: *January 2005*
Part number: *SII52008-2.0*

Chapter 9. IEEE 1149.1 (JTAG) Boundary-Scan Testing for Stratix II Devices
Revised: *January 2005*
Part number: *SII52009-2.0*

Chapter 10. Package Information for Stratix II Devices

Revised:	*January 2005*
Part number:	*SII52010-2.0*

Chapter 11. High-Speed Board Layout Guidelines

Revised:	*March 2005*
Part number:	*SII52012-1.2*

About this Handbook

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

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 Design.*

Visual Cue	Meaning
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.
✓	The checkmark indicates a procedure that consists of one step only.
☞	The hand points to information that requires special attention.
⚠ CAUTION	The caution indicates required information that needs special consideration and understanding and should be read prior to starting or continuing with the procedure or process.
⚠	The warning indicates information that should be read prior to starting or continuing the procedure or processes
↵	The angled arrow indicates you should press the Enter key.
👣	The feet direct you to more information on a particular topic.

Section I. Clock Management

This section provides information on the different types of phase-locked loops (PLLs). The feature-rich enhanced PLLs assist designers in managing clocks internally and also have the ability to drive off chip to control system-level clock networks. The fast PLLs offer general-purpose clock management with multiplication and phase shifting as well as high-speed outputs to manage the high-speed differential I/O interfaces. This section contains detailed information on the features, the interconnections to the logic array and off chip, and the specifications for both types of PLLs.

This section contains the following chapter:

- Chapter 1, PLLs in Stratix II Devices

Revision History

The table below shows the revision history for Chapter 1.

Chapter	Date / Version	Changes Made
1	March 2005, v2.2	Minor content updates.
	January 2005, v2.1	Minor content updates.
	January 2005, v2.0	● Updated Figures 1–37 and 1–46. ● Updated Tables 1–10 and 1–21.
	October 2004, v1.2	● Updated the "Introduction" section. ● Updated Table 1–1. ● Updated "Clock Output Connections" section. ● Updated Table 1–21.
	July 2004, v1.1	● Updated Tables 1–1, 1–2, 1–3, 1–5, 1–7, 1–9, 1–16. ● Updated Figures 1–5, 1–7, 1–47, and 1–48. ● Updated "Enhanced Lock Detect Circuit" section.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.

1. PLLs in Stratix II Devices

SII52001-2.2

Introduction

Stratix® II devices have up to 12 phase-locked loops (PLLs) that provide robust clock management and synthesis for device clock management, external system clock management, and high-speed I/O interfaces. Stratix II PLLs are highly versatile and can be used as a zero delay buffer, a jitter attenuator, low skew fan out buffer, or a frequency synthesizer.

Stratix II devices feature both enhanced PLLs and fast PLLs. Each device has up to four enhanced PLLs and up to eight fast PLLs. Both enhanced and fast PLLs are feature rich, supporting advanced capabilities such as clock switchover, reconfigurable phase shift, PLL reconfiguration, and reconfigurable bandwidth. PLLs can be used for general-purpose clock management, supporting multiplication, phase shifting, and programmable duty cycle. In addition, enhanced PLLs support external clock feedback mode, spread-spectrum clocking, and counter cascading. Fast PLLs offer high speed outputs to manage the high-speed differential I/O interfaces.

Stratix II devices also support a power-down mode where clock networks that are not being used can easily be turned off, reducing the overall power consumption of the device. In addition, Stratix II PLLs support dynamic selection of the PLL input clock from up to five possible sources, giving you the flexibility to choose from multiple (up to four) clock sources to feed the primary and secondary clock input ports.

The Altera® Quartus® II software enables the PLLs and their features without requiring any external devices.

Table 1–1 shows the PLLs available for each Stratix II device.

Table 1–1. Stratix II Device PLL Availability *Note (1)*

Device	Fast PLLs								Enhanced PLLs			
	1	2	3	4	7	8	9	10	5	6	11	12
EP2S15	✓	✓	✓	✓					✓	✓		
EP2S30	✓	✓	✓	✓					✓	✓		
EP2S60	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP2S90 *(2)*	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP2S130 *(3)*	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EP2S180	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Notes for Table 1–1:

(1) The EP2S60 device in the 1,020-pin package contains 12 PLLs. EP2S60 devices in the 484-pin and 672-pin packages contain fast PLLs 1–4 and enhanced PLLs 5 and 6.

(2) EP2S90 devices in the 1020-pin and 1508-pin packages contain 12 PLLs. EP2S90 devices in the 484-pin and 780-pin packages contain fast PLLs 1–4 and enhanced PLLs 5 and 6.

(3) EP2S130 devices in the 1020-pin and 1508-pin packages contain 12PLLs. The EP2S130 device in the 780-pin package contains fast PLLs 1–4 and enhanced PLLs 5 and 6.

Table 1–2 shows the enhanced PLL and fast PLL features in Stratix II devices.

Table 1–2. Stratix II PLL Features

Feature	Enhanced PLL	Fast PLL
Clock multiplication and division	*m/(n ×* post-scale counter) *(1)*	*m/(n ×* post-scale counter) *(2)*
Phase shift	Down to 125-ps increments *(3)*	Down to 125-ps increments *(3)*
Clock switchover	✓	✓ *(4)*
PLL reconfiguration	✓	✓
Reconfigurable bandwidth	✓	✓
Spread-spectrum clocking	✓	
Programmable duty cycle	✓	✓
Number of clock outputs per PLL *(5)*	6	4
Number of dedicated external clock outputs per PLL	Three differential or six singled-ended	*(6)*
Number of feedback clock inputs per PLL	1 *(7)*	

Notes to Table 1–2:

(1) For enhanced PLLs, *m* and *n* range from 1 to 512 and post-scale counters range from 1 to 512 with 50% duty cycle. For non-50% duty-cycle clock outputs, post-scale counters range from 1 to 256.

(2) For fast PLLs, *n* can range from 1 to 4. The post-scale and *m* counters range from 1 to 32. For non-50% duty-cycle clock outputs, post-scale counters range from 1 to 16.

(3) The smallest phase shift is determined by the voltage controlled oscillator (VCO) period divided by eight. The supported phase-shift range is from 125- to 250-ps. Stratix II devices can shift all output frequencies in increments of at least 45°. Smaller degree increments are possible depending on the frequency and divide parameters. For non-50% duty cycle clock outputs post-scale counters range from 1 to 256.

(4) Stratix II fast PLLs only support manual clock switchover.

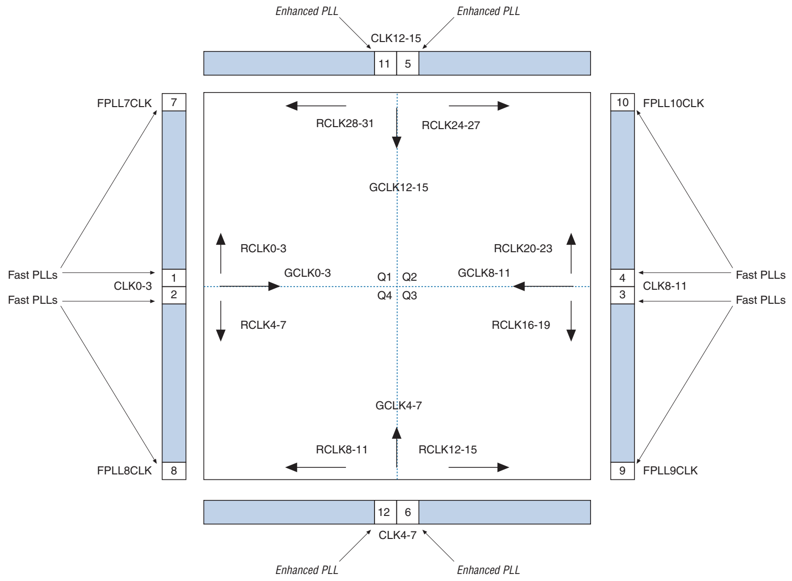
(5) The clock outputs can be driven to internal clock networks or to a pin.

(6) The PLL clock outputs of the fast PLLs can drive to any I/O pin to be used as an external clock output. For high-speed differential I/O pins, the device uses a data channel to generate the transmitter output clock (`txclkout`).

(7) If the design uses external feedback input pins, you will lose one (or two, if f_{BIN} is differential) dedicated output clock pin.

Figure 1–1 shows a top-level diagram of Stratix II device and PLL locations. See "Clock Control Block" on page 1–76 for more detail on PLL connections to global and regional clocks networks.

Figure 1–1. Stratix II PLL Locations



Enhanced PLLs

Stratix II devices contain up to four enhanced PLLs with advanced clock management features. The main goal of a PLL is to synchronize the phase and frequency of an internal and external clock to an input reference clock. There are a number of components that comprise a PLL to achieve this phase alignment.

Enhanced PLL Hardware Overview

Stratix II PLLs align the rising edge of the reference input clock to a feedback clock using the phase-frequency detector (PFD). The falling edges are determined by the duty-cycle specifications. The PFD produces an up or down signal that determines whether the VCO needs to operate at a higher or lower frequency.

The PFD output is applied to the charge pump and loop filter, which produces a control voltage for setting the VCO frequency. If the PFD produces an up signal, then the VCO frequency increases. A down signal decreases the VCO frequency. The PFD outputs these up and down signals to a charge pump. If the charge pump receives an up signal, current is driven into the loop filter. Conversely, if it receives a down signal, current is drawn from the loop filter.

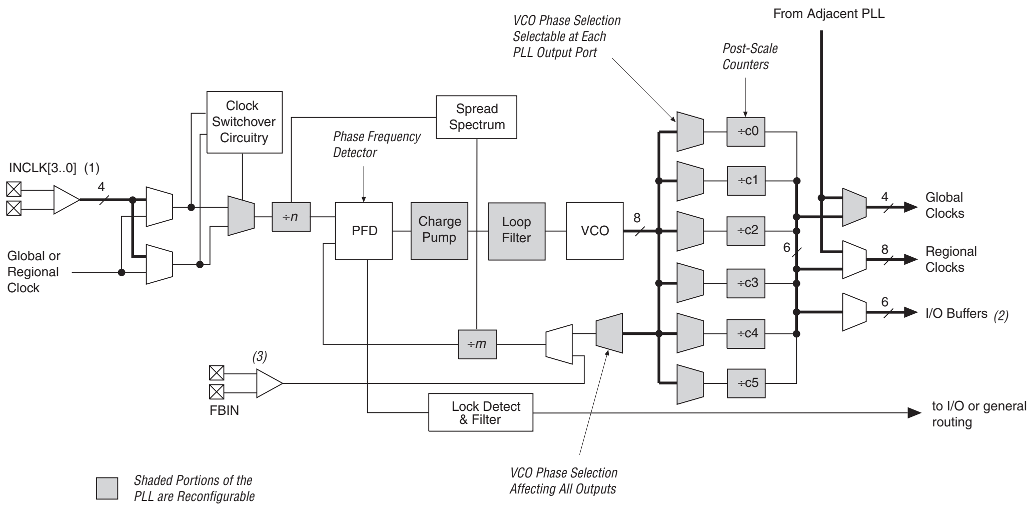
The loop filter converts these up and down signals to a voltage that is used to bias the VCO. The loop filter also removes glitches from the charge pump and prevents voltage over-shoot, which filters the jitter on the VCO.

The voltage from the loop filter determines how fast the VCO operates. The VCO is implemented as a four-stage differential ring oscillator. A divide counter (*m*) is inserted in the feedback loop to increase the VCO frequency above the input reference frequency. VCO frequency (f_{VCO}) is equal to (*m*) times the input reference clock (f_{REF}). The input reference clock (f_{REF}) to the PFD is equal to the input clock (f_{IN}) divided by the pre-scale counter (*n*). Therefore, the feedback clock (f_{FB}) applied to one input of the PFD is locked to the f_{REF} that is applied to the other input of the PFD.

The VCO output can feed up to six post-scale counters (`C0`, `C1`, `C2`, `C3`, `C4`, and `C5`). These post-scale counters allow a number of harmonically related frequencies to be produced within the PLL.

Figure 1–2 shows a simplified block diagram of the major components of the Stratix II enhanced PLL. Figure 1–3 shows the enhanced PLL's outputs and dedicated clock outputs.

Figure 1–2. Stratix II Enhanced PLL



Notes to Figure 1–2:

(1) Each clock source can come from any of the four clock pins located on the same side of the device as the PLL.

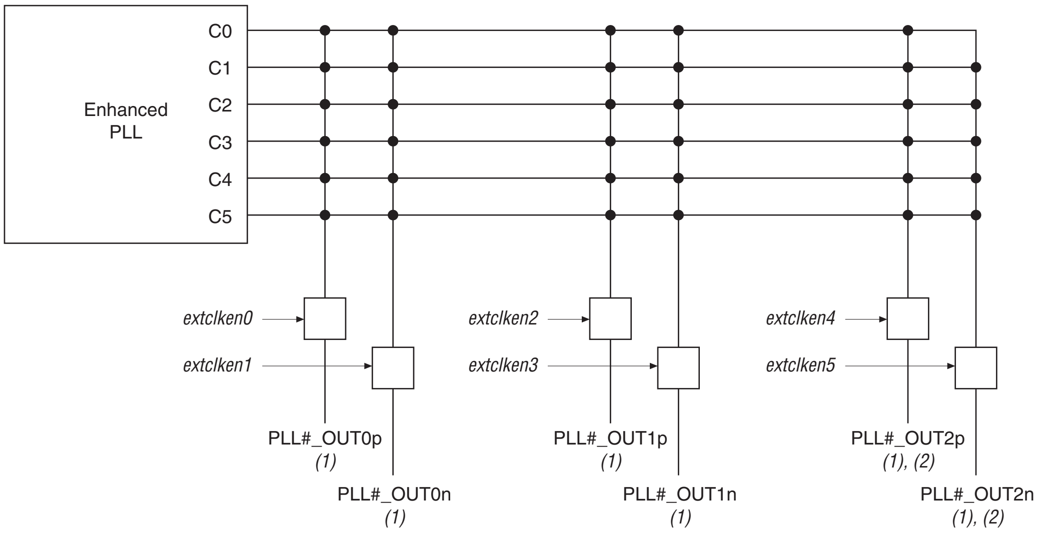
(2) PLLs 5, 6, 11, and 12 each have six single-ended dedicated clock outputs or three differential dedicated clock outputs.

(3) If the design uses external feedback input pins, you will lose one (or two, if f_{BIN} is differential) dedicated output clock pin. Every Stratix II device has at least two enhanced PLLs with one single-ended or differential external feedback input per PLL.

External Clock Outputs

Enhanced PLLs 5, 6, 11, and 12 each support up to six single-ended clock outputs (or three differential pairs). See Figure 1–3.

Figure 1–3. External Clock Outputs for Stratix II PLLs 5, 6, 11 & 12



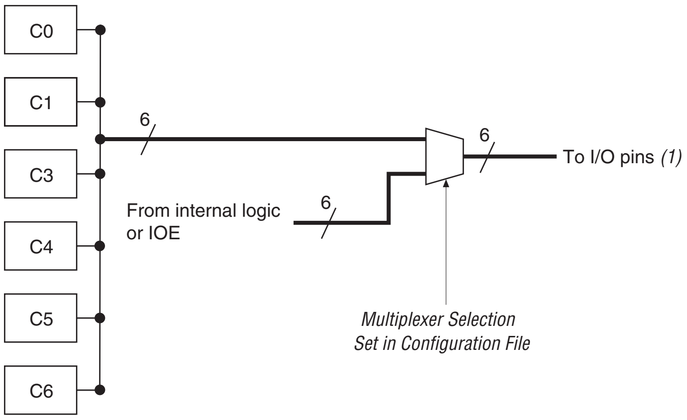
Notes to Figure 1–3:

(1) These clock output pins can be fed by any one of the `C[5..0]` counters.

(2) These clock output pins are used as either external clock outputs or for external feedback. If the design uses external feedback input pins, you will lose one (or two, if f_{BIN} is differential) dedicated output clock pin.

Any of the six output counters `C[5..0]` can feed the dedicated external clock outputs, as shown in Figure 1–4. Therefore, one counter or frequency can drive all output pins available from a given PLL. The dedicated output clock pins (`PLL_OUT`) from each enhanced PLL are powered by a separate power pin (e.g., `VCC_PLL5_OUT`, `VCC_PLL6_OUT`, etc.), reducing the overall output jitter by providing improved isolation from switching I/O pins.

Figure 1–4. External Clock Output Connectivity to PLL Output Counters for Stratix II PLLs 5, 6, 11 & 12 *Note (1)*


C0
C1
C3
C4
C5
C6
6
6
6
From internal logic or IOE
To I/O pins (1)
Multiplexer Selection Set in Configuration File

Note to Figure 1–4:
(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 I/O element (IOE) outputs.

Each pin of a single-ended output pair can either be in phase or 180° out of phase. The Quartus II software places the `NOT` gate in the design into the IOE to implement 180° phase with respect to the other pin in the pair. 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, PCML, HyperTransport™ technology, differential HSTL, and differential SSTL. See Table 1–5, in the "Enhanced PLL Pins" section on page 1–11 to determine which I/O standards the enhanced PLL clock pins support.

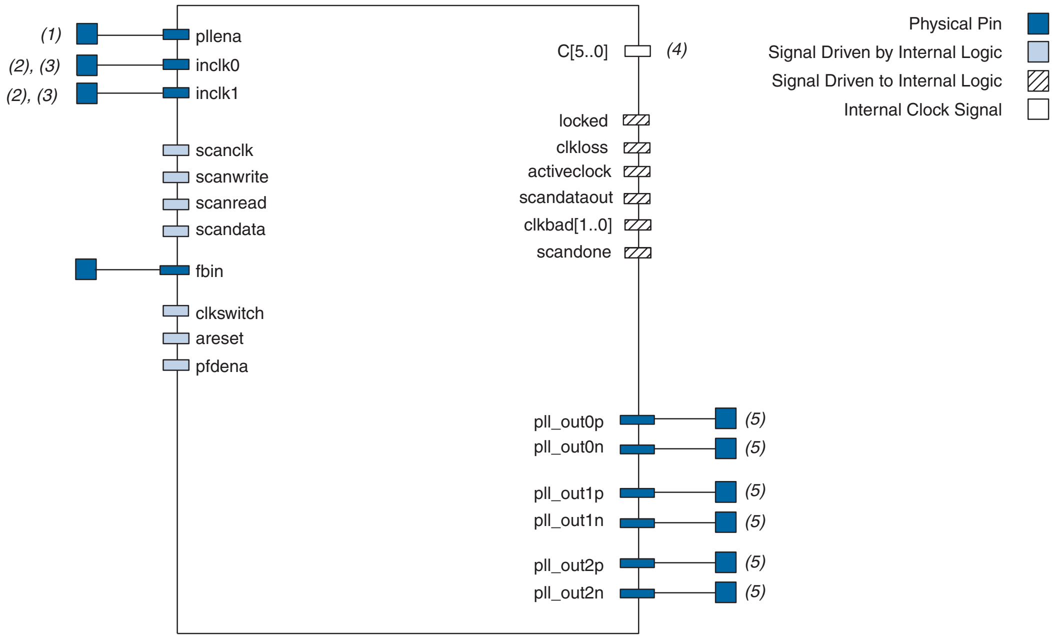
When in single-ended or differential mode, one power pin supports six single-ended or three differential outputs. Both outputs use the same I/O standard in single-ended mode to maintain performance. You can also use the external clock output pins as user output pins if external enhanced PLL clocking is not needed.

The enhanced PLL can also drive out to any regular I/O pin through the global or regional clock network. For this case, jitter on the output clock is pending characterization

Enhanced PLL Software Overview

Stratix II enhanced PLLs are enabled in the Quartus II software by using the `altpll` megafunction. Figure 1–5 shows the available ports (as they are named in the Quartus II `altpll` megafunction) of the Stratix II enhanced PLL.

Figure 1–5. Stratix II Enhanced PLL Ports



Notes to Figure 1–5:
(1) Enhanced and fast PLLs share this input pin.
(2) These are either single-ended or differential pins.
(3) The primary and secondary clock input can be fed from any one of four clock pins located on the same side of the device as the PLL.
(4) Can drive to the global or regional clock networks or the dedicated external clock output pins.
(5) These dedicated output clocks are fed by the `C[5..0]` counters.

Tables 1–3 and 1–4 describe all the enhanced PLL ports.

Table 1–3. Enhanced PLL Input Signals (Part 1 of 2)

Port	Description	Source	Destination
`inclk0`	Primary clock input to the PLL.	Pin, another PLL, or internal logic	*n* counter
`inclk1`	Secondary clock input to the PLL.	Pin, another PLL, or internal logic	*n* counter
`fbin`	External feedback input to the PLL.	Pin	PFD
`pllena`	Enable pin for enabling or disabling all or a set of PLLs. Active high.	Pin	General PLL control signal

Table 1–3. Enhanced PLL Input Signals (Part 2 of 2)

Port	Description	Source	Destination
`clkswitch`	Switch-over signal used to initiate external clock switch-over control. Active high.	Logic array	PLL switch-over circuit
`areset`	Signal used to reset the PLL which resynchronizes all the counter outputs. Active high.	Logic array	General PLL control signal
`pfdena`	Enables the outputs from the phase frequency detector. Active high.	Logic array	PFD
`scanclk`	Serial clock signal for the real-time PLL reconfiguration feature.	Logic array	Reconfiguration circuit
`scandata`	Serial input data stream for the real-time PLL reconfiguration feature.	Logic array	Reconfiguration circuit
`scanwrite`	Enables writing the data in the scan chain into the PLL. Active high.	Logic array	Reconfiguration circuit
`scanread`	Enables scan data to be written into the scan chain. Active high.	Logic array	Reconfiguration circuit

Table 1–4. Enhanced PLL Output Signals (Part 1 of 2)

Port	Description	Source	Destination
`c[5..0]`	PLL output counters driving regional, global or external clocks.	PLL counter	Internal or external clock

`pll_out [2..0]p`
`pll_out [2..0]n`
	These are three differential or six single-ended external clock output pins fed from the `C[5..0]` PLL counters. *p* and *n* are the positive (*p*) and negative (*n*) pins for differential pins.	PLL counter	Pin(s)

`clkloss`	Signal indicating the switch-over circuit detected a switch-over condition.	PLL switch-over circuit	Logic array
`clkbad[1..0]`	Signals indicating which reference clock is no longer toggling. `clkbad1` indicates `inclk1` status, `clkbad0` indicates `inclk0` status. 1= good; 0=bad	PLL switch-over circuit	Logic array
`locked`	Lock or gated lock output from lock detect circuit. Active high.	PLL lock detect	Logic array

Table 1–4. Enhanced PLL Output Signals (Part 2 of 2)

Port	Description	Source	Destination
`activeclock`	Signal to indicate which clock (`0 = inclk0` or `1 = inclk1`) is driving the PLL. If this signal is low, `inclk0` drives the PLL, If this signal is high, `inclk1` drives the PLL	PLL clock multiplexer	Logic array
`scandataout`	Output of the last shift register in the scan chain.	PLL scan chain	Logic array
`scandone`	Signal indicating when the PLL has completed reconfiguration. 1 to 0 transition indicates that the PLL has been reconfigured.	PLL scan chain	Logic array

Enhanced PLL Pins

Table 1–5 lists the I/O standards support by the enhanced PLL clock outputs.

Table 1–5. I/O Standards Supported for Stratix II Enhanced PLL Pins (Part 1 of 2) Note (1)

I/O Standard	Input		Output
	INCLK	FBIN	EXTCLK
LVTTL	✓	✓	✓
LVCMOS	✓	✓	✓
2.5 V	✓	✓	✓
1.8 V	✓	✓	✓
1.5 V	✓	✓	✓
3.3-V PCI	✓	✓	✓
3.3-V PCI-X	✓	✓	✓
SSTL-2 class I	✓	✓	✓
SSTL-2 class II	✓	✓	✓
SSTL-18 class I	✓	✓	✓
SSTL-18 class II	✓	✓	✓
1.8-V HSTL class I	✓	✓	✓
1.8-V HSTL class II	✓	✓	✓

Table 1–5. I/O Standards Supported for Stratix II Enhanced PLL Pins (Part 2 of 2) Note (1)

I/O Standard	Input		Output
	INCLK	FBIN	EXTCLK
1.5-V HSTL class I	✓	✓	✓
1.5-V HSTL class II	✓	✓	✓
Differential SSTL-2 class I	✓	✓	✓
Differential SSTL-2 class II	✓	✓	✓
Differential SSTL-18 class I	✓	✓	✓
Differential SSTL-18 class II	✓	✓	✓
1.8-V differential HSTL class I	✓	✓	✓
1.8-V differential HSTL class II	✓	✓	✓
1.5-V differential HSTL class I	✓	✓	✓
1.5-V differential HSTL class II	✓	✓	✓
LVDS	✓	✓	✓
HyperTransport technology	✓	✓	✓
Differential LVPECL	✓	✓	✓

Note to Table 1–5:
(1) The enhanced PLL external clock output bank does not allow a mixture of both single-ended and differential I/O standards.

Table 1–6 shows the physical pins and their purpose for the Stratix II enhanced PLLs. For `inclk` port connections to pins see "Clock Control Block" on page 1–76.

Table 1–6. Stratix II Enhanced PLL Pins (Part 1 of 2)

Pin	Description
`CLK4p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 6 or 12.
`CLK5p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 6 or 12.
`CLK6p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 6 or 12.
`CLK7p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 6 or 12.
`CLK12p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 5 or 11.
`CLK13p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 5 or 11.
`CLK14p/n`	Single-ended or differential pins that can drive the `inclk` port for PLLs 5 or 11.

Table 1–6. Stratix II Enhanced PLL Pins (Part 2 of 2)

Pin	Description
CLK15p/n	Single-ended or differential pins that can drive the inclk port for PLLs 5 or 11.
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.
PLL11_FBp/n	Single-ended or differential pins that can drive the fbin port for PLL 11.
PLL12_FBp/n	Single-ended or differential pins that can drive the fbin port for PLL 12.
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.
PLL5_OUT[2..0]p/n	Single-ended or differential pins driven by C[5..0] ports from PLL 5.
PLL6_OUT[2..0]p/n	Single-ended or differential pins driven by C[5..0] ports from PLL 6.
PLL11_OUT[2..0]p/n	Single-ended or differential pins driven by C[5..0] ports from PLL 11.
PLL12_OUT[2..0]p/n	Single-ended or differential pins driven by C[5..0] ports from PLL 12.
VCCA_PLL5	Analog power for PLL 5. You must connect this pin to 1.2 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.
VCCA_PLL6	Analog power for PLL 6. You must connect this pin to 1.2 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.
VCCA_PLL11	Analog power for PLL 11. You must connect this pin to 1.2 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.
VCCA_PLL12	Analog power for PLL 12. You must connect this pin to 1.2 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.
VCCD_PLL	Digital power for PLLs. You must connect this pin to 1.2 V, even if the PLL is not used.
VCC_PLL5_OUT	External clock output V_{CCIO} power for PLL5_OUT0p, PLL5_OUT0n, PLL5_OUT1p, PLL5_OUT1n, PLL5_OUT2p, and PLL5_OUT2n outputs from PLL 5.
VCC_PLL6_OUT	External clock output V_{CCIO} power for PLL5_OUT0p, PLL5_OUT0n, PLL5_OUT1p, PLL5_OUT1n and PLL5_OUT2p, PLL5_OUT2n outputs from PLL 6.
VCC_PLL11_OUT	External clock output V_{CCIO} power for PLL5_OUT0p, PLL5_OUT0n, PLL5_OUT1p, PLL5_OUT1n and PLL5_OUT2p, PLL5_OUT2n outputs from PLL 11.
VCC_PLL12_OUT	External clock output V_{CCIO} power for PLL5_OUT0p, PLL5_OUT0n, PLL5_OUT1p, PLL5_OUT1n and PLL5_OUT2p, PLL5_OUT2n outputs from PLL 12.

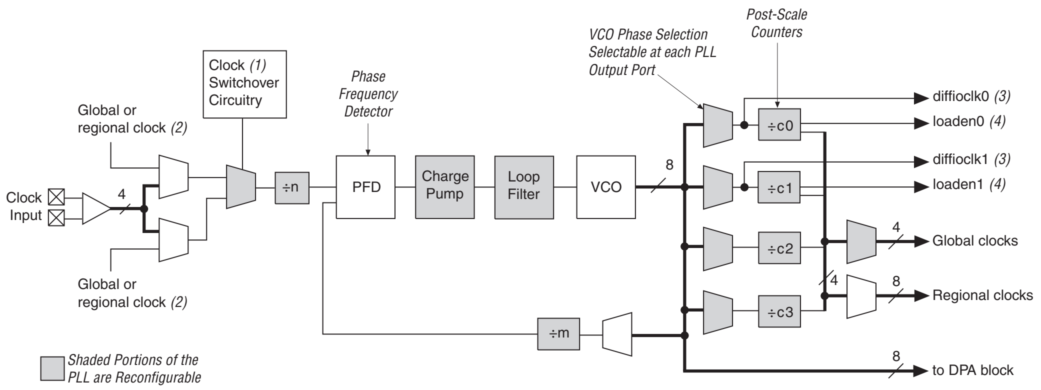
Fast PLLs

Stratix II devices contain up to eight fast PLLs with high-speed differential I/O interface capability along with general-purpose features.

Fast PLL Hardware Overview

Figure 1–6 shows a diagram of the fast PLL.

Figure 1–6. Stratix II Fast PLL Block Diagram



Notes to Figure 1–6:

(1) Stratix II fast PLLs only support manual clock switchover.

(2) The global or regional clock input can be driven by an output from another PLL, a pin-driven global or regional clock or internally-generated global signals.

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

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

External Clock Outputs

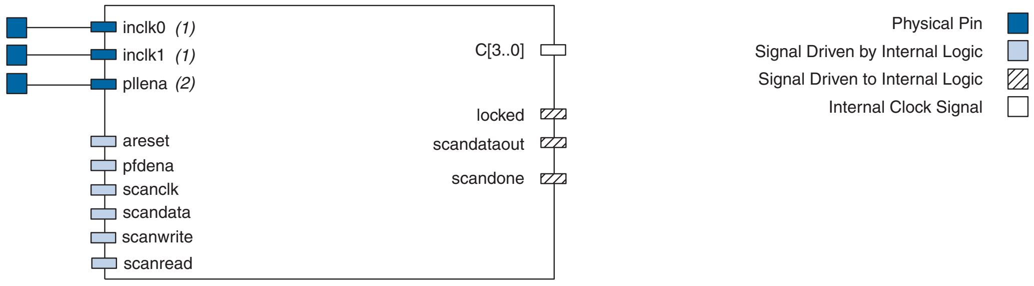
Each fast PLL supports differential or single-ended outputs for source-synchronous transmitters or for general-purpose external clocks. There are no dedicated external clock output pins. The fast PLL global or regional outputs can drive any I/O pin as an external clock output pin. The I/O standards supported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

For more information, see the *Selectable I/O Standards* chapter in Volume 2 of the *Stratix II Device Handbook.*

Fast PLL Software Overview

Stratix II fast PLLs are enabled in the Quartus II software by using the `altpll` megafunction. Figure 1–7 shows the available ports (as they are named in the Quartus II `altpll` megafunction) of the Stratix II fast PLL.

Figure 1–7. Stratix II Fast PLL Ports & Physical Destinations



Notes to Figure 1–7:

(1) This input pin is either single-ended or differential.

(2) This input pin is shared by all enhanced and fast PLLs.

Tables 1–7 and 1–8 show the description of all fast PLL ports.

Table 1–7. Fast PLL Input Signals

Name	Description	Source	Destination
`inclk0`	Primary clock input to the fast PLL.	Pin, another PLL, or internal logic	*n* counter
`inclk1`	Secondary clock input to the fast PLL.	Pin, another PLL, or internal logic	*n* counter
`pllena`	Enable pin for enabling or disabling all or a set of PLLs Active high.	Pin	PLL control signal
`clkswitch`	Switch-over signal used to initiate external clock switch-over control. Active high.	Logic array	Reconfiguration circuit
`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
`scanclk`	Serial clock signal for the real-time PLL control feature.	Logic array	Reconfiguration circuit
`scandata`	Serial input data stream for the real-time PLL control feature.	Logic array	Reconfiguration circuit
`scanwrite`	Enables writing the data in the scan chain into the PLL Active high.	Logic array	Reconfiguration circuit
`scanread`	Enables scan data to be written into the scan chain Active high.	Logic array	Reconfiguration circuit

Table 1–8. Fast PLL Output Signals

Name	Description	Source	Destination
`c[3..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
`scandataout`	Output of the last shift register in the scan chain.	PLL scan chain	Logic array
`scandone`	Signal indicating when the PLL has completed reconfiguration. 1 to 0 transition indicates the PLL has been reconfigured.	PLL scan chain	Logic array

Fast PLL Pins

Table 1–9 shows the I/O standards supported by the fast PLL input pins.

Table 1–9. I/O Standards Supported for Stratix II Fast PLL Pins (Part 1 of 2)

I/O Standard	INCLK
LVTTL	✓
LVCMOS	✓
2.5 V	✓
1.8 V	✓
1.5 V	✓
3.3-V PCI	
3.3-V PCI-X	
SSTL-2 class I	✓
SSTL-2 class II	✓
SSTL-18 class I	✓
SSTL-18 class II	✓
1.8-V HSTL class I	✓
1.8-V HSTL class II	✓
1.5-V HSTL class I	✓
1.5-V HSTL class II	✓
Differential SSTL-2 class I	
Differential SSTL-2 class II	
Differential SSTL-18 class I	
Differential SSTL-18 class II	

Table 1–9. I/O Standards Supported for Stratix II Fast PLL Pins (Part 2 of 2)

I/O Standard	INCLK
1.8-V differential HSTL class I	
1.8-V differential HSTL class II	
1.5-V differential HSTL class I	
1.5-V differential HSTL class II	
LVDS	✓
HyperTransport technology	✓
Differential LVPECL	

Table 1–10 shows the physical pins and their purpose for the fast PLLs. For `inclk` port connections to pins, see "Clocking" on page 1–58.

Table 1–10. Fast PLL Pins (Part 1 of 2)

Pin	Description
CLK0p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 1, 2, 7 or 8.
CLK1p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 1, 2, 7 or 8.
CLK2p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 1, 2, 7 or 8.
CLK3p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 1, 2, 7 or 8.
CLK8p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 3, 4, 9 or 10.
CLK9p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 3, 4, 9 or 10.
CLK10p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 3, 4, 9 or 10.
CLK11p/n	Single-ended or differential pins that can drive the `inclk` port for PLLs 3, 4, 9 or 10.
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.
FPLL10CLKp/n	Single-ended or differential pins that can drive the `inclk` port for PLL 10.
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 GND.
VCCD_PLL	Digital power for PLLs. You must connect this pin to 1.2 V, even if the PLL is not used.
VCCA_PLL1	Analog power for PLL 1. You must connect this pin to 1.2 V, even if the PLL is not used.
GNDA_PLL1	Analog ground for PLL 1. Your can connect this pin to the GND plane on the board.
VCCA_PLL2	Analog power for PLL 2. You must connect this pin to 1.2 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.

Table 1–10. Fast PLL Pins (Part 2 of 2)

Pin	Description
VCCA_PLL3	Analog power for PLL 3. You must connect this pin to 1.2 V, even if the PLL is not used.
GNDA_PLL3	Analog ground for PLL 3. You can connect this pin to the GND plane on the board.
VCCA_PLL4	Analog power for PLL 4. You must connect this pin to 1.2 V, even if the PLL is not used.
GNDA_PLL4	Analog ground for PLL 4. You can connect this pin to the GND plane on the board.
GNDA_PLL7	Analog ground for PLL 7. You can connect this pin to the GND plane on the board.
VCCA_PLL8	Analog power for PLL 8. You must connect this pin to 1.2 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.
VCCA_PLL9	Analog power for PLL 9. You must connect this pin to 1.2 V, even if the PLL is not used.
GNDA_PLL9	Analog ground for PLL 9. You can connect this pin to the GND plane on the board.
VCCA_PLL10	Analog power for PLL 10. You must connect this pin to 1.2 V, even if the PLL is not used.
GNDA_PLL10	Analog ground for PLL 10. You can connect this pin to the GND plane on the board.

Clock Feedback Modes

Stratix II PLLs support up to five different clock feedback modes. Each mode allows clock multiplication and division, phase shifting, and programmable duty cycle. Table 1–11 shows which modes are supported by which PLL type.

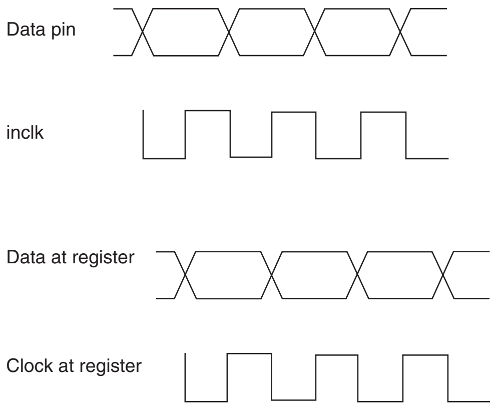
Table 1–11. Clock Feedback Mode Availability

Clock Feedback Mode	Mode Available in	
	Enhanced PLLs	Fast PLLs
Source synchronous mode	Yes	Yes
No compensation mode	Yes	Yes
Normal mode	Yes	Yes
Zero delay buffer mode	Yes	No
External feedback mode	Yes	No

Source-Synchronous Mode

If data and clock arrive at the same time at the input pins, they are guaranteed to keep the same phase relationship at the clock and data ports of any IOE input register. Figure 1–8 shows an example waveform of the clock and data in this mode. This mode is recommended for source-synchronous data transfers. Data and clock signals at the IOE experience similar buffer delays as long as the same I/O standard is used.

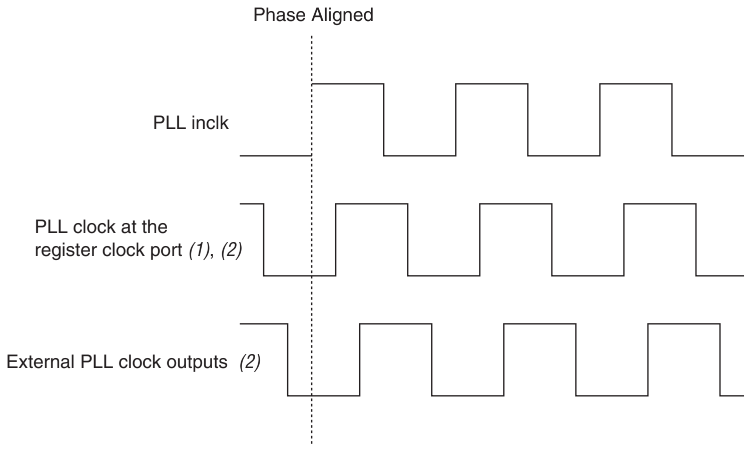
Figure 1–8. Phase Relationship Between Clock & Data in Source-Synchronous Mode



No Compensation Mode

In this mode, the PLL does not compensate for any clock networks. This provides better jitter performance because the clock feedback into the PFD does not pass through as much circuitry. Both the PLL internal and external clock outputs are phase shifted with respect to the PLL clock input. Figure 1–9 shows an example waveform of the PLL clocks' phase relationship in this mode.

Figure 1–9. Phase Relationship between PLL Clocks in No Compensation Mode

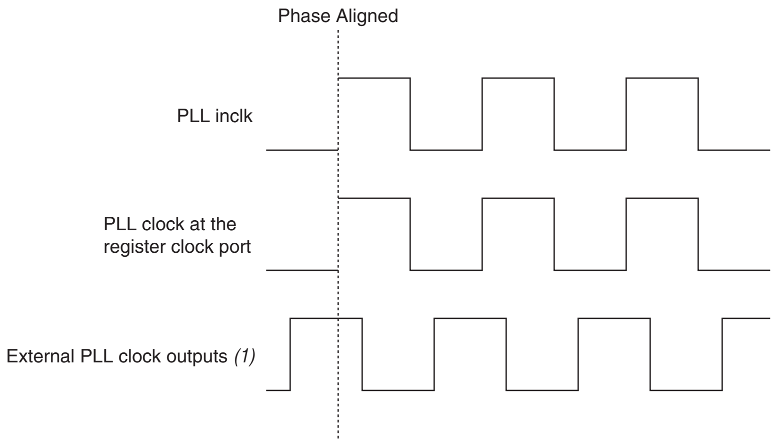

Phase Aligned
PLL inclk
PLL clock at the register clock port (1), (2)
External PLL clock outputs (2)

Notes to Figure 1–9.
(1) Internal clocks fed by the PLL are phase-aligned to each other.
(2) The PLL clock outputs can lead or lag the PLL input clocks.

Normal Mode

An internal clock in normal mode 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. In normal mode, the delay introduced by the GCLK or RCLK network is fully compensated. Figure 1–10 shows an example waveform of the PLL clocks' phase relationship in this mode.

Figure 1–10. Phase Relationship Between PLL Clocks in Normal Mode

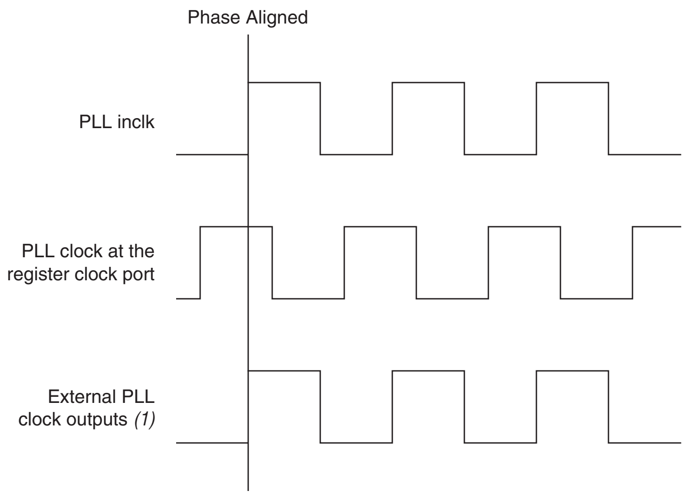

Phase Aligned
PLL inclk
PLL clock at the register clock port
External PLL clock outputs *(1)*

Note to Figure 1–10:
(1) The external clock output can lead or lag the PLL internal clock signals.

Zero Delay Buffer Mode

In the zero delay buffer mode, the external clock output pin is phase-aligned with the clock input pin for zero delay through the device. Figure 1–11 shows an example waveform of the PLL clocks' phase relationship in this mode. When using this mode, Altera requires that you use the same I/O standard on the input clock, and output clocks.

Figure 1–11. Phase Relationship Between PLL Clocks in Zero Delay Buffer Mode


Phase Aligned
PLL inclk
PLL clock at the register clock port
External PLL clock outputs *(1)*

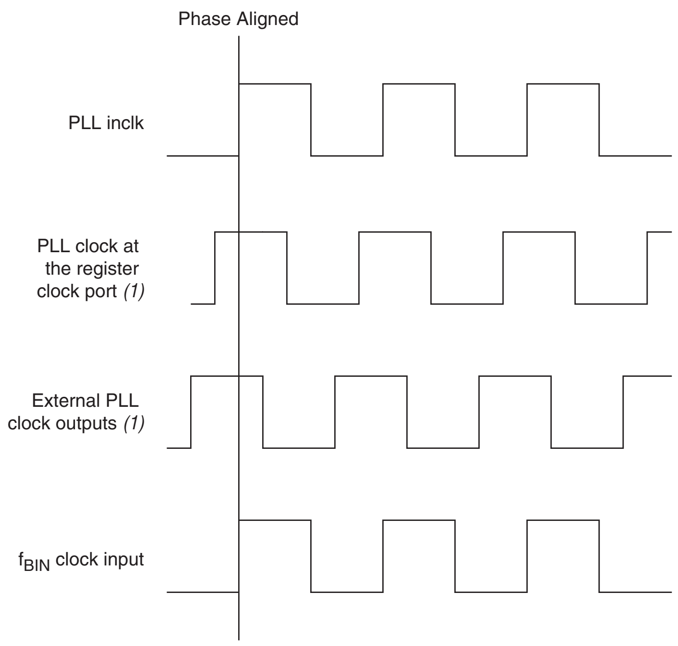
Note to Figure 1–11:

(1) The internal PLL clock output can lead or lag the external PLL clock outputs.

External Feedback Mode

In the external feedback mode, the external feedback input pin, `fbin`, is phase-aligned with the clock input pin, (see Figure 1–12). Aligning these clocks allows you to remove clock delay and skew between devices. This mode is possible on all enhanced PLLs. PLLs 5, 6, 11, and 12 support feedback for one of the dedicated external outputs, either one single-ended or one differential pair. In this mode, one C counter feeds back to the PLL `fbin` input, becoming part of the feedback loop. In this mode, you will be using one of the dedicated external clock outputs (two if a differential I/O standard is used) as the PLL `fbin` input pin. When using this mode, Altera requires that you use the same I/O standard on the input clock, feedback input, and output clocks.

Figure 1–12. Phase Relationship Between PLL Clocks in External Feedback Mode


Phase Aligned
PLL inclk
PLL clock at the register clock port (1)
External PLL clock outputs (1)
f_{BIN} clock input

Note to Figure 1–12:

(1) The PLL clock outputs can lead or lag the f_{BIN} clock input.

Hardware Features

Stratix II PLLs support a number of features for general-purpose clock management. This section discusses clock multiplication and division implementation, phase-shifting implementations and programmable duty cycles. Table 1–12 shows which feature is available in which type of Stratix II PLL.

Table 1–12. Stratix II PLL Hardware Features (Part 1 of 2)

Hardware Features	Availability	
	Enhanced PLL	Fast PLL
Clock multiplication and division	*m* (*n* × post-scale counter)	*m* (*n* × post-scale counter)
m counter value	Ranges from 1 through 512	Ranges from 1 through 32
n counter value	Ranges from 1 through 512	Ranges from 1 through 4
Post-scale counter values	Ranges from 1 through 512 *(1)*	Ranges from 1 through 32 *(2)*

Table 1–12. Stratix II PLL Hardware Features (Part 2 of 2)

Hardware Features	Availability	
	Enhanced PLL	Fast PLL
Phase shift	Down to 125-ps increments *(3)*	Down to 125-ps increments *(3)*
Programmable duty cycle	Yes	Yes

Notes to Table 1–12:

(1) Post-scale counters range from 1 through 512 if the output clock uses a 50% duty cycle. For any output clocks using a non-50% duty cycle, the post-scale counters range from 1 through 256.

(2) Post-scale counters range from 1 through 32 if the output clock uses a 50% duty cycle. For any output clocks using a non-50% duty cycle, the post-scale counters range from 1 through 16.

(3) The smallest phase shift is determined by the VCO period divided by 8. For degree increments, the Stratix II device can shift all output frequencies in increments of at least 45°. Smaller degree increments are possible depending on the frequency and divide parameters.

Clock Multiplication & Division

Each Stratix II 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 factor, n, and is then multiplied by the m feedback factor. The control loop drives the VCO to match f_{IN} (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. For example, if output frequencies required from one PLL are 33 and 66 MHz, then the Quartus II software sets the VCO to 660 MHz (the least common multiple of 33 and 66 MHz within the VCO range). Then, the post-scale counters scale down the VCO frequency for each output port.

There is one pre-scale counter, n, and one multiply counter, m, per PLL, with a range of 1 to 512 for both m and n in enhanced PLLs. For fast PLLs, m ranges from 1 to 32 while n ranges from 1 to 4. There are six generic post-scale counters in enhanced PLLs that can feed regional clocks, global clocks, or external clock outputs, all ranging from 1 to 512 with a 50% duty cycle setting for each PLL. The post-scale counters range from 1 to 256 with any non-50% duty cycle setting. In fast PLLs, there are four post-scale counters (`C0`, `C1`, `C2`, `C3`) for the regional and global clock output ports. All post-scale counters range from 1 to 32 with a 50% duty cycle setting. For non-50% duty cycle clock outputs, the post-scale counters range from 1 to 16. If the design uses a high-speed I/O interface, you can connect the dedicated `dffioclk` clock output port to allow the high-speed VCO frequency to drive the serializer/deserializer (SERDES).

The Quartus II software automatically chooses the appropriate scaling factors according to the input frequency, multiplication, and division values entered into the `altpll` megafunction.

Phase-Shift Implementation

Phase shift is used to implement a robust solution for clock delays in Stratix II devices. Phase shift is implemented by using a combination of the VCO phase output and the counter starting time. The VCO phase output and counter starting time is the most accurate method of inserting delays, since it is purely based on counter settings, which are independent of process, voltage, and temperature.

Stratix II PLLs do not support programmable delay elements because these delay elements require considerable area on the die and are sensitive to process, voltage, and temperature.

You can phase shift the output clocks from the Stratix II enhanced PLL in either:

- Fine resolution using VCO phase taps
- Coarse resolution using counter starting time

The VCO phase tap and counter starting time is implemented by allowing any of the output counters (`C[5..0]` or *m*) to use any of the eight phases of the VCO as the reference clock. This allows you to adjust the delay time with a fine resolution. The minimum delay time that you can insert using this method is defined by:

$$\Phi_{fine} = \frac{1}{8}T_{VCO} = \frac{1}{8f_{VCO}} = \frac{N}{8Mf_{REF}}$$

where f_{REF} is input reference clock frequency.

For example, if f_{REF} is 100 MHz, *n* is 1, and *m* is 8, then f_{VCO} is 800 MHz and Φ_{fINE} equals 156.25 ps. This phase shift is defined by the PLL operating frequency, which is governed by the reference clock frequency and the counter settings.

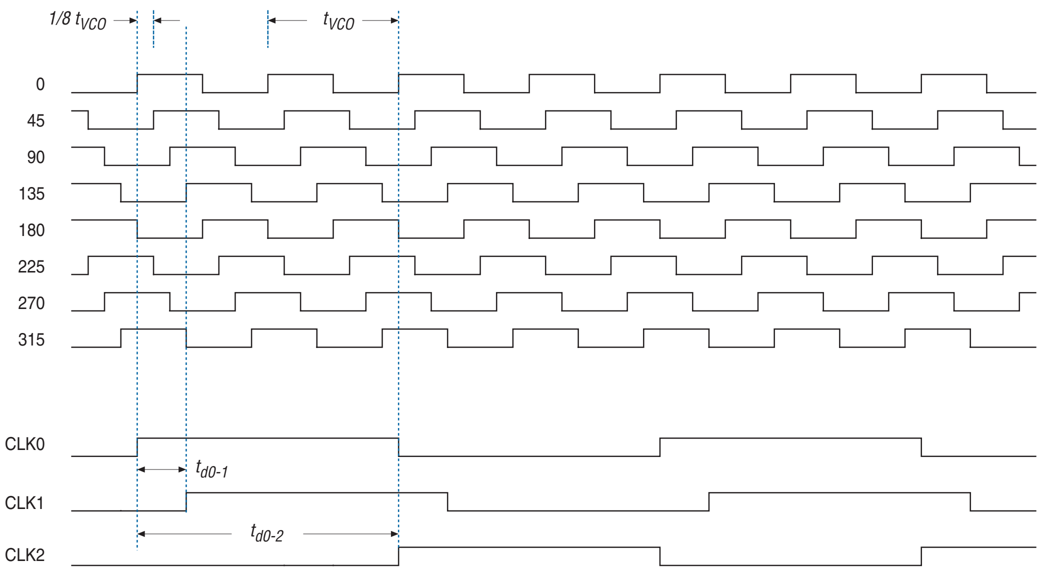
You can also delay the start of the counters for a predetermined number of counter clocks. You can express phase shift as:

$$\Phi_{coarse} = \frac{C-1}{f_{VCO}} = \frac{(C-1)N}{Mf_{REF}}$$

where *C* is the count value set for the counter delay time, (this is the initial setting in the PLL usage section of the compilation report in the Quartus II software). If the initial value is 1, C – 1 = 0° phase shift.

Figure 1–13 shows an example of phase shift insertion using the fine resolution using VCO phase taps method. The eight phases from the VCO are shown and labeled for reference. For this example, `CLK0` is based off the 0phase from the VCO and has the *C* value for the counter set to one. The `CLK1` signal is divided by four, two VCO clocks for high time and two VCO clocks for low time. `CLK1` is based off the 135 phase tap from the VCO and also has the *C* value for the counter set to one. The `CLK1` signal is also divided by 4. In this case, the two clocks are offset by 3 Φ_{FINE}. `CLK2` is based off the `0phase` from the VCO but has the *C* value for the counter set to three. This creates a delay of 2 Φ_{COARSE}, (two complete VCO periods).

Figure 1–13. Delay Insertion Using VCO Phase Output & Counter Delay Time



You can use the coarse and fine phase shifts as described above to implement clock delays in Stratix II devices. The phase-shift parameters are set in the Quartus II software.

Programmable Duty Cycle

The programmable duty cycle allows enhanced and fast PLLs to generate clock outputs with a variable duty cycle. This feature is supported on each enhanced and fast PLL post-scale counter `C[]`. The duty cycle setting is achieved by a low and high time count setting for the post-scale counters. The Quartus II software uses the frequency input and the required multiply or divide rate to determine the duty cycle choices. The post-scale counter value determines the precision of the duty cycle. 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 `C0` counter is ten, 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 the `fbin` pin to 50%. Combining the programmable duty cycle with programmable phase shift allows the generation of precise non-overlapping clocks.

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.

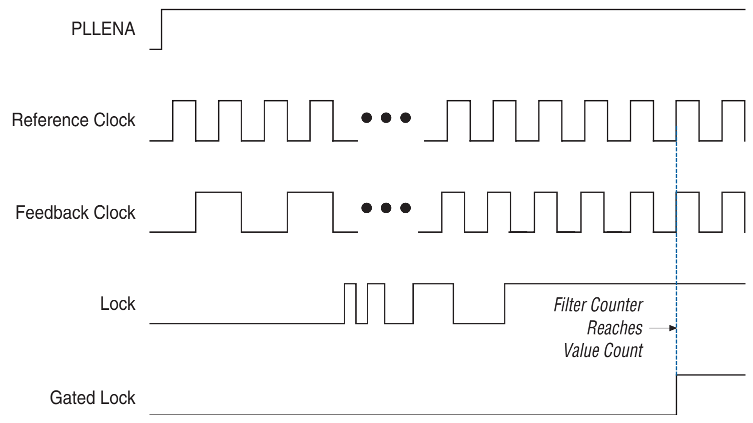
Enhanced Lock Detect Circuit

The lock output indicates that the PLL has locked onto 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. Either a gated lock signal or an ungated lock signal from the locked port can drive the logic array or an output pin. The Stratix II enhanced and fast PLLs include a programmable counter that holds the lock signal low for a user-selected number of input clock transitions. This allows the PLL to lock before enabling the lock signal. You can use the Quartus II software to set the 20-bit counter value.

The device resets and enables both the counter and the PLL simultaneously when the `pllena` signal is asserted. Enhanced PLLs and fast PLLs support this feature. To ensure correct circuit operation, and to ensure that the output clocks have the correct phase relationship with respect to the input clock, Altera recommends that the input clock be running before the Stratix II device is powered up.

Figure 1–14 shows the timing waveform for the lock and gated lock signals.

Figure 1–14. Timing Waveform for Lock & Gated Lock Signals



pllenable

The `pllenable` pin is a dedicated pin that enables or disables all PLLs on the Stratix II device. When the `pllenable` pin is low, the clock output ports are driven low 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.

pfdena

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 `clkloss` or gated `locked` status signals, to trigger `pfdena`.

areset

The `areset` signal is the reset or resynchronization input for each PLL. The device input pins or internal logic can drive these input signals. When driven high, the PLL counters reset, clearing the PLL output and placing the PLL out of lock. The VCO is 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 starts at a higher value than desired as the PLL locks.

clkena

If the system cannot tolerate the higher output frequencies when using `pfdena` higher value, the `clkena` signals can disable the output clocks until the PLL locks. The `clkena` signals control the regional, global, and external clock outputs. The `clkena` signals are registered on the falling edge of the counter output clock to enable or disable the clock without glitches. See Figure 1–53 in the "Clock Control Block" section on page 1–76 of this document for more information on the `clkena` signals.

Advanced Features

Stratix II PLLs offer a variety of advanced features, such as counter cascading, clock switchover, PLL reconfiguration, reconfigurable bandwidth, and spread-spectrum clocking. Table 1–13 shows which advanced features are available in which type of Stratix II PLL.

Table 1–13. Stratix II PLL Advanced Features

Advanced Feature	Availability	
	Enhanced PLLs	Fast PLLs *(1)*
Counter cascading	✓	
Clock switchover	✓	✓
PLL reconfiguration	✓	✓
Reconfigurable bandwidth	✓	✓
Spread-spectrum clocking	✓	

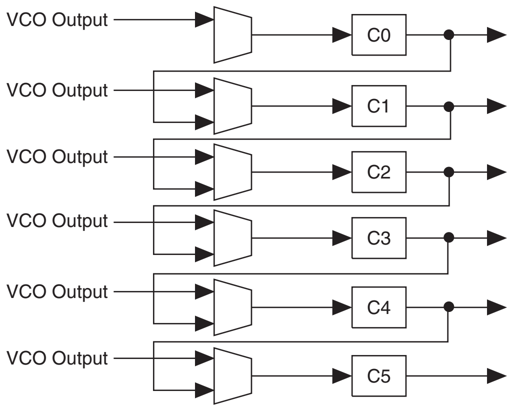
Note to Table 1–13:

(1) Stratix II fast PLLs only support manual clock switchover, not automatic clock switchover.

Counter Cascading

The Stratix II enhanced PLL supports counter cascading to create post-scale counters larger than 512. This is implemented by feeding the output of one counter into the input of the next counter in a cascade chain, as shown in Figure 1–15.

Figure 1–15. Counter Cascading



When cascading counters to implement a larger division of the high-frequency VCO clock, the cascaded counters behave as one counter with the product of the individual counter settings. For example, if C0 = 4 and C1 = 2, then the cascaded value is C0 × C1 = 8.

The Stratix II fast PLL does not support counter cascading.

Counter cascading is set in the configuration file, meaning they can not be cascaded using PLL reconfiguration.

Clock Switchover

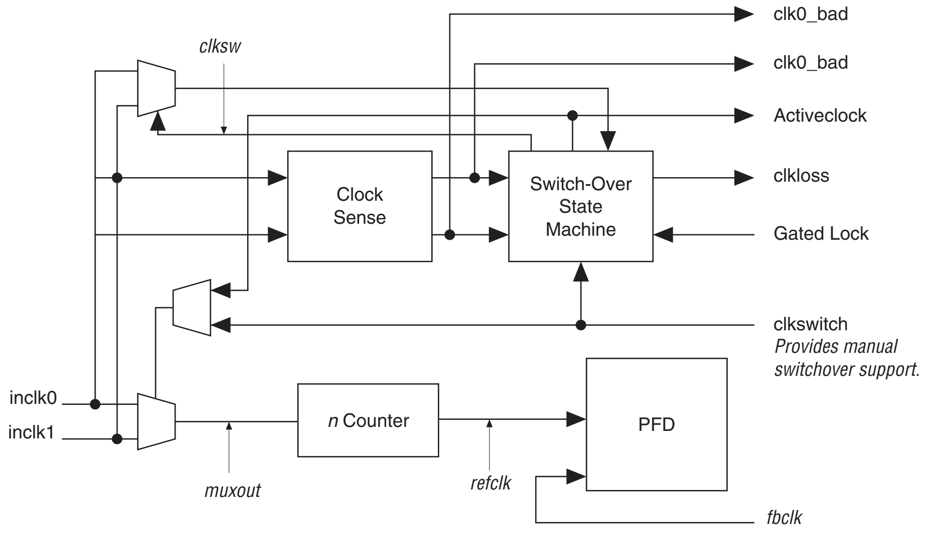
The clock switchover feature allows the PLL to switch between two reference input clocks. Use this feature for clock redundancy or for a dual clock domain application such as in a system that turns on the redundant clock if the primary clock stops running. The design can perform clock switchover automatically, when the clock is no longer toggling, or based on a user control signal, `clkswitch`.

Enhanced PLLs support both automatic and manual switchover, while fast PLLs only support manual switchover.

Automatic Clock Switchover

Stratix II device PLLs support a fully configurable clock switchover capability. Figure 1–16 shows the block diagram of the switch-over circuit built into the enhanced PLL. When the primary clock signal is not present the clock sense block automatically switches from the primary to the secondary clock for PLL reference. The design sends out the `clk0_bad`, `clk1_bad`, and the `clk_loss` signals from the PLL to implement a custom switchover circuit.

Figure 1–16. Automatic Clock Switchover Circuit Block Diagram



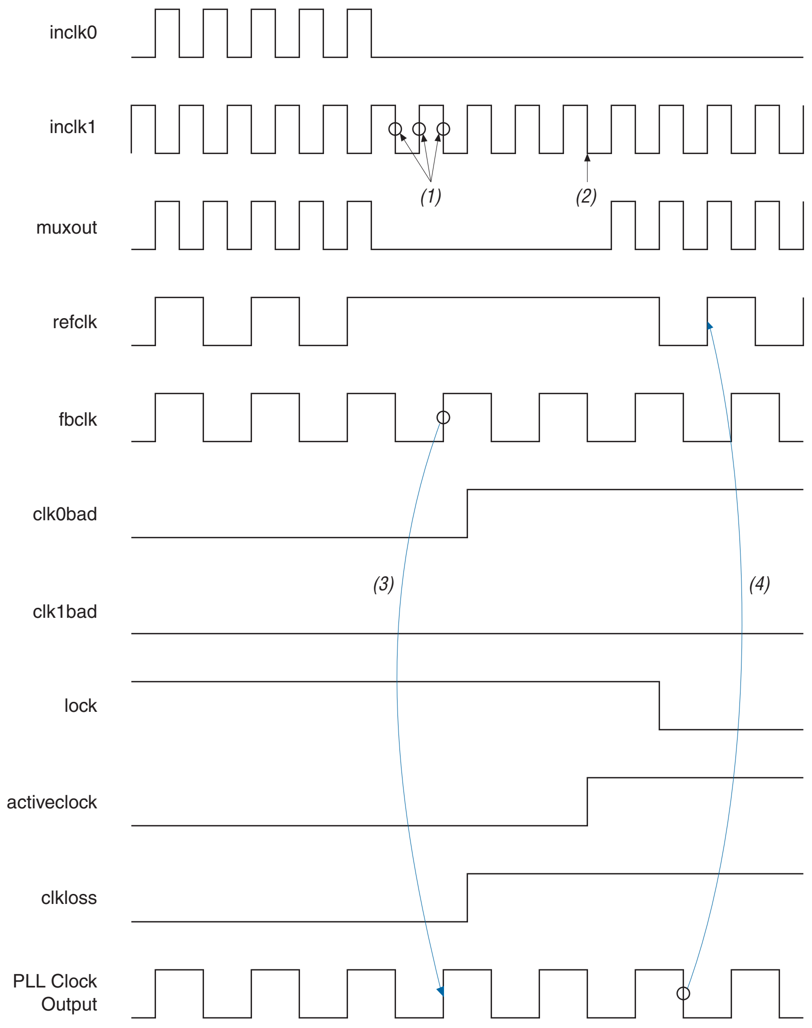
There are at least three possible ways to use the clock switchover feature.

- Use the switchover circuitry for switching from a primary to secondary input 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 shown on the bottom of Figure 1–16. In this case, the secondary clock becomes the reference clock for the PLL. This automatic switchover feature only works for switching from the primary to secondary clock.

- 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 should choose the secondary clock frequency so the VCO operates within the recommended range of 500 to 1000 MHz. You should also set the *m* and *n* counters accordingly to keep the VCO operating frequency in the recommended range.
- Use the gated lock to control switchovers if for some reason the PLL loses lock. The gated lock signal goes low to force the switchover state machine to switch to the secondary clock. If an external PLL is driving the Stratix II PLL, excessive jitter on the clock input could cause the PLL to lose lock. Since the switchover circuit still senses clock edges, it might not sense a switch condition. In this case, you can control switchover based on the loss of the primary clock by using the gated locked signal.

Figure 1–17 shows an example waveform of the switchover feature when using the automatic `clkloss` detection. Here, the `inclk0` signal gets stuck low. After the `inclk1` signal gets stuck at low for approximately two clock cycles, the clock sense circuitry drives the `clk0_bad` signal high. Also, since the reference clock signal is not toggling, the `clk_loss` signal goes low indicating a switch condition. Then, the switchover state machine controls the multiplexer through the `clksw` signal to switch to the secondary clock.

Figure 1–17. Automatic Switchover Upon Clock Loss Detection

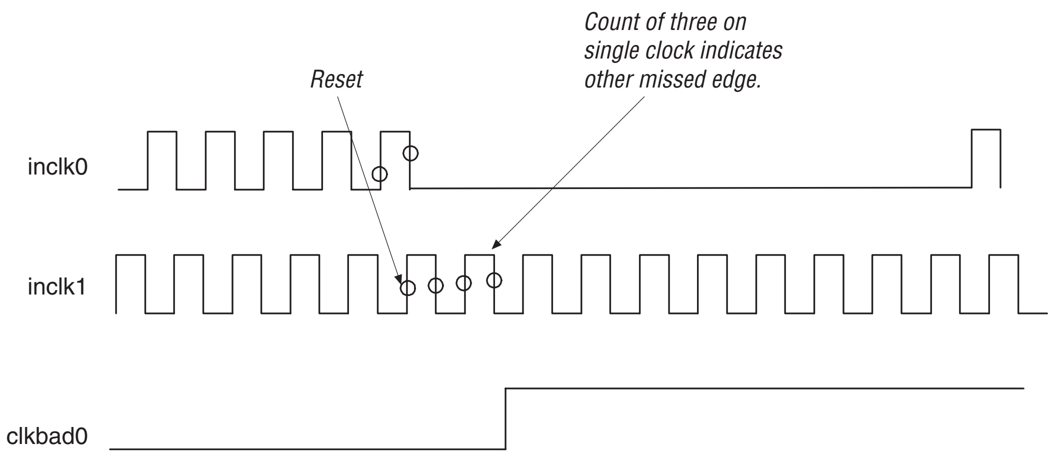


Notes to Figure 1–17:

(1) The number of clock edges before allowing switchover is determined by the counter setting.
(2) Switchover is enabled on the falling edge of `INCLK1`.
(3) The rising edge of `FBCLK` causes the VCO frequency to decrease.
(4) The rising edge of `REFCLK` starts the PLL lock process again, and the VCO frequency increases.

The switch-over state machine has two counters that count the edges of the primary and the secondary clocks; `counter0` counts the number of `inclk0` edges and `counter1` counts the number of `inclk1` edges. The counters get reset to zero when the count values reach 1, 1; 1, 2; 2, 1; or 2, 2 for `inclock0` and `inclock1`, respectively. For example, if `counter0` counts two edges, its count is set to two and if `counter1` counts two edges before the `counter0` sees another edge, they are both reset to 0. If for some reason one of the counters counts to three, it means the other clock missed an edge. The `clkbad0` or `clkbad1` signal goes high, and the switchover circuitry signals a switch condition. See Figure 1–18.

Figure 1–18. Clock-Edge Detection for Switchover



Manual Override

When using automatic switchover, you can switch input clocks by using the manual override feature with the `clkswitch` input.

☞ The manual override feature available in automatic clock switchover is different from manual clock switchover.

Figure 1–19 shows an example of a waveform illustrating the switchover feature when controlled by `clkswitch`. In this case, both clock sources are functional and `inclk0` is selected as the primary clock. `clkswitch` goes high, which starts the switchover sequence. On the falling edge of `inclk0`, the counter's reference clock, `muxout`, is gated off to prevent any clock glitching. On the falling edge of `inclk1`, the reference clock multiplexer switches from `inclk0` to `inclk1` as the PLL reference. This is also when the `clksw` signal changes to indicate which clock is selected as primary and which is secondary.

The `clkloss` signal mirrors the `clkswitch` signal and `activeclock` mirrors `clksw` in this mode. Since both clocks are still functional during the manual switch, neither `clk_bad` signal goes high. Since the switchover circuit is edge-sensitive, the falling edge of the `clkswitch` signal does not cause the circuit to switch back from `inclk1` to `inclk0`. When the `clkswitch` signal goes high again, the process repeats. `clkswitch` and automatic switch only works if the clock being switched to is available. If the clock is not available, the state machine waits until the clock is available.

Figure 1–19. Clock Switchover Using the CLKSWITCH Control

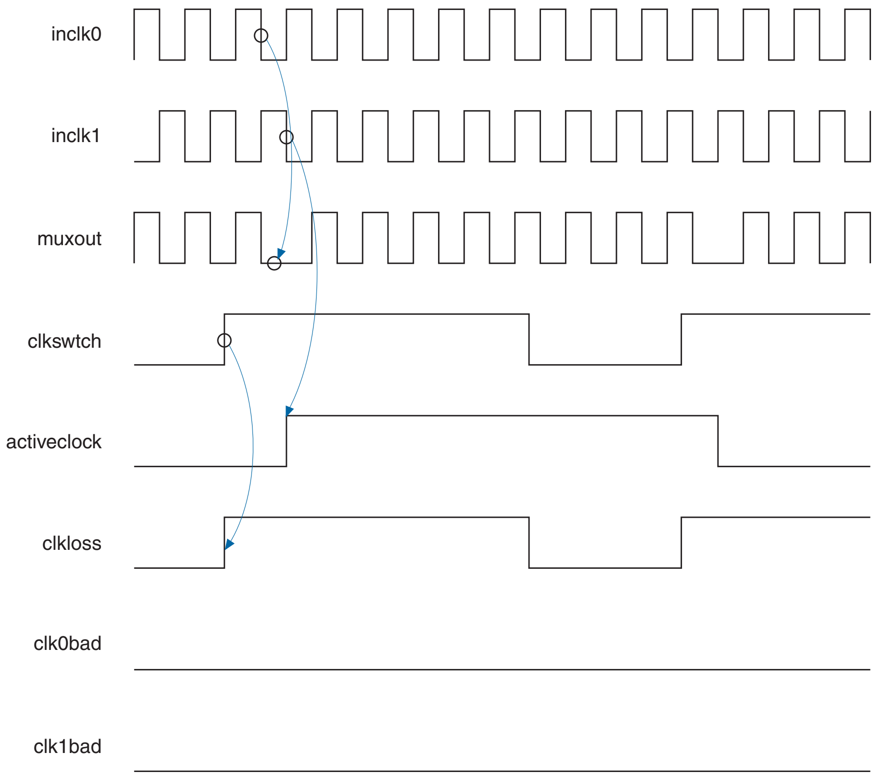
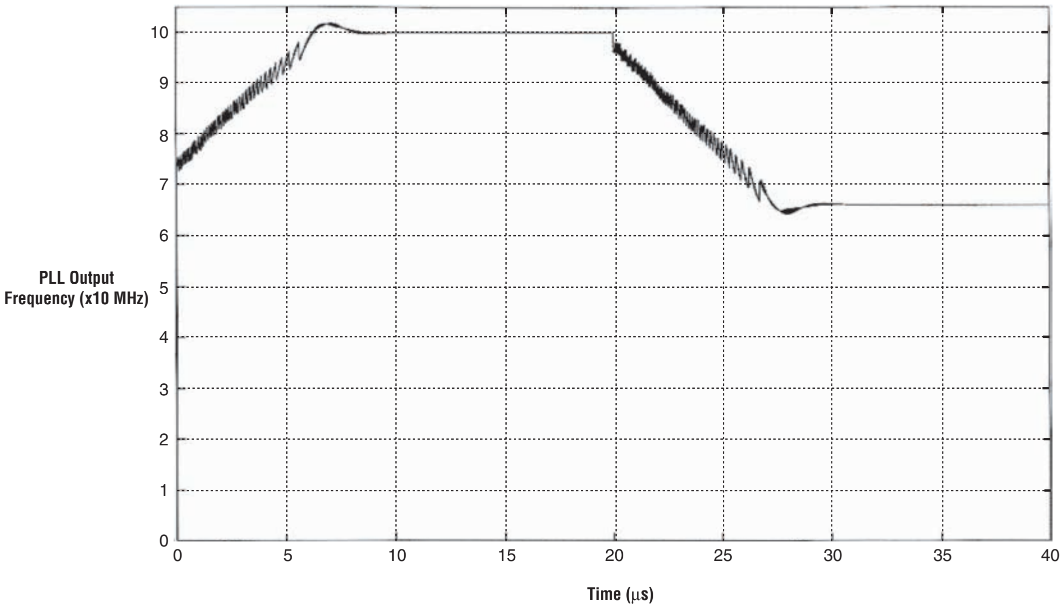


Figure 1–20 shows a simulation of using switchover for two different reference frequencies. In this example simulation, the reference clock is either 100 or 66 MHz. The PLL begins with f_{IN} = 100 MHz and is allowed to lock. At 20 µs, the clock is switched to the secondary clock, which is at 66 MHz.

Figure 1–20. Switchover Simulation *Note (1)*



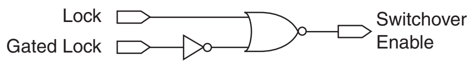
Note to Figure 1–20:

(1) This simulation was performed under the following conditions: the *n* counter is set to 2, the *m* counter is set to 16, and the output counter is set to 8. Therefore, the VCO operates at 800 MHz for the 100-MHz input references and at 528 MHz for the 66-MHz reference input.

Lock Signal-Based Switchover

The lock circuitry can initiate the automatic switchover. This is useful for cases where the input clock is still clocking, but its characteristics have changed so that the PLL is not locked to it. The switchover enable is based on both the gated and ungated lock signals. If the ungated lock is low, the switchover is not enabled until the gated lock has reached its terminal count. You must activate the switchover enable if the gated lock is high, but the ungated lock goes low. The switchover timing for this mode is similar to the waveform shown in Figure 1–19 for `clkswitch` control, except the switchover enable replaces `clkswitch`. Figure 1–16 shows the switchover enable circuit when controlled by lock and gated lock.

Figure 1–21. Switchover Enable Circuit



Manual Clock Switchover

Stratix II enhanced and fast PLLs support manual switchover, where the `clkswitch` signal controls whether `inclk0` or `inclk1` is the input clock to the PLL. If `clkswitch` is low, then `inclk0` is selected; if `clkswitch` is high, then `inclk1` is selected. Figure 1–22 shows the block diagram of the manual switchover circuit in fast PLLs. The block diagram of the manual switchover circuit in enhanced PLLs is shown in Figure 1–22.

Figure 1–22. Manual Clock Switchover Circuitry in Fast PLLs

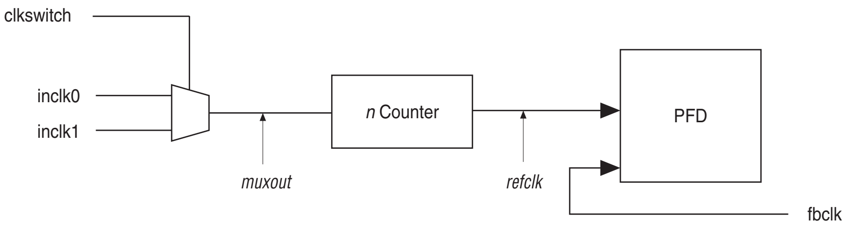
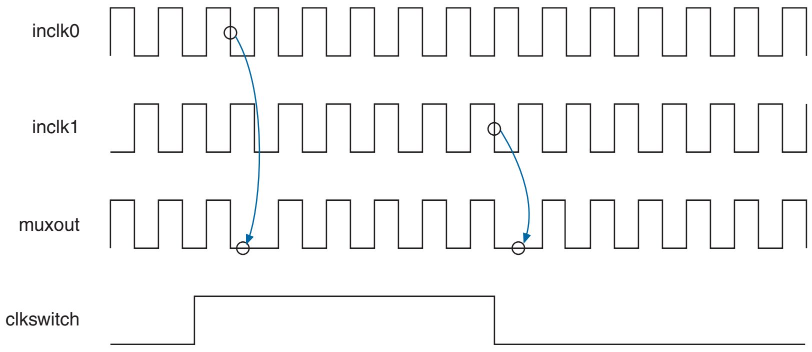


Figure 1–23 shows an example of a waveform illustrating the switchover feature when controlled by `clkswitch`. In this case, both clock sources are functional and `inclk0` is selected as the primary clock. `clkswitch` goes high, which starts the switch-over sequence. On the falling edge of `inclk0`, the counter's reference clock, `muxout`, is gated off to prevent any clock glitching. On the rising edge of `inclk1`, the reference clock multiplex switches from `inclk0` to `inclk1` as the PLL reference. When the `clkswitch` signal goes low, the process repeats, causing the circuit to switch back from `inclk1` to `inclk0`.

Figure 1–23. Manual Switchover



Software Support

Table 1–14 summarizes the signals used for clock switchover.

Table 1–14. altpll MegaFunction Clock Switchover Signals

Port	Description	Source	Destination
inclk0	Reference `clk0` to the PLL.	I/O pin	Clock switchover circuit
inclk1	Reference `clk1` to the PLL.	I/O pin	Clock switchover circuit
clkbad0 *(1)*	Signal indicating that `inclk0` is no longer toggling.	Clock switchover circuit	Logic array
clkbad1 *(1)*	Signal indicating that `inclk1` is no longer toggling.	Clock switchover circuit	Logic array
clkswitch	Switchover signal used to initiate clock switchover asynchronously. When used in manual switchover, clkswitch is used as a select signal between inclk0 and inclk1 `clswitch = 0 inclk0` is selected and vice versa.	Logic array or I/O pin	Clock switchover circuit
clkloss *(1)*	Signal indicating that the switchover circuit detected a switch condition.	Clock switchover circuit	Logic array
locked	Signal indicating that the PLL has lost lock.	PLL	Clock switchover circuit
activeclock *(1)*	Signal to indicate which clock (0 = `inclk0`, 1= `inclk1`) is driving the PLL.	PLL	Logic array

Note for Table 1–14:
(1) These ports are only available for enhanced PLLs and in auto mode and when using automatic switchover.

All the switchover ports shown in Table 1–14 are supported in the `altpll` megafunction in the Quartus II software. The `altpll` megafunction supports two methods for clock switchover:

- When selecting an enhanced PLL, you can enable both the automatic and the manual switchover, making all the clock switchover ports available.
- When selecting a fast PLL, you can use only enable the manual clock switchover option to select between `inclk0` or `inclk1`. The `clkloss`, `activeclock` and the `clkbad0`, and `clkbad1` signals are not available when manual switchover is selected.

If the primary and secondary clock frequencies are different, the Quartus II software selects the proper parameters to keep the VCO within the recommended frequency range.

For more information on PLL software support in the Quartus II software, see the *altpll Megafunction User Guide*.

Guidelines

Use the following guidelines to design with clock switchover in PLLs.

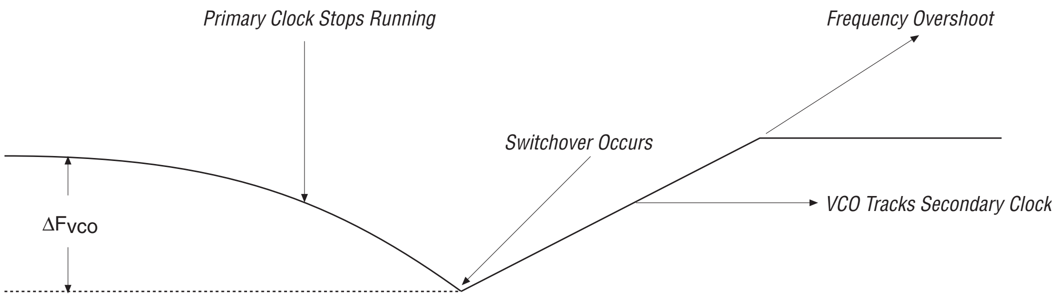
- When using automatic switchover, the `clkswitch` signal has a minimum pulse width based on the two reference clock periods. The `CLKSWITCH` pulse width must be greater than or equal to the period of the current reference clock (t_{from_clk}) multiplied by two plus the rounded-up version of the ratio of the two reference clock periods. For example, if t_{to_clk} is equal to t_{from_clk}, then the `CLKSWITCH` pulse width should be at least three times the period of the clock pulse.

  $$t_{CLKSWITCHCHmin} \geq t_{from\_clk} \times [2 + int_{round\_up} (t_{to\_clk} \div t_{from\_clk})]$$

- Applications that require a clock switchover feature and a small frequency drift should use a low-bandwidth PLL. The low-bandwidth PLL reacts slower than a high-bandwidth PLL to reference input clock changes. When the switchover happens, a low-bandwidth PLL propagates the stopping of the clock to the output slower than a high-bandwidth PLL. A low-bandwidth PLL filters out jitter on the reference clock. However, be aware that the low-bandwidth PLL also increases lock time.
- Stratix II device PLLs can use both the automatic clock switchover and the `clkswitch` input simultaneously. Therefore, the switchover circuitry can automatically switch from the primary to the secondary clock. Once the primary clock stabilizes again, the `clkswitch` signal can switch back to the primary clock. During switchover, the `PLL_VCO` continues to run and slows down, generating frequency drift on the PLL outputs. The `clkswitch` signal controls switchover with its rising edge only.
- If the clock switchover event is glitch-free, after the switch occurs, there is still a finite resynchronization period to lock onto a new clock as the VCO ramps up. The exact amount of time it takes for the PLL to relock is dependent on the PLL configuration. Use the PLL programmable bandwidth feature to adjust the relock time.
- If the phase relationship between the input clock to the PLL and output clock from the PLL is important in your design, assert `areset` for 10ns after performing a clock switchover. Wait for the locked signal (or gated lock) to go high before re-enabling the output clocks from the PLL.

- Figure 1–24 shows how the VCO frequency gradually decreases when the primary clock is lost and then increases as the VCO locks on to the secondary clock. After the VCO locks on to the secondary clock, some overshoot can occur (an over-frequency condition) in the VCO frequency.

Figure 1–24. VCO Switchover Operating Frequency



- Disable the system during switchover if it is not tolerant to frequency variations during the PLL resynchronization period. There are two ways to disable the system. First, the system may require some time to stop before switchover occurs. The switchover circuitry includes an optional five-bit counter to delay when the reference clock is switched. You have the option to control the time-out setting on this counter (up to 32 cycles of latency) before the clock source switches. You can use these cycles for disaster recovery. The clock output frequency varies slightly during those 32 cycles since the VCO can still drift without an input clock. Programmable bandwidth can control the PLL response to limit drift during this 32 cycle period.
- A second option available is the ability to use the PFD enable signal (`pfdena`) along with user-defined control logic. In this case you can use `clk0_bad` and `clk1_bad` status signals to turn off the PFD so the VCO maintains its last frequency. You can also use the state machine to switch over to the secondary clock. Upon re-enabling the PFD, output clock enable signals (`clkena`) can disable clock outputs during the switchover and resynchronization period. Once the lock indication is stable, the system can re-enable the output clock(s).

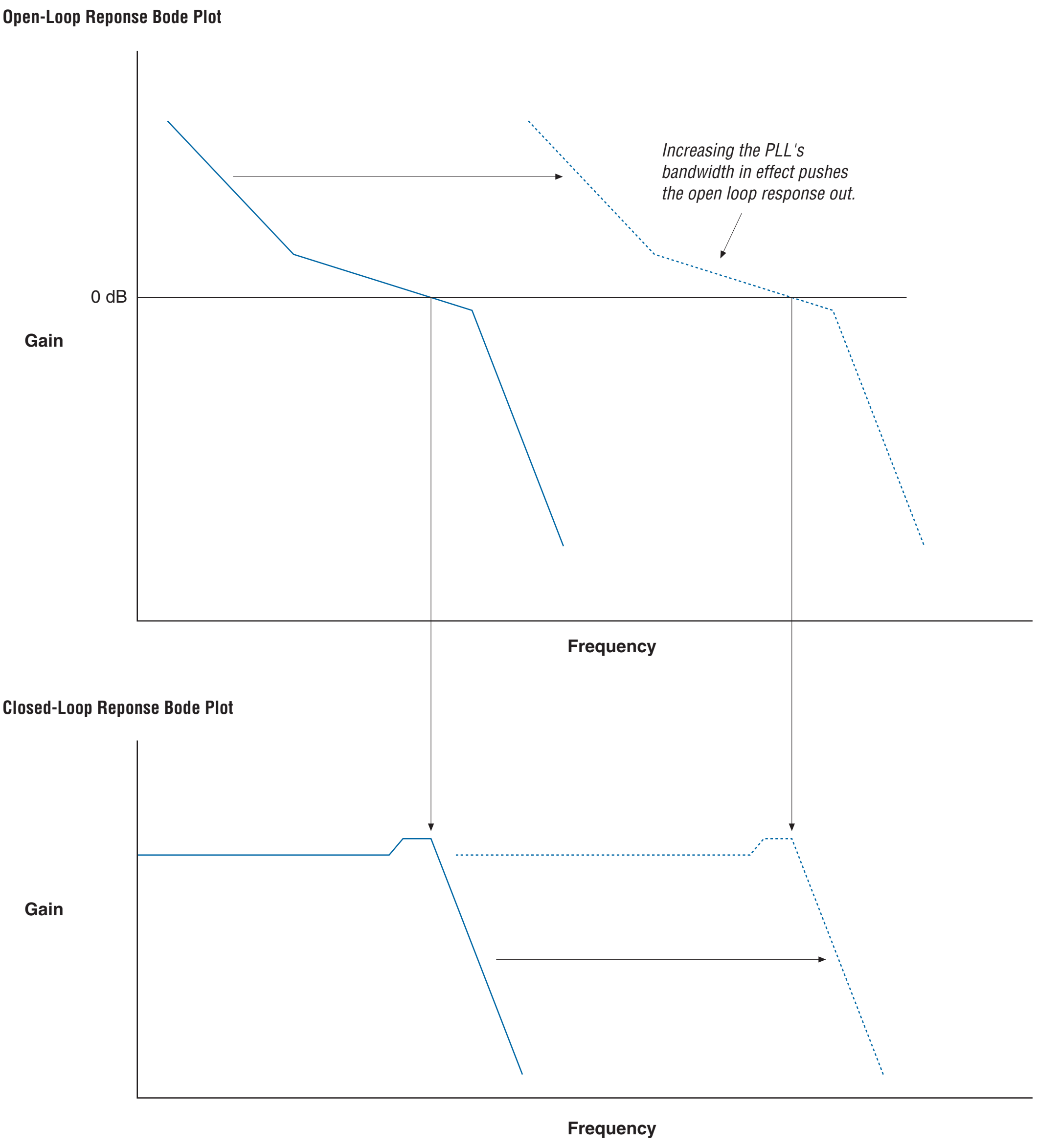
Reconfigurable Bandwidth

Stratix II enhanced and fast PLLs provide advanced control of the PLL bandwidth using the PLL loop's programmable characteristics, including loop filter and charge pump.

Background

PLL bandwidth is the measure of the PLL's ability to track the input clock and jitter. The closed-loop gain 3-dB frequency in the PLL determines the PLL bandwidth. The bandwidth is approximately the unity gain point for open loop PLL response. As Figure 1–25 shows, these points correspond to approximately the same frequency.

Figure 1–25. Open- & Closed-Loop Response Bode Plots



A high-bandwidth PLL provides a fast lock time and tracks jitter on the reference clock source, passing it through to the PLL output. A low-bandwidth PLL filters out reference clock, but increases lock time. Stratix II enhanced and fast PLLs allows you to control the bandwidth over a finite range to customize the PLL characteristics for a particular application. The programmable bandwidth feature in Stratix II PLLs benefits applications requiring clock switchover (e.g., TDMA frequency hopping wireless, and redundant clocking).

The bandwidth and stability of such a system is determined by the charge pump current, the loop filter resistor value, the high-frequency capacitor value (in the loop filter), and the *m*-counter value. You can use the Quartus II software to control these factors and to set the bandwidth to the desired value within a given range.

You can set the bandwidth to the appropriate value to balance the need for jitter filtering and lock time. Figures 1–26 and 1–27 show the output of a low- and high-bandwidth PLL, respectively, as it locks onto the input clock.

Figure 1–26. Low-Bandwidth PLL Lock Time

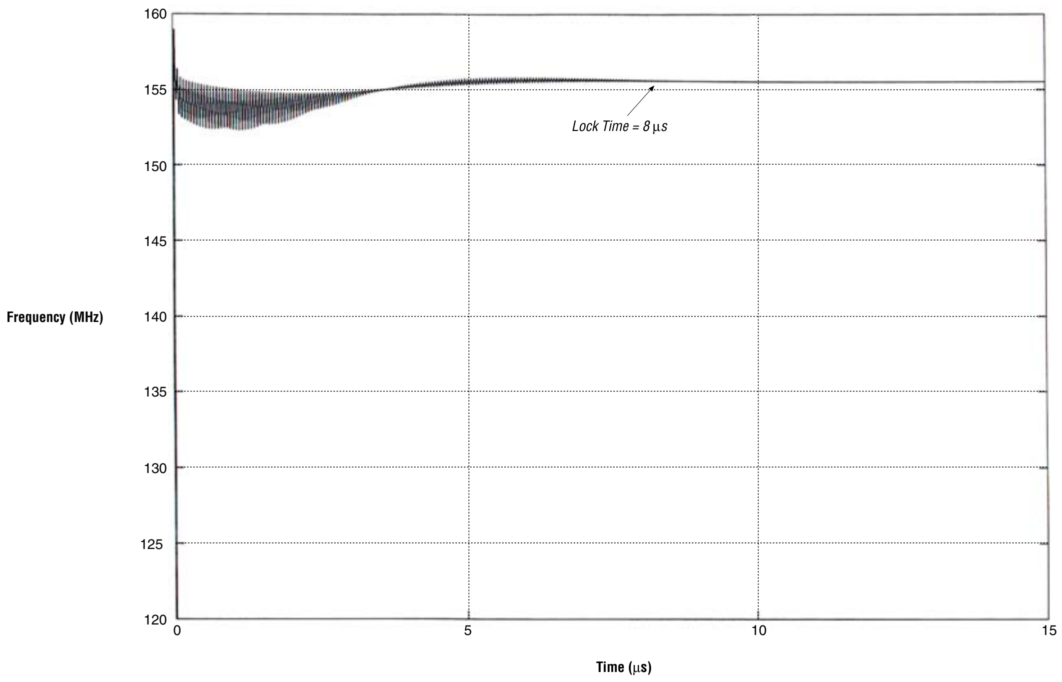
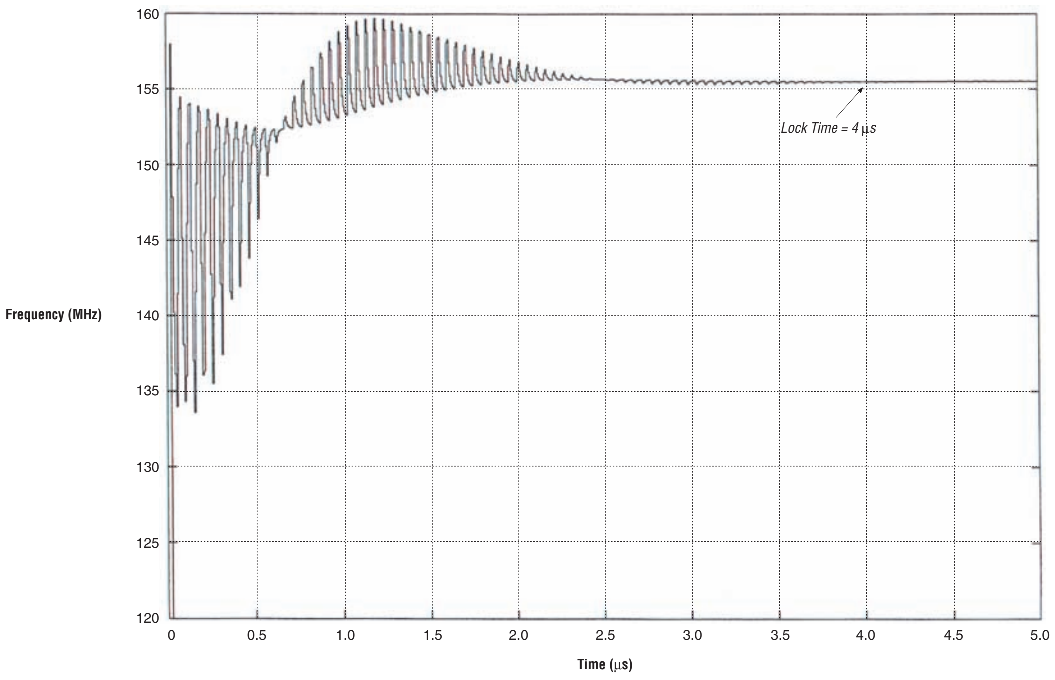


Figure 1–27. High-Bandwidth PLL Lock Time



A high-bandwidth PLL can benefit a system that has two cascaded PLLs. If the first PLL uses spread spectrum (as user-induced jitter), the second PLL can track the jitter that is feeding it by using a high-bandwidth setting. A low-bandwidth PLL can, in this case, lose lock due to the spread-spectrum-induced jitter on the input clock.

A low-bandwidth PLL benefits a system using clock switchover. When the clock switchover happens, the PLL input temporarily stops. A low-bandwidth PLL would react more slowly to changes to its input clock and take longer to drift to a lower frequency (caused by the input stopping) than a high-bandwidth PLL. Figures 1–28 and 1–29 demonstrate this property. The two plots show the effects of clock switchover with a low- or high-bandwidth PLL. When the clock switchover happens, the output of the low-bandwidth PLL (see Figure 1–28) drifts to a lower frequency more slowly than the high-bandwidth PLL output (see Figure 1–29).

Figure 1–28. Effect of Low Bandwidth on Clock Switchover

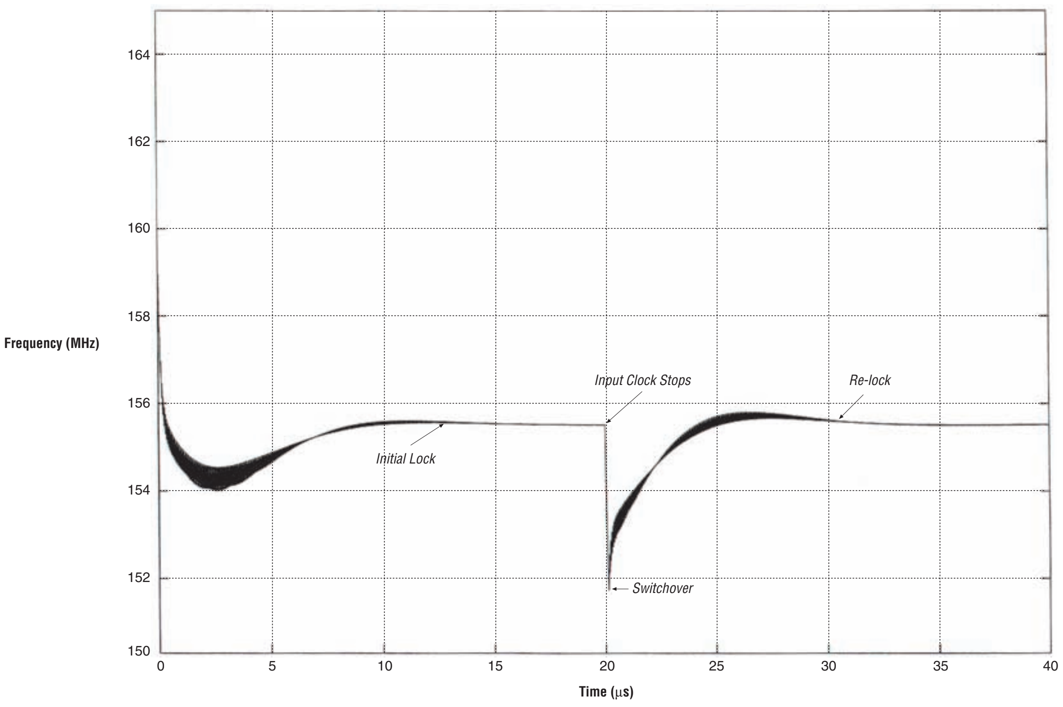
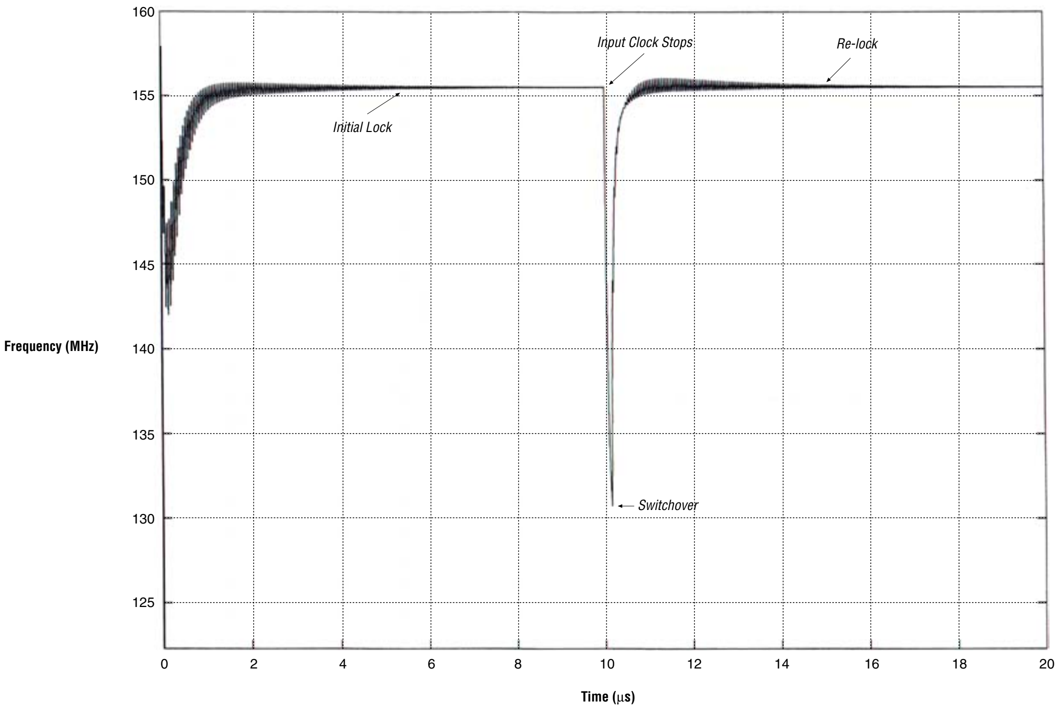


Figure 1–29. Effect of High Bandwidth on Clock Switchover



Implementation

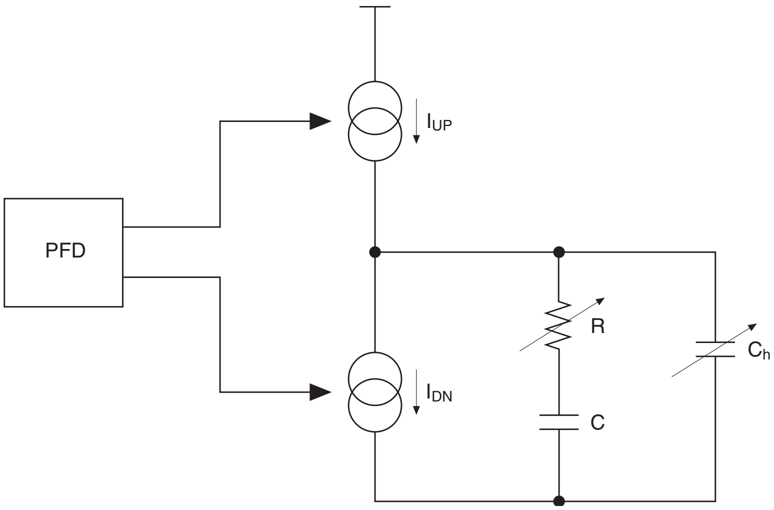
Traditionally, external components such as the VCO or loop filter control a PLL's bandwidth. Most loop filters are made up of passive components such as resistors and capacitors that take up unnecessary board space and increase cost. With Stratix II PLLs, all the components are contained within the device to increase performance and decrease cost.

Stratix II device PLLs implement reconfigurable bandwidth by giving you control of the charge pump current and loop filter resistor (R) and high-frequency capacitor C_H values (see Table 1–15). The Stratix II device enhanced PLL bandwidth ranges from 100 kHz to 10 MHz. The Stratix II device fast PLL bandwidth ranges from 2 to 30 MHz.

☞ The bandwidth ranges are preliminary and subject to change.

The charge pump current directly affects the PLL bandwidth. The higher the charge pump current, the higher the PLL bandwidth. You can choose from a fixed set of values for the charge pump current. Figure 1–30 shows the loop filter and the components that can be set through the Quartus II software. The components are the loop filter resistor, R, and the high frequency capacitor, C_H, and the charge pump current, I_{UP} or I_{DN}.

Figure 1–30. Loop Filter Programmable Components



Software Support

The Quartus II software provides two levels of bandwidth control.

Megafunction-Based Bandwidth Setting

The first level of programmable bandwidth allows you to enter a value for the desired bandwidth directly into the Quartus II software using the `altpll` megafunction. You can also set the bandwidth parameter in the `altpll` megafunction to the desired bandwidth. The Quartus II software selects the best bandwidth parameters available to match your bandwidth request. If the individual bandwidth setting request is not available, the Quartus II software selects the closest achievable value.

Advanced Bandwidth Setting

An advanced level of control is also possible using advanced loop filter parameters. You can dynamically change the charge pump current, loop filter resistor value, and the loop filter (high frequency) capacitor value.

The parameters for these changes are: `charge_pump_current`, `loop_filter_r`, and `loop_filter_c`. Each parameter supports the specific range of values listed in Table 1–15.

Table 1–15. Advanced Loop Filter Parameters

Parameter	Values
Resistor values (kΩ)	*(1)*
High-frequency capacitance values (pF)	*(1)*
Charge pump current settings (µA)	*(1)*

Note to Table 1–15:
(1) Contact Altera Applications for more information.

For more information on Quartus II software support of reconfigurable bandwidth, see the *PLL Reconfiguration* section of the *Quartus II Handbook*.

PLL Reconfiguration

PLLs use several divide counters and different VCO phase taps to perform frequency synthesis and phase shifts. In Stratix II enhanced and fast PLLs, the counter value and phase are configurable in real time. In addition, you can change the loop filter and charge pump components, which affect the PLL bandwidth, on the fly. You can control these PLL components to update the output clock frequency, PLL bandwidth, and phase-shift variation in real time, without the need to reconfigure the entire FPGA.

For more information on PLL reconfiguration, see *AN 367: Implementing PLL Reconfiguration in Stratix II Devices*.

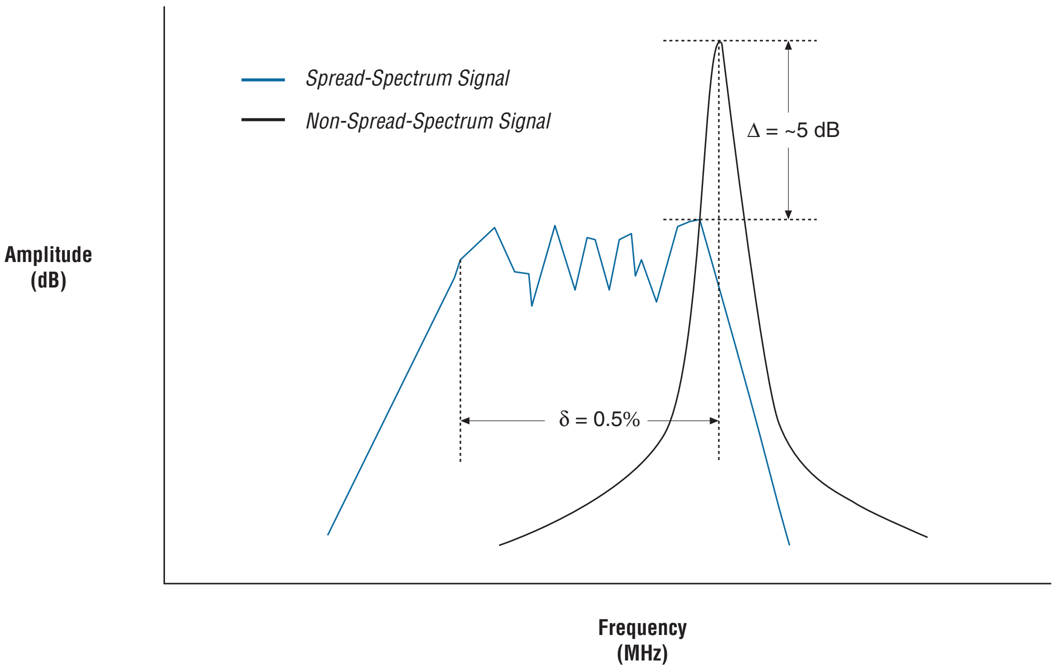
Spread-Spectrum Clocking

Digital clocks are square waves with short rise times and a 50% duty cycle. These high-speed clocks concentrate a significant amount of energy in a narrow bandwidth at the target frequency and at the higher frequency harmonics. This results in high energy peaks and increased electromagnetic interference (EMI). The radiated noise from the energy peaks travels in free air and, if not minimized, can lead to corrupted data and intermittent system errors, which can jeopardize system reliability.

Traditional methods for limiting EMI include shielding, filtering, and multi-layer printed circuit boards (PCBs). However, these methods significantly increase the overall system cost and sometimes are not enough to meet EMI compliance. Spread-spectrum technology provides you with a simple and effective technique for reducing EMI without additional cost and the trouble of re-designing a board.

Spread-spectrum technology modulates the target frequency over a small range. For example, if a 100-MHz signal has a 0.5% down-spread modulation, then the frequency is swept from 99.5 to 100 MHz. Figure 1–31 gives a graphical representation of the energy present in a spread-spectrum signal vs. a non-spread spectrum-signal. It is apparent that instead of concentrating the energy at the target frequency, the energy is re-distributed across a wider band of frequencies, which reduces peak energy. Not only is there a reduction in the fundamental peak EMI components, but there is also a reduction in EMI of the higher order harmonics. Since some regulations focus on peak EMI emissions, rather than average EMI emissions, spread-spectrum technology is a valuable method of EMI reduction.

Figure 1–31. Spread-Spectrum Signal Energy Versus Non-Spread-Spectrum Signal Energy



Spread-spectrum technology would benefit a design with high EMI emissions and/or strict EMI requirements. Device-generated EMI is dependent on frequency and output voltage swing amplitude and edge rate. For example, a design using LVDS already has low EMI emissions because of the low-voltage swing. The differential LVDS signal also allows for EMI rejection within the signal. Therefore, this situation may not require spread-spectrum technology.

☞ Spread-spectrum clocking is only supported in Stratix II enhanced PLLs, not fast PLLs.

Implementation

Stratix II device enhanced PLLs feature spread-spectrum technology to reduce the EMIs emitted from the device. The enhanced PLL provides approximately 0.5% down spread using a triangular, also known as linear, modulation profile. The modulation frequency is programmable and ranges from approximately 100 to 500 kHz. The spread percentage is based on the clock input to the PLL and the *m* and *n* settings. Spread-spectrum technology reduces the peak energy by four to six dB at the target frequency. However, this number is dependent on bandwidth and the *m* and *n* counter values and can vary from design to design.

Spread percentage, also known as modulation width, is defined as the percentage that the design modulates the target frequency. A negative (–) percentage indicates a down spread, a positive (+) percentage indicates an up spread, and a (±) indicates a center spread. Modulation frequency is the frequency of the spreading signal, or how fast the signal sweeps from the minimum to the maximum frequency. Down-spread modulation shifts the target frequency down by half the spread percentage, centering the modulated waveforms on a new target frequency.

The *m* and *n* counter values are toggled at the same time between two fixed values. The loop filter then slowly changes the VCO frequency to provide the spreading effect, which results in a triangular modulation. An additional spread-spectrum counter (shown in Figure 1–32) sets the modulation frequency. Figure 1–32 shows how spread-spectrum technology is implemented in the Stratix II device enhanced PLL.

Figure 1–32. Stratix II Spread-Spectrum Circuit Block Diagram

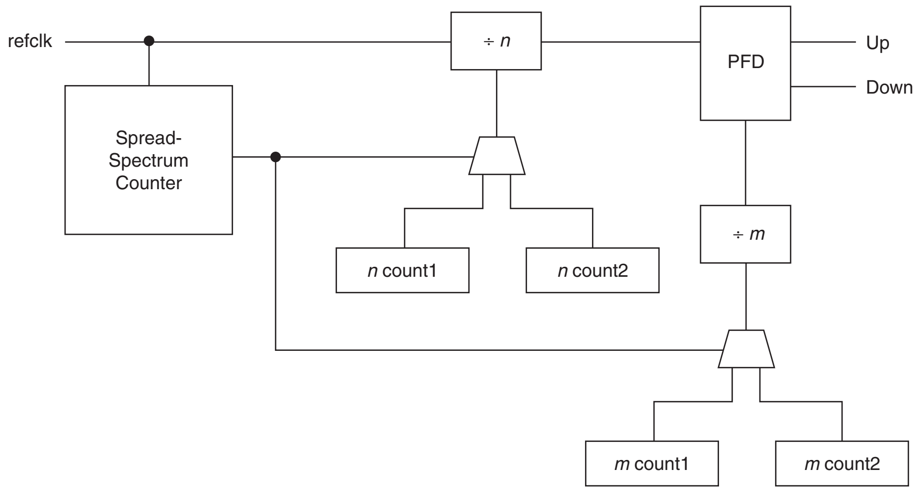


Figure 1–33 shows a VCO frequency waveform when toggling between different counter values. Since the enhanced PLL switches between two different *m* and *n* values, the result is a straight line between two frequencies, which gives a linear modulation. The magnitude of modulation is determined by the ratio of two *m*/*n* sets. The percent spread is determined by:

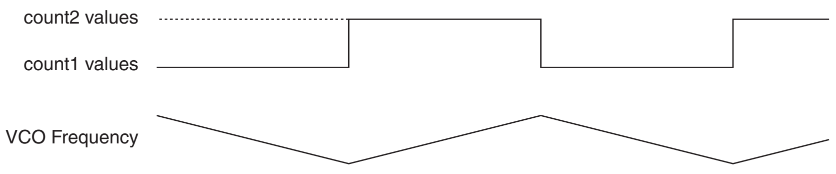
percent spread = $(f_{VCOmax}\ f_{VCOmin})/f_{VCOmax} = 1\ [(m2\ n1)/(m1\ n2)]$.

The maximum and minimum VCO frequency is defined as:

$f_{VCOmax} = (m1/n1)\ f_{REF}$

$f_{VCOmin} = (m2/n2)\ f_{REF}$

Figure 1–33. VCO Frequency Modulation Waveform



Software Support

You can enter the desired down-spread percentage and modulation frequency in the `altpll` megafunction through the Quartus II software. Alternatively, the `downspread` parameter in the `altpll` megafunction can be set to the desired down-spread percentage. Timing analysis ensures the design operates at the maximum spread frequency and meets all timing requirements.

For more information on PLL software support in the Quartus II software, see the *altpll Megafunction User Guide*.

Guidelines

If the design cascades PLLs, the source (upstream) PLL should have a low-bandwidth setting, while the destination (downstream) PLL should have a high-bandwidth setting. The upstream PLL must have a low-bandwidth setting because a PLL does not generate jitter higher than its bandwidth. The downstream PLL must have a high bandwidth setting to track the jitter. The design must use the spread-spectrum feature in a low-bandwidth PLL, and, therefore, the Quartus II software automatically sets the spread-spectrum PLL bandwidth to low.

If the programmable or reconfigurable bandwidth features are used, then you cannot use spread spectrum.

Stratix II devices can accept a spread-spectrum input with typical modulation frequencies. However, the device cannot automatically detect that the input is a spread-spectrum signal. Instead, the input signal looks like deterministic jitter at the input of the downstream PLL.

Spread spectrum can have a minor effect on the output clock by increasing the period jitter. Period jitter is the deviation of a clock's cycle time from its previous cycle position. Period jitter measures the variation of the clock output transition from its ideal position over consecutive edges.

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 and subsystem should use the Stratix II device as the clock source. Communication could fail if the Stratix II logic array is clocked by the spread-spectrum clock, but the data it receives from another device is not clocked by the spread spectrum.

Since spread spectrum affects the *m* counter values, all spread-spectrum PLL outputs are effected. 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 switching when using spread spectrum.

Board Layout

The enhanced and fast PLL circuits in Stratix II 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. Stratix II enhanced and fast PLLs use separate V_{CC} and ground pins to isolate circuitry and improve noise resistance.

V_{CCA} & GNDA

Each enhanced and fast PLL uses separate V_{CC} 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 V_{CCA} power pin to a 1.2-V power supply, even if you do not use the PLL. Isolate the power connected to V_{CCA} from the power to the rest of the Stratix II device or any other digital device on the board. You can use one of three different methods of isolating the V_{CCA} pin: separate V_{CCA} power planes, a partitioned V_{CCA} island within the V_{CCINT} plane, and thick V_{CCA} traces.

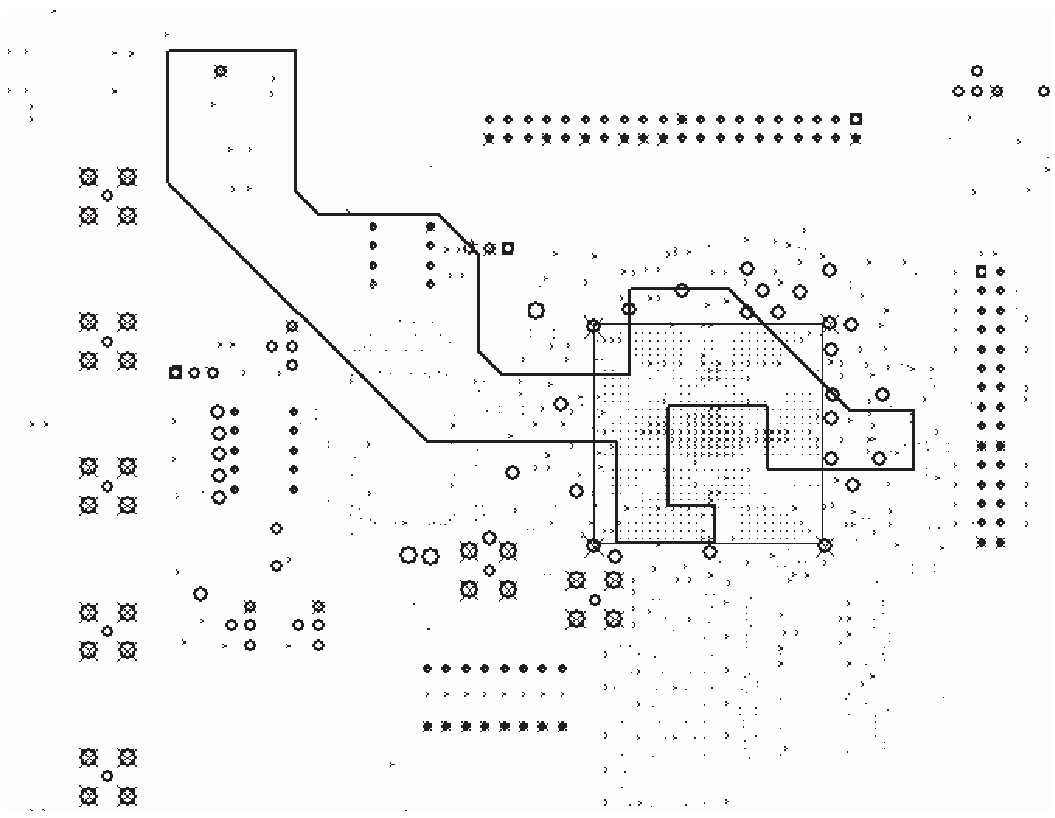
Separate V_{CCA} Power Plane

A mixed signal system is already partitioned into analog and digital sections, each with its own power planes on the board. To isolate the V_{CCA} pin using a separate V_{CCA} power plane, connect the V_{CCA} pin to the analog 1.2-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. Since it is expensive to add new planes to the board, you can create islands for V_{CCA_PLL}. Figure 1–34 shows an example board layout with an analog power island. The dielectric boundary that creates the island should be 25 mils thick. Figure 1–35 shows a partitioned plane within V_{CCINT} for V_{CCA}.

Figure 1–34. V_{CCINT} Plane Partitioned for V_{CCA} Island

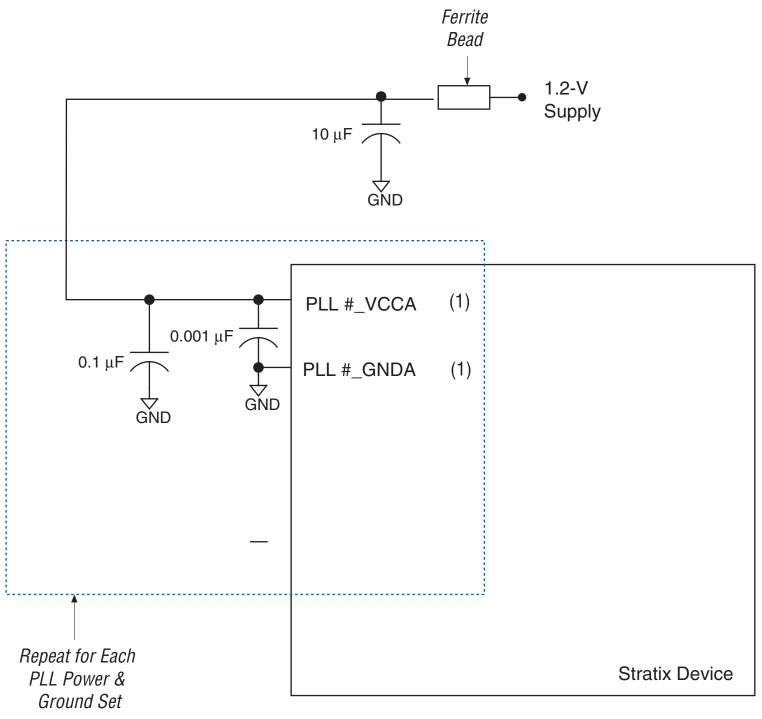


Thick V_{CCA} Trace

Due to board constraints, you may 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, as shown in Figure 1–35. 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 II device. You can connect the `GNDA` pins directly to the same ground plane as the device's digital ground.

Figure 1–35. PLL Power Schematic for Stratix II PLLs



Note to Figure 1–35

(1) Applies to PLLs 1 through 12.

V_{CCD}.

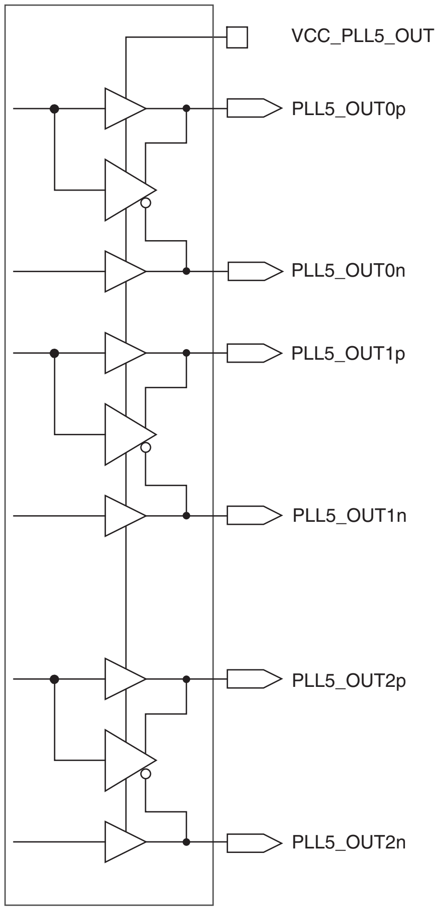
For more information on V_{CCD}, contact Altera Applications.

External Clock Output Power

Enhanced PLLs 5, 6, 11, and 12 also have isolated power pins for their dedicated external clock outputs (`VCC_PLL5_OUT`, `VCC_PLL6_OUT`, `VCC_PLL11_OUT` and `VCC_PLL12_OUT`, respectively). Since the dedicated external clock outputs from a particular enhanced PLL are powered by separate power pins, they are less susceptible to noise. They also reduce the overall jitter of the output clock by providing improved isolation from switching I/O pins.

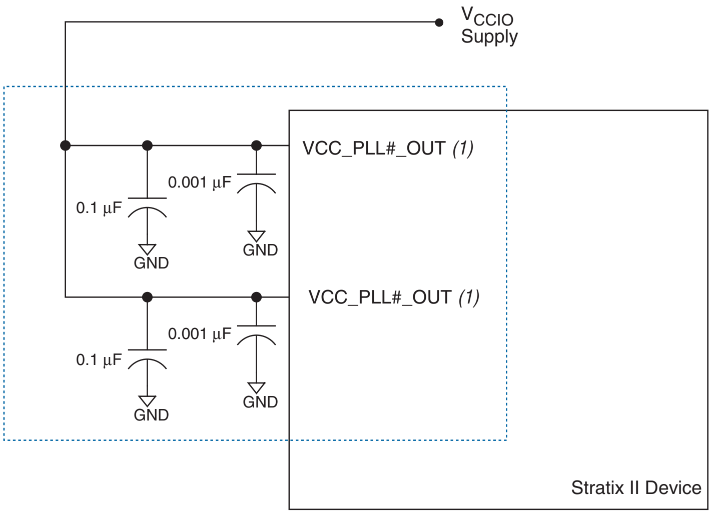
These outputs can by powered by 3.3, 2.5, 1.8, or 1.5 V, depending on the I/O standard for the clock output from a particular enhanced PLL, as shown in Figure 1–36.

Figure 1–36. External Clock Output Pin Association with Output Power



Filter each isolated power pin with a decoupling circuit shown in Figure 1–37. Decouple the isolated power pins with parallel combination of 0.1- and 0.001-µF ceramic capacitors located as close as possible to the Stratix II device.

Figure 1–37. Stratix II PLL External Clock Output Power Ball Connection *Note (1)*



Note to Figure 1–37:
(1) Applies only to enhanced PLLs 5, 6, 11, and 12.

Guidelines

Use the following guidelines for optimal jitter performance on the external clock outputs from enhanced PLLs 5, 6, 11, and 12. If all outputs are running at the same frequency, these guidelines are not necessary to improve performance.

- 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.

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 lower current strength. The higher-frequency output should have an improved performance, but this may degrade the performance of your lower-frequency clock output.

PLL Specifications

See the *DC & Switching Characteristics* chapter in Volume 1 of the *Stratix II Device Handbook* for information on PLL timing specifications

Clocking

Stratix II 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 II devices provide 16 dedicated global clock networks and 32 regional clock networks. These clocks are organized into a hierarchical clock structure that allows for 24 unique clock sources per device quadrant with low skew and delay. This hierarchical clocking scheme provides up to 48 unique clock domains within the entire Stratix II device. Table 1–16 lists the clock resources available on Stratix II devices.

There are 16 dedicated clock pins (`CLK[15..0]`) on Stratix II devices to drive either the global or regional clock networks. Four clock pins drive each side of the Stratix II device, as shown in Figures 1–38 and 1–39**.** Enhanced and fast PLL outputs can also drive the global and regional clock networks.

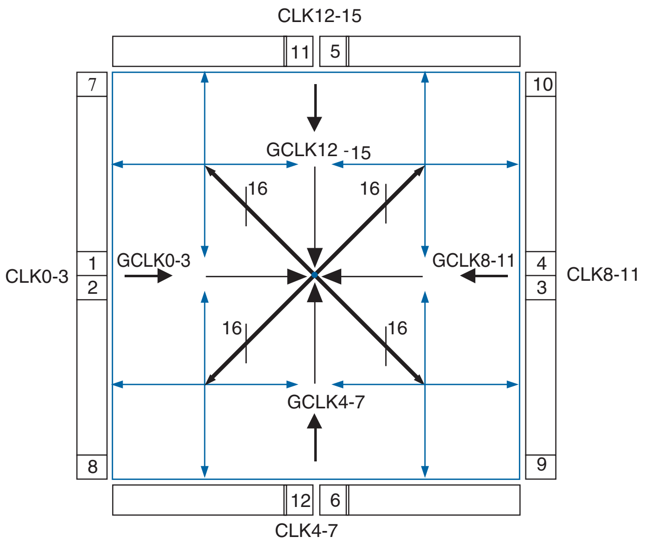
Table 1–16. Clock Resource Availability in Stratix II Devices

Description	Stratix II Device Availability
Number of clock input pins	24
Number of GCLK networks	16
Number of RCLK networks	32
GCLK input sources	Clock input pins, PLL outputs, logic array
RCLK input sources	Clock input pins, PLL outputs, logic array
Number of unique clock sources in a quadrant	24 (16 GCLK and 8 RCLK clocks)
Number of unique clock sources in the entire device	48 (16 GCLK and 32 RCLK clocks)
Power-down mode	GCLK, RCLK networks, dual-regional clock region
Clocking regions for high fan-out applications	Quadrant region, dual-regional, entire device via GCLK or RCLK networks

Global Clock Network

Global clocks drive throughout the entire device, feeding all device quadrants. All resources within the device IOEs, adaptive logic modules (ALMs), digital signal processing (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 by an 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 1–38 shows the 16 dedicated CLK pins driving global clock networks.

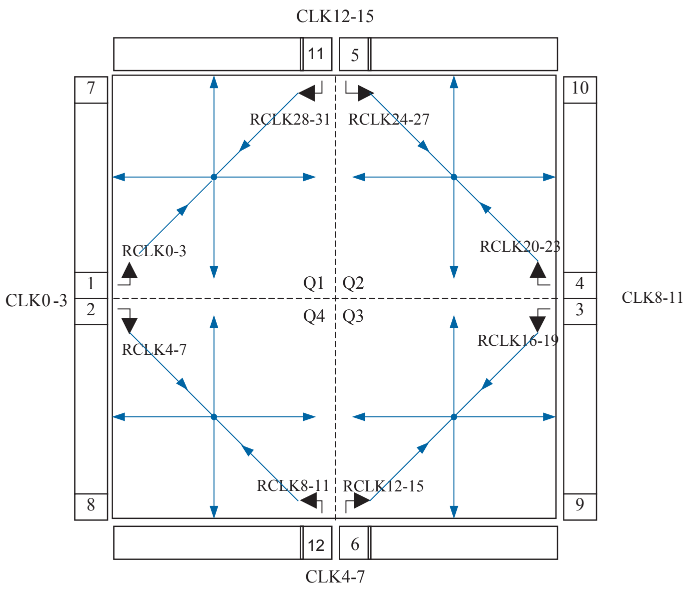
Figure 1–38. Global Clocking



Regional Clock Network

Eight regional clock networks within each quadrant of the Stratix II device are driven by the dedicated CLK[15..0] input pins or from PLL outputs. 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. Internal logic can also drive the regional clock networks for internally generated regional clocks and asynchronous clears, clock enables, or other control signals with large fanout.The CLK clock pins symmetrically drive the RCLK networks within a particular quadrant, as shown in Figure 1–39. Refer to Table 1–17 on page 1–63 and Table 1–18 on page 1–63 for RCLK connections from CLK pins and PLLs.

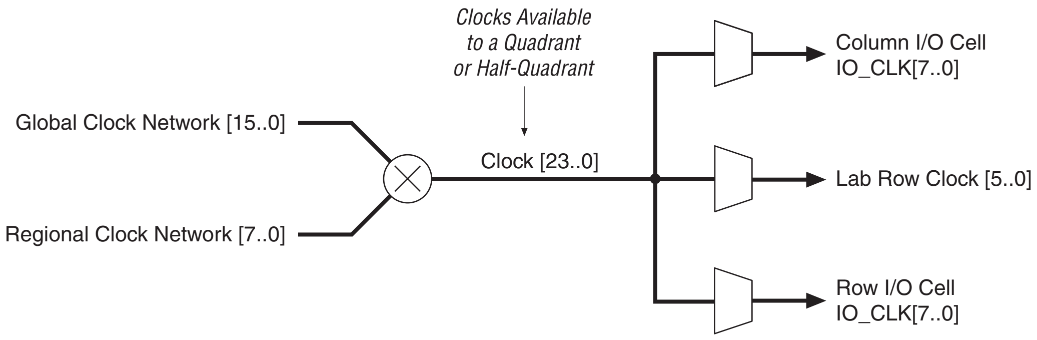
Figure 1–39. Regional Clocking



Clock Sources Per Region

Each Stratix II device has 16 global clock networks and 32 regional clock networks that provide 48 unique clock domains for the entire device. There are 24 unique clocks available in each quadrant (16 GCLK and 8 RCLK clocks) as the input resources for registers (see Figure 1–40).

Figure 1–40. Hierarchical Clock Networks Per Quadrant


Clocks Available to a Quadrant or Half-Quadrant
Global Clock Network [15..0]
Regional Clock Network [7..0]
Clock [23..0]
Column I/O Cell IO_CLK[7..0]
Lab Row Clock [5..0]
Row I/O Cell IO_CLK[7..0]

Stratix II clock networks provide three different clocking regions:

- Entire device clock region
- Quadrant clock region
- Dual-regional clock region

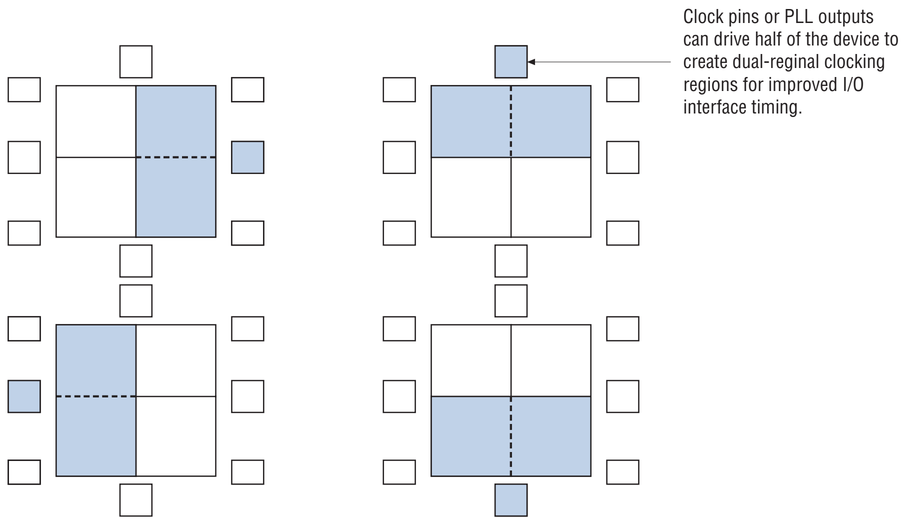
These clock network options provide more flexibility for routing signals that have high fan-out to improve the interface timing. By having various sized clock regions, it is possible to prioritize the number of registers the network can reach versus the total delay of the network.

In the first clock scheme, a source (not necessarily a clock signal) drives a GCLK network that can be routed through the entire device. This has the maximum delay for a low skew high fan-out signal but allows the signal to reach every block within the device. This is a good option for routing global resets or clear signals.

In the second clock scheme, a source drives a single-quadrant region. This represents the fastest, low-skew, high-fan-out signal-routing resource within a quadrant. The limitation to this resource is that it only covers a single quadrant.

In the third clock scheme, a single source (clock pin or PLL output) can generate a dual-regional clock by driving two regional clock network lines (one from each quadrant). This allows logic that spans multiple quadrants to utilize the same low-skew clock. The routing of this signal on an entire side has approximately the same speed as in a quadrant clock region. The internal logic-array routing that can drive a regional clock also supports this feature. This means internal logic can drive a dual-regional clock network. Corner fast PLL outputs only span one quadrant and hence cannot form a dual-regional clock network. Figure 1–41 shows this feature pictorially.

Figure 1–41. Stratix II Dual-Regional Clock Region


Clock pins or PLL outputs can drive half of the device to create dual-reginal clocking regions for improved I/O interface timing.

The 16 clock input pins, enhanced or fast PLL outputs, and internal logic array can be the clock input sources to drive onto either GCLK or RCLK networks. The `CLKn` pins also drive the global clock network as shown in Table 1–20 on page 1–66. Tables 1–17 and 1–18 for the connectivity between `CLK` pins as well as the global and regional clock networks.

Clock Inputs

The clock input pins `CLK[15..0]` are also used for high fan-out control signals, such as asynchronous clears, presets, clock enables, or protocol signals such as `TRDY` and `IRDY` for PCI through GCLK or RCLK networks.

Internal Logic Array

Each GCLK and RCLK network can also be driven by logic-array routing to enable internal logic to drive a high fan-out, low-skew signal.

PLL Outputs

All clock networks can be driven by the PLL counter outputs.

Table 1–17 shows the connection of the clock pins to the global clock resources. The reason for the higher level of connectivity is to support user controllable global clock multiplexing.

Table 1–17. Clock Input Pin Connectivity to Global Clock Networks

Clock resource	CLK (Pin)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
GCLK0	✓	✓														
GCLK1	✓	✓														
GCLK2			✓	✓												
GCLK3			✓	✓												
GCLK4					✓	✓										
GCLK5					✓	✓										
GCLK6							✓	✓								
GCLK7							✓	✓								
GCLK8									✓	✓						
GCLK9									✓	✓						
GCLK10											✓	✓				
GCLK11											✓	✓				
GCLK12													✓	✓		
GCLK13													✓	✓		
GCLK14															✓	✓
GCLK15															✓	✓

Table 1–18 summarizes the connectivity between the clock pins and the regional clock networks. Here, each clock pin can drive two regional clock networks, facilitating stitching of the clock networks to support the ability to drive two quadrants with the same clock or signal.

Table 1–18. Clock Input Pin Connectivity to Regional Clock Networks (Part 1 of 2)

Clock Resource	CLK (Pin)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RCLK0	✓															
RCLK1		✓														
RCLK2			✓													
RCLK3				✓												

Table 1–18. Clock Input Pin Connectivity to Regional Clock Networks (Part 2 of 2)

Clock Resource	CLK (Pin)															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RCLK4	✓															
RCLK5		✓														
RCLK6			✓													
RCLK7				✓												
RCLK8					✓											
RCLK9						✓										
RCLK10							✓									
RCLK11								✓								
RCLK12					✓											
RCLK13						✓										
RCLK14							✓									
RCLK15								✓								
RCLK16									✓							
RCLK17										✓						
RCLK18											✓					
RCLK19												✓				
RCLK20									✓							
RCLK21										✓						
RCLK22											✓					
RCLK23												✓				
RCLK24													✓			
RCLK25														✓		
RCLK26															✓	
RCLK27																✓
RCLK28													✓			
RCLK29														✓		
RCLK30															✓	
RCLK31																✓

Clock Input Connections

Four CLK pins drive each enhanced PLL. You can use any of the pins for clock switchover inputs into the PLL. The CLK pins are the primary clock source for clock switchover, which is controlled in the Quartus II software. Enhanced PLLs 5, 6, 11, and 12 also have feedback input pins, as shown in Table 1–19.

Input clocks for fast PLLs 1, 2, 3, and 4 come from CLK pins. 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 an 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 applications. CLK pins cannot drive these fast PLLs in high-speed differential I/O mode.

Table 1–19 shows which PLLs are available in each Stratix II device and which input clock pin drives which PLLs.

Table 1–19. Stratix II Device PLLs & PLL Clock Pin Drivers (Part 1 of 2)

Input Pin	All Devices						EP2S60 to EP2S180 Devices					
	Fast PLLs				Enhanced PLLs		Fast PLLs				Enhanced PLLs	
	1	2	3	4	5	6	7	8	9	10	11	12
CLK0	✓	✓					✓(1)	✓(1)				
CLK1 (2)	✓	✓					✓(1)	✓(1)				
CLK2	✓	✓					✓(1)	✓(1)				
CLK3 (2)	✓	✓					✓(1)	✓(1)				
CLK4						✓						✓
CLK5						✓						✓
CLK6						✓						✓
CLK7						✓						✓
CLK8			✓	✓					✓(1)	✓(1)		
CLK9(2)			✓	✓					✓(1)	✓(1)		
CLK10			✓	✓					✓(1)	✓(1)		
CLK11(2)			✓	✓					✓(1)	✓(1)		
CLK12					✓						✓	
CLK13					✓						✓	
CLK14					✓						✓	

Table 1–19. Stratix II Device PLLs & PLL Clock Pin Drivers (Part 2 of 2)

Input Pin	All Devices						EP2S60 to EP2S180 Devices					
	Fast PLLs				Enhanced PLLs		Fast PLLs				Enhanced PLLs	
	1	2	3	4	5	6	7	8	9	10	11	12
CLK15					✓						✓	
PLL5_FB					✓							
PLL6_FB						✓						
PLL11_FB											✓	
PLL12_FB												✓
PLL_ENA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
FPLL7CLK *(2)*							✓					
FPLL8CLK *(2)*								✓				
FPLL9CLK *(2)*									✓			
FPLL10CLK *(2)*										✓		

Notes to Table 1–19:
(1) Clock connection is available. For more information on the maximum frequency, contact Altera Applications.
(2) This is a dedicated high-speed clock input. For more information on the maximum frequency, contact Altera Applications.

CLK(n) Pin Connectivity to Global Clock Networks

In Stratix II devices, the `clk(n)` pins can also feed the global clock network. Table 1–20 shows the `clk(n)` pin connectivity to global clock networks.

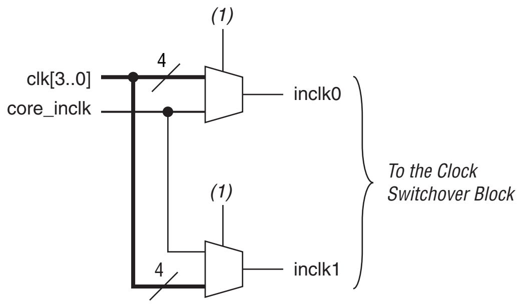
Table 1–20. CLK(n) Pin Connectivity to Global Clock Network

Clock Resource	CLK(n) pin							
	4	5	6	7	12	13	14	15
GCLK4	✓							
GCLK5		✓						
GCLK6			✓					
GCLK7				✓				
GCLK12					✓			
GCLK13						✓		
GCLK14							✓	
GCLK15								✓

Clock Source Control For Enhanced PLLs

The clock input multiplexer for enhanced PLLs is shown in Figure 1–42. This block allows selection of the PLL clock reference from several different sources. The clock source to an enhanced PLL can come from any one of four clock input pins `CLK[3..0]`, or from a logic-array clock. The clock input pin connections to the respective enhanced PLLs are shown in Table 1–19 above. The multiplexer select lines are set in the configuration file only. Once programmed, this block cannot be changed without loading a new configuration file. The Quartus II software automatically sets the multiplexer select signals depending on the clock sources that a user selects in the design.

Figure 1–42. Enhanced PLL Clock Input Multiplex Logic



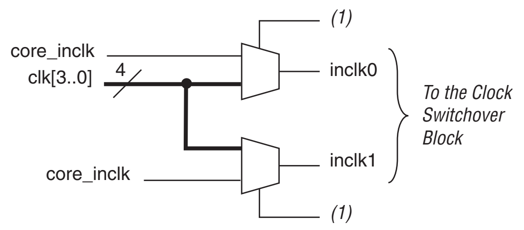
Note to Figure 1–42:

(1) The input clock multiplexing is controlled through a configuration file only and cannot be dynamically controlled in user mode.

Clock Source Control for Fast PLLs

Each center fast PLL has five clock input sources, four from clock input pins, and one from a logic array signal. When using clock input pins as the clock source, you can perform manual clock switchover among the input clock sources. The clock input multiplexer control signals for performing clock switchover are from core signals. Figure 1–43 shows the clock input multiplexer control circuit for a center fast PLL.

Figure 1–43. Center Fast PLL Clock Input Multiplexer Control

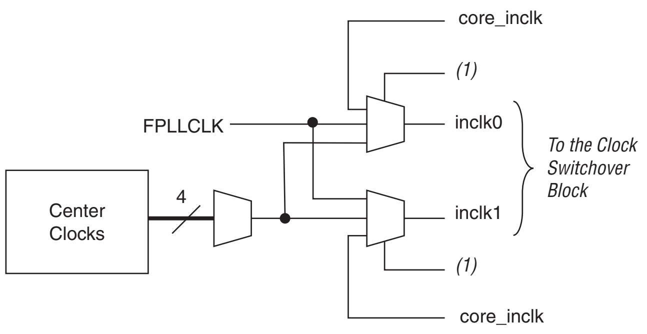


Note to Figure 1–43:

(1) The input clock multiplexing is controlled through a configuration file only and cannot be dynamically controlled in user mode.

Each corner fast PLL has three clock input sources, one from a dedicated corner clock input pin, one from a center clock input pin, and one from a logic array clock. Figure 1–44 shows a block diagram showing the clock input multiplexer control circuit for a corner fast PLL. Only the corner `FPLLCLK` pin is fully compensated.

Figure 1–44. Corner Fast PLL Clock Input Multiplexer Control



Note to Figure 1–44:

(1) The input clock multiplexing is controlled through a configuration file only and cannot be dynamically controlled in user mode.

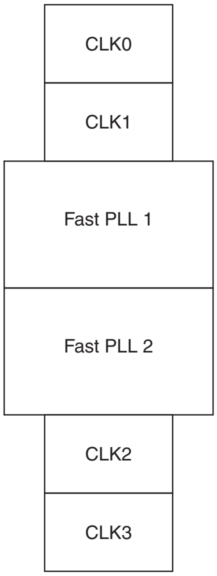
Delay Compensation for Fast PLLs

Each center fast PLL can be fed by any one of four possible input clock pins. Among the four clock input signals, only two are fully compensated, i.e., the clock delay to the fast PLL matches the delay in the data input path when used in the LVDS receiver mode. The two clock inputs that match the data input path are located right next to the fast

PLL. The two clock inputs that do not match the data input path are located next to the neighboring fast PLL. Figure 1–45 shows the above description for the left side center fast PLL pair.

Fast PLL 1 and PLL 2 can choose among `CLK[3..0]` as the clock input source. However, for fast PLL 1, only `CLK0` and `CLK1` have their delay matched to the data input path delay when used in the LVDS receiver mode operation. The delay from `CLK2` or `CLK3` to fast PLL 1 does not match the data input delay. For fast PLL 2, only `CLK2` and `CLK3` have their delay matched to the data input path delay in LVDS receiver mode operation. The delay from `CLK0` or `CLK1` to fast PLL 2 does not match the data input delay. The same arrangement applies to the right side center fast PLL pair. For corner fast PLLs, only the corner `FPLLCLK` pins are fully compensated. For LVDS receiver operation, it is recommended to use the delay compensated clock pins only.

Figure 1–45. Delay Compensated Clock Input Pins for Center Fast PLL Pair



Clock Output Connections

Enhanced PLLs have outputs for eight regional clock outputs and four global clock outputs. There is line sharing between clock pins, global and regional clock networks and all PLL outputs. See Tables 1–17 through 1–21 and Figures 1–46 through 1–50 to validate your clocking scheme.

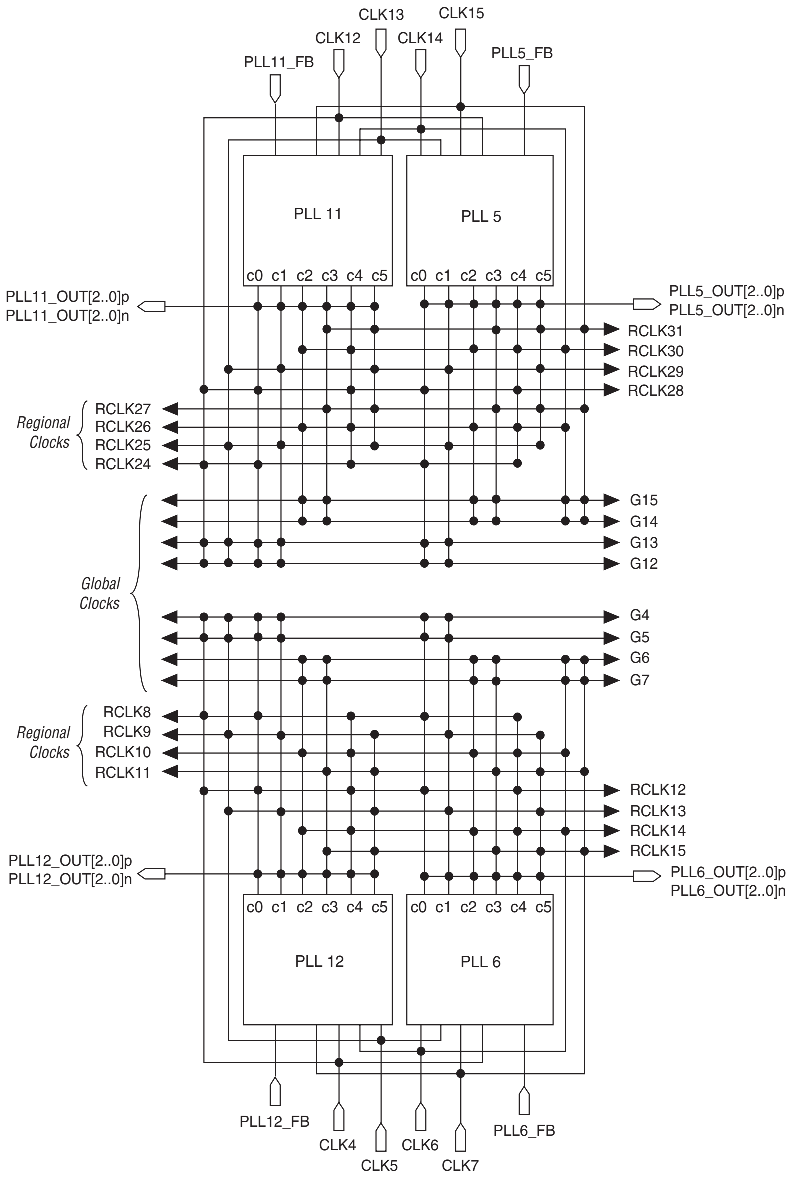
The Quartus II software automatically maps to regional and global clocks to avoid any restrictions. Enhanced PLLs 5, 6, 11, and 12 drive out to single-ended pins as shown in Table 1–21.

You can connect each fast PLL 1, 2, 3, or 4 output (`C0`, `C1`, `C2`, and `C3`) to either a global or a regional clock. There is line sharing between clock pins, `FPLLCLK` pins, global and regional clock networks, and all PLL outputs. The Quartus II software will automatically map to regional and global clocks to avoid any restrictions.

Figure 1–46 shows the clock input and output connections from the enhanced PLLs.

☞ EP2S15 and EP2S30 devices have only two enhanced PLLs (5, 6), but the connectivity from these two PLLs to the global or regional clock networks remains the same.

Figure 1–46. Stratix II Top & Bottom Enhanced PLLs, Clock Pin & Logic Array Signal Connectivity to Global & Regional Clock Networks *Note (1)*



Note to Figure 1–46:

(1) The redundant connection dots facilitate stitching of the clock networks to support the ability to drive two quadrants with the same clock.

Table 1–21 shows the global and regional clocks that the PLL outputs drive.

Table 1–21. Stratix II Global & Regional Clock Outputs From PLLs (Part 1 of 2)

Clock Network	PLL Number & Type											
	EP2S15 through EP2S30 Devices						EP2S60 through EP2S180 Devices					
	Fast PLLs				Enhanced PLLs		Fast PLLs				Enhanced PLLs	
	1	2	3	4	5	6	7	8	9	10	11	12
GCLK0	✓	✓					✓	✓				
GCLK1	✓	✓					✓	✓				
GCLK2	✓	✓					✓	✓				
GCLK3	✓	✓					✓	✓				
GCLK4						✓						✓
GCLK5						✓						✓
GCLK6						✓						✓
GCLK7						✓						✓
GCLK8			✓	✓					✓	✓		
GCLK9			✓	✓					✓	✓		
GCLK10			✓	✓					✓	✓		
GCLK11			✓	✓					✓	✓		
GCLK12					✓						✓	
GCLK13					✓						✓	
GCLK14					✓						✓	
GCLK15					✓						✓	
RCLK0	✓	✓					✓					
RCLK1	✓	✓					✓					
RCLK2	✓	✓					✓					
RCLK3	✓	✓					✓					
RCLK4	✓	✓						✓				
RCLK5	✓	✓						✓				
RCLK6	✓	✓						✓				
RCLK7	✓	✓						✓				
RCLK8						✓						✓
RCLK9						✓						✓
RCLK10						✓						✓

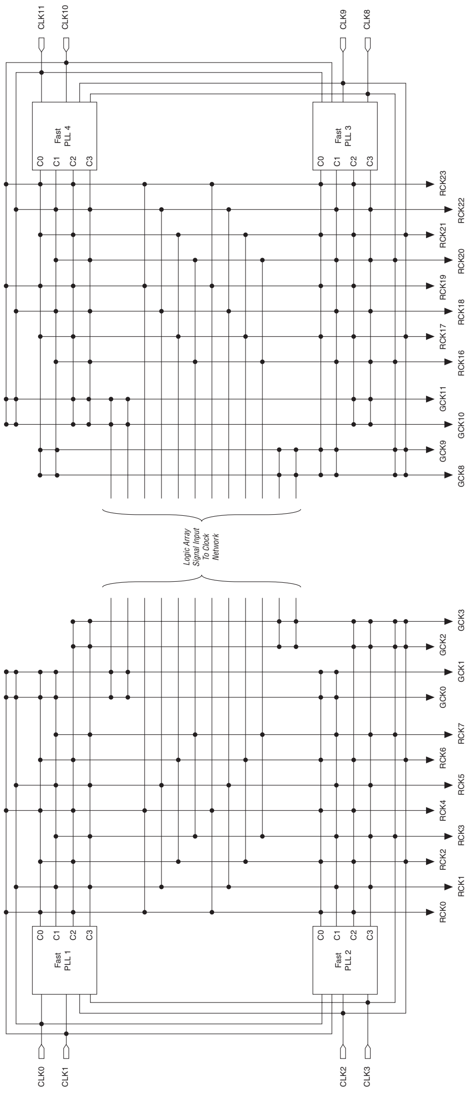
Table 1–21. Stratix II Global & Regional Clock Outputs From PLLs (Part 2 of 2)

Clock Network	PLL Number & Type											
	EP2S15 through EP2S30 Devices						EP2S60 through EP2S180 Devices					
	Fast PLLs				Enhanced PLLs		Fast PLLs				Enhanced PLLs	
	1	2	3	4	5	6	7	8	9	10	11	12
RCLK11						✓						✓
RCLK12						✓						✓
RCLK13						✓						✓
RCLK14						✓						✓
RCLK15						✓						✓
RCLK16			✓	✓					✓			
RCLK17			✓	✓					✓			
RCLK18			✓	✓					✓			
RCLK19			✓	✓					✓			
RCLK20			✓	✓						✓		
RCLK21			✓	✓						✓		
RCLK22			✓	✓						✓		
RCLK23			✓	✓						✓		
RCLK24					✓						✓	
RCLK25					✓						✓	
RCLK26					✓						✓	
RCLK27					✓						✓	
RCLK28					✓						✓	
RCLK29					✓						✓	
RCLK30					✓						✓	
RCLK31					✓						✓	
External Clock Output												
PLL5_OUT[3..0]p/n					✓							
PLL6_OUT[3..0]p/n						✓						
PLL11_OUT[3..0]p/n											✓	
PLL12_OUT[3..0]p/n												✓

The fast PLLs also drive high-speed SERDES clocks for differential I/O interfacing. For information on these `FPLLCLK` pins, contact Altera Applications.

Figures 1–47 and 1–48 show the global and regional clock input and output connections from the Stratix II fast PLLs.

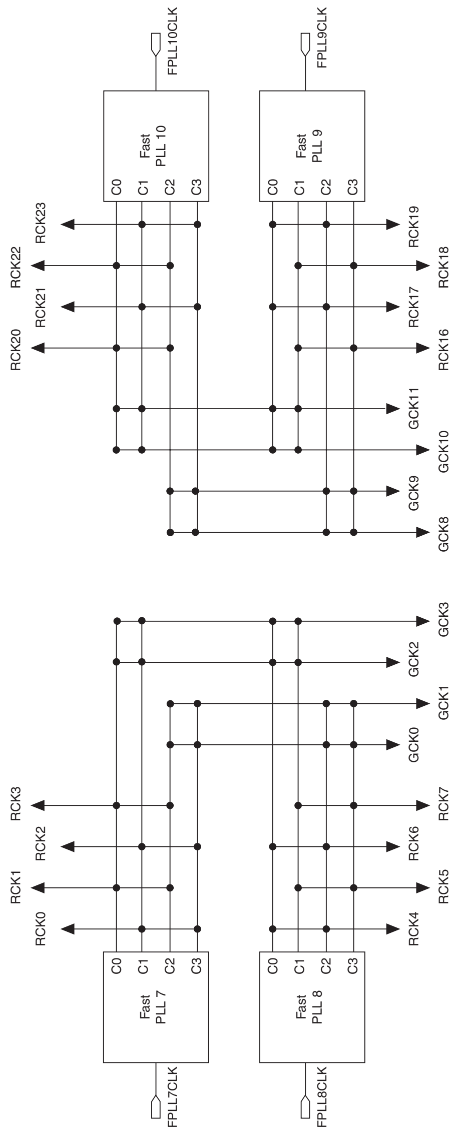
Figure 1–47. Stratix II Center Fast PLLs, Clock Pin & Logic Array Signal Connectivity to Global & Regional Clock Networks *Note (1)*



Note to Figure 1–47:

(1) The redundant connection dots facilitate stitching of the clock networks to support the ability to drive two quadrants with the same clock.

Figure 1–48. Corner Fast PLLs, Clock Pin & Logic Array Signal Connectivity to Global & Regional Clock Networks *Note (1)*



Note to Figure 1–48:

(1) The corner FPLLs can also be driven through the global or regional clock networks. The global or regional clock input to the FPLL can be driven from another PLL, a pin-driven global or regional clock, or internally-generated global signal.

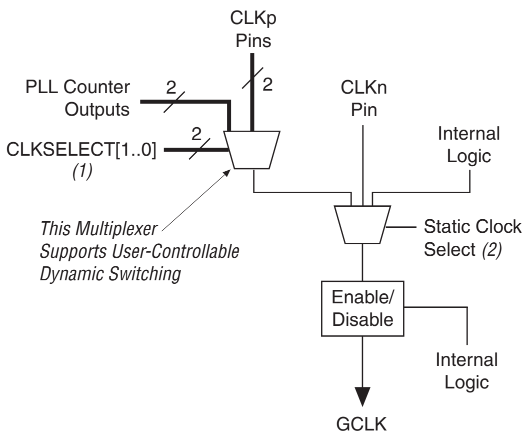
Clock Control Block

Each global and regional clock has its own clock control block. The control block has two functions:

- Clock source selection (dynamic selection for global clocks)
- Clock power-down (dynamic clock enable or disable)

Figures 1–49 and 1–50 show the global clock and regional clock select blocks, respectively.

Figure 1–49. Stratix II Global Clock Control Block

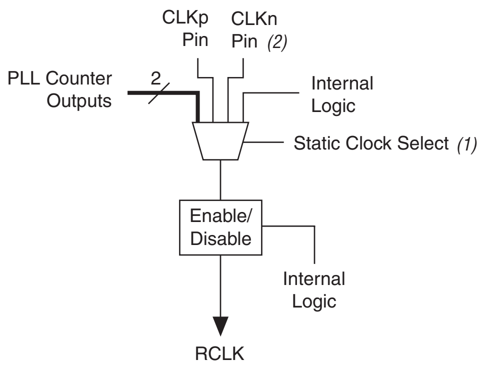


Notes to Figure 1–49:

(1) These clock select signals can only be dynamically controlled through internal logic when the device is operating in user mode.

(2) These clock select signals can only be set through a configuration file and cannot be dynamically controlled during user-mode operation.

Figure 1–50. Regional Clock Control Block



Notes to Figure 1–50:

(1) These clock select signals can only be dynamically controlled through a configuration file and cannot be dynamically controlled during user-mode operation.

(2) Only the `CLK`*n* pins on the top and bottom for the device feed to regional clock select blocks.

For the global clock select block, the clock source selection can be controlled either statically or dynamically. You have the option to statically select the clock source in configuration file generated by the Quartus II software, or you can control the selection dynamically by using internal logic to drive the multiplexer select inputs. When selecting statically, the clock source can be set to any of the inputs to the select multiplexer. When selecting the clock source dynamically, you can either select two PLL outputs (such as `CLK0` or `CLK1`), or a combination of clock pins or PLL outputs.

When using the `altclkctrl` megafunction to implement clock source (dynamics) selection, the inputs from the clock pins feed the `inclock[0..1]` ports of the multiplexer, while the PLL outputs feed the `inclock[2..3]` ports. You can choose from among these inputs using the `CLKSELECT[1..0]` signal.

For the regional clock select block, the clock source selection can only be controlled statically using configuration bits. Any of the inputs to the clock select multiplexer can be set as the clock source.

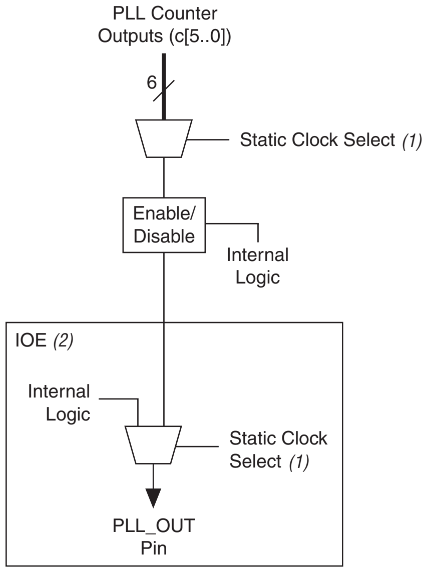
The Stratix II clock networks can be disabled (powered down) by both static and dynamic approaches. When a clock net is powered down, all the logic fed by the clock net is in an off-state, thereby reducing the overall power consumption of the device.

The global and regional clock networks that are not used are automatically powered down through configuration bit settings in the configuration file (SRAM Object File (**.sof**) or Programmer Object File (**.pof**)) generated by the Quartus II software.

The dynamic clock enable or disable feature allows the internal logic to control power up or down synchronously on `GCLK` and `RCLK` nets, including dual-regional clock regions. This function is independent of the PLL and is applied directly on the clock network, as shown in Figure 1–49 on page 1–76 and Figure 1–50 on page 1–77.

The input clock sources and the `clkena` signals for the global and regional clock network multiplexers can be set through the Quartus II software using the `altclkctrl` megafunction. The dedicated external clock output pins can also be enabled or disabled using the `altclkctrl` megafunction. Figure 1–51 shows the external PLL output clock control block.

Figure 1–51. Stratix II External PLL Output Clock Control Block


PLL Counter Outputs (c[5..0])
6
Static Clock Select *(1)*
Enable/ Disable
Internal Logic
IOE *(2)*
Internal Logic
Static Clock Select *(1)*
PLL_OUT Pin

Notes to Figure 1–51:

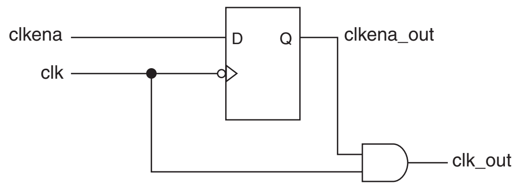
(1) These clock select signals can only be set through a configuration file and cannot be dynamically controlled during user mode operation.

(2) The clock control block feeds to a multiplexer within the `PLL_OUT` pin’s IOE. The `PLL_OUT` pin is a dual-purpose pin. Therefore, this multiplexer selects either an internal signal or the output of the clock control block.

clkena Signals

Figure 1–52 shows how `clkena` is implemented.

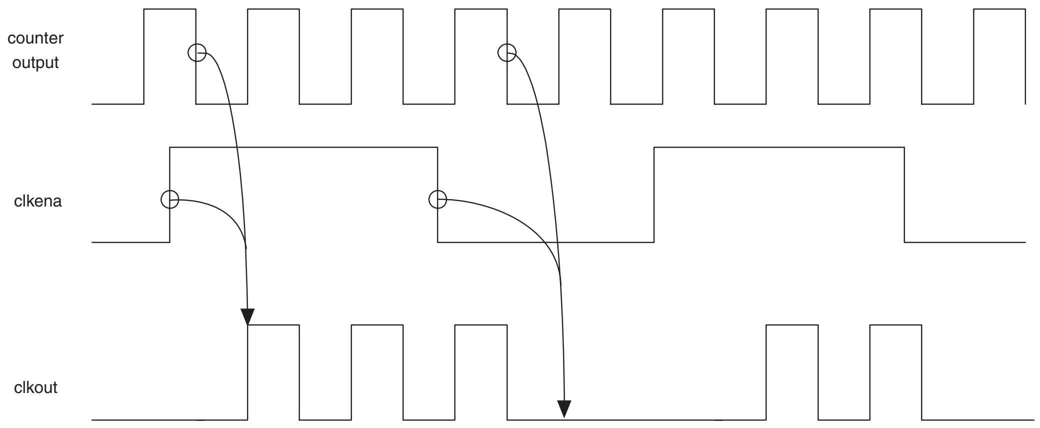
Figure 1–52. clkena Implementation



In Stratix II devices, the `clkena` signals are supported at the clock network level. This allows you to gate off the clock even when a PLL is not being used.

The `clkena` signals can also be used to control the dedicated external clocks from enhanced PLLs. Upon re-enabling, the PLL does not need a resynchronization or relock period unless the PLL is using external feedback mode. Figure 1–53 shows the waveform example for a clock output enable. `clkena` is synchronous to the falling edge of the counter output.

Figure 1–53. Clkena Signals



Note to Figure 1–53

(1) The `clkena` signals can be used to enable or disable the GCLK and RCLK networks or the `PLL_OUT` pins.

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.

Conclusion

Stratix II device enhanced and fast PLLs provide you with complete control of device 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® II devices and the supported external memory interfaces.

This section contains the following chapters:

- Chapter 2, TriMatrix Embedded Memory Blocks in Stratix II Devices
- Chapter 3, External Memory Interfaces

Revision History

The table below shows the revision history for Chapters 2 and 3.

Chapter	Date / Version	Changes Made
2	March 2005, v2.1	Updated Table 2–1.
	January 2005, v2.0	● Updated the “M512 Blocks” and “M4K Blocks” sections. ● Updated Tables 2–1.
	July 2004, v1.1	● Updated “Mixed-Port Read-During-Write Mode” section. ● Updated Figures 2–2, 2–3, and 2–4. ● Updated “Address Clock Enable Support” section.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.
3	March 2005, v2.2	Updated Table 3–1.
	January 2005, v2.1	Minor content changes.
	January 2005, v2.0	● Updated the “External Memory Standards” and “Stratix II DDR Memory Support Overview” sections. ● Updated Figures 3–1, 3–7, and 3–11. ● Added Table 3–2. ● Updated Tables 3–2 and Tables 3–4.
	October 2004, v1.2	Updated Tables 3–4 and 3–5.
	July 2004, v1.1	● Modified Figure 3–2 to show that DDR only supports burst lengths of four. ● Updated notes for Tables 3–2 and 3–4. ● Added Table 3–5. ● Changed “tristate” to “high-impedance state.” ● Revised notes for Figures 3–9 and 3–12.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.

2. TriMatrix Embedded Memory Blocks in Stratix II Devices

SII52002-2.1

Introduction

Stratix® II devices feature the TriMatrix™ memory structure, consisting of three sizes of embedded RAM blocks that efficiently address the memory needs of FPGA designs.

TriMatrix memory includes 512-bit M512 blocks, 4-Kbit M4K blocks, and 512-Kbit M-RAM blocks, which are each configurable to support many features. TriMatrix memory provides up to 9 megabits of RAM at up to 400 MHz operation, and up to 16 terabits per second of total memory bandwidth per device. This chapter describes TriMatrix memory blocks, modes, and features.

TriMatrix Memory Overview

The TriMatrix architecture provides complex memory functions for different applications in FPGA designs. For example, M512 blocks are used for first-in first-out (FIFO) functions and clock domain buffering where memory bandwidth is critical; M4K blocks are ideal for applications requiring medium-sized memory, such as asynchronous transfer mode (ATM) cell processing; and M-RAM blocks are suitable for large buffering applications, such as internet protocol (IP) packet buffering and system cache.

The TriMatrix memory blocks support various memory configurations, including single-port, simple dual-port, true dual-port (also known as bidirectional dual-port), shift register, and read-only memory (ROM) modes. The TriMatrix memory architecture also includes advanced features and capabilities, such as parity-bit support, byte enable support, pack mode support, address clock enable support, mixed port width support, and mixed clock mode support.

Table 2–1 summarizes the features supported by the three sizes of TriMatrix memory.

Table 2–1. Summary of TriMatrix Memory Features

Feature	M512 Blocks	M4K Blocks	M-RAM Blocks
Maximum performance	500 MHz	550 MHz	400 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 8K × 64 8K × 72 4K × 128 4K × 144
Parity bits	✓	✓	✓
Byte enable	✓	✓	✓
Pack mode		✓	✓
Address clock enable		✓	✓
Single-port memory	✓	✓	✓
Simple dual-port memory	✓	✓	✓
True dual-port memory		✓	✓
Embedded shift register	✓	✓	
ROM	✓	✓	
FIFO buffer	✓	✓	✓
Simple dual-port mixed width support	✓	✓	✓
True dual-port mixed width support		✓	✓
Memory initialization file (**.mif**)	✓	✓	
Mixed-clock mode	✓	✓	✓
Power-up condition	Outputs cleared	Outputs cleared	Outputs unknown
Register clears	Output registers only	Output registers only	Output registers only
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

Table 2–2 shows the capacity and distribution of the TriMatrix memory blocks in each Stratix II family member.

Table 2–2. TriMatrix Memory Capacity & Distribution in Stratix II Devices

Device	M512 Columns/Blocks	M4K Columns/Blocks	M-RAM Blocks	Total RAM Bits
EP2S15	4/104	3/78	0	419,328
EP2S30	6/202	4/144	1	1,369,728
EP2S60	7/329	5/255	2	2,544,192
EP2S90	8/488	6/408	4	4,520,448
EP2S130	9/699	7/609	6	6,747,840
EP2S180	11/930	8/768	9	9,383,040

Parity Bit Support

All TriMatrix memory blocks (M512, M4K, and M-RAM) support one parity bit for each byte.

Parity bits add to the amount of memory in each random access memory (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 adaptive logic modules (ALMs), implements parity checking for error detection to ensure data integrity. Parity-size data words can also be used for other purposes such as storing user-specified control bits.

See the *Using Parity to Detect Memory Errors in Stratix Devices* White Paper for more information on using the parity bit to detect memory errors.

Byte Enable Support

All TriMatrix memory blocks support byte enables that mask the input data so that only specific bytes, nibbles, or bits of data are written. The unwritten bytes or bits retain the previous written value. The write enable (`wren`) signals, along with the byte enable (`byteena`) signals, control the RAM blocks' write operations. The default value for the byte enable signals is high (enabled), in which case writing is controlled only by the write enable signals. There is no clear port to the byte enable registers.

M512 Blocks

M512 blocks support byte enables for all data widths, including 1, 2, 4, 8, 9, 16, and 18 bits. For memory block configurations with widths of less than one byte (×8/×9), the byte-enable feature is only supported if the memory block is instantiated as the full width of the memory configuration. For example, a 128 × 4 memory block supports the byte enable. However, you can not use the byte-enable feature on two groups of four bits in a 64 × 8 memory block. For memory configurations less than one byte, the write enable or clock enable signals can optionally be used to control the write operation. Table 2–3 summarizes the byte selection.

Table 2–3. Byte Enable for Stratix II M512 Blocks *Note (1)*

byteena[1..0]	data ×1	data ×2	data ×4	data ×8	data ×9	data ×16	data ×18
[0] = 1	[0]	[1..0]	[3..0]	[7..0]	[8..0]	[7..0]	[8..0]
[1] = 1	-	-	-	-	-	[15..8]	[17..9]

Note to Table 2–3:
(1) Any combination of byte enables is possible.

M4K Blocks

M4K blocks support byte enables for all data widths, including 1, 2, 4, 8, 9, 16, 18, 32, and 36 bits. For memory block configurations with widths of less than one byte (×8/×9), the byte-enable feature is only supported if the memory block is instantiated as the full width of the memory configuration. For example, a 1024 × 4 memory block supports the byte enable. However, you can not use the byte-enable feature on two groups of four bits in a 512 × 8 memory block. For memory configurations less than one byte, the write enable or clock enable signals can optionally be used to control the write operation. Table 2–4 summarizes the byte selection.

Table 2–4. Byte Enable for Stratix II M4K Blocks (Part 1 of 2) *Note (1)*

byteena [3..0]	data ×1 *(2)*	data ×2 *(2)*	data ×4 *(2)*	data ×8 *(2)*	data ×9 *(2)*	data ×16	data ×18	data ×32	data ×36
[0] = 1	[0]	[1..0]	[3..0]	[7..0]	[8..0]	[7..0]	[8..0]	[7..0]	[8..0]
[1] = 1	-	-	-	-	-	[15..8]	[17..9]	[15..8]	[17..9]
[2] = 1	-	-	-	-	-	-	-	[23..16]	[26..18]
[3] = 1	-	-	-	-	-	-	-	[31..24]	[35..27]

Table 2–4. Byte Enable for Stratix II M4K Blocks (Part 2 of 2) Note (1)

byteena [3..0]	data ×1 (2)	data ×2 (2)	data ×4 (2)	data ×8 (2)	data ×9 (2)	data ×16	data ×18	data ×32	data ×36

Notes to Table 2–4:
(1) Any combination of byte enables is possible.
(2) For true dual-port mode, for data widths of 1, 2, 4, 8, and 9: `set byteena[0] = 1` and `byteena[2] = 1`, whereas for single port and simple dual-port modes set only `byteena[0] = 1`.

M-RAM Blocks

M-RAM blocks support byte enables for all data widths, including 8, 9, 16, 18, 32, 36, 64, and 72 bits. In the 128× or ×144 simple dual-port mode, the two sets of byte enable signals (`byteena_a` and `byteena_b`) combine to form the necessary 16-byte enables. Table 2–5 summarizes the byte selection for M-RAM blocks.

Table 2–5. Byte Enable for Stratix II M-RAM Blocks Note (1)

byteena	data ×8	data ×9	data ×16	data ×18	data ×32	data ×36	data ×64	data ×72
[0] = 1	[7..0]	[8..0]	[7..0]	[8..0]	[7..0]	[8..0]	[7..0]	[8..0]
[1] = 1	-	-	[15..8]	[17..9]	[15..8]	[17..9]	[15..8]	[17..9]
[2] = 1	-	-	-	-	[23..16]	[26..18]	[23..16]	[26..18]
[3] = 1	-	-	-	-	[31..24]	[35..27]	[31..24]	[35..27]
[4] = 1	-	-	-	-	-	-	[39..32]	[44..36]
[5] = 1	-	-	-	-	-	-	[47..40]	[53..45]
[6] = 1	-	-	-	-	-	-	[55..48]	[62..54]
[7] = 1	-	-	-	-	-	-	[63..56]	[71..63]

Note to Table 2–5:
(1) Any combination of byte enables is possible.

Table 2–6 summarizes the byte selection for ×144 mode.

Table 2–6. Stratix II M-RAM Combined Byte Selection for ×144 Mode (Part 1 of 2) Note (1)

byteena	data ×128	data ×144
[0] = 1	[7..0]	[8..0]
[1] = 1	[15..8]	[17..9]
[2] = 1	[23..16]	[26..18]
[3] = 1	[31..24]	[35..27]

Table 2–6. Stratix II M-RAM Combined Byte Selection for ×144 Mode (Part 2 of 2) Note (1)

byteena	data ×128	data ×144
[4] = 1	[39..32]	[44..36]
[5] = 1	[47..40]	[53..45]
[6] = 1	[55..48]	[62..54]
[7] = 1	[63..56]	[71..63]
[8] = 1	[71..64]	[80..72]
[9] = 1	[79..72]	[89..73]
[10] = 1	[87..80]	[98..90]
[11] = 1	[95..88]	[107..99]
[12] = 1	[103..96]	[116..108]
[13] = 1	[111..104]	[125..117]
[14] = 1	[119..112]	[134..126]
[15] = 1	[127..120]	[143..135]

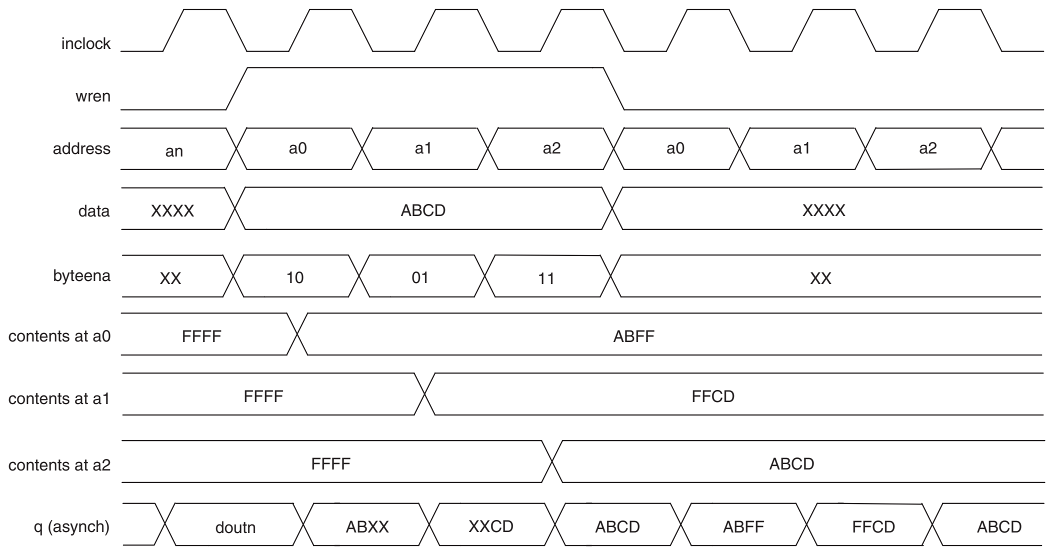
Note to Table 2–6:
(1) Any combination of byte enables is possible.

Byte Enable Functional Waveform

Figure 2–1 shows how the write enable (`wren`) and byte enable (`byteena`) signals control the operations of the RAM.

When a byte enable bit is de-asserted during a write cycle, the corresponding data byte output appears as a “don't care” or unknown value. When a byte enable bit is asserted during a write cycle, the corresponding data byte output will be the newly written data.

Figure 2–1. Stratix II Byte Enable Functional Waveform



Pack Mode Support

Stratix II M4K and M-RAM memory blocks support pack mode. In M4K and M-RAM memory blocks, two single-port memory blocks can be implemented in a single block under the following conditions:

- Each of the two independent block sizes is equal to or less than half of the M4K or M-RAM block size.
- Each of the single-port memory blocks is configured in single-clock mode.

Thus, each of the single-port memory blocks access up to half of the M4K or M-RAM memory resources such as clock, clock enables, and asynchronous clear signals.

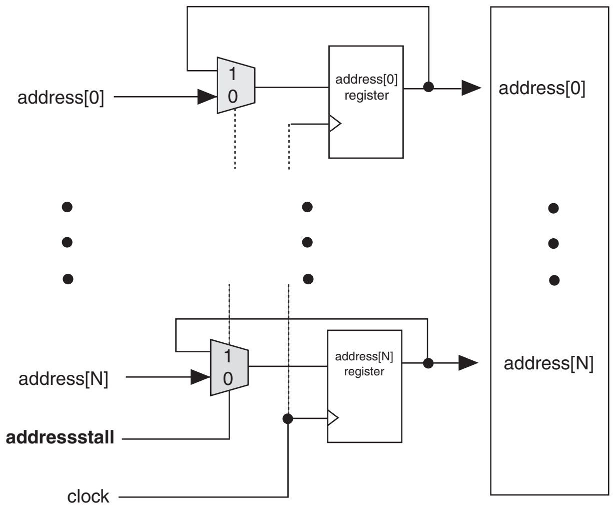
See the "Single-Port Mode" and "Single-Clock Mode" sections of this chapter for more information.

Address Clock Enable Support

Stratix II M4K and M-RAM memory blocks support address clock enable, which is used to hold the previous address value for as long as the signal is enabled. When the memory blocks are configured in dual-port mode, each port has its own independent address clock enable.

Figure 2–2 shows an address clock enable block diagram. Placed in the address register, the address signal output by the address register is fed back to the input of the register via a multiplexer. The multiplexer output is selected by the address clock enable (`addressstall`) signal. Address latching is enabled when the `addressstall` signal turns high. The output of the address register is then continuously fed into the input of the register; therefore, the address value can be held until the `addressstall` signal turns low.

Figure 2–2. Stratix II Address Clock Enable Block Diagram



Address clock enable is typically used for cache memory applications, which require one port for read and another port for write. The default value for the address clock enable signals is low (disabled). Figures 2–3 and 2–4 show the address clock enable waveform during the read and write cycles, respectively.

Figure 2–3. Stratix II Address Clock Enable During Read Cycle Waveform

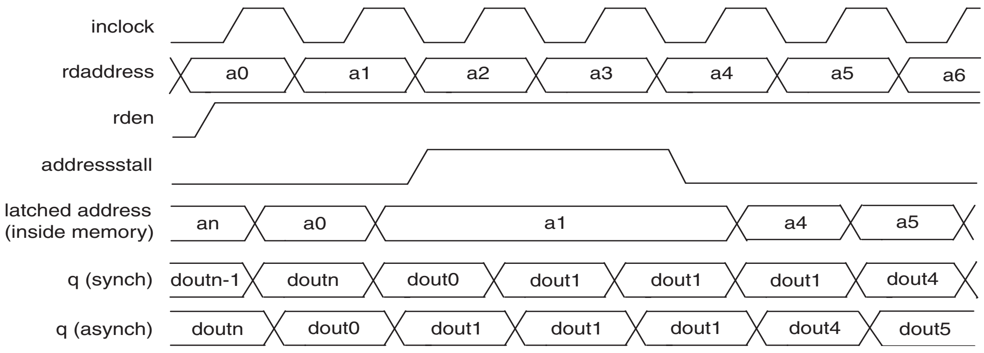
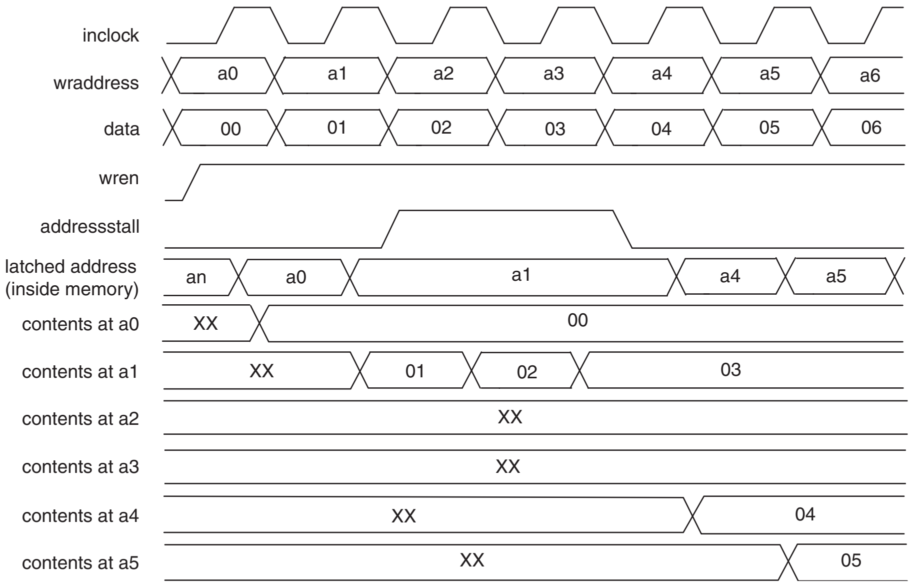


Figure 2–4. Stratix II Address Clock Enable During Write Cycle Waveform



Memory Modes

Stratix II TriMatrix memory blocks include input registers that synchronize writes, and output registers to pipeline data to improve system performance. All TriMatrix memory blocks are fully synchronous, meaning that all inputs are registered, but outputs can be either registered or unregistered.

TriMatrix memory does not support asynchronous memory (unregistered outputs).

Depending on which TriMatrix memory block you use, the memory has various modes, including:

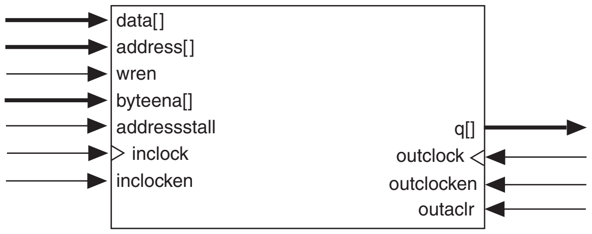
- Single-port
- Simple dual-port
- True dual-port (bidirectional dual-port)
- Shift-register
- ROM
- FIFO

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

Single-Port Mode

All TriMatrix memory blocks support the single-port mode that supports non-simultaneous read and write operations. Figure 2–5 shows the single-port memory configuration for TriMatrix memory.

Figure 2–5. Single-Port Memory *Note (1)*



Note to Figure 2–5:

(1) Two single-port memory blocks can be implemented in a single M4K or M-RAM block.

M4K and M-RAM memory blocks can also be halved 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 or M-RAM block, first ensure that each of the two independent RAM blocks is equal to or less than half the size of the M4K or M-RAM block. Secondly, assign both single-port RAMs to the same M4K or M-RAM block.

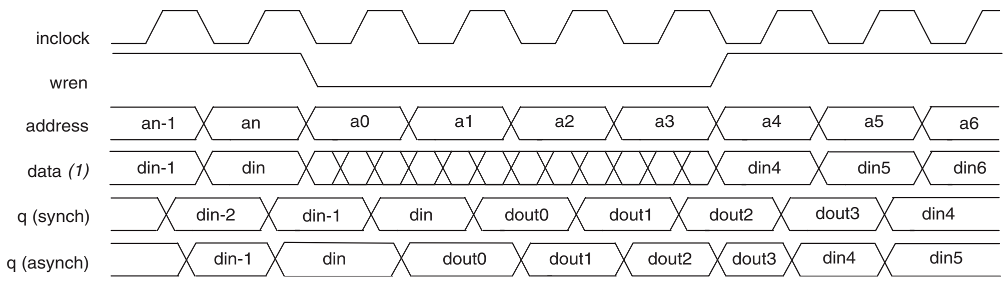
In single-port RAM configuration, the outputs can only be in read-during-write mode, which means that during the write operation, data written to the RAM flows through to the RAM outputs. When the output registers are bypassed, the new data is available on the rising edge of the same clock cycle on which it was written. See the "Read-During-Write Operation at the Same Address"section for more information about read-during-write mode. Table 2–7 shows the port width configurations for TriMatrix blocks in single-port mode.

Table 2–7. Stratix II Port Width Configurations for M512, M4K, and M-RAM Blocks (Single-Port Mode)

	M512 Blocks	M4K Blocks	M-RAM Blocks
Port Width 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

Figure 2–6 shows timing waveforms for read and write operations in single-port mode.

Figure 2–6. Stratix II Single-Port Timing Waveforms



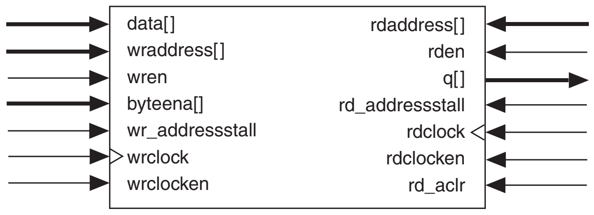
Note to Figure 2–6:

(1) The crosses in the `data` waveform during read mean "don't care."

Simple Dual-Port Mode

All TriMatrix memory blocks support simple dual-port mode which supports a simultaneous read and write operation. Figure 2–7 shows the simple dual-port memory configuration for TriMatrix memory.

Figure 2–7. Stratix II Simple Dual-Port Memory Note (1)



Note to Figure 2–7:

(1) Simple dual-port RAM supports input/output clock mode in addition to the read/write clock mode shown.

TriMatrix memory supports mixed-width configurations, allowing different read and write port widths. Tables 2–8 through 2–10 show the mixed width configurations for the M512, M4K, and M-RAM blocks, respectively.

Table 2–8. Stratix II 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	✓	✓	✓	✓	✓		
256 × 2	✓	✓	✓	✓	✓		
128 × 4	✓	✓	✓	✓	✓		
64 × 8	✓	✓	✓	✓	✓		
32 × 16	✓	✓	✓	✓	✓		
64 × 9						✓	✓
32 × 18						✓	✓

Table 2–9. Stratix II M4K Block Mixed-Width Configurations (Simple Dual-Port Mode)

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

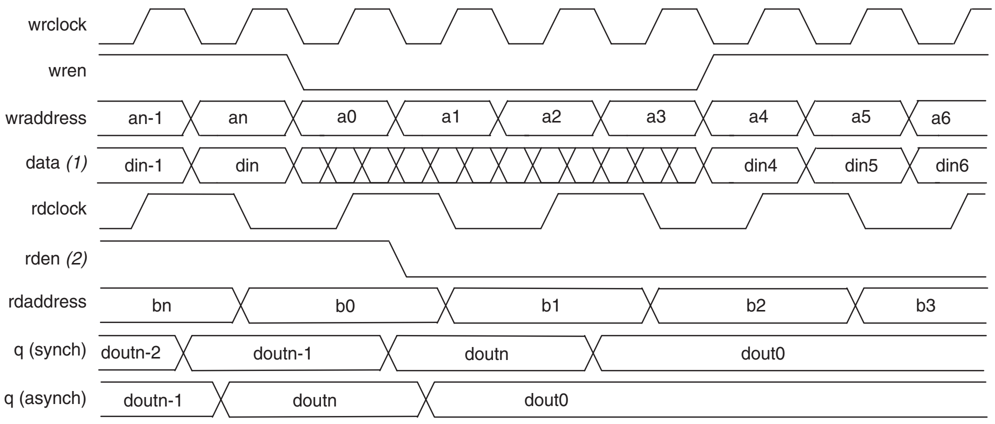
Table 2–10. Stratix II M-RAM Block Mixed-Width Configurations (Simple Dual-Port Mode)

Read Port	Write Port				
	64K × 9	32K × 18	18K × 36	8K × 72	4K × 144
64K × 9	✓	✓	✓	✓	
32K × 18	✓	✓	✓	✓	
18K × 36	✓	✓	✓	✓	
8K × 72	✓	✓	✓	✓	
4K × 144					✓

In simple dual-port mode, M512 and M4K blocks have one write enable and one read enable signal. However, M-RAM blocks contain only a write enable signal that controls the read and write operation. There is no separate read enable signal. The write enable is held high to perform a write operation. M-RAM blocks are always enabled for read operation. If the read address and the write address select the same address location during a write operation, M-RAM block output is unknown.

TriMatrix memory blocks do not support a clear port on the write enable and read enable registers. 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. See the "Read-During-Write Operation at the Same Address" section for more information. Figure 2–8 shows timing waveforms for read and write operations in simple dual-port mode.

Figure 2–8. Stratix II Simple Dual-Port Timing Waveforms



Notes to Figure 2–8:

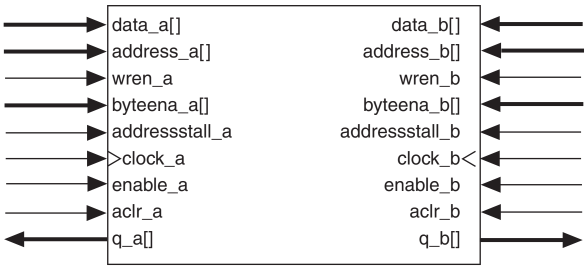
(1) The crosses in the `data` waveform during read mean "don't care."

(2) The read enable `rden` signal is not available in M-RAM blocks. The M-RAM block in simple dual-port mode always reads out the data stored at the current read address location.

True Dual-Port Mode

Stratix II M4K and M-RAM memory blocks support the true dual-port mode. True dual-port mode supports any combination of two-port operations: two reads, two writes, or one read and one write at two different clock frequencies. Figure 2–9 shows Stratix II true dual-port memory configuration.

Figure 2–9. Stratix II True Dual-Port Memory Note (1)



Note to Figure 2–9:

(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 as follows:

- 256 × 16-bit (×18-bit with parity) (M4K)
- 8K × 64-bit (×72-bit with parity) (M-RAM)

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. Table 2–11 lists the possible M4K block mixed-port width configurations.

Table 2–11. Stratix II M4K Block Mixed-Port Width Configurations (True Dual-Port)

Read Port	Write Port						
	4K × 1	2K × 2	1K × 4	512 × 8	256 × 16	512 × 9	256 × 18
4K × 1	✓	✓	✓	✓	✓		
2K × 2	✓	✓	✓	✓	✓		
1K × 4	✓	✓	✓	✓	✓		
512 × 8	✓	✓	✓	✓	✓		
256 × 16	✓	✓	✓	✓	✓		
512 × 9						✓	✓
256 × 18						✓	✓

Table 2–12 lists the possible M-RAM block mixed-port width configurations.

Table 2–12. Stratix II M-RAM Block Mixed-Port Width Configurations (True Dual-Port)

Read Port	Write Port			
	64K × 9	32K × 18	18K × 36	8K × 72
64K × 9	✓	✓	✓	✓
32K × 18	✓	✓	✓	✓
18K × 36	✓	✓	✓	✓
8K × 72	✓	✓	✓	✓

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 on which it was written. See the "Read-During-Write Operation at the Same Address" section for waveforms and information on mixed-port read-during-write mode.

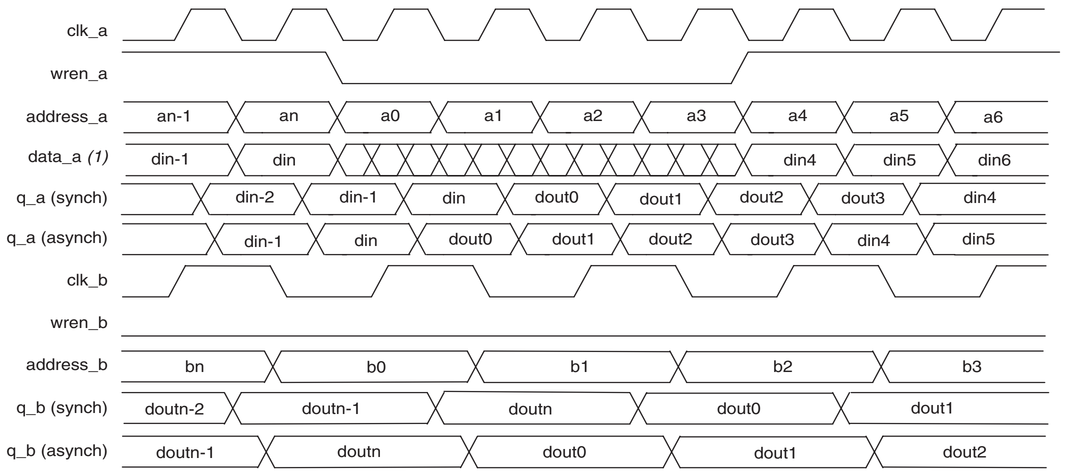
Potential write contentions must be resolved external to the RAM because writing to the same address location at both ports results in unknown data storage at that location. For a valid write operation to the same address of the M-RAM block, the rising edge of the write clock for port A must occur following 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 will be invalid.

See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for the maximum synchronous write cycle time.

Figure 2–10 shows true dual-port timing waveforms for the write operation at port A and the read operation at port B.

Figure 2–10. Stratix II True Dual-Port Timing Waveforms



Note to Figure 2–10:

(1) The crosses in the `data_a` waveform during write mean "don't care."

Shift-Register Mode

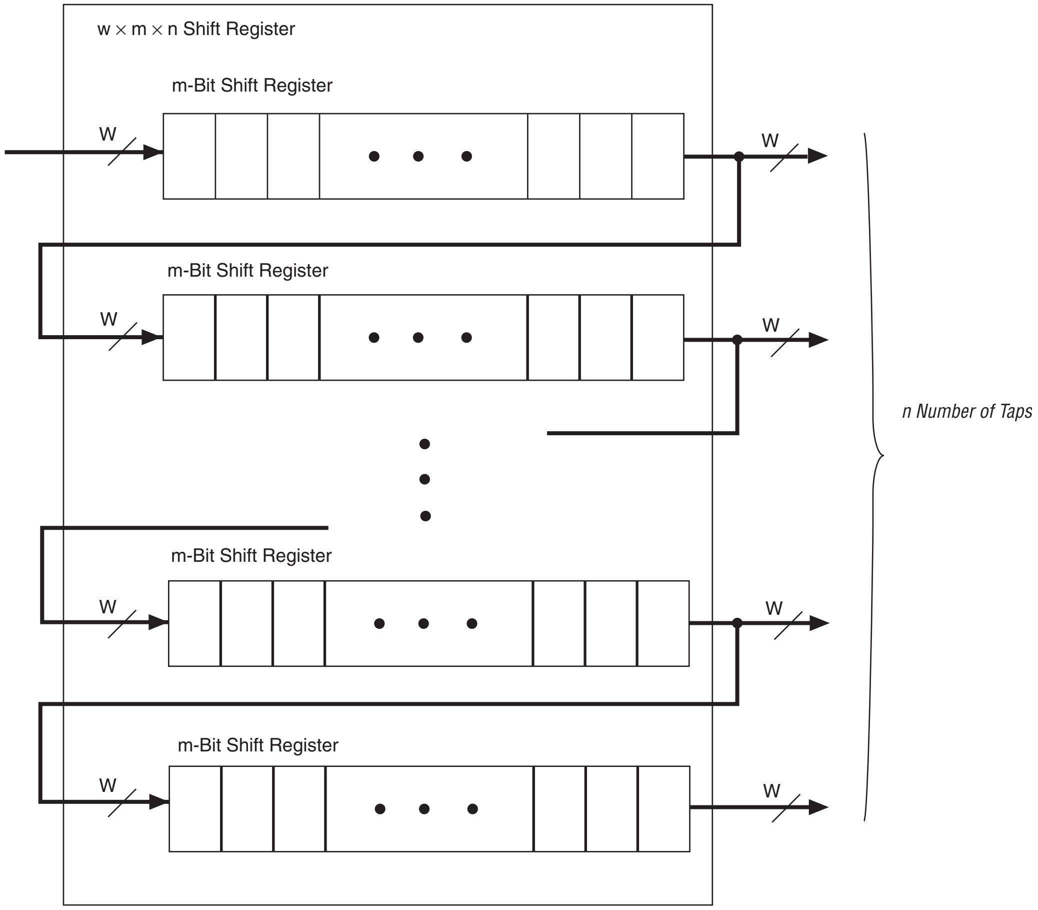
M512 and M4K memory block support the shift register mode.

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 quickly exhaust 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.

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), and 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.

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 2–11 shows the TriMatrix memory block in the shift-register mode.

Figure 2–11. Stratix II Shift-Register Memory Configuration



ROM Mode

M512 and M4K memory blocks support ROM mode. A memory initialization file (**.mif**) initializes the ROM contents of these blocks. The address lines of the ROM are registered. The outputs can be registered or unregistered. The ROM read operation is identical to the read operation in the single-port RAM configuration.

FIFO Buffers Mode

TriMatrix memory blocks support the FIFO mode. M512 memory blocks are ideal for designs with many shallow FIFO buffers. All memory configurations have synchronous inputs; however, the FIFO buffer outputs are always combinational. Simultaneous read and write from an empty FIFO buffer is not supported.

See the *Single- & Dual-Clock FIFO Megafunctions User Guide* and *FIFO Partitioner Function User Guide* for more information on FIFO buffers.

Clock Modes

Depending on which TriMatrix memory mode is selected, the following clock modes are available:

- Independent
- Input/output
- Read/write
- Single-clock

Table 2–13 shows these clock modes supported by all TriMatrix blocks when configured as respective memory modes.

Table 2–13. Stratix II TriMatrix Memory Clock Modes

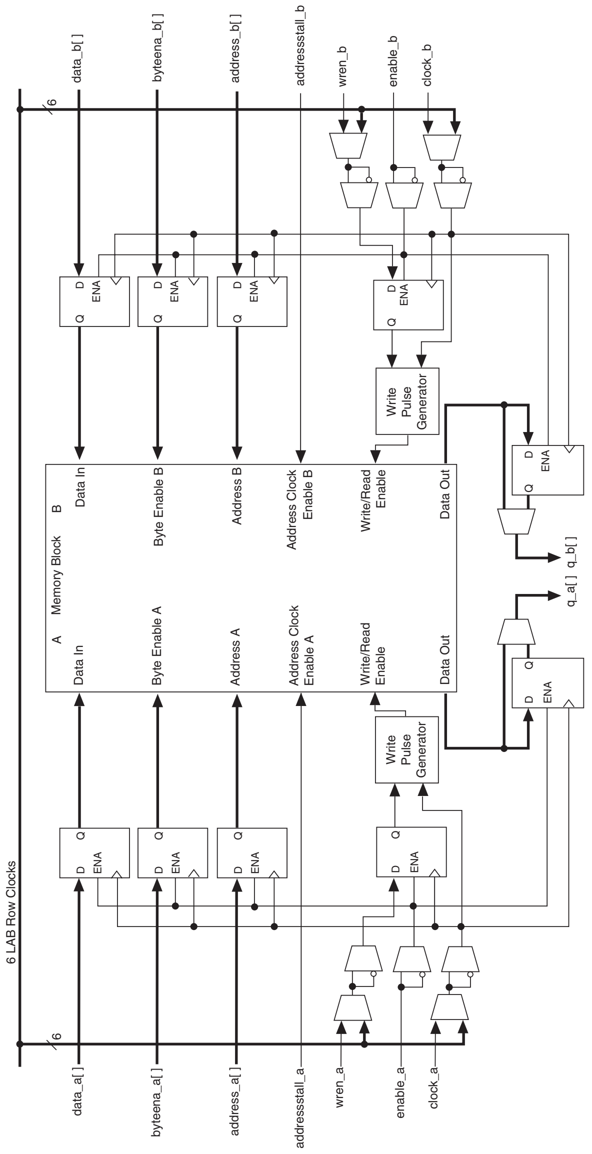
Clocking Modes	True Dual-Port Mode	Simple Dual-Port Mode	Single-Port Mode
Independent	✓		
Input/output	✓	✓	✓
Read/write		✓	
Single clock	✓	✓	✓

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 for port A and B registers. Asynchronous clear signals for the registers, however, are supported.

Figure 2–12 shows a TriMatrix memory block in independent clock mode.

Figure 2–12. Stratix II TriMatrix Memory Block in Independent Clock Mode *Note (1)*



Note to Figure 2–12:

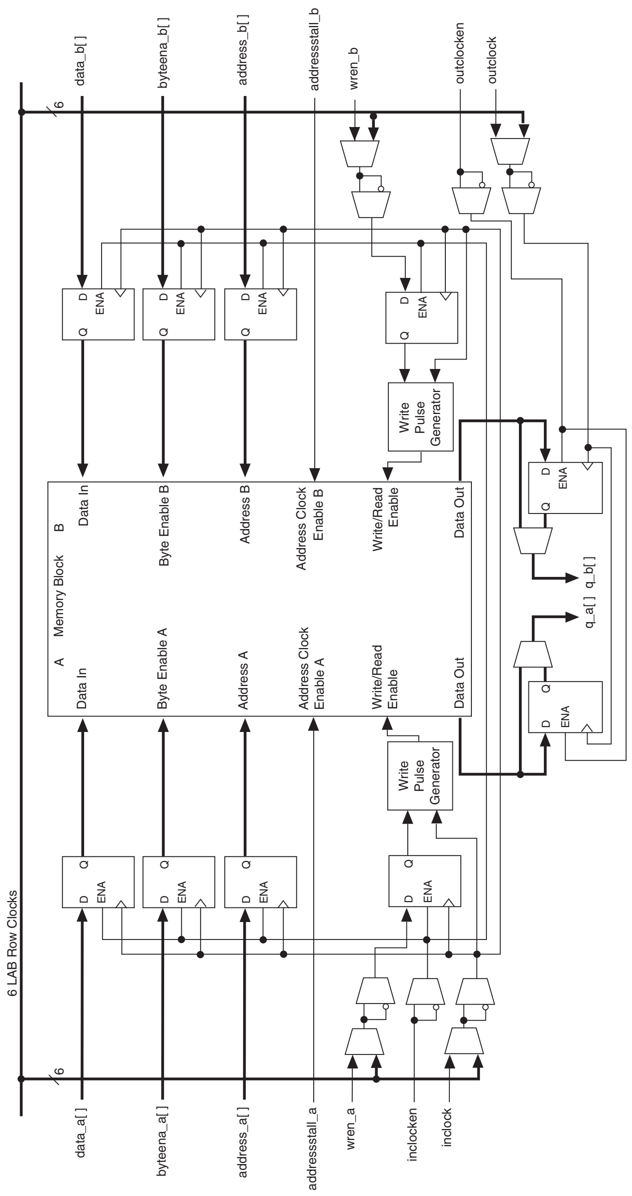
(1) Violating the setup or hold time on the memory block address registers could corrupt the memory contents. This applies to both read and write operations.

Input/Output Clock Mode

Stratix II 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 the following inputs into the memory block: data input, write enable, and address. The other clock controls the blocks' data output registers. Each memory block port also supports independent clock enables for input and output registers. Asynchronous clear signals for the registers, however, are not supported.

Figures 2–13 through 2–15 show the memory block in input/output clock mode for true dual-port, simple dual-port, and single-port modes, respectively.

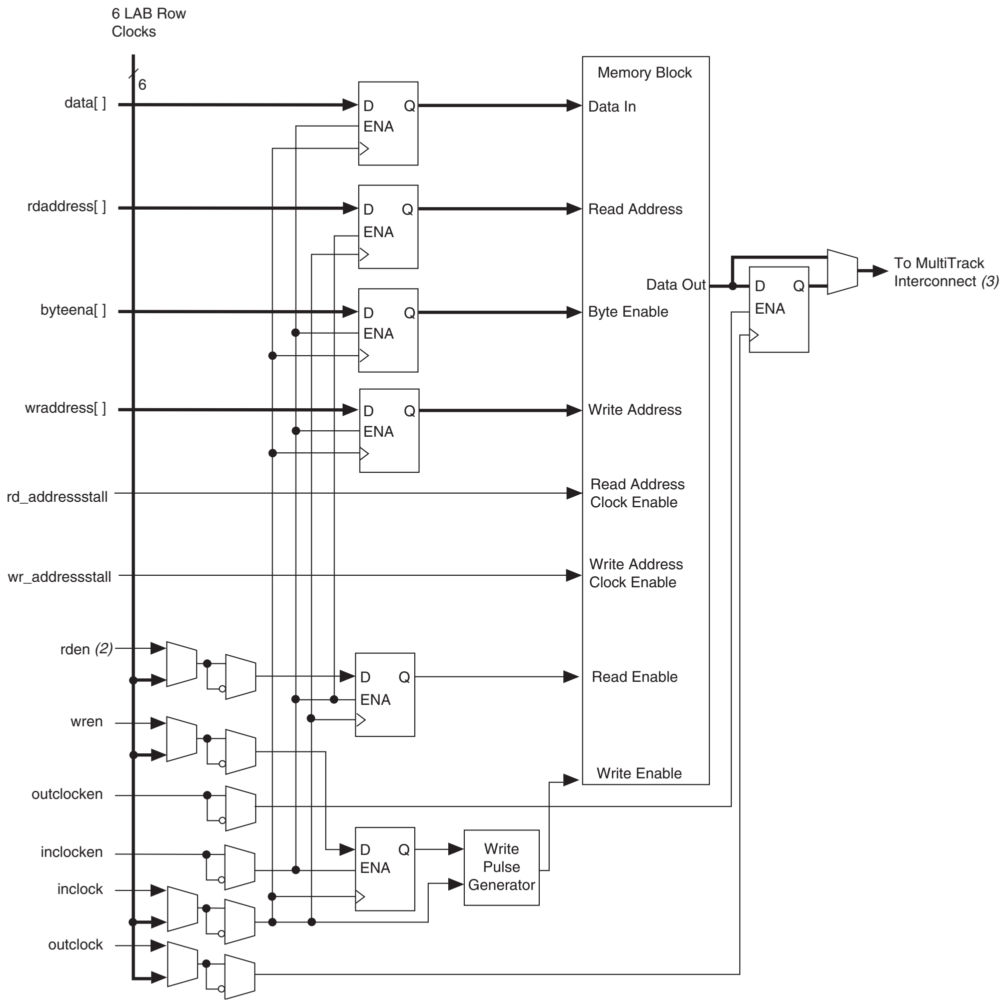
Figure 2–13. Stratix II Input/Output Clock Mode in True Dual-Port Mode *Note (1)*



Note to Figure 2–13:

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

Figure 2–14. Stratix II Input/Output Clock Mode in Simple Dual-Port Mode *Note (1)*



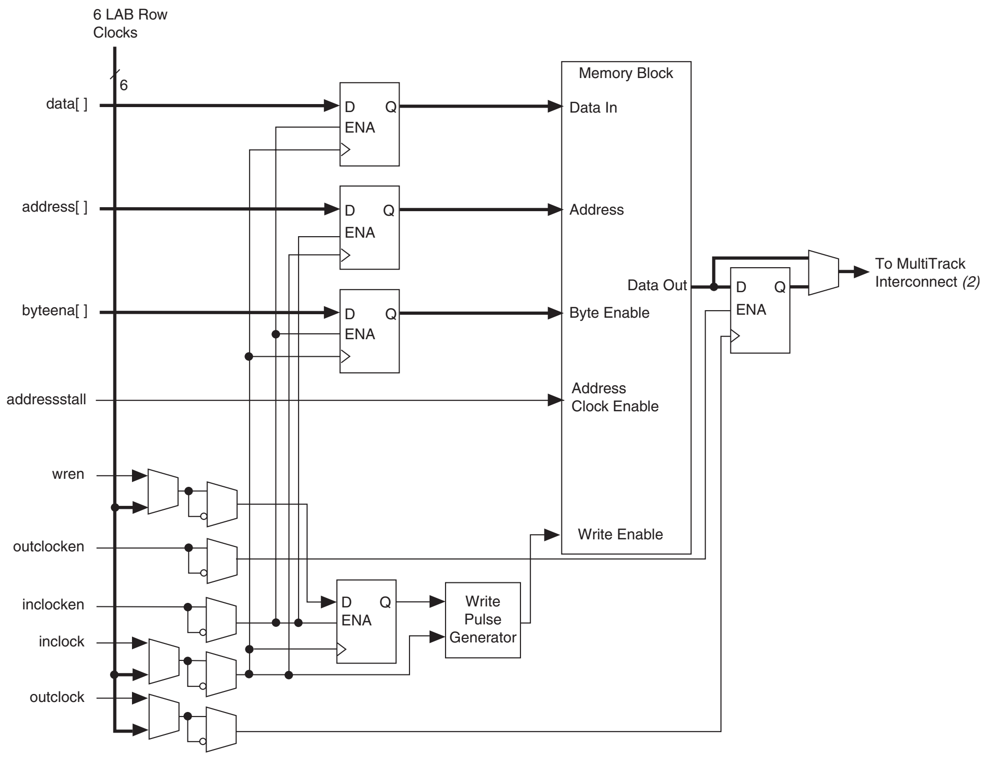
Notes to Figure 2–14:

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

(2) The read enable `rden` signal is not available in the M-RAM block. An M-RAM block in simple dual-port mode is always reading out the data stored at the current read address location.

(3) See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for more information on the MultiTrack™ interconnect.

Figure 2–15. Stratix II Input/Output Clock Mode in Single-Port Mode *Note (1)*



Note to Figure 2–15:

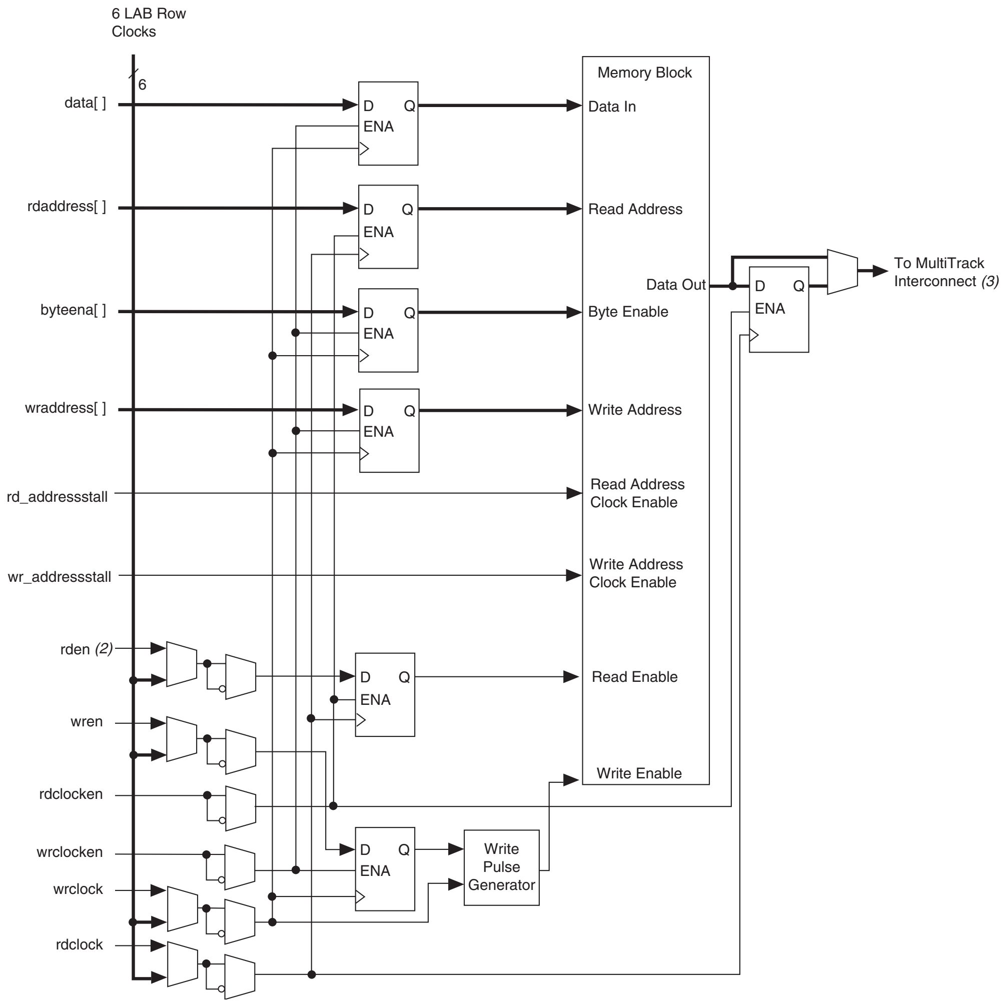
(1) Violating the setup or hold time on the memory block address registers could corrupt the memory contents. This applies to both read and write operations.

(2) See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for more information on the MultiTrack interconnect.

Read/Write Clock Mode

Stratix II TriMatrix memory blocks can implement read/write clock mode for simple dual-port memory. This mode uses up to two clocks. The write clock controls the blocks' data inputs, write address, and write enable signals. The read clock controls the data output, read address, and read enable signals. The memory blocks support independent clock enables for each clock for the read- and write-side registers. Asynchronous clear signals for the registers, however, are not supported. Figure 2–16 shows a memory block in read/write clock mode.

Figure 2–16. Stratix II Read/Write Clock Mode *Note (1)*



Notes to Figure 2–16:

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

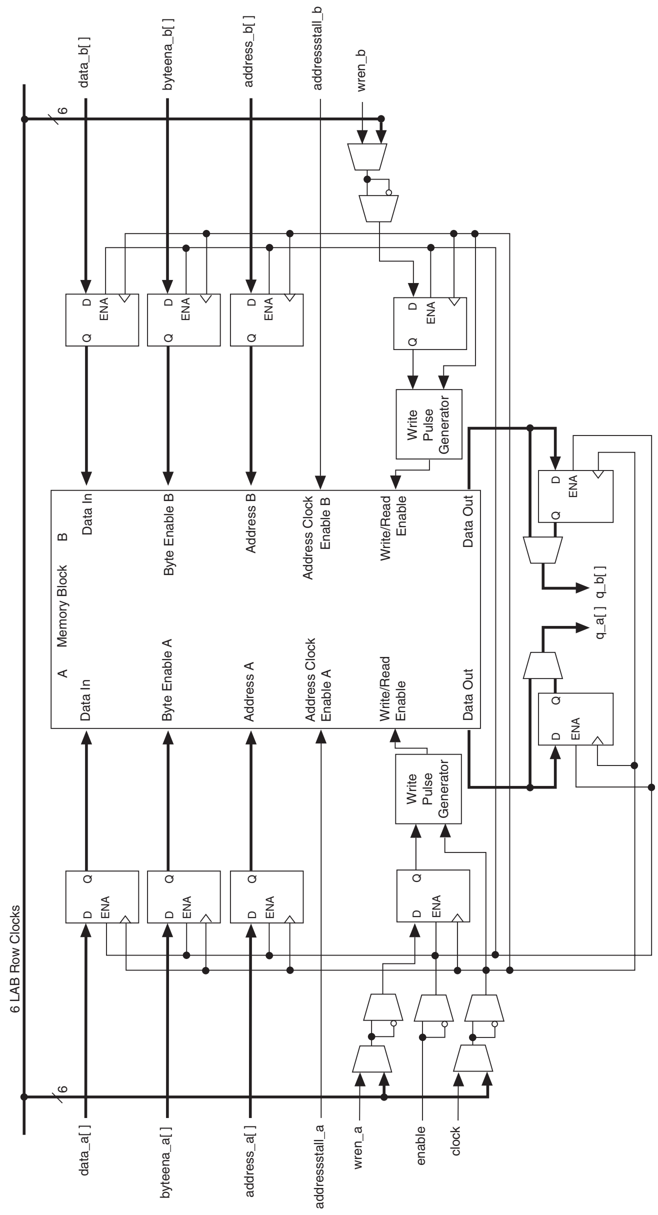
(2) The read enable `rden` signal is not available in the M-RAM block. An M-RAM block in simple dual-port mode is always reading the data stored at the current read address location.

(3) See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for more information on the MultiTrack interconnect.

Single-Clock Mode

Stratix II TriMatrix memory blocks implement single-clock mode for true dual-port, simple dual-port, and single-port memory. In this mode, a single clock, together with clock enable, is used to control all registers of the memory block. Asynchronous clear signals for the registers, however, are not supported. Figures 2–17 through 2–19 show the memory block in single-clock mode for true dual-port, simple dual-port, and single-port modes, respectively.

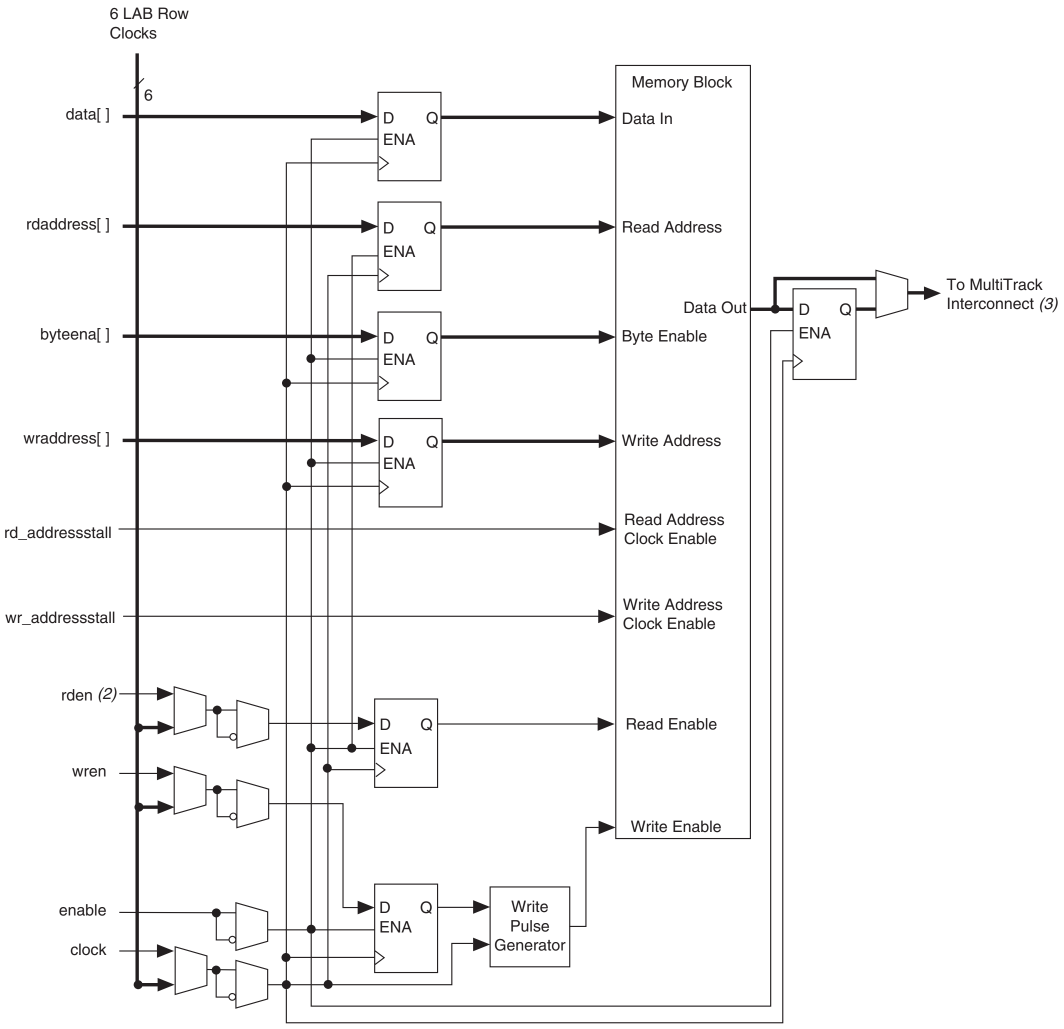
Figure 2–17. Stratix II Single-Clock Mode in True Dual-Port Mode *Note (1)*



Note to Figure 2–17:

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

Figure 2–18. Stratix II Single-Clock Mode in Simple Dual-Port Mode *Note (1)*



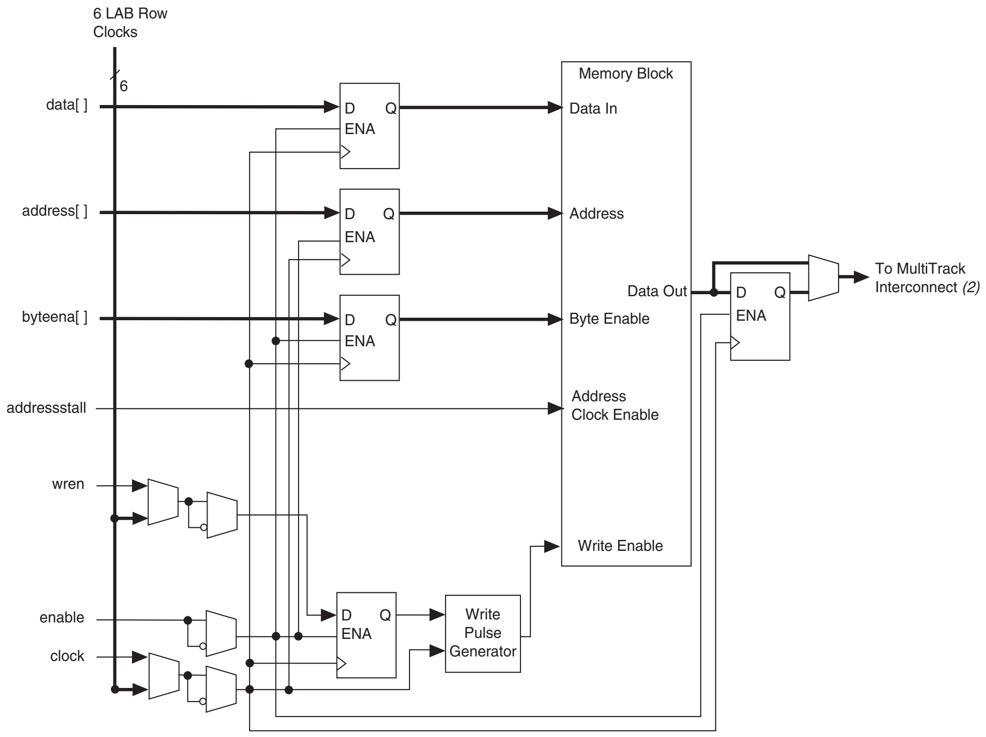
Notes to Figure 2–18:

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

(2) The read enable `rden` signal is not available in the M-RAM block. An M-RAM block in simple dual-port mode is always reading the data stored at the current read address location.

(3) See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for more information on the MultiTrack interconnect.

Figure 2–19. Stratix II Single-Clock Mode in Single-Port Mode *Note (1)*



Note to Figure 2–19:

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

(2) See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for more information on the MultiTrack interconnect.

Designing With TriMatrix Memory

When instantiating TriMatrix memory, it is important to understand the features that set it apart from other memory architectures. The following sections describe the unique attributes and functionality of TriMatrix memory.

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.

See *Application Note 207: TriMatrix Memory Selection Using the Quartus II Software* for more information on selecting the appropriate memory block.

Synchronous & Pseudo-Asynchronous Modes

The TriMatrix memory architecture implements synchronous RAM by registering the input and output signals to the RAM block. The inputs to all TriMatrix memory blocks are registered providing synchronous write cycles, while the output registers can be bypassed. In a synchronous operation, RAM generates its own self-timed strobe write enable signal derived from the global or regional clock. In contrast, a circuit using asynchronous RAM must generate the RAM write enable signal while ensuring that its data and address signals meet setup and hold time specifications relative to the write enable signal. During a synchronous operation, the RAM is used in pipelined mode (inputs and outputs registered) or flow-through mode (only inputs registered). However, in an asynchronous memory, neither the input nor the output is registered.

While Stratix II devices do not support asynchronous memory, they do support a pseudo-asynchronous read where the output data is available during the clock cycle when the read address is driven into it. Pseudo-asynchronous 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.

See *AN 210: Converting Memory from Asynchronous to Synchronous for Stratix & Stratix GX Designs* for more information.

Power-up Conditions & Memory Initialization

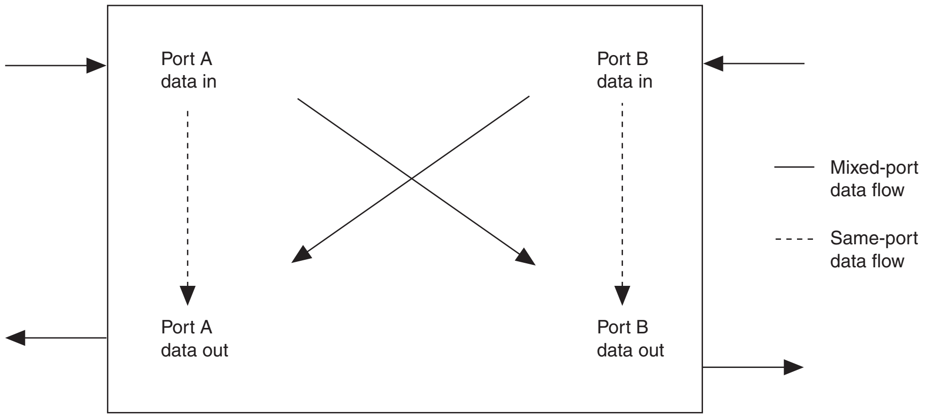
Upon power up, TriMatrix memory is in an idle state. The M512 and M4K block outputs always power-up to zero, regardless of whether the output registers are used or bypassed. Even if an MIF is used to pre-load the contents of the RAM block, the outputs will still power-up as cleared. For example, if address 0 is pre-initialized to FF, the M512 and M4K blocks power up with the output at 00.

M-RAM blocks do not support MIFs; therefore, they cannot be pre-loaded with data upon power up. M-RAM blocks asynchronous outputs and memory controls always power up to an unknown state. If M-RAM block outputs are registered, the registers power up as cleared. When a read is performed immediately after power up, the output from the read operation will be undefined since the M-RAM contents are not initialized. The read operation will continue to be undefined for a given address until a write operation is performed for that address.

Read-During-Write Operation at the Same Address

The "Same-Port Read-During-Write Mode" and "Mixed-Port Read-During-Write Mode" 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 read-during-write data flows: same-port and mixed-port. Figure 2–20 shows the difference between these flows.

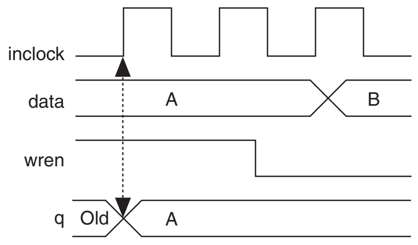
Figure 2–20. Stratix II Read-During-Write Data Flow



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 on which it was written. This behavior is valid on all memory block sizes. Figure 2–21 shows 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 2–7). The non-masked bytes are read out as shown in Figure 2–21.

Figure 2–21. Stratix II Same-Port Read-during-Write Functionality Note (1)



Note to Figure 2–21:

(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. Figures 2–22 and 2–23 show sample functional waveforms where both ports have the same address. These figures assume that the outputs are not registered.

The `DONT_CARE` setting allows memory implementation in any TriMatrix memory block, whereas 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.

The RAM outputs are unknown for a mixed-port read-during-write operation of the same address location of an M-RAM block, as shown in Figure 2–23.

Figure 2–22. Stratix II Mixed-Port Read-during-Write: OLD_DATA

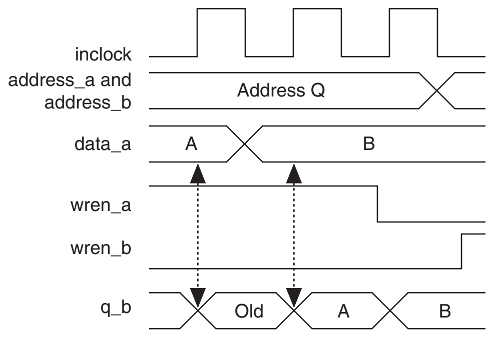
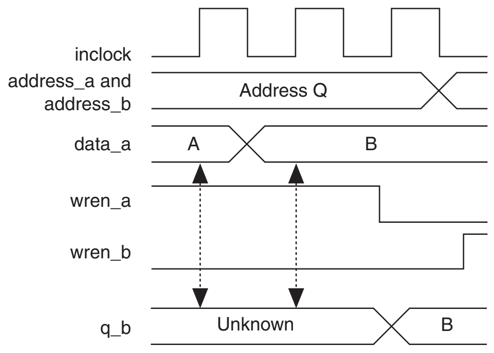


Figure 2–23. Stratix II Mixed-Port Read-during-Write: DONT_CARE



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

The TriMatrix memory structure of Stratix II devices provides an enhanced RAM architecture with high memory bandwidth. It addresses the needs of different memory applications in FPGA designs with features such as different memory block sizes and modes, byte enables, parity bit storage, address clock enables, mixed clock mode, shift register mode, mixed-port width support, and true dual-port mode.

3. External Memory Interfaces

SII52003-2.2

Introduction

Stratix® II devices support a broad range of external memory interfaces such as double data rate (DDR) SDRAM, DDR2 SDRAM, RLDRAM II, QDRII SRAM, and single data rate (SDR) SDRAM. Its dedicated phase-shift circuitry allows the Stratix II device to interface with an external memory at twice the system clock speed (up to 300 MHz/600 Megabits per second (Mbps) with RLDRAM II). In addition to external memory interfaces, you can also use the dedicated phase-shift circuitry for other applications that require a shifted input signal.

Typical I/O architectures transmit a single data word on each positive clock edge and are limited to the associated clock speed. To achieve a 400-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 new memory architectures use a DDR I/O interface. Although Stratix II devices also support the mature and well established SDR external memory, this chapter focuses on DDR memory standards. These DDR memory standards cover a broad range of applications for embedded processor systems, image processing, storage, communications, and networking.

Table 3–1 summarizes the maximum clock rate the Stratix II device can support with external memory devices.

Table 3–1. Stratix II Maximum Clock Rate Support for External Memory Interfaces (Part 1 of 2) *Notes (1), (2)*

Memory Standards	–3 Speed Grade (MHz)	–4 Speed Grade (MHz)	–5 Speed Grade (MHz)
DDR SDRAM	200	200	200
DDR2 SDRAM *(3)*	267	267	233
RLDRAM II *(3)*	300	250 *(4)*	200

Table 3–1. Stratix II Maximum Clock Rate Support for External Memory Interfaces (Part 2 of 2) *Notes (1), (2)*

Memory Standards	–3 Speed Grade (MHz)	–4 Speed Grade (MHz)	–5 Speed Grade (MHz)
QDRII SRAM	250	200	200

Notes for Table 3–1:
(1) Numbers are preliminary until characterization is final.
(2) Assumes that the dedicated circuitry is used in the interface.
(3) This applies for both interfaces with modules and components. When you are not using the dedicated circuits, Stratix II supports up to 150MHz interface with this memory.
(4) You can under-clock a 300-MHz RLDRAM II device to achieve this clock rate.

This chapter describes the hardware features in Stratix II devices that facilitate the high-speed memory interfacing for each DDR memory standard. It then lists the Stratix II feature enhancements from Stratix devices and briefly explains how each memory standard uses the Stratix II features.

You can use this document with *AN 325: Interfacing RLDRAM II with Stratix II, Stratix, and Stratix GX Devices*, *AN 326: Interfacing QDRII SRAM with Stratix II, Stratix, and Stratix GX Devices*, *AN 327: Interfacing DDR SDRAM with Stratix II Devices*, and *AN 328: Interfacing DDR2 SDRAM with Stratix II Devices*.

External Memory Standards

The following sections describe how to use the Stratix II external memory interfacing features.

For details on each memory standard, refer to the appropriate Stratix II application notes at **www.altera.com**.

DDR & DDR2 SDRAM

DDR SDRAM is a memory architecture that transmits and receives data at twice the clock speed. These devices transfer data on both the rising and falling edge of the clock signal. DDR2 SDRAM is a second generation memory based on the DDR SDRAM architecture and transfers data to Stratix II devices at up to 267 MHz/533 Mbps. Stratix II devices can support DDR SDRAM at up to 200 MHz/400 Mbps.

Interface Pins

DDR and DDR2 SDRAM devices use interface pins such as data (DQ), data strobe (DQS), clock, command, and address pins. Data is sent and captured at twice the system clock rate by transferring data on the clock's positive and negative edge. The commands and addresses still only use one active (positive) edge of a clock. DDR2 and DDR SDRAM use single-ended data strobes (DQS). DDR2 SDRAM can also use optional differential data strobes (DQS and DQS#). However, Stratix II devices do not use the optional differential data strobes for DDR2 SDRAM interfaces since DQS and DQSn pins in Stratix II devices are not differential. You can leave the DDR SDRAM memory DQS# pin unconnected. Only the shifted DQS signal from the DQS logic block is used to capture data.

DDR and DDR2 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/×18 mode in Stratix II devices (see "Data & Data Strobe Pins" on page 3–11). To support a ×16 DDR SDRAM device, you need to configure the Stratix II device to use two sets of DQ pins in ×8/×9 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 II devices to use four sets of DQS/DQ groups in ×8/×9 mode.

Connect the memory device's DQ and DQS pins to the Stratix II DQ and DQS pins, respectively, as listed in the Stratix II pin tables. DDR and DDR2 SDRAM also uses active-high data mask, DM, pins for writes. You can connect the memory's DM pins to any of the Stratix II I/O pins in the same bank as the DQ pins of the FPGA. There is one DM pin per DQS/DQ group in a DDR or DDR2 SDRAM device.

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, and can only support DDR SDRAM at speeds up to 150 MHz.

For more information, see *AN 327: Interfacing DDR SDRAM with Stratix II Devices* and *AN 328 Interfacing DDR2 SDRAM with Stratix II Devices*.

You can also use I/O banks 1, 2, 5, or 6 to interface with DDR2 SDRAM devices not using the dedicated circuitry. However, you need to simulate your system to ensure performance because these I/O banks only support SSTL-18 Class I outputs.

If the DDR or DDR2 SDRAM device supports error correction coding (ECC), the design will use an extra DQS/DQ group for the ECC pins.

You can use any of the user I/O pins for commands and addresses to the DDR SDRAM. Due to the symmetrical setup and hold time for the command and address pins at the memory, you may need to generate these signals from the system clock's negative edge.

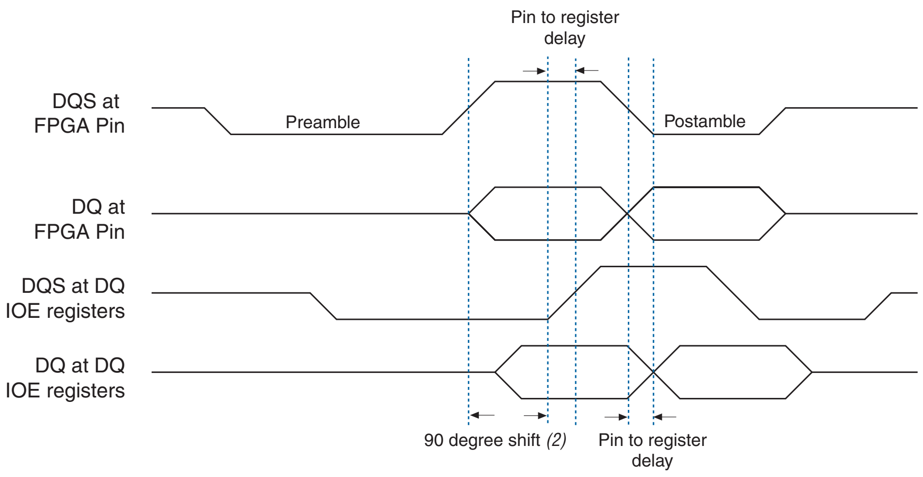
The clocks to the SDRAM device are called CK and CK# pins. Use any of the user I/O pins via the DDR registers to generate the CK and CK# signals to meet the DDR SDRAM or DDR2 SDRAM device's t_{DQSS} requirement. The memory device's t_{DQSS} requires that the write DQS signal's positive edge must be within 25% of the positive edge of the DDR SDRAM or DDR2 SDRAM clock input. Using regular I/O pins for CK and CK# also ensures that any PVT variations on the DQS signals are tracked the same way by these CK and CK# pins.

☞ For better performance and matched timing, it is best to use I/O banks 3, 4, 7, and 8 for `CK`, `CK#` address and command signals.

Read & Write Operations

When reading from the memory, DDR and DDR2 SDRAM devices send the data edge-aligned with respect to the data strobe. To properly read the data in, the data strobe needs to be center-aligned with respect to the data inside the FPGA. Stratix II devices feature dedicated circuitry to shift this data strobe to the middle of the data window. Figure 3–1 shows an example of how the memory sends out the data and data strobe for a burst-of-two operation.

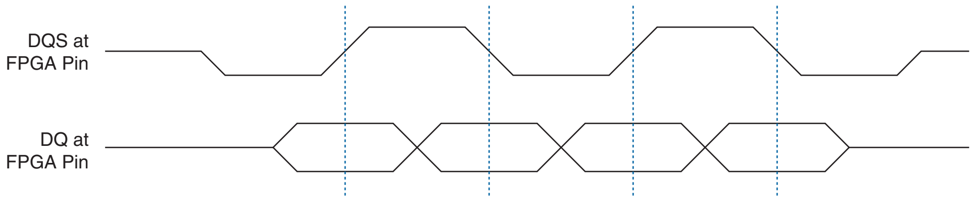
Figure 3–1. Example of a 90° Shift on the DQS Signal *Note (1)*



Notes to Figure 3–1:
(1) DDR2 SDRAM does not support burst length of two.
(2) The phase-shift needed is not necessarily 90°.

During write operations to a DDR or DDR2 SDRAM device, the FPGA needs to send the data to the memory center-aligned with respect to the data strobe. Stratix II devices use a PLL to center-align the data by generating a 0° phase-shifted system clock for the write data strobes and a –90° phase-shifted write clock for the write data pins for DDR and DDR2 SDRAM. Figure 3–2 shows an example of the relationship between the data and data strobe during a burst-of-four write.

Figure 3–2. DQ & DQS Relationship During a DDR & DDR2 SDRAM Write



Note to Figure 3–2:
(1) This example shows a write for a burst length of four. DDR SDRAM also supports burst lengths of two.

For more information on DDR SDRAM and DDR2 SDRAM specifications, refer to JEDEC standard publications JESD79C and JESD79-2, respectively, from **www.jedec.org**, or see *AN 327: Interfacing DDR SDRAM with Stratix II Devices* and *AN 328: Interfacing DDR2 SDRAM with Stratix II 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 a RLDRAM II CIO device are bidirectional while the data pins in a RLDRAM II SIO device are unidirectional. Instead of bidirectional data strobes, RLDRAM II uses differential free-running read and write clocks to accompany the data. As in DDR or DDR2 SDRAM, data is sent and captured at twice the system clock rate by transferring data on the clock's positive and negative edge. The commands and addresses still only use one active (positive) edge of a clock.

If the data pins are bidirectional, connect them to the Stratix II DQ pins. If the data pins are unidirectional, connect the RLDRAM II device Q ports to the Stratix II device DQ pins and connect the D ports to any user I/O pins in I/O banks 3, 4, 7, or 8 for better performance. RLDRAM II also uses active-high data mask, DM, 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 RLDRAM II CIO devices. When interfacing with RLDRAM II 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 RLDRAM II device.

Connect the read clock pins (QK) to Stratix II 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 II devices requires that you ground the DQS pin output enables. The Stratix II device uses the shifted QK signal from the DQS logic block captures data. You can leave the QK# signal of the RLDRAM II device unconnected, as DQS and DQSn in Stratix II are not differential.

RLDRAM II devices have both input clocks (CK and CK#) and write clocks (DK and DK#). Use an external clock buffer to generate CK, CK#, DK, and DK# to meet the CK, CK#, DK, and DK# skew and skew rate requirements from the RLDRAM II device.

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 II 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/QK#. When writing to the RLDRAM II device, data must be center-aligned with the write clock (DK/DK#). The RLDRAM II interface uses the same scheme as in DDR or DDR2 SDRAM interfaces, where 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 and DDR2 SDRAM as shown in Figure 3–1 on page 3–5 and 3–2 on page 3–5.

For details on RLDRAM II, go to **www.rldram.com** or see *AN 325: Interfacing RLDRAM II with Stratix II Devices*.

QDRII SRAM

QDRII SRAM is a second generation of QDR SRAM devices. QDRII SRAM devices, which can transfer four words per clock cycle, fulfill the requirements facing next-generation communications system designers. QDRII SRAM devices provide concurrent reads and writes, zero latency, and increased data throughput, allowing simultaneous access to the same address location.

Interface Pins

QDRII SRAM uses two separate, unidirectional data ports for read and write operations, enabling QDR data transfer. The control signals are sampled on the rising edge of the clock. QDRII SRAM uses shared address lines for reads and writes. QDRII SRAM burst-of-two devices sample the read address on the rising edge of the clock and sample the write address on the falling edge of the clock while QDRII SRAM burst-of-four devices sample both read and write addresses on the clock's rising edge. Connect the memory device's Q ports (read data) to the Stratix II DQ pins. You can use any of the Stratix II device user I/O pins in I/O banks 3, 4, 7, or 8 for the D ports (write data), commands, and addresses.

QDRII SRAM uses the following clock signals:

- Input clocks K and Kn
- Output clocks C and Cn
- 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 II devices use single-clock mode for single-device QDRII SRAM interfacing where the K and Kn are used for both read and write operations, and the C and Cn clocks are unused. You should use both C or Cn and K or Kn clocks when interfacing with a bank of multiple QDRII SRAM devices with a single controller.

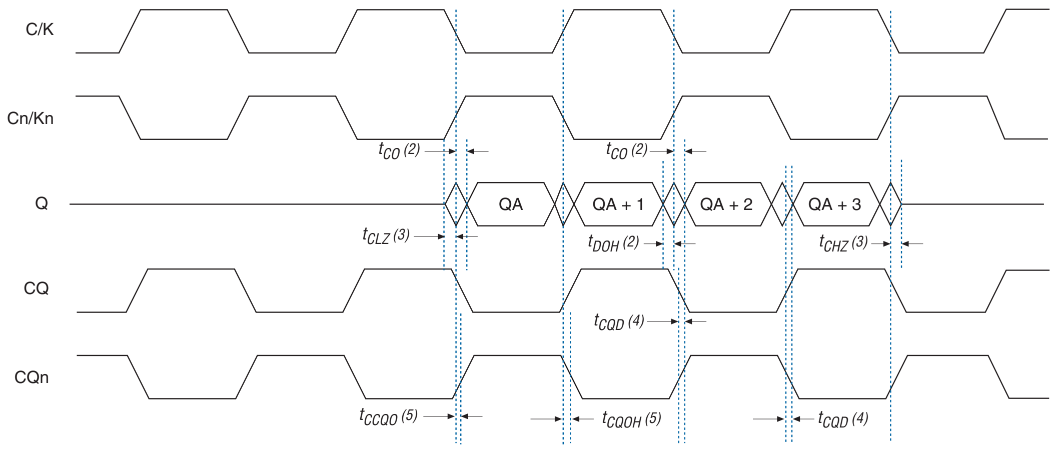
You can generate C, Cn, K, and Kn clocks using any of the I/O registers via the DDR registers. Due to strict skew requirements between K and Kn signals, use adjacent pins to generate the clock pair. Surround the pair with buffer pins tied to V_{CC} and ground for better noise immunity from other signals.

Connect CQ and CQn pins to the Stratix II DQS and DQSn pins. You must configure the DQS and DQSn as bidirectional pins. However, since CQ and CQn pins are output-only pins from the memory, the Stratix II device QDRII SRAM memory interface requires that you ground the DQS and DQSn output enable. To capture data presented by the memory, connect the shifted CQ signal to the input latch and connect the active-high input registers and the shifted CQn signal is connected to the active-low input register.

Read & Write Operations

Figure 3–3 shows the data and clock relationships in QDRII SRAM devices at the memory pins during reads. QDRII SRAM devices send data within a t_{CO} time after each rising edge of the read 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 read clock C or Cn in multi-clock mode or the input clock K or Kn in single clock mode. The CQ and CQn clocks are edge-aligned with the read data signal. These clocks accompany the read data for data capture in Stratix II devices.

Figure 3–3. Data & Clock Relationship During a QDRII SRAM Read *Note (1)*



Notes to Figure 3–3:

(1) The timing parameter nomenclature is based on the Cypress QDRII SRAM data sheet for CY7C1313V18.
(2) t_{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_{CCQO} and t_{CQOH} are skew measurements between the C or Cn clocks (or the K or Kn clocks 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.

For more information on QDRII SRAM, go to **www.qdrsram.com** or see *AN 326: Interfacing QDRII SRAM with Stratix II Devices*.

Stratix II DDR Memory Support Overview

Table 3–2 shows the I/O standard associated with the external memory interfaces.

Table 3–2. External Memory Support in Stratix II Devices

Memory Standard	I/O Standard
DDR SDRAM *(1)*	SSTL-2 Class II
DDR2 SDRAM *(2)*	SSTL-18 Class II*(3)*
RLDRAM II *(4)*	1.8-V HSTL *(3)*
QDRII SRAM *(4)*	1.8-V HSTL *(3)*

Notes to Table 3–2:

(1) DDR SDRAM is also supported in the Stratix II side I/O banks (I/O banks 1, 2, 5, and 6) up to 150 MHz without the dedicated phase-shift circuitry.

(2) Stratix II supports DDR2 SDRAM in I/O banks 1, 2, 5, and 6 with SSTL-18 Class I.

(3) These I/O standards are supported in I/O banks 3, 4, 7, and 8 on the top and bottom of the Stratix II device. I/O banks 1, 2, 5, and 6 only support the input operation of these I/O standards.

(4) For maximum performance, Altera® recommends using the 1.8-V HSTL I/O standard. RLDRAM II and QDRII SRAM devices also support the 1.5-V HSTL I/O standard.

Stratix II devices support the data strobe or read clock signal (DQS) used in DDR SDRAM, DDR2 SDRAM, RLDRAM II, and QDRII SRAM devices with dedicated circuitry. Stratix II devices also support the DQSn signal (the DQS complement signal) for external memory types that require them, for example QDRII SRAM. DQS and DQSn signals are usually associated with a group of data (DQ) pins. However, these are not differential buffers and cannot be used in DDR2 SDRAM or RLDRAM II interfaces.

You can also interface with these external memory devices without the use of dedicated circuitry at a lower performance than what is stated in Table 3–2.

Stratix II devices contain dedicated circuitry to shift the incoming DQS signals by 0°, 22.5°, 30°, 36°, 45°, 60°, 67.5°, 72°, 90°, 108°, 120°, or 144°. The DQS phase-shift circuitry uses a frequency reference to dynamically generate control signals for the delay chains in each of the DQS and DQSn 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. This phase-shift circuitry has been enhanced in Stratix II devices to support more phase-shift options with less jitter for use in other DDR applications.

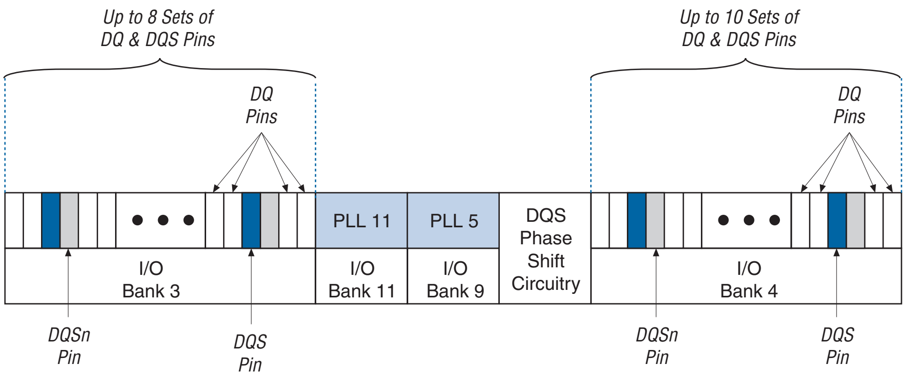
Besides the DQS dedicated phase-shift circuitry, each DQS and DQSn pin has its own DQS logic block that sets the delay for the signal input to the pin. Using the DQS dedicated phase-shift circuitry with the DQS logic block allows for phase-shift fine-tuning. Additionally, every IOE in a Stratix II device contains six registers and one latch to achieve DDR operation.

DDR Memory Interface Pins

Stratix II devices use data (DQ), data strobe (DQS and DQSn), and clock pins to interface with external memory.

Figure 3–4 shows the DQ, DQS, and DQSn pins in the Stratix II I/O banks on the top of the device. A similar arrangement is repeated at the bottom of the device.

Figure 3–4. DQ & DQS Pins Per I/O Bank



Data & Data Strobe Pins

Stratix II data pins for the DDR memory interfaces are called DQ pins. Stratix II devices can use either bidirectional data strobes or unidirectional read clocks. Depending on the external memory interface, either the memory device's read data strobes or read clocks feed the DQS (and DQSn) pins.

In every Stratix II 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 memory up to 300 MHz/600 Mbps (with RLDRAM II). These I/O banks support DQS signals and its complement DQSn signals with DQ bus modes of ×4, ×8/×9, ×16/×18, or ×32/×36.

In ×4 mode, each DQS/DQSn pin drives up to four DQ pins within that group. In ×8/×9 mode, each DQS/DQSn pin drives up to nine DQ pins within that group to support one parity bit and the eight data bits. If the parity bit or any data bit is not used, the extra DQ pins can be used as regular user I/O pins. Similarly, with ×16/×18 and ×32/×36 modes, each DQS/DQSn pin drives up to 18 and 36 DQ pins respectively. There are two parity bits in the ×16/×18 mode and four parity bits in the ×32/×36 mode. Table 3–3 shows the number of DQS/DQ groups supported in each Stratix II density/package combination.

Table 3–3. DQS & DQ Bus Mode Support *Notes (1), (2)*

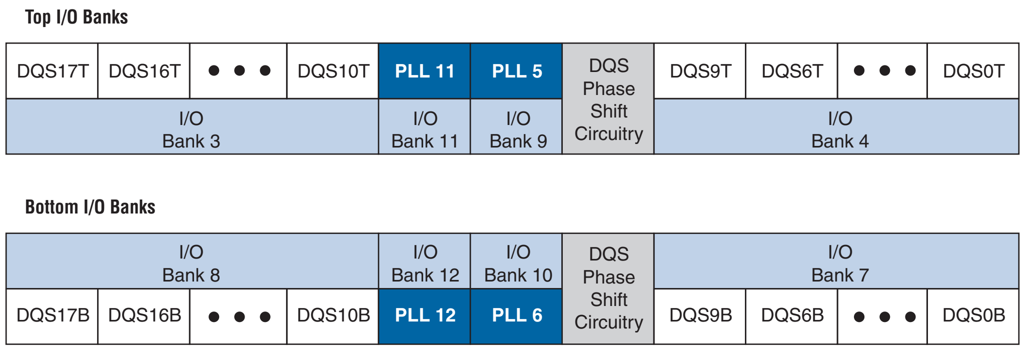
Device	Package	Number of ×4 Groups	Number of ×8/×9 Groups	Number of ×16/×18 Groups	Number of ×32/×36 Groups
EP2S15	484-pin FineLine BGA	8	4	0	0
	672-pin FineLine BGA	18	8	4	0
EP2S30	484-pin FineLine BGA	8	4	0	0
	672-pin FineLine BGA	18	8	4	0
EP2S60	484-pin FineLine BGA	8	4	0	0
	672-pin FineLine BGA	18	8	4	0
	1,020-pin FineLine BGA	36	18	8	4
EP2S90	484-pin Hybrid FineLine BGA	*(3)*	*(3)*	*(3)*	*(3)*
	780-pin FineLine BGA	*(3)*	*(3)*	*(3)*	*(3)*
	1,020-pin FineLine BGA	36	18	8	4
	1,508-pin FineLine BGA	36	18	8	4
EP2S130	780-pin FineLine BGA	*(3)*	*(3)*	*(3)*	*(3)*
	1,020-pin FineLine BGA	36	18	8	4
	1,508-pin FineLine BGA	36	18	8	4
EP2S180	1,020-pin FineLine BGA	36	18	8	4
	1,508-pin FineLine BGA	36	18	8	4

Notes to Table 3–3:
(1) Numbers are preliminary until devices are available.
(2) Check with the pin table for each DQS/DQ group in the different modes.
(3) The number of DQ and DQS buses will be updated in a future version of the *Stratix II Device Handbook*.

The DQS pins are listed in the Stratix II pin tables as `DQS[17..0]T` or `DQS[17..0]B`. The `T` denotes pins on the top of the device and the `B` denotes pins on the bottom of the device. The complement DQSn pins are marked as `DQSn[17..0]T` or `DQSn[17..0]B`. The corresponding DQ pins are marked as `DQ[17..0]T[3..0]`, where `[17..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, DQS, or DQSn pins, these pins are available as regular I/O pins. Figure 3–5 shows the DQS pins in Stratix II I/O banks.

The DQ pin numbering is based on ×4 mode. There are up to 8 DQS/DQ groups in ×4 mode in I/O banks 3 and 8 and up to 10 DQS/DQ groups in ×4 mode in I/O banks 4 and 7. In ×8/×9 mode, two adjacent ×4 DQS/DQ groups plus one parity pin are combined; one pair of DQS/DQSn pins from the combined groups can drive all the DQ and parity pins. Since there is an even number of DQS/DQ groups in an I/O bank, combining groups is efficient. Similarly, in ×16/×18 mode, four adjacent ×4 DQS/DQ groups plus two parity pins are combined and one pair of DQS/DQSn pins from the combined groups can drive all the DQ and parity pins. In ×32/×36 mode, eight adjacent DQS/DQ groups are combined and one pair of DQS/DQSn pins can drive all the DQ and parity pins in the combined groups.

Figure 3–5. DQS Pins in Stratix II I/O Banks *Notes (1), (2), (3)*



Notes to Figure 3–5:

(1) There are up to 18 pairs of DQS and DQSn pins on both the top and bottom of the device. See Table 3–3 for the exact number of DQS and DQSn pin pairs in each device package.

(2) See Table 3–4 for the available DQS and DQSn pins in each mode and package.

(3) Each DQS pin has a complement DQSn pin. DQS and DQSn pins are not differential.

Table 3–4 shows which DQS and DQSn pins are available in each mode and package in the Stratix II device family.

Table 3–4. Available DQS & DQSn Pins in Each Mode & Package Note (1)

Mode	Package				
	484-Pin FineLine BGA	484-Pin Hybrid FineLine BGA	672-Pin FineLine BGA	780-Pin FineLine BGA	1,020-Pin FineLine BGA 1,508-Pin FineLine BGA
×4	7, 9, 11, 13	*(2)*	Odd-numbered pins only	*(2)*	All DQS and DQSn pins
×8/×9	7,11	*(2)*	3, 7, 11, 15	*(2)*	Even-numbered pins only
×16/×18	N/A	*(2)*	5, 13	*(2)*	3, 7, 11, 15
×32/×36	N/A	*(2)*	N/A	*(2)*	5, 13

Note to Table 3–4:

(1) The numbers correspond to the DQS and DQSn pin numbering in the Stratix II pin table. There are two sets of DQS/DQ groups, one corresponding with the top side of the device and one with the bottom side of the device.

(2) These values will be updated in the next version of the *Stratix II Device Handbook*.

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. The DQSn pins can be configured as input, output, or bidirectional pins. You can use the `altdq` and `altdqs` megafunctions to configure the DQ and DQS/DQSn paths, respectively. Altera recommends you use the respective Altera memory controller MegaCore® for your external memory interface data paths. 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.

Stratix II side I/O banks (I/O banks 1, 2, 5, and 6) support SDR, DDR, and DDR2 SDRAM interfaces. They 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 only support DDR SDRAM at speeds up to 150 MHz. You need to use the SSTL-18 Class I I/O standard when interfacing with DDR2 SDRAM devices using pins in I/O bank 1, 2, 5, or 6. These I/O banks do not support the SSTL-18 class II or HSTL I/O standard on output and bidirectional pins, which is required to interface with DDR2 SDRAM, RLDRAM II, or QDRII SRAM.

Table 3–5 shows the maximum clock rate supported for the DDR SDRAM interface in the Stratix II device side I/O banks.

Table 3–5. Maximum Clock Rate for DDR SDRAM in Stratix II Side I/O Banks

Stratix II Device Speed Grade	Maximum DDR SDRAM Speed (MHz)
-3	150
-4	133
-5	133

Clock Pins

You can use any of the DDR I/O registers to generate clocks to the memory device. For better performance, use the same I/O bank as the data and address/command pins.

Command & Address Pins

You can use any of the user I/O pins in the top or bottom bank of the device for commands and addresses. For better performance, use the same I/O bank as the data pins.

Other Pins (Parity, DM, ECC & QVLD Pins)

You can use any of the DQ pins for the parity pins in Stratix II devices. The Stratix II device family has support for parity in the ×8/×9, ×16/×18, and ×32/×36 mode. There is one parity bit available per 8 bits of data pins. Check with the pin table for the exact pins for each mode of the DQS/DQ groups.

The data mask, DM, pins are only required when writing to DDR SDRAM, DDR2 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 will mask the DQ signals. You can use any of the I/O pins in the same bank as the DQ pins (or the RLDRAM II SIO's D pins) for the DM signals. Each group of DQS and DQ signals in DDR and DDR2 SDRAM devices requires a DM pin. There is one DM pin per RLDRAM II device. The DDR register, clocked by the –90° shifted clock, creates the DM signals, similar to DQ output signals.

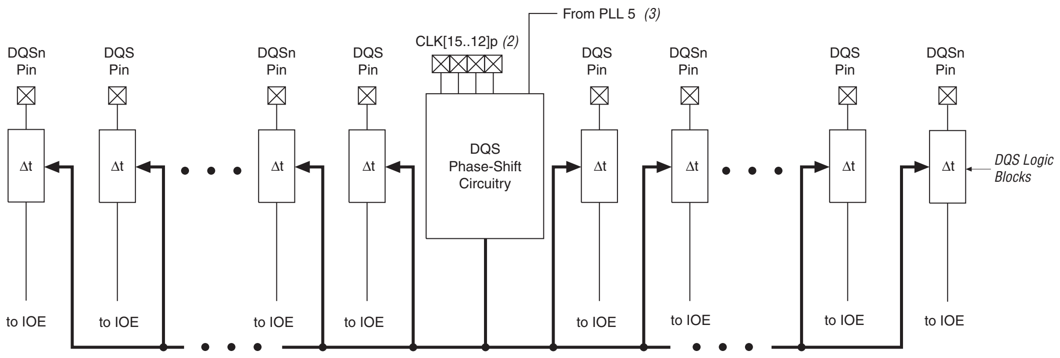
Some DDR SDRAM and DDR2 SDRAM devices support error correction coding (ECC), which is a method of detecting and automatically correcting errors in data transmission. In 72-bit DDR SDRAM, there are eight ECC pins on top of the 64 data pins. Connect the DDR and DDR2 SDRAM ECC pins to a Stratix II device DQS/DQ group. 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 QK/QK# signals and is sent half a clock cycle before data starts coming out of the memory. You need to connect QVLD pins to the DQVLD pin on the Stratix II device. The DQVLD pin can be used as a regular user I/O pin if not used for QVLD.

DQS Phase-Shift Circuitry

The Stratix II phase-shift circuitry and the DQS logic block control the DQS and DQSn pins. Each Stratix II device contains two phase-shifting circuits. There is one circuit for I/O banks 3 and 4, and another circuit for I/O banks 7 and 8. The phase-shifting circuit on the top of the device can control all the DQS and DQSn pins in the top I/O banks and the phase-shifting circuit on the bottom of the device can control all the DQS and DQSn pins in the bottom I/O banks. Figure 3–6 shows the DQS and DQSn pin connections to the DQS logic block and the DQS phase-shift circuitry.

Figure 3–6. DQS & DQSn Pins & the DQS Phase-Shift Circuitry *Note (1)*



Notes to Figure 3–6:

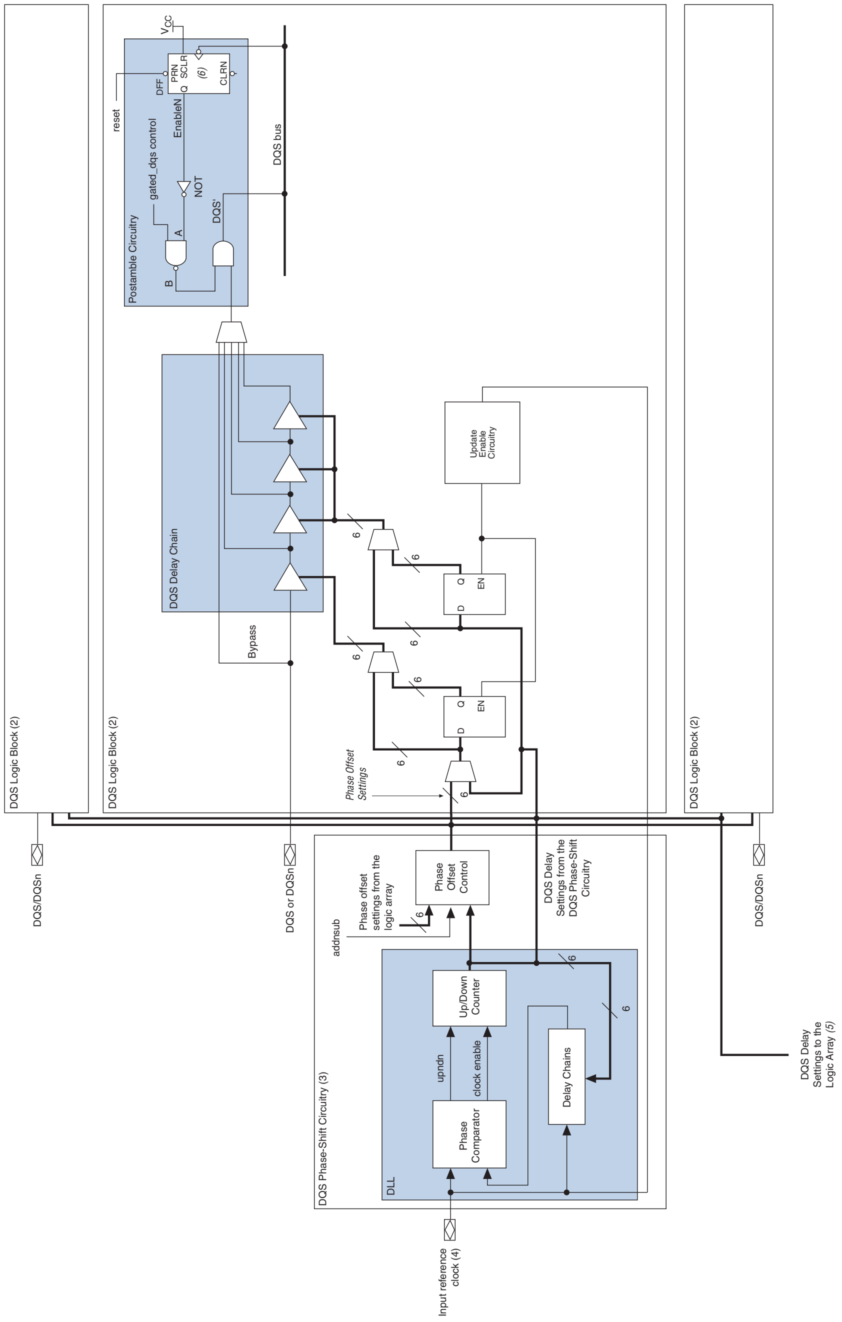
(1) There are up to 18 pairs of DQS and DQSn pins available on the top or the bottom of the Stratix II device, up to 8 on the left side of the DQS phase-shift circuitry (I/O banks 3 and 8), and up to 10 on the right side (I/O bank 4 and 7).

(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-shift circuitry on the bottom of the device. You can also use a phase-locked loop (PLL) clock output as a reference clock to the phase-shift circuitry. The reference clock can also be used in the logic array.

(3) You can only use PLL 5 to feed the DQS phase-shift circuitry on the top of the device and PLL 6 to feed the DQS phase-shift circuitry on the bottom of the device.

Figure 3–7 shows the connections between the DQS phase-shift circuitry and the DQS logic block.

Figure 3–7. DQS Phase-Shift Circuitry & DQS Logic Block Connections *Note (1)*



Notes to Figure 3–7:
(1) All features of the DQS phase-shift circuitry and the DQS logic block are accessible from the `altdqs` megafunction in the Quartus® II software.
(2) DQS logic block is available on every DQS and DQSn pin.
(3) There is one DQS phase-shift circuit on the top and bottom side of the device.
(4) The input reference clock can come from `CLK[15..12]p` or PLL 5 for the DQS phase-shift circuitry on the top side of the device or from `CLK[7..4]p` or PLL 6 for the DQS phase-shift circuitry on the bottom side of the device.
(5) Each individual DQS and DQSn pair can have individual DQS delay settings from the logic array.
(6) This register is one of the DQS IOE input registers.

The phase-shift circuitry is only used during read transactions where the DQS and DQSn pins are acting as input clocks or strobes. The phase-shift circuitry can shift the incoming DQS signal by 0°, 22.5°, 30°, 36°, 45°, 60°, 67.5°, 72°, 90°, 108°, 120°, or 144°. The shifted DQS signal is then used as clocks at the DQ IOE input registers.

Figure 3–1 on page 3–5 shows an example where the DQS signal is shifted by 90°. The DQS signals goes through the 90° shift delay set by the DQS phase-shift circuitry and the DQS logic block and some routing delay from the DQS pin to the DQ IOE registers. The DQ signals only goes through routing delay from the DQ pin to the DQ IOE registers and maintains the 90° relationship between the DQS and DQ signals at the DQ IOE registers since the software will automatically set delay chains to match the routing delay between the pins and the IOE registers for the DQ and DQS input paths.

All 18 DQS and DQSn pins on either the top or bottom of the device can have their input signal phase shifted by a different degree amount but all must be referenced at one particular frequency. For example you can have a 90° phase shift on `DQS0T` and have a 60° phase shift on `DQS1T` both referenced from a 200-MHz clock. Not all phase-shift combinations are supported, however. The phase shifts on the same side of the device must all be a multiple of 22.5° (up to 90°), a multiple of 30° (up to 120°), or a multiple of 36° (up to 144°).

In order 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) or 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). Stratix II devices can also use PLLs 5 or 6 as the reference clock to the DQS phase-shift circuitry on the top or bottom of the device, respectively. PLL 5 is connected to the DQS phase-shift circuitry on the top side of the device and PLL 6 is connected to the DQS phase-shift circuitry on the bottom side of the device. Both the top and bottom phase-shift circuits need unique clock pins or PLL clock outputs for the reference clock.

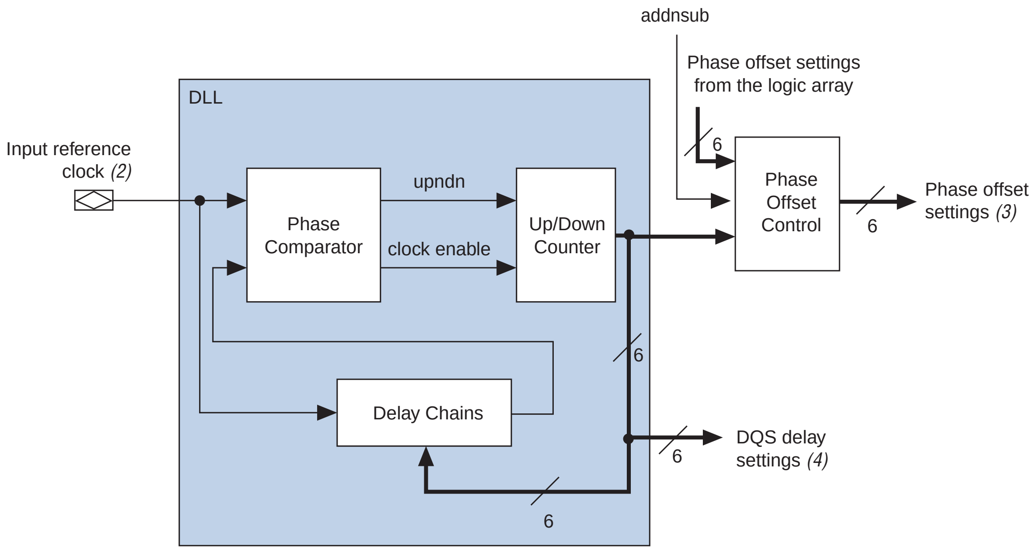
DLL

The DQS phase-shift circuitry uses a delay-locked loop (DLL) to dynamically measure the clock period needed by the DQS/DQSn pin (see Figure 3–8). The DQS phase-shift circuitry then uses the clock period to generate the correct phase shift. The DLL in the Stratix II DQS phase-shift circuitry can operate between 100 and 300 MHz in either fast lock mode or low jitter mode. The fast lock mode requires fewer clock cycles to calculate the input clock period, but the low jitter mode is more accurate. Use the `altdqs` megafunction in the Quartus II software to set these modes. The default setting is low jitter mode. The phase-shift circuitry needs a maximum of 256 clock cycles in the fast lock mode and needs a maximum of 1,280 clock cycles in the low jitter mode to calculate the correct input clock period. Data sent during these clock cycles may not be properly captured.

☞ You can still use the DQS phase-shift circuitry for DDR SDRAM interfaces that are less than 100 MHz. The DQS signal will be shifted by 2.5 ns and you can add more shift by using the phase offset module. Even if the DQS signal is not shifted exactly to the middle of the DQ valid window, the IOE should still be able to capture the data in this low frequency application.

The DLL can be reset from either the logic array or a user I/O pin. This signal is not shown in Figure 3–8. Each time the DLL is reset, you must wait for 256 or 1,280 clock cycles (depending on the DLL mode) before you can capture the data properly.

Figure 3–8. Simplified Diagram of the DQS Phase-Shift Circuitry *Note (1)*



Notes to Figure 3–8:

(1) All features of the DQS phase-shift circuitry are accessible from the `altdqs` megafunction in the Quartus II software.
(2) The input reference clock for the DQS phase-shift circuitry on the top side of the device can come from `CLK[15..12]p` or PLL 5. The input reference clock for the DQS phase-shift circuitry on the bottom side of the device can come from `CLK[7..4]p` or PLL 6.
(3) Phase offset settings can only go to the DQS logic blocks.
(4) DQS delay settings can go to the logic array and/or to the DQS logic block.

The input reference clock goes into the DLL to a chain of up to 16 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 (DQS delay settings) that will increase or decrease 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 DQS delay settings (the up/down counter output) are updated every eight clock cycles. If the low jitter mode is enabled, the phase comparator also issues a clock enable signal to the up/down counter to tell the counter when to update the DQS settings. In the low jitter mode, the enable signal is only active when the `upndn` signal has been incremented or decremented by 4, otherwise the clock enable is off and the DQS delay settings do not get updated. This enable signal is always active if the DLL is in the fast lock mode.

The `altdqs` megafunction sets the DLL in low jitter mode by default. This setting is not configurable on the fly, so you must generate a new programming file if you want to switch between fast lock and low jitter mode. Use low jitter mode whenever possible. Use fast lock mode if your timing margin is sufficient.

The DQS delay settings contain the control bits to shift the signal on the input DQS pin by the amount set in the `altdqs` megafunction. For the 0° shift, both the DLL and the DQS logic block are bypassed. Since Stratix II DQS and DQ pins are designed such that the pin to IOE delays are matched, the skew between the DQ and DQS pin at the DQ IOE registers is negligible when the 0° shift is implemented. You can feed the DQS delay settings to the DQS logic block and the logic array.

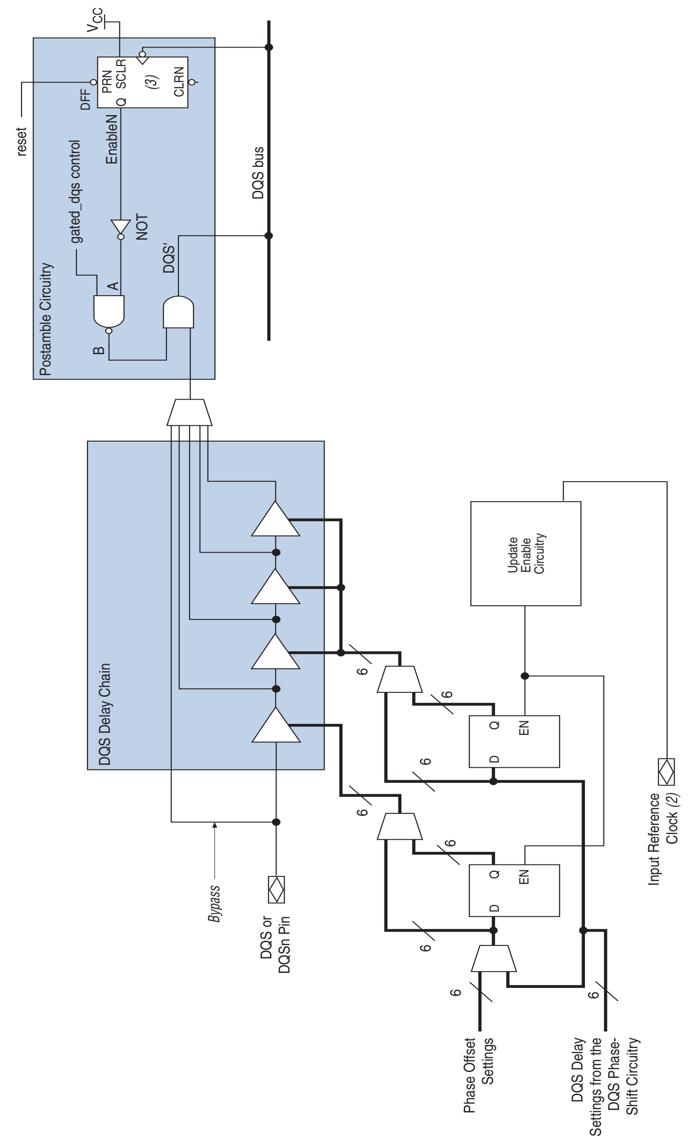
Phase Offset Control

The DQS phase-shift circuitry also contains a phase offset control module that can add or subtract a phase offset amount from the DQS delay setting (phase offset settings from the logic array in Figure 3–8). You should use the phase offset control module for making small shifts to the input signal and use the DQS phase-shift circuitry for larger signal shifts. For example, if you need the input signal to be shifted by 75°, you can set the `altdqs` megafunction to generate a 72° phase shift with a phase offset of +3°. The phase offset settings are passed into the phase offset control module as a 6-bit unsigned number along with the `addnsub` signal to indicate phase addition or subtraction. You can also bypass the DLL and only send the phase offset amount to the DQS logic block. The resolution for the phase-offset delay is ~14 ps.

DQS Logic Block

Each DQS and DQSn pin is connected to a separate DQS logic block (see Figure 3–9). The logic block contains DQS delay chains and postamble circuitry.

Figure 3–9. Simplified Diagram of the DQS Logic Block Note (1)



Notes to Figure 3–9:

(1) All features of the DQS logic block are accessible from the `altdqs` megafunction in the Quartus II software.

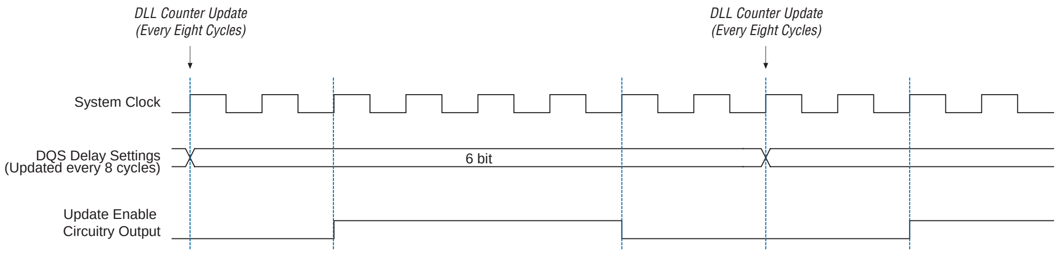
(2) The input reference clock for the DQS phase-shift circuitry on the top side of the device can come from `CLK[15..12]p` or PLL 5. The input reference clock for the DQS phase-shift circuitry on the top side of the device can come from `CLK[7..4]p` or PLL 6.

(3) This register is one of the DQS IOE input registers.

The DQS delay chains consist of a set of variable delay chains to allow the input DQS and DQSn signals to be shifted by the amount given by the DQS phase-shift circuitry. There are four delay elements in the DQS delay chain; the first delay chain closest to the DQS pin can either be shifted by the phase offset settings when used, or by the DQS delay settings when the phase offset control is not used. The number of delay chains used is transparent to the users because the `altdqs` megafunction automatically sets it. The DQS delay settings can come from the DQS phase-shift circuitry on the same side of the device as the target DQS logic block or from the logic array. When you apply a 0° shift in the `altdqs` megafunction, the DQS delay chains are bypassed.

Both the DQS delay settings and the phase offset settings pass through a latch before going into the DQS delay chains. The latches reduce skew between the DQS delay settings in all the DQS logic blocks. You can disable the latch if the design does not use multiple DQS pins (skew is only an issue if the design uses multiple DQS pins). The update enable circuitry enables the latch to allow enough time for the DQS delay settings to travel from the DQS phase-shift circuitry to all the DQS logic blocks before the next change. It uses the input reference clock to generate the update enable output. The `altdqs` megafunction uses this circuit by default. See Figure 3–10 for an example waveform of the update enable circuitry output.

Figure 3–10. DQS Update Enable Waveform



For external memory interfaces that use a bidirectional read strobe like DDR and DDR2 SDRAM, the DQS signal is low before going to or coming from a high-impedance state. See Figure 3–1. 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 a high-impedance state, is called the postamble. There are preamble and postamble specifications for both read and write operations in DDR and DDR2 SDRAM. The DQS postamble circuitry ensures data is not lost when there is noise on the DQS line at the end of a read postamble time. It is to be used with one of the DQS IOE input registers such that the DQS postamble control signal can ground the shifted DQS signal used to clock the DQ input registers at the end of a read operation. This ensures that any glitches on the DQS input signals at the end of the read postamble time do not affect the DQ IOE registers.

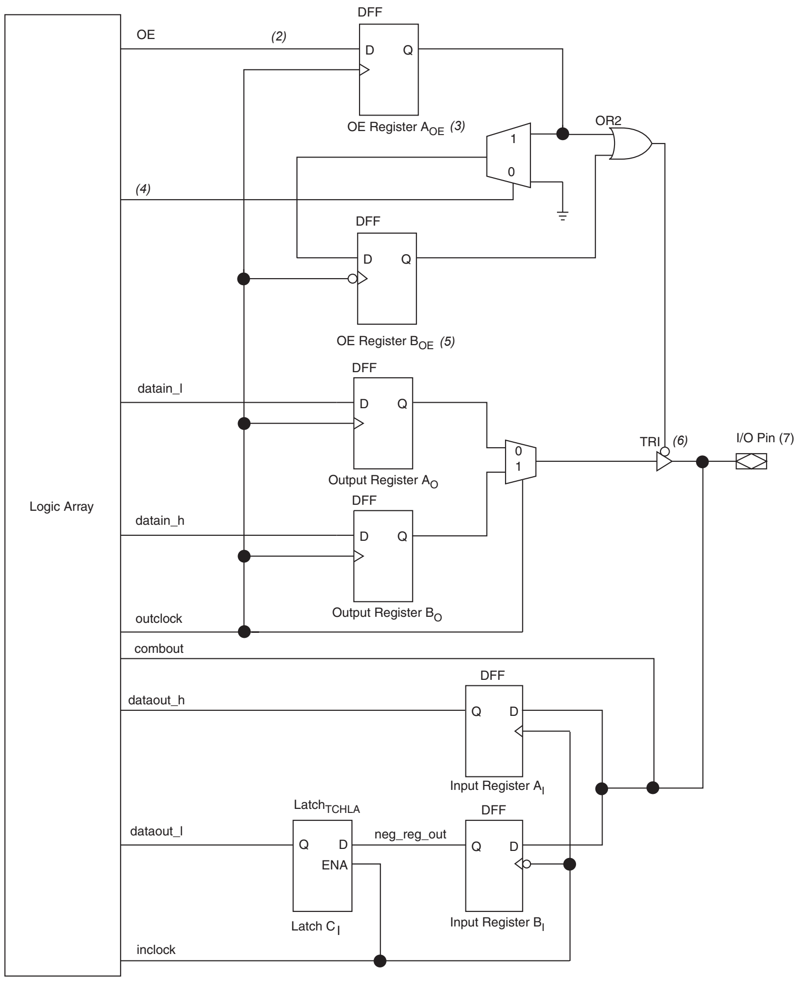
See *AN 327: Interfacing DDR SDRAM with Stratix II Devices* and *AN 328: Interfacing DDR2 SDRAM with Stratix II Devices* for more details.

The shifted DQS signal then goes to the DQS bus to clock the IOE input registers of the DQ pins. It can also go into the logic array for resynchronization purposes. The shifted DQSn signal can only go to the active-low input register in the DQ IOE and is only used for QDRII SRAM interfaces.

DDR Registers

Each IOE in a Stratix II device 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 active low output enable register extends the high-impedance state of the pin by a half clock cycle to provide the external memory's DQS write preamble time specification. Figure 3–11 shows the six registers and the latch in the Stratix II IOE and Figure 3–12 shows how the second OE register extends the DQS high-impedance state by half a clock cycle during a write operation.

Figure 3–11. Bidirectional DDR I/O Path in Stratix II Devices *Note (1)*



Notes to Figure 3–11:

(1) All control signals can be inverted at the IOE. The signal names used here match with Quartus II software naming convention.

(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.

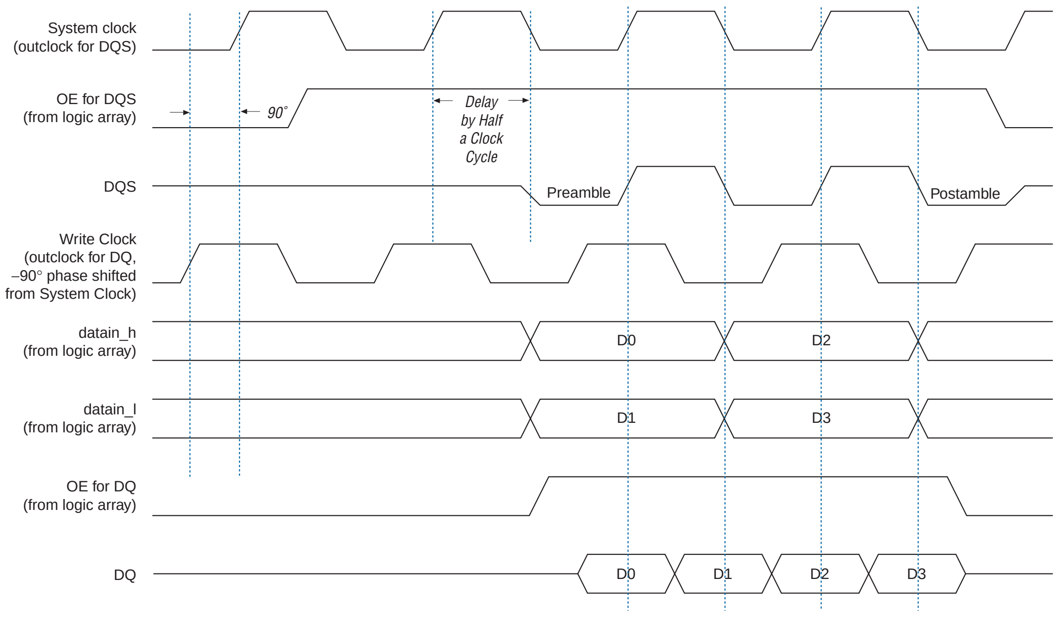
(4) This select line is to choose whether the `OE` signal should be delayed by half-a-clock cycle.

(5) The B_{OE} register generates the delayed enable signal for the write strobes or write clocks for memory interfaces.

(6) The tristate enable is by default active low. You can, however, 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 3–12. Extending the OE Disable by Half-a-Clock Cycle for a Write Transaction Note (1)

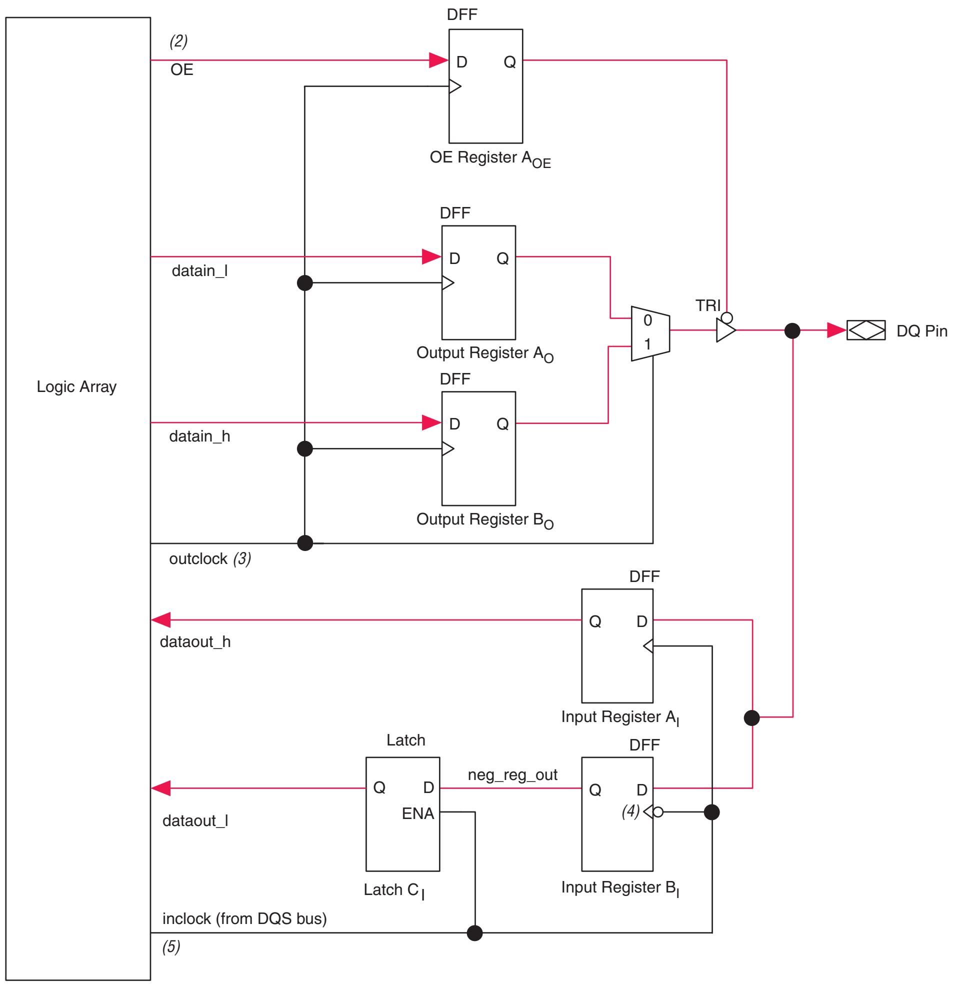

System clock (outclock for DQS)
OE for DQS (from logic array)
90°
Delay by Half a Clock Cycle
DQS
Preamble
Postamble
Write Clock (outclock for DQ, –90° phase shifted from System Clock)
datain_h (from logic array)
D0
D2
datain_l (from logic array)
D1
D3
OE for DQ (from logic array)
DQ
D0
D1
D2
D3

Note to Figure 3–12:

(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 3–13 and 3–14 summarize the IOE registers used for the DQ and DQS signals.

Figure 3–13. DQ Configuration in Stratix II IOE Note (1)



Notes to Figure 3–13:

(1) You can use the `altdq` megafunction to generate the DQ signals. The signal names used here match with Quartus II software naming convention.

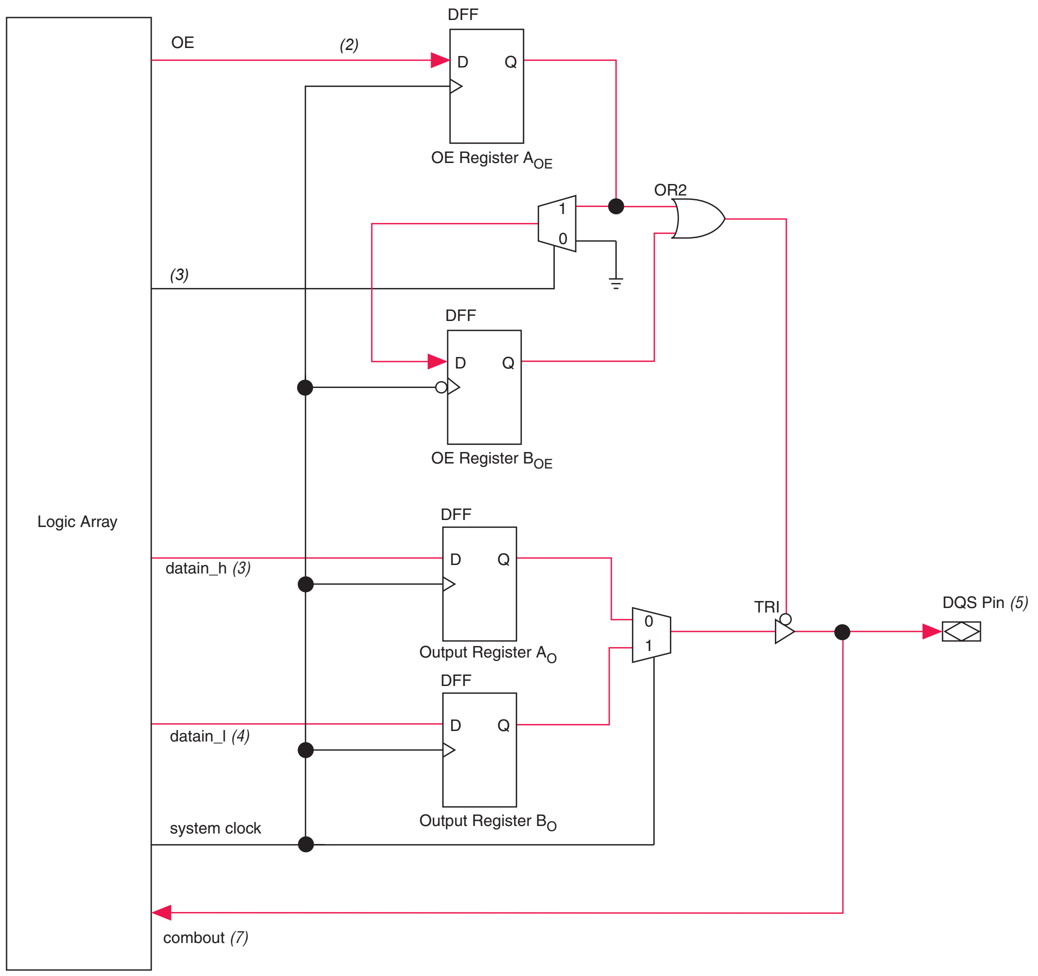
(2) The `OE` signal is active low, but the Quartus II software implements this as active high and automatically adds an inverter before the `OE` register A_{OE} during compilation.

(3) The `outclock` signal is –90° phase shifted from the system clock.

(4) The shifted DQS or DQSn signal can clock this register. Only use the DQSn signal for QDRII SRAM interfaces.

(5) The shifted DQS signal must be inverted before going to the DQ IOE. The inversion is automatic if you use the `altdq` megafunction to generate the DQ signals. Connect this port to the `combout` port in the `altdqs` megafunction.

Figure 3–14. DQS Configuration in Stratix II IOE *Note (1)*



Notes to Figure 3–14:

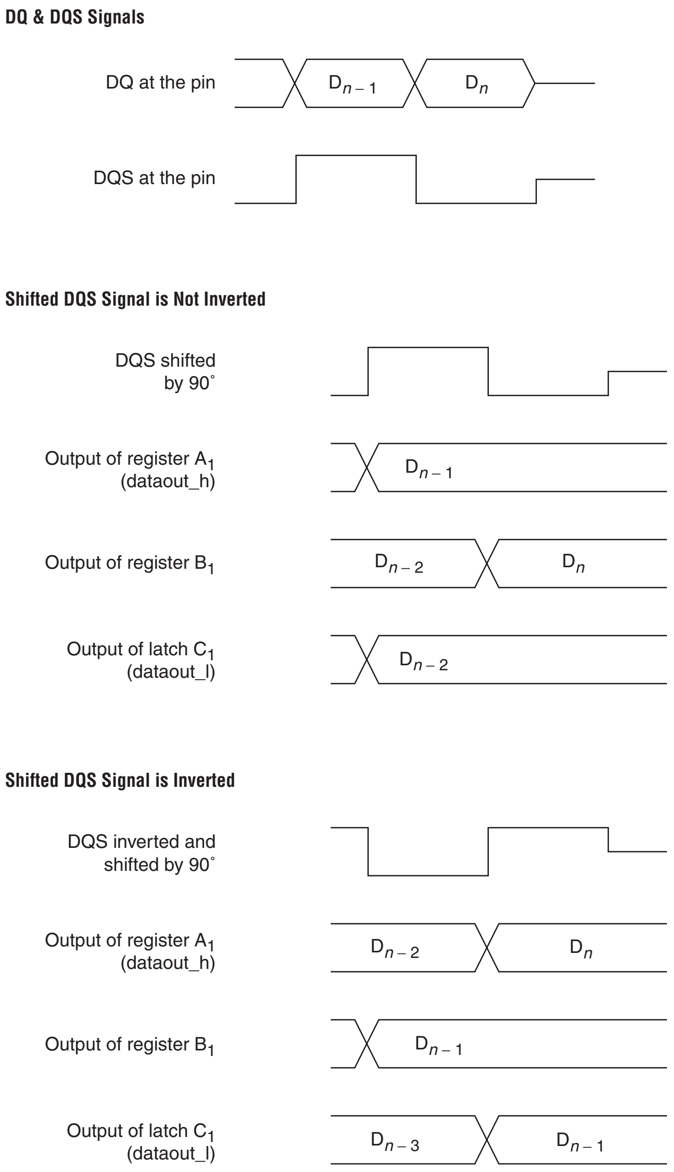
(1) You can use the `altdqs` megafunction to generate the DQS signals. The signal names used here match with Quartus II software naming convention.
(2) The `OE` signal is active low, but the Quartus II software implements this as active high and automatically adds an inverter before `OE` register A_{OE} during compilation.
(3) The select line can be chosen in the `altdqs` megafunction.
(4) The `datain_l` and `datain_h` pins are usually connected to ground and V_{CC}, respectively.
(5) DQS postamble circuitry and handling is not shown in this diagram. For more information, see *AN 327: Interfacing DDR SDRAM with Stratix II Devices* and *AN 328: Interfacing DDR2 SDRAM with Stratix II Devices*.
(6) DQS logic blocks are only available with DQS and DQSn pins.
(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.

The Stratix II DDR IOE structure requires you to invert the incoming DQS signal to ensure proper data transfer. The `altdq` megafunction automatically adds the inverter to the `inclock` port when it generates the DQ signals. As shown in Figure 3–11 on page 3–26, 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 1. In a DDR memory read operation, the last data coincides with DQS being low. If you do not invert the DQS pin, you will not get this last data as the latch does not open until the next rising edge of the DQS signal.

Figure 3–15 shows waveforms of the circuit shown in Figure 3–13 on page 3–28.

The first set of waveforms in Figure 3–15 shows the edge-aligned relationship between the DQ and DQS signals at the Stratix II device pins. The second set of waveforms in Figure 3–15 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 3–15 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. Register A_I, register B_I, and latch C_I refer to the nomenclature in Figure 3–13 on page 3–28.

Figure 3–15. DQ Captures with Non-Inverted & Inverted Shifted DQS


DQ & DQS Signals
DQ at the pin
D_{n-1}
D_n
DQS at the pin
Shifted DQS Signal is Not Inverted
DQS shifted by 90˚
Output of register A_1 (dataout_h)
D_{n-1}
Output of register B_1
D_{n-2}
D_n
Output of latch C_1 (dataout_l)
D_{n-2}
Shifted DQS Signal is Inverted
DQS inverted and shifted by 90˚
Output of register A_1 (dataout_h)
D_{n-2}
D_n
Output of register B_1
D_{n-1}
Output of latch C_1 (dataout_l)
D_{n-3}
D_{n-1}

PLL

When using the Stratix II 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 either shifted by –90° or 90° from the system clock and is used to generate the DQ signals during writes.

When using the Stratix II 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. Since the side I/O banks do not have dedicated circuitry, one PLL captures data from the DDR SDRAM and another PLL generates the write signals, commands, and addresses to the DDR SDRAM device. Stratix II side I/O banks can support DDR SDRAM up to 150 MHz.

Enhancements In Stratix II Devices

Stratix II external memory interfaces support differs from Stratix external memory interfaces support in the following ways:

- A PLL output can now be used as the input reference clock to the DLL.
- The shifted DQS signal can now go into the logic array.
- The DLL in Stratix II devices has more phase-shift options than in Stratix devices. It also has the option to add phase offset settings.
- The DLL has been enhanced to offer low jitter mode.
- Stratix II devices have DQS logic blocks with each DQS pin that helps with fine tuning the phase shift.
- The DQS delay settings can be routed from the DLL into the logic array. You can also bypass the DLL and send the DQS delay settings from the logic array to the DQS logic block.
- Stratix II devices support DQSn pins.
- The DQS/DQ groups now support ×4, ×9, ×18, and ×36 bus modes.
- The DQS pins have been enhanced with the DQS postamble circuitry.

Conclusion

Stratix II devices support SDR SDRAM, DDR SDRAM, DDR2 SDRAM, RLDRAM II, and QDRII external memories. Stratix II devices feature high-speed interfaces that transfer data between external memory devices at up to 300 MHz/600 Mbps. DQS phase-shift circuitry and DQS logic blocks within the Stratix II devices allow you to fine-tune the phase shifts for the input clocks or strobes to properly align clock edges as needed to capture data.

Section III. I/O Standards

This section provides information on Stratix® II single-ended, voltage-referenced, and differential I/O standards.

This section contains the following chapters:

Revision History

The table below shows the revision history for Chapters 4 and 5.

Chapter	Date / Version	Changes Made
4	January 2005, v2.0	● Updated the "Differential I/O Standards", "LVDS", "On-Chip Termination", "1.8 V", and "1.5 V"sections ● Updated Tables 4–2, 4–5, 4–6, and 4–7.
	July 2004, v1.1	● Updated "LVTTL" and "LVCMOS" in "Single-Ended I/O Standards". ● Updated Figure 4–20. ● Updated Tables 4–4 and 4–6. ● Added "I/O Pin Placement with Respect to High-Speed Differential I/O Pins" section. ● Updated "Single-Ended I/O Standards" and "Differential I/O Standards" sections.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.
5	January 2005, 2.0	Updated the "Differential Transmitter", "Differential I/O Termination", and "Differential Pin Placement Guidelines" sections.
	October 2004, v1.2	● Updated Table 5–2.
	July 2004, v1.1	● Updated Table 5–2, Device EP2S130. ● Updated Figures 5–1 and 5–7.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.

4. Selectable I/O Standards in Stratix II Devices

SII52004-2.0

Introduction

This chapter provides guidelines for using industry I/O standards in Stratix® II devices, including:

- I/O features
- I/O standards
- External memory interfaces
- I/O banks
- Design considerations

Stratix II I/O Features

Stratix II devices contain an abundance of adaptive logic modules (ALMs), embedded memory, high-bandwidth digital signal processing (DSP) blocks, and extensive routing resources, all of which can operate at very high core speed.

The Stratix II device I/O structure is designed to ensure that these internal capabilities are fully utilized. There are numerous I/O features to assist in high-speed data transfer into and out of the device including:

- Single-ended, non-voltage-referenced and voltage-referenced I/O standards
- High-speed differential I/O standards featuring serializer/deserializer (SERDES) and dynamic phase alignment (DPA), capable of 1 gigabit per second (Gbps) performance for low-voltage differential signaling (LVDS)
- Double data rate (DDR) I/O pins
- Programmable output drive strength for voltage-referenced and non-voltage-referenced single-ended I/O standards
- Programmable bus-hold
- Programmable pull-up resistor
- Open-drain output
- On-chip series termination
- On-chip differential termination
- Peripheral component interconnect (PCI) clamping diode
- Hot socketing

For a detailed description of each I/O feature, refer to the *Stratix II Architecture* chapter in Volume 1 of the *Stratix II Device Handbook*

Stratix II I/O Standards Support

Stratix II devices support a wide range of industry I/O standards. Table 4–1 shows which I/O standards Stratix II devices support as well as typical applications. The performance target of each I/O standard is provided in Table 4–2.

Table 4–1. I/O Standard Applications

I/O Standard	Application
LVTTL	General purpose
LVCMOS	General purpose
2.5 V	General purpose
1.8 V	General purpose
1.5 V	General purpose
3.3-V PCI	PC and embedded system
3.3-V PCI-X	PC and embedded system
SSTL-2 class I	DDR SDRAM
SSTL-2 class II	DDR SDRAM
SSTL-18 class I	DDR2 SDRAM
SSTL-18 class II	DDR2 SDRAM
1.8-V HSTL class I	QDRII SRAM/RLDRAM II/SRAM
1.8-V HSTL class II	QDRII SRAM/RLDRAM II/SRAM
1.5-V HSTL class I	SRAM
1.5-V HSTL class II	QDRII SRAM/SRAM
Differential SSTL-2 class I	DDR SDRAM
Differential SSTL-2 class II	DDR SDRAM
Differential SSTL-18 class I	DDR2 SDRAM
Differential SSTL-18 class II	DDR2 SDRAM
1.8-V differential HSTL class I	Clock interfaces
1.8-V differential HSTL class II	Clock interfaces
1.5-V differential HSTL class I	Clock interfaces
1.5-V differential HSTL class II	Clock interfaces
LVDS	High-speed communications
HyperTransport™ technology	PCB interfaces
Differential LVPECL	Video graphics and clock distribution

Table 4–2. I/O Standard Performance Target Notes (1), (2), (3)

I/O Standard	Clock Rate / Single Data Rate (MHz)	Double Data Rate (Mbps)
LVTTL	300	600
LVCMOS	300	600
2.5 V	300	600
1.8 V	250	500
1.5 V	200	400
3.3-V PCI	66	-
3.3-V PCI-X	133	266
SSTL-2 class I	200	400
SSTL-2 class II	200	400
SSTL-18 class I	267	533
SSTL-18 class II	267	533
1.8-V HSTL class I	300	600
1.8-V HSTL class II	300	600
1.5-V HSTL class I	250	500
1.5-V HSTL class II	250	500
Differential SSTL-2 class I	200	400
Differential SSTL-2 class II	200	400
Differential SSTL-18 class I	267	533
Differential SSTL-18 class II	267	533
1.8-V differential HSTL class I	300	600
1.8-V differential HSTL class II	300	600
1.5-V differential HSTL class I	300	600
1.5-V differential HSTL class II	300	600
LVDS	500	1000
HyperTransport technology	500	1000
Differential LVPECL	450	900

Notes to Table 4–2:

(1) The data rate and clock rate requirements for different applications may be different.

(2) At different I/O banks, the performance target of each I/O standard could be different when configured as either an input or output.

(3) Performance target for each I/O standard varies across device speed grades. For more information, see the *DC & Switching Characteristics* chapter in Volume 1 of the *Stratix II Device Handbook*.

Single-Ended I/O Standards

In non-voltage-referenced single-ended I/O standards, the voltage at the input must be above a set voltage to be considered "on" (high, or logic value 1) or below another voltage to be considered "off" (low, or logic value 0). Voltages between the limits are undefined logically, and may fall into either a logic value 0 or 1. The non-voltage-referenced single-ended I/O standards supported by Stratix II devices are:

- Low-voltage transistor-transistor logic (LVTTL)
- Low-voltage complementary metal-oxide semiconductor (LVCMOS)
- 1.5 V
- 1.8 V
- 2.5 V
- 3.3-V PCI
- 3.3-V PCI-X

Voltage-referenced, single-ended I/O standards provide faster data rates with less board-level noise and power consumption. These standards use a constant reference voltage at the input levels. The incoming signals are compared with this constant voltage and the difference between the two defines "on" and "off" states. Stratix II devices support stub series terminated logic (SSTL) and high-speed transceiver logic (HSTL) voltage-referenced I/O standards.

LVTTL

The LVTTL standard is formulated under EIA/JEDEC Standard, JESD8-B (Revision of JESD8-A): Interface Standard for Nominal 3-V/3.3-V Supply Digital Integrated Circuits.

The standard defines DC interface parameters for digital circuits operating from a 3.0- or 3.3-V power supply and driving or being driven by LVTTL-compatible devices. The 3.3-V LVTTL standard is a general-purpose, single-ended standard used for 3.3-V applications. This I/O standard does not require input reference voltages (V_{REF}) or termination voltages (V_{TT}). Stratix II devices support both input and output levels for 3.3-V LVTTL operation. Stratix II devices support a V_{CCIO} voltage level of 3.3 V ± 5% as specified as the narrow range for the voltage supply by the EIA/JEDEC standard.

LVCMOS

The LVCMOS standard is formulated under EIA/JEDEC Standard, JESD8-B (Revision of JESD8-A): Interface Standard for Nominal 3-V/3.3-V Supply Digital Integrated Circuits.

The standard defines DC interface parameters for digital circuits operating from a 3.0- or 3.3-V power supply and driving or being driven by LVCMOS-compatible devices. The 3.3-V LVCMOS I/O standard is a general-purpose, single-ended standard used for 3.3-V applications. While LVCMOS has its own output specification, it specifies the same input voltage requirements as LVTTL. These I/O standards do not require V_{REF} or V_{TT}. Stratix II devices support both input and output levels for 3.3-V LVCMOS operation. Stratix II devices support a V_{CCIO} voltage level of 3.3 V ± 5% as specified as the narrow range for the voltage supply by the EIA/JEDEC standard.

2.5 V

The 2.5-V I/O standard is formulated under EIA/JEDEC Standard, EIA/JESD8-5: 2.5-V± 0.2-V (Normal Range), and 1.8-V – 2.7-V (Wide Range) Power Supply Voltage and Interface Standard for Non-Terminated Digital Integrated Circuit.

The 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. This standard is a general-purpose, single-ended standard used for 2.5-V applications. It does not require the use of a V_{REF} or a V_{TT}. Stratix II devices support both input and output levels for 2.5-V operation with V_{CCIO} voltage level support of 2.5 V ± 5%, which is narrower than defined in the Normal Range of the EIA/JEDEC standard.

1.8 V

The 1.8-V I/O standard is formulated under EIA/JEDEC Standard, EIA/JESD8-7: 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 Circuit.

The 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. This standard is a general-purpose, single-ended standard used for 1.8-V applications. It does not require the use of a V_{REF} or a V_{TT}. Stratix II devices support both input and output levels for 1.8-V operation with V_{CCIO} voltage level support of 1.8 V ± 5%, which is narrower than defined in the Normal Range of the EIA/JEDEC standard.

1.5 V

The 1.5-V I/O standard is formulated under EIA/JEDEC Standard, JESD8-11: 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 Circuit.

The 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. This standard is a general-purpose, single-ended standard used for 1.5-V applications. It does not require the use of a V_{REF} or a V_{TT}. Stratix II devices support both input and output levels for 1.5-V operation V_{CCIO} voltage level support of 1.8 V ± 5%, which is narrower than defined in the Normal Range of the EIA/JEDEC standard.

3.3-V PCI

The 3.3-V PCI I/O standard is formulated under PCI Local Bus Specification Revision 2.2 developed by the PCI Special Interest Group (SIG).

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 2.2 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 II devices are fully compliant with the 3.3-V PCI Local Bus Specification Revision 2.2 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 II devices support both input and output levels operation.

3.3-V PCI-X

The 3.3-V PCI-X I/O standard is formulated under PCI-X Local Bus Specification Revision 1.0a developed by the PCI SIG.

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 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 II 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-X standard does not require input reference voltages or board terminations. Stratix II devices support both input and output levels operation.

SSTL-2 Class I & SSTL-2 Class II

The 2.5-V SSTL-2 standard is formulated under JEDEC Standard, JESD8-9A: Stub Series Terminated Logic for 2.5-V (SSTL_2).

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. 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 4–1 and 4–2). Stratix II devices support both input and output levels.

Figure 4–1. 2.5-V SSTL Class I Termination

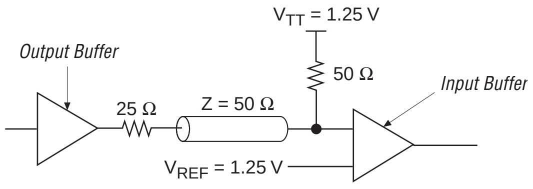
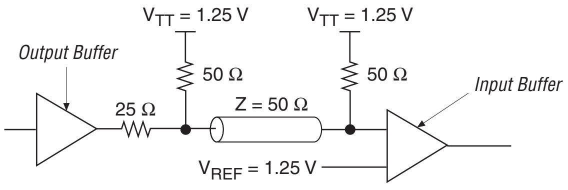


Figure 4–2. 2.5-V SSTL Class II Termination



SSTL-18 Class I & SSTL-18 Class II

The 1.8-V SSTL-18 standard is formulated under JEDEC Standard, JESD8-15: Stub Series Terminated Logic for 1.8-V (SSTL_18).

The SSTL-18 I/O standard is a 1.8-V memory bus standard used for applications such as high-speed DDR2 SDRAM interfaces. 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. There are no class definitions for the SSTL-18 standard in the JEDEC specification. The specification of this I/O standard is based on an environment that consists of both series and parallel terminating resistors. Altera provides solutions to two derived applications in JEDEC specification, and names them class I and class II to be consistent with other SSTL standards. Figures 4–3 and 4–4 show SSTL-18 class I and II termination, respectively. Stratix II devices support both input and output levels.

Figure 4–3. 1.8-V SSTL Class I Termination

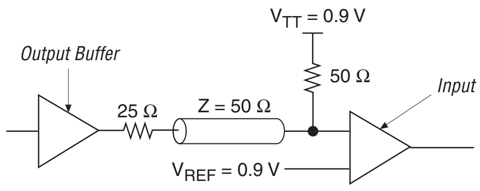
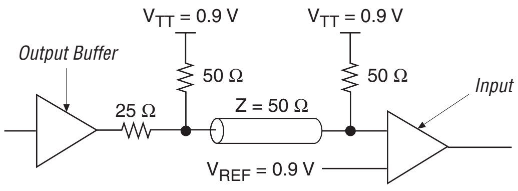


Figure 4–4. 1.8-V SSTL Class II Termination



1.8-V HSTL Class I & 1.8-V HSTL Class II

The HSTL standard is a technology-independent I/O standard developed by JEDEC to provide voltage scalability. It is used for applications designed to operate in the 0.0- to 1.8-V HSTL logic switching range such as quad data rate (QDR) memory clock interfaces.

Although JEDEC specifies a maximum V_{CCIO} value of 1.6 V, there are various memory chip vendors with HSTL standards that require a V_{CCIO} of 1.8 V. Stratix II devices support interfaces to chips with V_{CCIO} of 1.8 V for HSTL. Figures 4–5 and 4–6 show the nominal V_{REF} and V_{TT} required to track the higher value of V_{CCIO}. The value of V_{REF} is selected to provide optimum noise margin in the system. Stratix II devices support both input and output levels operation.

Figure 4–5. 1.8-V HSTL Class I Termination

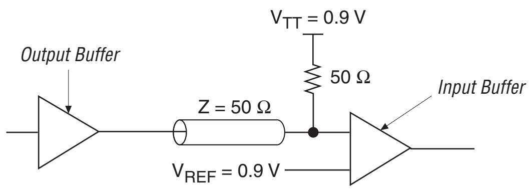
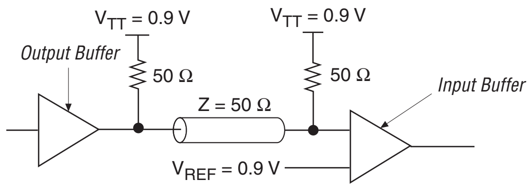


Figure 4–6. 1.8-V HSTL Class II Termination



1.5-V HSTL Class I & 1.5-V HSTL Class II

The 1.5-V HSTL standard is formulated under EIA/JEDEC Standard, EIA/JESD8-6: A 1.5-V Output Buffer Supply Voltage Based Interface Standard for Digital Integrated Circuits.

The 1.5-V HSTL I/O standard is used for applications designed to operate in the 0.0- to 1.5-V HSTL logic nominal switching range. This standard defines single-ended input and output specifications for all HSTL-compliant digital integrated circuits. The 1.5-V HSTL I/O standard in Stratix II devices is compatible with the 1.8-V HSTL I/O standard in APEX™ 20KE, APEX 20KC, and in Stratix II devices themselves because the input and output voltage thresholds are compatible. See Figures 4–7 and 4–8. Stratix II devices support both input and output levels with V_{REF} and V_{TT}.

Figure 4–7. 1.5-V HSTL Class I Termination

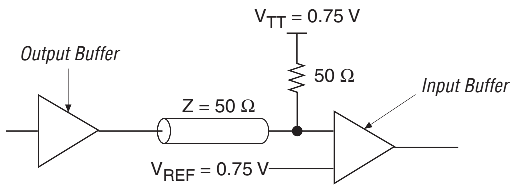
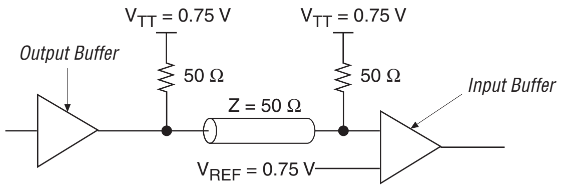


Figure 4–8. 1.5-V HSTL Class II Termination



Differential I/O Standards

Differential I/O standards are used to achieve even faster data rates with higher noise immunity. Apart from LVDS, LVPECL, and HyperTransport technology, Stratix II devices also support differential versions of SSTL and HSTL standards.

For detailed information on differential I/O standards, refer to the *High-Speed Differential I/O Interfaces with DPA in Stratix II Devices* chapter in Volume 2 of the *Stratix II Device Handbook*.

Differential SSTL-2 Class I & Differential SSTL-2 Class II

The 2.5-V differential SSTL-2 standard is formulated under JEDEC Standard, JESD8-9A: Stub Series Terminated Logic for 2.5-V (SSTL_2).

This 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. Stratix II devices support both input and output levels. See Figures 4–9 and 4–10 for details on differential SSTL-2 termination.

Stratix II devices support differential SSTL-2 I/O standards in pseudo-differential mode, which is implemented by using two SSTL-2 single-ended buffers.

The Quartus® II software only supports pseudo-differential standards on the `INCLK`, `FBIN` and `EXTCLK` ports of enhanced PLL, as well as on DQS pins when DQS megafunction (ALTDQS, Bidirectional Data Strobe) is used. Two single-ended output buffers are automatically programmed to have opposite polarity so as to implement a pseudo-differential output. A proper V_{REF} voltage is required for the two single-ended input buffers to implement a pseudo-differential input. In this case, only the positive polarity input is used in the speed path while the negative input is not connected internally. In other words, only the non-inverted pin is required to be specified in your design, while the Quartus II software will automatically generate the inverted pin for you.

Although the Quartus II software does not support pseudo-differential SSTL-2 I/O standards on the left and right I/O banks, you can implement these standards at these banks. You need to create two pins in the designs and configure the pins with single-ended SSTL-2 standards. However, this is limited only to pins that support the differential pin-pair I/O function and is dependent on the single-ended SSTL-2 standards support at these banks.

Figure 4–9. Differential SSTL-2 Class I Termination

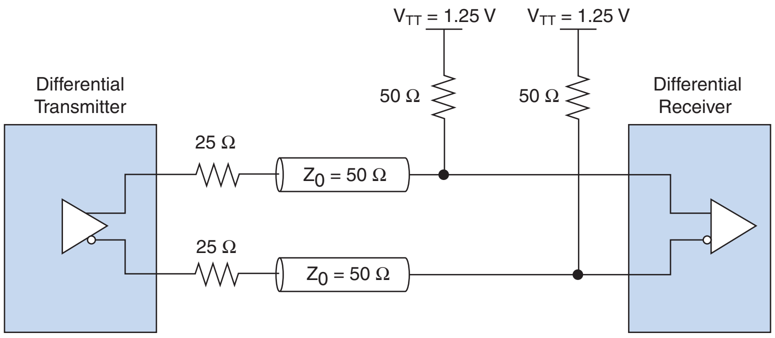
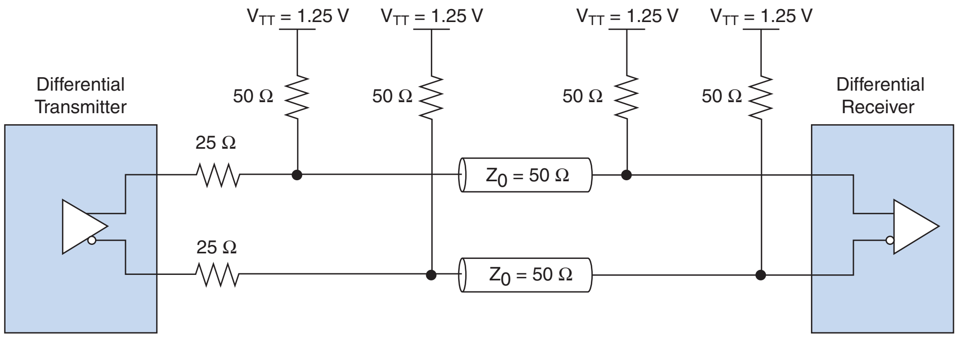


Figure 4–10. Differential SSTL-2 Class II Termination



Differential SSTL-18 Class I & Differential SSTL-18 Class II

The 1.8-V differential SSTL-18 standard is formulated under JEDEC Standard, JESD8-15: Stub Series Terminated Logic for 1.8-V (SSTL_18).

The differential SSTL-18 I/O standard is a 1.8-V standard used for applications such as high-speed DDR2 SDRAM interfaces. This standard supports differential signals in systems using the SSTL-18 standard and supplements the SSTL-18 standard for differential clocks. Stratix II devices support both input and output levels. See Figures 4–11 and 4–12 for details on differential SSTL-18 termination.

Stratix II devices support differential SSTL-18 I/O standards in pseudo-differential mode, which is implemented by using two SSTL-18 single-ended buffers.

The Quartus II software only supports pseudo-differential standards on the `INCLK`, `FBIN` and `EXTCLK` ports of enhanced PLL, as well as on DQS pins when DQS megafunction (ALTDQS, Bidirectional Data Strobe) is used. Two single-ended output buffers are automatically programmed to have opposite polarity so as to implement a pseudo-differential output. A proper V_{REF} voltage is required for the two single-ended input buffers to implement a pseudo-differential input. In this case, only the positive polarity input is used in the speed path while the negative input is not connected internally. In other words, only the non-inverted pin is required to be specified in your design, while the Quartus II software will automatically generate the inverted pin for you.

Although the Quartus II software does not support pseudo-differential SSTL-18 I/O standards on the left and right I/O banks, you can implement these standards at these banks. You need to create two pins in the designs and configure the pins with single-ended SSTL-18 standards. However, this is limited only to pins that support the differential pin-pair I/O function and is dependent on the single-ended 1.8-V HSTL standards support at these banks.

Figure 4–11. Differential SSTL-18 Class I Termination

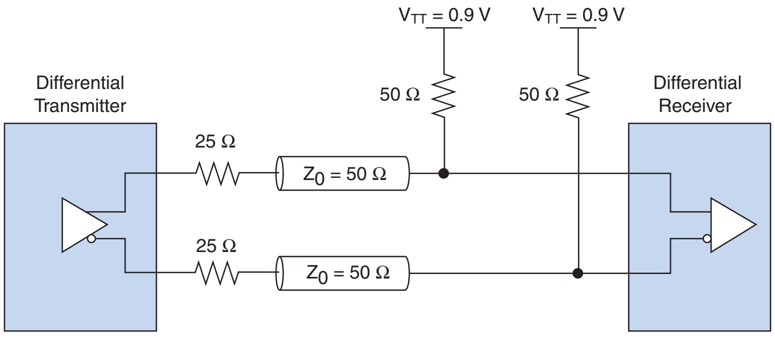
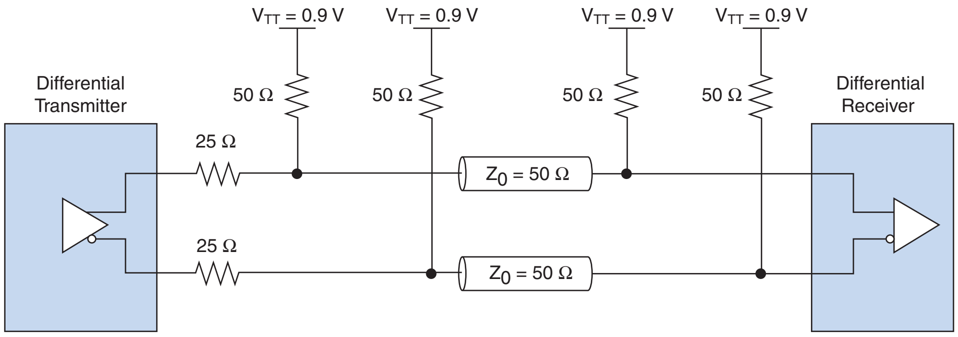


Figure 4–12. Differential SSTL-18 Class II Termination



1.8-V Differential HSTL Class I & 1.8-V Differential HSTL Class II

The 1.8-V differential HSTL specification is the same as the 1.8-V single-ended HSTL specification. It is used for applications designed to operate in the 0.0- to 1.8-V HSTL logic switching range such as QDR memory clock interfaces. Stratix II devices support both input and output levels. See Figures 4–13 and 4–14 for details on 1.8-V differential HSTL termination.

Stratix II devices support 1.8-V differential HSTL I/O standards in pseudo-differential mode, which is implemented by using two 1.8-V HSTL single-ended buffers.

The Quartus II software only supports pseudo-differential standards on the `INCLK`, `FBIN` and `EXTCLK` ports of enhanced PLL, as well as on DQS pins when DQS megafunction (ALTDQS, Bidirectional Data Strobe) is used. Two single-ended output buffers are automatically programmed to have opposite polarity so as to implement a pseudo-differential output. A proper V_{REF} voltage is required for the two single-ended input buffers to implement a pseudo-differential input. In this case, only the positive polarity input is used in the speed path while the negative input is not connected internally. In other words, only the non-inverted pin is required to be specified in your design, while the Quartus II software will automatically generate the inverted pin for you.

Although the Quartus II software does not support 1.8-V pseudo-differential HSTL I/O standards on left/right I/O banks, you can implement these standards at these banks. You need to create two pins in the designs and configure the pins with single-ended 1.8-V HSTL standards. However, this is limited only to pins that support the differential pin-pair I/O function and is dependent on the single-ended 1.8-V HSTL standards support at these banks.

Figure 4–13. 1.8-V Differential HSTL Class I Termination

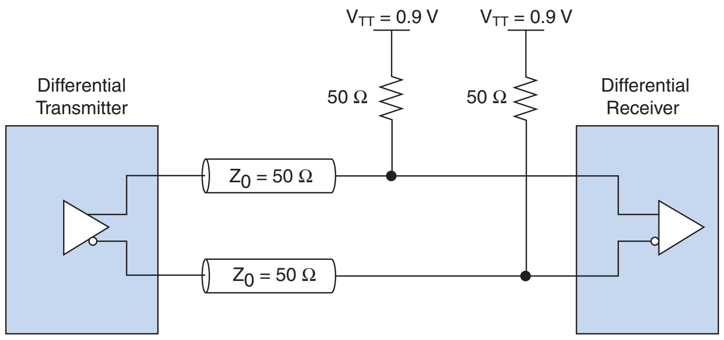
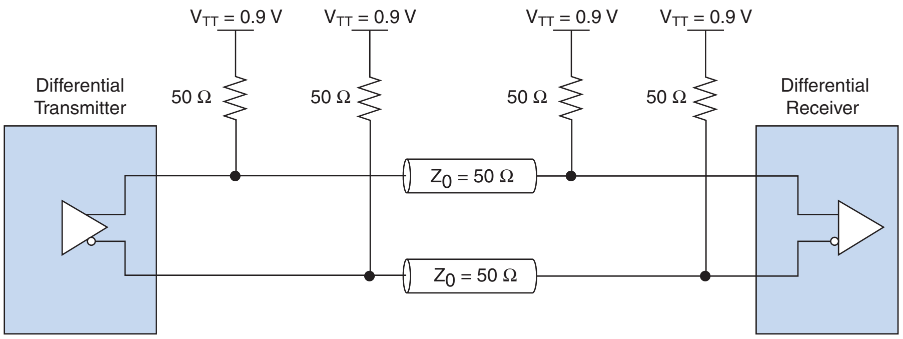


Figure 4–14. 1.8-V Differential HSTL Class II Termination



1.5-V Differential HSTL Class I & 1.5-V Differential HSTL Class II

The 1.5-V differential HSTL standard is formulated under EIA/JEDEC Standard, EIA/JESD8-6: A 1.5-V Output Buffer Supply Voltage Based Interface Standard for Digital Integrated Circuits.

The 1.5-V differential HSTL specification is the same as the 1.5-V single-ended HSTL specification. It is used for applications designed to operate in the 0.0- to 1.5-V HSTL logic switching range, such as QDR memory clock interfaces. Stratix II devices support both input and output levels. See Figures 4–15 and 4–16 for details on the 1.5-V differential HSTL termination.

Stratix II devices support 1.5-V differential HSTL I/O standards in pseudo-differential mode, which is implemented by using two 1.5-V HSTL single-ended buffers.

The Quartus II software only supports pseudo-differential standards on the `INCLK`, `FBIN` and `EXTCLK` ports of enhanced PLL, as well as on DQS pins when DQS megafunction (ALTDQS, Bidirectional Data Strobe) is used. Two single-ended output buffers are automatically programmed to have opposite polarity so as to implement a pseudo-differential output. A proper V_{REF} voltage is required for the two single-ended input buffers to implement a pseudo-differential input. In this case, only the positive polarity input is used in the speed path while the negative input is not connected internally. In other words, only the non-inverted pin is required to be specified in your design, while the Quartus II software will automatically generate the inverted pin for you.

Although the Quartus II software does not support 1.5-V pseudo-differential HSTL I/O standards on left/right I/O banks, you can implement these standards at these banks. You need to create two pins in the designs and configure the pins with single-ended 1.5-V HSTL standards. However, this is limited only to pins that support the differential pin-pair I/O function and is dependent on the single-ended 1.8-V HSTL standards support at these banks.

Figure 4–15. 1.5-V Differential HSTL Class I Termination

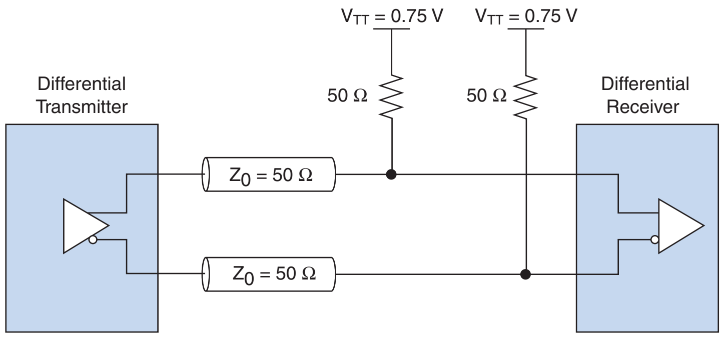
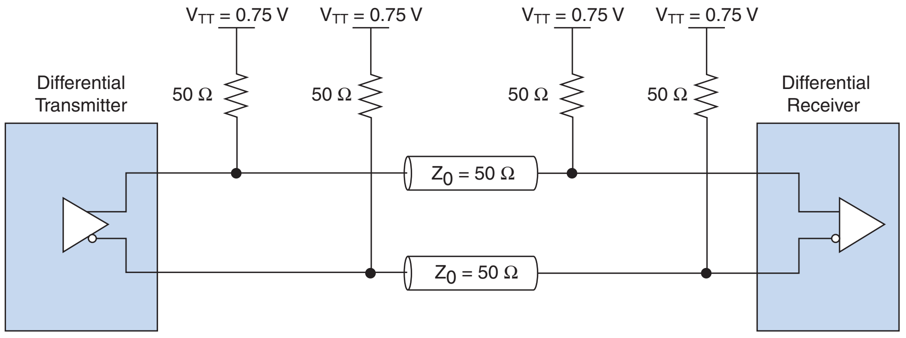


Figure 4–16. 1.5-V Differential HSTL Class II Termination



LVDS

The LVDS standard is formulated under ANSI/TIA/EIA Standard, ANSI/TIA/EIA-644: Electrical Characteristics of Low Voltage Differential Signaling Interface Circuits.

The LVDS I/O standard is a differential high-speed, low-voltage swing, low-power, general-purpose I/O interface standard requiring a 2.5- or 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 megabit per second (Mbps). However, devices can operate at slower speeds if needed, and there is a theoretical maximum of 1.923 Gbps. Stratix II devices are capable of running at a maximum data rate of 1 Gbps and still meet the ANSI/TIA/EIA-644 standard.

Because of the low-voltage swing of the LVDS I/O standard, the electromagnetic interference (EMI) effects are much smaller than complementary metal-oxide semiconductor (CMOS), transistor-to-transistor logic (TTL), and positive (or psuedo) emitter coupled logic (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 II devices provide an optional 100-Ω differential LVDS termination resistor in the device using on-chip differential termination. Stratix II devices support both input and output levels.

Differential LVPECL

The low-voltage positive (or pseudo) emitter coupled logic (LVPECL) 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. Figures 4–17 and 4–18 show two alternate termination schemes for LVPECL. Stratix II devices support both input and output level operations.

Figure 4–17. LVPECL DC Coupled Termination

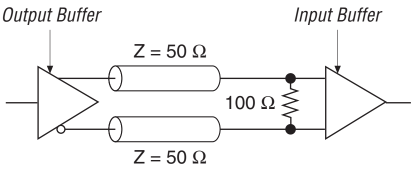
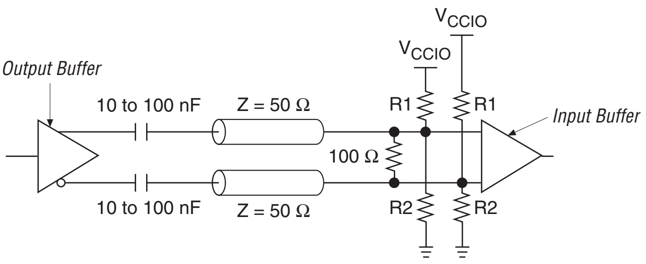


Figure 4–18. LVPECL AC Coupled Termination



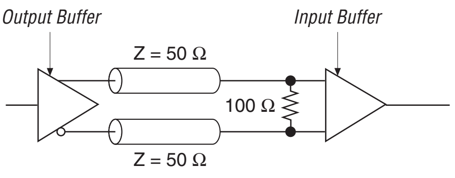
HyperTransport Technology

The HyperTransport standard is formulated by the HyperTransport Consortium.

The HyperTransport I/O standard is a differential high-speed, high-performance I/O interface standard requiring a 2.5- or 3.3-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 I/O standard is a point-to-point standard in which each HyperTransport bus consists of two point-to-point unidirectional links. Each link is 2 to 32 bits.

The HyperTransport 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. Figure 4–19 shows HyperTransport termination. Stratix II devices include an optional 100-Ω differential HyperTransport termination resistor in the device using on-chip differential termination. Stratix II devices support both input and output level operations.

Figure 4–19. HyperTransport Termination



Stratix II External Memory Interface

The increasing demand for higher-performance data processing systems often requires memory-intensive applications. Stratix II devices can interface with many types of external memory. Table 4–3 shows the external memory interface standards supported in Stratix II devices.

Table 4–3. External Memory Interface Support In Stratix II Devices

Memory Interface Standard	I/O Standard	Performance Target *(1)*
DDR SDRAM	SSTL-2 class I and II	200 MHz / 400 Mbps
DDR2 SDRAM	SSTL-18 class I and II	267 MHz / 533 Mbps
RLDRAM II	1.5-/1.8-V HSTL class I and II	300 MHz / 600 Mbps
QDRII SRAM	1.5-V HSTL class II/ 1.8-V HSTL class I and II	250 MHz / 1,000 Mbps
SDR SDRAM	LVTTL	200 MHz / 200 Mbps

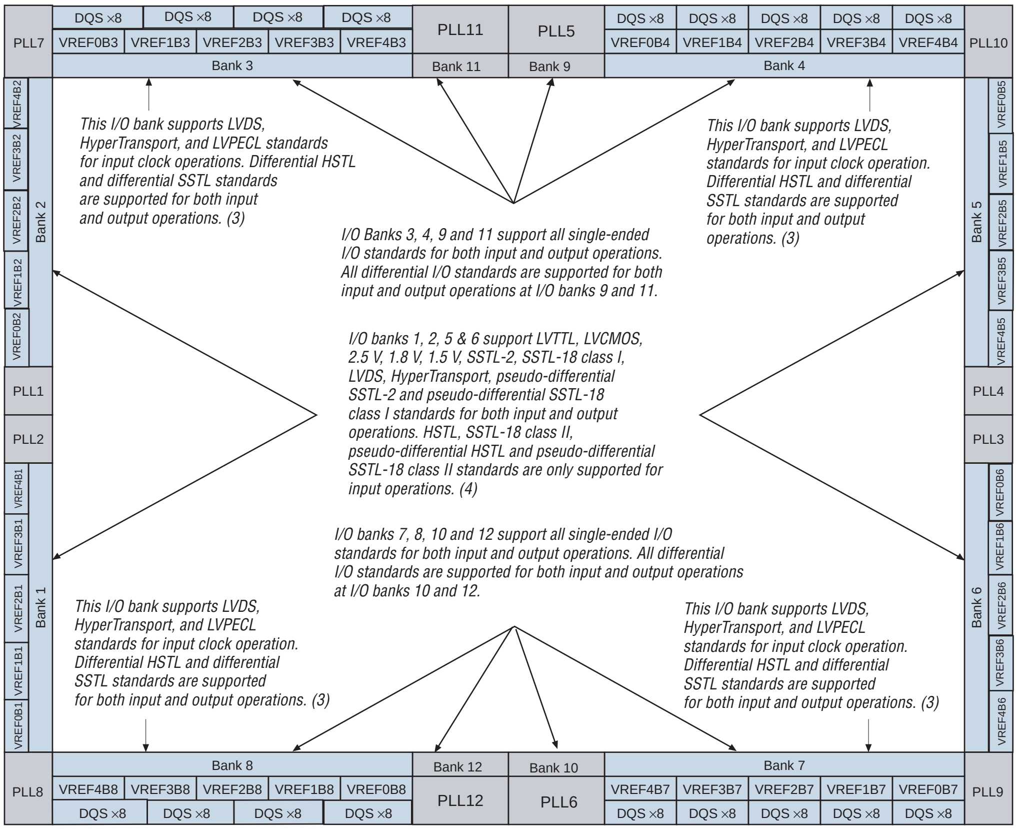
Note to Table 4–3:
(1) These numbers are preliminary.

See Chapter 3, External Memory Interfaces in Volume 2 of the *Stratix II Device Handbook* for more information on the external memory interface support in Stratix II devices.

Stratix II I/O Banks

Stratix II devices have eight general I/O banks and four enhanced phase-locked loop (PLL) external clock output banks, as shown in Figure 4–20. I/O banks 1, 2, 5, and 6 are on the left or right sides of the device and I/O banks 3, 4, and 7 through 12 are at the top or bottom of the device.

Figure 4–20. Stratix II I/O Banks *Notes (1), (2)*



PLL7
DQS ×8
VREF0B3
VREF1B3
VREF2B3
VREF3B3
VREF4B3
Bank 3
PLL11
Bank 11
PLL5
Bank 9
VREF0B4
VREF1B4
VREF2B4
VREF3B4
VREF4B4
Bank 4
PLL10
Bank 2
VREF4B2
VREF3B2
VREF2B2
VREF1B2
VREF0B2
Bank 5
VREF0B5
VREF1B5
VREF2B5
VREF3B5
VREF4B5
PLL1
PLL2
PLL4
PLL3
Bank 1
VREF4B1
VREF3B1
VREF2B1
VREF1B1
VREF0B1
Bank 6
VREF0B6
VREF1B6
VREF2B6
VREF3B6
VREF4B6
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operations. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
I/O Banks 3, 4, 9 and 11 support all single-ended I/O standards for both input and output operations. All differential I/O standards are supported for both input and output operations at I/O banks 9 and 11.
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operation. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
I/O banks 1, 2, 5 & 6 support LVTTL, LVCMOS, 2.5 V, 1.8 V, 1.5 V, SSTL-2, SSTL-18 class I, LVDS, HyperTransport, pseudo-differential SSTL-2 and pseudo-differential SSTL-18 class I standards for both input and output operations. HSTL, SSTL-18 class II, pseudo-differential HSTL and pseudo-differential SSTL-18 class II standards are only supported for input operations. (4)
I/O banks 7, 8, 10 and 12 support all single-ended I/O standards for both input and output operations. All differential I/O standards are supported for both input and output operations at I/O banks 10 and 12.
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operation. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operation. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
PLL8
Bank 8
VREF4B8
VREF3B8
VREF2B8
VREF1B8
VREF0B8
Bank 12
PLL12
Bank 10
PLL6
Bank 7
VREF4B7
VREF3B7
VREF2B7
VREF1B7
VREF0B7
PLL9

Notes to Figure 4–20:

(1) Figure 4–20 is a top view of the silicon die that corresponds to a reverse view for flip-chip packages. It is a graphical representation only. See the pin list and Quartus II software for exact locations.

(2) Depending on the size of the device, different device members have different numbers of V_{REF} groups.

(3) Differential HSTL and differential SSTL standards are available for bidirectional operations on DQS pin and input-only operations on PLL clock input pins; LVDS, LVPECL, and HyperTransport standards are available for input-only operations on PLL clock input pins. See the "Differential I/O Standards" section for more details.

(4) Quartus II software does not support differential SSTL and differential HSTL standards at left/right I/O banks. See the "Differential I/O Standards" section if you need to implement these standards at these I/O banks.

Programmable I/O Standards

Stratix II device programmable I/O standards deliver high-speed and high-performance solutions in many complex design systems. This section discusses the I/O standard support in the I/O banks of Stratix II devices.

Regular I/O Pins

Most Stratix II device pins are multi-function pins. These pins support regular inputs and outputs as their primary function, and offer an optional function such as DQS, differential pin-pair, or PLL external clock outputs. For example, a multi-function pin in the enhanced PLL external clock output bank may be configured as a PLL external clock output when it is not used as a regular I/O pin.

Table 4–4 shows the I/O standards supported when a pin is used as a regular I/O pin in the I/O banks of Stratix II devices.

Table 4–4. Stratix II Regular I/O Standards Support (Part 1 of 2)

I/O Standard	General I/O Bank								Enhanced PLL External Clock Output Bank *(1)*			
	1	2	3	4	5	6	7	8	9	10	11	12
LVTTL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LVCMOS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
1.8 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
1.5 V	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3.3-V PCI			✓	✓			✓	✓	✓	✓	✓	✓
3.3-V PCI-X			✓	✓			✓	✓	✓	✓	✓	✓
SSTL-2 class I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SSTL-2 class II	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SSTL-18 class I	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SSTL-18 class II	*(2)*	*(2)*	✓	✓	*(2)*	*(2)*	✓	✓	✓	✓	✓	✓
1.8-V HSTL class I	*(2)*	*(2)*	✓	✓	*(2)*	*(2)*	✓	✓	✓	✓	✓	✓
1.8-V HSTL class II	*(2)*	*(2)*	✓	✓	*(2)*	*(2)*	✓	✓	✓	✓	✓	✓
1.5-V HSTL class I	*(2)*	*(2)*	✓	✓	*(2)*	*(2)*	✓	✓	✓	✓	✓	✓
1.5-V HSTL class II	*(2)*	*(2)*	✓	✓	*(2)*	*(2)*	✓	✓	✓	✓	✓	✓
Differential SSTL-2 class I	*(3)*	*(3)*	*(4)*	*(4)*	*(3)*	*(3)*	*(4)*	*(4)*				
Differential SSTL-2 class II	*(3)*	*(3)*	*(4)*	*(4)*	*(3)*	*(3)*	*(4)*	*(4)*				
Differential SSTL-18 class I	*(3)*	*(3)*	*(4)*	*(4)*	*(3)*	*(3)*	*(4)*	*(4)*				
Differential SSTL-18 class II	*(5)*	*(5)*	*(4)*	*(4)*	*(5)*	*(5)*	*(4)*	*(4)*				
1.8-V differential HSTL class I	*(5)*	*(5)*	*(4)*	*(4)*	*(5)*	*(5)*	*(4)*	*(4)*				
1.8-V differential HSTL class II	*(5)*	*(5)*	*(4)*	*(4)*	*(5)*	*(5)*	*(4)*	*(4)*				

Table 4–4. Stratix II Regular I/O Standards Support (Part 2 of 2)

I/O Standard	General I/O Bank								Enhanced PLL External Clock Output Bank *(1)*			
	1	2	3	4	5	6	7	8	9	10	11	12
1.5-V differential HSTL class I	*(5)*	*(5)*	*(4)*	*(4)*	*(5)*	*(5)*	*(4)*	*(4)*				
1.5-V differential HSTL class II	*(5)*	*(5)*	*(4)*	*(4)*	*(5)*	*(5)*	*(4)*	*(4)*				
LVDS	✓	✓	*(6)*	*(6)*	✓	✓	*(6)*	*(6)*	✓	✓	✓	✓
HyperTransport technology	✓	✓	*(6)*	*(6)*	✓	✓	*(6)*	*(6)*	✓	✓	✓	✓
Differential LVPECL			*(6)*	*(6)*			*(6)*	*(6)*	✓	✓	✓	✓

Notes to Table 4–4:
(1) A mixture of single-ended and differential I/O standards is not allowed in enhanced PLL external clock output bank.
(2) This I/O standard is only supported for the input operation in this I/O bank.
(3) This I/O standard is supported for both input and output operations in this I/O bank. See the "Differential I/O Standards" section for details.
(4) This I/O standard is supported for both input and output operations for pins that support the DQS function. See the "Differential I/O Standards" section for details.
(5) This I/O standard is only supported for the input operation in this I/O bank. See the "Differential I/O Standards" section for details.
(6) This I/O standard is only supported for the input operation for pins that support PLL `INCLK` function in this I/O bank.

Clock I/O Pins

The PLL clock I/O pins consist of clock inputs (`INCLK`), external feedback inputs (`FBIN`), and external clock outputs (`EXTCLK`). Clock inputs are located at the left and right I/O banks (banks 1, 2, 5, and 6) to support fast PLLs, and at the top and bottom I/O banks (banks 3, 4, 7, and 8) to support enhanced PLLs. Both external clock outputs and external feedback inputs are located at enhanced PLL external clock output banks (banks 9, 10, 11, and 12) to support enhanced PLLs. Table 4–5 shows the PLL clock I/O support in the I/O banks of Stratix II devices.

Table 4–5. I/O Standards Supported for Stratix II PLL Pins (Part 1 of 2)

I/O Standard	Enhanced PLL *(1)*			Fast PLL
	Input		Output	Input
	INCLK	FBIN	EXTCLK	INCLK
LVTTL	✓	✓	✓	✓
LVCMOS	✓	✓	✓	✓
2.5 V	✓	✓	✓	✓

Table 4–5. I/O Standards Supported for Stratix II PLL Pins (Part 2 of 2)

I/O Standard	Enhanced PLL (1)			Fast PLL
	Input		Output	Input
	INCLK	FBIN	EXTCLK	INCLK
1.8 V	✓	✓	✓	✓
1.5 V	✓	✓	✓	✓
3.3-V PCI	✓	✓	✓	
3.3-V PCI-X	✓	✓	✓	
SSTL-2 class I	✓	✓	✓	✓
SSTL-2 class II	✓	✓	✓	✓
SSTL-18 class I	✓	✓	✓	✓
SSTL-18 class II	✓	✓	✓	✓
1.8-V HSTL class I	✓	✓	✓	✓
1.8-V HSTL class II	✓	✓	✓	✓
1.5-V HSTL class I	✓	✓	✓	✓
1.5-V HSTL class II	✓	✓	✓	✓
Differential SSTL-2 class I	✓	✓	✓	
Differential SSTL-2 class II	✓	✓	✓	
Differential SSTL-18 class I	✓	✓	✓	
Differential SSTL-18 class II	✓	✓	✓	
1.8-V differential HSTL class I	✓	✓	✓	
1.8-V differential HSTL class II	✓	✓	✓	
1.5-V differential HSTL class I	✓	✓	✓	
1.5-V differential HSTL class II	✓	✓	✓	
LVDS	✓	✓	✓	✓
HyperTransport technology	✓	✓	✓	✓
Differential LVPECL	✓	✓	✓	

Note to Table 4–5:

(1) The enhanced PLL external clock output bank does not allow a mixture of both single-ended and differential I/O standards.

For more information, see the *PLLs in Stratix II Devices* chapter in Volume 2 of the *Stratix II Device Handbook*

Voltage Levels

Stratix II device specify a range of allowed voltage levels for supported I/O standards. Table 4–6 shows only typical values for input and output V_{CCIO}, V_{REF}, as well as the board V_{TT}.

Table 4–6. Stratix II I/O Standards & Voltage Levels (Part 1 of 2) *Note (1)*

I/O Standard	V_{CCIO} (V)				Input V_{REF} (V)	Termination V_{TT} (V)
	Input Operation		Output Operation			
	Top & Bottom I/O Banks	Left & Right I/O Banks	Top & Bottom I/O Banks	Left & Right I/O Banks		
LVTTL	3.3/2.5	3.3/2.5	3.3	3.3	NA	NA
LVCMOS	3.3/2.5	3.3/2.5	3.3	3.3	NA	NA
2.5 V	3.3/2.5	3.3/2.5	2.5	2.5	NA	NA
1.8 V	1.8/1.5	1.8/1.5	1.8	1.8	NA	NA
1.5 V	1.8/1.5	1.8/1.5	1.5	1.5	NA	NA
3.3-V PCI	3.3	NA	3.3	NA	NA	NA
3.3-V PCI-X	3.3	NA	3.3	NA	NA	NA
SSTL-2 class I	2.5	2.5	2.5	2.5	1.25	1.25
SSTL-2 class II	2.5	2.5	2.5	2.5	1.25	1.25
SSTL-18 class I	1.8	1.8	1.8	1.8	0.90	0.90
SSTL-18 class II	1.8	1.8	1.8	NA	0.90	0.90
1.8-V HSTL class I	1.8	1.8	1.8	NA	0.90	0.90
1.8-V HSTL class II	1.8	1.8	1.8	NA	0.90	0.90
1.5-V HSTL class I	1.5	1.5	1.5	NA	0.75	0.75
1.5-V HSTL class II	1.5	1.5	1.5	NA	0.75	0.75
Differential SSTL-2 class I	2.5	2.5	2.5	2.5	1.25	1.25
Differential SSTL-2 class II	2.5	2.5	2.5	2.5	1.25	1.25
Differential SSTL-18 class I	1.8	1.8	1.8	1.8	0.90	0.90
Differential SSTL-18 class II	1.8	1.8	1.8	NA	0.90	0.90
1.8-V differential HSTL class I	1.8	1.8	1.8	NA	0.90	0.90
1.8-V differential HSTL class II	1.8	1.8	1.8	NA	0.90	0.90

Table 4–6. Stratix II I/O Standards & Voltage Levels (Part 2 of 2) Note (1)

I/O Standard	V_{CCIO} (V)				Input V_{REF} (V)	Termination V_{TT} (V)
	Input Operation		Output Operation			
	Top & Bottom I/O Banks	Left & Right I/O Banks	Top & Bottom I/O Banks	Left & Right I/O Banks		
1.5-V differential HSTL class I	1.5	1.5	1.5	NA	0.75	0.75
1.5-V differential HSTL class II	1.5	1.5	1.5	NA	0.75	0.75
LVDS (2)	3.3/2.5/1.8/1.5	2.5	3.3	2.5	NA	NA
HyperTransport technology (2)	3.3/2.5/1.8/1.5	2.5	3.3	2.5	NA	NA
Differential LVPECL (2)	3.3/2.5/1.8/1.5	NA	3.3	NA	NA	NA

Notes to Table 4–6:

(1) Any input pins with PCI-clamping-diode enabled force the V_{CCIO} to 3.3 V.

(2) LVDS, HyperTransport, and LVPECL output operation in the top and bottom banks is only supported in PLL banks 9-12. The V_{CCIO} level for differential output operation in the PLL banks is 3.3 V. The V_{CCIO} level for output operation in the left and right I/O banks is 2.5 V.

See the *DC and Switching Characteristics* chapter in Volume 1 of the *Stratix II Device Handbook* for detailed electrical characteristics of each I/O standard.

On-Chip Termination

Stratix II devices feature on-chip termination to provide I/O impedance matching and termination capabilities. Apart from maintaining signal integrity, this feature also minimizes the need for external resistor networks, thereby saving board space and reducing costs.

Stratix II devices support on-chip series termination (R_S) for single-ended I/O standards and on-chip differential termination (R_D) for differential I/O standards. This section discusses the on-chip series termination support.

For more information on differential on-chip termination, see Chapter 5, High-Speed Differential I/O Interfaces with DPA in Stratix II Devices in Volume 2 of the *Stratix II Device Handbook.*

The Stratix II device family supports I/O driver on-chip series termination (R_S) through drive strength control for single-ended I/Os. There are two ways to implement the R_S in Stratix II devices:

- R_S without calibration for both row I/Os and column I/Os
- R_S with calibration only for column I/Os

On-Chip Series Termination without Calibration

Stratix II devices support driver impedance matching to provide the I/O driver with controlled output impedance that closely matches the impedance of the transmission line. As a result, reflections can be significantly reduced. Stratix II devices support on-chip series termination for single-ended I/O standards as shown in Figure 4–21. The R_S shown in Figure 4–21 is the intrinsic impedance of transistors. The typical R_S values are 25Ω and 50Ω. Once matching impedance is selected, current drive strength is no longer selectable. Table 4–7 shows the list of output standards that support on-chip series termination without calibration.

Figure 4–21. Stratix II On-Chip Series Termination without Calibration

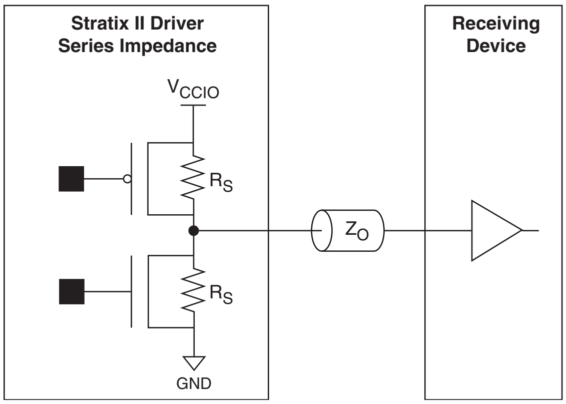


Table 4–7. Selectable I/O Drivers with On-Chip Series Termination without Calibration (Part 1 of 2)

I/O Standard	On-chip Series Termination Setting		
	Row I/O	Column I/O	Unit
3.3V LVTTL	50	50	Ω
	25	25	Ω
3.3V LVCMOS	50	50	Ω
	25	25	Ω

Table 4–7. Selectable I/O Drivers with On-Chip Series Termination without Calibration (Part 2 of 2)

I/O Standard	On-chip Series Termination Setting		
	Row I/O	Column I/O	Unit
2.5V LVTTL	50	50	Ω
	25	25	Ω
2.5V LVCMOS	50	50	Ω
	25	25	Ω
1.8V LVTTL	50	50	Ω
		25	Ω
1.8V LVCMOS	50	50	Ω
		25	Ω
1.5V LVTTL		50	Ω
1.5V LVCMOS		50	Ω
2.5V SSTL Class I	50	50	Ω
2.5V SSTL Class II	25	25	Ω
1.8V SSTL Class i	50	50	Ω
1.8V SSTL Class II		25	Ω
1.8V HSTL Class I	50	50	Ω
1.8V HSTL Class II		25	Ω
1.5V HSTL Class I		50	Ω

To use on-chip termination for the SSTL Class 1 standard, users should select the 50-Ω on-chip series termination setting for replacing the external 25-Ω R_S (to match the 50-Ω transmission line). For the SSTL Class 2 standard, users should select the 25-Ω on-chip series termination setting (to match the 50-Ω transmission line and the near end 50-Ω pull-up to V_{TT}).

For more information on tolerance specifications for on-chip termination without calibration, refer to the "On-Chip Termination Specifications" section in the *DC & Switching Characteristics* chapter in Volume 1 of the *Stratix II Device Handbook*.

On-Chip Series Termination with Calibration

Stratix II devices support on-chip series termination with calibration in column I/Os in top and bottom banks. Every column I/O buffer consists of a group of transistors in parallel. Each transistor can be individually enabled or disabled. The on-chip series termination calibration circuit compares the total impedance of the transistor group to the external 25-Ω or 50-Ω resistors connected to the `RUP` and `RDN` pins, and dynamically enables or disables the transistors until they match (as shown in Figure 4–22). The R_S shown in Figure 4–22 is the intrinsic impedance of transistors. Calibration happens at the end of device configuration. Once the calibration circuit finds the correct impedance, it powers down and stops changing the characteristics of the drivers.

Figure 4–22. Stratix II On-Chip Series Termination with Calibration

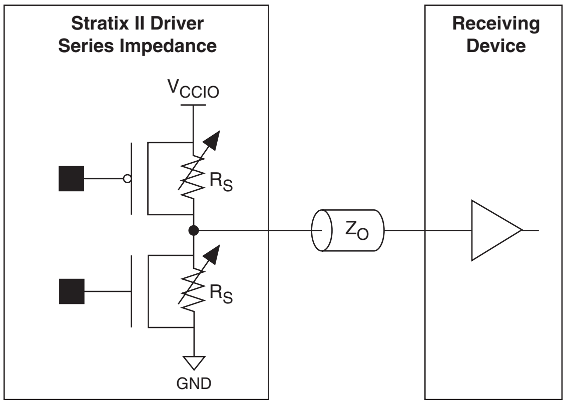


Table 4–8 shows the list of output standards that support on-chip series termination with calibration.

Table 4–8. Selectable I/O Drivers with On-Chip Series Termination with Calibration (Part 1 of 2)

I/O Standard	On-chip Series Termination Setting (Column I/O)	Unit
3.3V LVTTL	50	Ω
	25	Ω
3.3V LVCMOS	50	Ω
	25	Ω
2.5V LVTTL	50	Ω
	25	Ω
2.5V LVCMOS	50	Ω
	25	Ω

Table 4–8. Selectable I/O Drivers with On-Chip Series Termination with Calibration (Part 2 of 2)

I/O Standard	On-chip Series Termination Setting (Column I/O)	Unit
1.8V LVTTL	50	Ω
	25	Ω
1.8V LVCMOS	50	Ω
	25	Ω
1.5 LVTTL	50	Ω
1.5 LVCMOS	50	Ω
2.5V SSTL Class I	50	Ω
2.5V SSTL Class II	25	Ω
1.8V SSTL Class I	50	Ω
1.8V SSTL Class II	25	Ω
1.8V HSTL Class I	50	Ω
1.8V HSTL Class II	25	Ω
1.5V HSTL Class I	50	Ω

There are two separate sets of calibration circuits in the Stratix II devices:

- One calibration circuit for top banks 3 and 4
- One calibration circuit for bottom banks 7 and 8

Calibration circuits rely on the external pull-up reference resistor (R_{UP}) and pull-down reference resistor (R_{DN}) to achieve accurate on-chip series termination. There is one pair of `RUP` and `RDN` pins in bank 4 for the calibration circuit for top I/O banks 3 and 4. Similarly, there is one pair of `RUP` and `RDN` pins in bank 7 for the calibration circuit for bottom I/O banks 7 and 8. Since two banks share the same calibration circuitry, they must have the same V_{CCIO} voltage if both banks enable on-chip series termination with calibration. If banks 3 and 4 have different V_{CCIO} voltages, only bank 4 can enable on-chip series termination with calibration because the `RUP` and `RDN` pins are located in bank 4. Bank 3 still can use on-chip series termination, but without calibration. The same rule applies to banks 7 and 8.

The `RUP` and `RDN` pins are dual-purpose I/Os, which means they can be used as regular I/Os if the calibration circuit is not used. When used for calibration, the `RUP` pin is connected to V_{CCIO} through an external 25-Ω or 50-Ω resistor for an on-chip series termination value of 25Ω or 50Ω, respectively. The RDN pin is connected to GND through an external 25-Ω or 50-Ω resistor for an on-chip series termination value of 25Ω or 50Ω, respectively.

For more information on tolerance specifications for on-chip termination with calibration, refer to the "On-Chip Termination Specifications" section in the *DC & Switching Characteristics* chapter in Volume 1 of the *Stratix II Device Handbook*.

Design Considerations

While Stratix II devices feature various I/O capabilities for high-performance and high-speed system designs, there are several other considerations that require attention to ensure the success of those designs.

I/O Termination

I/O termination requirements for single-ended and differential I/O standards are discussed in this section.

Single-Ended I/O Standards

Although single-ended, non-voltage-referenced I/O standards do not require termination, impedance matching is necessary to reduce reflections and improve signal integrity.

Voltage-referenced I/O standards require both an input reference voltage, $V_{REF,}$ and a termination voltage, V_{TT}. The reference voltage of the receiving device tracks the termination voltage of the transmitting device. Each voltage-referenced I/O standard requires a unique termination setup. For example, a proper resistive signal termination scheme is critical in SSTL standards to produce a reliable DDR memory system with superior noise margin.

Stratix II on-chip series termination provides the convenience of no external components. External pull-up resistors can be used to terminate the voltage-referenced I/O standards such as SSTL-2 and HSTL.

See the "Stratix II I/O Standards Support" section for more information on the termination scheme of various single-ended I/O standards.

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 II devices provide an optional differential on-chip resistor when using LVDS and HyperTransport standards.

I/O Banks Restrictions

Each I/O bank can simultaneously support multiple I/O standards. The following sections provide guidelines for mixing non-voltage-referenced and voltage-referenced I/O standards in Stratix II devices.

Non-Voltage-Referenced Standards

Each Stratix II device I/O bank has its own V_{CCIO} pins and supports only one V_{CCIO}, either 1.5, 1.8, 2.5, or 3.3 V. An I/O bank can simultaneously support any number of input signals with different I/O standard assignments, as shown in Table 4–9.

For output signals, a single I/O bank supports non-voltage-referenced output signals that are driving at the same voltage as V_{CCIO}. Since an I/O bank can only have one V_{CCIO} value, 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 standard inputs and outputs and 3.3-V LVCMOS inputs (not output or bidirectional pins).

Table 4–9. Acceptable Input Levels for LVTTL & LVCMOS

Bank V_{CCIO} (V)	Acceptable Input Levels (V)			
	3.3	2.5	1.8	1.5
3.3	✓	✓ *(1)*		
2.5	✓	✓		
1.8	✓ *(2)*	✓ *(2)*	✓	✓ *(1)*
1.5	✓ *(2)*	✓ *(2)*	✓	✓

Notes to Table 4–9:

(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.

Voltage-Referenced Standards

To accommodate voltage-referenced I/O standards, each Stratix II device's I/O bank supports multiple V_{REF} pins feeding a common V_{REF} bus. The number of available V_{REF} pins increases as device density increases. If these pins are not used as V_{REF} pins, they cannot be used as generic I/O pins. However, each bank can only have a single V_{CCIO} voltage level and a single V_{REF} voltage level at a given time.

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.

Because of performance reasons, voltage-referenced input standards use their own V_{CCIO} level as the power source. For example, you can only place 1.5-V HSTL input pins in an I/O bank with a 1.5-V V_{CCIO}. See the "Stratix II I/O Banks" section for details on input V_{CCIO} for voltage-referenced standards.

Voltage-referenced bidirectional and output signals must be the same as the I/O bank's V_{CCIO} voltage. For example, you can only place SSTL-2 output pins in an I/O bank with a 2.5-V V_{CCIO}.

See the "I/O Placement Guidelines" section for details on voltage-referenced I/O standards placement.

Mixing Voltage-Referenced and Non-Voltage-Referenced Standards

An I/O bank can support both non-voltage-referenced and voltage-referenced pins by applying each of the rule sets individually. For example, an I/O bank can support SSTL-18 inputs and 1.8-V inputs and outputs with a 1.8-V V_{CCIO} and a 0.9-V V_{REF}. Similarly, an I/O bank can support 1.5-V standards, 2.5-V (inputs, but not outputs), and HSTL I/O standards with a 1.5-V V_{CCIO} and 0.75-V V_{REF}.

I/O Placement Guidelines

The I/O placement guidelines help to reduce noise issues that may be associated with a design such that Stratix II FPGAs can maintain an acceptable noise level on the V_{CCIO} supply. Because Stratix II devices require each bank to be powered separately for V_{CCIO}, these noise issues have no effect when crossing bank boundaries and, as such, these rules need not be applied.

This section provides I/O placement guidelines for the programmable I/O standards supported by Stratix II devices and includes essential information for designing systems using their devices' selectable I/O capabilities.

V_{REF} Pin Placement Restrictions

There are at least two dedicated V_{REF} pins per I/O bank to drive the V_{REF} bus. Larger Stratix II devices have more V_{REF} pins per I/O bank. There are at most 40 pins in an I/O bank per one V_{REF} pin, which is also known as a "V_{REF} group." Since Stratix II devices offer only flip-chip packages, a V_{REF} pin does not take a pad's place in layout.

There are limits to the number of pins that a V_{REF} pin can support. For example, each output pin adds some noise to the V_{REF} level and an excessive number of outputs make the level too unstable to be used for incoming signals.

Restrictions on the placement of single-ended voltage-referenced I/O pads with respect to V_{REF} pins help maintain an acceptable noise level on the V_{CCIO} supply and prevent output switching noise from shifting the V_{REF} rail.

Input Pins
Each V_{REF} pin supports a maximum of 40 input pads.

Output Pins
When a voltage-referenced input or bidirectional pad does not exist in a bank, the number of output pads that can be used in that bank depends on the total number of available pads in that same bank. However, when a voltage-referenced input exists, a design can use up to 20 output pads per V_{REF} pin in a bank.

Bidirectional Pins

Bidirectional pads must satisfy both input and output guidelines simultaneously. The general formulas for input and output rules are shown in Table 4–10.

Table 4–10. Bidirectional Pin Limitation Formulas

Rules	Formulas
Input	<Total number of bidirectional pins> + <Total number of V_{REF} input pins, if any> ≤ 40 per V_{REF} pin
Output	<Total number of bidirectional pins> + <Total number of output pins, if any> – <Total number of pins from smallest OE group, if more than one OE groups> ≤ 20 per V_{REF} pin

- If the same output enable (OE) controls all the bidirectional pads (bidirectional pads in the same OE group are driving in and out at the same time) and there are no other outputs or voltage-referenced inputs in the bank, then the voltage-referenced input is never active at the same time as an output. Therefore, the output limitation rule does not apply. However, since the bidirectional pads are linked to the same OE, the bidirectional pads will all act as inputs at the same time. Therefore, there is a limit of 40 input pads, as follows:

  <Total number of bidirectional pins> + <Total number of V_{REF} input pins> ≤ 40 per V_{REF} pin

- If any of the bidirectional pads are controlled by different OE and there are no other outputs or voltage-referenced inputs in the bank, then one group of bidirectional pads can be used as inputs and another group is used as outputs. In such cases, the formula for the output rule is simplified, as follows:

  <Total number of bidirectional pins> – <Total number of pins from smallest OE group> ≤ 20 per V_{REF} pin

- Consider a case where eight bidirectional pads are controlled by OE1, eight bidirectional pads are controlled by OE2, six bidirectional pads are controlled by OE3, and there are no other outputs or voltage-referenced inputs in the bank. While this totals 22 bidirectional pads, it is safely allowable because there would be a possible maximum of 16 outputs per V_{REF} pin, assuming the worst case where OE1 and OE2 are active and OE3 is inactive. This is useful for DDR SDRAM applications.

- When at least one additional voltage-referenced input and no other outputs exist in the same V_{REF} group, the bidirectional pad limitation must simultaneously adhere to the input and output limitations. The input rule becomes:

  <Total number of bidirectional pins> + <Total number of V_{REF} input pins> ≤ 40 per V_{REF} pin

  Whereas the output rule is simplified as:

  <Total number of bidirectional pins> ≤ 20 per V_{REF} pin

- When at least one additional output exists but no voltage-referenced inputs exist, the output rule becomes:

  <Total number of bidirectional pins> + <Total number of output pins> – <Total number of pins from smallest OE group> ≤ 20 per V_{REF} pin

- When additional voltage-referenced inputs and other outputs exist in the same V_{REF} group, then the bidirectional pad limitation must again simultaneously adhere to the input and output limitations. The input rule is:

  <Total number of bidirectional pins> + <Total number of V_{REF} input pins> ≤ 40 per V_{REF} pin

  Whereas the output rule is given as:

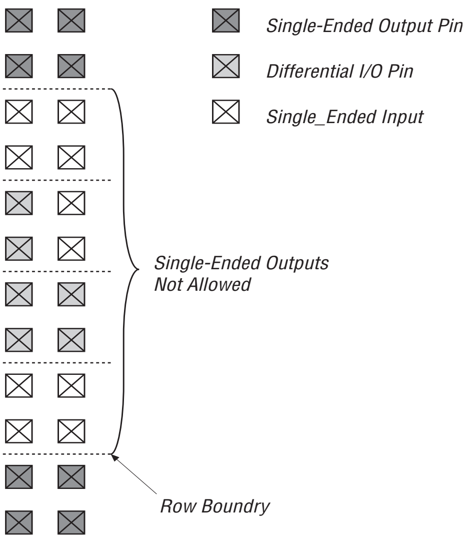
  <Total number of bidirectional pins> + <Total number of output pins> – <Total number of pins from smallest OE group> ≤ 20 per V_{REF} pin

I/O Pin Placement with Respect to High-Speed Differential I/O Pins

Regardless of whether or not the SERDES circuitry is utilized, there is a restriction on the placement of single-ended output pins with respect to high-speed differential I/O pins. As shown in Figure 4–23, all single-ended outputs must be placed at least one LAB row away from the differential I/O pins. There are no restrictions on the placement of single-ended input pins with respect to differential I/O pins. Single-ended input pins may be placed within the same LAB row as differential I/O pins. However, the single-ended input's IOE register is not available. The input must be implemented within the core logic.

This single-ended output pin placement restriction only applies when using the LVDS or HyperTransport I/O standards in the left and right I/O banks. There are no restrictions for single-ended output pin placement with respect to differential clock pins in the top and bottom I/O banks.

Figure 4–23. Single-Ended Output Pin Placement with Respect to Differential I/O Pins



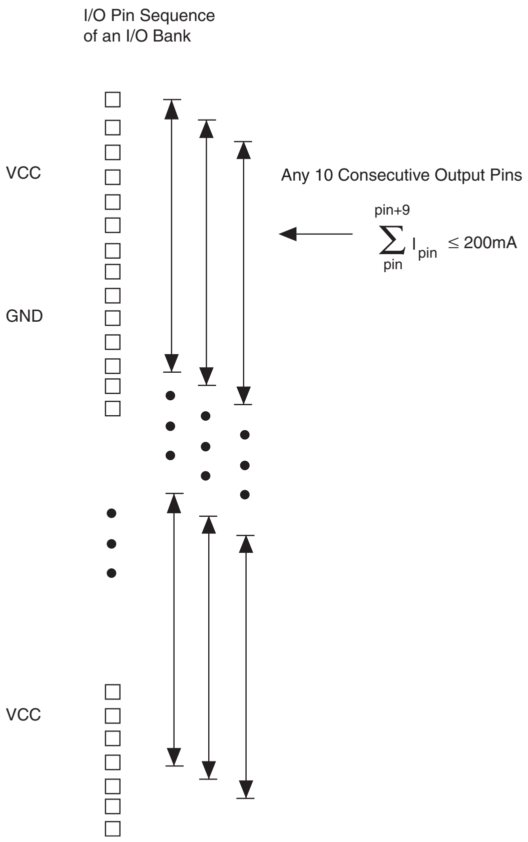
DC Guidelines

Power budgets are essential to ensure the reliability and functionality of a system application. You are often required to perform power dissipation analysis on each device in the system to come out with the total power dissipated in that system, which is comprised of a static component and a dynamic component.

The static power consumption of a device is the total DC current flowing from V_{CCIO} to ground. This is primarily due to diode leakage current or the sub-threshold conductance of the device.

For any ten consecutive pads in an I/O bank of Stratix II devices, Altera recommends a maximum current of 200 mA, as shown in Figure 4–24, because the placement of V_{CCIO}/ground (GND) bumps are regular, 10 I/O pins per pair of power pins. This limit is on the static power consumed by an I/O standard, as shown in Table 4–11. Limiting static power is a way to improve reliability over the lifetime of the device.

Figure 4–24. DC Current Density Restriction *Notes (1), (2)*


I/O Pin Sequence of an I/O Bank
VCC
GND
VCC
Any 10 Consecutive Output Pins
$\sum_{pin}^{pin+9} I_{pin} \leq 200mA$

Notes to Figure 4–24:

(1) The consecutive pads do not cross I/O banks.

(2) V_{REF} pins do not affect DC current calculation because there are no V_{REF} pads.

Table 4–11 shows the I/O standard DC current specification.

Table 4–11. Stratix II I/O Standard DC Current Specification (Part 1 of 2) *Note (1)*

I/O Standard	I_{PIN} (mA), Top & Bottom I/O Banks	I_{PIN} (mA), Left & Right I/O Banks
LVTTL	*(2)*	*(2)*
LVCMOS	*(2)*	*(2)*
2.5 V	*(2)*	*(2)*
1.8 V	*(2)*	*(2)*

Table 4–11. Stratix II I/O Standard DC Current Specification (Part 2 of 2) *Note (1)*

I/O Standard	I_{PIN} (mA), Top & Bottom I/O Banks	I_{PIN} (mA), Left & Right I/O Banks
1.5 V	*(2)*	*(2)*
3.3-V PCI	1.5	NA
3.3-V PCI-X	1.5	NA
SSTL-2 class I	12 *(3)*	12 *(3)*
SSTL-2 class II	24 *(3)*	16 *(3)*
SSTL-18 class I	12 *(3)*	10 *(3)*
SSTL-18 class II	18 *(3)*	NA
1.8-V HSTL class I	12 *(3)*	NA
1.8-V HSTL class II	20 *(3)*	NA
1.5-V HSTL class I	12 *(3)*	NA
1.5-V HSTL class II	20 *(3)*	NA
Differential SSTL-2 class I	8.1	8.1
Differential SSTL-2 class II	16.4	16.4
Differential SSTL-18 class I	6.7	6.7
Differential SSTL-18 class II	NA	NA
1.8-V differential HSTL class I	NA	NA
1.8-V differential HSTL class II	NA	NA
1.5-V differential HSTL class I	NA	NA
1.5-V differential HSTL class II	NA	NA
LVDS	12	12
HyperTransport technology	16	16
Differential LVPECL	10	10

Notes to Table 4–11:

(1) The current value obtained for differential HSTL and differential SSTL standards is per pin and not per differential pair, as opposed to the per-pair current value of LVDS and HyperTransport standards.

(2) The DC power specification of each I/O standard depends on the current sourcing and sinking capabilities of the I/O buffer programmed with that standard, as well as the load being driven. LVTTL, LVCMOS, 2.5-V, 1.8-V, and 1.5-V outputs are not included in the static power calculations because they normally do not have resistor loads in real applications. The voltage swing is rail-to-rail with capacitive load only. There is no DC current in the system.

(3) This I_{PIN} value represents the DC current specification for the default current strength of the I/O standard. The I_{PIN} varies with programmable drive strength and is the same as the drive strength as set in Quartus II software. See the *Stratix II Architecture* chapter in Volume 1 of the *Stratix II Device Handbook* for a detailed description of the programmable drive strength feature of voltage-referenced I/O standards.

Table 4–11 only shows the limit on the static power consumed by an I/O standard. The amount of power used at any moment could be much higher, and is based on the switching activities.

Conclusion

Stratix II devices provide I/O capabilities that allow you to work in compliance with current and emerging I/O standards and requirements. With the Stratix II devices features, such as programmable driver strength, you can reduce board design interface costs and increase the development flexibility.

Further Information

See the following sources for more information:

- *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook*
- The *PLLs in Stratix II Devices chapter* in Volume 2 of *the Stratix II Device Handbook.*
- The *High-Speed Board Layout Guidelines* chapter in Volume 2 of the *Stratix II Device Handbook*

References

See the following references for more information:

- Interface Standard for Nominal 3V / 3.3-V Supply Digital Integrated Circuits, JESD8-B, Electronic Industries Association, September 1999.
- 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.
- 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.
- 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.
- 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.
- Stub Series Terminated Logic for 2.5-V (SSTL-2), JESD8-9A, Electronic Industries Association, December 2000.
- Stub Series Terminated Logic for 1.8 V (SSTL-18), Preliminary JC42.3, Electronic Industries Association.
- 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.

- 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 with DPA in Stratix II Devices

SII52005-2.0

Introduction

The Stratix® II device family offers high-speed differential I/O capabilities to support source-synchronous communication protocols such as HyperTransport™ technology, Rapid I/O, XSBI, and SPI.

Stratix II devices have the following dedicated circuitry for high-speed differential I/O support:

- Differential I/O buffer
- Transmit serializer
- Receive deserializer
- Data realignment circuit
- Dynamic phase aligner (DPA)
- Synchronizer (FIFO buffer)
- Analog PLLs (fast PLLs)

For high-speed differential interfaces, Stratix II devices can accommodate different differential I/O standards such as:

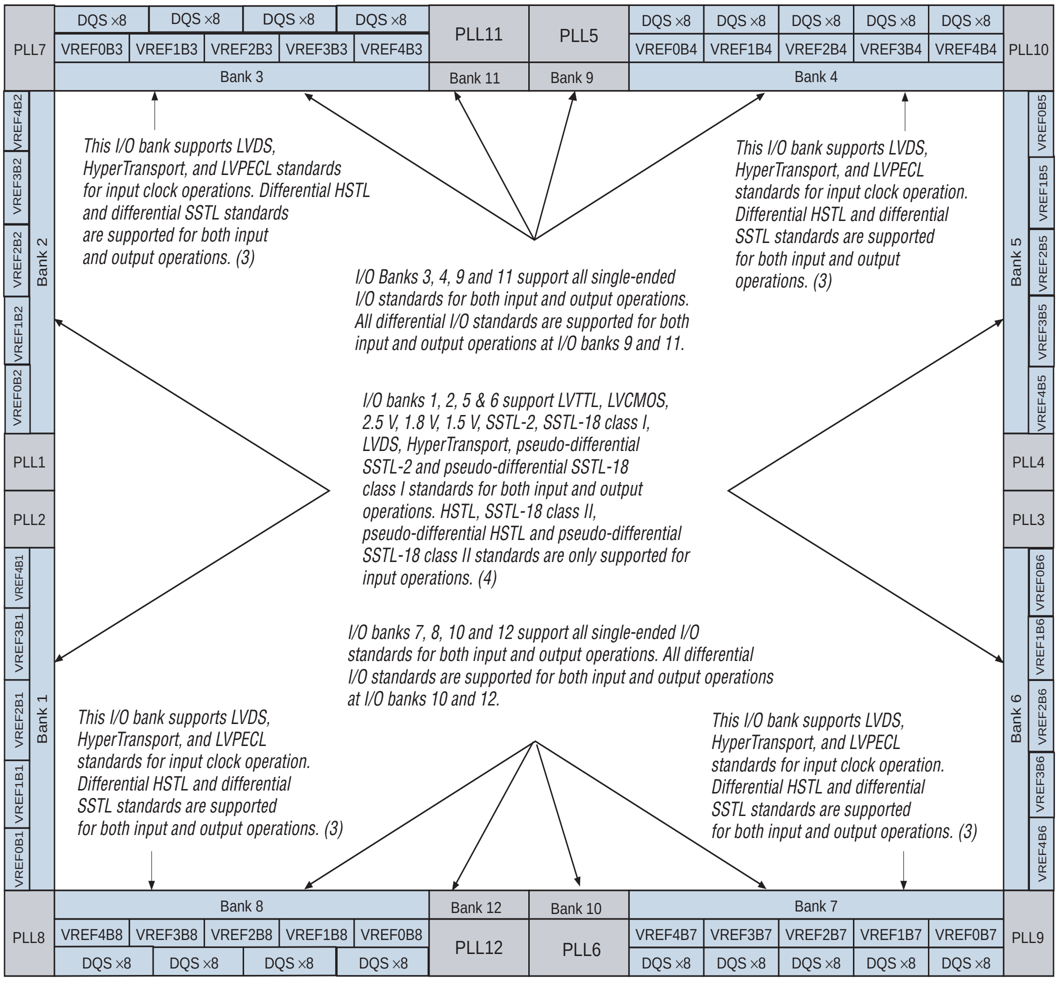
- LVDS
- HyperTransport technology
- HSTL
- SSTL
- LVPECL

☞ HSTL, SSTL, and LVPECL I/O standards can be used only for PLL clock inputs and outputs in differential mode.

I/O Banks

Stratix II inputs and outputs are partitioned into banks located on the periphery of the die. The inputs and outputs that support LVDS and Hypertransport technology are located in four banks, two on the left and two on the right side of the device. LVPECL, HSTL, and SSTL standards are supported on certain top and bottom banks of the die (banks 9 to 12) when used as differential clock inputs/outputs. Differential HSTL and SSTL standards can be supported on banks 3, 4, 7, and 8 if the pins on these banks are used as DQS/DQSn pins. Figure 5–1 shows where the banks and the PLLs are located on the die.

Figure 5–1. I/O Banks Notes (1), (2), (5), (6)



DQS ×8
PLL7
VREF0B3
VREF1B3
VREF2B3
VREF3B3
VREF4B3
Bank 3
PLL11
Bank 11
PLL5
Bank 9
VREF0B4
VREF1B4
VREF2B4
VREF3B4
VREF4B4
Bank 4
PLL10
VREF4B2
VREF3B2
VREF2B2
VREF1B2
VREF0B2
Bank 2
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operations. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
I/O Banks 3, 4, 9 and 11 support all single-ended I/O standards for both input and output operations. All differential I/O standards are supported for both input and output operations at I/O banks 9 and 11.
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operation. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
VREF0B5
VREF1B5
VREF2B5
VREF3B5
VREF4B5
Bank 5
PLL1
PLL2
I/O banks 1, 2, 5 & 6 support LVTTL, LVCMOS, 2.5 V, 1.8 V, 1.5 V, SSTL-2, SSTL-18 class I, LVDS, HyperTransport, pseudo-differential SSTL-2 and pseudo-differential SSTL-18 class I standards for both input and output operations. HSTL, SSTL-18 class II, pseudo-differential HSTL and pseudo-differential SSTL-18 class II standards are only supported for input operations. (4)
PLL4
PLL3
VREF4B1
VREF3B1
VREF2B1
VREF1B1
VREF0B1
Bank 1
I/O banks 7, 8, 10 and 12 support all single-ended I/O standards for both input and output operations. All differential I/O standards are supported for both input and output operations at I/O banks 10 and 12.
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operation. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
This I/O bank supports LVDS, HyperTransport, and LVPECL standards for input clock operation. Differential HSTL and differential SSTL standards are supported for both input and output operations. (3)
VREF0B6
VREF1B6
VREF2B6
VREF3B6
VREF4B6
Bank 6
Bank 8
PLL8
VREF4B8
VREF3B8
VREF2B8
VREF1B8
VREF0B8
Bank 12
PLL12
Bank 10
PLL6
Bank 7
VREF4B7
VREF3B7
VREF2B7
VREF1B7
VREF0B7
PLL9

Notes to Figure 5–1:

(1) Figure 5–1 is a top view of the silicon die that corresponds to a reverse view for flip-chip packages. It is a graphical representation only. See the pin list and Quartus II software for exact locations.

(2) Depending on the size of the device, different device members have different numbers of V_{REF} groups.

(3) Differential HSTL and differential SSTL standards are available for bidirectional operations on DQS pin and input-only operations on PLL clock input pins; LVDS, LVPECL, and HyperTransport standards are available for input-only operations on PLL clock input pins. See the *Selectable I/O Standards in Stratix II Devices* chapter of the *Stratix II Device Handbook*, Volume 2 for more details.

(4) Quartus II software does not support differential SSTL and differential HSTL standards at left/right I/O banks. See the *Selectable I/O Standards in Stratix II Devices* chapter of the *Stratix II Device Handbook*, Volume 2 if you need to implement these standards at these I/O banks.

(5) Banks 11 and 12 are available only in EP2S60, EP2S90, EP2S130, and EP2S180 devices.

(6) PLLs 7, 8, 9 10, 11, and 12 are available only in EP2S60, EP2S90, EP2S130, and EP2S180 devices.

Table 5–1 lists the differential I/O standards supported by each bank.

Table 5–1. Supported Differential I/O Types

Bank	Row I/O (Banks 1, 2, 5 & 6)			Column I/O (Banks, 3, 4 & 7 through 12)		
Type	Clock Inputs	Clock Outputs	Data or Regular I/O Pins	Clock Inputs	Clock Outputs	Data or Regular I/O Pins
Differential HSTL				✓	✓	*(1)*
Differential SSTL				✓	✓	*(1)*
LVPECL				✓	✓	
LVDS	✓	✓	✓	✓	✓	
HyperTransport technology	✓	✓	✓	✓	✓	

Note to Table 5–1:
(1) Used as both inputs and outputs on the `DQS/DQSn` pins.

Table 5–2 shows the total number of differential channels available in Stratix II devices. The available channels are divided evenly between the left and right banks of the die. Non-dedicated clocks in the left and right banks can also be used as data receiver channels. The total number of receiver channels includes these four non-dedicated clock channels. Pin migration is available for different size devices in the same package.

Table 5–2. Differential Channels in Stratix II Devices *Notes (1), (2), (3)*

Device	484-Pin FineLine BGA	484-Pin Hybrid FineLine BGA	672-Pin FineLine BGA	780-Pin FineLine BGA	1,020-Pin FineLine BGA	1,508-Pin FineLine BGA *(3)*
EP2S15	38 transmitters 42 receivers		38 transmitters 42 receivers			
EP2S30	38 transmitters 42 receivers		58 transmitters 62 receivers			
EP2S60	38 transmitters 42 receivers		58 transmitters 62 receivers		84 transmitters 84 receivers	
EP2S90		38 transmitters 42 receivers		64 transmitters 68 receivers	94 transmitters 92 receivers	118 transmitters 118 receivers
EP2S130				64 transmitters 68 receivers	90 transmitters 94 receivers	156 transmitters 156 receivers
EP2S180					88 transmitters 92 receivers	156 transmitters 156 receivers

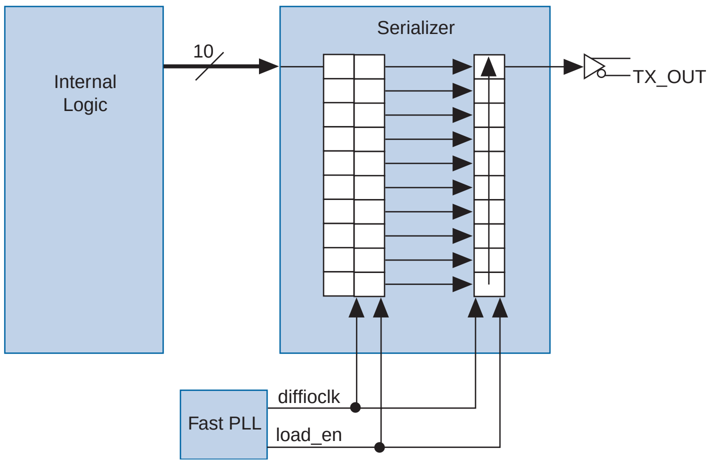
Notes to Table 5–2:
(1) Pin count does not include dedicated PLL input and output pins.
(2) The total number of receiver channels includes the four non-dedicated clock channels that can optionally be used as data channels.
(3) Within the 1,508-pin FBGA package, 92 receiver channels and 92 transmitter channels are vertically migratable.

Differential Transmitter

The Stratix II transmitter has dedicated circuitry to provide support for LVDS and HyperTransport signaling. The dedicated circuitry consists of a differential buffer, a serializer, and a shared fast PLL. The differential buffer can drive out LVDS or HyperTransport signal levels that are statically set in the Quartus® II software. The serializer takes data from a parallel bus up to 10 bits wide from the internal logic, clocks it into the load registers, and serializes it using the shift registers before sending the data to the differential buffer. The most significant bit (MSB) is transmitted first. The load and shift registers are clocked by the `diffioclk` (a fast PLL clock running at the serial rate) and controlled by the load enable signal generated from the fast PLL. The serialization factor can be statically set to ×4, ×5, ×6, ×7, ×8, ×9, or ×10 using the

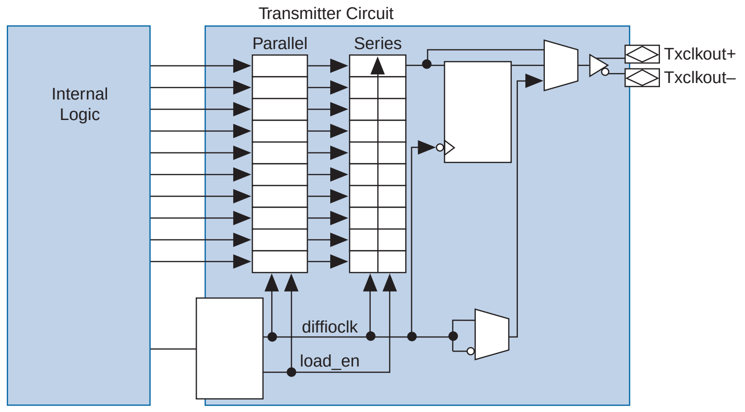
Quartus II software. The load enable signal is automatically generated by the fast PLL and is derived from the serialization factor setting. Figure 5–2 is a block diagram of the Stratix II transmitter.

Figure 5–2. Stratix II Transmitter Block Diagram



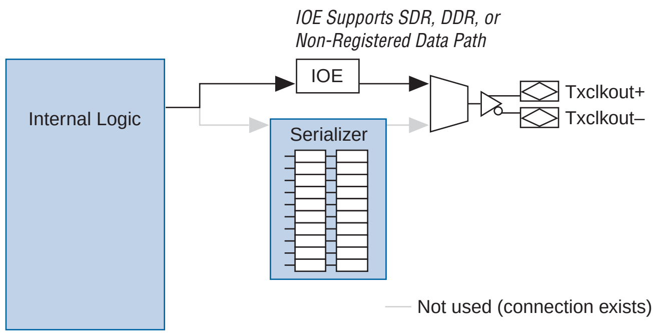
Each Stratix II transmitter data channel can be configured to operate as a transmitter clock output. This flexibility allows the designer to place the output clock near the data outputs to simplify board layout and reduce clock-to-data skew. Different applications often require specific clock to data alignments or specific data rate to clock rate factors. The transmitter can output a clock signal at the same rate as the data with a maximum frequency of 717 MHz. The output clock can also be divided by a factor of 2, 4, 8, or 10. The phase of the clock in relation to the data can be set at 0° or 180° (edge or center aligned). The fast PLL provides additional support for other phase shifts in 45° increments. These settings are made statically in the Quartus II MegaWizard® software. Figure 5–3 shows the Stratix II transmitter in clock output mode.

Figure 5–3. Stratix II Transmitter in Clock Output Mode



The Stratix II serializer can be bypassed to support DDR (×2) and SDR (×1) operations. The I/O element (IOE) contains two data output registers that each can operate in either DDR or SDR mode. The clock source for the registers in the IOE can come from any routing resource, from the fast PLL, or from the enhanced PLL. Figure 5–4 shows the bypass path.

Figure 5–4. Stratix II Serializer Bypass


IOE Supports SDR, DDR, or Non-Registered Data Path
IOE
Internal Logic
Serializer
Txclkout+
Txclkout–
Not used (connection exists)

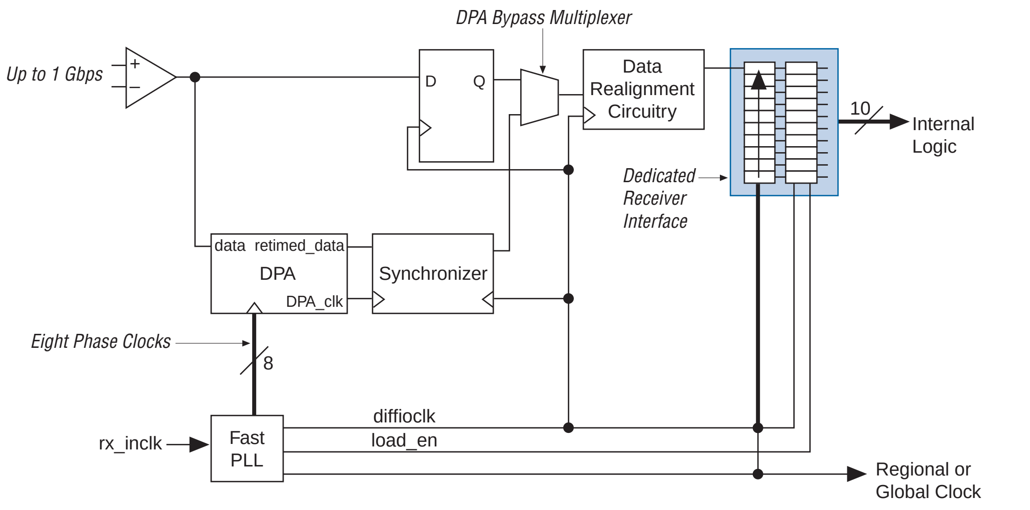
Differential Receiver

The Stratix II receiver has dedicated circuitry to support high-speed LVDS and HyperTransport signaling, along with enhanced data reception. Each receiver consists of a differential buffer, dynamic phase aligner (DPA), synchronization FIFO buffer, data realignment circuit, deserializer, and a shared fast PLL. The differential buffer receives LVDS or HyperTransport signal levels, which are statically set by the Quartus II software. The DPA block aligns the incoming data to one of eight clock phases to maximize the receiver's skew margin. The DPA circuit can be bypassed on a channel-by-channel basis if it is not needed. Set the DPA bypass statically in the Quartus II MegaWizard Plug-In or dynamically by using the optional `RX_DPLL_ENABLE` port.

The synchronizer circuit is a 1-bit wide by 6-bit deep FIFO buffer that compensates for any phase difference between the DPA block and the deserializer. If necessary, the data realignment circuit inserts a single bit of latency in the serial bit stream to align the word boundary. The deserializer includes shift registers and parallel load registers, and sends a maximum of 10 bits to the internal logic. The data path in the Stratix II receiver is clocked by either the `diffioclk` signal or the DPA recovered clock. The deserialization factor can be statically set to 4, 5, 6, 7, 8, 9, or 10 by using the Quartus II software. The fast PLL automatically generates the load enable signal, which is derived from the deserialization factor setting.

Figure 5–5 shows a block diagram of the Stratix II receiver.

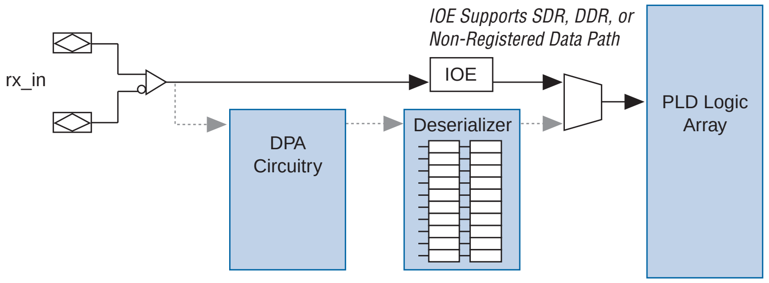
Figure 5–5. Receiver Block Diagram



The Stratix II deserializer, like the serializer, can also be bypassed to support DDR (×2) and SDR (×1) operations. The DPA and data realignment circuit cannot be used when the deserializer is bypassed. The IOE contains two data input registers that can operate in DDR or SDR mode. The clock source for the registers in the IOE can come from any routing resource, from the fast PLL, or from the enhanced PLL.

Figure 5–6 shows the bypass path.

Figure 5–6. Stratix II Deserializer Bypass



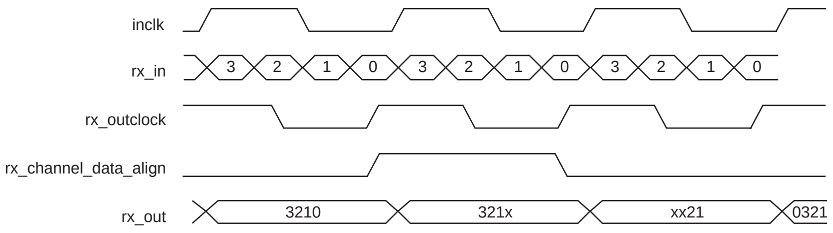
Receiver Data Realignment Circuit

The data realignment circuit aligns the word boundary of the incoming data by inserting bit latencies into the serial stream. An optional `RX_CHANNEL_DATA_ALIGN` port controls the bit insertion of each receiver independently controlled from the internal logic. The data slips one bit for every pulse on the `RX_CHANNEL_DATA_ALIGN` port. The following are requirements for the `RX_CHANNEL_DATA_ALIGN` port:

- The minimum pulse width is one period of the parallel clock in the logic array.
- The minimum low time between pulses is one period of parallel clock.
- There is no maximum high or low time.
- Valid data is available two parallel clock cycles after the rising edge of `RX_CHANNEL_DATA_ALIGN`.

Figure 5–7 shows receiver output (`RX_OUT`) after one bit slip pulse with the deserialization factor set to 4.

Figure 5–7. Data Realignment Timing

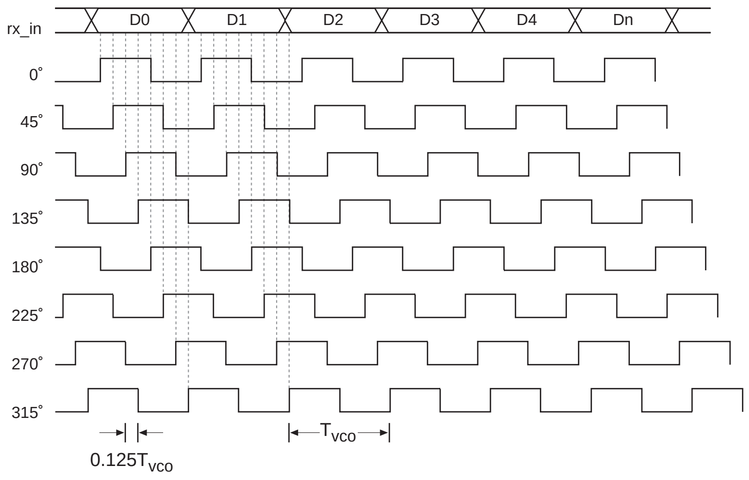


The data realignment circuit can have up to 11 bit-times of insertion before a rollover occurs. The programmable bit rollover point can be from 1 to 11 bit-times independent of the deserialization factor. An optional status port, `RX_CDA_MAX`, is available to the FPGA from each channel to indicate when the preset rollover point is reached.

Dynamic Phase Aligner

The DPA block takes in high-speed serial data from the differential input buffer and selects one of eight phase clocks to sample the data. The DPA chooses a phase closest to the phase of the serial data. The maximum phase offset between the data and the phase-aligned clock is 1/8 UI, which is the maximum quantization error of the DPA. The eight phases are equally divided, giving a 45-degree resolution. Figure 5–8 shows the possible phase relationships between the DPA clocks and the incoming serial data.

Figure 5–8. DPA Clock Phase to Data Bit Relationship



Each DPA block continuously monitors the phase of the incoming data stream and selects a new clock phase if needed. The selection of a new clock phase can be prevented by the optional `RX_DPLL_HOLD` port, which is available for each channel.

The DPA block requires a training pattern and a training sequence of at least 256 repetitions of the training pattern. The training pattern is not fixed, so the designer can use any training pattern with at least one transition. An optional output port, `RX_DPA_LOCKED`, is available to the internal logic, to indicate when the DPA block has settled on the closest phase to the incoming data phase. The RX_DPA_LOCKED de-asserts, depending on what is selected in the Quartus II MegaWizard Plug-In, when either a new phase is selected, or when the DPA has moved two phases in the same direction. The data may still be valid even when the RX_DPA_LOCKED is de-asserted. Use data checkers to validate the data when RX_DPA_LOCKED is de-asserted.

An independent reset port, RX_RESET, is available to reset the DPA circuitry. The DPA circuit must be retrained after reset.

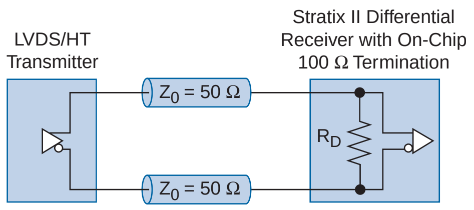
Synchronizer

The synchronizer is a 1-bit × 6-bit deep FIFO buffer that compensates for the phase difference between the recovered clock from the DPA circuit and the diffioclk that clocks the rest of the logic in the receiver. The synchronizer can only compensate for phase differences, not frequency differences between the data and the receiver's INCLK. An optional port, RX_FIFO_RESET, is available to the internal logic to reset the synchronizer.

Differential I/O Termination

Stratix II devices provide an on-chip 100-Ω differential termination option on each differential receiver channel for LVDS and HyperTransport standards. The on-chip termination eliminates the need to supply an external termination resistor, simplifying the board design and reducing reflections caused by stubs between the buffer and the termination resistor. You can enable on-chip termination in the Quartus II assignments editor. Differential on-chip termination is supported across the full range of supported differential data rates as shown in the *High-Speed I/O Specifications* section of the *DC& Switching Characteristics* in Volume 1 of the *Stratix II Device Handbook*.

Figure 5–9. On-Chip Differential Termination

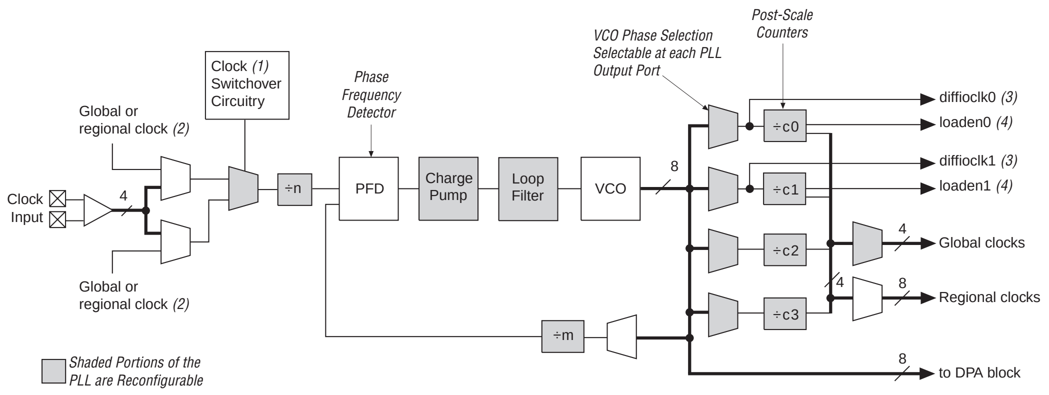

LVDS/HT Transmitter
Stratix II Differential Receiver with On-Chip 100 Ω Termination
Z_0 = 50 Ω
Z_0 = 50 Ω
R_D

Fast PLL

The high-speed differential I/O receiver and transmitter channels use the fast PLL to generate the parallel global clocks (rx- or tx- clock) and high-speed clocks (diffioclk). Figure 5–1 shows the locations of the fast PLLs. The fast PLL VCO operates at the clock frequency of the data rate. Each fast PLL offers a single serial data rate support, but up to two separate serialization and/or deserialization factors (from the C0 and C1 fast PLL clock outputs) can be used. Clock switchover and dynamic fast PLL reconfiguration is available in high-speed differential I/O support mode. For additional information on the fast PLL, refer to the *PLLs in Stratix II Devices* chapter in Volume 2 of the *Stratix II Handbook*.

Figure 5–10 shows a block diagram of the fast PLL in high-speed differential I/O support mode.

Figure 5–10. Fast PLL Block Diagram



Notes to Figure 5–10:

(1) Stratix II fast PLLs only support manual clock switchover.

(2) The global or regional clock input can be driven by an output from another PLL, a pin-driven global or regional clock or internally-generated global signals.

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

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

Clocking

The fast PLLs feed in to the differential receiver and transmitter channels through the LVDS/DPA clock network. The center fast PLLs can independently feed the banks above and below them. The corner PLLs can feed only the banks adjacent to them. Figures 5–11 and 5–12 show the LVDS and DPA clock networks of the Stratix II devices.

Figure 5–11. Fast PLL & LVDS/DPA Clock for EP2S15 & EP2S30 Devices

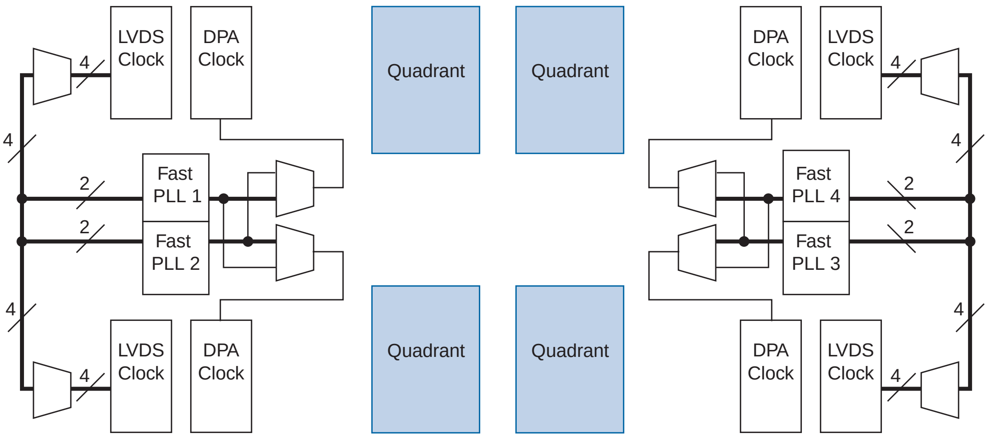
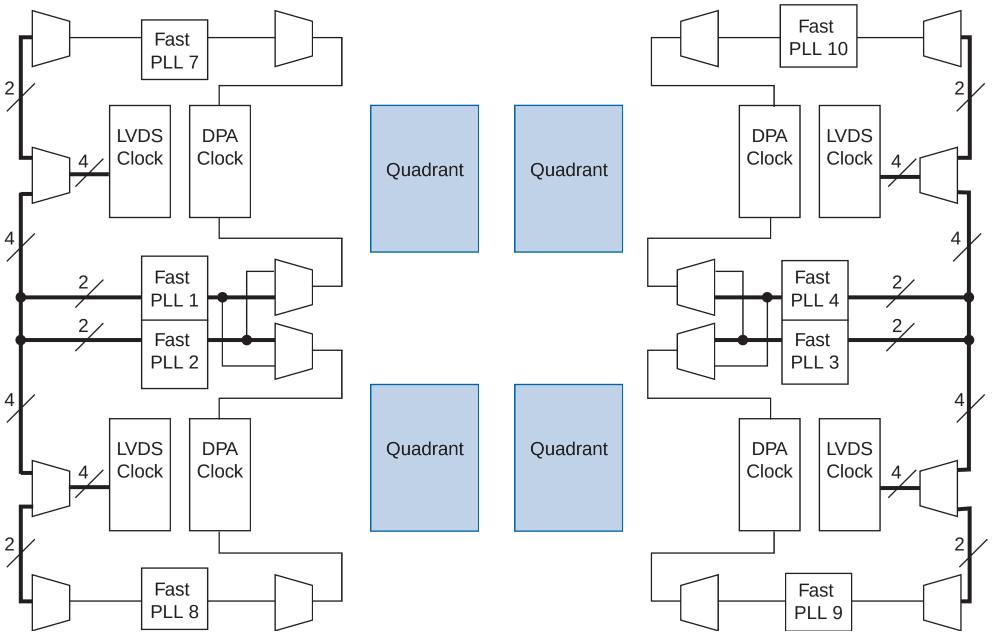


Figure 5–12. Fast PLL & LVDS/DPA Clocks for EP2S60, EP2S90, EP2S130 & EP2S180 Devices



Source Synchronous Timing Budget

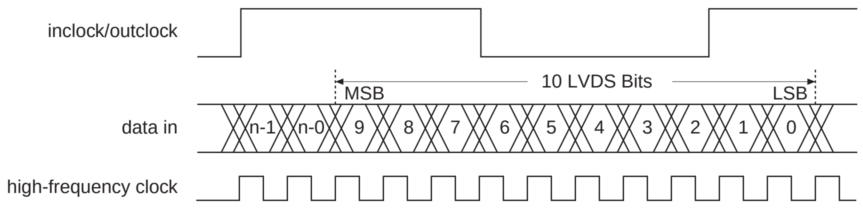
This section discusses the timing budget, waveforms, and specifications for source-synchronous signaling in Stratix II devices. LVDS and HyperTransport I/O standards enable high-speed data transmission. This high data transmission rate results in better overall system performance. To take advantage of fast system performance, it is important to 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 the use of timing parameters provided by IC vendors and is strongly influenced by board skew, cable skew, and clock jitter. This section defines the source-synchronous differential data orientation timing parameters, the timing budget definitions for Stratix II devices, and 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 1 Gbps and SERDES factor of 10, the external clock is multiplied by 10, and phase-alignment can be set in the PLL to coincide with the sampling window of each data bit. The data is sampled on the falling edge of the multiplied clock. Figure 5–13 shows the data bit orientation of the ×10 mode.

Figure 5–13. Bit Orientation in the Quartus II Software



Differential I/O Bit Position

Data synchronization is necessary for successful data transmission at high frequencies. Figure 5–14 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 5–3.

Figure 5–14. Bit Order for One Channel of Differential Data

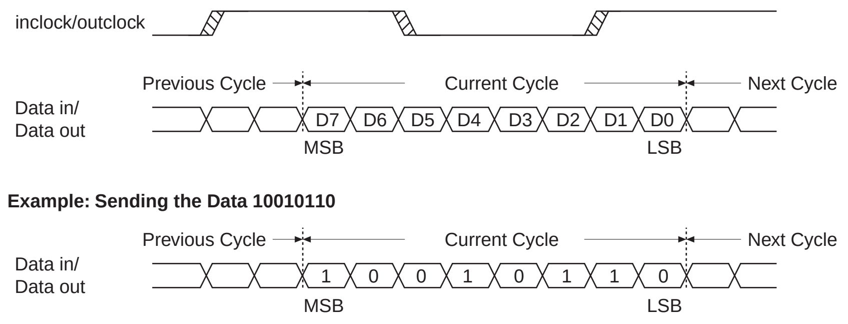


Table 5–3 shows the conventions for differential bit naming for 18 differential channels. The MSB and LSB positions increase with the number of channels used in a system.

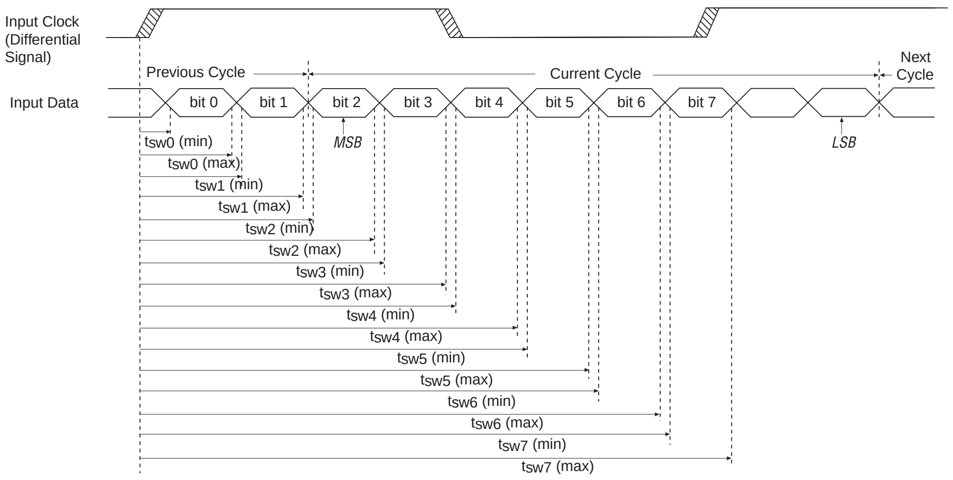
Table 5–3. LVDS Bit Naming

Receiver Channel Data 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

Input Timing Waveform

Figure 5–15 shows the essential operations and the timing relationship between the clock cycle and the incoming serial data. The bit positions are shown as an example only and do not represent the word boundary for all cases. Word alignment circuit or the bit slipper control circuit is necessary if a specific word boundary is needed.

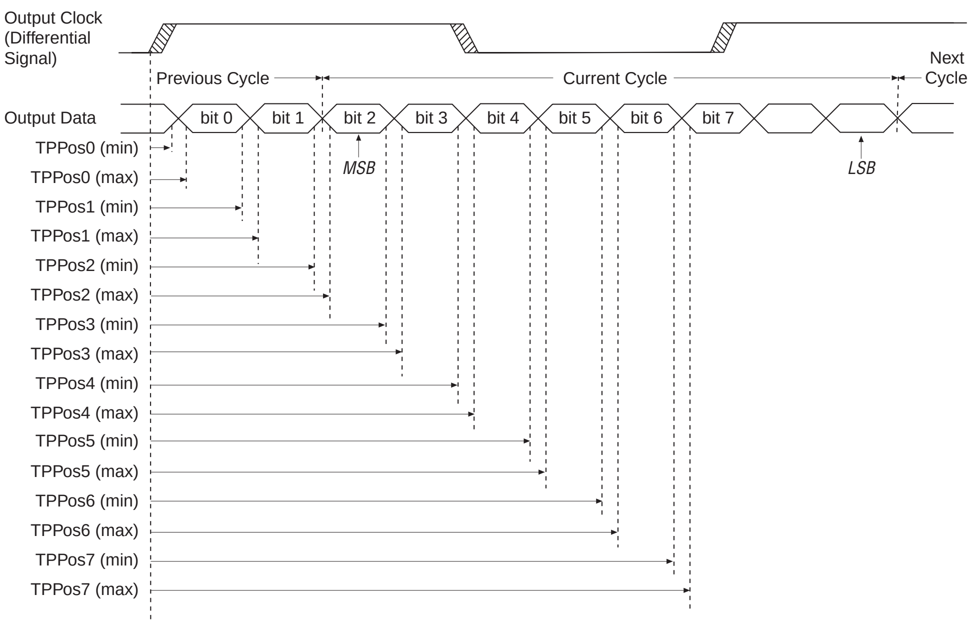
Figure 5–15. Input Timing Waveform



Output Timing

The output timing waveform in Figure 5–16 shows the relationship between the output clock and the serial output data stream.

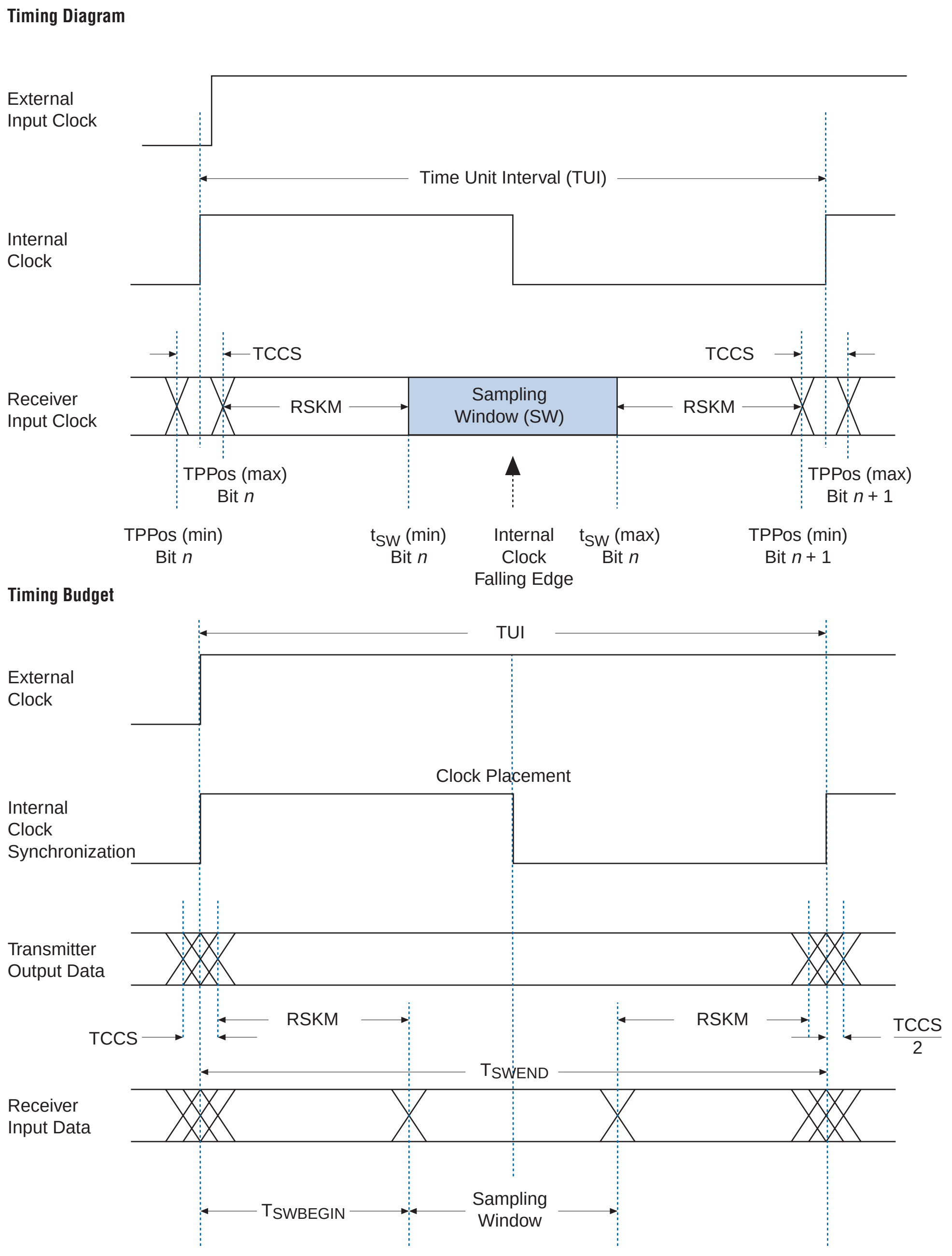
Figure 5–16. Output Timing Waveform



Receiver Skew Margin

Changes in system environment, such as temperature, media (cable, connector, or PCB) loading effect, the receiver's setup and hold times, and internal skew, reduce the sampling window for the receiver. The timing margin between the receiver’s clock input and the data input sampling window is called Receiver Skew Margin (RSKM). Figure 5–17 shows the relationship between the RSKM and the receiver’s sampling window.

Figure 5–17. Differential High-Speed Timing Diagram & Timing Budget



Differential Pin Placement Guidelines

In order to ensure proper high-speed operation, differential pin placement guidelines have been established. The Quartus II compiler automatically checks that these guidelines are followed and will issue an error message if these guidelines are not met. This section is separated into guidelines with and without DPA usage.

DPA Usage Guidelines

The Stratix II device has differential receivers and transmitters on the left and right banks of the device. Each receiver has a dedicated DPA circuit to align the phase of the clock to the data phase of its associated channel. When a channel or channels of left or right banks are used in DPA mode, the guidelines listed below must be adhered to.

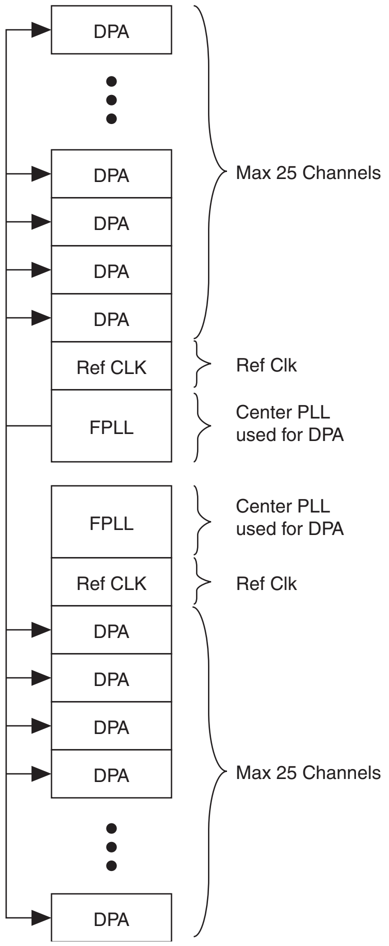
DPA & Single-Ended I/Os

- When there is a DPA channel enabled in a bank, single-ended I/Os are not allowed in the bank. Only differential I/O standards are allowed in the bank.

FPLL/DPA Channel Driving Distance

- Each FPLL can drive up to 25 adjacent channels in DPA mode in a single bank (not including the reference clock channel). All device/package offerings, except for those in the 1508-pin package, have less than 25 channels in a bank.
- Unused channels can be within the 25 limit, but all used channels must be in DPA mode from the same FPLL. Center FPLLs can drive two banks simultaneously, such that up to 50 channels (25 on the bank above and 25 on the bank below the FPLL) can be driven as shown in Figure 5–18.
- If one center FPLL drives DPA channels in the bank above and below it, the other center FPLL cannot drive DPA channels.

Figure 5–18. Driving Capabilities of a Center FPLL

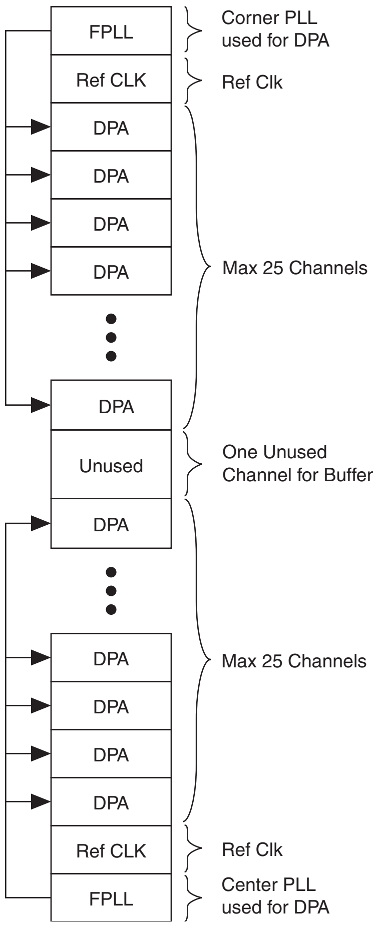


Using Corner & Center FPLLs

- If a differential bank is being driven by two FPLLs, where the corner PLL is driving one group and the center FPLL is driving another group, there must be at least 1 row of separation between the two groups of DPA channels (see Figure 5–19). The two groups can operate at independent frequencies.

- No separation is necessary if a single FPLL is driving DPA channels as well as non-DPA channels as long as the DPA channels are contiguous.

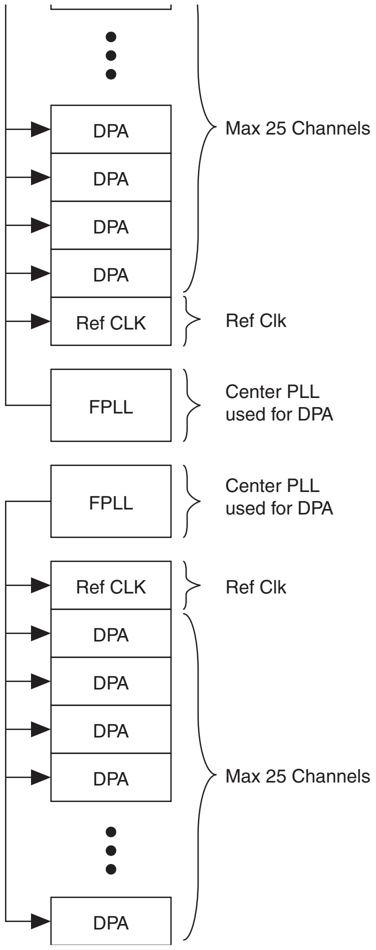
Figure 5–19. Usage of Corner and Center FPLLs Driving DPA Channels in a Single Bank



Using Both Center FPLLs

- Both center FPLLs can be used for DPA as long as they drive DPA channels in their adjacent quadrant only. See Figure 5–20.
- Both center FPLLs cannot be used for DPA if one of the FPLLs drives the top and bottom banks, or if they are driving cross banks (e.g. lower FPLL drives top bank and top FPLL drives lower bank).

Figure 5–20. Center FPLL Usage when Driving DPA Channels



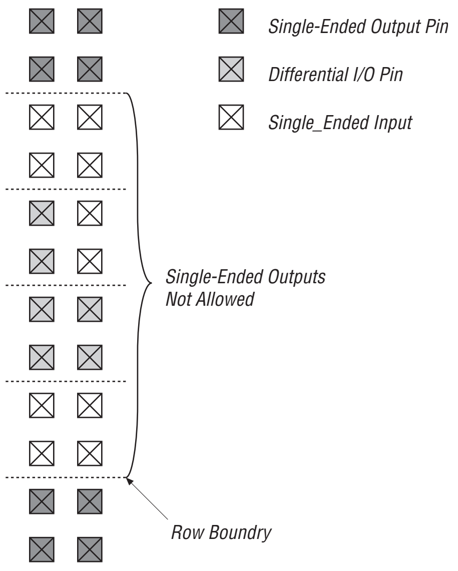
Non-DPA Differential I/O Usage Guidelines

When a differential channel or channels of left or right banks are used in non-DPA mode, you must adhere to the following guidelines.

High-Speed Differential I/Os & Single-Ended I/Os

- Single-ended I/Os are allowed in the same I/O bank as long as the single-ended I/O standard uses the same V_{CCIO} as the high-speed I/O bank. Single-ended inputs can be in the same LAB row; however, IOE input registers are not available for the single-ended I/Os placed in the same LAB row as differential I/Os. The input register must be implemented within the core logic.
- Single-ended output pins must be at least one LAB row away from differential I/O pins, as shown in Figure 5–21.

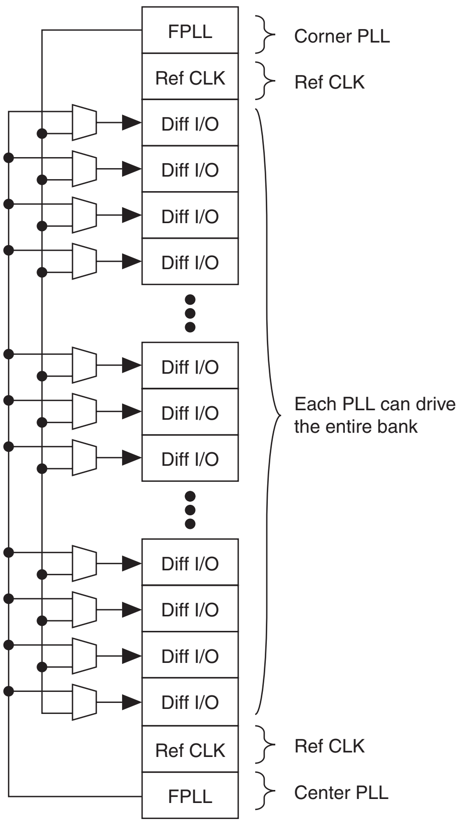
Figure 5–21. Single-Ended Output Pin Placement with Respect to Differential I/O Pins



FPLL/Differential I/O Driving Distance

- As shown in Figure 5–22, each FPLL can drive all the channels in the entire bank, except for within the 1508-pin FBGA package (see the "Differential I/Os in the FBGA 1508-Pin Package Guidelines" section).

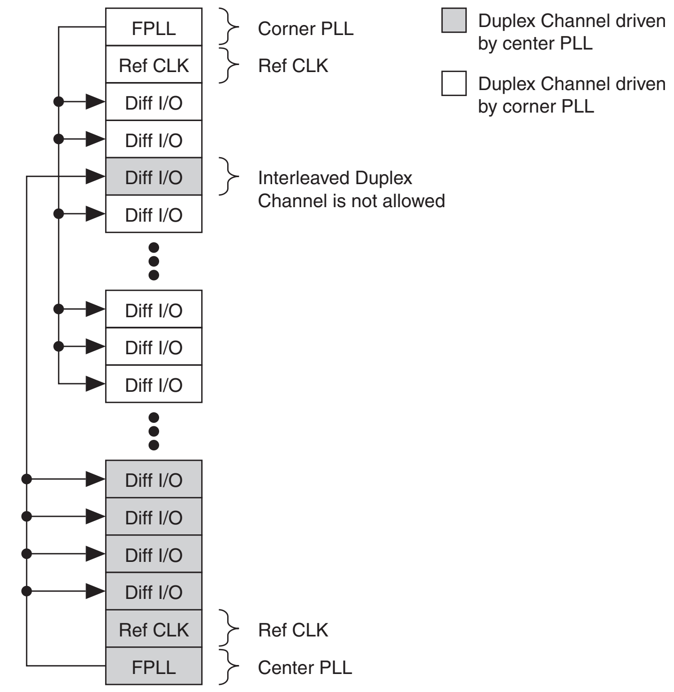
Figure 5–22. FPLL Driving Capability When Driving Non-DPA Differential Channels



Using Corner & Center FPLLs

- The corner and center FPLLs can be used as long as the channels driven by separate FPLLs do not have their transmitter or receiver channels interleaved. Figure 5–23 shows illegal placement of differential channels when using corner and center FPLLs.
- If one FPLL is driving transmitter channels only, and the other FPLL drives receiver channels only, the channels driven by those FPLLs can overlap each other.
- Center FPLLs can be used for both transmitter and receiver channels.

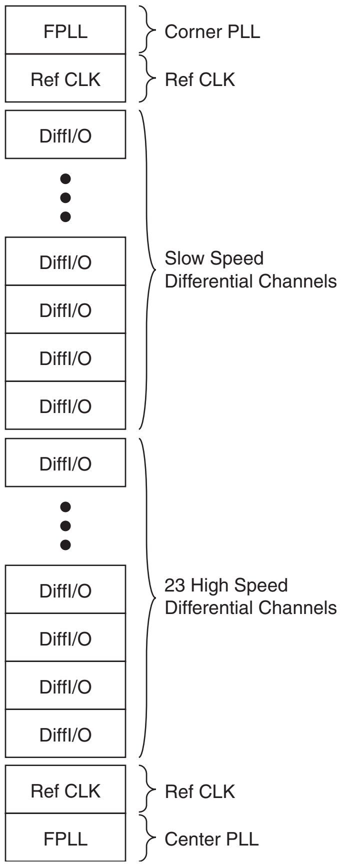
Figure 5–23. Illegal Placement of Interlaced Duplex Channels in an I/O Bank



Differential I/Os in the FBGA 1508-Pin Package Guidelines

- In the 1508-pin package, channels more than 23 rows away from the center FPLLs, not including the reference clock row, will operate at a reduced rate due to the package routing (see Figure 5–24).

Figure 5–24. Fast and Slow Speed Channels in a 1508-Pin Package


FPLL
Corner PLL
Ref CLK
Ref CLK
DiffI/O
Slow Speed Differential Channels
23 High Speed Differential Channels
Ref CLK
Ref CLK
FPLL
Center PLL

Board Design Considerations

This section explains how to achieve the optimal performance from the Stratix II high-speed I/O block and ensure first-time success in implementing a functional design with optimal signal quality. For more information on board layout recommendations and I/O pin terminations, refer to *AN 224: High-Speed Board Layout Guidelines*.

To achieve the best performance from the device, pay attention to the impedances of traces and connectors, differential routing, and termination techniques. Use this section together with the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook*.

The Stratix II high-speed module generates signals that travel over the media at frequencies as high as one Gbps. Board designers should use the following guidelines:

- Base board 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 HMZD or VHDM connectors for backplane designs. Two suppliers of high-performance connectors are Teradyne Corp (**www.teradyne.com**) and Tyco International Ltd. (**www.tyco.com**).
- Design backplane and card traces so that trace impedance matches the connector's or the termination's impedance.
- Keep an 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 when the TCCS value increases.
- Limit vias, because they cause impedance discontinuities.
- Use the common bypass capacitor values such as 0.001, 0.01, 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.
- Route signals on adjacent layers orthogonally to each other.

Conclusion

Stratix II high-speed differential inputs and outputs, with their DPA and data realignment circuitry, allow users to build a robust multi-Gigabit system. The DPA circuitry allows users to compensate for any timing skews resulting from physical layouts. The data realignment circuitry allows the devices to align the data packet between the transmitter and receiver. Together with the on-chip differential termination, Stratix II devices can be used as a single-chip solution for high-speed applications.

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.

This section contains the following chapter:

- Chapter 6, DSP Blocks in Stratix II Devices

Revision History

The table below shows the revision history for Chapter 6.

Chapter	Date / Version	Changes Made
6	July 2004, v1.1	Updated "Saturation & Rounding" section. Updated reference on page 6–31.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.

6. DSP Blocks in Stratix II Devices

SII52006-1.1

Introduction

Stratix® II devices have dedicated digital signal processing (DSP) blocks optimized for DSP applications requiring high data throughput. These DSP blocks combined with the flexibility of programmable logic devices (PLDs), provide designers with the ability to implement various high performance DSP functions easily. Complex systems such as CDMA2000, voice over Internet protocol (VoIP), high-definition television (HDTV) require high performance DSP blocks to process data. These system designs typically use DSP blocks as finite impulse response (FIR) filters, complex FIR filters, fast Fourier transform (FFT) functions, discrete cosine transform (DCT) functions, and correlators.

Stratix II DSP blocks consists of a combination of dedicated blocks that perform multiplication, addition, subtraction, accumulation, and summation operations. Designers can configure these blocks to implement arithmetic functions like multipliers, multiply-adders and multiply-accumulators which are necessary for most DSP functions.

Along with the DSP blocks, the TriMatrix™ memory structures in Stratix II devices also support various soft multiplier implementations. The combination of soft multipliers and dedicated DSP blocks increases the number of multipliers available in Stratix II devices and provides the user with a wide variety of implementation options and flexibility when designing their systems.

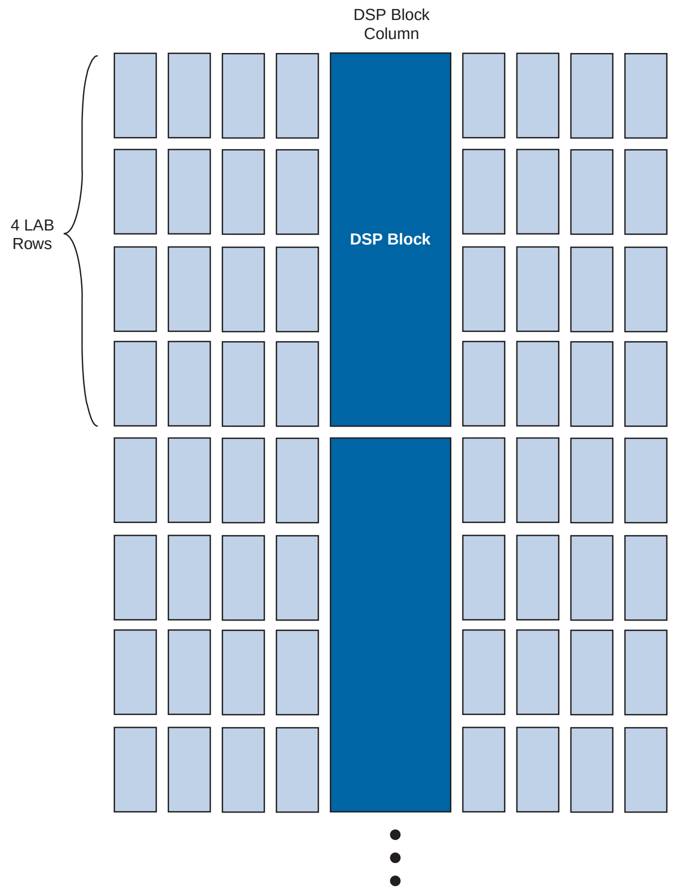
See the *Stratix II Device Family Data Sheet* in Volume 1 of the *Stratix II Device Handbook* for more information on Stratix II Devices.

DSP Block Overview

Each Stratix II device has two to four columns of DSP blocks that efficiently implement multiplication, multiply-accumulate (MAC) and multiply-add functions. Figure 6–1 shows the arrangement of one of the DSP block columns with the surrounding LABs. Each DSP block can be configured 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 with Adjacent LABs



The multipliers then feed an adder or accumulator block within the DSP block. Stratix II device multipliers support rounding and saturation on Q1.15 input formats. The DSP block also has input registers that can be configured to operate in a shift register chain for efficient implementation of functions like FIR filters. The accumulator within the DSP block can be initialized to any value and supports rounding and saturation on Q1.15 input formats to the multiplier. A single DSP block can be broken down to operate different configuration modes simultaneously.

For more information on Q1.15 formatting, see the "Saturation & Rounding" section.

The number of DSP blocks per column and the number of columns available increases with device density. Table 6–1 shows the number of DSP blocks in each Stratix II device and the multipliers that you can implement.

Table 6–1. Number of DSP Blocks in Stratix II Devices *Note (1)*

Device	DSP Blocks	9 × 9 Multipliers	18 × 18 Multipliers	36 × 36 Multipliers
EP2S15	12	96	48	12
EP2S30	16	128	64	16
EP2S60	36	288	144	36
EP2S90	48	384	192	48
EP2S130	63	504	252	63
EP2S180	96	768	384	96

Note to Table 6–1:

(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.

In addition to the DSP block multipliers, you can use the Stratix II device TriMatrix memory blocks for soft multipliers. The availability of soft multipliers increases the number of multipliers available within the device. Table 6–2 shows the total number of multipliers available in Stratix II devices using DSP blocks and soft multipliers.

Table 6–2. Number of Multipliers in Stratix II Devices

Device	DSP Blocks (18 × 18)	Soft Multipliers (16 × 16) *(1)*, *(2)*	Total Multipliers *(3)*, *(4)*
EP2S15	48	100	148 (3.08)
EP2S30	64	189	253 (3.95)
EP2S60	144	325	469 (3.26)
EP2S90	192	509	701 (3.65)
EP2S130	252	750	1,002 (3.98)
EP2S180	384	962	1,346 (3.51)

Notes to Table 6–2:
(1) Soft multipliers implemented in sum of multiplication mode. RAM blocks are configured with 18-bit data widths and sum of coefficients up to 18-bits.
(2) Soft multipliers are only implemented in M4K and M512 TriMatrix memory blocks, not M-RAM blocks.
(3) The number in parentheses represents the increase factor, which is the total number of multipliers with soft multipliers divided by the number of 18 × 18 multipliers supported by DSP blocks only.
(4) The total number of multipliers may vary according to the multiplier mode used.

See the *Stratix II Architecture* chapter in Volume 1 of the *Stratix II Device Handbook* for more information on Stratix II TriMatrix memory blocks. Refer to Techniques for Implementing Multipliers in Stratix II Devices for more information on soft multipliers.

Figure 6–2 shows the DSP block configured for 18 × 18 multiplier mode. Figure 6–3 shows the 9 × 9 multiplier configuration of the DSP block.

Figure 6–2. DSP Block in 18 × 18 Mode

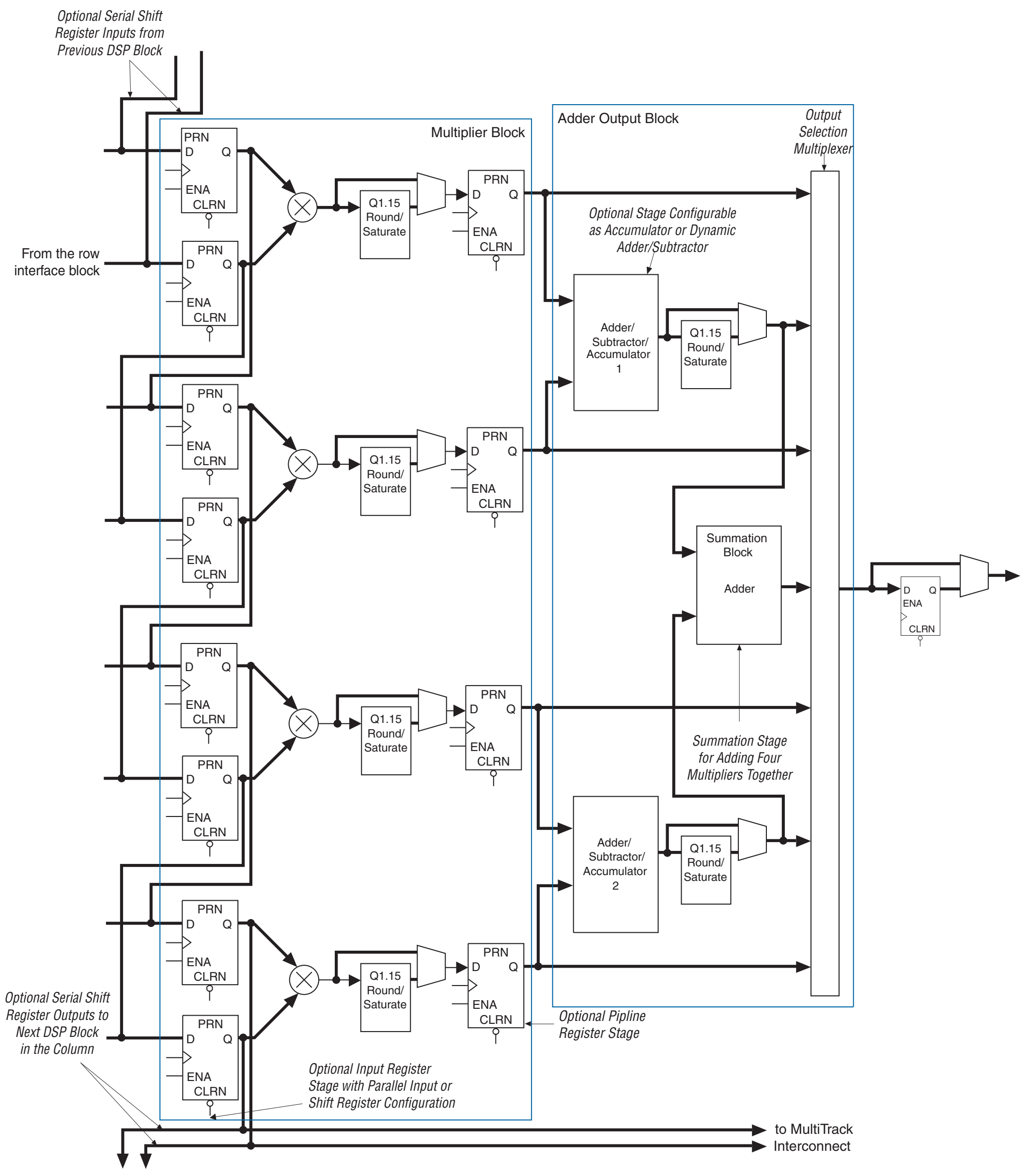
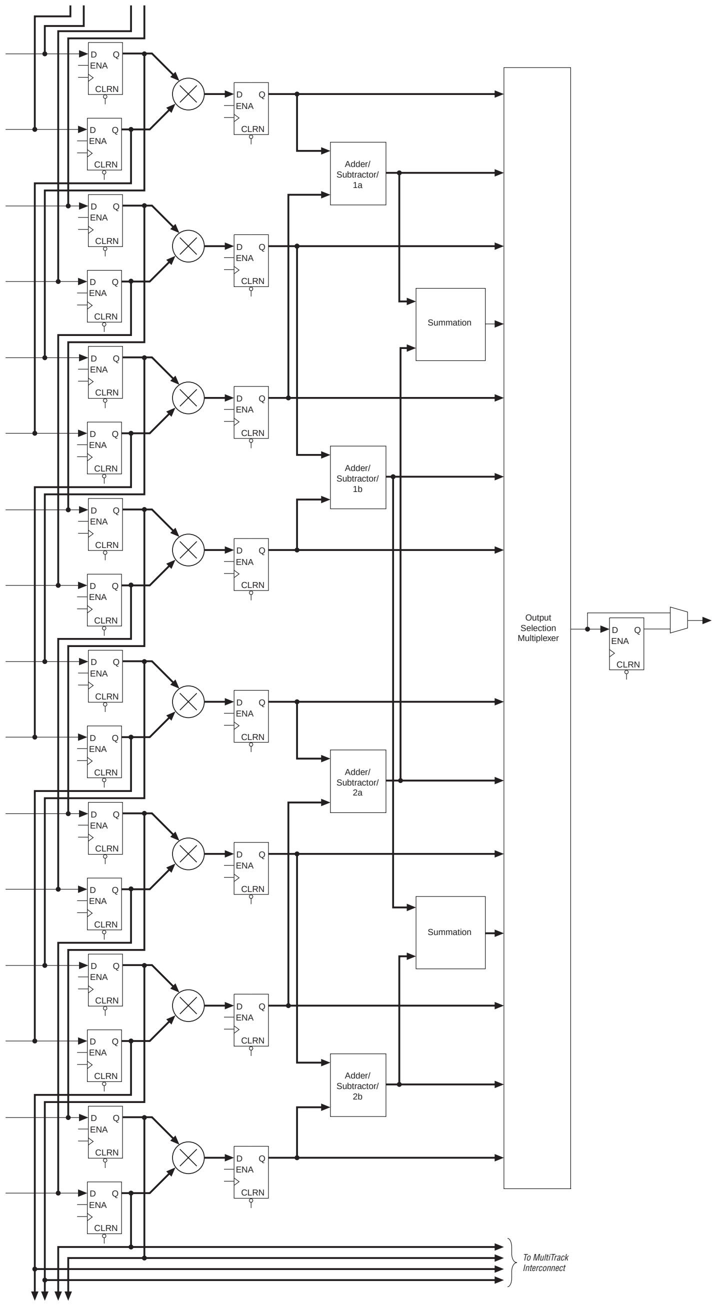

Optional Serial Shift Register Inputs from Previous DSP Block
Multiplier Block
Adder Output Block
Output Selection Multiplexer
From the row interface block
PRN
D
Q
ENA
CLRN
Q1.15 Round/ Saturate
Optional Stage Configurable as Accumulator or Dynamic Adder/Subtractor
Adder/ Subtractor/ Accumulator 1
Summation Block
Adder
Summation Stage for Adding Four Multipliers Together
Adder/ Subtractor/ Accumulator 2
Optional Serial Shift Register Outputs to Next DSP Block in the Column
Optional Pipline Register Stage
Optional Input Register Stage with Parallel Input or Shift Register Configuration
to MultiTrack Interconnect

Figure 6–3. DSP Block in 9 × 9 Mode



Architecture

The DSP block consists of the following elements:

- A multiplier block
- An adder/subtractor/accumulator block
- A summation block
- Input and output interfaces
- Input and output registers

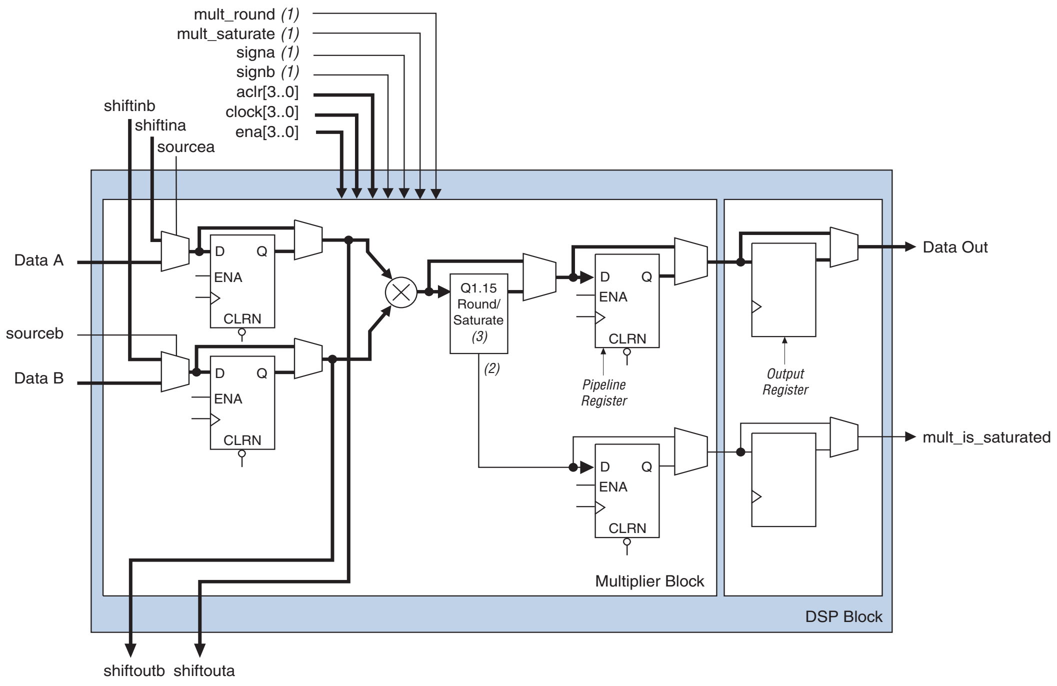
Multiplier Block

Each multiplier block has the following elements:

- Input registers
- A multiplier block
- A rounding and/or saturation stage for Q1.15 input formats
- A pipeline output register

Figure 6–4 shows the multiplier block architecture.

Figure 6–4. Multiplier Block Architecture



Notes to Figure 6–4:
(1) These signals are not registered or registered once to match the data path pipeline.
(2) You can send these signals through either one or two pipeline registers.
(3) The rounding and/or saturation is only supported in 18 × 18-bit signed multiplication for Q1.15 inputs.

Input Registers

Each multiplier operand can feed an input register or directly to the multiplier. The following DSP block signals control each input register within the DSP block:

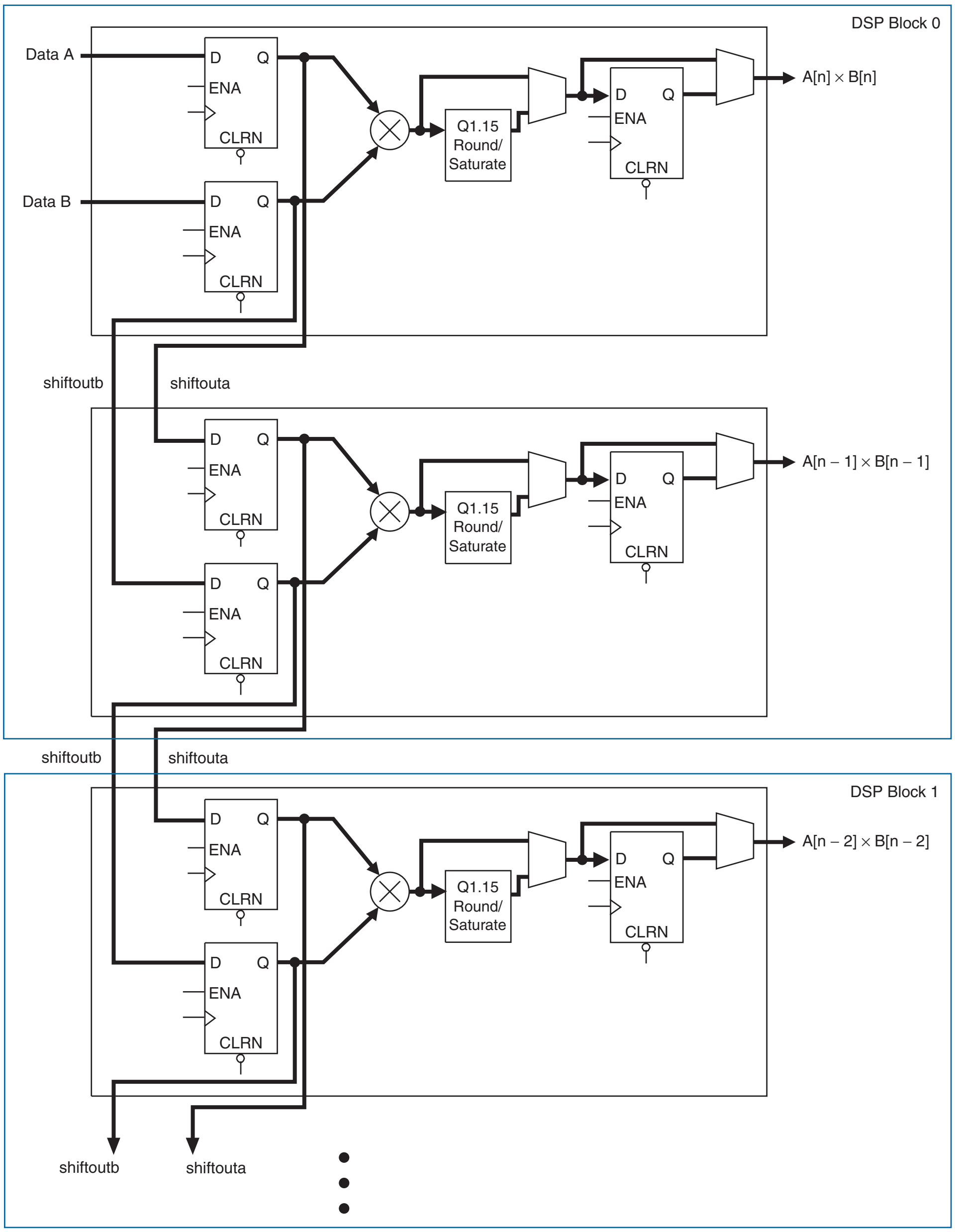
- `clock[3..0]`
- `ena[3..0]`
- `aclr[3..0]`

The input registers feed the multiplier and drive two dedicated shift output lines, `shiftouta` and `shiftoutb`. The dedicated shift outputs from one multiplier block directly feed input registers of the adjacent multiplier below it within the same DSP block or the first multiplier in the next DSP block to form a shift register chain, as shown in Figure 6–5. The dedicated shift register chain spans a single column but longer shift register chains requiring multiple columns can be implemented using regular FPGA routing resources. Therefore, this shift register chain can be of any length up to 768 registers in the largest member of the Stratix II device family.

Shift registers are useful in DSP functions like 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 significantly reduces the LE resources required, avoids routing congestion, and results in predictable timing.

Stratix II DSP blocks allow the user to dynamically select whether a particular multiplier operand is fed by regular data input or the dedicated shift register input using the `sourcea` and `sourceb` signals. A logic 1 value on the `sourcea` signal indicates that data A is fed by the dedicated scan-chain; a logic 0 value indicates that it is fed by regular data input. This feature allows the implementation of a dynamically loadable shift register where the shift register operates normally using the scan-chains and can also be loaded dynamically in parallel using the data input value.

Figure 6–5. Shift Register Chain *Note (1)*



Note to Figure 6–5:

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

Table 6–3 shows the summary of input register modes for the DSP block.

Table 6–3. Input Register Modes

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

Multiplier Stage

The multiplier stage supports 9 × 9, 18 × 18, or 36 × 36 multipliers as well as other smaller multipliers in between these configurations. See "Operational Modes" on page 6–19 for details. Depending on the data width of the multiplier, a single DSP block can perform many multiplications in parallel.

Each multiplier operand can be a unique signed or unsigned number. Two signals, `signa` and `signb`, control the representation of each operand respectively. A logic 1 value on the `signa` signal indicates that data A is a signed number while a logic 0 value indicates an unsigned number. Table 6–4 shows the sign of the multiplication result for the various operand sign representations. The result of the multiplication is signed if any one of the operands is a signed value.

Table 6–4. Multiplier Sign Representation

Data A (signa Value)	Data B (signb Value)	Result
Unsigned (logic 0)	Unsigned (logic 0)	Unsigned
Unsigned (logic 0)	Signed (logic 1)	Signed
Signed (logic 1)	Unsigned (logic 0)	Signed
Signed (logic 1)	Signed (logic 1)	Signed

There is only one `signa` and one `signb` signal for each 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.

When the `signa` and `signb` signals are unused, the Quartus® II software sets the multiplier to perform unsigned multiplication by default.

Saturation & Rounding

The DSP blocks have hardware support to perform optional saturation and rounding after each 18 × 18 multiplier for Q1.15 input formats.

Designs must use 18 × 18 multipliers for the saturation and rounding options because the Q1.15 input format requires 16-bit input widths.

Q1.15 input format multiplication requires signed multipliers. The most significant bit (MSB) in the Q1.15 input format represents the value's sign bit. Use signed multipliers to ensure the proper sign extension during multiplication.

The Q1.15 format uses 16 bits to represent each fixed point input. The MSB is the sign bit, and the remaining 15-bits are used to represent the value after the decimal place (or the fractional value). This Q1.15 value is equivalent to an integer number representation of the 16-bits divided by 2^{15}, as shown in the following equations.

$$-\frac{1}{2} = 1\ 100\ 0000\ 0000\ 0000 = -\frac{0x4000}{2^{15}}$$

$$\frac{1}{8} = 0\ 001\ 0000\ 0000\ 0000 = \frac{0x1000}{2^{15}}$$

All Q1.15 numbers are between –1 and 1.

When performing multiplication, even though the Q1.15 input only uses 16 of the 18 multiplier inputs, the entire 18-bit input bus is transmitted to the multiplier. This is like a 1.17 input, where the two least significant bits (LSBs) are always 0.

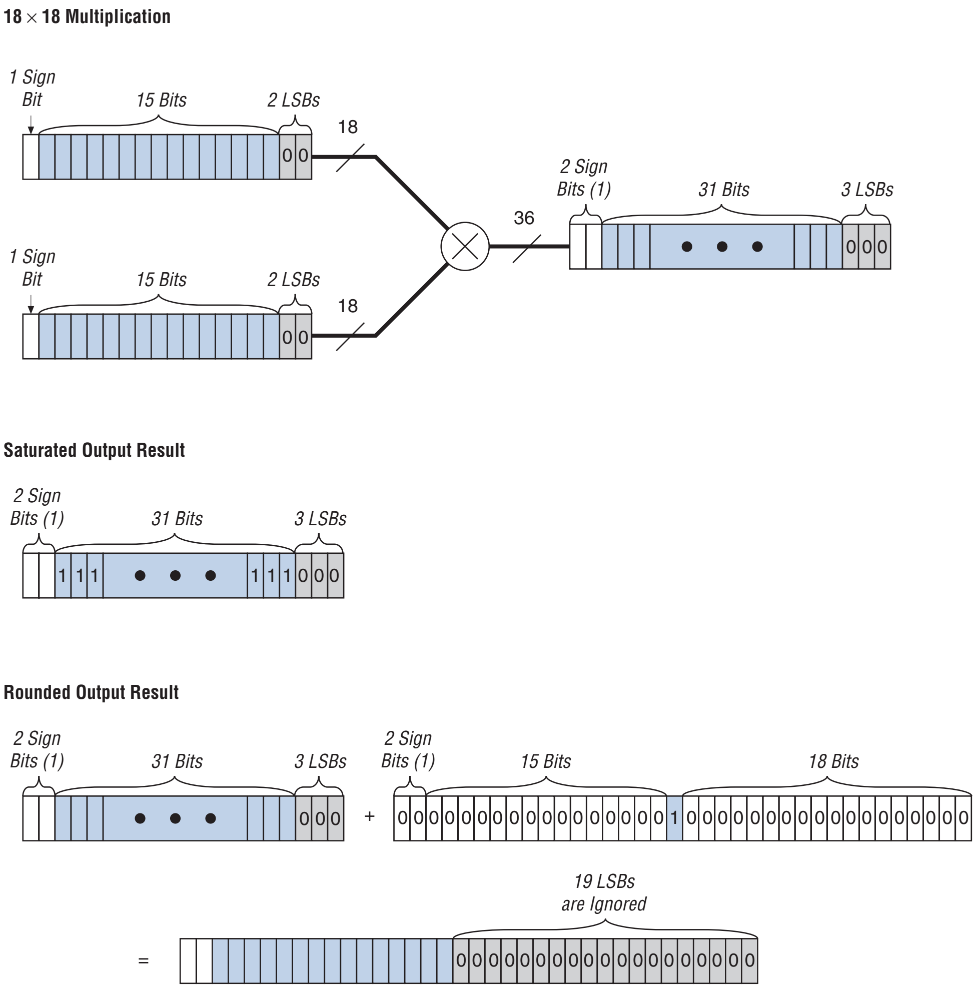
The multiplier output will be a 2.34 value (36 bits total) before performing any rounding or saturation. The two MSBs are sign bits. Since the output only requires one sign bit, you can ignore one of the two MSBs, resulting in a Q1.34 value before rounding or saturation.

When the design performs saturation, the multiplier output gets saturated to 0x7FFFFFFF in a 1.31 format. This uses bits [34..3] of the overall 36-bit multiplier output. The three LSBs are set to 0.

The DSP block obtains the `mult_is_saturated` or `accum_is_saturated` overflow signal value from the LSB of the multiplier or accumulator output. Therefore, whenever saturation occurs, the LSB of the multiplier or accumulator output will send a `1` to the `mult_is_saturated` or `accum_is_saturated` overflow signal. At all other times, this overflow signal is `0` when saturation is enabled or reflects the value of the LSB of the multiplier or accumulator output.

When the design performs rounding, it adds 0x00008000 in 1.31 format to the multiplier output, and it only uses bits [34..15] of the overall 36-bit multiplier output. Adding 0x00008000 in 1.31 format to the 36-bit multiplier result is equivalent to adding 0x0 0004 0000 in 2.34 format. The 16 LSBs are set to 0. Figure 6–6 shows which bits are used when the design performs rounding and saturation for the multiplication.

Figure 6–6. Rounding & Saturation Bits



Note to Figure 6–6:

(1) Both sign bits are the same. The design only uses one sign bit, and the other one is ignored.

If the design performs a `multiply_accumulate` or `multiply_add` operation, the multiplier output is input to the adder/subtractor/accumulator blocks as a 2.31 value, and the three LSBs are 0.

Pipeline Registers

The output from the multiplier can feed a pipeline register or this register can be bypassed. Pipeline registers may be implemented for any multiplier size and increase the DSP block's maximum performance, especially when using the subsequent DSP block adder stages. Pipeline registers split up the long signal path between the adder/subtractor/accumulator block and the adder/output block, creating two shorter paths.

Adder/Output Block

The adder/output block has the following elements:

- An adder/subtractor/accumulator block
- A summation block
- An output select multiplexer
- Output registers

Figure 6–7 shows the adder/output block architecture.

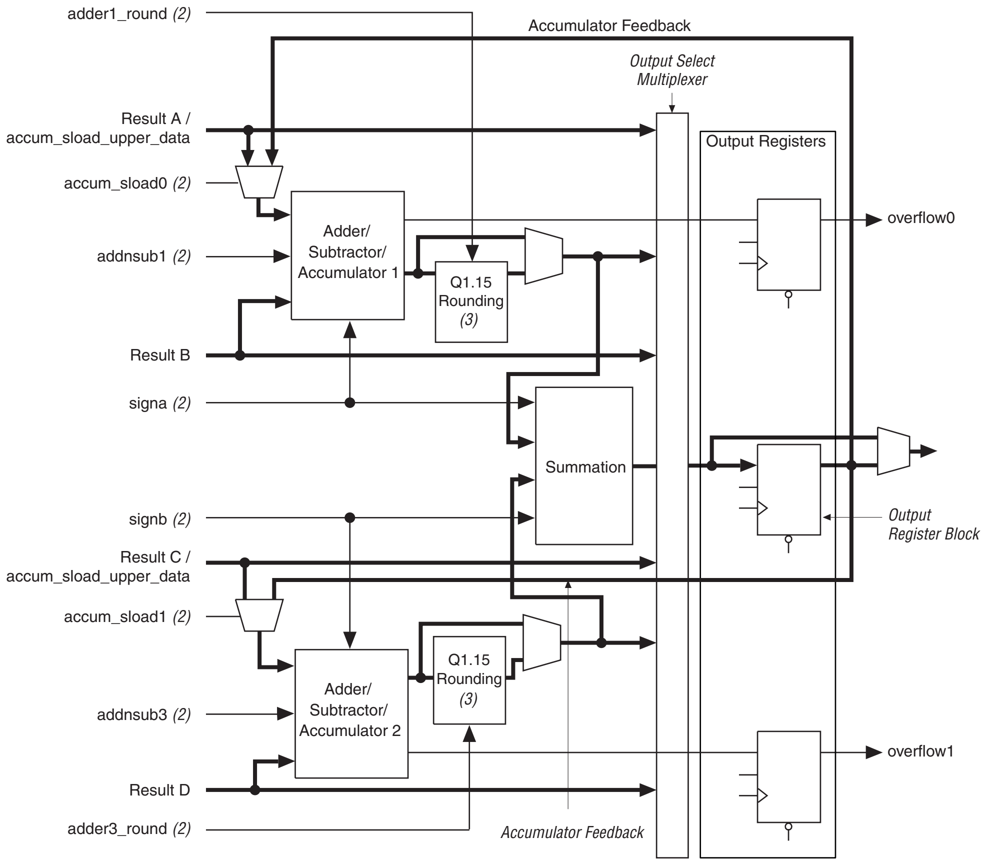
The adder/output block can be configured as:

- An output interface
- An accumulator which can be optionally loaded
- A 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, 9 × 9 complex multiplier, or 18 × 18 complex multiplier

The output select multiplexer sets the output configuration of the DSP block. The output registers can be used to register the output of the adder/output block.

☞ The adder/output block cannot be used independently from the multiplier.

Figure 6–7. Adder/Output Block Architecture *Note (1)*



Notes to Figure 6–7:

(1) The adder/output block is in 18 × 18 mode. In 9 × 9 mode, there are four adder/subtractor blocks and two summation blocks.

(2) You can send these signals through a pipeline register. The pipeline length can be set to 1 or 2.

(3) Q1.15 inputs are not available in 9 × 9 or 36 × 36 modes.

Adder/Subtractor/Accumulator Block

The adder/subtractor/accumulator block is the first level adder stage of the adder/output block. This block can be configured 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 6–7. The accumulator can be set up to perform addition only, subtraction only or the `addnsub` signal can be used to dynamically control the accumulation direction. A logic 1 value on the `addnsub` signal indicates that the accumulator is performing addition while a logic 0 value indicates subtraction.

Each accumulator can be cleared by either clearing the DSP block output register or by using the `accum_sload` signal. The accumulator clear using the `accum_sload` signal is independent from the resetting of the output registers so the accumulation can be cleared and a new one can begin without losing any clock cycles. The `accum_sload` signal controls a feedback multiplexer that specifies that the output of the multiplier should be summed with a zero instead of the accumulator feedback path.

The accumulator can also be initialized/preloaded with a non-zero value using the `accum_sload` signal and the `accum_sload_upper_data` bus with one clock cycle latency. Preloading the accumulator is done by adding the result of the multiplier with the value specified on the `accum_sload_upper_data` bus. As in the case of the accumulator clearing, the `accum_sload` signal specifies to the feedback multiplexer that the `accum_sload_upper_data` signal should feed the accumulator instead of the accumulator feedback signal. The `accum_sload_upper_data` signal only loads the upper 36-bits of the accumulator. To load the entire accumulator, the value for the lower 16-bits must be sent through the multiplier feeding that accumulator with the multiplier set to perform a multiplication by one.

The overflow signal will go high on the positive edge of the clock when the accumulator detects an overflow or underflow. The overflow signal will stay high for only one clock cycle after an overflow or underflow is detected even if the overflow or underflow condition is still present. A latch external to the DSP block has to be used to preserve the overflow signal indefinitely or until the latch is cleared.

The DSP blocks support Q1.15 input format saturation and rounding in each accumulator. The following signals are available that can control if saturation or rounding or both is performed to the output of the accumulator:

- `accum_round`
- `accum_saturation`
- `accum_is_saturated` output

Each DSP block has two sets of accum_round and accum_saturation signals which control if rounding or saturation is performed on the accumulator output respectively (one set of signals for each accumulator). Rounding and saturation of the accumulator output is only available when implementing an 16 × 16 multiplier-accumulator to conform to the bit widths required for Q1.15 input format computation. A logic 1 value on the accum_round and accum_saturation signal indicates that rounding or saturation is performed while a logic 0 indicates that no rounding or saturation is performed. A logic 1 value on the accum_is_saturated output signal tells the user that saturation has occurred to the result of the accumulator.

Figure 6–10 shows the DSP block configured to perform multiplier-accumulator operations.

Adder/Subtractor

The addnsub1 or addnsub3 signals specify whether the user is performing addition or subtraction. A logic 1 value on the addnsub1 or addnsub3 signals indicates that the adder/subtractor is performing addition while a logic 0 value indicates subtraction. These signals can be dynamically controlled using logic external to the DSP block. If the first stage is configured as a subtractor, the output is A – B and C – D.

The adder/subtractor block share the same signa and signb signals as the multiplier block. The signa and signb signals can be pipelined with a latency of one or two clock cycles or not.

The DSP blocks support Q1.15 input format rounding (not saturation) after each adder/subtractor. The addnsub1_round and addnsub3_round signals determine if rounding is performed to the output of the adder/subtractor.

The addnsub1_round signal controls the rounding of the top adder/subtractor and the addnsub3_round signal controls the rounding of the bottom adder/subtractor. Rounding of the adder output is only available when implementing an 16 × 16 multiplier-adder to conform to the bit widths required for Q1.15 input format computation. A logic 1 value on the addnsub_round signal indicates that rounding is performed while a logic 0 indicates that no rounding is performed.

Summation Block

The output of the adder/subtractor block feeds an optional summation block, which is an adder block that sums the outputs of both adder/subtractor blocks. The summation block is used when more than two multiplier results are summed. This is useful in applications such as FIR filtering.

Output Select Multiplexer

The outputs of the different elements of the adder/output block are routed through an output select multiplexer. Depending on the operational mode of the DSP block, the output multiplexer selects whether the outputs of the DSP blocks comes from the outputs of the multiplier block, the outputs of the adder/subtractor/accumulator, or the output of the summation block. The output select multiplier configuration is set automatically by software based on the DSP block operational mode specified by the user.

Output Registers

You can use the output registers to register the DSP block output. The following signals can control each output register within the DSP block:

- `clock[3..0]`
- `ena[3..0]`
- `aclr[3..0]`

The output registers can be used in any DSP block operational mode.

☞ The output registers form part of the accumulator in the multiply-accumulate mode.

See the *Stratix II Architecture* chapter in Volume 1 of the *Stratix II Device Handbook* for more information on the DSP block routing and interface.

Operational Modes

The DSP block can be used in one of four basic operational modes, or a combination of two modes, depending on the application needs. Table 6–5 shows the four basic operational modes and the number of multipliers that can be implemented within a single DSP block depending on the mode.

The Quartus II software includes megafunctions used to control the mode of operation of the multipliers. After the user has made the appropriate parameter settings using the megafunction's MegaWizard Plug-In Manager, the Quartus II software automatically configures the DSP block.

Stratix II DSP blocks can operate in different modes simultaneously. For example, a single DSP block can be broken down to operate a 9 × 9 multiplier as well as an 18 × 18 multiplier-adder where both multiplier's input a and input b have the same sign representations. This increases DSP block resource efficiency and allows you to implement more

Table 6–5. DSP Block Operational Modes

Mode	Number of Multipliers		
	9 × 9	18 × 18	36 × 36
Simple multiplier	Eight multipliers with eight product outputs	Four multipliers with four product outputs	One multiplier
Multiply accumulate	-	Two 52-bit multiply-accumulate blocks	-
Two-multiplier adder	Four two-multiplier adder (two 9 × 9 complex multiply)	Two two-multiplier adder (one 18 × 18 complex multiply)	-
Four-multiplier adder	Two four-multiplier adder	One four-multiplier adder	-

multipliers within a Stratix II device. The Quartus II software will automatically place multipliers that can share the same DSP block resources within the same block.

Additionally, you can set up each Stratix II DSP block to dynamically switch between the following three modes:

- Up to four 18-bit independent multipliers
- Up to two 18-bit multiplier-accumulators
- One 36-bit multiplier

Each half of a Stratix II DSP block has separate mode control signals, which allows you to implement multiple 18-bit multipliers or multiplier-accumulators within the same DSP block and dynamically switch them independently (if they are in separate DSP block halves). If the design requires a 36-bit multiplier, you must switch the entire DSP block to accommodate the it since the multiplier requires the entire DSP block. The smallest input bit width that supports dynamic mode switching is 18 bits.

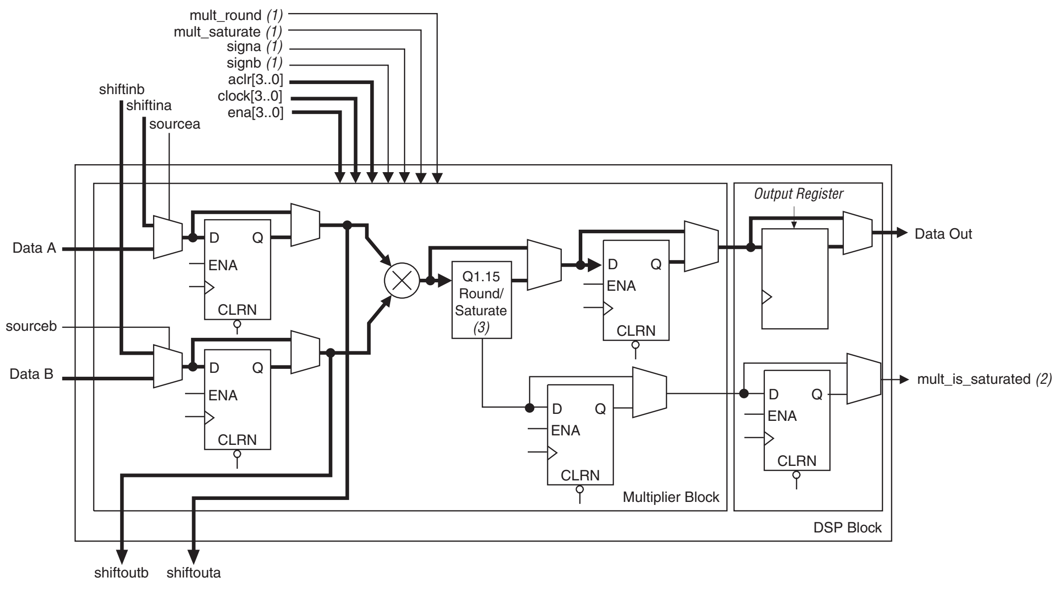
Simple Multiplier Mode

In simple multiplier mode, the DSP block performs individual multiplication operations for general-purpose multipliers and for applications such as computing equalizer coefficient updates which require many individual multiplication operations.

9- & 18-Bit Multipliers

Each DSP block multiplier can be configured for 9- or 18-bit multiplication. A single DSP block can support up to eight individual 9 × 9 multipliers or up to four individual 18 × 18 multipliers. For operand widths up to 9-bits, a 9 × 9 multiplier will be implemented and for operand widths from 10- to 18-bits, an 18 × 18 multiplier will be implemented. Figure 6–8 shows the DSP block in the simple multiplier operation mode.

Figure 6–8. Simple Multiplier Mode



Notes to Figure 6–8:

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

(2) This signal has the same latency as the data path.

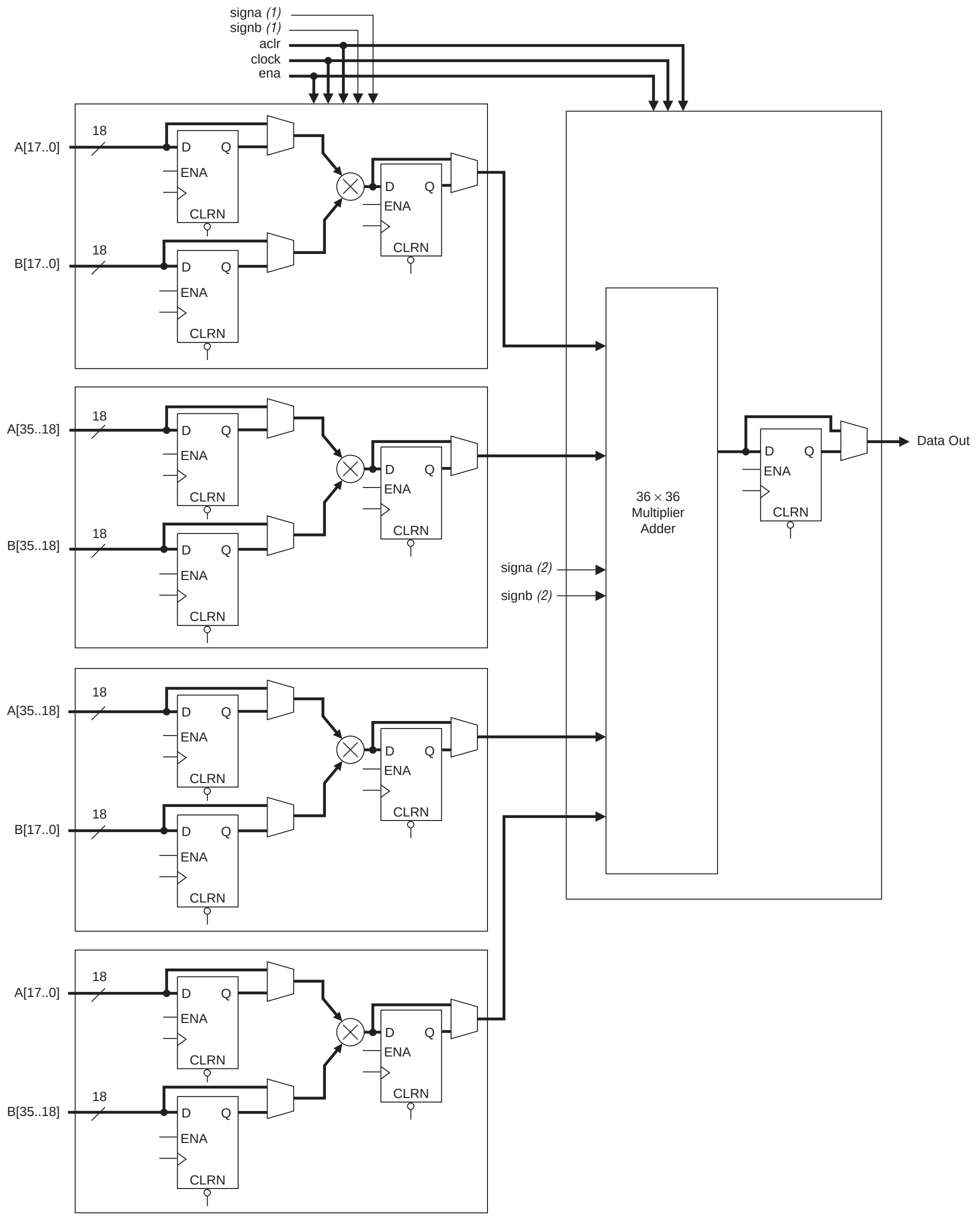
(3) The rounding and saturation is only supported in 18- × 18-bit signed multiplication for Q1.15 inputs.

The multiplier operands can accept signed integers, unsigned integers or a combination of both. The `signa` and `signb` signals can be changed dynamically and can be registered in the DSP block. Additionally, the multiplier inputs and result can be registered independently. The pipeline registers within the DSP block can be used to pipeline the multiplier result, increasing the performance of the DSP block.

36-Bit Multiplier

The 36-bit multiplier is also a simple multiplier mode but uses the entire DSP block, including the adder/output block to implement the 36 × 36-bit multiplication operation. The device inputs 18-bit sections of the 36-bit input into the four 18-bit multipliers. The adder/output block adds the partial products obtained from the multipliers using the summation block. Pipeline registers can be used between the multiplier stage and the summation block to speed up the multiplication. The 36 × 36-bit multiplier supports signed, unsigned as well as mixed sign multiplication. Figure 6–9 shows the DSP block configured to implement a 36-bit multiplier.

Figure 6–9. 36-Bit Multiplier



Notes to Figure 6–9:

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

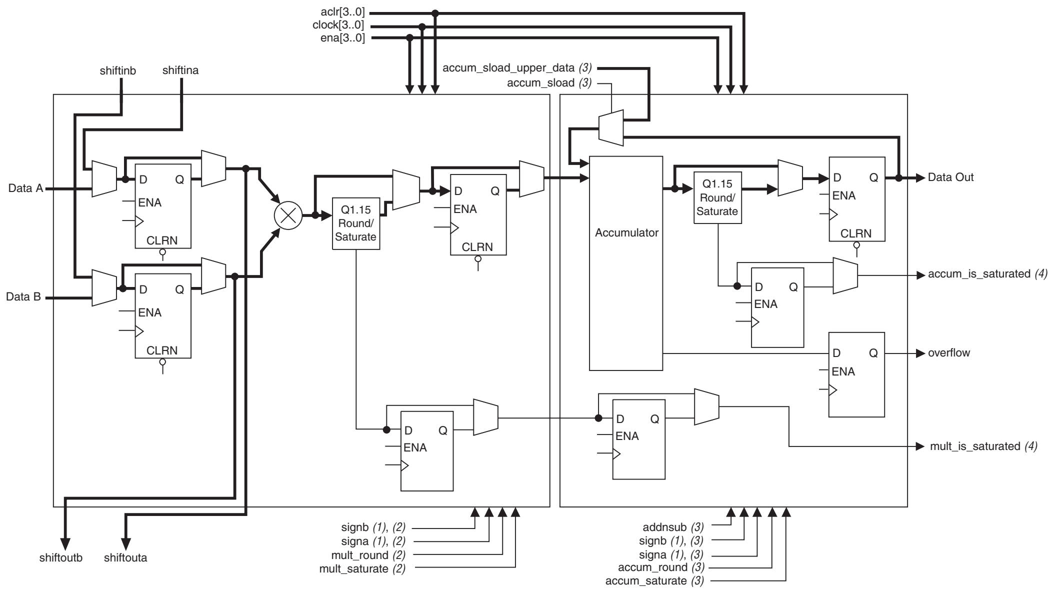
(2) These signals are either not registered, registered once, or registered twice to match the data path pipeline.

The 36-bit multiplier is useful for applications requiring more than 18-bit precision, for example, for mantissa multiplication of precision floating-point arithmetic applications.

Multiply Accumulate Mode

In multiply accumulate mode, the output of the multiplier stage feeds the adder/output block which is configured as an accumulator or subtractor. Figure 6–10 shows the DSP block configured to operate in multiply accumulate mode.

Figure 6–10. Multiply Accumulate Mode



Notes to Figure 6–10:

(1) The signa and signb signals are the same in the multiplier stage and the adder/output block.
(2) These signals are not registered or registered once to match the data path pipeline.
(3) You can send these signals through either one or two pipeline registers.
(4) These signals match the latency of the data path.

A single DSP block can implement up to two independent 18-bit multiplier accumulators. The Quartus II software implements smaller multiplier accumulators by tying the unused lower-order bits of the 18-bit multiplier to ground.

The multiplier accumulator output can be up to 52-bits wide to account for a 36-bit multiplier 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` signal can be used to clear the accumulator so that a new accumulation operation can begin without losing any clock cycles. This signal can be unregistered or registered once or twice. The `accum_sload` signal can also be used to preload the accumulator with a value specified on the `accum_sload_upper_data` signal with a one clock cycle penalty. The `accum_sload_upper_data` signal only loads the upper 36-bits (bits `[51..16]` of the accumulator). To load the entire accumulator, the value for the lower 16-bits (bits `[15..0]`) must be sent through the multiplier feeding that accumulator with the multiplier set to perform a multiplication by one. Bits `[17..16]` are overlapped by both the `accum_sload_upper_data` signal and the multiplier output. Either one of these signals can be used to load bits `[17..16]`.

The overflow signal indicates an overflow or underflow in the accumulator. This signal gets updated every clock cycle due to a new accumulation operation every cycle. To preserve the signal, an external latch can be used. The `addnsub` signal can be used to specify if an accumulation or subtraction is performed dynamically.

☞ The DSP block can implement just an accumulator (without multiplication) by specifying a multiply by one at the multiplier stage followed by an accumulator to force the Quartus II software to implement the function within the DSP block.

Multiply Add Mode

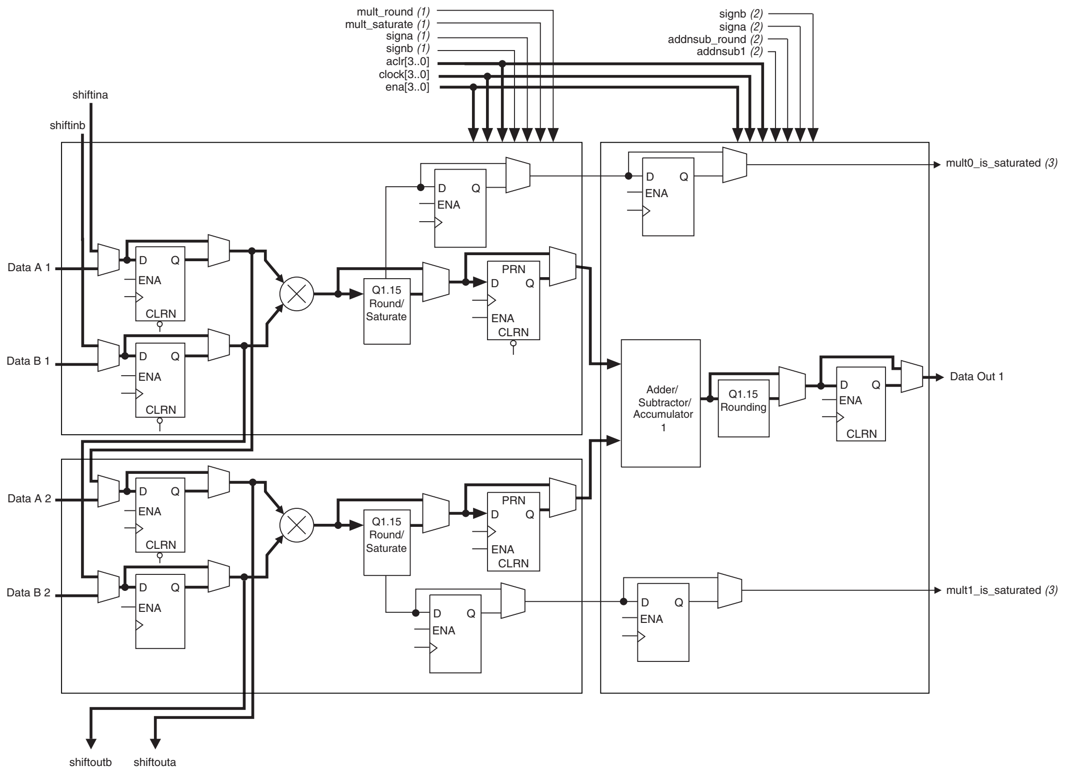
In multiply add mode, the output of the multiplier stage feeds the adder/output block which is configured as an adder or subtractor to sum or subtract the outputs of two or more multipliers. The DSP block can be configured to implement either a two-multiply add (where the outputs of two multipliers are added/subtracted together) or a four-multiply add function (where the outputs of four multipliers are added or subtracted together).

☞ The adder block within the DSP block can only be used if it follows multiplication operations.

Two-Multiplier Adder

In the two-multiplier adder configuration, the DSP block can implement four 9-bit or smaller multiplier adders or two 18-bit multiplier adders. The adders can be configured to take the sum of both multiplier outputs or the difference of both multiplier outputs. The user has the option to vary the summation/subtraction operation dynamically. These multiply add functions are useful for applications such as FFTs and complex FIR filters. Figure 6–11 shows the DSP block configured in the two-multiplier adder mode.

Figure 6–11. Two-Multiplier Adder Mode



Notes to Figure 6–11:

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

(2) You can send these signals through a pipeline register. The pipeline length can be set to 1 or 2.

(3) These signals match the latency of the data path.

Complex Multiply

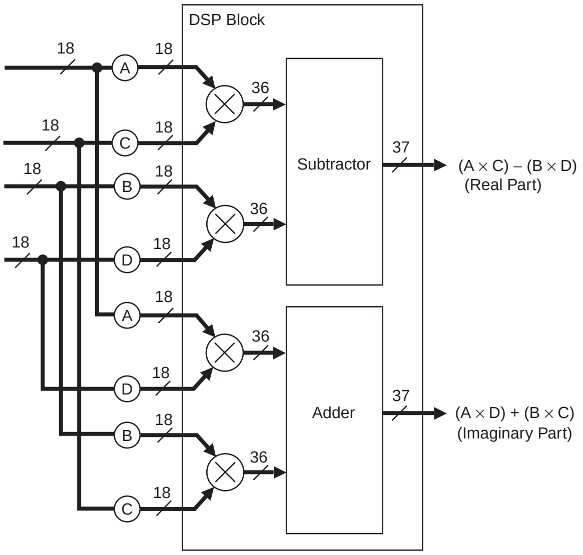
The DSP block can be configured to implement complex multipliers using the two-multiplier adder mode. A single DSP block can implement one 18 × 18-bit complex multiplier or two 9 × 9-bit complex multipliers.

A complex multiplication can be written as:

$$(a + \mathrm{j}b) \times (c + \mathrm{j}d) = ((a \times c) - (b \times d)) + \mathrm{j}\,((a \times d) + (b \times c))$$

To implement this complex multiplication within the DSP block, the real part $((a \times c) - (b \times d))$ is implemented using two multipliers feeding one subtractor block while the imaginary part $((a \times d) + (b \times c))$ is implemented using another two multipliers feeding an adder block, for data up to 18-bits. Figure 6–12 shows an 18-bit complex multiplication. For data widths up to 9-bits, a DSP block can perform two separate complex multiplication operations using eight 9-bit multipliers feeding four adder/subtractor/accumulator blocks. Resources external to the DSP block must be used to route the correct real and imaginary input components to the appropriate multiplier inputs to perform the correct computation for the complex multiplication operation.

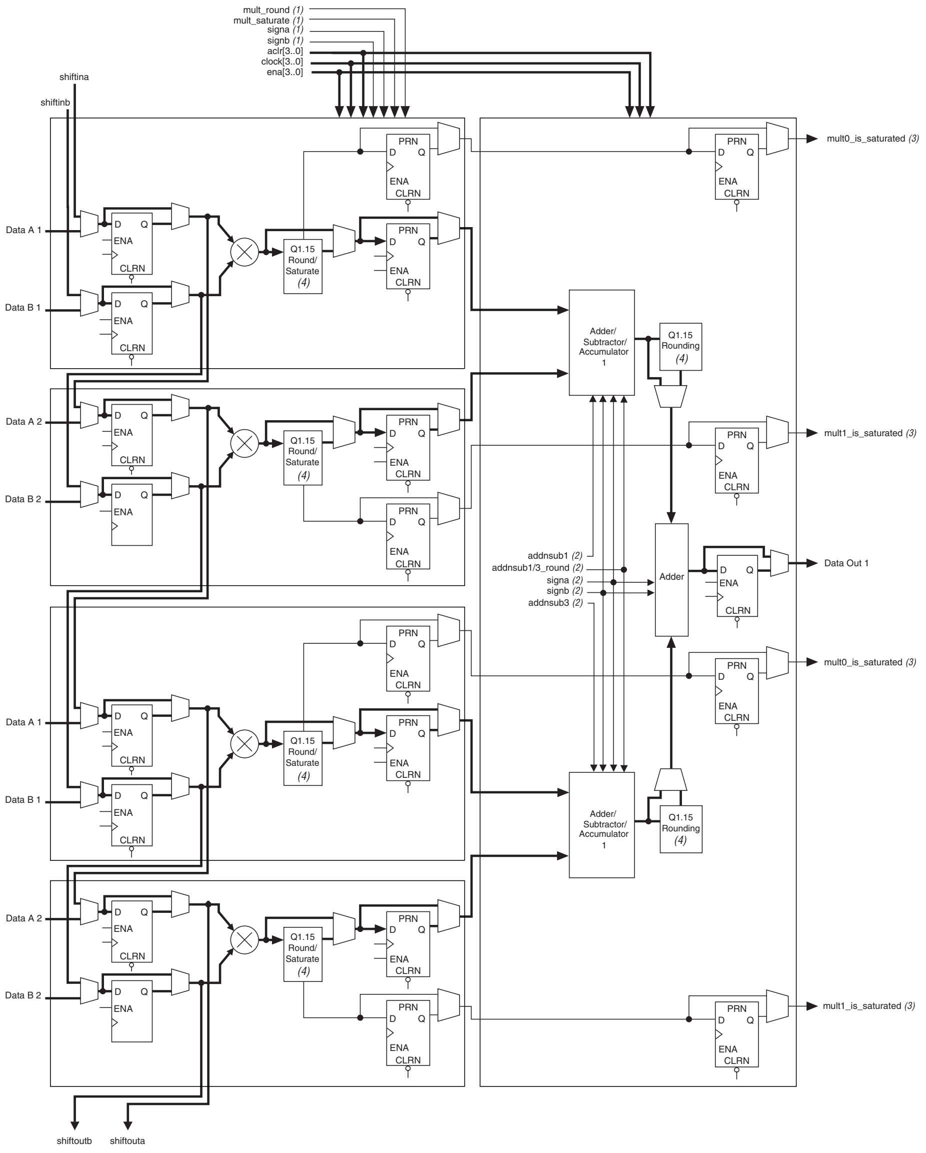
Figure 6–12. Complex Multiplier Using Two-Multiplier Adder Mode



Four-Multiplier Adder

In the four-multiplier adder configuration, the DSP block can implement one 18 × 18 or two individual 9 × 9 multiplier adders. These modes are useful for implementing one-dimensional and two-dimensional filtering applications. The four-multiplier adder is performed in two addition stages. The outputs of two of the four multipliers are initially summed in the two first-stage adder/subtractor/accumulator blocks. The results of these two adder/subtractor/accumulator blocks are then summed in the final stage summation block to produce the final four-multiplier adder result. Figure 6–13 shows the DSP block configured in the four-multiplier adder mode.

Figure 6–13. Four-Multiplier Adder Mode



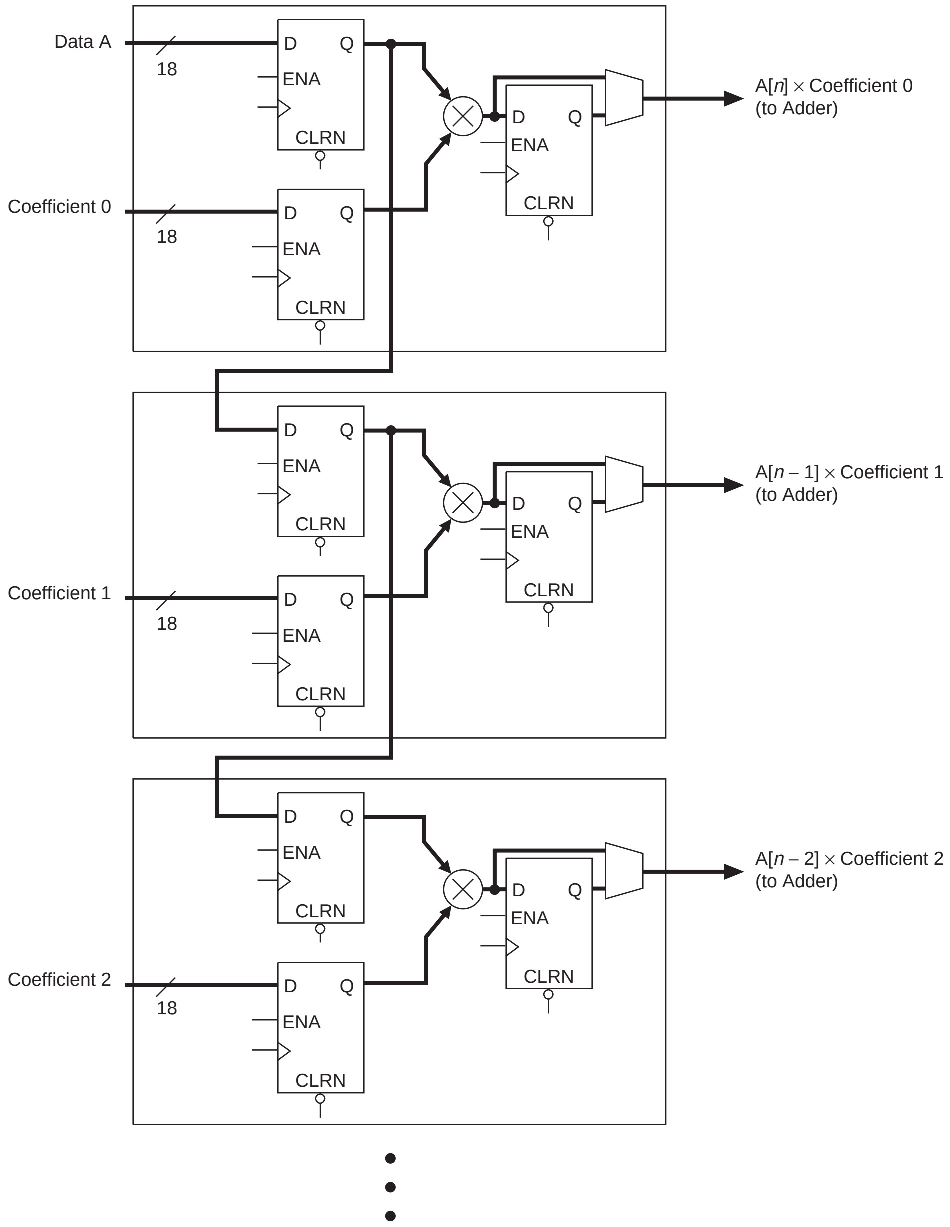
Notes to Figure 6–13:

(1) These signals are not registered or registered once to match the data path pipeline.
(2) You should send these signals through the pipeline register to match the latency of the data path.
(3) These signals match the latency of the data path.
(4) The rounding and saturation is only supported in 18- × 18-bit signed multiplication for Q1.15 inputs.

FIR Filter

The four-multiplier adder mode can be used to implement FIR filter and complex FIR filter applications. To do this, the DSP block is set up in a four-multiplier adder mode with one set of input registers configured as shift registers using the dedicated shift register chain. The set of input registers configured as shift registers will contain the input data while the inputs configured as regular inputs will hold the filter coefficients. Figure 6–14 shows the DSP block configured in the four-multiplier adder mode using input shift registers.

Figure 6–14. FIR Filter Implemented Using the Four-Multiplier Adder Mode with Input Shift Registers


Data A
18
D
Q
ENA
CLRN
Coefficient 0
18
A[n] × Coefficient 0 (to Adder)
Coefficient 1
18
A[n – 1] × Coefficient 1 (to Adder)
Coefficient 2
18
A[n – 2] × Coefficient 2 (to Adder)

The built-in input shift register chain within the DSP block eliminates the need for shift registers externally to the DSP block in logic elements (LEs). This architecture feature simplifies the filter design and improves the filter performance because all the filter circuitry is localized within the DSP block.

☞ Input shift registers for the 36-bit simple multiplier mode have to be implemented using external registers to the DSP block.

A single DSP block can implement a four tap 18-bit FIR filter. For filters larger than four taps, the DSP blocks can be cascaded with additional adder stages implemented using LEs.

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.

See Quartus II On-Line Help for instructions on using the megafunctions and the MegaWizard Plug-In Manager.

For more information, see the *Synthesis* section in Volume 1 of the *Quartus II Development Software Handbook*.

Conclusion

The Stratix II device DSP blocks are optimized to support DSP applications requiring high data throughput such as FIR filters, FFT functions and encoders. These DSP blocks are flexible and can be configured to implement one of several operational modes to suit a particular application. The built-in shift register chain, adder/subtractor/accumulator block and the summation block minimizes the amount of external logic required to implement these functions, resulting in efficient resource utilization and improved performance and data throughput for DSP applications. The Quartus II software, together with the LeonardoSpectrum and Synplify software provide a complete and easy-to-use flow for implementing these multiplier functions in the DSP blocks.

Section V. Configuration & Remote System Upgrades

This section provides configuration information for all of the supported configuration schemes for Stratix® II 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 designers how to perform remote and local upgrades for their designs.

This section contains the following chapters:

Revision History

The table below shows the revision history for Chapters 7 through 9.

Chapter	Date / Version	Changes Made
7	January 2005, v2.0	● Corrected data rate for FPP configuration using a MAX II device without Stratix II decompression nor the design security feature. ● Updated Figure 7–6. ● Updated Tables 7–3, 7–6, 7–7, 7–11, 7–14, and 7–17.
	July 2004, v1.1	● Removed reference to PLMSEPC-8 adapter from “Programming Serial Configuration Devices” section. ● Updated Tables 7–7 and 7–15. ● Updated Figure 7–2. ● Updated “VCCSEL Pin”, “Configuration Devices”, and “Design Security Using Configuration Bitstream Encryption” sections. ● Added timing specifications for `CONF_DONE` high to user mode with `CLKUSR` option on.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.
8	January 2005, v2.0	Added tables in the “Quartus II Software Support” section.
	July 2004, v1.1	Updated Table 8–1.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.
9	January 2005, v2.0	Updated the “Introduction” and “I/O Voltage Support in JTAG Chain” sections.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.

7. Configuring Stratix II Devices

SII52007-2.1

Introduction

Stratix® II devices use SRAM cells to store configuration data. Since SRAM memory is volatile, configuration data must be downloaded to Stratix II devices each time the device powers up. Stratix II devices can be configured using one of five configuration schemes: the fast passive parallel (FPP), active serial (AS), passive serial (PS), passive parallel asynchronous (PPA), and Joint Test Action Group (JTAG) configuration schemes. All configuration schemes use either an external controller (for example, a MAX® II device or microprocessor) or a configuration device.

Configuration Devices

The Altera enhanced configuration devices (EPC16, EPC8, and EPC4) support a single-device configuration solution for high-density devices 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.

☞ For information on enhanced configuration devices, see the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet* chapter and the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook*.

The Altera serial configuration devices (EPCS4 and EPCS1) support a single-device configuration solution for Stratix II devices and are used in the AS configuration scheme. Serial configuration devices offer a low cost, low pin count configuration solution.

☞ For information on serial configuration devices, see the *Serial Configuration Devices (EPCS1 & EPCS4) Data Sheet* chapter.

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.

☞ For more information on EPC2, EPC1, and EPC1441 configuration devices, see the *Configuration Devices for SRAM-Based LUT Devices Data Sheet* chapter.

The configuration scheme is selected by driving the Stratix II device MSEL pins either high or low as shown in Table 7–1. The MSEL pins are powered by the V_{CCPD} power supply of the bank they reside in. The MSEL[3..0] pins have 5-kΩ internal pull-down resistors that are always active. During POR and during reconfiguration, the MSEL pins have to be at LVTTL V_{IL} and V_{IH} levels to be considered a logic low and logic high.

☞ To avoid any problems with detecting an incorrect configuration scheme, hard-wire the MSEL[] pins to V_{CCPD} and GND, without any pull-up or pull-down resistors. Do not drive the MSEL[] pins by a microprocessor or another device.

Table 7–1. Stratix II Configuration Schemes

Configuration Scheme	MSEL3	MSEL2	MSEL1	MSEL0
Fast passive parallel (FPP)	0	0	0	0
Passive parallel asynchronous (PPA)	0	0	0	1
Passive serial (PS)	0	0	1	0
Remote system upgrade FPP *(1)*	0	1	0	0
Remote system upgrade PPA *(1)*	0	1	0	1
Remote system upgrade PS *(1)*	0	1	1	0
Fast AS (40 MHz) *(2)*	1	0	0	0
Remote system upgrade fast AS (40 MHz) *(2)*	1	0	0	1
FPP with decompression and/or design security feature enabled *(3)*	1	0	1	1
Remote system upgrade FPP with decompression and/or design security feature enabled *(1)*, *(3)*	1	1	0	0
AS (20 MHz) *(2)*	1	1	0	1
Remote system upgrade AS (20 MHz) *(2)*	1	1	1	0
JTAG-based configuration *(5)*	*(4)*	*(4)*	*(4)*	*(4)*

Notes to Table 7–1:

(1) These schemes require that you drive the RUnLU pin to specify either remote update or local update. For more information about remote system upgrades in Stratix II devices, see Chapter 8, Remote System Upgrades with Stratix II Devices in Volume 2 of the *Stratix II Device Handbook*.

(2) Only the EPCS16 and EPCS64 devices support up to a 40 MHz DCLK. Other EPCS devices support up to a 20 MHz DCLK. See the *Serial Configuration Devices Data Sheet* for more information.

(3) These modes are only supported when using a MAX II device or a microprocessor with flash memory for configuration. In these modes, the host system must output a DCLK that is 4× the data rate.

(4) Do not leave the MSEL pins floating. Connect them to V_{CCPD} or ground. 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.

(5) JTAG-based configuration takes precedence over other configuration schemes, which means MSEL pin settings are ignored.

Stratix II devices offer the design security, decompression, and remote system upgrade features. Design security using configuration bitstream encryption is available in Stratix II devices, which protects your designs. Stratix II devices can receive a compressed configuration bit stream and decompress this data in real-time, reducing storage requirements and configuration time. You can make real-time system upgrades from remote locations of your Stratix II designs by using the remote system upgrade feature.

Table 7–2 shows the approximate uncompressed configuration file sizes for Stratix II devices.

Table 7–2. Stratix II .rbf Sizes Notes (1), (2)

Device	Data Size (MBits)	Data Size (MBytes)
EP2S15	5.0	0.625
EP2S30	10.1	1.2625
EP2S60	17.1	2.1375
EP2S90	27.5	3.4375
EP2S130	39.6	4.95
EP2S180	52.4	6.55

Notes to Table 7–2:
(1) These values are preliminary.
(2) **.rbf**: Raw Binary File.

Use the data in Table 7–2 to estimate the file size before design compilation. Different configuration file formats, such as a Hexidecimal (**.hex**) or Tabular Text File (**.ttf**) format, will have different file sizes. However, for any specific version of the Quartus® II software, any design targeted for the same device will have the same uncompressed configuration file size. If you are using compression, the file size can vary after each compilation since the compression ratio is dependent on the design.

This chapter explains the Stratix II device configuration features and describes how to configure Stratix II devices using the supported configuration schemes. This chapter configuration pin descriptions and the Stratix II device configuration file format. In this chapter, the generic term device(s) includes all Stratix II devices.

For more information on setting device configuration options or creating configuration files, see *Software Settings* in Volume 2 of the *Configuration Handbook*.

Configuration Features

Stratix II devices offer configuration data decompression to reduce configuration file storage, design security using data encryption to protect your designs, and remote system upgrades to allow for remotely updating your Stratix II designs. Table 7–3 summarizes which configuration features can be used in each configuration scheme.

Table 7–3. Stratix II Configuration Features

Configuration Scheme	Configuration Method	Design Security	Decompression	Remote System Upgrade
FPP	MAX II device or a Microprocessor with flash memory	✓ (1)	✓ (1)	✓
	Enhanced Configuration Device		✓ (2)	✓
AS	Serial Configuration Device	✓	✓	✓ (3)
PS	MAX II device or a Microprocessor with flash memory	✓	✓	✓
	Enhanced Configuration Device	✓	✓	✓
	Download cable	✓	✓	
PPA	MAX II device or a Microprocessor with flash memory			✓
JTAG	MAX II device or a Microprocessor with flash memory			

Notes to Table 7–3:

(1) In these modes, the host system must send a DCLK that is 4× the data rate.
(2) The enhanced configuration device decompression feature is available, while the Stratix II decompression feature is not available.
(3) Only remote update mode is supported when using the AS configuration scheme. Local update mode is not supported.

Configuration Data Decompression

Stratix II devices support configuration data decompression, which saves configuration memory space and time. This feature allows you to store compressed configuration data in configuration devices or other memory and transmit this compressed bit stream to Stratix II devices. During configuration, the Stratix II device decompresses the bit stream in real time and programs its SRAM cells.

Preliminary data indicates that compression typically reduces configuration bit stream size by 35 to 55%.

Stratix II devices support decompression in the FPP (when using a MAX II device/microprocessor + flash), AS, and PS configuration schemes. Decompression is not supported in the PPA configuration scheme nor in JTAG-based configuration.

☞ When using FPP mode, the intelligent host must provide a `DCLK` that is 4× the data rate. Therefore, the configuration data must be valid for four `DCLK` cycles.

The decompression feature supported by Stratix II devices is different from the decompression feature in enhanced configuration devices (EPC16, EPC8, and EPC4 devices), although they both use the same compression algorithm. The data decompression feature in the enhanced configuration devices allows them to store compressed data and decompress the bitstream before transmitting it to the target devices. When using Stratix II devices in FPP mode with enhanced configuration devices, the Stratix II decompression feature is not available, but the enhanced configuration device decompression feature is.

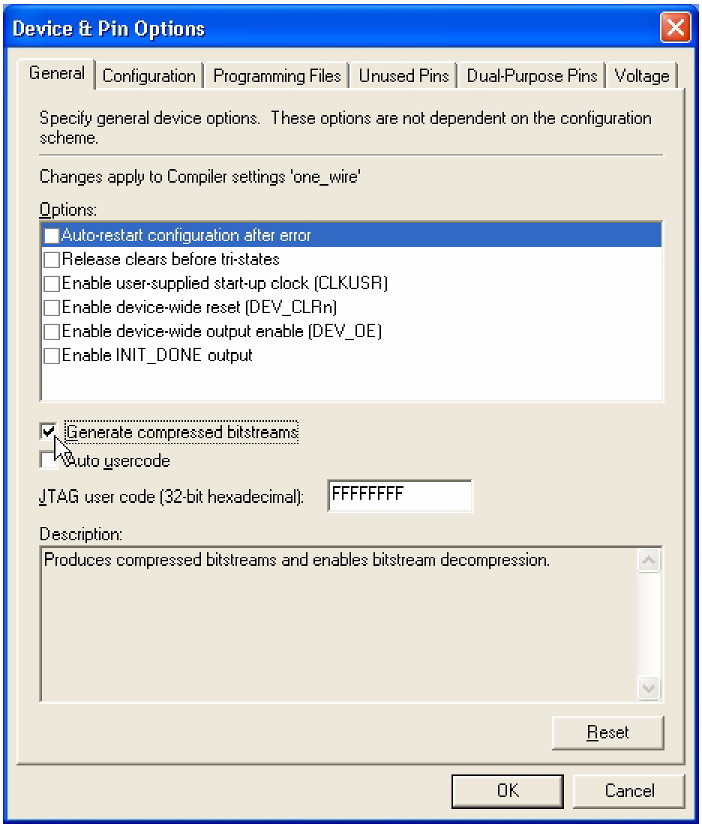
In PS mode, you should use the Stratix II decompression feature since sending compressed configuration data reduces configuration time. You should not use both the Stratix II device and the enhanced configuration device decompression features simultaneously. The compression algorithm is not intended to be recursive and could expand the configuration file instead of compressing it further.

When you enable compression, the Quartus II software generates configuration files with compressed configuration data. This compressed file reduces the storage requirements in the configuration device or flash memory, and decreases the time needed to transmit the bitstream to the Stratix II device. The time required by a Stratix II device to decompress a configuration file is less than the time needed to transmit the configuration data to the device.

There are two methods to enable compression for Stratix II bitstreams: before design compilation (in the **Compiler Settings** menu) and after design compilation (in the **Convert Programming Files** window).

To enable compression in the project's compiler settings, select **Device** under the **Assignments** menu to bring up the **Settings** window. After selecting your Stratix II device, open the **Device & Pin Options** window, and in the **General** settings tab enable the check box for **Generate compressed bitstreams** (as shown in Figure 7–1).

Figure 7–1. Enabling Compression for Stratix II Bitstreams in Compiler Settings



Specify general device options. These options are not dependent on the configuration scheme.

Changes apply to Compiler settings 'one_wire'

- [ ] Auto-restart configuration after error
- [ ] Release clears before tri-states
- [ ] Enable user-supplied start-up clock (CLKUSR)
- [ ] Enable device-wide reset (DEV_CLRn)
- [ ] Enable device-wide output enable (DEV_OE)
- [ ] Enable INIT_DONE output
- [✓] Generate compressed bitstreams
- [ ] Auto usercode

JTAG user code (32-bit hexadecimal): FFFFFFFF

Description: Produces compressed bitstreams and enables bitstream decompression.

Compression can also be enabled when creating programming files from the **Convert Programming Files** window.

1. Click **Convert Programming Files** (File menu).

2. Select the programming file type (POF, SRAM HEXOUT, RBF, or TTF).

3. For POF output files, select a configuration device.

4. In the **Input files to convert** box, select **SOF Data**.

5. Select **Add File** and add a Stratix II device SOF(s).

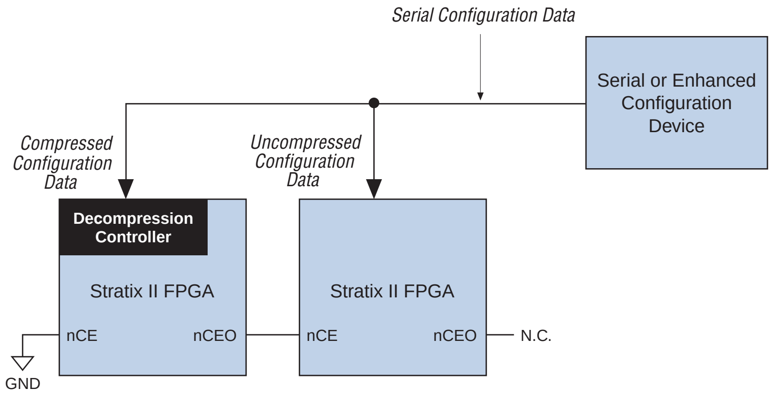
6. Select the name of the file you added to the **SOF Data** area and click **Properties**.

7. Check the **Compression** check box.

When multiple Stratix II devices are cascaded, the compression feature can be selectively enabled for each device in the chain if you are using a serial configuration scheme. Figure 7–2 depicts a chain of two Stratix II devices. The first Stratix II device has compression enabled and therefore receives a compressed bit stream from the configuration device. The second Stratix II device has the compression feature disabled and receives uncompressed data.

In a multi-device FPP configuration chain all Stratix II devices in the chain must either enable of disable the decompression feature. You can not selectively enable the compression feature for each device in the chain because of the `DATA` and `DCLK` relationship.

Figure 7–2. Compressed and Uncompressed Configuration Data in the Same Configuration File


Serial Configuration Data
Serial or Enhanced Configuration Device
Compressed Configuration Data
Uncompressed Configuration Data
Decompression Controller
Stratix II FPGA
nCE
nCEO
Stratix II FPGA
nCE
nCEO
N.C.
GND

You can generate programming files for this setup from the **Convert Programming Files** window (File menu) in the Quartus II software.

Design Security Using Configuration Bitstream Encryption

Stratix II devices are the industry's first devices with the ability to decrypt a configuration bitstream using the Advanced Encryption Standard (AES) algorithm—the most advanced encryption algorithm available today. When using the design security feature, a 128-bit security key is stored in the Stratix II device. In order to successfully configure a Stratix II device which has the design security feature enabled, it must be configured with a configuration file that was encrypted using the same 128-bit security key. The security key can be stored in non-volatile memory inside the Stratix II device. This non-volatile memory does not require any external devices, such as a battery back-up, for storage.

An encryption configuration file is the same size as a non-encryption configuration file. When using a serial configuration scheme such as passive serial (PS) or active serial (AS), configuration time is the same whether or not the design security feature is enabled. If the fast passive parallel (FPP) scheme us used with the design security or decompression feature, a 4× `DCLK` is required. This results in a slower configuration time when compared to the configuration time of an FPGA that has neither the design security, nor decompression feature enabled. For more information about this feature, contact Altera applications.

Remote System Upgrade

Stratix II devices feature remote and local update. For more information about this feature, see Chapter 8, Remote System Upgrades with Stratix II Devices in Volume 2 of the *Stratix II Device Handbook.*

V_{CCPD} Pins

Stratix II devices also offer a new power supply, V_{CCPD}, which must be connected to 3.3-V in order to power the 3.3-V/2.5-V buffer available on the configuration input pins and JTAG pins. V_{CCPD} applies to all the JTAG input pins (`TCK`, `TMS`, `TDI`, and `TRST`) and the configuration pins: `nCONFIG`, `DCLK` (when used as an input), `nIO_Pullup`, `DATA[7..0]`, `RUnLU`, `nCE`, `nWS`, `nRS`, `CS`, `nCS` and `CLKUSR`.

V_{CCPD} must ramp-up from 0-V to 3.3-V within 100 ms. If V_{CCPD} is not ramped up within this specified time, your Stratix II device will not configure successfully. If your system does not allow for a V_{CCPD} ramp-up time of 100 ms or less, you must hold `nCONFIG` low until all power supplies are stable.

VCCSEL Pin

The `VCCSEL` pin allows the V_{CCIO} setting (of the banks where the configuration inputs reside) to be independent of the voltage required by the configuration inputs. Therefore, when selecting V_{CCIO}, the V_{IL} and V_{IH} levels driven to the configuration inputs are not a factor.

The configuration input pins (nCONFIG, DCLK (when used as an input), nIO_Pullup, RUnLU, nCE, nWS, nRS, CS, nCS, and CLKUSR) have a dual buffer design. These pins have a 3.3-V/2.5-V input buffer and a 1.8-V/1.5-V input buffer. The VCCSEL input pin selects which input buffer is used during configuration. After configuration, the dual-purpose configuration pins are powered by the V_{CCIO} pins. The 3.3-V/2.5-V input buffer is powered by V_{CCPD}, while the 1.8-V/1.5-V input buffer is powered by V_{CCIO}.

VCCSEL is sampled during power-up. Therefore, the VCCSEL setting cannot change on the fly or during a reconfiguration. The VCCSEL input buffer is powered by V_{CCINT} and has an internal 5-kΩ pull-down resistor that is always active.

☞ VCCSEL must be hardwired to V_{CCPD} or GND.

A logic high selects the 1.8-V/1.5-V input buffer, and a logic low selects the 3.3-V/2.5-V input buffer. VCCSEL should be set to comply with the logic levels driven out of the configuration device or MAX II device or a microprocessor with flash memory.

If you need to support 3.3-V or 2.5-V configuration input voltages, set VCCSEL low. You can set the bank V_{CCIO} that contains the configuration inputs to any supported voltage. If you need to support 1.8-V or 1.5-V configuration input voltages, set VCCSEL to a logic high and the V_{CCIO} of the bank that contains the configuration inputs to 1.8 or 1.5-V.

VCCSEL also sets the POR trip point for I/O bank 8 to ensure that this I/O bank has powered up to the appropriate voltage levels before configuration begins. Upon power-up, the device will not release nSTATUS until V_{CCINT} and V_{CCIO} of bank 8 is above its POR trip point. If you set VCCSEL to ground (logic low), this sets the POR trip point for bank 8 to a voltage consistent with 3.3-V/2.5-V signaling, which means the POR trip point for these I/O banks may be as high as 1.8V. If V_{CCIO} of any of the configuration banks is set to 1.8-V or 1.5-V, the voltage supplied to this I/O bank(s) may never reach the POR trip point, which will cause the device to never begin configuration.

If the V_{CCIO} of I/O bank 8 is set to 1.5-V or 1.8-V and the configuration signals used require 3.3-V or 2.5-V signaling, you should set VCCSEL to V_{CCPD} (logic high) in order to lower the POR trip point to enable successful configuration.

Table 7–4 shows how you should set the VCCSEL depending on the V_{CCIO} setting of bank 8 and your configuration input signaling voltages.

Table 7–4. VCCSEL Setting

VCCIO (bank 8)	Configuration Input Signaling Voltage	VCCSEL
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	VCCPD
3.3-V/2.5-V	1.8-V/1.5-V	Not Supported

The VCCSEL signal does not control TDO, nCEO, or 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 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 inherit the I/O standards specified in the design.

For more information on multi-volt support, including information on using TDO and nCEO in multi-volt systems, refer to the "MultiVolt I/O Interface" section of the *Stratix II Architecture* chapter in Volume 1 of the *Stratix II Handbook*.

Fast Passive Parallel Configuration

Fast passive parallel (FPP) configuration in Stratix II devices is designed to meet the continuously increasing demand for faster configuration times. Stratix II devices are designed with the capability of receiving byte-wide configuration data per clock cycle. Table 7–5 shows the MSEL pin settings when using the FFP configuration scheme.

Table 7–5. Stratix II MSEL Pin Settings for FPP Configuration Schemes (Part 1 of 2)

Configuration Scheme	MSEL3	MSEL2	MSEL1	MSEL0
FPP when not using remote system upgrade or decompression and/or design security feature	0	0	0	0
FPP when using remote system upgrade *(1)*	0	1	0	0
FPP with decompression and/or design security feature enabled *(2)*	1	0	1	1

Table 7–5. Stratix II MSEL Pin Settings for FPP Configuration Schemes (Part 2 of 2)

Configuration Scheme	MSEL3	MSEL2	MSEL1	MSEL0
FPP when using remote system upgrade and decompression and/or design security feature *(1)*, *(2)*	1	1	0	0

Notes to Table 7–5:

(1) These schemes require that you drive the RUnLU pin to specify either remote update or local update. For more information about remote system upgrade in Stratix II devices, see the Chapter 8, Remote System Upgrades with Stratix II Devices in Volume 2 of the *Stratix II Device Handbook*.

(2) These modes are only supported when using a MAX II device or a microprocessor with flash memory for configuration. In these modes, the host system must output a DCLK that is 4× the data rate.

FPP configuration of Stratix II devices can be performed using an intelligent host, such as a MAX II device, or microprocessor, or an Altera enhanced configuration device.

FPP Configuration Using a MAX II Device as an External Host

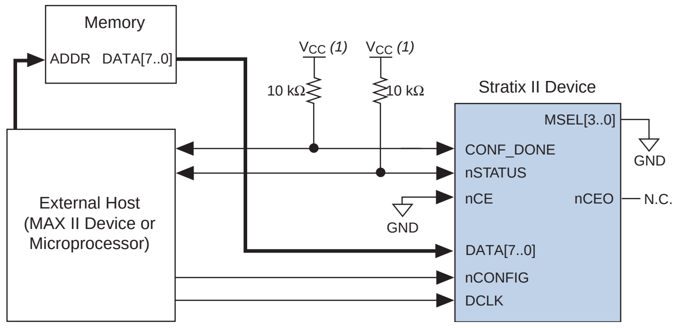
FPP configuration using compression and an external host provides the fastest method to configure Stratix II devices. In the FPP configuration scheme, a MAX II device can be used as an intelligent host that controls the transfer of configuration data from a storage device, such as flash memory, to the target Stratix II device. Configuration data can be stored in RBF, HEX, or TTF format. When using the MAX II devices as an intelligent host, a design that controls the configuration process, such as fetching the data from flash memory and sending it to the device, must be stored in the MAX II device.

☞ If you are using the Stratix II decompression and/or design security feature, the external host must be able to send a DCLK frequency that is 4× the data rate.

The 4× DCLK signal does not require an additional pin and is sent on the DCLK pin. The maximum DCLK frequency is 100 MHz, which results in a maximum data rate of 200 Mbps. If you are not using the Stratix II decompression nor are using the design security feature, the data rate is 8× the DCLK frequency.

Figure 7–3 shows the configuration interface connections between the Stratix II device and a MAX II device for single device configuration.

Figure 7–3. Single Device FPP Configuration Using an External Host



Note to Figure 7–3:

(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for the device. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

Upon power-up, the Stratix II device goes through a Power-On Reset (POR). The POR delay is dependent on the `PORSEL` pin setting; when `PORSEL` is driven low, the POR time is approximately 100 ms, if `PORSEL` is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold `nSTATUS` low, and tri-state all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If `nIO_pullup` is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If `nIO_pullup` is driven high, the weak pull-up resistors are disabled.

The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *DC & Switching Characteristic* chapter in the *Stratix II Device Handbook*.

The configuration cycle consists of three stages: reset, configuration and initialization. While `nCONFIG` or `nSTATUS` are low, the device is in the reset stage. To initiate configuration, the MAX II device must drive the `nCONFIG` pin from low-to-high.

V_{CCINT}, V_{CCIO}, and V_{CCPD} of the banks where the configuration and JTAG pins reside need to be fully powered to the appropriate voltage levels in order to begin the configuration process.

When nCONFIG goes high, the device comes out of reset and releases the open-drain nSTATUS pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once nSTATUS is released, the device is ready to receive configuration data and the configuration stage begins. When nSTATUS is pulled high, the MAX II device places the configuration data one byte at a time on the DATA[7..0] pins.

☞ The Stratix II device receives configuration data on its DATA[7..0] pins and the clock is received on the DCLK pin. Data is latched into the device on the rising edge of DCLK. If you are using the Stratix II decompression and/or design security feature, configuration data is latched on the rising edge of every fourth DCLK cycle. After the configuration data is latched in, it is processed during the following three DCLK cycles.

Data is continuously clocked into the target device until CONF_DONE goes high. The CONF_DONE pin will go high one byte early in parallel configuration (FPP and PPA) modes. The last byte is required for serial configuration (AS and PS) modes. After the device has received the next to last byte of the configuration data successfully, it releases the open-drain CONF_DONE pin, which is pulled high by an external 10-kΩ pull-up resistor. A low-to-high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin.

In Stratix II devices, the initialization clock source is either the Stratix II internal oscillator (typically 10 MHz) or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If the internal oscillator is used, the Stratix II device will provide itself with enough clock cycles for proper initialization. Therefore, if the internal oscillator is the initialization clock source, sending the entire configuration file to the device is sufficient to configure and initialize the device. Driving DCLK to the device after configuration is complete does not affect device operation.

You can also synchronize initialization of multiple devices or to delay initialization by using the CLKUSR option. The **Enable user-supplied start-up clock** (**CLKUSR**) option can be turned on in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. Supplying a clock on CLKUSR will not affect the configuration process. The CONF_DONE pin will go high one byte early in parallel configuration (FPP and PPA) modes. The last byte is required for serial configuration (AS and PS) modes. After the CONF_DONE pin transitions high, CLKUSR will be enabled after the time specified as t_{CD2CU}. After this time period elapses, the Stratix II devices require 299 clock cycles to initialize properly and enter user mode. Stratix II devices support a CLKUSR f_{MAX} of 100 MHz.

An optional `INIT_DONE` pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. This **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If the `INIT_DONE` pin is used, it will be high due to an external 10-kΩ pull-up resistor 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. When initialization is complete, the `INIT_DONE` pin will be released and pulled high. The MAX II device must be able to detect this low-to-high transition which signals the device has entered user mode. When initialization is complete, the device enters user mode. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design.

To ensure `DCLK` and `DATA[7..0]` are not left floating at the end of configuration, the MAX II device must drive them either high or low, whichever is convenient on your board. The `DATA[7..0]` pins are available as user I/O pins after configuration. When you select the FPP scheme in the Quartus II software, as a default, these I/O pins are tri-stated in user mode and should be driven by the MAX II device. To change this default option in the Quartus II software, select the **Dual-Purpose Pins** tab of the **Device & Pin Options** dialog box.

The configuration clock (`DCLK`) speed must be below the specified frequency to ensure correct configuration. No maximum `DCLK` period exists, which means you can pause configuration by halting `DCLK` for an indefinite amount of time.

☞ If you are using the Stratix II decompression and/or design security feature and need to stop `DCLK`, it can only be stopped three clock cycles after the last data byte was latched into the Stratix II device.

By stopping `DCLK`, the configuration circuit allows enough clock cycles to process the last byte of latched configuration data. When the clock restarts, the MAX II device must provide data on the `DATA[7..0]` pins prior to sending the first `DCLK` rising edge.

If an error occurs during configuration, the device drives its `nSTATUS` pin low, resetting itself internally. The low signal on the `nSTATUS` pin also alerts the MAX II device that there is an error. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device & Pin Options** (dialog box) is turned on, the device releases `nSTATUS` after a reset time-out period (maximum of 40 µs). After `nSTATUS` is released and pulled high by a pull-up resistor, the MAX II device can try to reconfigure the target device without needing to pulse nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 40 µs) on nCONFIG to restart the configuration process.

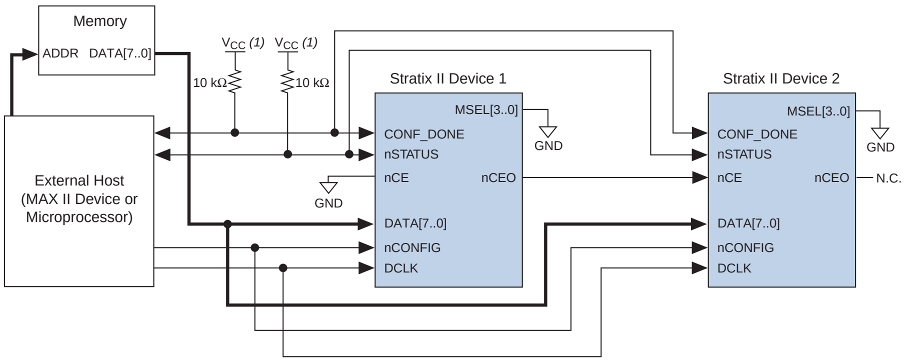
The MAX II device can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The CONF_DONE pin must be monitored by the MAX II device to detect errors and determine when programming completes. If all configuration data is sent, but the CONF_DONE or INIT_DONE signals have not gone high, the MAX II device will reconfigure the target device.

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).

When the device is in user-mode, initiating a reconfiguration is done by transitioning the nCONFIG pin low-to-high. The nCONFIG pin should be low for at least 40 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 7–4 shows how to configure multiple devices using a MAX II device. This circuit is similar to the FPP configuration circuit for a single device, except the Stratix II devices are cascaded for multi-device configuration.

Figure 7–4. Multi-Device FPP Configuration Using an External Host



Note to Figure 7–4:

(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O standard on the device and the external host.

In multi-device FPP configuration, 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. The second device in the chain begins configuration within one clock cycle; therefore, the transfer of data destinations is transparent to the MAX II device. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA[7..0], and CONF_DONE) are connected to every device in the chain. The configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

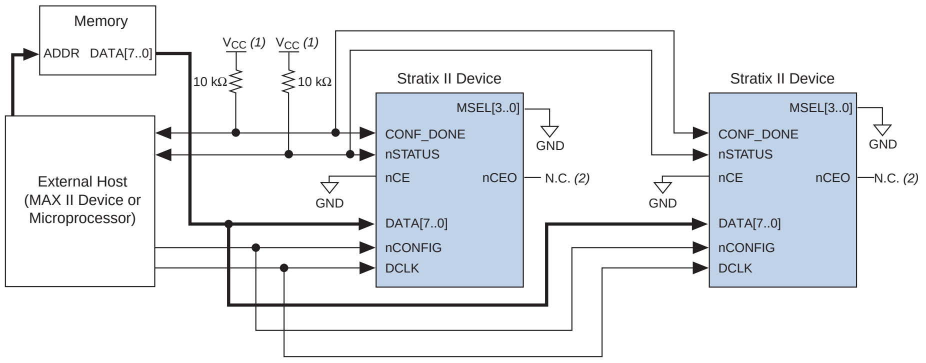
All nSTATUS and CONF_DONE pins are tied together and if any device detects an error, configuration stops for the entire chain and the entire chain must be reconfigured. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This behavior is similar to a single device detecting an error.

If the **Auto-restart configuration after error** option is turned on, the devices release their nSTATUS pins after a reset time-out period (maximum of 40 µs). After all nSTATUS pins are released and pulled high, the MAX II device can try to reconfigure the chain without pulsing nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 40 µs) on nCONFIG to restart the configuration process.

In a multi-device FPP configuration chain, all Stratix II devices in the chain must either enable or disable the decompression and/or design security feature. You can not selectively enable the decompression and/or design security feature for each device in the chain because of the DATA and DCLK relationship. If the chain contains devices that do not support design security, you should use a serial configuration scheme.

If a system has multiple devices that contain the same configuration data, tie all device nCE inputs to GND, and leave nCEO pins floating. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA[7..0], and CONF_DONE) are connected to every device in the chain. Configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Devices must be the same density and package. All devices will start and complete configuration at the same time. Figure 7–5 shows multi-device FPP configuration when both Stratix II devices are receiving the same configuration data.

Figure 7–5. Multiple-Device FPP Configuration Using an External Host When Both Devices Receive the Same Data



Notes to Figure 7–5:

(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

(2) The nCEO pins of both Stratix II devices are left unconnected when configuring the same configuration data into multiple devices.

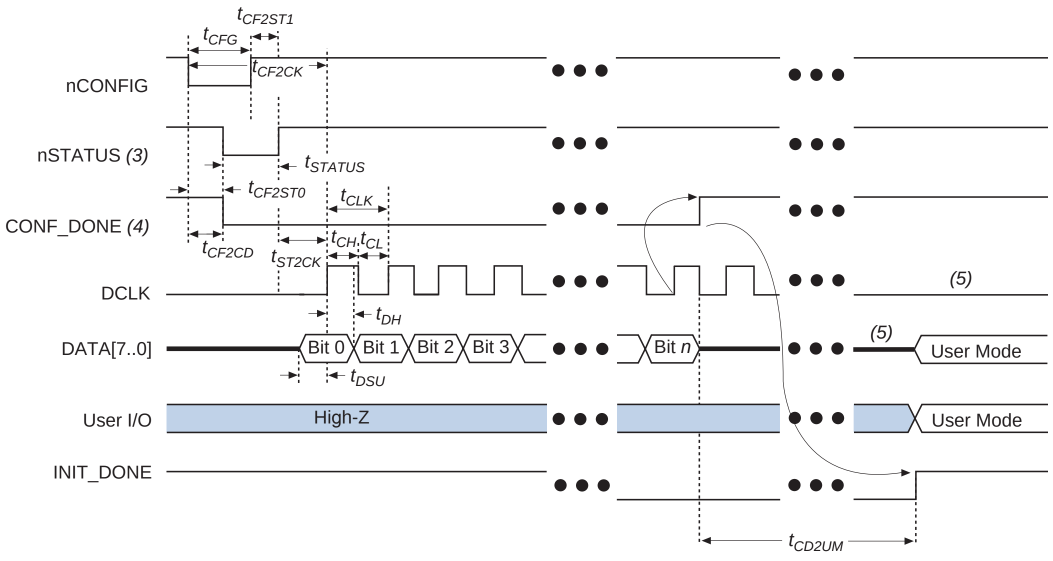
You can use a single configuration chain to configure Stratix II devices with other Altera devices that support FPP configuration, such as Stratix devices. To ensure that all devices in the chain complete configuration at the same time or that an error flagged by one device initiates reconfiguration in all devices, tie all of the device CONF_DONE and nSTATUS pins together.

For more information on configuring multiple Altera devices in the same configuration chain, see *Configuring Mixed Altera device Chains* in the *Configuration Handbook.*

FPP Configuration Timing

Figure 7–6 shows the timing waveform for FPP configuration when using a MAX II device as an external host. This waveform shows the timing when the decompression and the design security feature are not enabled.

Figure 7–6. FPP Configuration Timing Waveform Notes (1), (2)



Notes to Figure 7–6:

(1) This timing waveform should be used when the decompression and design security feature are not used.
(2) 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.
(3) Upon power-up, the Stratix II device holds nSTATUS low for the time of the POR delay.
(4) Upon power-up, before and during configuration, CONF_DONE is low.
(5) DCLK should not be left floating after configuration. It should be driven high or low, whichever is more convenient. DATA[7..0] are available as user I/O pins after configuration and the state of these pins depends on the dual-purpose pin settings.

Table 7–6 defines the timing parameters for Stratix II devices for FPP configuration when the decompression and the design security feature are not enabled.

Table 7–6. FPP Timing Parameters for Stratix II Devices (Part 1 of 2) Notes (1), (2)

Symbol	Parameter	Min	Max	Units
t_{POR}	POR delay	12	100	ms
t_{CF2CD}	nCONFIG low to CONF_DONE low		800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		800	ns
t_{CFG}	nCONFIG low pulse width	2		µs
t_{STATUS}	nSTATUS low pulse width	10	100 (3)	µs

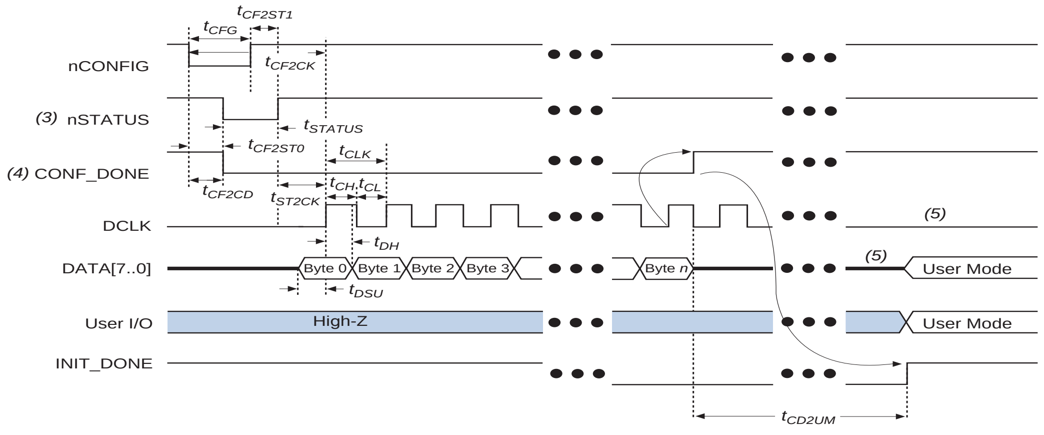
Table 7–6. FPP Timing Parameters for Stratix II Devices (Part 2 of 2) *Notes (1), (2)*

Symbol	Parameter	Min	Max	Units
t_{CF2ST1}	nCONFIG high to nSTATUS high		100 *(3)*	µs
t_{CF2CK}	nCONFIG high to first rising edge on DCLK	100		µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	2		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 frequency		100	MHz
t_R	Input rise time		40	ns
t_F	Input fall time		40	ns
t_{CD2UM}	CONF_DONE high to user mode *(4)*	20	40	µs
t_{CD2CU}	CONF_DONE high to CLKUSR enabled	4 × maximum DCLK period		
t_{CD2UMC}	CONF_DONE high to user mode with CLKUSR option on	t_{CD2CU} + (299 × CLKUSR period)		

Notes to Table 7–6:
(1) This information is preliminary.
(2) These timing parameters should be used when the decompression and design security feature are not used.
(3) This value is obtainable if users do not delay configuration by extending the nCONFIG or nSTATUS low pulse width.
(4) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device.

Figure 7–7 shows the timing waveform for FPP configuration when using a MAX II device as an external host. This waveform shows the timing when the decompression and/or the design security feature are enabled.

Figure 7–7. FPP Configuration Timing Waveform With Decompression or Design Security Feature Enabled Notes (1), (2)



Notes to Figure 7–7:

(1) This timing waveform should be used when the decompression and/or design security feature are used.
(2) 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.
(3) Upon power-up, the Stratix II device holds `nSTATUS` low for the time of the POR delay.
(4) Upon power-up, before and during configuration, `CONF_DONE` is low.
(5) `DCLK` should not be left floating after configuration. It should be driven high or low, whichever is more convenient. `DATA[7..0]` are available as user I/O pins after configuration and the state of these pins depends on the dual-purpose pin settings.
(6) If needed, `DCLK` can be paused by holding it low. When `DCLK` restarts, the external host must provide data on the `DATA[7..0]` pins prior to sending the first `DCLK` rising edge.

Table 7–7 defines the timing parameters for Stratix II devices for FPP configuration when the decompression and/or the design security feature are enabled.

Table 7–7. FPP Timing Parameters for Stratix II Devices With Decompression or Design Security Feature Enabled Notes (1), (2)

Symbol	Parameter	Min	Max	Units
t_{POR}	POR delay	12	100	ms
t_{CF2CD}	nCONFIG low to CONF_DONE low		800	ns
t_{CF2ST0}	nCONFIG low to nSTATUS low		800	ns
t_{CFG}	nCONFIG low pulse width	2		µs
t_{STATUS}	nSTATUS low pulse width	10	100 (3)	µs
t_{CF2ST1}	nCONFIG high to nSTATUS high		100 (3)	µs
t_{CF2CK}	nCONFIG high to first rising edge on DCLK	100		µs
t_{ST2CK}	nSTATUS high to first rising edge of DCLK	2		µs
t_{DSU}	Data setup time before rising edge on DCLK	7		ns
t_{DH}	Data hold time after rising edge on DCLK	30		ns
t_{CH}	DCLK high time	4		ns
t_{CL}	DCLK low time	4		ns
t_{CLK}	DCLK period	10		ns
f_{MAX}	DCLK frequency		100	MHz
t_{DATA}	Data rate		200	Mbps
t_R	Input rise time		40	ns
t_F	Input fall time		40	ns
t_{CD2UM}	CONF_DONE high to user mode (4)	20	40	µs
t_{CD2CU}	CONF_DONE high to CLKUSR enabled	4 × maximum DCLK period		
t_{CD2UMC}	CONF_DONE high to user mode with CLKUSR option on	t_{CD2CU} + (299 × CLKUSR period)		

Notes to Table 7–7:

(1) This information is preliminary.
(2) These timing parameters should be used when the decompression and design security feature are used.
(3) This value is obtainable if users do not delay configuration by extending the nCONFIG or nSTATUS low pulse width.
(4) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device.

Device configuration options and how to create configuration files are discussed further in the *Software Settings* chapter in the *Configuration Handbook.*

FPP Configuration Using a Microprocessor

In the FPP configuration scheme, a microprocessor can control the transfer of configuration data from a storage device, such as flash memory, to the target Stratix II device.

All information in "FPP Configuration Using a MAX II Device as an External Host" on page 7–11 is also applicable when using a microprocessor as an external host. Refer to that section for all configuration and timing information.

FPP Configuration Using an Enhanced Configuration Device

In the FPP configuration scheme, an enhanced configuration device sends a byte of configuration data every DCLK cycle to the Stratix II device. Configuration data is stored in the configuration device.

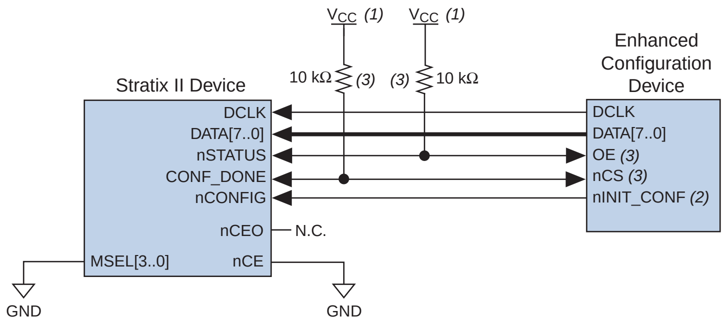
When configuring your Stratix II device using FPP mode and an enhanced configuration device, the enhanced configuration device decompression feature is available while the Stratix II decompression feature and design security feature are not.

Figure 7–8 shows the configuration interface connections between the Stratix II device and the enhanced configuration device for single device configuration.

The figures in this chapter only show the configuration-related pins and the configuration pin connections between the configuration device and the device.

For more information on the enhanced configuration device and flash interface pins, such as PGM[2..0], EXCLK, PORSEL, A[20..0], and DQ[15..0], refer to the *Enhanced Configuration Devices (EPC4, EPC8, & EPC16) Data Sheet.*

Figure 7–8. Single Device FPP Configuration Using an Enhanced Configuration Device



Notes to Figure 7–8:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The `nINIT_CONF` pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active. This means an external pull-up resistor should not be used on the `nINIT_CONF`-`nCONFIG` line. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' `OE` and `nCS` pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

The value of the internal pull-up resistors on the enhanced configuration devices can be found in the *Enhanced Configuration Devices (EPC4, EPC8, & EPC16) Data Sheet*.

When using enhanced configuration devices, you can connect the device's `nCONFIG` pin to `nINIT_CONF` pin of the enhanced configuration device, which allows the `INIT_CONF` JTAG instruction to initiate device configuration. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor. An internal pull-up resistor on the `nINIT_CONF` pin is always active in the enhanced configuration devices, which means an external pull-up resistor should not be used if `nCONFIG` is tied to `nINIT_CONF`.

Upon power-up, the Stratix II device goes through a POR. The POR delay is dependent on the `PORSEL` pin setting; when `PORSEL` is driven low, the POR time is approximately 100 ms, if `PORSEL` is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold `nSTATUS` low, and tri-state all user I/O pins. The configuration device also goes through a POR delay to allow the power supply to stabilize. The POR time for enhanced configuration devices can be set to either 100 ms or 2 ms, depending on its `PORSEL` pin setting. If the `PORSEL` pin is connected to GND, the POR delay is 100 ms. If the `PORSEL` pin is connected to V_{CC}, the POR delay is 2 ms. 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 selecting a POR time, you need to ensure that the device completes power-up before the enhanced configuration device exits POR. Altera recommends that you use a 12-ms POR time for the Stratix II device, and use a 100-ms POR time for the enhanced configuration device.

When both devices complete POR, they release their open-drain `OE` or `nSTATUS` pin, which is then pulled high by a pull-up resistor. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If `nIO_pullup` is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If `nIO_pullup` is driven high, the weak pull-up resistors are disabled.

The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *Stratix II Device Handbook*.

When the power supplies have reached the appropriate operating voltages, the target device senses the low-to-high transition on `nCONFIG` and initiates the configuration cycle. The configuration cycle consists of three stages: reset, configuration and initialization. While `nCONFIG` or `nSTATUS` are low, the device is in reset. The beginning of configuration can be delayed by holding the `nCONFIG` or `nSTATUS` pin low.

V_{CCINT}, V_{CCIO} and V_{CCPD} of the banks where the configuration and JTAG pins reside need to be fully powered to the appropriate voltage levels in order to begin the configuration process.

When `nCONFIG` goes high, the device comes out of reset and releases the `nSTATUS` pin, which is pulled high by a pull-up resistor. Enhanced configuration devices have an optional internal pull-up resistor on the `OE` pin. This option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If this internal pull-up resistor is not used, an external 10-kΩ pull-up resistor on the `OE-nSTATUS` line is required. Once `nSTATUS` is released, the device is ready to receive configuration data and the configuration stage begins.

When nSTATUS is pulled high, the configuration device's OE pin also goes high and the configuration device clocks data out to the device using its internal oscillator. The Stratix II device receives configuration data on its DATA[7..0] pins and the clock is received on the DCLK pin. A byte of data is latched into the device on each rising edge of DCLK.

After the device has received all configuration data successfully, it releases the open-drain CONF_DONE pin which is pulled high by a pull-up resistor. Since CONF_DONE is tied to the configuration device's nCS pin, the configuration device is disabled when CONF_DONE goes high. Enhanced configuration devices have an optional internal pull-up resistor on the nCS pin. This option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If this internal pull-up resistor is not used, an external 10-kΩ pull-up resistor on the nCS-CONF_DONE line is required. A low to high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin.

In Stratix II devices, the initialization clock source is either the Stratix II internal oscillator (typically 10 MHz) or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If the internal oscillator is used, the Stratix II device will provide itself with enough clock cycles for proper initialization. You also have the flexibility to synchronize initialization of multiple devices or to delay initialization by using the CLKUSR option. The **Enable user-supplied start-up clock** (**CLKUSR**) option can be turned on in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. Supplying a clock on CLKUSR will not affect the configuration process. After all configuration data has been accepted and CONF_DONE goes high, CLKUSR will be enabled after the time specified as t_{CD2CU}. After this time period elapses, the Stratix II devices require 299 clock cycles to initialize properly and enter user mode. Stratix II devices support a CLKUSR f_{MAX} of 100 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. The **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If the INIT_DONE pin is used, it will be high due to an external 10-kΩ pull-up resistor 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. When initialization is complete, the INIT_DONE pin will be released and pulled high. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design. The enhanced configuration device will drive DCLK low and DATA[7..0] high at the end of configuration.

If an error occurs during configuration, the device drives its nSTATUS pin low, resetting itself internally. Since the nSTATUS pin is tied to OE, the configuration device will also be reset. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box) is turned on, the device will automatically initiate reconfiguration if an error occurs. The Stratix II device will release its nSTATUS pin after a reset time-out period (maximum of 40 µs). When the nSTATUS pin is released and pulled high by a pull-up resistor, the configuration device reconfigures the chain. If this option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low for at least 40 µs to restart configuration. The external system can pulse nCONFIG if nCONFIG is under system control rather than tied to V_{CC}.

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 device has not configured successfully. Enhanced configuration devices wait for 64 DCLK cycles after the last configuration bit was sent for CONF_DONE to reach a high state. In this case, the configuration device pulls its OE pin low, which in turn drives the target device's nSTATUS pin low. If the Auto-restart configuration after error option is set in the software, the target device resets and then releases its nSTATUS pin after a reset time-out period (maximum of 40 µs). When nSTATUS returns to a logic high level, the configuration device will try to reconfigure the device.

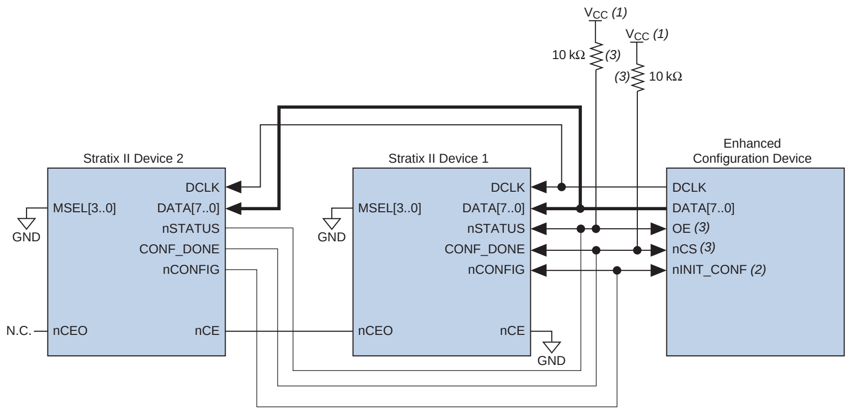
When CONF_DONE is sensed low after configuration, the configuration device recognizes that the target device has not configured successfully. Therefore, your system should not pull CONF_DONE low to delay initialization. Instead, you should use the CLKUSR option to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain will initialize together if their CONF_DONE pins are tied together.

If the optional CLKUSR pin is used and nCONFIG is pulled low to restart configuration during device initialization, ensure CLKUSR continues toggling during the time nSTATUS is low (maximum of 40 µs).

When the device is in user-mode, a reconfiguration can be initiated by pulling the nCONFIG pin low. The nCONFIG pin should be low for at least 40 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Since CONF_DONE is pulled low, this will activate the configuration device as it will see its nCS pin drive low. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 7–9 shows how to configure multiple Stratix II devices with an enhanced configuration device. This circuit is similar to the configuration device circuit for a single device, except the Stratix II devices are cascaded for multi-device configuration.

Figure 7–9. Multi-Device FPP Configuration Using an Enhanced Configuration Device



Notes to Figure 7–9:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The `nINIT_CONF` pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active. This means an external pull-up resistor should not be used on the `nINIT_CONF`-`nCONFIG` line. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' `OE` and `nCS` pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-up resistors on configuration device** option when generating programming files.

Enhanced configuration devices cannot be cascaded.

When performing multi-device configuration, you must generate the configuration device's POF from each project's SOF. You can combine multiple SOFs using the **Convert Programming Files** window in the Quartus II software.

For more information on how to create configuration files for multi-device configuration chains, see *Software Settings* in Volume 2 of the *Configuration Handbook*.

In multi-device FPP configuration, 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. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA[7..0], and CONF_DONE) are connected to every device in the chain. Pay special attention to the configuration signals because they may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device.

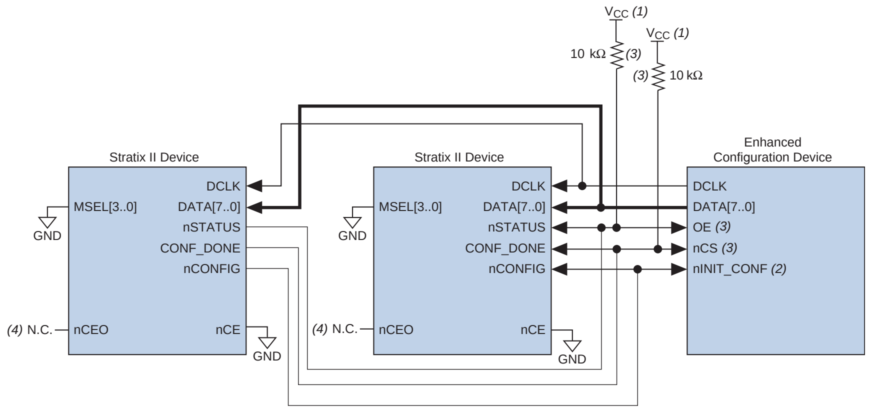
When configuring multiple devices, configuration does not begin until all devices release their OE or nSTATUS pins. Similarly, since all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

Since all nSTATUS and CONF_DONE pins are tied together, if any device detects an error, configuration stops for the entire chain and the entire chain must be reconfigured. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This low signal drives the OE pin low on the enhanced configuration device and drives nSTATUS low on all devices, which causes them to enter a reset state. This behavior is similar to a single device detecting an error.

If the **Auto-restart configuration after error** option is turned on, the devices will automatically initiate reconfiguration if an error occurs. The devices will release their nSTATUS pins after a reset time-out period (maximum of 40 µs). When all the nSTATUS pins are released and pulled high, the configuration device tries to reconfigure the chain. If the **Auto-restart configuration after error** option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low for at least 40 µs to restart configuration. The external system can pulse nCONFIG if nCONFIG is under system control rather than tied to V_{CC}.

Your system may have multiple devices that contain the same configuration data. To support this configuration scheme, all device nCE inputs are tied to GND, while nCEO pins are left floating. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA[7..0], and CONF_DONE) are connected to every device in the chain. Configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Devices must be the same density and package. All devices will start and complete configuration at the same time. Figure 7–10 shows multi-device FPP configuration when both Stratix II devices are receiving the same configuration data.

Figure 7–10. Multiple-Device FPP Configuration Using an Enhanced Configuration Device When Both devices Receive the Same Data



Notes to Figure 7–10:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The `nINIT_CONF` pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active. This means an external pull-up resistor should not be used on the `nINIT_CONF`-`nCONFIG` line. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' `OE` and `nCS` pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

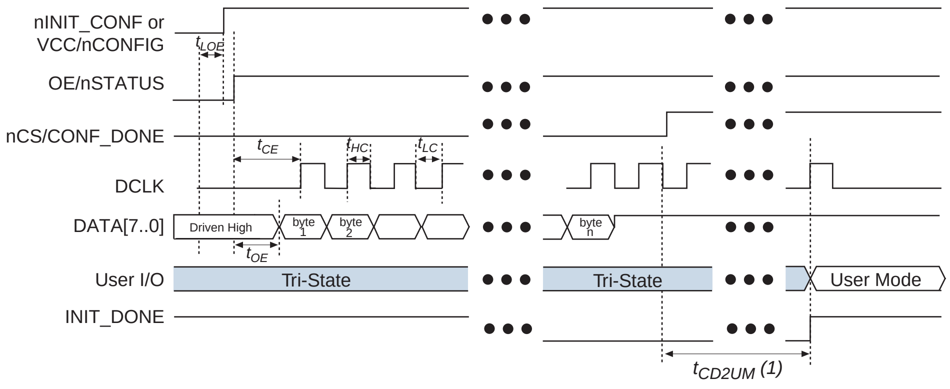
(4) The `nCEO` pins of both devices are left unconnected when configuring the same configuration data into multiple devices.

You can use a single enhanced configuration chain to configure multiple Stratix II devices with other Altera devices that support FPP configuration, such as Stratix and Stratix GX devices. To ensure that all devices in the chain complete configuration at the same time or that an error flagged by one device initiates reconfiguration in all devices, all of the device `CONF_DONE` and `nSTATUS` pins must be tied together.

For more information on configuring multiple Altera devices in the same configuration chain, see *Configuring Mixed Altera device Chains* in the *Configuration Handbook*.

Figure 7–11 shows the timing waveform for the FPP configuration scheme using an enhanced configuration device.

Figure 7–11. Stratix II FPP Configuration Using an Enhanced Configuration Device Timing Waveform



Note to Figure 7–11:

(1) The initialization clock can come from the Stratix II internal oscillator or the CLKUSR pin.

For timing information, refer to the *Enhanced Configuration Devices (EPC4, EPC8, and EPC16) Data Sheet* in the *Configuration Handbook*.

Device configuration options and how to create configuration files are discussed further in the *Software Settings* section in of the *Configuration Handbook*.

Active Serial Configuration (Serial Configuration Devices)

In the AS configuration scheme, Stratix II devices are configured using a serial configuration device. These configuration devices are low cost devices with non-volatile memory that feature a simple four-pin interface and a small form factor. These features make serial configuration devices an ideal low-cost configuration solution.

For more information on serial configuration devices, see the *Serial Configuration Devices Data Sheet* in the *Configuration Handbook*.

Serial configuration devices provide a serial interface to access configuration data. During device configuration, Stratix II devices read configuration data via the serial interface, decompress data if necessary, and configure their SRAM cells. This scheme is referred to as the AS configuration scheme because the device controls the configuration interface. This scheme contrast the PS configuration scheme where the configuration device controls the interface.

☞ The Stratix II decompression and design security feature are fully available when configuring your Stratix II device using AS mode.

Table 7–8 shows the MSEL pin settings when using the AS configuration scheme.

Table 7–8. Stratix II MSEL Pin Settings for AS Configuration Schemes

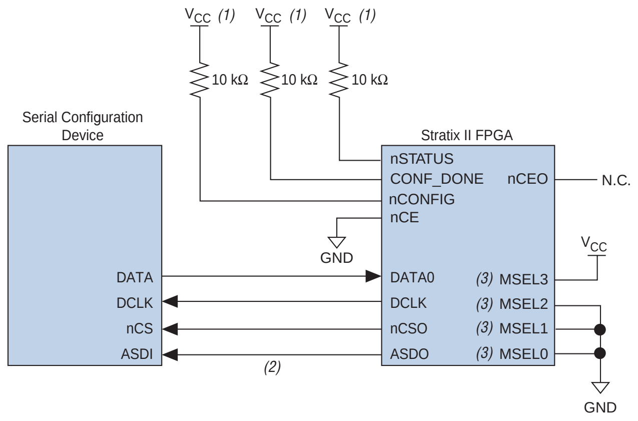
Configuration Scheme	MSEL3	MSEL2	MSEL1	MSEL0
Fast AS (40 MHz) *(1)*	1	0	0	0
Remote system upgrade fast AS (40 MHz) *(1)*	1	0	0	1
AS (20 MHz) *(1)*	1	1	0	1
Remote system upgrade AS (20 MHz) *(1)*	1	1	1	0

Note to Table 7–8:

(1) Only the EPCS16 and EPCS64 devices support a DCLK up to 40 MHz clock; other EPCS devices support a DCLK up to 20 MHz. See the *Serial Configuration Devices Data Sheet* for more information.

Serial configuration devices have a four-pin interface: serial clock input (DCLK), serial data output (DATA), AS data input (ASDI), and an active-low chip select (nCS). This four-pin interface connects to Stratix II device pins, as shown in Figure 7–12.

Figure 7–12. Single Device AS Configuration



Notes to Figure 7–12:
(1) Connect the pull-up resistors to a 3.3-V supply.
(2) Stratix II devices use the `ASDO` to `ASDI` path to control the configuration device.
(3) If using an EPCS4 device, `MSEL[3..0]` should be set to 1101. See Table 7–8 for more details.

Upon power-up, the Stratix II device goes through a POR. The POR delay is dependent on the `PORSEL` pin setting. When `PORSEL` is driven low, the POR time is approximately 100 ms. If `PORSEL` is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold `nSTATUS` and `CONF_DONE` low, and tri-state all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If `nIO_pullup` is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If `nIO_pullup` is driven high, the weak pull-up resistors are disabled.

The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *Operating Conditions table* in the *Stratix II Device Handbook*.

The configuration cycle consists of three stages: reset, configuration and initialization. While `nCONFIG` or `nSTATUS` are low, the device is in reset. After POR, the Stratix II device releases `nSTATUS`, which is pulled high by an external 10-kΩ pull-up resistor, and enters configuration mode.

To begin configuration, power the V_{CCINT}, V_{CCIO}, and V_{CCPD} voltages (for the banks where the configuration and JTAG pins reside) to the appropriate voltage levels.

The serial clock (DCLK) generated by the Stratix II device controls the entire configuration cycle and provides the timing for the serial interface. Stratix II devices use an internal oscillator to generate DCLK. Using the MSEL[] pins, you can select to use either a 40- or 20-MHz oscillator.

☞ Only the EPCS16 and EPCS64 devices support a DCLK up to 40-MHz clock; other EPCS devices support a DCLK up to 20-MHz. See the *Serial Configuration Devices Data Sheet* for more information.

Table 7–9 shows the active serial DCLK output frequencies.

Table 7–9. Active Serial DCLK Output Frequency *Note (1)*

Oscillator	Minimum	Typical	Maximum	Units
40 MHz *(2)*	20	26	40	MHz
20 MHz	10	13	20	MHz

Notes to Table 7–9:

(1) These values are preliminary.

(2) Only the EPCS16 and EPCS64 devices support a DCLK up to 40-MHz clock; other EPCS devices support a DCLK up to 20-MHz. See the *Serial Configuration Devices Data Sheet* for more information.

The serial configuration device latches input/control signals on the rising edge of DCLK and drives out configuration data on the falling edge. Stratix II devices drive out control signals on the falling edge of DCLK and latch configuration data on the rising edge of DCLK.

In configuration mode, the Stratix II device enables the serial configuration device by driving its nCSO output pin low, which connects to the chip select (nCS) pin of the configuration device. The Stratix II device uses the serial clock (DCLK) and serial data output (ASDO) pins to send operation commands and/or read address signals to the serial configuration device. The configuration device provides data on its serial data output (DATA) pin, which connects to the DATA0 input of the Stratix II device.

After all configuration bits are received by the Stratix II device, it releases the open-drain CONF_DONE pin, which is pulled high by an external 10-kΩ resistor. Initialization begins only after the CONF_DONE signal reaches a logic high level. All AS configuration pins, DATA0, DCLK, nCSO, and ASDO, have weak internal pull-up resistors, which are always active. Therefore, after configuration these pins will be driven high.

In Stratix II devices, the initialization clock source is either the Stratix II 10-MHz (typical) internal oscillator (separate from the active serial internal oscillator) or the optional `CLKUSR` pin. By default, the internal oscillator is the clock source for initialization. If the internal oscillator is used, the Stratix II device will provide itself with enough clock cycles for proper initialization. You also have the flexibility to synchronize initialization of multiple devices or to delay initialization by using the `CLKUSR` option. The **Enable user-supplied start-up clock** (**`CLKUSR`**) option can be turned on in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. When you **Enable** the **user supplied start-up clock** option, the `CLKUSR` pin is the initialization clock source. Supplying a clock on `CLKUSR` will not affect the configuration process. After all configuration data has been accepted and `CONF_DONE` goes high, `CLKUSR` will be enabled after 600 ns. After this time period elapses, the Stratix II devices require 299 clock cycles to initialize properly and enter user mode. Stratix II devices support a `CLKUSR` f_{MAX} of 100 MHz.

An optional `INIT_DONE` pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. The **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If the `INIT_DONE` pin is used, it will be high due to an external 10-kΩ pull-up resistor 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. When initialization is complete, the `INIT_DONE` pin will be released and pulled high. This low-to-high transition signals that the device has entered user mode. When initialization is complete, the device enters user mode. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design.

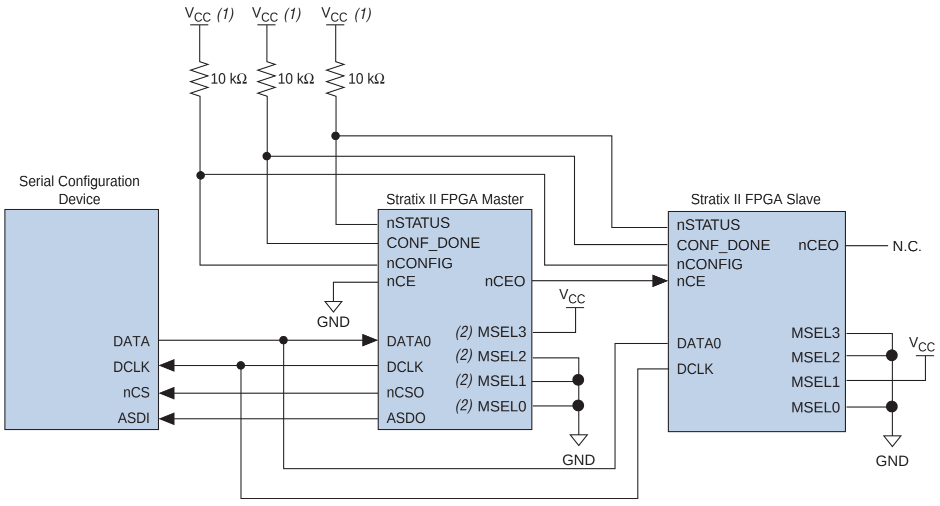
If an error occurs during configuration, the Stratix II device asserts the `nSTATUS` signal low indicating a data frame error, and the `CONF_DONE` signal will stay low. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box) is turned on, the Stratix II device resets the configuration device by pulsing `nCSO`, releases `nSTATUS` after a reset time-out period (about 40 µs), and retries configuration. If this option is turned off, the system must monitor `nSTATUS` for errors and then pulse `nCONFIG` low for at least 40 µs to restart configuration.

When the Stratix II device is in user mode, you can initiate reconfiguration by pulling the `nCONFIG` pin low. The `nCONFIG` pin should be low for at least 40 µs. When `nCONFIG` is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the Stratix II device, reconfiguration begins.

You can configure multiple Stratix II devices using a single serial configuration device. You can cascade multiple Stratix II devices using the chip-enable (nCE) and chip-enable-out (nCEO) pins. The first device in the chain must have its nCE pin connected to ground. You must connect its nCEO pin to the nCE pin of the next device in the chain. When the first device captures all of its configuration data from the bit stream, it drives the nCEO pin low, enabling the next device in the chain. You must leave the nCEO pin of the last device unconnected. The nCONFIG, nSTATUS, CONF_DONE, DCLK, and DATA0 pins of each device in the chain are connected (see Figure 7–13).

This first Stratix II device in the chain is the configuration master and controls configuration of the entire chain. You must connect its MSEL pins to select the AS configuration scheme. The remaining Stratix II devices are configuration slaves and you must connect their MSEL pins to select the PS configuration scheme. Any other Altera device that supports PS configuration can also be part of the chain as a configuration slave. Figure 7–13 shows the pin connections for this setup.

Figure 7–13. Multi-Device AS Configuration



Notes to Figure 7–13:
(1) Connect the pull-up resistors to a 3.3-V supply.
(2) If using an EPCS4 device, MSEL[3..0] should be set to 1101. See Table 7–8 for more details.

As shown in Figure 7–13, the nSTATUS and CONF_DONE pins on all target devices are connected together with external pull-up resistors. These pins are open-drain bidirectional pins on the devices. When the first device asserts nCEO (after receiving all of its configuration data), it releases its CONF_DONE pin. But the subsequent devices in the chain keep this shared CONF_DONE line low until they have received their configuration data. When all target devices in the chain have received their configuration data and have released CONF_DONE, the pull-up resistor drives a high level on this line and all devices simultaneously enter initialization mode.

If an error occurs at any point during configuration, the nSTATUS line is driven low by the failing device. If you enable the Auto-restart configuration after error option, reconfiguration of the entire chain begins after a reset time-out period (a maximum of 40 µs). If the **Auto-restart configuration after error** 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}.

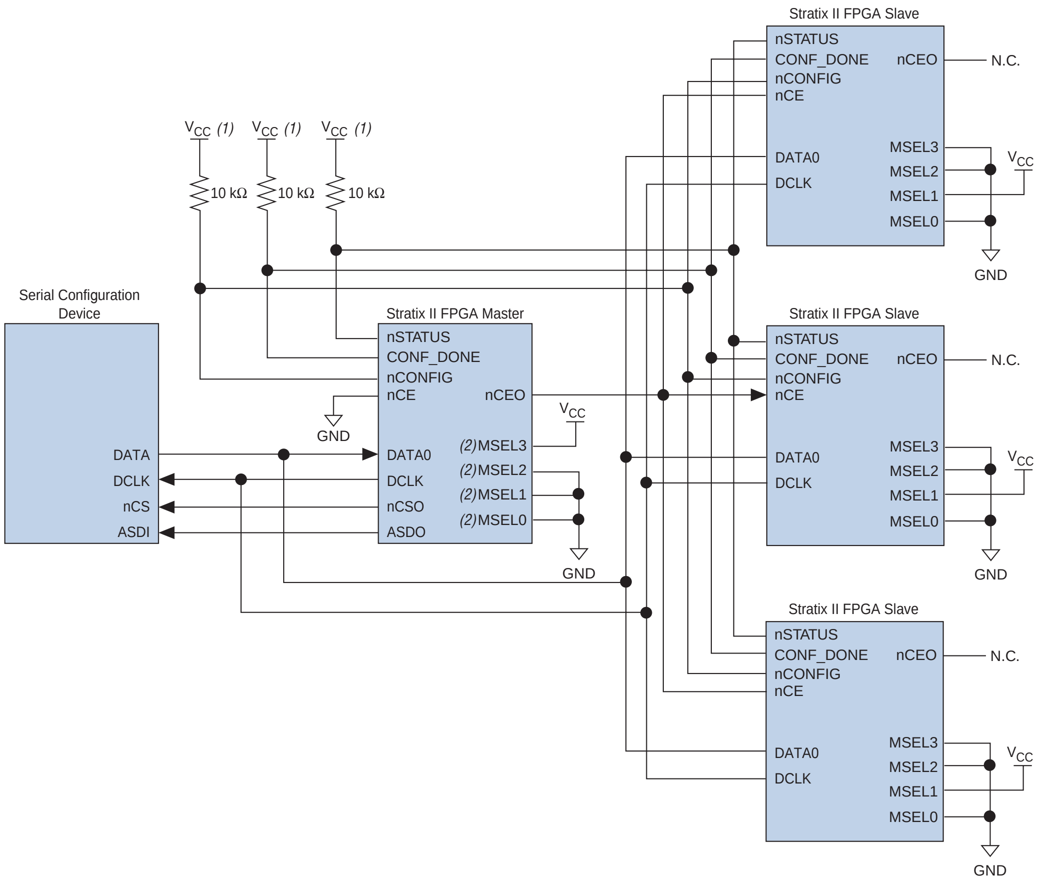
☞ While you can cascade Stratix II devices, serial configuration devices cannot be cascaded or chained together.

If the configuration bit stream size exceeds the capacity of a serial configuration device, you must select a larger configuration device and/or enable the compression feature. When configuring multiple devices, the size of the bitstream is the sum of the individual devices' configuration bitstreams.

A system may have multiple devices that contain the same configuration data. In active serial chains, this can be implemented by storing two copies of the SOF in the serial configuration device. The first copy would configure the master Stratix II device, and the second copy would configure all remaining slave devices concurrently. All slave devices must be the same density and package. The setup is similar to Figure 7–13, where the master is setup in active serial mode and the slave devices are setup in passive serial mode.

To configure four identical Stratix II devices with the same SOF, you could setup the chain similar to the example shown in Figure 7–14. The first device is the master device and its MSEL pins should be set to select AS configuration. The other three slave devices are set up for concurrent configuration and its MSEL pins should be set to select PS configuration. The nCEO pin from the master device drives the nCE input pins on all three slave devices, and the DATA and DCLK pins connect in parallel to all four devices. During the first configuration cycle, the master device reads its configuration data from the serial configuration device while holding nCEO high. After completing its configuration cycle, the master drives nCE low and transmits the second copy of the configuration data to all three slave devices, configuring them simultaneously.

Figure 7–14. Multi-Device AS Configuration When devices Receive the Same Data



Notes to Figure 7–14:

(1) Connect the pull-up resistors to a 3.3-V supply.

(2) If using an EPCS4 device, `MSEL[3..0]` should be set to 1101. See Table 7–8 for more details.

Estimating Active Serial Configuration Time

Active serial configuration time is dominated by the time it takes to transfer data from the serial configuration device to the Stratix II device. This serial interface is clocked by the Stratix II `DCLK` output (generated from an internal oscillator). As listed in Table 7–9, the `DCLK` minimum frequency when choosing to use the 40-MHz oscillator is 20 MHz (50 ns). Therefore, the maximum configuration time estimate for an EP2S15 device (5 MBits of uncompressed data) is:

RBF Size (minimum `DCLK` period / 1 bit per `DCLK` cycle) = estimated maximum configuration time

5 Mbits × (50 ns / 1 bit) = 250 ms

To estimate the typical configuration time, use the typical `DCLK` period as listed in Table 7–9. With a typical `DCLK` period of 38.46 ns, the typical configuration time is 192 ms. Enabling compression reduces the amount of configuration data that is transmitted to the Stratix II device, which also reduces configuration time. On average, compression reduces configuration time by 50%.

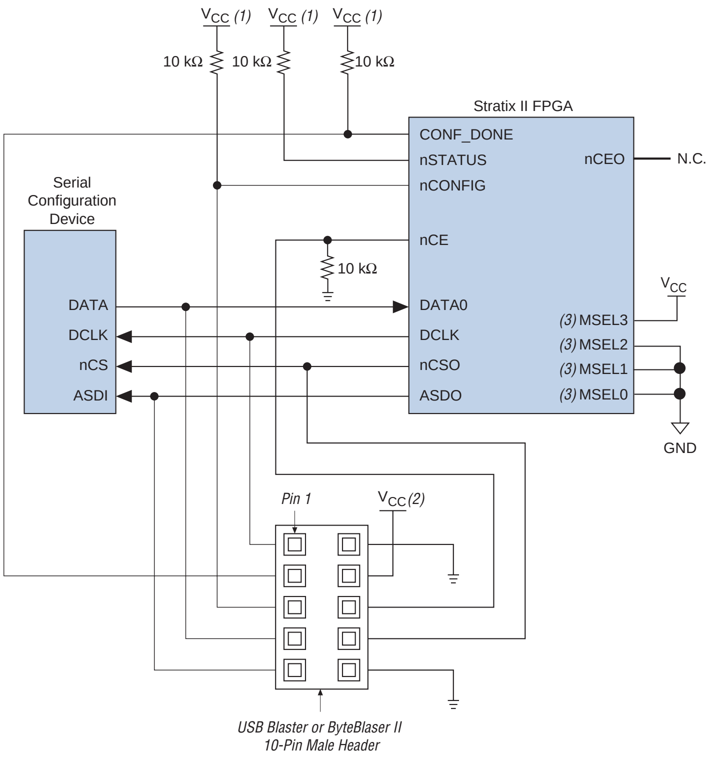
Programming Serial Configuration Devices

Serial configuration devices are non-volatile, flash-memory-based devices. You can program these devices in-system using the USB-Blaster™ or ByteBlaster™ II download cable. Alternatively, you can program them using the Altera Programming Unit (APU), supported third-party programmers, or a microprocessor with the SRunner software driver.

You can perform in-system programming of serial configuration devices via the AS programming interface. During in-system programming, the download cable disables device access to the AS interface by driving the `nCE` pin high. Stratix II devices are also held in reset by a low level on `nCONFIG`. After programming is complete, the download cable releases `nCE` and `nCONFIG`, allowing the pull-down and pull-up resistors to drive GND and V_{CC}, respectively. Figure 7–15 shows the download cable connections to the serial configuration device.

For more information on the USB Blaster download cable, see the *USB-Blaster USB Port Download Cable Data Sheet*. For more information on the ByteBlaster II cable, see the *ByteBlaster II Download Cable Data Sheet*.

Figure 7–15. In-System Programming of Serial Configuration Devices



Notes to Figure 7–15:

(1) Connect these pull-up resistors to 3.3-V supply.

(2) Power up the ByteBlaster II cable's V_{CC} with a 3.3-V supply.

(3) If using an EPCS4 device, `MSEL[3..0]` should be set to `1101`. See Table 7–8 for more details.

You can program serial configuration devices by using the Quartus II software with the Altera programming hardware (APU) and the appropriate configuration device programming adapter. The EPCS1 and EPCS4 devices are offered in an eight-pin small outline integrated circuit (SOIC) package.

In production environments, serial configuration devices can be programmed using multiple methods. Altera programming hardware or other third-party programming hardware can be used to program blank serial configuration devices before they are mounted onto printed circuit boards (PCBs). Alternatively, you can use an on-board microprocessor to program the serial configuration device in-system using C-based software drivers provided by Altera.

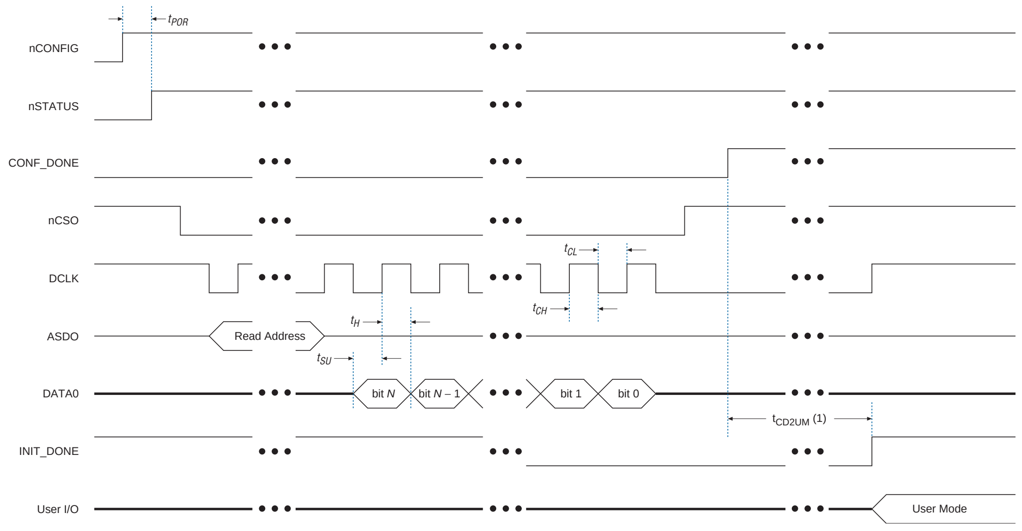
A serial configuration device can be programmed in-system by an external microprocessor using SRunner. SRunner is a software driver developed for embedded serial configuration device programming, which can be easily customized to fit in different embedded systems. SRunner is able to read a raw programming data (**.rpd**) file and write to the serial configuration devices. The serial configuration device programming time using SRunner is comparable to the programming time with the Quartus II software.

For more information about SRunner, see the *SRunner: An Embedded Solution for EPCS Programming* White Paper and the source code on the Altera web site at **www.altera.com**.

For more information on programming serial configuration devices, see the *Serial Configuration Devices (EPCS1 & EPCS4) Data Sheet* in the *Configuration Handbook*.

Figure 7–16 shows the timing waveform for the AS configuration scheme using a serial configuration device.

Figure 7–16. AS Configuration Timing



Note to Figure 7–16:

(1) The initialization clock can come from the Stratix II internal oscillator or the `CLKUSR` pin.

Passive Serial Configuration

PS configuration of Stratix II devices can be performed using an intelligent host, such as a MAX II device or microprocessor with flash memory, an Altera configuration device, or a download cable. In the PS scheme, an external host (MAX II device, embedded processor, configuration device, or host PC) controls configuration. Configuration data is clocked into the target Stratix II devices via the `DATA0` pin at each rising edge of `DCLK`.

☞ The Stratix II decompression and design security feature are fully available when configuring your Stratix II device using PS mode.

Table 7–10 shows the `MSEL` pin settings when using the PS configuration scheme.

Table 7–10. Stratix II MSEL Pin Settings for PS Configuration Schemes

Configuration Scheme	MSEL3	MSEL2	MSEL1	MSEL0
PS	0	0	1	0
PS when using Remote System Upgrade *(1)*	0	1	1	0

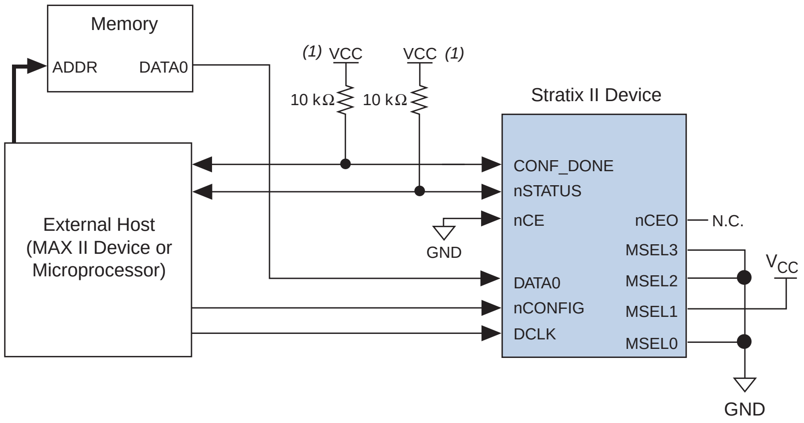
Note to Table 7–10:

(1) This scheme requires that you drive the `RUnLU` pin to specify either remote update or local update. For more information about remote system upgrade in Stratix II devices, see Chapter 8, Remote System Upgrades with Stratix II Devices in Volume 2 of the *Stratix II Device Handbook*.

PS Configuration Using a MAX II Device as an External Host

In the PS configuration scheme, a MAX II device can be used as an intelligent host that controls the transfer of configuration data from a storage device, such as flash memory, to the target Stratix II device. Configuration data can be stored in RBF, HEX, or TTF format. Figure 7–17 shows the configuration interface connections between the Stratix II device and a MAX II device for single device configuration.

Figure 7–17. Single Device PS Configuration Using an External Host



Note to Figure 7–17:

(1) Connect the pull-up resistor to a supply that provides an acceptable input signal for the device. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

Upon power-up, the Stratix II device goes through a POR. The POR delay is dependent on the `PORSEL` pin setting; when `PORSEL` is driven low, the POR time is approximately 100 ms, if `PORSEL` is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold `nSTATUS` low, and tri-state all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If `nIO_pullup` is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If `nIO_pullup` is driven high, the weak pull-up resistors are disabled.

The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *Stratix II Device Handbook.*

The configuration cycle consists of three stages: reset, configuration, and initialization. While `nCONFIG` or `nSTATUS` are low, the device is in reset. To initiate configuration, the MAX II device must generate a low-to-high transition on the `nCONFIG` pin.

V_{CCINT}, V_{CCIO}, and V_{CCPD} of the banks where the configuration and JTAG pins reside need to be fully powered to the appropriate voltage levels in order to begin the configuration process.

When nCONFIG goes high, the device comes out of reset and releases the open-drain nSTATUS pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once nSTATUS is released, the device is ready to receive configuration data and the configuration stage begins. When nSTATUS is pulled high, the MAX II device should place the configuration data one bit at a time on the DATA0 pin. The least significant bit (LSB) of each data byte must be sent first. For example, if the RBF contains the byte sequence 02 1B EE 01 FA, the serial bitstream you should transmit to the device is 0100-0000 1101-1000 0111-0111 1000-0000 0101-1111.

The Stratix II device receives configuration data on its DATA0 pin and the clock is received on the DCLK pin. Data is latched into the device on the rising edge of DCLK. Data is continuously clocked into the target device until CONF_DONE goes high. After the device has received all configuration data successfully, it releases the open-drain CONF_DONE pin, which is pulled high by an external 10-kΩ pull-up resistor. A low-to-high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin.

In Stratix II devices, the initialization clock source is either the Stratix II internal oscillator (typically 10 MHz) or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If the internal oscillator is used, the Stratix II device will provide itself with enough clock cycles for proper initialization. Therefore, if the internal oscillator is the initialization clock source, sending the entire configuration file to the device is sufficient to configure and initialize the device. Driving DCLK to the device after configuration is complete does not affect device operation.

You also have the flexibility to synchronize initialization of multiple devices or to delay initialization by using the CLKUSR option. The **Enable user-supplied start-up clock** (**CLKUSR**) option can be turned on in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. Supplying a clock on CLKUSR will not affect the configuration process. After all configuration data has been accepted and CONF_DONE goes high, CLKUSR will be enabled after the time specified as t_{CD2CU}. After this time period elapses, the Stratix II devices require 299 clock cycles to initialize properly and enter user mode. Stratix II devices support a CLKUSR f_{MAX} of 100 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. The **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If the INIT_DONE pin is used it will be high due to an external 10-kΩ pull-up resistor 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. When initialization is complete, the INIT_DONE pin will be released and pulled high. The MAX II device must be able to detect this low-to-high transition which signals the device has entered user mode. When initialization is complete, the device enters user mode. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design.

To ensure DCLK and DATA0 are not left floating at the end of configuration, the MAX II device must drive them either high or low, whichever is convenient on your board. The DATA[0] pin is available as a user I/O pin after configuration. When the PS scheme is chosen in the Quartus II software, as a default this I/O pin is tri-stated in user mode and should be driven by the MAX II device. To change this default option in the Quartus II software, select the **Dual-Purpose Pins** tab of the **Device & Pin Options** dialog box.

The configuration clock (DCLK) speed must be below the specified frequency to ensure correct configuration. No maximum DCLK period exists, which means you can pause configuration by halting DCLK for an indefinite amount of time.

If an error occurs during configuration, the device drives its nSTATUS pin low, resetting itself internally. The low signal on the nSTATUS pin also alerts the MAX II device that there is an error. If the **Auto-restart configuration after error** option (available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box) is turned on, the Stratix II device releases nSTATUS after a reset time-out period (maximum of 40 µs). After nSTATUS is released and pulled high by a pull-up resistor, the MAX II device can try to reconfigure the target device without needing to pulse nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 40 µs) on nCONFIG to restart the configuration process.

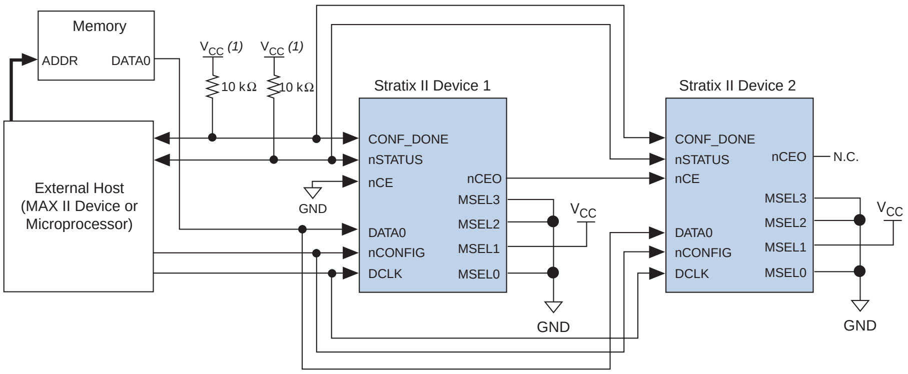
The MAX II device can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. The CONF_DONE pin must be monitored by the MAX II device to detect errors and determine when programming completes. If all configuration data is sent, but CONF_DONE or INIT_DONE have not gone high, the MAX II device must reconfigure the target device.

☞ If the optional CLKUSR pin is being 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).

When the device is in user-mode, you can initiate a reconfiguration by transitioning the nCONFIG pin low-to-high. The nCONFIG pin must be low for at least 40 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 7–18 shows how to configure multiple devices using a MAX II device. This circuit is similar to the PS configuration circuit for a single device, except Stratix II devices are cascaded for multi-device configuration.

Figure 7–18. Multi-Device PS Configuration Using an External Host



Note to Figure 7–18:

(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

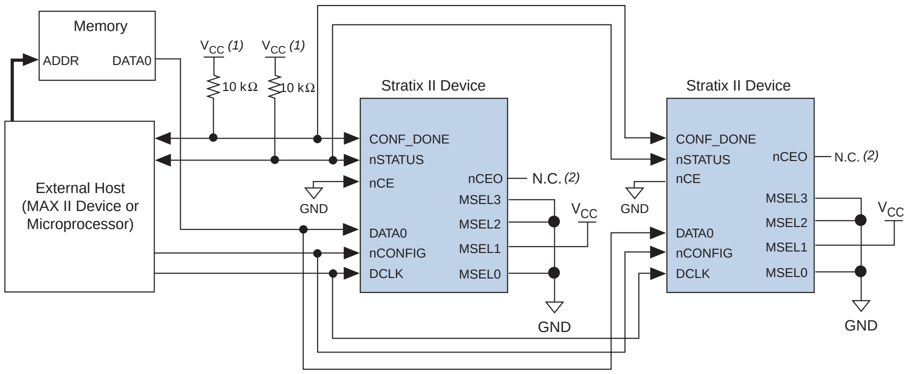
In multi-device PS configuration the first device's nCE pin is connected to GND while its nCEO pin is connected to nCE of the next device in the chain. The last device's nCE input comes from the previous device, while its nCEO pin is left floating. After the first device completes configuration in a 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. The second device in the chain begins configuration within one clock cycle. Therefore, the transfer of data destinations is transparent to the MAX II device. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) are connected to every device in the chain. Configuration signals can require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

Since all nSTATUS and CONF_DONE pins are tied together, if any device detects an error, configuration stops for the entire chain and the entire chain must be reconfigured. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This behavior is similar to a single device detecting an error.

If the **Auto-restart configuration after error** option is turned on, the devices release their nSTATUS pins after a reset time-out period (maximum of 40 µs). After all nSTATUS pins are released and pulled high, the MAX II device can try to reconfigure the chain without needing to pulse nCONFIG low. If this option is turned off, the MAX II device must generate a low-to-high transition (with a low pulse of at least 40 µs) on nCONFIG to restart the configuration process.

In your system, you can have multiple devices that contain the same configuration data. To support this configuration scheme, all device nCE inputs are tied to GND, while nCEO pins are left floating. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) are connected to every device in the chain. Configuration signals can require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device. Devices must be the same density and package. All devices will start and complete configuration at the same time. Figure 7–19 shows multi-device PS configuration when both Stratix II devices are receiving the same configuration data.

Figure 7–19. Multiple-Device PS Configuration When Both devices Receive the Same Data



Notes to Figure 7–19:

(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

(2) The nCEO pins of both devices are left unconnected when configuring the same configuration data into multiple devices.

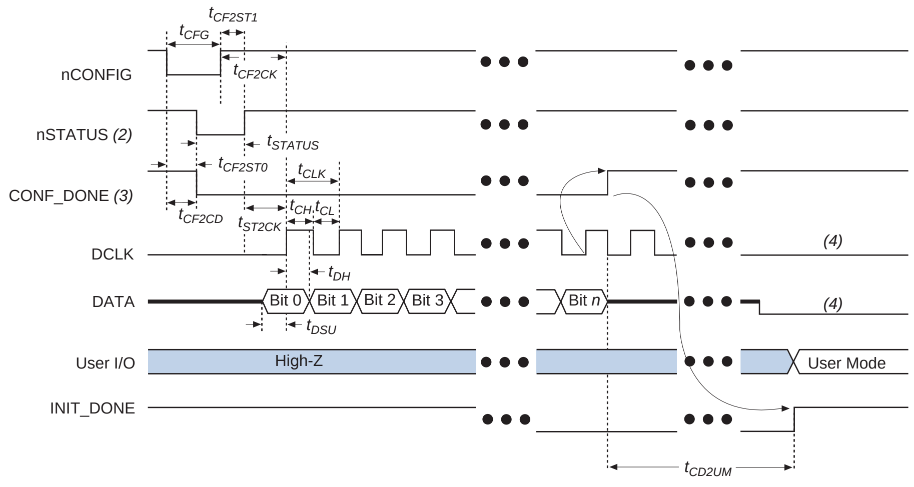
You can use a single configuration chain to configure Stratix II devices with other Altera devices. To ensure that all devices in the chain complete configuration at the same time or that an error flagged by one device initiates reconfiguration in all devices, all of the device `CONF_DONE` and `nSTATUS` pins must be tied together.

For more information on configuring multiple Altera devices in the same configuration chain, see *Configuring Mixed Altera device Chains* in the *Configuration Handbook*.

PS Configuration Timing

Figure 7–20 shows the timing waveform for PS configuration when using a MAX II device as an external host.

Figure 7–20. PS Configuration Timing Waveform *Note (1)*



Notes to Figure 7–20:

(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 more convenient. `DATA[0]` is available as a user I/O pin after configuration and the state of this pin depends on the dual-purpose pin settings.

Table 7–11 defines the timing parameters for Stratix II devices for PS configuration.

Table 7–11. PS Timing Parameters for Stratix II Devices *Note (1)*

Symbol	Parameter	Min	Max	Units
t_{POR}	POR delay	12	100	ms
t_{CF2CD}	`nCONFIG` low to `CONF_DONE` low		800	ns
t_{CF2ST0}	`nCONFIG` low to `nSTATUS` low		800	ns
t_{CFG}	`nCONFIG` low pulse width	2		µs
t_{STATUS}	`nSTATUS` low pulse width	10	100 *(2)*	µs
t_{CF2ST1}	`nCONFIG` high to `nSTATUS` high		100 *(2)*	µs
t_{CF2CK}	`nCONFIG` high to first rising edge on `DCLK`	100		µs
t_{ST2CK}	`nSTATUS` high to first rising edge of `DCLK`	2		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` frequency		100	MHz
t_R	Input rise time		40	ns
t_F	Input fall time		40	ns
t_{CD2UM}	`CONF_DONE` high to user mode *(3)*	20	40	µs
t_{CD2CU}	`CONF_DONE` high to `CLKUSR` enabled	4 × maximum `DCLK` period		
t_{CD2UMC}	`CONF_DONE` high to user mode with CLKUSR option on	t_{CD2CU} + (299 × CLKUSR period)		

Notes to Table 7–11:

(1) This information is preliminary.
(2) This value is applicable if users do not delay configuration by extending the `nCONFIG` or `nSTATUS` low pulse width.
(3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting the device.

Device configuration options and how to create configuration files are discussed further in *Software Settings* in Volume 2 of the *Configuration Handbook*.

An example PS design that uses a MAX II device as the external host for configuration will be available when devices are available.

PS Configuration Using a Microprocessor

In the PS configuration scheme, a microprocessor can control the transfer of configuration data from a storage device, such as flash memory, to the target Stratix II device.

All information in the "PS Configuration Using a MAX II Device as an External Host" section is also applicable when using a microprocessor as an external host. Refer to that section for all configuration and timing information.

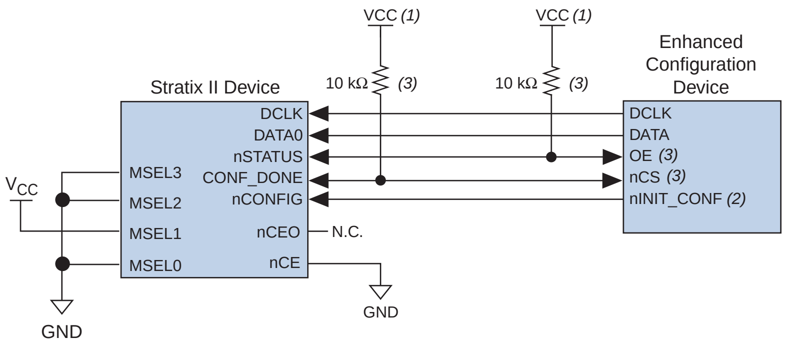
PS Configuration Using a Configuration Device

You can use an Altera configuration device, such as an enhanced configuration device, EPC2, or EPC1 device, to configure Stratix II devices using a serial configuration bitstream. Configuration data is stored in the configuration device. Figure 7–21 shows the configuration interface connections between the Stratix II device and a configuration device.

☞ The figures in this chapter only show the configuration-related pins and the configuration pin connections between the configuration device and the device.

For more information on the enhanced configuration device and flash interface pins (such as `PGM[2..0]`, `EXCLK`, `PORSEL`, `A[20..0]`, and `DQ[15..0]`), see the *Enhanced Configuration Devices* (*EPC4, EPC8, & EPC16*) *Data Sheet*.

Figure 7–21. Single Device PS Configuration Using an Enhanced Configuration Device



Notes to Figure 7–21:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The `nINIT_CONF` pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active, meaning an external pull-up resistor should not be used on the `nINIT_CONF`-`nCONFIG` line. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' `OE` and `nCS` pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

The value of the internal pull-up resistors on the enhanced configuration devices and EPC2 devices can be found in the Operating Conditions table of the *Enhanced Configuration Devices* (*EPC4, EPC8, & EPC16*) *Data Sheet* or the *Configuration Devices for SRAM-based LUT Devices Data Sheet*.

When using enhanced configuration devices or EPC2 devices, `nCONFIG` of the device can be connected to `nINIT_CONF` of the configuration device, which allows the `INIT_CONF` JTAG instruction to initiate device configuration. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used or not available (e.g., on EPC1 devices), `nCONFIG` must be pulled to V_{CC} either directly or through a resistor. An internal pull-up resistor on the `nINIT_CONF` pin is always active in enhanced configuration devices and EPC2 devices, which means an external pull-up resistor should not be used if `nCONFIG` is tied to `nINIT_CONF`.

Upon power-up, the Stratix II device goes through a POR. The POR delay is dependent on the `PORSEL` pin setting. When `PORSEL` is driven low, the POR time is approximately 100 ms. If `PORSEL` is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold `nSTATUS` low, and tri-state all user I/O pins. The configuration device also goes through a POR delay to allow the power supply to stabilize. The

POR time for EPC2 and EPC1 devices is 200 ms (maximum). The POR time for enhanced configuration devices can be set to either 100 ms or 2 ms, depending on its PORSEL pin setting. If the PORSEL pin is connected to GND, the POR delay is 100 ms. If the PORSEL pin is connected to V_{CC}, the POR delay is 2 ms. 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 selecting a POR time, you need to ensure that the device completes power-up before the enhanced configuration device exits POR. Altera recommends that you choose a POR time for the Stratix II device of 12 ms, while selecting a POR time for the enhanced configuration device of 100 ms.

When both devices complete POR, they release their open-drain OE or nSTATUS pin, which is then pulled high by a pull-up resistor. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If nIO_pullup is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If nIO_pullup is driven high, the weak pull-up resistors are disabled.

The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *DC & Switching Characteristics* chapter in the *Stratix II Device Handbook*.

When the power supplies have reached the appropriate operating voltages, the target device senses the low-to-high transition on nCONFIG and initiates the configuration cycle. The configuration cycle consists of three stages: reset, configuration, and initialization. While nCONFIG or nSTATUS are low, the device is in reset. The beginning of configuration can be delayed by holding the nCONFIG or nSTATUS pin low.

To begin configuration, power the V_{CCINT}, V_{CCIO}, and V_{CCPD} voltages (for the banks where the configuration and JTAG pins reside) to the appropriate voltage levels.

When nCONFIG goes high, the device comes out of reset and releases the nSTATUS pin, which is pulled high by a pull-up resistor. Enhanced configuration and EPC2 devices have an optional internal pull-up resistor on the OE pin. This option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If this internal pull-up resistor is not used, an external 10-kΩ pull-up resistor on the OE-nSTATUS line is required. Once nSTATUS is released, the device is ready to receive configuration data and the configuration stage begins.

When nSTATUS is pulled high, OE of the configuration device also goes high and the configuration device clocks data out serially to the device using its internal oscillator. The Stratix II device receives configuration data on its DATA0 pin and the clock is received on the DCLK pin. Data is latched into the device on the rising edge of DCLK.

After the device has received all configuration data successfully, it releases the open-drain CONF_DONE pin, which is pulled high by a pull-up resistor. Since CONF_DONE is tied to the configuration device's nCS pin, the configuration device is disabled when CONF_DONE goes high. Enhanced configuration and EPC2 devices have an optional internal pull-up resistor on the nCS pin. This option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If this internal pull-up resistor is not used, an external 10-kΩ pull-up resistor on the nCS-CONF_DONE line is required. A low-to-high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin.

In Stratix II devices, the initialization clock source is either the Stratix II internal oscillator (typically 10 MHz) or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If you are using internal oscillator, the Stratix II device will supply itself with enough clock cycles for proper initialization. You also have the flexibility to synchronize initialization of multiple devices or to delay initialization by using the CLKUSR option. You can turn on the **Enable user-supplied start-up clock** (**CLKUSR**) option in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. Supplying a clock on CLKUSR will not affect the configuration process. After all configuration data has been accepted and CONF_DONE goes high, CLKUSR will be enabled after the time specified as t_{CD2CU}. After this time period elapses, the Stratix II devices require 299 clock cycles to initialize properly and enter user mode. Stratix II devices support a CLKUSR f_{MAX} of 100 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. The **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If you are using the INIT_DONE pin, it will be high due to an external 10-kΩ pull-up resistor 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 goes low. When initialization is complete, the INIT_DONE pin is released and pulled high. This low-to-high transition signals that the device has entered user mode. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design. Enhanced configuration devices and EPC2 devices drive DCLK low and DATA0 high (EPC1 devices drive DCLK low and tri-state DATA) at the end of configuration.

If an error occurs during configuration, the device drives its nSTATUS pin low, resetting itself internally. Since the nSTATUS pin is tied to OE, the configuration device will also be reset. If the **Auto-restart configuration after error** option, available in the Quartus II software, from the **General** tab of the **Device & Pin Options** dialog box is turned on, the device automatically initiates reconfiguration if an error occurs. The Stratix II device will release its nSTATUS pin after a reset time-out period (maximum of 40 µs). When the nSTATUS pin is released and pulled high by a pull-up resistor, the configuration device reconfigures the chain. If this option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low for at least 40 µs to restart configuration. The external system can pulse nCONFIG if nCONFIG is under system control rather than tied to V_{CC}.

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 device has not configured successfully. Enhanced configuration devices wait for 64 DCLK cycles after the last configuration bit was sent for CONF_DONE to reach a high state. EPC2 devices wait for 16 DCLK cycles. In this case, the configuration device pulls its OE pin low, driving the target device's nSTATUS pin low. If the **Auto-restart configuration after error** option is set in the software, the target device resets and then releases its nSTATUS pin after a reset time-out period (maximum of 40 µs). When nSTATUS returns to a logic high level, the configuration device tries to reconfigure the device.

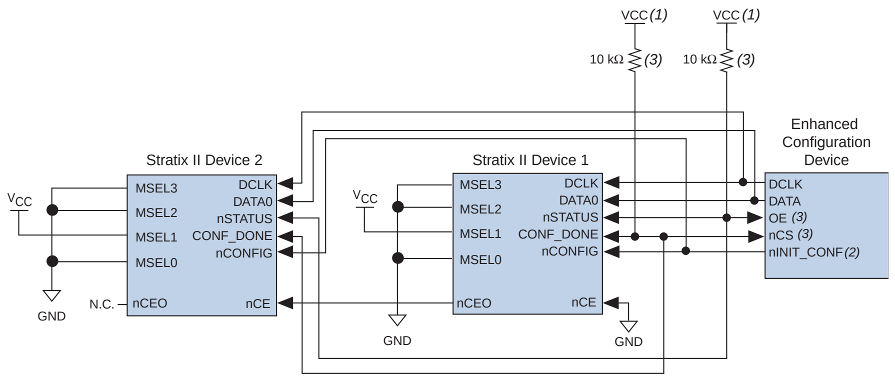
When CONF_DONE is sensed low after configuration, the configuration device recognizes that the target device has not configured successfully. Therefore, your system should not pull CONF_DONE low to delay initialization. Instead, use the CLKUSR option to synchronize the initialization of multiple devices that are not in the same configuration chain. Devices in the same configuration chain will initialize together if their CONF_DONE pins are tied together.

☞ If you are using the optional CLKUSR pin 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).

When the device is in user-mode, pulling the nCONFIG pin low will initiate a reconfiguration. The nCONFIG pin should be low for at least 40 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Since CONF_DONE is pulled low, this will activate the configuration device since it will see its nCS pin drive low. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 7–22 shows how to configure multiple devices with an enhanced configuration device. This circuit is similar to the configuration device circuit for a single device, except Stratix II devices are cascaded for multi-device configuration.

Figure 7–22. Multi-Device PS Configuration Using an Enhanced Configuration Device



Notes to Figure 7–22:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The nINIT_CONF pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active, meaning an external pull-up resistor should not be used on the nINIT_CONF-nCONFIG line. The nINIT_CONF pin does not need to be connected if its functionality is not used. If nINIT_CONF is not used, nCONFIG must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' OE and nCS pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

Enhanced configuration devices cannot be cascaded.

When performing multi-device configuration, you must generate the configuration device's POF from each project's SOF. You can combine multiple SOFs using the **Convert Programming Files** window in the Quartus II software.

For more information on how to create configuration files for multi-device configuration chains, see the *Software Settings* chapter of the *Configuration Handbook.*

In multi-device PS configuration, 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, prompting the second device to begin configuration. All other configuration pins (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) are connected to every device in the chain. Configuration signals can require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DCLK and DATA lines are buffered for every fourth device.

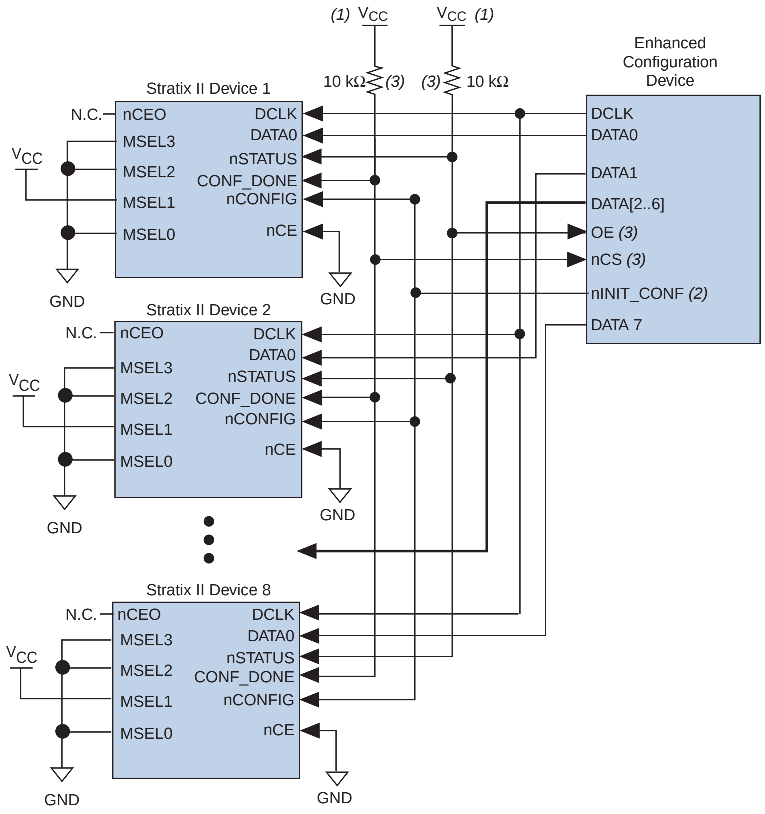
When configuring multiple devices, configuration does not begin until all devices release their OE or nSTATUS pins. Similarly, since all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

Since all nSTATUS and CONF_DONE pins are tied together, if any device detects an error, configuration stops for the entire chain and the entire chain must be reconfigured. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This low signal drives the OE pin low on the enhanced configuration device and drives nSTATUS low on all devices, causing them to enter a reset state. This behavior is similar to a single device detecting an error.

If the **Auto-restart configuration after error** option is turned on, the devices will automatically initiate reconfiguration if an error occurs. The devices will release their nSTATUS pins after a reset time-out period (maximum of 40 µs). When all the nSTATUS pins are released and pulled high, the configuration device tries to reconfigure the chain. If the **Auto-restart configuration after error** option is turned off, the external system must monitor nSTATUS for errors and then pulse nCONFIG low for at least 40 µs to restart configuration. The external system can pulse nCONFIG if nCONFIG is under system control rather than tied to V_{CC}.

The enhanced configuration devices also support parallel configuration of up to eight devices. The n-bit (*n* = 1, 2, 4, or 8) PS configuration mode allows enhanced configuration devices to concurrently configure devices or a chain of devices. In addition, these devices do not have to be the same device family or density as they can be any combination of Altera devices. An individual enhanced configuration device DATA line is available for each targeted device. Each DATA line can also feed a daisy chain of devices. Figure 7–23 shows how to concurrently configure multiple devices using an enhanced configuration device.

Figure 7–23. Concurrent PS Configuration of Multiple Devices Using an Enhanced Configuration Device



Notes to Figure 7–23:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The `nINIT_CONF` pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active, meaning an external pull-up resistor should not be used on the `nINIT_CONF`-`nCONFIG` line. The `nINIT_CONF` pin does not need to be connected if its functionality is not used. If `nINIT_CONF` is not used, `nCONFIG` must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' `OE` and `nCS` pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

The Quartus II software only allows the selection of n-bit PS configuration modes, where n must be 1, 2, 4, or 8. However, you can use these modes to configure any number of devices from 1 to 8. When configuring SRAM-based devices using n-bit PS modes, use Table 7–12 to select the appropriate configuration mode for the fastest configuration times.

Table 7–12. Recommended Configuration Using n-Bit PS Modes

Number of Devices *(1)*	Recommended Configuration Mode
1	1-bit PS
2	2-bit PS
3	4-bit PS
4	4-bit PS
5	8-bit PS
6	8-bit PS
7	8-bit PS
8	8-bit PS

Note to Table 7–12:

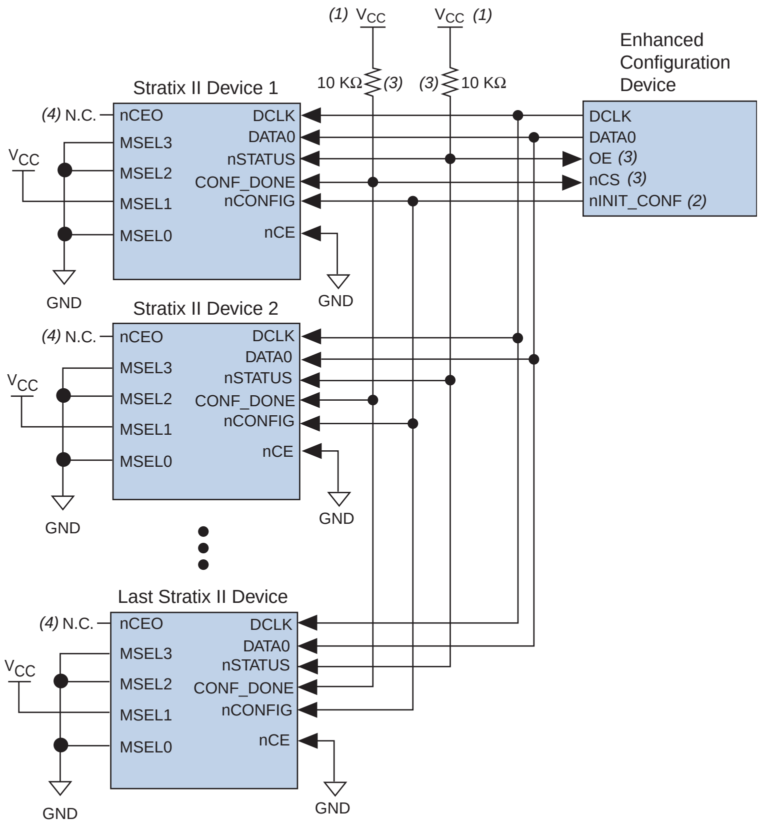
(1) Assume that each `DATA` line is only configuring one device, not a daisy chain of devices.

For example, if you configure three devices, you would use the 4-bit PS mode. For the `DATA0`, `DATA1`, and `DATA2` lines, the corresponding SOF data is transmitted from the configuration device to the device. For `DATA3`, you can leave the corresponding Bit3 line blank in the Quartus II software. On the PCB, leave the `DATA3` line from the enhanced configuration device unconnected.

Alternatively, you can daisy chain two devices to one `DATA` line while the other `DATA` lines drive one device each. For example, you could use the 2-bit PS mode to drive two devices with `DATA` Bit0 (two EP2S10 devices) and the third device (EP2S20 device) with `DATA` Bit1. This 2-bit PS configuration scheme requires less space in the configuration flash memory, but can increase the total system configuration time.

A system may have multiple devices that contain the same configuration data. To support this configuration scheme, all device `nCE` inputs are tied to GND, while `nCEO` pins are left floating. All other configuration pins (`nCONFIG`, `nSTATUS`, `DCLK`, `DATA0`, and `CONF_DONE`) are connected to every device in the chain. Configuration signals can require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the `DCLK` and `DATA` lines are buffered for every fourth device. Devices must be the same density and package. All devices will start and complete configuration at the same time. Figure 7–24 shows multi-device PS configuration when the Stratix II devices are receiving the same configuration data.

Figure 7–24. Multiple-Device PS Configuration Using an Enhanced Configuration Device When devices Receive the Same Data



Notes to Figure 7–24:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The nINIT_CONF pin is available on enhanced configuration devices and has an internal pull-up resistor that is always active, meaning an external pull-up resistor should not be used on the nINIT_CONF-nCONFIG line. The nINIT_CONF pin does not need to be connected if its functionality is not used. If nINIT_CONF is not used, nCONFIG must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' OE and nCS pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

(4) The nCEO pins of all devices are left unconnected when configuring the same configuration data into multiple devices.

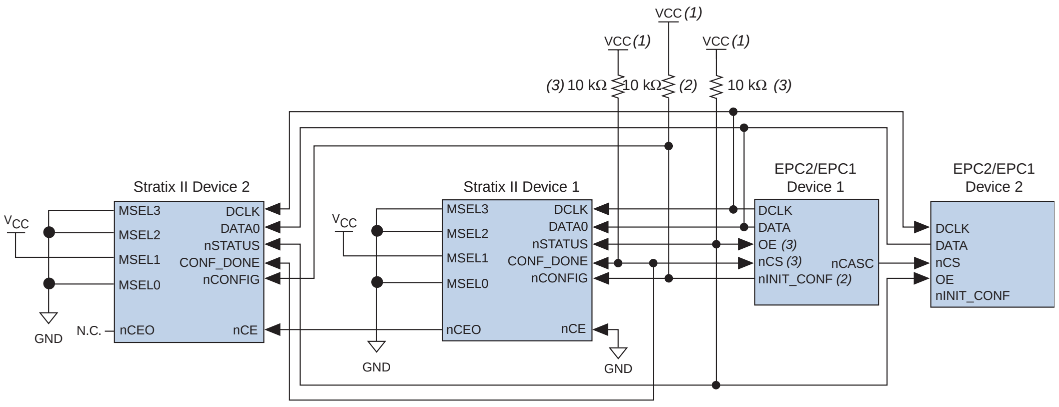
You can cascade several EPC2 or EPC1 devices to configure multiple Stratix II devices. The first configuration device in the chain is the master configuration device, while the subsequent devices are the slave devices. The master configuration device sends DCLK to the Stratix II devices and to the slave configuration devices. The first EPC device's nCS pin is connected to the CONF_DONE pins of the devices, while its nCASC pin is connected to nCS of the next configuration device in the chain. The last device's nCS input comes from the previous device, while its nCASC pin is left floating. When all data from the first configuration device is sent, it drives nCASC low, which in turn drives nCS on the next configuration device. Since a configuration device requires less than one clock cycle to activate a subsequent configuration device, the data stream is uninterrupted.

☞ Enhanced configuration devices cannot be cascaded.

Since all nSTATUS and CONF_DONE pins are tied together, if any device detects an error, the master configuration device stops configuration for the entire chain and the entire chain must be reconfigured. For example, if the master configuration device does not detect CONF_DONE going high at the end of configuration, it resets the entire chain by pulling its OE pin low. This low signal drives the OE pin low on the slave configuration device(s) and drives nSTATUS low on all devices, causing them to enter a reset state. This behavior is similar to the device detecting an error in the configuration data.

Figure 7–25 shows how to configure multiple devices using cascaded EPC2 or EPC1 devices.

Figure 7–25. Multi-Device PS Configuration Using Cascaded EPC2 or EPC1 Devices



Notes to Figure 7–25:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) The nINIT_CONF pin (available on enhanced configuration devices and EPC2 devices only) has an internal pull-up resistor that is always active, meaning an external pull-up resistor should not be used on the nINIT_CONF-nCONFIG line. The nINIT_CONF pin does not need to be connected if its functionality is not used. If nINIT_CONF is not used or not available (e.g., on EPC1 devices), nCONFIG must be pulled to V_{CC} either directly or through a resistor.

(3) The enhanced configuration devices' and EPC2 devices' OE and nCS pins have internal programmable pull-up resistors. External 10-kΩ pull-up resistors should be used. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-ups on configuration device** option when generating programming files.

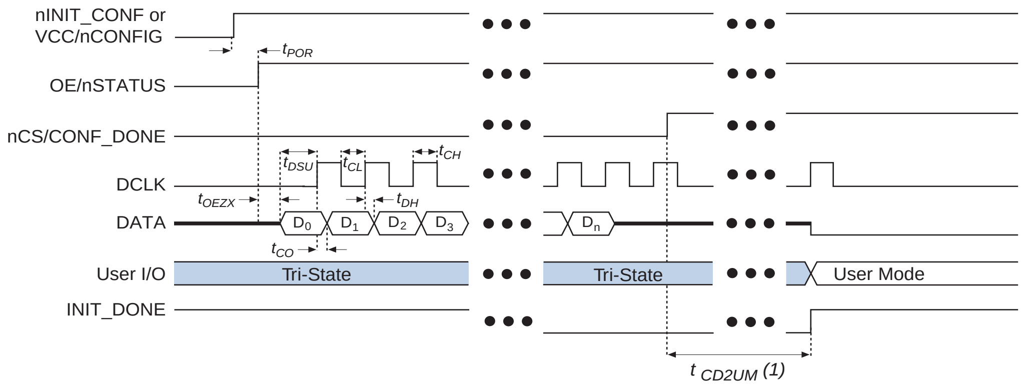
When using enhanced configuration devices or EPC2 devices, nCONFIG of the device can be connected to nINIT_CONF of the configuration device, allowing the INIT_CONF JTAG instruction to initiate device configuration. The nINIT_CONF pin does not need to be connected if its functionality is not used. If you are not using nINIT_CONF, or if it isn't available(e.g. on EPC1 devices), nCONFIG must be pulled to V_{CC} either directly or through a resistor. An internal pull-up resistor on the nINIT_CONF pin is always active in the enhanced configuration devices and the EPC2 devices, which means that you shouldn't be using an external pull-up resistor if nCONFIG is tied to nINIT_CONF. If you are using multiple EPC2 devices to configure a Stratix II device(s), only the first EPC2 has its nINIT_CONF pin tied to the device's nCONFIG pin.

You can use a single configuration chain to configure Stratix II devices with other Altera devices. To ensure that all devices in the chain complete configuration at the same time or that an error flagged by one device initiates reconfiguration in all devices, all of the device CONF_DONE and nSTATUS pins must be tied together.

For more information on configuring multiple Altera devices in the same configuration chain, see the *Configuring Mixed Altera device Chains* chapter in the *Configuration Handbook*.

Figure 7–26 shows the timing waveform for the PS configuration scheme using a configuration device.

Figure 7–26. Stratix II PS Configuration Using a Configuration Device Timing Waveform



Note to Figure 7–26:
(1) The initialization clock can come from the Stratix II internal oscillator or the CLKUSR pin.

For timing information, refer to the *Enhanced Configuration Devices (EPC4, EPC8 & EPC16) Data Sheet* or the *Configuration Devices for SRAM-Based LUT Devices Data Sheet* in the *Configuration Handbook*.

Device configuration options and how to create configuration files are discussed further in the *Software Settings* chapter of the *Configuration Handbook*.

PS Configuration Using a Download Cable

In this section, the generic term "download cable" includes the Altera USB-Blaster™ universal serial bus (USB) port download cable, MasterBlaster™ serial/USB communications cable, ByteBlaster™ II parallel port download cable, and the ByteBlaster MV parallel port download cable.

In PS configuration with a download cable, an intelligent host (such as a PC) transfers data from a storage device to the device via the USB Blaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV cable.

Upon power-up, the Stratix II device goes through a POR. The POR delay is dependent on the PORSEL pin setting. When PORSEL is driven low, the POR time is approximately 100 ms. If PORSEL is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold nSTATUS low, and tri-state all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If nIO_pullup is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If nIO_pullup is driven high, the weak pull-up resistors are disabled.

The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *DC & Switching Characteristics* chapter in the *Stratix II Device Handbook*.

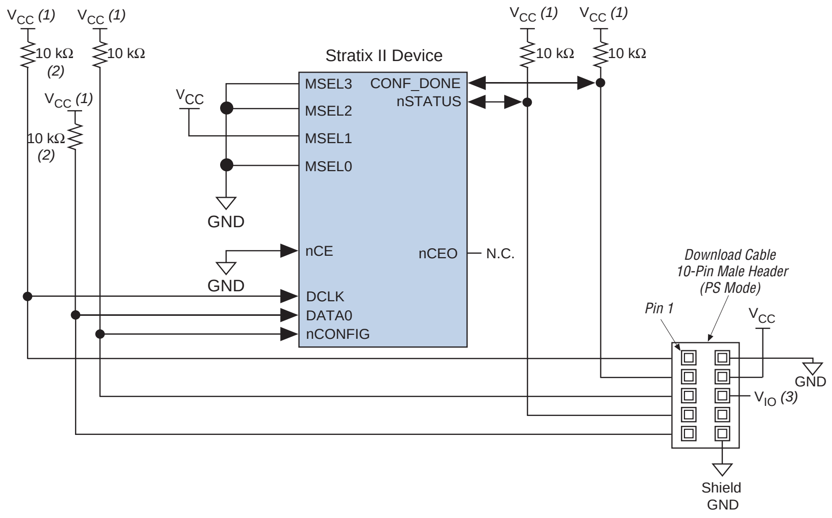
The configuration cycle consists of three stages: reset, configuration and initialization. While nCONFIG or nSTATUS are low, the device is in reset. To initiate configuration in this scheme, the download cable generates a low-to-high transition on the nCONFIG pin.

To begin configuration, power the V_{CCINT}, V_{CCIO}, and V_{CCPD} voltages (for the banks where the configuration and JTAG pins reside) to the appropriate voltage levels.

When nCONFIG goes high, the device comes out of reset and releases the open-drain nSTATUS pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once nSTATUS is released the device is ready to receive configuration data and the configuration stage begins. The programming hardware or download cable then places the configuration data one bit at a time on the device's DATA0 pin. The configuration data is clocked into the target device until CONF_DONE goes high.

When using a download cable, setting the **Auto-restart configuration after error** option does not affect the configuration cycle because you must manually restart configuration in the Quartus II software 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 device using the Quartus II programmer and 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 device with the Quartus II programmer and a download cable. Figure 7–27 shows PS configuration for Stratix II devices using a USB Blaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV cable.

Figure 7–27. PS Configuration Using a USB Blaster, MasterBlaster, ByteBlaster II or ByteBlasterMV Cable



Notes to Figure 7–27:

(1) The pull-up resistor should be connected to the same supply voltage as the USB Blaster, MasterBlaster (V_{IO} pin), ByteBlaster II or ByteBlasterMV cable.

(2) The pull-up resistors on `DATA0` and `DCLK` are only needed if the download cable is the only configuration scheme used on your board. This ensures that `DATA0` and `DCLK` are not left floating after configuration. For example, if you are also using a configuration device, the pull-up resistors on `DATA0` and `DCLK` are not needed.

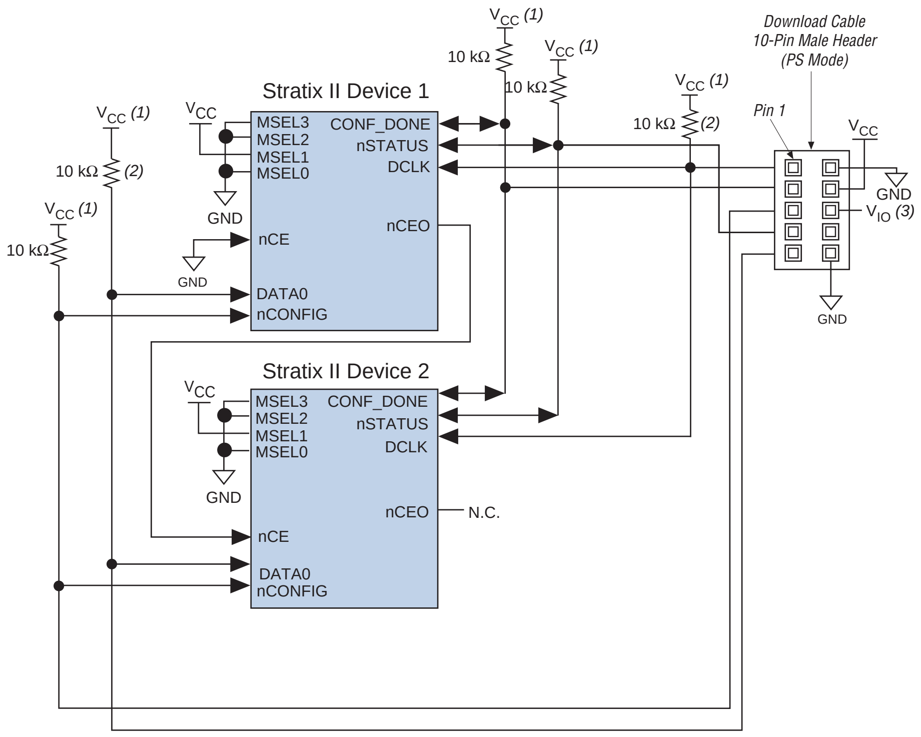
(3) Pin 6 of the header is a V_{IO} reference voltage for the MasterBlaster output driver. V_{IO} should match the device's V_{CCIO}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB Blaster and ByteBlaster II cables, this pin is connected to `nCE` when it is used for active serial programming, otherwise it is a no connect.

You can use a download cable to configure multiple Stratix II devices by connecting each device's `nCEO` pin to the subsequent device's `nCE` pin. The first device's `nCE` pin is connected to GND while its `nCEO` pin is connected to the `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. All other configuration pins, `nCONFIG`, `nSTATUS`, `DCLK`, `DATA0`, and `CONF_DONE` are connected to every 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. The **Auto-restart configuration after error** option does not affect the configuration cycle because you must manually restart configuration in the Quartus II software when an error occurs.

Figure 7–28 shows how to configure multiple Stratix II devices with a download cable.

Figure 7–28. Multi-Device PS Configuration using a USB Blaster, MasterBlaster, ByteBlaster II or ByteBlasterMV Cable



Notes to Figure 7–28:

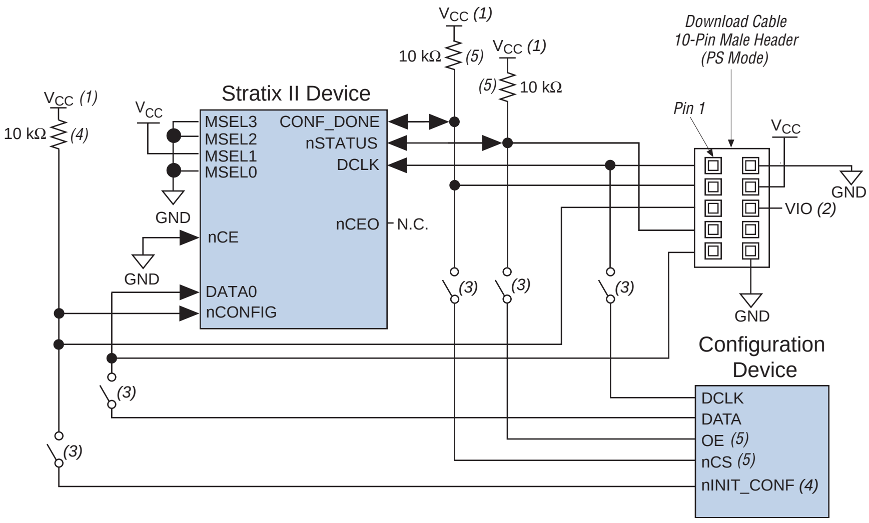
(1) The pull-up resistor should be connected to the same supply voltage as the USB Blaster, MasterBlaster (V_{IO} pin), ByteBlaster II or ByteBlasterMV cable.

(2) The pull-up resistors on DATA0 and DCLK are only needed if the download cable is the only configuration scheme used on your board. This is to ensure that DATA0 and DCLK are not left floating after configuration. For example, if you are also using a configuration device, the pull-up resistors on DATA0 and DCLK are not needed.

(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}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB Blaster and ByteBlaster II cables, this pin is connected to nCE when it is used for active serial programming, otherwise it is a no connect.

If you are using a download cable to configure device(s) on a board that also has configuration devices, electrically isolate the configuration device from the target device(s) and cable. One way of isolating the configuration device is to add logic, such as a multiplexer, that can select between the configuration device and the cable. The multiplexer chip allows bidirectional transfers on the nSTATUS and CONF_DONE signals. Another option is to add switches to the five common signals (nCONFIG, nSTATUS, DCLK, DATA0, and CONF_DONE) between the cable and the configuration device. The last option is to remove the configuration device from the board when configuring the device with the cable. Figure 7–29 shows a combination of a configuration device and a download cable to configure an device.

Figure 7–29. PS Configuration with a Download Cable & Configuration Device Circuit



Notes to Figure 7–29:

(1) The pull-up resistor should be connected to the same supply voltage as the configuration device.

(2) 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}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB Blaster and ByteBlaster II cables, this pin is connected to nCE when it is used for active serial programming, otherwise it is a no connect.

(3) You should not attempt configuration with a download cable while a configuration device is connected to a Stratix II 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.

(4) The nINIT_CONF pin (available on enhanced configuration devices and EPC2 devices only) has an internal pull-up resistor that is always active. This means an external pull-up resistor should not be used on the nINIT_CONF-nCONFIG line. The nINIT_CONF pin does not need to be connected if its functionality is not used. If nINIT_CONF is not used or not available (e.g., on EPC1 devices), nCONFIG must be pulled to V_{CC} either directly or through a resistor.

(5) The enhanced configuration devices' and EPC2 devices' OE and nCS pins have internal programmable pull-up resistors. If internal pull-up resistors are used, external pull-up resistors should not be used on these pins. The internal pull-up resistors are used by default in the Quartus II software. To turn off the internal pull-up resistors, check the **Disable nCS and OE pull-up resistors on configuration device** option when generating programming files.

For more information on how to use the USB Blaster, MasterBlaster, ByteBlaster II or ByteBlasterMV cables, refer to the following data sheets:

- USB Blaster USB Port Download Cable Data Sheet
- MasterBlaster Serial/USB Communications Cable Data Sheet
- ByteBlaster II Parallel Port Download Cable Data Sheet
- ByteBlasterMV Parallel Port Download Cable Data Sheet

Passive Parallel Asynchronous Configuration

Passive parallel asynchronous (PPA) configuration uses an intelligent host, such as a microprocessor, to transfer configuration data from a storage device, such as flash memory, to the target Stratix II device.

Configuration data can be stored in RBF, HEX, or TTF format. The host system outputs byte-wide data and the accompanying strobe signals to the device. When using PPA, pull the `DCLK` pin high through a 10-kΩ pull-up resistor to prevent unused configuration input pins from floating.

You cannot use the Stratix II decompression and design security feature if you are configuring your Stratix II device using PPA mode.

Table 7–13 shows the `MSEL` pin settings when using the PS configuration scheme.

Table 7–13. Stratix II MSEL Pin Settings for PPA Configuration Schemes

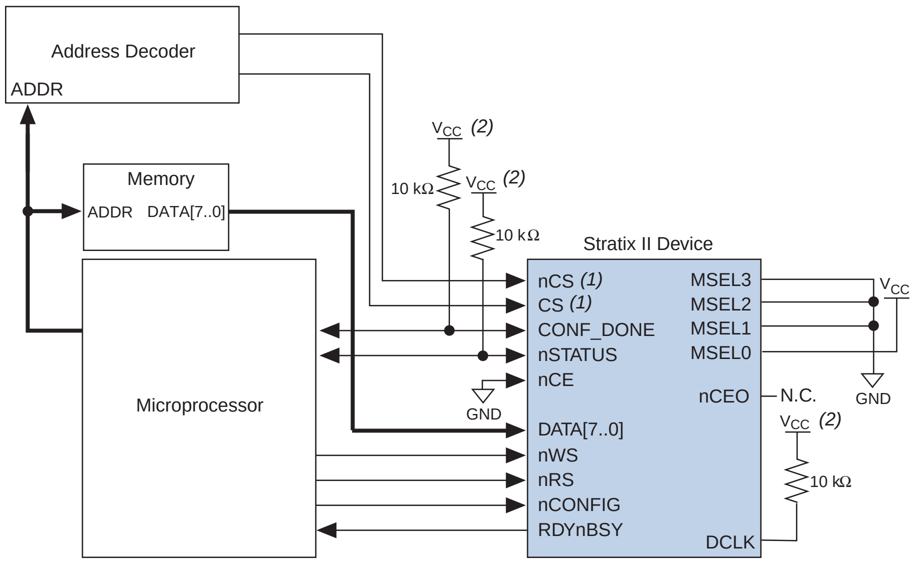
Configuration Scheme	MSEL3	MSEL2	MSEL1	MSEL0
PPA	0	0	0	1
Remote System Upgrade PPA *(1)*	0	1	0	1

Notes to Table 7–13:

(1) This scheme requires that you drive the `RUnLU` pin to specify either remote update or local update. For more information about remote system upgrade in Stratix II devices, see the Chapter 8, Remote System Upgrades with Stratix II Devices in Volume 2 of the *Stratix II Device Handbook.*

Figure 7–30 shows the configuration interface connections between the device and a microprocessor for single device PPA configuration. The microprocessor or an optional address decoder can control the device's chip select pins, `nCS` and `CS`. The address decoder allows the microprocessor to select the Stratix II device by accessing a particular address, which simplifies the configuration process. Hold the `nCS` and `CS` pins active during configuration and initialization.

Figure 7–30. Single Device PPA Configuration Using a Microprocessor



Notes to Figure 7–30:

(1) If not used, the `CS` pin can be connected to V_{CC} directly. If not used, the `nCS` pin can be connected to GND directly.

(2) The pull-up resistor should be connected to a supply that provides an acceptable input signal for the device. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

During PPA configuration, it is only required to use either the `nCS` or `CS` pin. Therefore, if you are using only one chip-select input, the other must be tied to the active state. For example, `nCS` can be tied to ground while `CS` is toggled to control configuration. The device's `nCS` or `CS` pins can be toggled during PPA configuration if the design meets the specifications set for t_{CSSU}, t_{WSP}, and t_{CSH} listed in Table 7–14.

Upon power-up, the Stratix II device goes through a POR. The POR delay is dependent on the `PORSEL` pin setting. When `PORSEL` is driven low, the POR time is approximately 100 ms. If `PORSEL` is driven high, the POR time is approximately 12 ms. During POR, the device will reset, hold `nSTATUS` low, and tri-state all user I/O pins. Once the device successfully exits POR, all user I/O pins continue to be tri-stated. If `nIO_pullup` is driven low during power-up and configuration, the user I/O pins and dual-purpose I/O pins will have weak pull-up resistors which are on (after POR) before and during configuration. If `nIO_pullup` is driven high, the weak pull-up resistors are disabled.

☞ The value of the weak pull-up resistors on the I/O pins that are on before and during configuration can be found in the *DC & Switching Characteristics* chapter in the *Stratix II Device Handbook*.

The configuration cycle consists of three stages: reset, configuration and initialization. While nCONFIG or nSTATUS are low, the device is in reset. To initiate configuration, the microprocessor must generate a low-to-high transition on the nCONFIG pin.

☞ To begin configuration, power the V_{CCINT}, V_{CCIO}, and V_{CCPD} voltages (for the banks where the configuration and JTAG pins reside) to the appropriate voltage levels.

When nCONFIG goes high, the device comes out of reset and releases the open-drain nSTATUS pin, which is then pulled high by an external 10-kΩ pull-up resistor. Once nSTATUS is released the device is ready to receive configuration data and the configuration stage begins. When nSTATUS is pulled high, the microprocessor should then assert the target device's nCS pin low and/or CS pin high. Next, the microprocessor places an 8-bit configuration word (one byte) on the target device's DATA[7..0] pins and pulses the nWS pin low.

On the rising edge of nWS, the target device latches in a byte of configuration data and drives its RDYnBSY signal low, which indicates it is processing the byte of configuration data. The microprocessor can then perform other system functions while the Stratix II device is processing the byte of configuration data.

During the time RDYnBSY is low, the Stratix II device internally processes the configuration data using its internal oscillator (typically 100 MHz). When the device is ready for the next byte of configuration data, it will drive RDYnBSY high. If the microprocessor senses a high signal when it polls RDYnBSY, the microprocessor sends the next byte of configuration data to the device.

Alternatively, the nRS signal can be strobed low, causing the RDYnBSY signal to appear on DATA7. Because RDYnBSY does not need to be monitored, this pin doesn't need to be connected to the microprocessor. Do not drive data onto the data bus while nRS is low because it will cause contention on the DATA7 pin. If you are not using the nRS pin to monitor configuration, it should be tied high.

To simplify configuration and save an I/O port, the microprocessor can wait for the total time of t_{BUSY} (max) + t_{RDY2WS} + t_{W2SB} before sending the next data byte. In this set-up, nRS should be tied high and RDYnBSY does not need to be connected to the microprocessor. The t_{BUSY}, t_{RDY2WS}, and t_{W2SB} timing specifications are listed in Table 7–14 on page 7–75.

Next, the microprocessor checks nSTATUS and CONF_DONE. If nSTATUS is not low and CONF_DONE is not high, the microprocessor sends the next data byte. However, if nSTATUS is not low and all the configuration data has been received, the device is ready for initialization. The CONF_DONE pin will go high one byte early in parallel configuration (FPP and PPA) modes. The last byte is required for serial configuration (AS and PS) modes. A low-to-high transition on CONF_DONE indicates configuration is complete and initialization of the device can begin. The open-drain CONF_DONE pin is pulled high by an external 10-kΩ pull-up resistor.

In Stratix II devices, the initialization clock source is either the Stratix II internal oscillator (typically 10 MHz) or the optional CLKUSR pin. By default, the internal oscillator is the clock source for initialization. If the internal oscillator is used, the Stratix II device will provide itself with enough clock cycles for proper initialization. Therefore, if the internal oscillator is the initialization clock source, sending the entire configuration file to the device is sufficient to configure and initialize the device.

You also have the flexibility to synchronize initialization of multiple devices or to delay initialization by using the CLKUSR option. The **Enable user-supplied start-up clock** (**CLKUSR**) option can be turned on in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. Supplying a clock on CLKUSR does not affect the configuration process. After CONF_DONE goes high, CLKUSR will be enabled after the time specifiedas t_{CD2CU}. After this time period elapses, the Stratix II devices require 299 clock cycles to initialize properly and enter user mode. Stratix II devices support a CLKUSR f_{MAX} of 100 MHz.

An optional INIT_DONE pin is available, which signals the end of initialization and the start of user-mode with a low-to-high transition. This **Enable INIT_DONE Output** option is available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box. If the INIT_DONE pin is used it will be high due to an external 10-kΩ pull-up resistor 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. When initialization is complete, the INIT_DONE pin will be released and pulled high. The microprocessor must be able to detect this low-to-high transition which signals the device has entered user mode. When initialization is complete, the device enters user mode. In user-mode, the user I/O pins will no longer have weak pull-up resistors and will function as assigned in your design.

To ensure DATA[7..0] is not left floating at the end of configuration, the microprocessor must drive them either high or low, whichever is convenient on your board. After configuration, the nCS, CS, nRS, nWS, RDYnBSY, and DATA[7..0] pins can be used as user I/O pins. When choosing the PPA scheme in the Quartus II software as a default, these I/O pins are tri-stated in user mode and should be driven by the microprocessor. To change this default option in the Quartus II software, select the **Dual-Purpose Pins** tab of the **Device & Pin Options** dialog box.

If an error occurs during configuration, the device drives its nSTATUS pin low, resetting itself internally. The low signal on the nSTATUS pin also alerts the microprocessor that there is an error. If the **Auto-restart configuration after error** option-available in the Quartus II software from the **General** tab of the **Device & Pin Options** dialog box-is turned on, the device releases nSTATUS after a reset time-out period (maximum of 40 µs). After nSTATUS is released and pulled high by a pull-up resistor, the microprocessor can try to reconfigure the target device without needing to pulse nCONFIG low. If this option is turned off, the microprocessor must generate a low-to-high transition (with a low pulse of at least 40 µs) on nCONFIG to restart the configuration process.

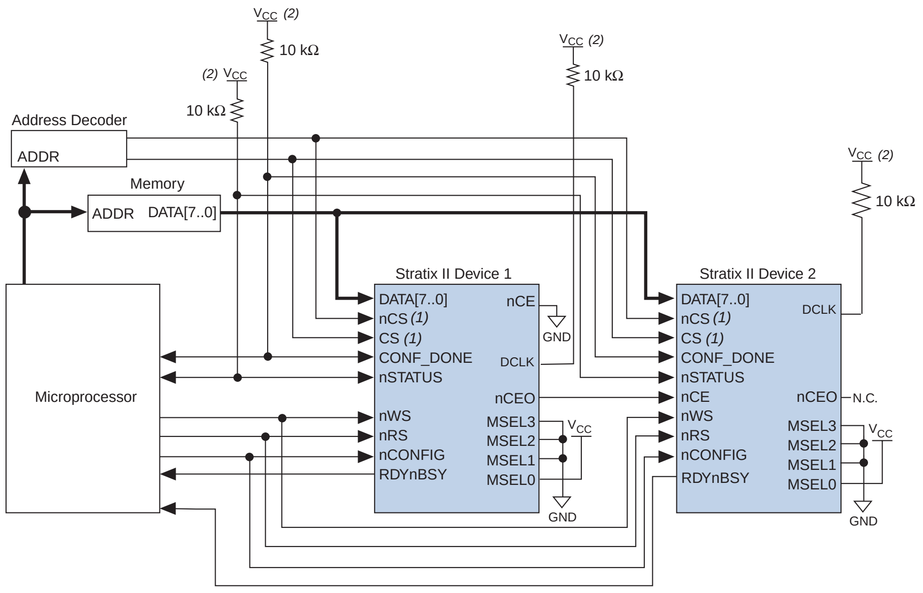
The microprocessor can also monitor the CONF_DONE and INIT_DONE pins to ensure successful configuration. To detect errors and determine when programming completes, monitor the CONF_DONE pin with the microprocessor. If the microprocessor sends all configuration data but CONF_DONE or INIT_DONE has not gone high, the microprocessor must reconfigure the target device.

☞ If you are using the optional CLKUSR pin and nCONFIG is pulled low to restart configuration during device initialization, ensure CLKUSR continues toggling during the time nSTATUS is low (maximum of 40 µs).

When the device is in user-mode, a reconfiguration can be initiated by transitioning the nCONFIG pin low-to-high. The nCONFIG pin should go low for at least 40 µs. When nCONFIG is pulled low, the device also pulls nSTATUS and CONF_DONE low and all I/O pins are tri-stated. Once nCONFIG returns to a logic high level and nSTATUS is released by the device, reconfiguration begins.

Figure 7–31 shows how to configure multiple Stratix II devices using a microprocessor. This circuit is similar to the PPA configuration circuit for a single device, except the devices are cascaded for multi-device configuration.

Figure 7–31. Multi-Device PPA Configuration Using a Microprocessor



Notes to Figure 7–31:

(1) If not used, the `CS` pin can be connected to V_{CC} directly. If not used, the `nCS` pin can be connected to GND directly.

(2) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

In multi-device PPA configuration 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. The second device in the chain begins configuration within one clock cycle. Therefore, the transfer of data destinations is transparent to the microprocessor.

Each device's `RDYnBSY` pin can have a separate input to the microprocessor. Alternatively, if the microprocessor is pin limited, all the `RDYnBSY` pins can feed an AND gate and the output of the AND gate can feed the microprocessor. For example, if you have two devices in a PPA configuration chain, the second device's `RDYnBSY` pin will be high during the time that the first device is being configured. When the first device has been successfully configured, it will drive `nCEO` low to activate the next device in the chain and drive its `RDYnBSY` pin high. Therefore, since

RDYnBSY signal is driven high before configuration and after configuration before entering user-mode, the device being configured will govern the output of the AND gate.

The nRS signal can be used in multi-device PPA chain since the Stratix II device will tri-state its DATA[7..0] pins before configuration and after configuration before entering user-mode to avoid contention. Therefore, only the device that is currently being configured will respond to the nRS strobe by asserting DATA7.

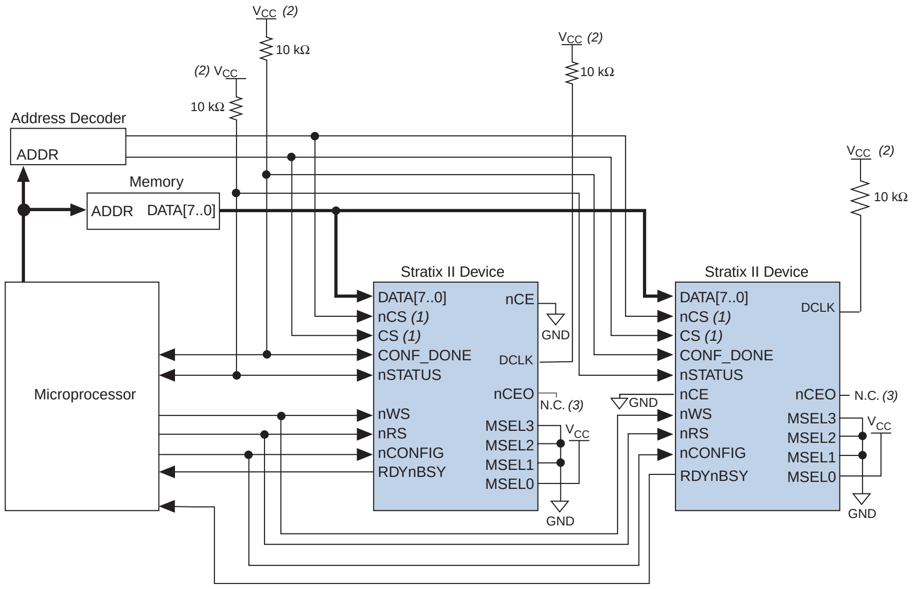
All other configuration pins (nCONFIG, nSTATUS, DATA[7..0], nCS, CS, nWS, nRS and CONF_DONE) are connected to every device in the chain. It is not necessary to tie nCS and CS together for every device in the chain, as each device's nCS and CS input can be driven by a separate source. Configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DATA lines are buffered for every fourth device. Because all device CONF_DONE pins are tied together, all devices initialize and enter user mode at the same time.

Since all nSTATUS and CONF_DONE pins are tied together, if any device detects an error, configuration stops for the entire chain and the entire chain must be reconfigured. For example, if the first device flags an error on nSTATUS, it resets the chain by pulling its nSTATUS pin low. This behavior is similar to a single device detecting an error.

If the **Auto-restart configuration after error** option is turned on, the devices release their nSTATUS pins after a reset time-out period (maximum of 40 µs). After all nSTATUS pins are released and pulled high, the microprocessor can try to reconfigure the chain without needing to pulse nCONFIG low. If this option is turned off, the microprocessor must generate a low-to-high transition (with a low pulse of at least 40 µs) on nCONFIG to restart the configuration process.

In your system, you may have multiple devices that contain the same configuration data. To support this configuration scheme, all device nCE inputs are tied to GND, while nCEO pins are left floating. All other configuration pins (nCONFIG, nSTATUS, DATA[7..0], nCS, CS, nWS, nRS and CONF_DONE) are connected to every device in the chain. Configuration signals may require buffering to ensure signal integrity and prevent clock skew problems. Ensure that the DATA lines are buffered for every fourth device. Devices must be the same density and package. All devices will start and complete configuration at the same time. Figure 7–32 shows multi-device PPA configuration when both devices are receiving the same configuration data.

Figure 7–32. Multiple-Device PPA Configuration Using a Microprocessor When Both devices Receive the Same Data



Notes to Figure 7–32:

(1) If not used, the `CS` pin can be connected to V_{CC} directly. If not used, the `nCS` pin can be connected to GND directly.

(2) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device and the external host.

(3) The `nCEO` pins of both devices are left unconnected when configuring the same configuration data into multiple devices.

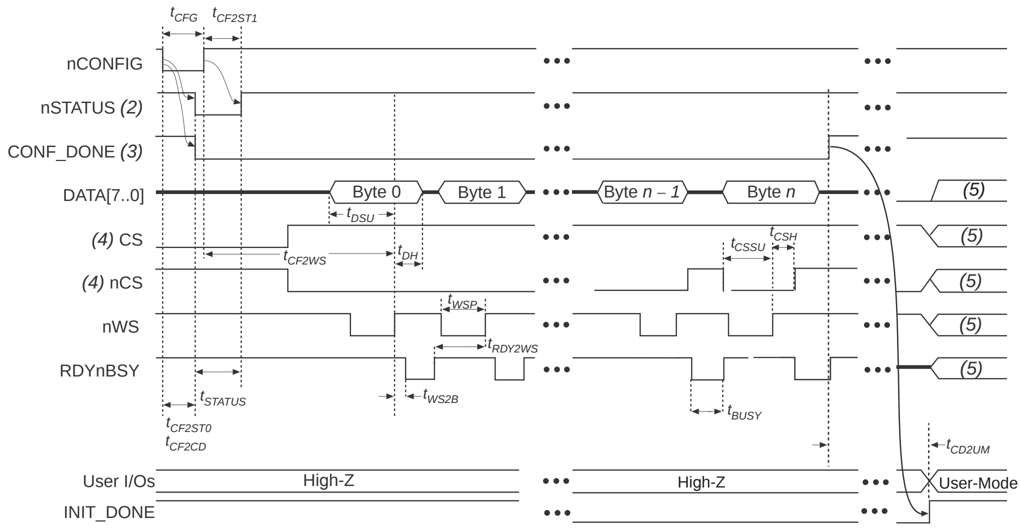
You can use a single configuration chain to configure Stratix II devices with other Altera devices that support PPA configuration, such as Stratix, Mercury™, APEX™ 20K, ACEX® 1K, and FLEX® 10KE devices. To ensure that all devices in the chain complete configuration at the same time or that an error flagged by one device initiates reconfiguration in all devices, all of the device `CONF_DONE` and `nSTATUS` pins must be tied together.

For more information on configuring multiple Altera devices in the same configuration chain, see the *Configuring Mixed Altera Device Chains* chapter in the *Configuration Handbook*.

PPA Configuration Timing

Figure 7–33 shows the timing waveform for the PPA configuration scheme using a microprocessor.

Figure 7–33. Stratix II PPA Configuration Timing Waveform Using nWS *Note (1)*

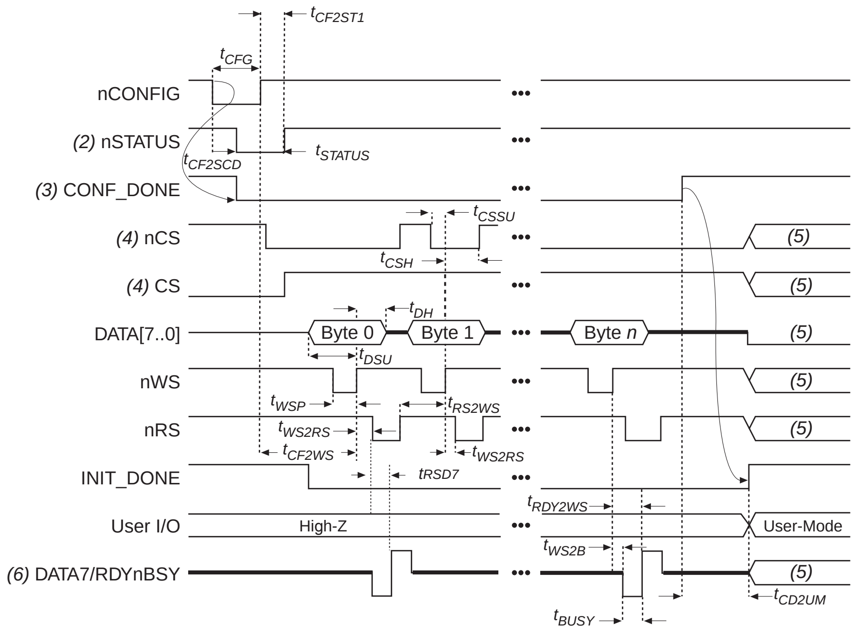


Notes to Figure 7–33:

(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) The user can toggle `nCS` or `CS` during configuration if the design meets the specification for t_{CSSU}, t_{WSP}, and t_{CSH}.
(5) `DATA[7..0]`, `CS`, `nCS`, `nWS`, `nRS`, and `RDYnBSY` are available as user I/O pins after configuration and the state of theses pins depends on the dual-purpose pin settings.

Figure 7–34 shows the timing waveform for the PPA configuration scheme when using a strobed `nRS` and `nWS` signal.

Figure 7–34. Stratix II PPA Configuration Timing Waveform Using nRS & nWS Note (1)



Notes to Figure 7–34:

(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) The user can toggle `nCS` or `CS` during configuration if the design meets the specification for t_{CSSU}, t_{WSP}, and t_{CSH}.

(5) `DATA[7..0]`, `CS`, `nCS`, `nWS`, `nRS`, and `RDYnBSY` are available as user I/O pins after configuration and the state of theses pins depends on the dual-purpose pin settings.

(6) `DATA7` is a bidirectional pin. It is an input for configuration data input, but it is an output to show the status of `RDYnBSY`.

Table 7–14 defines the timing parameters for Stratix II devices for PPA configuration.

Table 7–14. PPA Timing Parameters for Stratix II Devices (Part 1 of 2) Note (1)

Symbol	Parameter	Min	Max	Units
t_{POR}	POR delay	12	100	ms
t_{CF2CD}	`nCONFIG` low to `CONF_DONE` low		800	ns
t_{CF2ST0}	`nCONFIG` low to `nSTATUS` low		800	ns
t_{CFG}	`nCONFIG` low pulse width	2		µs

Table 7–14. PPA Timing Parameters for Stratix II Devices (Part 2 of 2) Note (1)

Symbol	Parameter	Min	Max	Units
t_{STATUS}	`nSTATUS` low pulse width	10	100 *(2)*	µs
t_{CF2ST1}	`nCONFIG` high to `nSTATUS` high		100 *(2)*	µs
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_{CF2WS}	`nCONFIG` high to first rising edge on `nWS`	100		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_{WSP}	`nWS` low pulse width	15		ns
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_R	Input rise time		40	ns
t_F	Input fall time		40	ns
t_{CD2UM}	`CONF_DONE` high to user mode *(3)*	20	40	µs
t_{CD2CU}	`CONF_DONE` high to `CLKUSR` enabled	40		ns
t_{CD2UMC}	`CONF_DONE` high to user mode with CLKUSR option on	t_{CD2CU} + (299 × CLKUSR period)		

Notes to Table 7–14:

(1) This information is preliminary.

(2) This value is obtainable if users do not delay configuration by extending the `nCONFIG` or `nSTATUS` low pulse width.

(3) The minimum and maximum numbers apply only if the internal oscillator is chosen as the clock source for starting up the device.

Device configuration options and how to create configuration files are discussed further in the *Software Settings* chapter in the *Configuration Handbook*.

JTAG 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 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. The JTAG circuitry can also be used to shift configuration data into the device. The Quartus II software automatically generates SOFs that can be used for JTAG configuration with a download cable in the Quartus II software programmer.

For more information on JTAG boundary-scan testing, see the following documents:

- Chapter 9, IEEE 1149.1 (JTAG) Boundary-Scan Testing for Stratix II Devices in the *Stratix II Device Handbook, Volume II*
- *Jam Programming & Testing Language Specification*

Stratix II devices are designed such that JTAG instructions have precedence over any device configuration modes. Therefore, JTAG configuration can take place without waiting for other configuration modes to complete. For example, if you attempt JTAG configuration of Stratix II devices during PS configuration, PS configuration will be terminated and JTAG configuration will begin.

You cannot use the Stratix II decompression feature or the design security feature if you are configuring your Stratix II device when using JTAG-based configuration.

A device operating in JTAG mode uses four required pins, `TDI`, `TDO`, `TMS`, and `TCK`, and one optional pin, `TRST`. The `TCK` pin has an internal weak pull-down resistor, while the `TDI`, `TMS`, and `TRST` pins have weak internal pull-up resistors (typically 25 kΩ). The `TDO` output pin is powered by V_{CCIO} in I/O bank 4. All of the JTAG input pins are powered by the 3.3-V V_{CCPD} pin. All user I/O pins are tri-stated during JTAG configuration. Table 7–15 explains each JTAG pin's function.

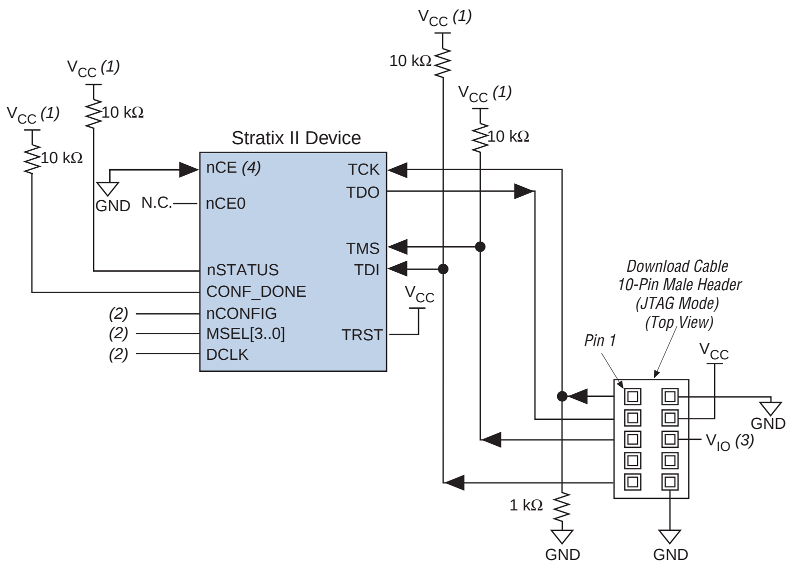
The TDO output is powered by the V_{CCIO} power supply of I/O bank 4. For recommendations on how to connect a JTAG chain with multiple voltages across the devices in the chain, refer to the *IEEE 1149.1 (JTAG) Boundary Scan Testing in Stratix II Devices* chapter in Volume 2 of the *Stratix II Handbook*.

Table 7–15. Dedicated JTAG Pins

Pin Name	Pin Type	Description
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. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V_{CC}.
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. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by leaving this pin unconnected.
TMS	Test mode select	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}.
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. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to GND.
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. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to GND.

During JTAG configuration, data can be downloaded to the device on the PCB through the USB Blaster, MasterBlaster, ByteBlaster II, or ByteBlasterMV download cable. Configuring devices through a cable is similar to programming devices in-system, except the TRST pin should be connected to V_{CC}. This ensures that the TAP controller is not reset. Figure 7–35 shows JTAG configuration of a single Stratix II device.

Figure 7–35. JTAG Configuration of a Single Device Using a Download Cable



Notes to Figure 7–35:

(1) The pull-up resistor should be connected to the same supply voltage as the USB Blaster, MasterBlaster (V_{IO} pin), ByteBlaster II, or ByteBlasterMV cable.

(2) The nCONFIG, MSEL[3..0] pins should be connected to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL[3..0] to ground. Pull DCLK either high or low, whichever is convenient on your board.

(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}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB Blaster and ByteBlaster II cables, this pin is connected to nCE when it is used for active serial programming, otherwise it is a no connect.

(4) nCE must be connected to GND or driven low for successful JTAG configuration.

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.

The Quartus II software verifies successful JTAG configuration upon completion. At the end of configuration, the software checks the state of CONF_DONE through the JTAG port. When Quartus II generates a (**.jam**) file for a multi-device chain, it contains instructions so that all the devices in the chain will be initialized at the same time. If CONF_DONE is not high, the Quartus II software indicates that configuration has failed. If

CONF_DONE is high, the software indicates that configuration was successful. After the configuration bit stream is transmitted serially via the JTAG TDI port, the TCK port is clocked an additional 299 cycles to perform device initialization.

Stratix II devices have dedicated JTAG pins that always function as JTAG pins. Not only can you perform JTAG testing on Stratix II devices before and after, but also during configuration. While other device families do not support JTAG testing during configuration, Stratix II devices support the bypass, idcode, and sample instructions during configuration without interrupting configuration. All other JTAG instructions may only be issued by first interrupting configuration and reprogramming I/O pins using the CONFIG_IO instruction.

The CONFIG_IO instruction allows I/O buffers to be configured via the JTAG port and when issued, interrupts configuration. This instruction allows you to perform board-level testing prior to configuring the Stratix II device or waiting for a configuration device to complete configuration. Once configuration has been interrupted and JTAG testing is complete, the part must be reconfigured via JTAG (PULSE_CONFIG instruction) or by pulsing nCONFIG low.

The chip-wide reset (DEV_CLRn) and chip-wide output enable (DEV_OE) pins on Stratix II 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 II devices, consider the dedicated configuration pins. Table 7–16 shows how these pins should be connected during JTAG configuration.

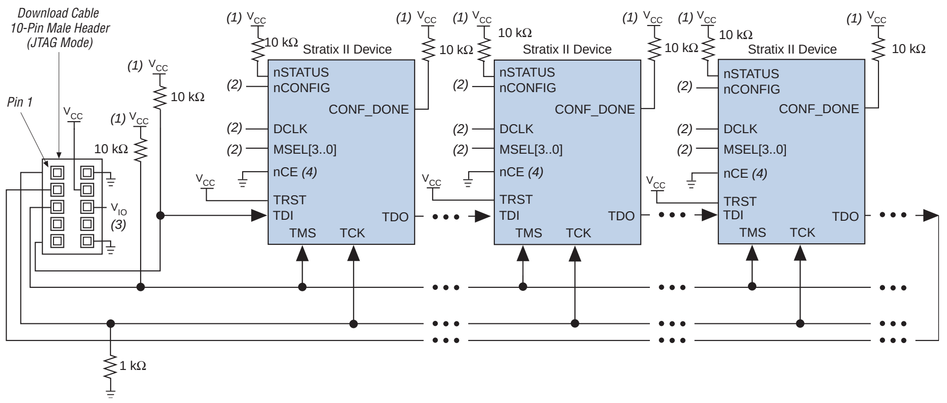
When programming a JTAG device chain, one JTAG-compatible header is connected to several devices. The number of devices in the JTAG chain is limited only by the drive capability of the download cable. When four or more devices are connected in a JTAG chain, Altera recommends buffering the TCK, TDI, and TMS pins with an on-board buffer.

JTAG-chain device programming is ideal when the system contains multiple devices, or when testing your system using JTAG BST circuitry. Figure 7–36 shows multi-device JTAG configuration.

Table 7–16. Dedicated Configuration Pin Connections During JTAG Configuration

Signal	Description
nCE	On all Stratix II 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 FPP, AS, PS, 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 II devices in the chain, `nCEO` can be left floating or connected to the `nCE` of the next device.
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, tie these pins to ground.
nCONFIG	Driven high by connecting to V_{CC}, pulling up via a resistor, or driven high 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.
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.

Figure 7–36. JTAG Configuration of Multiple Devices Using a Download Cable



Notes to Figure 7–36:

(1) The pull-up resistor should be connected to the same supply voltage as the USB Blaster, MasterBlaster (V_{IO} pin), ByteBlaster II or ByteBlasterMV cable.

(2) The `nCONFIG, MSEL[3..0]` pins should be connected to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect `nCONFIG` to V_{CC}, and `MSEL[3..0]` to ground. Pull `DCLK` either high or low, whichever is convenient on your board.

(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}. Refer to the *MasterBlaster Serial/USB Communications Cable Data Sheet* for this value. In the ByteBlasterMV cable, this pin is a no connect. In the USB Blaster and ByteBlaster II cables, this pin is connected to `nCE` when it is used for active serial programming, otherwise it is a no connect.

(4) `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 FPP, AS, PS 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, 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 will drive `nCE` of the next device low when it has successfully been JTAG configured.

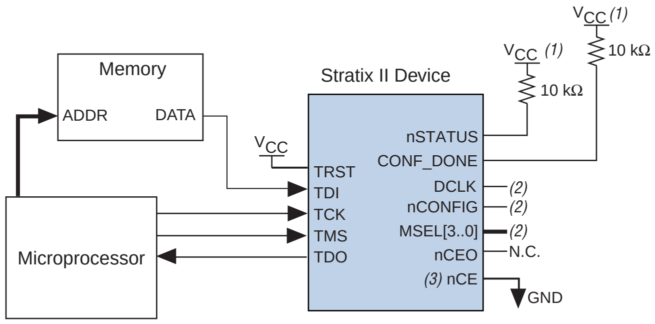
Other Altera devices that have JTAG support can be placed in the same JTAG chain for device programming and configuration.

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.

For more information on configuring multiple Altera devices in the same configuration chain, see the *Configuring Mixed Altera device Chains* chapter in the *Configuration Handbook*.

Figure 7–37 shows JTAG configuration of a Stratix II device with a microprocessor.

Figure 7–37. JTAG Configuration of a Single Device Using a Microprocessor



Notes to Figure 7–37:
(1) The pull-up resistor should be connected to a supply that provides an acceptable input signal for all devices in the chain. V_{CC} should be high enough to meet the V_{IH} specification of the I/O on the device.
(2) The `nCONFIG`, `MSEL[3..0]` pins should be connected to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect `nCONFIG` to V_{CC}, and `MSEL[3..0]` to ground. Pull `DCLK` either high or low, whichever is convenient on your board.
(3) `nCE` must be connected to GND or driven low for successful JTAG configuration.

Jam STAPL

Jam 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.

The Jam Player provides an interface for manipulating the IEEE Std. 1149.1 JTAG TAP state machine.

For more information on JTAG and Jam STAPL in embedded environments, see *AN 122: Using Jam STAPL for ISP & ICR via an Embedded Processor*. To download the jam player, visit the Altera web site at **www.altera.com**

Device Configuration Pins

The following tables describe the connections and functionality of all the configuration related pins on the Stratix II device. Table 7–17 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 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 1 of 9)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
V_{CCPD}	N/A	All	Power	Dedicated power pin. This pin is used to power the I/O pre-drivers and a 3.3-V/2.5-V buffer available on the configuration input pins and JTAG pins. V_{CCPD} applies to all the JTAG input pins (`TCK`, `TMS`, `TDI`, and `TRST`) and the configuration pins; `nCONFIG`, `DCLK` (when used as an input), `nIO_Pullup`, `DATA[7..0]`, `RUnLU`, `nCE`, `nWS`, `nRS`, `CS`, `nCS` and `CLKUSR`. This pin must be connected to 3.3-V. V_{CCPD} must ramp-up from 0-V to 3.3-V within 100 ms. If V_{CCPD} is not ramped up within this specified time, your Stratix II device will not configure successfully. If your system does not allow for a V_{CCPD} ramp-up time of 100 ms or less, you must hold `nCONFIG` low until all power supplies are stable.
`VCCSEL`	N/A	All	Input	Dedicated input that selects which input buffer is used on the configuration input pins; `nCONFIG`, `DCLK` (when used as an input), `RUnLU`, `nCE`, `nWS`, `nRS`, `CS`, `nCS`, and `CLKUSR`. The 3.3-V/2.5-V input buffer is powered by V_{CCPD}, while the 1.8-V/1.5-V input buffer is powered by V_{CCIO}. The `VCCSEL` input buffer has an internal 5-kΩ pull-down resistor that is always active. The `VCCSEL` input buffer is powered by V_{CCINT} and must be hardwired to V_{CCPD} or ground. 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. For more information, see "VCCSEL Pin" section.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 2 of 9)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`PORSEL`	N/A	All	Input	Dedicated input which selects between a POR time of 12 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 12 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 5-kΩ pull-down resistor that is always active. The `PORSEL` pin should be tied directly to V_{CCPD} or GND.
`nIO_PULLUP`	N/A	All	Input	Dedicated input that chooses whether the internal pull-up resistors on the user I/O pins and dual-purpose I/O pins (`nCSO`, `nASDO`, `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-up resistors, while a logic low turns them on. The `nIO_PULLUP` pin has an internal 5-kΩ pull-down resistor that is always active.
`MSEL[3..0]`	N/A	All	Input	The `nIO-PULLUP` input buffer is powered by V_{CCPD} and has an internal 5-kΩ pull-down resistor that is always active. The `nIO-PULLUP` can be tied directly to V_{CCPD} or use a 1-kΩ pull-up resistor or tied directly to GND. The MSEL[3..0] pins have internal 5-kΩ pull-down resistors that are always active. 4-bit configuration input that sets the Stratix II device configuration scheme. See Table 7–1 for the appropriate connections. These pins must be hard-wired to V_{CCPD} or GND.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 3 of 9)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`nCONFIG`	N/A	All	Input	Configuration control input. Pulling this pin low during user-mode will cause the device to lose its configuration data, enter a reset state, tri-state all I/O pins. Returning this pin to a logic high level will initiate 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.
`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 will cause the configuration device to attempt to configure the device, but since the device ignores transitions on `nSTATUS` in user-mode, the device 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.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 4 of 9)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`CONF_DONE`	N/A	All	Bidirectional open-drain	Status output. The target device 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. 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.
`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 device.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 5 of 9)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`nCEO`	N/A	All	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 `nCEO` pin is powered by V_{CCIO} in I/O bank 7. For recommendations on how to connect `nCEO` in a chain with multiple voltages across the devices in the chain, refer to the *MultiVolt I/O Interface* section in the *Stratix II Architecture* chapter in Volume 1 of the *Stratix II Handbook*.
`ASDO`	N/A in AS mode I/O in non-AS mode	AS	Output	Control signal from the Stratix II device to the serial configuration device in AS mode used to read out configuration data. In AS mode, `ASDO` has an internal pull-up resistor that is always active.
`nCSO`	N/A in AS mode I/O in non-AS mode	AS	Output	Output control signal from the Stratix II device to the serial configuration device in AS mode that enables the configuration device. In AS mode, `nCSO` has an internal pull-up resistor that is always active.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 6 of 9)

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`DCLK`	N/A	Synchronous configuration schemes (PS, FPP, AS)	Input (PS, FPP) Output (AS)	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 device on the rising edge of `DCLK`. In AS mode, `DCLK` is an output from the Stratix II device that provides timing for the configuration interface. In AS mode, `DCLK` has an internal pull-up resistor (typically 25 kΩ) that is always active. 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` will be 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.
`DATA0`	I/O	PS, FPP, PPA, AS	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 this pin resides in. In AS mode, `DATA0` has an internal pull-up resistor that is always active. After configuration, `DATA0` is available as a user I/O pin 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 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 7 of 9)

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 this pin are dependent on the V_{CCIO} of the I/O bank that this pin resides in. In serial configuration schemes, they function as user I/O pins during configuration, which means they are tri-stated. After PPA or FPP configuration, `DATA[7..1]` are available as user I/O pins 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_{IH} and V_{IL} levels for this pin are dependent on the V_{CCIO} of the I/O bank that this pin resides in. In serial configuration schemes, it functions as a user I/O pin 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 pin during configuration, which means it is tri-stated. After PPA configuration, `nWS` is available as a user I/O pins and the state of this pin depends on the **Dual-Purpose Pin** settings.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 8 of 9)

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 pin 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 will drive 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 pin during configuration, which means it is tri-stated. After PPA configuration, `RDYnBSY` is available as a user I/O pin and the state of this pin depends on the **Dual-Purpose Pin** settings.

Table 7–17. Dedicated Configuration Pins on the Stratix II Device (Part 9 of 9)

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 ground while `CS` is toggled to control configuration. In non-PPA schemes, it functions as a user I/O pin during configuration, which means it is tri-stated. After PPA configuration, `nCS` and `CS` are available as user I/O pins and the state of these pins depends on the **Dual-Purpose Pin** settings.
`RUnLU`	N/A if using Remote System Upgrade I/O if not	Remote System Upgrade 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 pin is available as general-purpose user I/O pin. When using remote system upgrade in AS more, the `RUnLU` pin is available as a general-purpose I/O pin.
`PGM[2..0]`	N/A if using Remote System Upgrade I/O if not using	Remote System Upgrade 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 system upgrade mode. When not using remote update or local update configuration modes, these pins are available as general-purpose user I/O pins. When using remote system upgrade in AS more, the `PGM[]` pins are available as general-purpose I/O pins.

Table 7–18 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-up resistors.

Table 7–18. 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 resistor. 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. When initialization is complete, the `INIT_DONE` pin will be released and pulled high and the device 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/O pins are tri-stated; when this pin is driven high, all I/O pins 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 7–19 describes the dedicated JTAG pins. JTAG pins must be kept stable before and during configuration to prevent accidental loading of JTAG instructions. The `TDI`, `TMS`, and `TRST` have weak internal pull-up resistors (typically 25 kΩ) while `TCK` has a weak internal pull-down

resistor. If you plan to use the SignalTap® embedded logic array analyzer, you need to connect the JTAG pins of the Stratix II device to a JTAG header on your board.

Table 7–19. 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`. The `TDI` pin is powered by the 3.3-V V_{CCPD} supply. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V_{CC}.
`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. The `TDO` pin is powered by V_{CCIO} in I/O bank 4. For recommendations on connecting a JTAG chain with multiple voltages across the devices in the chain, refer to the *I/O Voltage Support in JTAG Chain* section in the *IEEE 1149.1 (JTAG) Boundary Scan Testing in Stratix II Devices* chapter in Volume 2 of the *Stratix II Handbook*. 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`. The `TMS` pin is powered by the 3.3-V V_{CCPD} supply. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting this pin to V_{CC}.
`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. The `TCK` pin is powered by the 3.3-V V_{CCPD} supply. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting `TCK` to GND.
`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. The `TRST` pin is powered by the 3.3-V V_{CCPD} supply. If the JTAG interface is not required on the board, the JTAG circuitry can be disabled by connecting the `TRST` pin to GND.

Conclusion

Stratix II devices can be configured in a number of different schemes to fit your system's need. In addition, configuration bitstream encryption, configuration data decompression, and remote system upgrade support supplement the Stratix II configuration solution.

8. Remote System Upgrades with Stratix II Devices

SII52008-2.0

Introduction

System designers today face difficult challenges such as shortened design cycles, evolving standards, and system deployments in remote locations. Stratix® II FPGAs help overcome these challenges with their inherent re-programmability and dedicated circuitry to perform remote system upgrades. Remote system upgrades help deliver feature enhancements and bug fixes without costly recalls, reduce time-to-market, and extend product life.

Stratix II FPGAs feature dedicated remote system upgrade circuitry. Soft logic (either the Nios® embedded processor or user logic) implemented in a Stratix II device can download a new configuration image from a remote location, store it in configuration memory, and direct the dedicated remote system upgrade circuitry to initiate a reconfiguration cycle. The dedicated circuitry performs error detection during and after the configuration process, recovers from any error condition by reverting back to a safe configuration image, and provides error status information. This dedicated remote system upgrade circuitry is unique to Stratix and Stratix II FPGAs and helps to avoid system downtime.

Remote system upgrade is supported in all Stratix II configuration schemes: fast passive parallel (FPP), active serial (AS), passive serial (PS), and passive parallel asynchronous (PPA). Remote system upgrade can also be implemented in conjunction with advanced Stratix II features such as real-time decompression of configuration data and design security using the advanced encryption standard (AES) for secure and efficient field upgrades.

This chapter describes the functionality and implementation of the dedicated remote system upgrade circuitry. It also defines several concepts related to remote system upgrade, including factory configuration, application configuration, remote update mode, local update mode, the user watchdog timer, and page mode operation. Additionally, this chapter provides design guidelines for implementing remote system upgrade with the various supported configuration schemes.

Functional Description

The dedicated remote system upgrade circuitry in Stratix II FPGAs manages remote configuration and provides error detection, recovery, and status information. User logic or a Nios processor implemented in the FPGA logic array provides access to the remote configuration data source and an interface to the system's configuration memory.

Stratix II FPGA's remote system upgrade process involves the following steps:

1. A Nios processor (or user logic) implemented in the FPGA logic array receives new configuration data from a remote location. The connection to the remote source is a communication protocol such as the transmission control protocol/Internet protocol (TCP/IP), peripheral component interconnect (PCI), user datagram protocol (UDP), universal asynchronous receiver/transmitter (UART), or a proprietary interface.

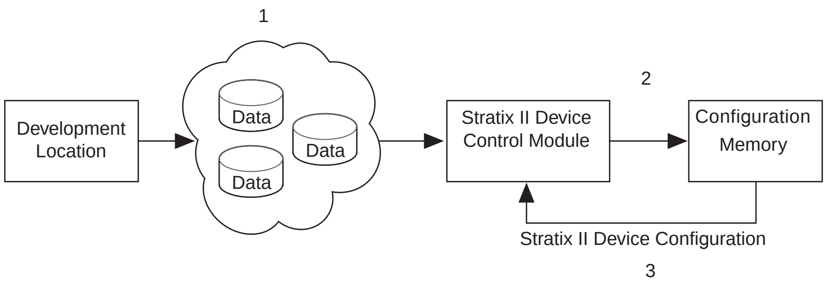
2. The Nios processor (or user logic) stores this new configuration data in non-volatile configuration memory. The non-volatile configuration memory can be any standard flash memory used in conjunction with an intelligent host (for example, a MAX® device or microprocessor), the serial configuration device, or the enhanced configuration device.

3. The Nios processor (or user logic) initiates a reconfiguration cycle with the new or updated configuration data.

4. The dedicated remote system upgrade circuitry detects and recovers from any error(s) that might occur during or after the reconfiguration cycle, and provides error status information to the user design.

Figure 8–1 shows the steps required for performing remote configuration updates. (The numbers in the figure below coincide with the steps above.)

Figure 8–1. Functional Diagram of Stratix II Remote System Upgrade


1
Development Location
Data
Data
Data
2
Stratix II Device Control Module
Configuration Memory
Stratix II Device Configuration
3

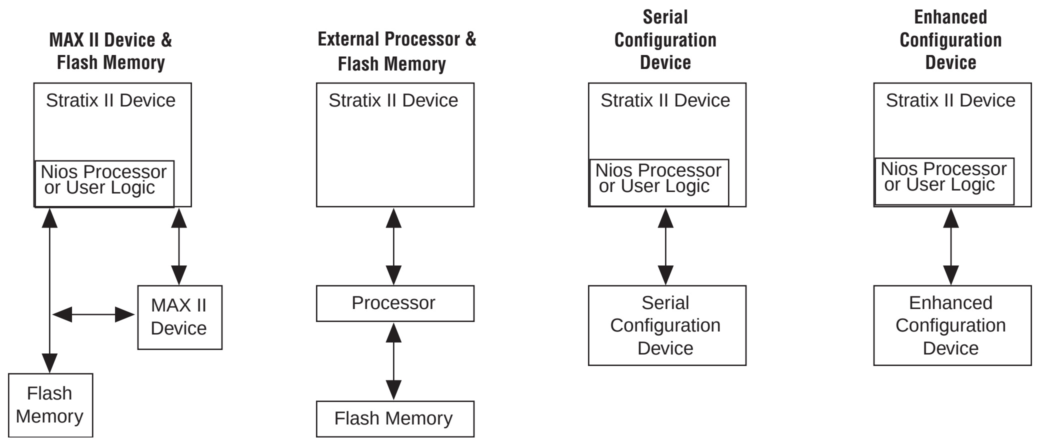
Stratix II FPGAs support remote system upgrade in the FPP, AS, PS, and PPA configuration schemes.

- Serial configuration devices use the AS scheme to configure Stratix II FPGAs.
- A MAX II device (or microprocessor and flash configuration schemes) uses FPP, PS, or PPA schemes to configure Stratix II FPGAs.
- Enhanced configuration devices use the FPP or PS configuration schemes to configure Stratix II FPGAs.

☞ The JTAG-based configuration scheme does not support remote system upgrade.

Figure 8–2 shows the block diagrams for implementing remote system upgrade with the various Stratix II configuration schemes.

Figure 8–2. Remote System Upgrade Block Diagrams for Various Stratix II Configuration Schemes


MAX II Device & Flash Memory
Stratix II Device
Nios Processor or User Logic
MAX II Device
Flash Memory
External Processor & Flash Memory
Stratix II Device
Processor
Flash Memory
Serial Configuration Device
Stratix II Device
Nios Processor or User Logic
Serial Configuration Device
Enhanced Configuration Device
Stratix II Device
Nios Processor or User Logic
Enhanced Configuration Device

You must set the mode select pins (`MSEL[3..0]`) and the `RUnLU` pin to select the configuration scheme and remote system upgrade mode best suited for your system. Table 8–1 lists the pin settings for Stratix II FPGAs. Standard configuration mode refers to normal FPGA configuration mode with no support for remote system upgrades, and the remote system upgrade circuitry is disabled. The following sections describe the local update and remote update remote system upgrade modes.

For more information on standard configuration schemes supported in Stratix II FPGAs, see the *Configuring Stratix II FPGAs* chapter of the *Stratix II Handbook*.

Table 8–1. Stratix II Remote System Upgrade Modes (Part 1 of 2)

Configuration Scheme	MSEL[3..0]	RUnLU	Remote System Upgrade Mode
FPP	0000	-	Standard
	0100 *(1)*	0	Local update
	0100 *(1)*	1	Remote update
FPP with decompression and/or design security feature enabled *(2)*	1011	-	Standard
	1100 *(1)*	0	Local update
	1100 *(1)*	1	Remote update
Fast AS (40 MHz) *(3)*	1000	-	Standard
	1001	N/A	Remote update

Table 8–1. Stratix II Remote System Upgrade Modes (Part 2 of 2)

Configuration Scheme	MSEL[3..0]	RUnLU	Remote System Upgrade Mode
AS (20 MHz) *(3)*	1101	-	Standard
	1110	N/A	Remote update
PS	0010	-	Standard
	0110 *(1)*	0	Local update
	0110 *(1)*	1	Remote update
PPA	0001	-	Standard
	0101 *(1)*	0	Local update
	0101 *(1)*	1	Remote update

Notes to Table 8–1:

(1) These schemes require that you drive the RUnLU pin to specify either remote update or local update mode. AS schemes do not use the RUnLU pin and only support the remote update mode.

(2) These modes are only supported when using a MAX II device or microprocessor and flash for configuration. In these modes, the host system must output a DCLK that is 4 x the data rate.

(3) The EPCS16 and EPCS64 serial configuration devices support a DCLK up to 40 MHz; other EPCS devices support a DCLK up to 20 MHz. See the *Serial Configuration Devices Data Sheet* for more information.

Configuration Image Types & Pages

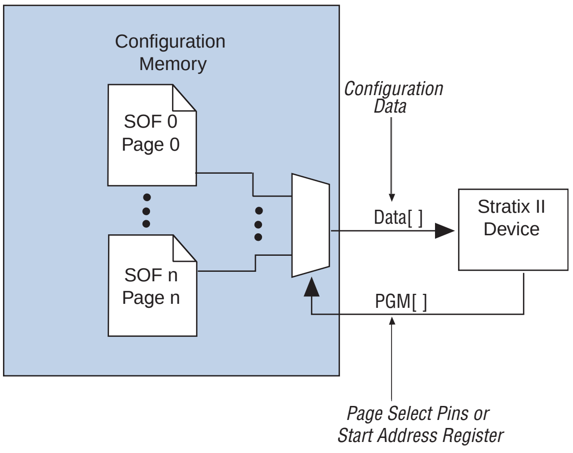
When using remote system upgrade, FPGA configuration bitstreams are classified as factory configuration images or application configuration images. An image, also referred to as a configuration, is a design loaded into the FPGA that performs certain user-defined functions. Each FPGA in your system requires one factory image and one or more application images. The factory image is a user-defined fall-back, or safe, configuration and is responsible for administering remote updates in conjunction with the dedicated circuitry. Application images implement user-defined functionality in the target FPGA.

A remote system update involves storing a new application configuration image or updating an existing one via the remote communication interface. After an application configuration image is stored or updated remotely, the user design in the FPGA initiates a reconfiguration cycle with the new image. Any errors during or after this cycle are detected by the dedicated remote system upgrade circuitry and cause the FPGA to automatically revert to the factory image. The factory image then performs error processing and recovery. While error processing functionality is limited to the factory configuration, both factory and application configurations can download and store remote updates and initiate system reconfiguration.

The Stratix II FPGA selects between the different configuration images stored in the system configuration memory using the page address pins or start address registers. A page is a section of the configuration memory space that contains one configuration image for each FPGA in the system. One page stores one system configuration, regardless of the number of FPGAs in the system.

Page address pins select the configuration image within an enhanced configuration device or flash memory (MAX II device or microprocessor setup). Page start address registers are used when Stratix II FPGAs are configured in AS mode with serial configuration devices. Figure 8–3 illustrates page mode operation in Stratix II FPGAs.

Figure 8–3. Page Mode Operation in Stratix II FPGAs



Stratix II devices drive out three page address pins, `PGM[2..0]`, to the MAX II device or microprocessor or enhanced configuration device. These page pins select between eight configuration pages. Page zero (`PGM[2..0] = 000`) must contain the factory configuration, and the other seven pages are application configurations. The `PGM[]` pins are pointers to the start address and length of each page, and the MAX II device, microprocessor, and enhanced configuration devices perform this translation.

☞ When implementing remote system upgrade with an intelligent-host-based configuration, your MAX II device or microprocessor should emulate the page mode feature supported by the enhanced configuration device, which translates PGM pointers to a memory address in the configuration memory. Your MAX II device or microprocessor must provide a similar translation feature.

For more information about the enhanced configuration device page mode feature, refer to the *Dynamic Configuration (Page Mode) Implementation* section of the *Using Altera Enhanced Configuration Devices* chapter in the *Configuration Handbook*.

When implementing remote system upgrade with AS configuration, a dedicated 7-bit page start address register inside the Stratix II FPGA determines the start addresses for configuration pages within the serial configuration device. The `PGM[6..0]` registers form bits `[22..16]` of the 24-bit start address while the other 17 bits are set to zero: `StAdd[23..0] = {1'b0, PGM[6..0], 16'b0}`. During AS configuration, the Stratix II FPGA uses this 24-bit page start address to obtain configuration data from the serial configuration devices.

Remote System Upgrade Modes

Remote system upgrade has two modes of operation: remote update mode and local update mode. The remote and local update modes allow you to determine the functionality of your system upon power up and offer different features. The `RUnLU` input pin selects between the remote update (logic high) and local update (logic low) modes.

Overview

In remote update mode, the Stratix II FPGA loads the factory configuration image upon power up. The user-defined factory configuration should determine which application configuration is to be loaded and trigger a reconfiguration cycle. Remote update mode allows up to eight configuration images (one factory plus seven application images) when used with the MAX II device or microprocessor and flash-based configuration or an enhanced configuration device.

When used with serial configuration devices, the remote update mode allows an application configuration to start at any flash sector boundary. This translates to a maximum of 128 pages in the EPCS64 and 32 pages in the EPCS16 device, where the minimum size of each page is 512 KBits. Additionally, the remote update mode features a user watchdog timer that can detect functional errors in an application configuration.

Local update mode is a simplified version of the remote update mode. In this mode, the Stratix II FPGA directly loads the application configuration, bypassing the factory configuration. This mode is useful if your system is required to boot into user mode with minimal startup time. It is also useful during system prototyping, as it allows you to verify functionality of the application configuration.

In local update mode, a maximum of two configuration images or pages is supported: one factory configuration, located at page address `PGM[2..0] = 000`, and one application configuration, located at page address `PGM[2..0] = 001`. Because the page address of the application configuration is fixed, the local update mode does not require the factory configuration image to determine which application is to be loaded. If any errors are encountered while loading the application configuration, the Stratix II FPGA reverts to the factory configuration. The user watchdog timer feature is not supported in this mode.

Also, local update mode does not support AS configuration with the serial configuration devices because these devices don't support a dynamic pointer to page 001 start address location.

Table 8–2 details the differences between remote and local update modes.

Table 8–2. Differences Between Remote & Local Update Modes (Part 1 of 2)

Features	Remote Update Mode	Local Update Mode
`RUnLU` input pin setting	1	0
Page selection upon power up	`PGM[2..0] = 000` (Factory)	`PGM[2..0] = 001` (Application)
Supported configurations	MAX II device or microprocessor-based configuration, serial configuration, and enhanced configuration devices (FPP, PS, AS, PPA)	MAX II device or microprocessor-based configuration and enhanced configuration devices (FPP, PS, PPA)
Number of pages supported	Eight pages for external host or controller based configuration; up to 128 pages (512 KBits/page) for serial configuration device	Two pages

Table 8–2. Differences Between Remote & Local Update Modes (Part 2 of 2)

Features	Remote Update Mode	Local Update Mode
User watchdog timer	Available	Disabled
Remote system upgrade control and status register	Read/write access allowed in factory configuration. Read access in application configuration	Only status register read access allowed in local update mode (factory and application configurations). Write access to control register is disabled

Remote Update Mode

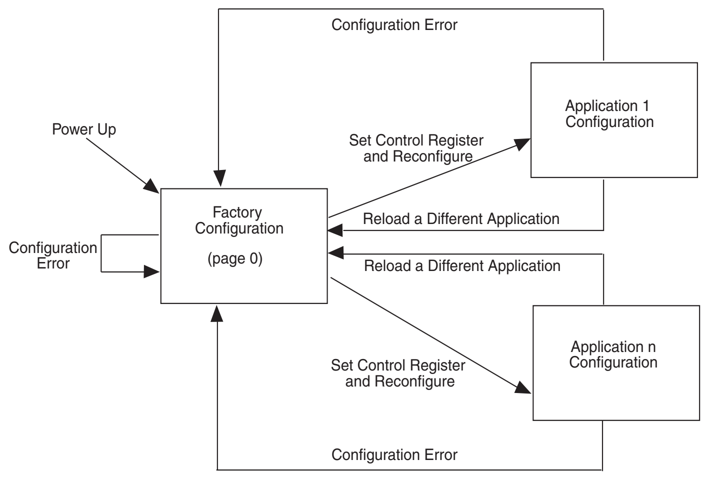
When the Stratix II device is first powered up in remote update mode, it loads the factory configuration located at page zero (page address pins `PGM[2..0] = "000"`; page registers `PGM[6..0] = "0000000"`). You should always store the factory configuration image for your system at page address zero. A factory configuration image is a bitstream for the FPGA(s) in your system that is programmed during production and is the fall-back image when errors occur. This image is stored in non-volatile memory and is never updated or modified using remote access. This corresponds to `PGM[2..0] = 000` of the enhanced configuration device or standard flash memory, and start address location 0x000000 in the serial configuration device.

The factory image is user designed and contains soft logic to:

- Process any errors based on status information from the dedicated remote system upgrade circuitry
- Communicate with the remote host and receive new application configurations, and store this new configuration data in the local non-volatile memory device
- Determine which application configuration is to be loaded into the FPGA
- Enable or disable the user watchdog timer and load its time-out value (optional)
- Instruct the dedicated remote system upgrade circuitry to initiate a reconfiguration cycle

Figure 8–4 shows the transitions between the factory and application configurations in remote update mode.

Figure 8–4. Transitions Between Configurations in Remote Update Mode


Configuration Error
Application 1 Configuration
Power Up
Set Control Register and Reconfigure
Factory Configuration (page 0)
Reload a Different Application
Configuration Error
Reload a Different Application
Application n Configuration
Set Control Register and Reconfigure
Configuration Error

After power up or a configuration error, the factory configuration logic should write the remote system upgrade control register to specify the page address of the application configuration to be loaded. The factory configuration should also specify whether or not to enable the user watchdog timer for the application configuration and, if enabled, specify the timer setting.

The user watchdog timer ensures that the application configuration is valid and functional. After confirming the system is healthy, the user-designed application configuration should reset the timer periodically during user-mode operation of an application configuration. This timer reset logic should be a user-designed hardware and/or software health monitoring signal that indicates error-free system operation. If the user application configuration detects a functional problem or if the system hangs, the timer is not reset in time and the dedicated circuitry updates the remote system upgrade status register, triggering the device to load the factory configuration. The user watchdog timer is automatically disabled for factory configurations.

☞ Only valid application configurations designed for remote update mode include the logic to reset the timer in user mode.

For more information about the user watchdog timer, see the "User Watchdog Timer" section on page 8–19.

If there is an error while loading the application configuration, the remote system upgrade status register is written by the Stratix II FPGA's dedicated remote system upgrade circuitry, specifying the cause of the reconfiguration. Actions that cause the remote system upgrade status register to be written are:

- `nSTATUS` driven low externally
- Internal `CRC` error
- User watchdog timer time out
- A configuration reset (logic array `nCONFIG` signal or external `nCONFIG` pin assertion)

The Stratix II device automatically loads the factory configuration located at page address zero. This user-designed factory configuration should read the remote system upgrade status register to determine the reason for reconfiguration. The factory configuration should then take appropriate error recovery steps and write to the remote system upgrade control register to determine the next application configuration to be loaded.

When the Stratix II device successfully loads the application configuration, it enters into user mode. In user mode, the soft logic (Nios processor or state machine and the remote communication interface) assists the Stratix II device in determining when a remote system update is arriving. When a remote system update arrives, the soft logic receives the incoming data, writes it to the configuration memory device, and triggers the device to load the factory configuration. The factory configuration reads the remote system upgrade status register, determines the valid application configuration to load, writes the remote system upgrade control register accordingly, and initiates system reconfiguration.

Stratix II FPGAs support the remote update mode in the AS, FPP, PS, and PPA configuration schemes. In the FPP, PS, and PPA schemes, the MAX II device, microprocessor, or enhanced configuration device should sample the `PGM[2..0]` outputs from the Stratix II FPGA and transmit the appropriate configuration image. In the AS scheme, the Stratix II device uses the page addresses to read configuration data out of the serial configuration device.

Local Update Mode

Local update mode is a simplified version of the remote update mode. This feature allows systems to load an application configuration immediately upon power up without loading the factory configuration first. Local update mode does not require the factory configuration to determine which application configuration to load, because only one application configuration is allowed (at page address one (`PGM [2..0] = 001`). You can update this application configuration remotely. If an error occurs while loading the application configuration, the factory configuration is automatically loaded.

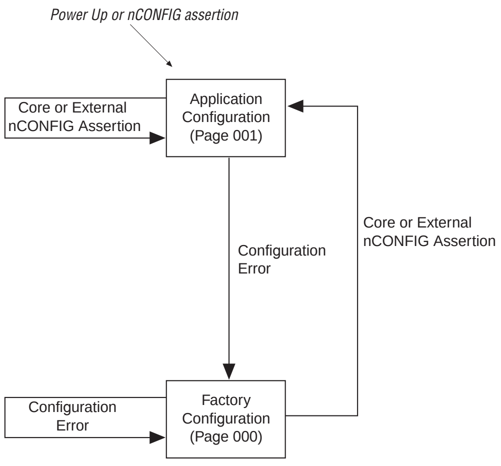
Upon power up or `nCONFIG` assertion, the dedicated remote system upgrade circuitry drives out "001" on the `PGM[]` pins selecting the application configuration stored in page one. If the device encounters any errors during the configuration cycle, the remote system upgrade circuitry retries configuration by driving `PGM[2..0]` to zero (`PGM[2..0] = 000`) to select the factory configuration image. The error conditions that trigger a return to the factory configuration are:

- An internal `CRC` error
- An external error signal (`nSTATUS` detected low)

When the remote system upgrade circuitry detects an external configuration reset (`nCONFIG` pulsed low) or internal configuration reset (logic array `nCONFIG` assertion), the device attempts to reload the application configuration from page one.

Figure 8–5 shows the transitions between configurations in local update mode.

Figure 8–5. Transitions Between Configurations in Local Update Mode


Power Up or nCONFIG assertion
Core or External nCONFIG Assertion
Application Configuration (Page 001)
Configuration Error
Core or External nCONFIG Assertion
Configuration Error
Factory Configuration (Page 000)

Stratix II FPGAs support local update mode in the FPP, PS, and PPA configuration schemes. In these schemes, the MAX II device, microprocessor, or enhanced configuration device should sample the `PGM[2..0]` outputs from the Stratix II FPGA and transmit the appropriate configuration image.

Local update mode is not supported with the AS configuration scheme, (or serial configuration device), because the Stratix II FPGA cannot determine the start address of the application configuration page upon power up. While the factory configuration is always located at memory address 0x000000, the application configuration can be located at any other sector boundary within the serial configuration device. The start address depends on the size of the factory configuration and is user selectable. Hence, only remote update mode is supported in the AS configuration scheme.

☞ Local update mode is not supported in the AS configuration scheme (with a serial configuration device).

Local update mode supports read access to the remote system upgrade status register. The factory configuration image can use this error status information to determine if a new application configuration must be downloaded from the remote source. After a remote update, the user design should assert the logic array configuration reset (`nCONFIG`) signal to load the new application configuration.

The device does not support write access to the remote system upgrade control register in local update mode. Write access is not required because this mode only supports one application configuration (eliminating the need to write in a page address) and does not support the user watchdog timer (eliminating the need to enable or disable the timer or specify its time-out value).

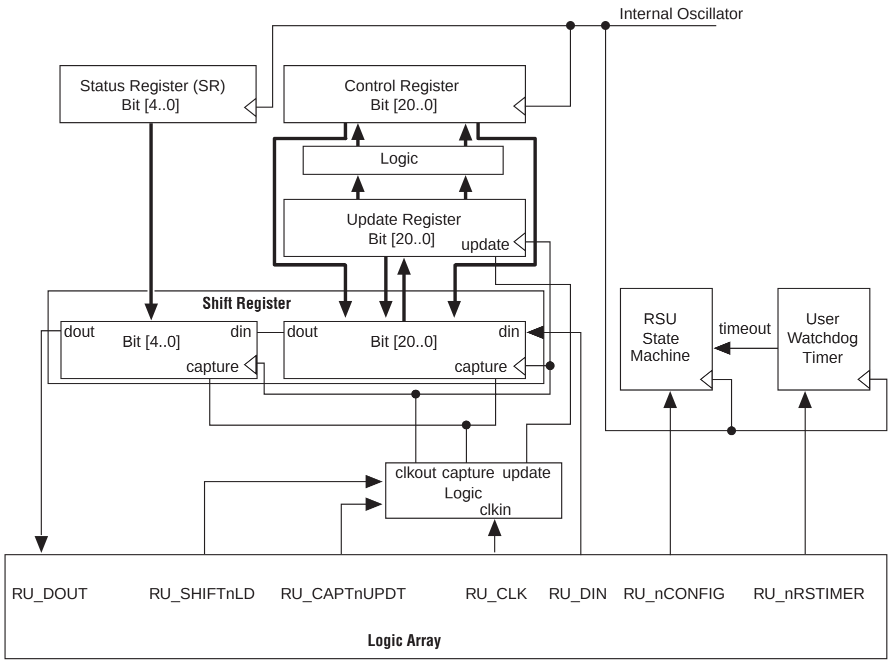
☞ The user watchdog timer is disabled in local update mode.

☞ Write access to the remote system upgrade control register is disabled in local update mode. However, the device supports read access to obtain error status information.

Dedicated Remote System Upgrade Circuitry

This section explains the implementation of the Stratix II remote system upgrade dedicated circuitry. The remote system upgrade circuitry is implemented in hard logic. This dedicated circuitry interfaces to the user-defined factory application configurations implemented in the FPGA logic array to provide the complete remote configuration solution. The remote system upgrade circuitry contains the remote system upgrade registers, a watchdog timer, and a state machine that controls those components. Figure 8–6 shows the remote system upgrade block's data path.

Figure 8–6. Remote System Upgrade Circuit Data Path



Remote System Upgrade Registers

The remote system upgrade block contains a series of registers that store the page addresses, watchdog timer settings, and status information. These registers are detailed in Table 8–3.

Table 8–3. Remote System Upgrade Registers

Register	Description
Shift register	This register is accessible by the logic array and allows the update, status, and control registers to be written and sampled by user logic. Write access is enabled in remote update mode for factory configurations to allow writes to the update register. Write access is disabled in local update mode and for all application configurations in remote update mode.
Control register	This register contains the current page address, the user watchdog timer settings, and one bit specifying whether the current configuration is a factory configuration or an application configuration. During a read operation in an application configuration, this register is read into the shift register. When a reconfiguration cycle is initiated, the contents of the update register are written into the control register.
Update register	This register contains data similar to that in the control register. However, it can only be updated by the factory configuration by shifting data into the shift register and issuing an update operation. When a reconfiguration cycle is triggered by the factory configuration, the control register is updated with the contents of the update register. During a read in a factory configuration, this register is read into the shift register.
Status register	This register is written to by the remote system upgrade circuitry on every reconfiguration to record the cause of the reconfiguration. This information is used by the factory configuration to determine the appropriate action following a reconfiguration. During a capture cycle, this register is read into the shift register.

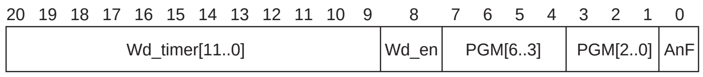
The remote system upgrade control and status registers are clocked by the 10-Mhz internal oscillator (the same oscillator that controls the user watchdog timer). However, the remote system upgrade shift and update registers are clocked by the user clock input (`RU_CLK`).

Remote System Upgrade Control Register

The remote system upgrade control register stores the application configuration page address and user watchdog timer settings. The control register functionality depends on the remote system upgrade mode selection. In remote update mode, the control register page address bits are set to all zeros (`7'b0 = 0000_000`) at power up in order to load the factory configuration. However, in local update mode the control register page address bits power up as (`7'b1 = 0000_001`) in order to select the application configuration. Additionally, the control register cannot be updated in local update mode, whereas a factory configuration in remote update mode has write access to this register.

The control register bit positions are shown in Figure 8–7 and defined in Table 8–4. In the figure, the numbers show the bit position of a setting within a register. For example, bit number 8 is the enable bit for the watchdog timer.

Figure 8–7. Remote System Upgrade Control Register



The application-not-factory (AnF) bit indicates whether the current configuration loaded in the Stratix II device is the factory configuration or an application configuration. This bit is set high at power up in local update mode, and is set low by the remote system upgrade circuitry when an error condition causes a fall-back to factory configuration. When the AnF bit is high, the control register access is limited to read operations. When the AnF bit is low, the register allows write operations and disables the watchdog timer.

In remote update mode, factory configuration design should set this bit high (1'b1) when updating the contents of the update register with application page address and watchdog timer settings.

Table 8–4. Remote System Upgrade Control Register Contents (Part 1 of 2)

Control Register Bit	Remote System Upgrade Mode	Value	Definition
AnF	Local update Remote update	1'b1 1'b0	Application not factory
PGM[2..0]	Local update Remote update (FPP, PS, PPA)	3'b001 3'b000	Page mode select
	Remote update (AS)	3'b000	AS configuration start address (StAdd[18..16])
PGM[6..3]	Local update Remote update (FPP, PS, PPA)	4'b0000 4'b0000	Not used
	Remote update (AS)	4'b0000	AS configuration start address (StAdd[22..19])

Table 8–4. Remote System Upgrade Control Register Contents (Part 2 of 2)

Control Register Bit	Remote System Upgrade Mode	Value	Definition
`Wd_en`	Remote update	1'b0	User watchdog timer enable bit
`Wd_timer[11..0]`	Remote update	12'b000000000000	User watchdog time-out value (most significant 12 bits of 29-bit count value: `{Wd_timer[11..0], 17'b0}`)

Remote System Upgrade Status Register

The remote system upgrade status register specifies the reconfiguration trigger condition. The various trigger and error conditions include:

- CRC (cyclic redundancy check) error during application configuration
- `nSTATUS` assertion by an external device due to an error
- FPGA logic array triggered a reconfiguration cycle, possibly after downloading a new application configuration image
- External configuration reset (`nCONFIG`) assertion
- User watchdog timer time out

Figure 8–8 and Table 8–5 specify the contents of the status register. The numbers in the figure show the bit positions within a 5-bit register.

Figure 8–8. Remote System Upgrade Status Register

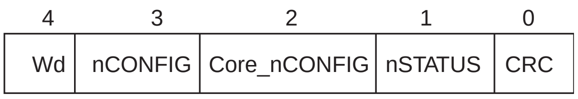


Table 8–5. Remote System Upgrade Status Register Contents (Part 1 of 2)

Status Register Bit	Definition	POR Reset Value
`CRC` (from configuration)	`CRC` error caused reconfiguration	1 bit '0'
`nSTATUS`	`nSTATUS` caused reconfiguration	1 bit '0'

Table 8–5. Remote System Upgrade Status Register Contents (Part 2 of 2)

Status Register Bit	Definition	POR Reset Value
`CORE` *(1)* `CORE_nCONFIG`	Device logic array caused reconfiguration	1 bit '0'
`nCONFIG`	`nCONFIG` caused reconfiguration	1 bit '0'
`Wd`	Watchdog timer caused reconfiguration	1 bit '0'

Note to Table 8–5:

(1) Logic array reconfiguration forces the system to load the application configuration data into the Stratix II device. This occurs after the factory configuration specifies the appropriate application configuration page address by updating the update register.

Remote System Upgrade State Machine

The remote system upgrade control and update registers have identical bit definitions, but serve different roles (see Table 8–3 on page 8–15). While both registers can only be updated when the FPGA is loaded with a factory configuration image, the update register writes are controlled by the user logic, and the control register writes are controlled by the remote system upgrade state machine.

In factory configurations, the user logic should send the `AnF` bit (set high), the page address, and watchdog timer settings for the next application configuration bit to the update register. When the logic array configuration reset (`RU_nCONFIG`) goes high, the remote system upgrade state machine updates the control register with the contents of the update register and initiates system reconfiguration from the new application page.

In the event of an error or reconfiguration trigger condition, the remote system upgrade state machine directs the system to load a factory or application configuration (page zero or page one, based on mode and error condition) by setting the control register accordingly. Table 8–6 lists the contents of the control register after such an event occurs for all possible error or trigger conditions.

The remote system upgrade status register is updated by the dedicated error monitoring circuitry after an error condition but before the factory configuration is loaded.

Table 8–6. Control Register Contents After an Error or Reconfiguration Trigger Condition

Reconfiguration Error/Trigger	Control Register Setting	
	Remote Update	Local Update

`nCONFIG` reset	All bits are 0	`PGM[6..0] = 7'b0000001`
		`AnF = 1`
		All other bits are 0

`nSTATUS` error	All bits are 0	All bits are 0

`CORE` triggered reconfiguration	Update register	`PGM[6..0] = 7'b0000001`
		`AnF = 1`
		All other bits are 0

`CRC` error	All bits are 0	All bits are 0
`Wd` time out	All bits are 0	All bits are 0

Read operations during factory configuration access the contents of the update register. This feature is used by the user logic to verify that the page address and watchdog timer settings were written correctly. Read operations in application configurations access the contents of the control register. This information is used by the user logic in the application configuration.

User Watchdog Timer

The user watchdog timer prevents a faulty application configuration from stalling the device indefinitely. The system uses the timer to detect functional errors after an application configuration is successfully loaded into the FPGA.

The user watchdog timer is a counter that counts down from the initial value loaded into the remote system upgrade control register by the factory configuration. The counter is 29-bits-wide and has a maximum count value of 2^{29}. When specifying the user watchdog timer value, specify only the most significant 12 bits. The granularity of the timer setting is 2^{15} cycles. The cycle time is based on the frequency of the 10-MHz internal oscillator. Table 8–7 specifies the operating range of the 10-MHz internal oscillator.

Table 8–7. 10-MHz Internal Oscillator Specifications Note (1)

Minimum	Typical	Maximum	Units
5	6.5	10	MHz

Note to Table 8–7:
(1) These values are preliminary.

The user watchdog timer begins counting once the application configuration enters FPGA user mode. This timer must be periodically reloaded or reset by the application configuration before the timer expires by asserting `RU_nRSTIMER`. If the application configuration does not reload the user watchdog timer before the count expires, a time-out signal is generated by the remote system upgrade dedicated circuitry. The time-out signal tells the remote system upgrade circuitry to set the user watchdog timer status bit (`Wd`) in the remote system upgrade status register and reconfigures the device by loading the factory configuration.

The user watchdog timer is not enabled during the configuration cycle of the FPGA. Errors during configuration are detected by the CRC engine. Also, the timer is disabled for factory configurations. Functional errors should not exist in the factory configuration since it is stored and validated during production and is never updated remotely.

☞ The user watchdog timer is disabled in factory configurations and during the configuration cycle of the application configuration. It is enabled after the application configuration enters user mode.

Interface Signals between Remote System Upgrade Circuitry & FPGA Logic Array

The dedicated remote system upgrade circuitry drives (or receives) seven signals to (or from) the FPGA logic array. The FPGA logic array uses these signals to read and write the remote system upgrade control, status, and update registers using the remote system upgrade shift register. Table 8–8 lists each of these seven signals and describes their functionality.

Except for `RU_nRSTIMER` and `RU_CAPTnUPDT`, the logic array signals are enabled for both remote and local update modes and for both factory and application configurations. `RU_nRSTIMER` is only valid for application configurations in remote update mode, since local update configurations and factory configurations have the user watchdog timer disabled. When RU_CAPTnUPDT is low, the device can write to the update register only for factory configurations in remote update mode, since this is the only case where the update register is written to by the user logic. When the RU_nCONFIG signal goes high, the contents of the update register are written into the control register for controlling the next configuration cycle.

Table 8–8. Interface Signals between Remote System Upgrade Circuitry & FPGA Logic Array (Part 1 of 2)

Signal Name	Signal Direction	Description
RU_nRSTIMER	Input to remote system upgrade block (driven by FPGA logic array)	Request from the application configuration 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	Input to remote system upgrade block (driven by FPGA logic array)	When driven low, this signal triggers the device to reconfigure. If asserted by the factory configuration in remote update mode, the application configuration specified in the remote update control register is loaded. If requested by the application configuration in remote update mode, the factory configuration is loaded. In the local updated mode, the application configuration is loaded whenever this signal is asserted.
RU_CLK	Input to remote system upgrade block (driven by FPGA logic array)	Clocks the remote system upgrade shift register and update register so that the contents of the status, control, and update registers can be read, and so that the contents of the update register can be loaded. The shift register latches data on the rising edge of this clock signal.
RU_SHIFTnLD	Input to remote system upgrade block (driven by FPGA logic array)	This pin determines if the shift register contents are shifted over during the next clock edge or loaded in/out. When this signal is driven high (1'b1), the remote system upgrade shift register shifts data left on each rising edge of RU_CLK. When RU_SHIFTnLD is driven low (1'b0) and RU_CAPTnUPDT is driven low (1'b0), the remote system upgrade update register is updated with the contents of the shift register on the rising edge of RU_CLK. When RU_SHIFTnLD is driven low (1'b0) and RU_CAPTnUPDT is driven high (1'b1), the remote system upgrade shift register captures the status register and either the control or update register (depending on whether the current configuration is application or factory, respectively) on the rising edge of RU_CLK.

Table 8–8. Interface Signals between Remote System Upgrade Circuitry & FPGA Logic Array (Part 2 of 2)

Signal Name	Signal Direction	Description
`RU_CAPTnUPDT`	Input to remote system upgrade block (driven by FPGA logic array)	This pin determines if the contents of the shift register are captured or updated on the next clock edge. When the `RU_SHIFTnLD` signal is driven high (1'b1), this input signal has no function. When `RU_SHIFTnLD` is driven low (1'b0) and `RU_CAPTnUPDT` is driven high (1'b1), the remote system upgrade shift register captures the status register and either the control or update register (depending on whether the current configuration is application or factory, respectively) on the rising edge of `RU_CLK`. When `RU_SHIFTnLD` is driven low (1'b0) and `RU_CAPTnUPDT` is driven low (1'b0), the remote system upgrade update register is updated with the contents of the shift register on the rising edge of `RU_CLK`. In local update mode, a low input on `RU_CAPTnUPDT` has no function, because the update register cannot be updated in this mode.
`RU_DIN`	Input to remote system upgrade block (driven by FPGA logic array)	Data to be written to the remote system upgrade shift register on the rising edge of `RU_CLK`. To load data into the shift register, `RU_SHIFTnLD` must be asserted.
`RU_DOUT`	Output from remote system upgrade block (driven to FPGA logic array)	Output data from the remote system upgrade shift register to be read by logic array logic. New data arrives on each rising edge of `RU_CLK`.

Remote System Upgrade Pin Descriptions

Table 8–9 describes the dedicated remote system upgrade configuration pins. For descriptions of all the configuration pins, refer to the *Configuring Stratix II Devices* chapter of the *Stratix II Handbook*.

Table 8–9. Stratix II Remote System Upgrade Pins

Pin Name	User Mode	Configuration Scheme	Pin Type	Description
`RUnLU`	N/A if using remote system upgrade in FPP, PS, or PPA modes. I/O if not using these modes.	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 pin is available as a general-purpose user I/O pin. When using remote configuration in AS mode, the `RUnLU` pin is available as a general-purpose I/O pin.
`PGM[2..0]`	N/A if using remote system upgrade in FPP, PS, or PPA modes. I/O if not using these modes.	Remote configuration in FPP, PS or PPA	Output	These output pins select one of eight pages in the memory (either flash or enhanced configuration device) when using remote update mode. When not using remote update or local update configuration modes, these pins are available as general-purpose user I/O pins. When using remote configuration in AS mode, the `PGM[]` pins are available as a general-purpose I/O pins.

Quartus II Software Support

Implementation in your design requires an remote system upgrade interface between the FPGA logic array and remote system upgrade circuitry. You also need to generate configuration files for production and remote programming of the system configuration memory. The Quartus® II software provides these features.

The two implementation options, `altremote_update` megafunction and remote system upgrade atom, are for the interface between the remote system upgrade circuitry and the FPGA logic array interface.

altremote_update Megafunction

The `altremote_update` megafunction provides a memory-like interface to the remote system upgrade circuitry and handles the shift register read/write protocol in FPGA logic. This implementation is suitable for designs that implement the factory configuration functions using a Nios processor in the FPGA.

Tables 8–10 and 8–11 describe the input and output ports available on the `altremote_update` megafunction. Table 8–12 shows the `param[2..0]` bit settings.

Table 8–10. 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.
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.

Table 8–10. Input Ports of the altremote_update Megafunction (Part 2 of 2)

Port Name	Required	Source	Description
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 8–10:

(1) Logic array source means that you can drive the port from internal logic or any general-purpose I/O pin.

Table 8–11. 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 8–11:

(1) Logic array destination means that you can drive the port to internal logic or any general-purpose I/O pin.

Table 8–12. 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 8–12. 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	Page mode selection. Writing of this parameter is only allowed when in the factory configuration.
			3 bit '000' - Remote configuration	
Current configuration (AnF)	101	1	1 bit '0' - Factory	Specifies whether the current configuration is factory or and application configuration. This parameter can only be read.
			1 bit '1' - Application	
Illegal values	001			
	110			
	111			

Remote System Upgrade Atom

The remote system upgrade atom is a WYSIWYG atom or primitive that can be instantiated in your design. The primitive is used to access the remote system upgrade shift register, logic array reset, and watchdog timer reset signals. The ports on this primitive are the same as those listed in Table 8–8 on page 8–21. This implementation is suitable for designs that implement the factory configuration functions using state machines (without a processor).

System Design Guidelines

The following general guidelines are applicable when implementing remote system upgrade in Stratix II FPGAs. Guidelines for specific configuration schemes are also discussed in this section.

- After downloading a new application configuration, the soft logic implemented in the FPGA can validate the integrity of the data received over the remote communication interface. This optional step helps avoid configuration attempts with bad or incomplete configuration data. However, in the event that bad or incomplete configuration data is sent to the FPGA, it detects the data corruption using the CRC signature attached to each configuration frame.
- The auto-reconfigure on configuration error option bit is ignored when remote system upgrade is enabled in your system. This option is always enabled in remote configuration designs, allowing your system to return to the safe factory configuration in the event of an application configuration error or user watchdog timer time out.

Remote System Upgrade With Serial Configuration Devices

Remote system upgrade support in the AS configuration scheme is similar to support in other schemes, with the following exceptions:

- The remote system upgrade block provides the AS configuration controller inside the Stratix II FPGA with a 7-bit page start address (`PGM[6..0]`) instead of driving the 3-bit page mode pins (`PGM[2..0]`) used in FPP, PS, and PPA configuration schemes. This 7-bit address forms the 24-bit configuration start address (`StAdd[23..0]`). Table 8–13 illustrates the start address generation using the page address registers.
- The configuration start address for factory configuration is always set to 24'b0.
- `RUnLU` and `PGM[2..0]` pins on the Stratix II device are not used in AS configuration scheme and can be used as regular I/O pins.
- The Nios ASMI peripheral can be used to update configuration data within the serial configuration device.

Table 8–13. AS Configuration Start Address Generation

Serial Configuration Device	Serial Configuration Device Density (MB)	Add[23]	PGM[6..0] (Add[22..16])	Add[15..0]
EPCS64	64	0	`MSB[6..0]`	All 0s
EPCS16	16	0	00, `MSB[4..0]`	All 0s
EPCS4	4	0	0000, `MSB[2..0]`	All 0s

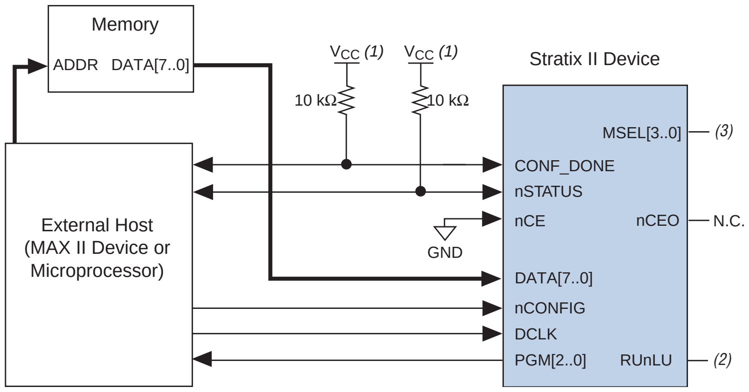
Remote System Upgrade With a MAX II Device or Microprocessor & Flash Device

This setup requires the MAX II device or microprocessor to support page addressing. MAX II or microprocessor devices implementing remote system upgrade should emulate the enhanced configuration device page mode feature. The `PGM[2..0]` output pins from the Stratix II device must be sampled to determine which configuration image is to be loaded into the FPGA.

If the FPGA does not release `CONF_DONE` after all data has been sent, the MAX II microprocessor should reset the FPGA back to the factory image by pulsing its `nSTATUS` pin low.

The MAX II device or microprocessor and flash configuration can use FPP, PS, or PPA. Decompression and design security features are supported in the FPP (requires 4× DCLK) and PS modes only. Figure 8–9 shows a system block diagram for remote system upgrade with the MAX II device or microprocessor and flash.

Figure 8–9. System Block Diagram for Remote System Upgrade With MAX II Device or Microprocessor & Flash Device



Notes to Figure 8–9:

(1) Connect the pull-up resistor to a supply that provides an acceptable input signal for the device.

(2) Connect `RUnLU` to GND or V_{CC} to select between remote and local update modes.

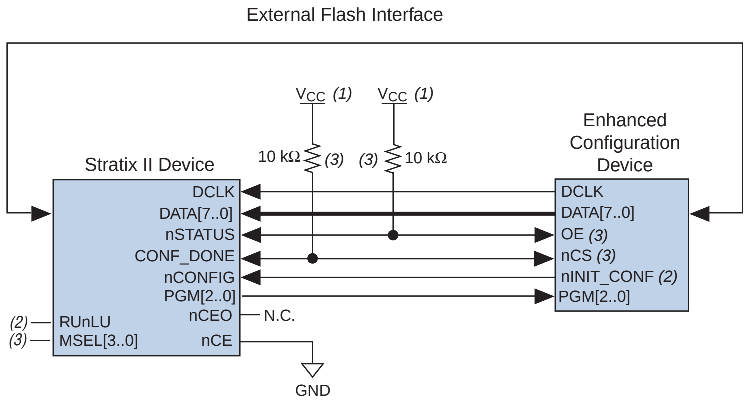
(3) Connect `MSEL[3..0]` to `0100` to enable remote update remote system upgrade mode.

Remote System Upgrade with Enhanced Configuration Devices

- Enhanced Configuration devices support remote system upgrade with FPP or PS configuration schemes. The Stratix II decompression and design security features are only supported in the PS mode. The enhanced configuration device's decompression feature is supported in both PS and FPP schemes.
- In remote update mode, neither the factory configuration nor the application configurations should alter the enhanced configuration device's option bits or the page 000 factory configuration data. This ensures that an error during remote update can always be resolved by reverting to the factory configuration located at page 000.

- The enhanced configuration device features an error checking mechanism to detect instances when the FPGA fails to detect the configuration preamble. In these instances, the enhanced configuration device pulses the `nSTATUS` signal low, and the remote system upgrade circuitry attempts to load the factory configuration. Figure 8–10 shows a system block diagram for remote system upgrade with enhanced configuration devices.

Figure 8–10. System Block Diagram for Remote System Upgrade with Enhanced Configuration Devices



Notes to Figure 8–10:

(1) Connect the pull-up resistor to a supply that provides an acceptable input signal for the device.

(2) Connect `RUnLU` to GND or V_{CC} to select between remote and local update modes.

(3) Connect `MSEL[3..0]` to `0100` to enable remote update remote system upgrade mode.

Conclusion

Stratix II devices offer remote system upgrade capability, where you can upgrade a system in real-time through any network. Remote system upgrade helps to deliver feature enhancements and bug fixes without costly recalls, reduces time to market, and extends product life cycles. The dedicated remote system upgrade circuitry in Stratix II devices provides error detection, recovery, and status information to ensure reliable reconfiguration.

9. IEEE 1149.1 (JTAG) Boundary-Scan Testing for Stratix II Devices

SII52009-2.0

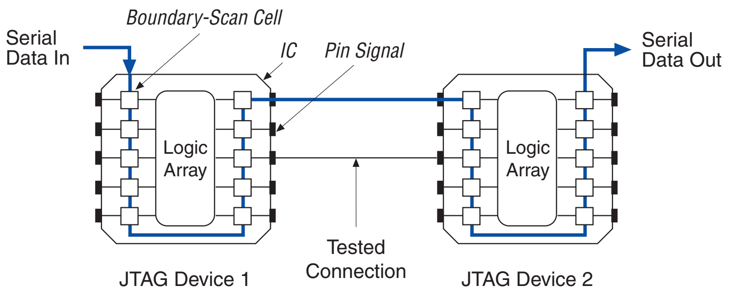
Introduction

As printed circuit boards (PCBs) become more complex, the need for thorough testing becomes increasingly important. Advances in surface-mount packaging and PCB manufacturing have resulted in smaller boards, making traditional test methods (e.g., external test probes and "bed-of-nails" test fixtures) harder to implement. As a result, cost savings from PCB space reductions are sometimes offset by cost increases in traditional testing methods.

In the 1980s, the Joint Test Action Group (JTAG) developed a specification for boundary-scan testing that was later standardized as the IEEE Std. 1149.1 specification. This boundary-scan test (BST) architecture offers the capability to efficiently test components on PCBs with tight lead spacing.

This BST architecture can test pin connections without using physical test probes and capture functional data while a device is operating normally. Boundary-scan cells in a device can force signals onto pins or capture data from pin or logic array signals. Forced test data is serially shifted into the boundary-scan cells. Captured data is serially shifted out and externally compared to expected results. Figure 9–1 illustrates the concept of boundary-scan testing.

Figure 9–1. IEEE Std. 1149.1 Boundary-Scan Testing



This chapter discusses how to use the IEEE Std. 1149.1 BST circuitry in Stratix® II devices, including:

- IEEE Std. 1149.1 BST architecture
- IEEE Std. 1149.1 boundary-scan register
- IEEE Std. 1149.1 BST operation control
- I/O Voltage Support in JTAG Chain
- Utilizing IEEE Std. 1149.1 BST circuitry
- Disabling IEEE Std. 1149.1 BST circuitry
- Guidelines for IEEE Std. 1149.1 boundary-scan testing
- Boundary-Scan Description Language (BSDL) support

In addition to BST, you can use the IEEE Std. 1149.1 controller for Stratix II device in-circuit reconfiguration (ICR). However, this chapter only discusses the BST feature of the IEEE Std. 1149.1 circuitry. For information on configuring Stratix II devices via the IEEE Std. 1149.1 circuitry, see the *Configuring Stratix II Devices* chapter of the *Stratix II Handbook, Volume 2*.

☞ Stratix II, Stratix, Cyclone II, and Cyclone devices must be within the first 17 devices in a chain when configuring via JTAG. All of these devices have the same JTAG controller. If any of the Stratix II, Stratix, Cyclone II, and Cyclone devices are in the 18th or further position, they will fail configuration. This does not affect SignalTap II or boundary-scan testing.

IEEE Std. 1149.1 BST Architecture

A Stratix II device operating in IEEE Std. 1149.1 BST mode uses four required pins, `TDI`, `TDO`, `TMS` and `TCK`, and one optional pin, `TRST`. The `TCK` pin has an internal weak pull-down resistor, while the `TDI`, `TMS` and `TRST` pins have weak internal pull-ups. The `TDO` output pin is powered by V_{CCIO} in I/O bank 4. All of the JTAG input pins are powered by the 3.3-V V_{CCPD} supply. All user I/O pins are tri-stated during JTAG configuration. For recommendations on how to connect a JTAG chain with multiple voltages across the devices in the chain, refer to the "I/O Voltage Support in JTAG Chain" section. Table 9–1 summarizes the functions of each of these pins.

Table 9–1. IEEE Std. 1149.1 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`.
`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.

Table 9–1. IEEE Std. 1149.1 Pin Descriptions

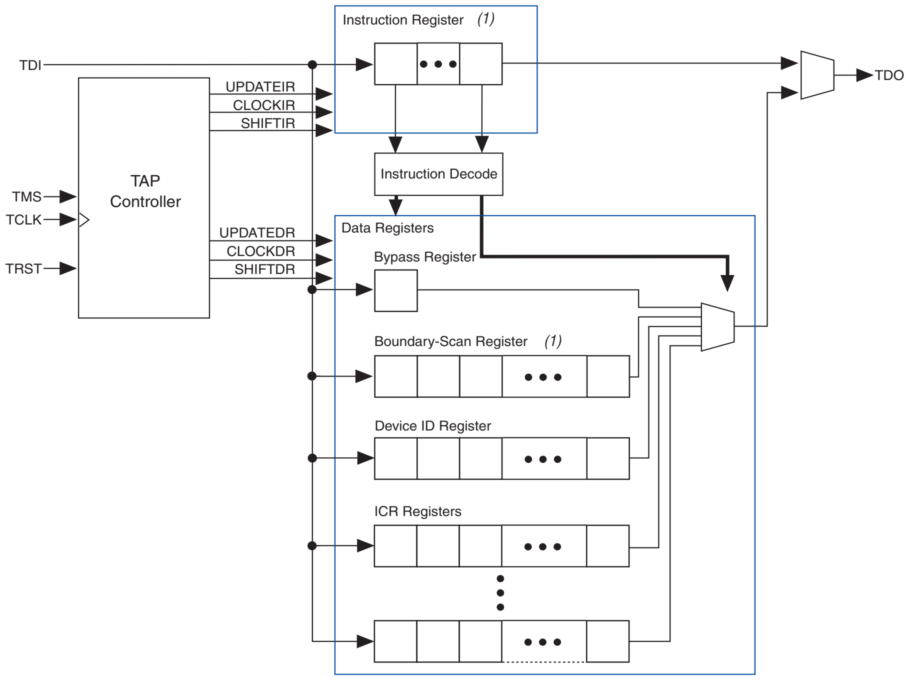
Pin	Description	Function
TMS	Test mode select	Input pin that provides the control signal to determine the transitions of the TAP controller state machine. Transitions within the state machine occur at the rising edge of TCK. Therefore, TMS must be set up before the rising edge of TCK. TMS is evaluated on the rising edge of TCK.
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.
TRST	Test reset input (optional)	Active-low input to asynchronously reset the boundary-scan circuit. This pin should be driven low when not in boundary-scan operation and for non-JTAG users the pin should be permanently tied to GND.

The IEEE Std. 1149.1 BST circuitry requires the following registers:

- The instruction register determines the action to be performed and the data register to be accessed.
- The bypass register is a 1-bit-long data register that provides a minimum-length serial path between TDI and TDO.
- The boundary-scan register is a shift register composed of all the boundary-scan cells of the device.

Figure 9–2 shows a functional model of the IEEE Std. 1149.1 circuitry.

Figure 9–2. IEEE Std. 1149.1 Circuitry



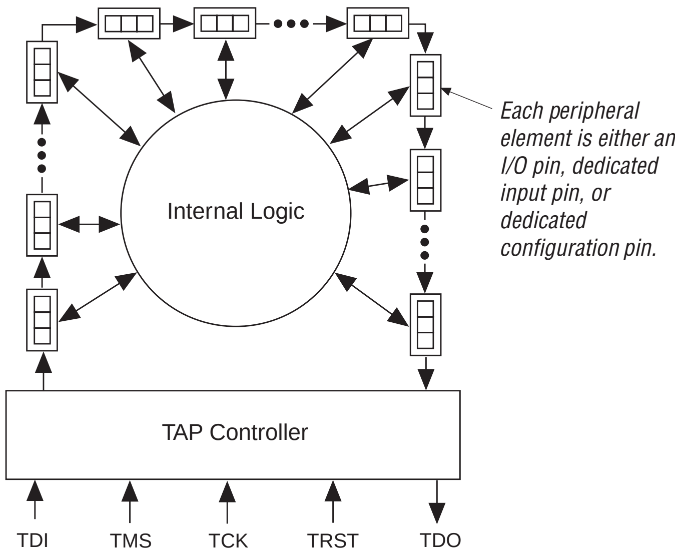
Note to Figure 9–2:
(1) Refer to the appropriate device data sheet for register lengths.

IEEE Std. 1149.1 boundary-scan testing is controlled by a test access port (TAP) controller. For more information on the TAP controller, see the "IEEE Std. 1149.1 BST Operation Control" section. The `TMS` and `TCK` pins operate the TAP controller, and the `TDI` and `TDO` pins provide the serial path for the data registers. The `TDI` pin also provides data to the instruction register, which then generates control logic for the data registers.

IEEE Std. 1149.1 Boundary-Scan Register

The boundary-scan register is a large serial shift register that uses the `TDI` pin as an input and the `TDO` pin as an output. The boundary-scan register consists of 3-bit peripheral elements that are associated with Stratix II I/O pins. You can use the boundary-scan register to test external pin connections or to capture internal data. See the *Configuration & Testing* chapter of the *Stratix II Handbook, Volume 1* for the Stratix II device boundary-scan register lengths. Figure 9–3 shows how test data is serially shifted around the periphery of the IEEE Std. 1149.1 device.

Figure 9–3. Boundary-Scan Register


Internal Logic
Each peripheral element is either an I/O pin, dedicated input pin, or dedicated configuration pin.
TAP Controller
TDI
TMS
TCK
TRST
TDO

Boundary-Scan Cells of a Stratix II Device I/O Pin

The Stratix II device 3-bit boundary-scan cell (BSC) consists of a set of capture registers and a set of update registers. The capture registers can connect to internal device data via the `OUTJ`, `OEJ`, and `PIN_IN` signals, while the update registers connect to external data through the `PIN_OUT`, and `PIN_OE` signals. The global control signals for the IEEE Std. 1149.1 BST registers (e.g., shift, clock, and update) are generated internally by the TAP controller. The `MODE` signal is generated by a decode of the instruction register. The data signal path for the boundary-scan register runs from the serial data in (SDI) signal to the serial data out (SDO) signal. The scan register begins at the `TDI` pin and ends at the `TDO` pin of the device.

Figure 9–4 shows the Stratix II device's user I/O boundary-scan cell.

Figure 9–4. Stratix II Device's User I/O BSC with IEEE Std. 1149.1 BST Circuitry

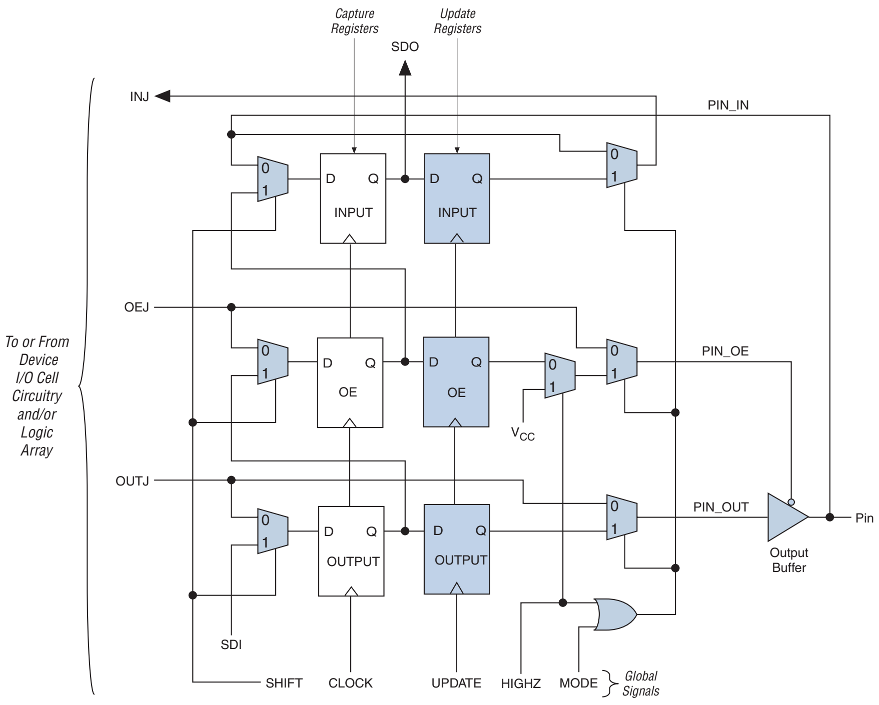


Table 9–2 describes the capture and update register capabilities of all boundary-scan cells within Stratix II devices.

Table 9–2. Stratix II Device Boundary Scan Cell Descriptions (Part 1 of 2) Note (1)

Pin Type	Captures			Drives			Comments
	Output Capture Register	OE Capture Register	Input Capture Register	Output Update Register	OE Update Register	Input Update Register	
User I/O pins	`OUTJ`	`OEJ`	`PIN_IN`	`PIN_OUT`	`PIN_OE`	`INJ`	
Dedicated clock input	0	1	`PIN_IN`	N.C. *(2)*	N.C. *(2)*	N.C. *(2)*	`PIN_IN` drives to clock network or logic array
Dedicated input *(3)*	0	1	`PIN_IN`	N.C. *(2)*	N.C. *(2)*	N.C. *(2)*	`PIN_IN` drives to control logic

Table 9–2. Stratix II Device Boundary Scan Cell Descriptions (Part 2 of 2) Note (1)

Pin Type	Captures			Drives			Comments
	Output Capture Register	OE Capture Register	Input Capture Register	Output Update Register	OE Update Register	Input Update Register	
Dedicated bidirectional *(4)*	0	`OEJ`	`PIN_IN`	N.C. *(2)*	N.C. *(2)*	N.C. *(2)*	`PIN_IN` drives to configuration control
Dedicated output *(5)*	`OUTJ`	0	0	N.C. *(2)*	N.C. *(2)*	N.C. *(2)*	`OUTJ` drives to output buffer

Notes to Table 9–2:

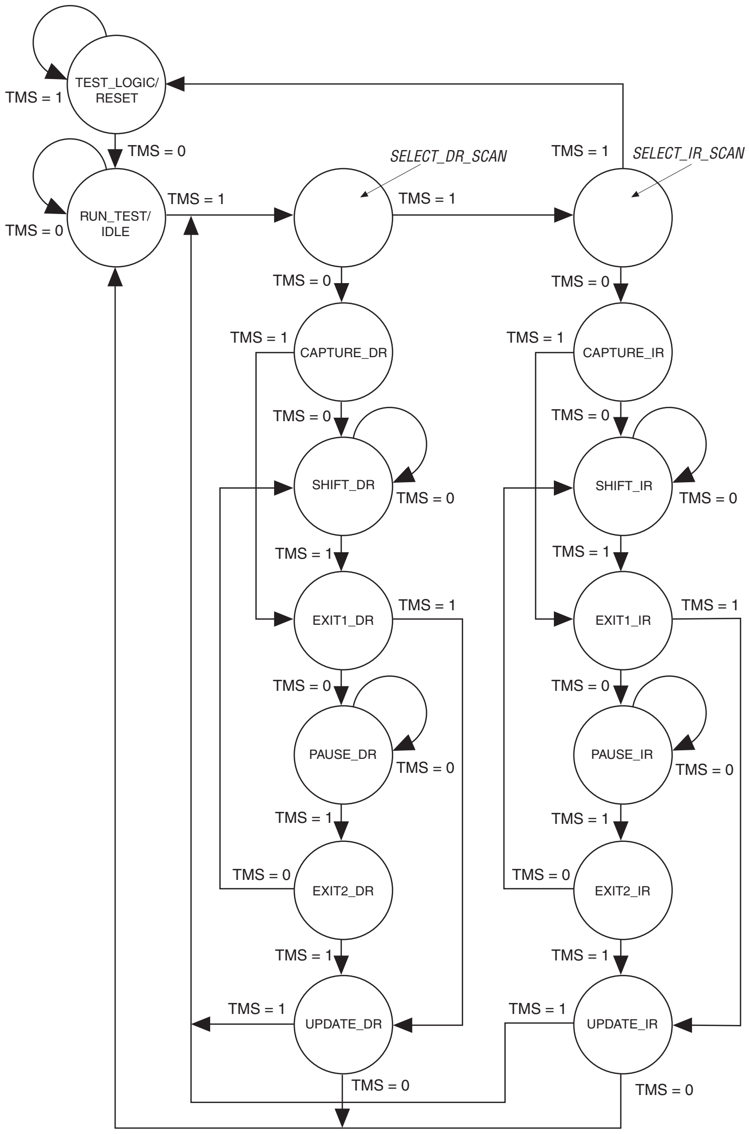
(1) `TDI`, `TDO`, `TMS`, `TCK`, all V_{CC} and GND pin types, `VREF`, and `TEMP_DIODE` pins do not have BSCs.
(2) N.C.: No Connect.
(3) This includes pins `PLL_ENA`, `nCONFIG`, `MSEL0`, `MSEL1`, `MSEL2`, `MSEL3`, `nCE`, `VCCSEL`, `PORSEL`, and `nIO_PULLUP`.
(4) This includes pins `CONF_DONE`, `nSTATUS`, and `DCLK`.
(5) This includes pin `nCEO`.

IEEE Std. 1149.1 BST Operation Control

Stratix II devices implement the following IEEE Std. 1149.1 BST instructions: `SAMPLE/PRELOAD`, `EXTEST`, `BYPASS`, `IDCODE`, `USERCODE`, `CLAMP` and `HIGHZ`. The BST instruction length is 10 bits. These instructions are described later in this chapter. For summaries of the BST instructions and their instruction codes, see the *Configuration & Testing* chapter of the *Stratix II Handbook, Volume 1*.

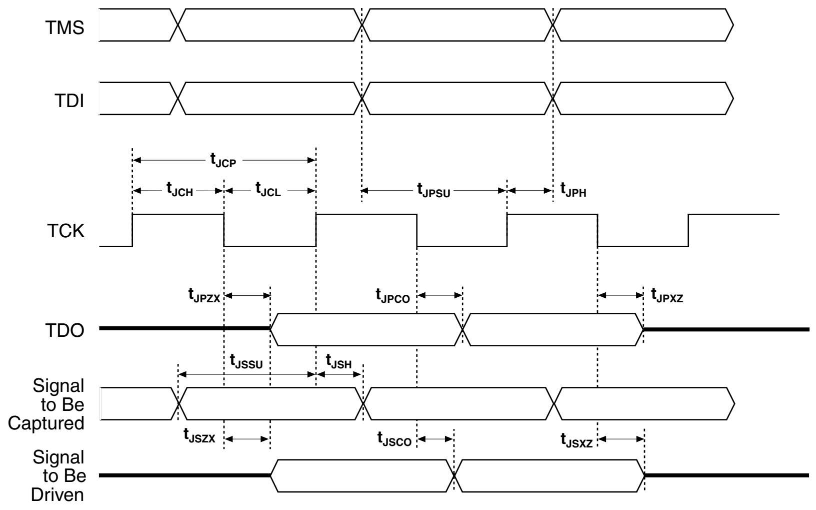
The IEEE Std. 1149.1 test access port (TAP) controller, a 16-state state machine clocked on the rising edge of `TCK`, uses the `TMS` pin to control IEEE Std. 1149.1 operation in the device. Figure 9–5 shows the TAP controller state machine.

Figure 9–5. IEEE Std. 1149.1 TAP Controller State Machine



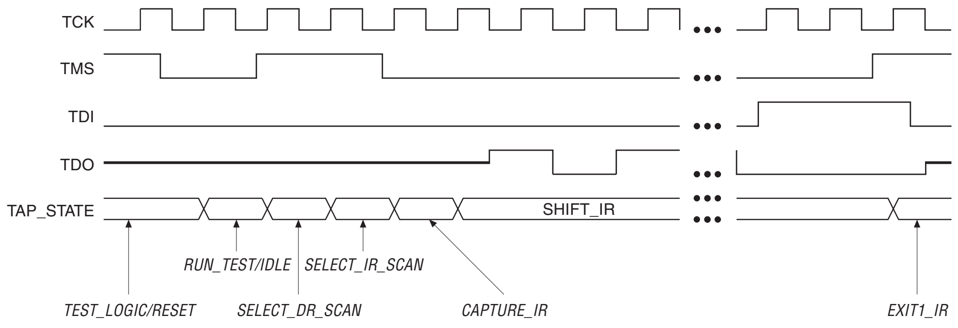
When the TAP controller is in the TEST_LOGIC/RESET state, the BST circuitry is disabled, the device is in normal operation, and the instruction register is initialized with IDCODE as the initial instruction. At device power-up, the TAP controller starts in this TEST_LOGIC/RESET state. In addition, forcing the TAP controller to the TEST_LOGIC/RESET state is done by holding TMS high for five TCK clock cycles or by holding the TRST pin low. Once in the TEST_LOGIC/RESET state, the TAP controller remains in this state as long as TMS is held high (while TCK is clocked) or TRST is held low. Figure 9–6 shows the timing requirements for the IEEE Std. 1149.1 signals.

Figure 9–6. IEEE Std. 1149.1 Timing Waveforms



To start IEEE Std. 1149.1 operation, select an instruction mode by advancing the TAP controller to the shift instruction register (SHIFT_IR) state and shift in the appropriate instruction code on the TDI pin. The waveform diagram in Figure 9–7 represents the entry of the instruction code into the instruction register. It shows the values of TCK, TMS, TDI, TDO, and the states of the TAP controller. From the RESET state, TMS is clocked with the pattern 01100 to advance the TAP controller to SHIFT_IR.

Figure 9–7. Selecting the Instruction Mode



The TDO pin is tri-stated in all states except in the SHIFT_IR and SHIFT_DR states. The TDO pin is activated at the first falling edge of TCK after entering either of the shift states and is tri-stated at the first falling edge of TCK after leaving either of the shift states.

When the SHIFT_IR state is activated, TDO is no longer tri-stated, and the initial state of the instruction register is shifted out on the falling edge of TCK. TDO continues to shift out the contents of the instruction register as long as the SHIFT_IR state is active. The TAP controller remains in the SHIFT_IR state as long as TMS remains low.

During the SHIFT_IR state, an instruction code is entered by shifting data on the TDI pin on the rising edge of TCK. The last bit of the instruction code must be clocked at the same time that the next state, EXIT1_IR, is activated. Set TMS high to activate the EXIT1_IR state. Once in the EXIT1_IR state, TDO becomes tri-stated again. TDO is always tri-stated except in the SHIFT_IR and SHIFT_DR states. After an instruction code is entered correctly, the TAP controller advances to serially shift test data in one of three modes (SAMPLE/PRELOAD, EXTEST, or BYPASS) that are described below.

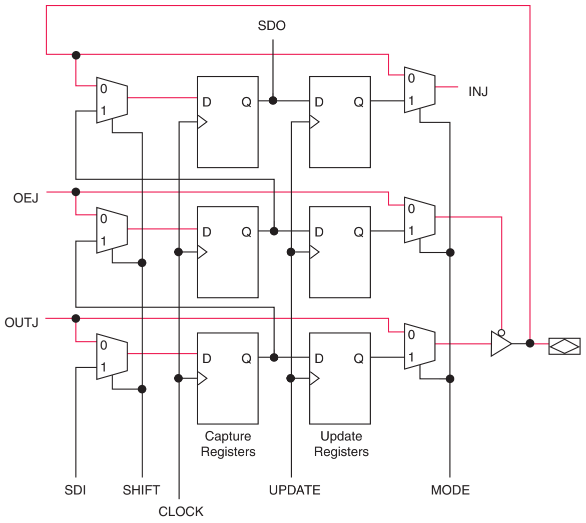
SAMPLE/PRELOAD Instruction Mode

The SAMPLE/PRELOAD instruction mode allows you to take a snapshot of device data without interrupting normal device operation. However, this instruction is most often used to preload the test data into the update registers prior to loading the EXTEST instruction. Figure 9–8 shows the capture, shift, and update phases of the SAMPLE/PRELOAD mode.

Figure 9–8. IEEE Std. 1149.1 BST SAMPLE/PRELOAD Mode

Capture Phase

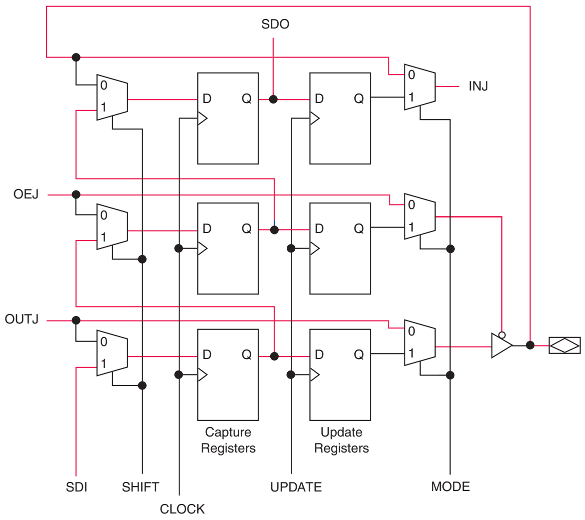
In the capture phase, the signals at the pin, OEJ and OUTJ, are loaded into the capture registers. The CLOCK signals is supplied by the TAP controller's CLOCKDR output. The data retained in these registers consists of signals from normal device operation.



Shift & Update Phases

In the shift phase, the previously captured signals at the pin, OEJ and OUTJ, are shifted out of the boundary-scan register via the TDO pin using CLOCK. As data is shifted out, the patterns for the next test can be shifted in via the TDI pin.

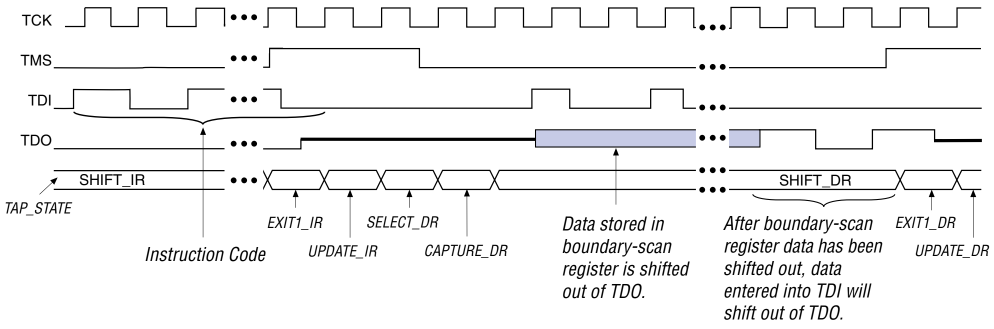
In the update phase, data is transferred from the capture to the UPDATE registers using the UPDATE clock. The data stored in the UPDATE registers can be used for the EXTEST instruction.



During the capture phase, multiplexers preceding the capture registers select the active device data signals. This data is then clocked into the capture registers. The multiplexers at the outputs of the update registers also select active device data to prevent functional interruptions to the device. During the shift phase, the boundary-scan shift register is formed by clocking data through capture registers around the device periphery and then out of the `TDO` pin. The device can simultaneously shift new test data into TDI and replace the contents of the capture registers. During the update phase, data in the capture registers is transferred to the update registers. This data can then be used in the EXTEST instruction mode. Refer to "EXTEST Instruction Mode" on page 9–12 for more information.

Figure 9–9 shows the SAMPLE/PRELOAD waveforms. The SAMPLE/PRELOAD instruction code is shifted in through the TDI pin. The TAP controller advances to the CAPTURE_DR state and then to the SHIFT_DR state, where it remains if TMS is held low. The data that was present in the capture registers after the capture phase is shifted out of the TDO pin. New test data shifted into the TDI pin appears at the TDO pin after being clocked through the entire boundary-scan register. Figure 9–9 shows that the instruction code at TDI does not appear at the TDO pin until after the capture register data is shifted out. If TMS is held high on two consecutive TCK clock cycles, the TAP controller advances to the UPDATE_DR state for the update phase.

Figure 9–9. SAMPLE/PRELOAD Shift Data Register Waveforms


TCK
TMS
TDI
TDO
SHIFT_IR
SHIFT_DR
TAP_STATE
Instruction Code
EXIT1_IR
UPDATE_IR
SELECT_DR
CAPTURE_DR
Data stored in boundary-scan register is shifted out of TDO.
After boundary-scan register data has been shifted out, data entered into TDI will shift out of TDO.
EXIT1_DR
UPDATE_DR

EXTEST Instruction Mode

The EXTEST instruction mode is used primarily to check external pin connections between devices. Unlike the SAMPLE/PRELOAD mode, EXTEST allows test data to be forced onto the pin signals. By forcing known logic high and low levels on output pins, opens and shorts can be detected at pins of any device in the scan chain.

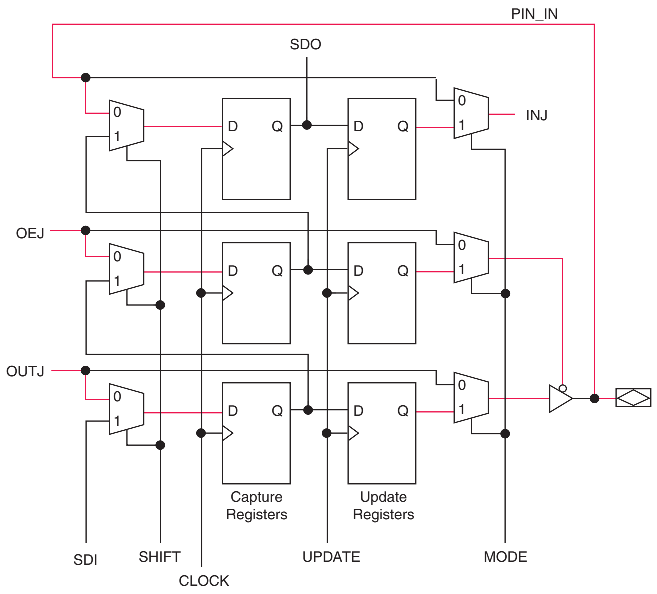
Figure 9–10 shows the capture, shift, and update phases of the EXTEST mode.

Figure 9–10. IEEE Std. 1149.1 BST EXTEST Mode

Capture Phase

In the capture phase, the signals at the pin, OEJ and OUTJ, are loaded into the capture registers. The CLOCK signals is supplied by the TAP controller's CLOCKDR output. Previously retained data in the update registers drive the PIN_IN, INJ, and allows the I/O pin to tri-state or drive a signal out.

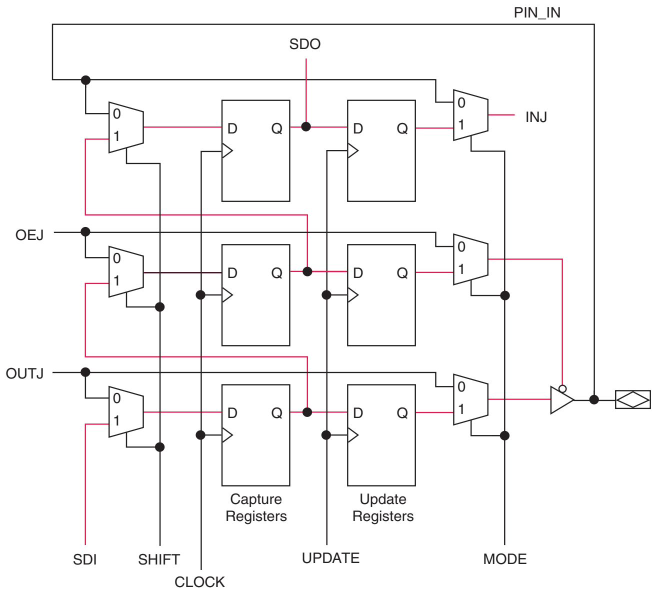
A "1" in the OEJ update register tri-states the output buffer.



Shift & Update Phases

In the shift phase, the previously captured signals at the pin, OEJ and OUTJ, are shifted out of the boundary-scan register via the TDO pin using CLOCK. As data is shifted out, the patterns for the next test can be shifted in via the TDI pin.

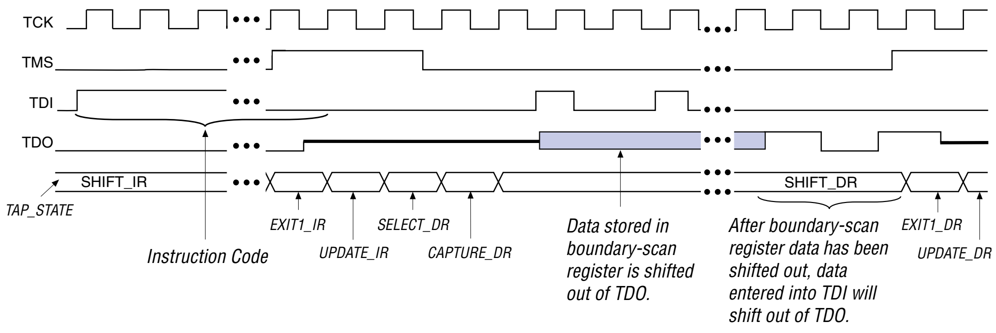
In the update phase, data is transferred from the capture registers to the update registers using the UPDATE clock. The update registers then drive the PIN_IN, INJ, and allow the I/O pin to tri-state or drive a signal out.



`EXTEST` selects data differently than `SAMPLE/PRELOAD`. `EXTEST` chooses data from the update registers as the source of the output and output enable signals. Once the `EXTEST` instruction code is entered, the multiplexers select the update register data. Thus, data stored in these registers from a previous EXTEST or SAMPLE/PRELOAD test cycle can be forced onto the pin signals. In the capture phase, the results of this test data are stored in the capture registers and then shifted out of TDO during the shift phase. New test data can then be stored in the update registers during the update phase.

The EXTEST waveform diagram in Figure 9–11 resembles the SAMPLE/PRELOAD waveform diagram, except for the instruction code. The data shifted out of TDO consists of the data that was present in the capture registers after the capture phase. New test data shifted into the TDI pin appears at the TDO pin after being clocked through the entire boundary-scan register.

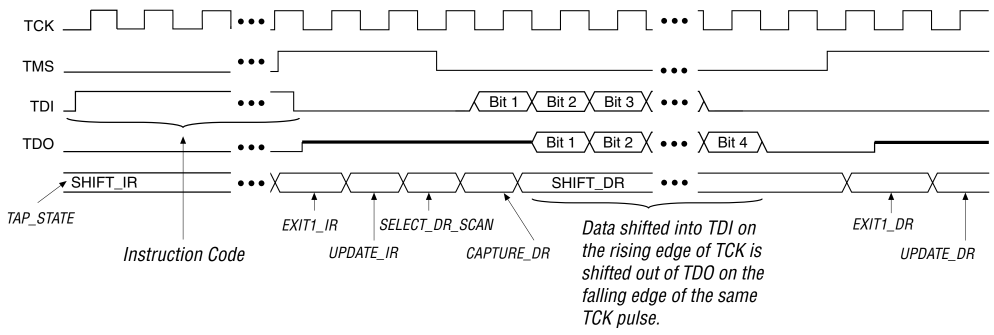
Figure 9–11. EXTEST Shift Data Register Waveforms


TCK
TMS
TDI
TDO
SHIFT_IR
TAP_STATE
Instruction Code
EXIT1_IR
UPDATE_IR
SELECT_DR
CAPTURE_DR
Data stored in boundary-scan register is shifted out of TDO.
SHIFT_DR
After boundary-scan register data has been shifted out, data entered into TDI will shift out of TDO.
EXIT1_DR
UPDATE_DR

BYPASS Instruction Mode

The BYPASS mode is activated when an instruction code of all ones is loaded in the instruction register. The waveforms in Figure 9–12 show how scan data passes through a device once the TAP controller is in the SHIFT_DR state. In this state, data signals are clocked into the bypass register from TDI on the rising edge of TCK and out of TDO on the falling edge of the same clock pulse.

Figure 9–12. BYPASS Shift Data Register Waveforms



IDCODE Instruction Mode

The `IDCODE` instruction mode is used to identify the devices in an IEEE Std. 1149.1 chain. When `IDCODE` is selected, the device identification register is loaded with the 32-bit vendor-defined identification code. The device ID register is connected between the `TDI` and `TDO` ports, and the device `IDCODE` is shifted out. The `IDCODE` for Stratix II devices are listed in the *Configuration & Testing* chapter of the *Stratix II Handbook*, *Volume 1*.

USERCODE Instruction Mode

The `USERCODE` instruction mode is used to examine the user electronic signature (UES) within the devices along an IEEE Std. 1149.1 chain. When this instruction is selected, the device identification register is connected between the `TDI` and `TDO` ports. The user-defined UES is shifted into the device ID register in parallel from the 32-bit `USERCODE` register. The UES is then shifted out through the device ID register. Note that the UES value will not be user defined until after the device has been configured. Before configuration, the UES value will be set to the default value.

CLAMP Instruction Mode

The `CLAMP` instruction mode is used to allow the state of the signals driven from the pins to be determined from the boundary-scan register while the bypass register is selected as the serial path between the `TDI` and `TDO` ports. The state of all signals driven from the pins will be completely defined by the data held in the boundary-scan register.

If you are testing after configuring the device, the programmable weak pull-up resister or the bus hold feature will override the `CLAMP` value (the value stored in the update register of the boundary-scan cell) at the pin.

HIGHZ Instruction Mode

The HIGHZ instruction mode is used to set all of the user I/O pins to an inactive drive state. These pins are tri-stated until a new JTAG instruction is executed. When this instruction is loaded into the instruction register, the bypass register is connected between the TDI and TDO ports.

If you are testing after configuring the device, the programmable weak pull-up resistor or the bus hold feature will override the HIGHZ value at the pin.

I/O Voltage Support in JTAG Chain

There can be several different devices in a JTAG chain. However, the user should be cautious if the chain contains devices that have different V_{CCIO} levels. The output voltage level of the TDO pin must meet the specifications of the TDI pin it drives. The TDI pin is powered by V_{CCPD} (3.3 V). For Stratix II devices, the TDO pin is powered by the V_{CCIO} power supply of bank 4. The TDI pin is powered by V_{CCPD} (3.3 V). See Table Table 9–3 for board design recommendations to ensure proper JTAG chain operation.

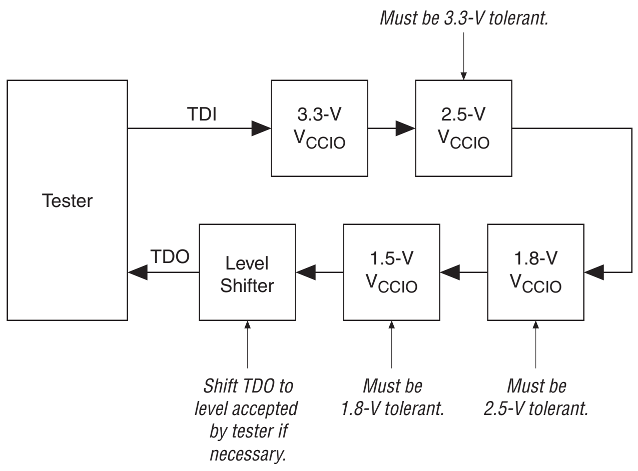
Table 9–3. Supported TDO/TDI Voltage Combinations

Device	TDI Input Buffer Power	Stratix II TDO V_{CCIO} Voltage Level in I/O Bank 4			
		V_{CCIO} = 3.3 V	V_{CCIO} = 2.5 V	V_{CCIO} = 1.8 V	V_{CCIO} = 1.5 V
Stratix II	Always V_{CCPD} (3.3V)	✓ (1)	✓ (2)	✓ (3)	level shifter required
Non-Stratix II	VCC = 3.3 V	✓ (1)	✓ (2)	✓ (3)	level shifter required
	VCC = 2.5 V	✓ (1), (4)	✓ (2)	✓ (3)	level shifter required
	VCC = 1.8 V	✓ (1), (4)	✓ (2), (5)	✓	level shifter required
	VCC = 1.5 V	✓ (1), (4)	✓ (2), (5)	✓ (6)	✓

Notes to Table 9–3:
(1) The TDO output buffer meets V_{OH} (MIN) = 2.4 V.
(2) The TDO output buffer meets V_{OH} (MIN) = 2.0 V.
(3) An external 250-Ω pull-up resistor is not required, but recommended if signal levels on the board are not optimal.
(4) Input buffer must be 3.3-V tolerant.
(5) Input buffer must be 2.5-V tolerant.
(6) Input buffer must be 1.8-V tolerant.

You can interface the TDI and TDO lines of the devices that have different V_{CCIO} levels by inserting a level shifter between the devices. If possible, the JTAG chain should be built such that a device with a higher V_{CCIO} level drives to a device with an equal or lower V_{CCIO} level. This way, a level shifter may be required only to shift the TDO level to a level acceptable to the JTAG tester. Figure 9–13 shows the JTAG chain of mixed voltages and how a level shifter is inserted in the chain.

Figure 9–13. JTAG Chain of Mixed Voltages



Using IEEE Std. 1149.1 BST Circuitry

Stratix II devices have dedicated JTAG pins and the IEEE Std. 1149.1 BST circuitry is enabled upon device power-up. Not only can you perform BST on Stratix II FPGAs before and after, but also during configuration. Stratix II FPGAs support the BYPASS, IDCODE and SAMPLE instructions during configuration without interrupting configuration. To send all other JTAG instructions, you must interrupt configuration using the CONFIG_IO instruction.

The CONFIG_IO instruction allows you to configure I/O buffers via the JTAG port, and when issued, interrupts configuration. This instruction allows you to perform board-level testing prior to configuring the Stratix II FPGA or waiting for a configuration device to complete configuration. Once configuration has been interrupted and JTAG BST is complete, the part must be reconfigured via JTAG (PULSE_CONFIG instruction) or by pulsing nCONFIG low.

When you perform JTAG boundary-scan testing before configuration, the nCONFIG pin must be held low.

The chip-wide reset (`DEV_CLRn`) and chip-wide output enable (`DEV_OE`) pins on Stratix II devices do not affect JTAG boundary-scan or configuration operations. Toggling these pins does not disrupt BST operation (other than the expected BST behavior).

When designing a board for JTAG configuration of Stratix II devices, the connections for the dedicated configuration pins need to be considered. For more information on using the IEEE Std.1149.1 circuitry for device configuration, see the *Configuring Stratix II Devices* chapter of the *Stratix II Handbook, Volume 2*.

Disabling IEEE Std. 1149.1 BST Circuitry

The IEEE Std. 1149.1 BST circuitry for Stratix II devices is enabled upon device power-up. Because this circuitry may be used for BST or in-circuit reconfiguration, this circuitry must be enabled only at specific times as mentioned in the section, "Using IEEE Std. 1149.1 BST Circuitry".

If the IEEE Std. 1149.1 circuitry will not be utilized at any time, the circuitry should be permanently disabled. Table 9–4 shows the pin connections necessary for disabling the IEEE Std. 1149.1 circuitry in Stratix II devices to ensure that the circuitry is not inadvertently enabled when it is not needed.

Table 9–4. Disabling IEEE Std. 1149.1 Circuitry

JTAG Pins (1)	Connection for Disabling
`TMS`	V_{CC}
`TCK`	GND
`TDI`	V_{CC}
`TDO`	Leave open
`TRST`	GND

Note to Table 9–4:
(1) There is no software option to disable JTAG in Stratix II devices. The JTAG pins are dedicated.

Guidelines for IEEE Std. 1149.1 Boundary-Scan Testing

Use the following guidelines when performing boundary-scan testing with IEEE Std. 1149.1 devices:

- If the "10..." pattern does not shift out of the instruction register via the `TDO` pin during the first clock cycle of the `SHIFT_IR` state, the TAP controller has not reached the proper state. To solve this problem, try one of the following procedures:

- Verify that the TAP controller has reached the SHIFT_IR state correctly. To advance the TAP controller to the SHIFT_IR state, return to the RESET state and send the code 01100 to the TMS pin.
- Check the connections to the V_{CC}, GND, JTAG, and dedicated configuration pins on the device.

- Perform a SAMPLE/PRELOAD test cycle prior to the first EXTEST test cycle to ensure that known data is present at the device pins when the EXTEST mode is entered. If the OEJ update register contains a 0, the data in the OUTJ update register will be driven out. The state must be known and correct to avoid contention with other devices in the system.
- Do not perform EXTEST testing during ICR. This instruction is supported before or after ICR, but not during ICR. Use the CONFIG_IO instruction to interrupt configuration and then perform testing, or wait for configuration to complete.
- If performing testing before configuration, hold nCONFIG pin low.
- After configuration, any pins in a differential pin pair cannot be tested. Therefore, performing BST after configuration requires editing of BSC group definitions that correspond to these differential pin pairs. The BSC group should be redefined as an internal cell. See the BSDL file for more information on editing.

For more information on boundary scan testing, contact Altera Applications.

Boundary-Scan Description Language (BSDL) Support

The Boundary-Scan Description Language (BSDL), a subset of VHDL, provides a syntax that allows you to describe the features of an IEEE Std. 1149.1 BST-capable device that can be tested. Test software development systems then use the BSDL files for test generation, analysis, and failure diagnostics. For more information, or to receive BSDL files for IEEE Std. 1149.1-compliant Stratix II devices, visit the Altera web site at **www.altera.com**.

Conclusion

The IEEE Std. 1149.1 BST circuitry available in Stratix II devices provides a cost-effective and efficient way to test systems that contain devices with tight lead spacing. Circuit boards with Altera and other IEEE Std. 1149.1-compliant devices can use the EXTEST, SAMPLE/PRELOAD, and BYPASS modes to create serial patterns that internally test the pin connections between devices and check device operation.

References

Bleeker, H., P. van den Eijnden, and F. de Jong. *Boundary-Scan Test: A Practical Approach*. Eindhoven, The Netherlands: Kluwer Academic Publishers, 1993.

Institute of Electrical and Electronics Engineers, Inc. *IEEE Standard Test Access Port and Boundary-Scan Architecture* (IEEE Std 1149.1-2001). New York: Institute of Electrical and Electronics Engineers, Inc., 2001.

Maunder, C. M., and R. E. Tulloss. *The Test Access Port and Boundary-Scan Architecture*. Los Alamitos: IEEE Computer Society Press, 1990.

Section VI. PCB Layout Guidelines

This section provides information for board layout designers to successfully layout their boards for Stratix® II devices. These chapters contain the required PCB layout guidelines and package specifications.

This section contains the following chapters:

Revision History

The table below shows the revision history for Chapters 10 through 11.

Chapter	Date / Version	Changes Made
10	January 2005, v2.0	Updated Table 10–2.
	October 2004, v1.1	Updated Tables 10–1 and 10–2.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.
11	March 2005, v1.2	Minor content updates.
	January 2005, v1.1	This chapter was formally chapter 12.
	February 2004, v1.0	Added document to the Stratix II Device Handbook.

10. Package Information for Stratix II Devices

SII52010-2.0

Introduction

This chapter provides package information for Altera® Stratix® II devices, including:

- Device and package cross reference
- Thermal resistance values
- Package outlines

Table 10–1 shows which Altera Stratix II devices are available in FineLine BGA® packages.

Table 10–1. Stratix II Devices in FineLine BGA Packages

Device	Package	Pins
EP2S15	Thermally enhanced flip-chip FineLine BGA	484
	Thermally enhanced flip-chip FineLine BGA	672
EP2S30	Thermally enhanced flip-chip FineLine BGA	484
	Thermally enhanced flip-chip FineLine BGA	672
EP2S60	Thermally enhanced flip-chip FineLine BGA	484
	Thermally enhanced flip-chip FineLine BGA	672
	Thermally enhanced flip-chip FineLine BGA	1,020
EP2S90	Thermally enhanced flip-chip Hybrid FineLine BGA	484
	Thermally enhanced flip-chip FineLine BGA	780
	Thermally enhanced flip-chip FineLine BGA	1,020
	Thermally enhanced flip-chip FineLine BGA	1,508
EP2S130	Thermally enhanced flip-chip FineLine BGA	780
	Thermally enhanced flip-chip FineLine BGA	1,020
	Thermally enhanced flip-chip FineLine BGA	1,508
EP2S180	Thermally enhanced flip-chip FineLine BGA	1,020
	Thermally enhanced flip-chip FineLine BGA	1,508

Thermal Resistance

Table 10–2 provides θ_{JA} (junction-to-ambient thermal resistance) and θ_{JC} (junction-to-case thermal resistance) values for Stratix II devices.

Table 10–2. Thermal Resistance of Stratix II Devices

Device	Pin Count	Package	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.	θ_{JC} (° C/W)
EP2S15	484	FineLine BGA	13.1	11.1	9.6	8.3	0.36
	672	FineLine BGA	12.2	10.2	8.8	7.6	0.36
EP2S30	484	FineLine BGA	12.6	10.6	9.1	7.9	0.21
	672	FineLine BGA	11.7	9.7	8.3	7.1	0.21
EP2S60	484	FineLine BGA	12.3	10.3	8.8	7.5	0.13
	672	FineLine BGA	11.4	9.4	7.8	6.7	0.13
	1,020	FineLine BGA	10.4	8.4	7.0	5.9	0.13
EP2S90	484	Hybrid FineLine BGA	12.0	9.9	8.3	7.1	0.07
	780	FineLine BGA	11.0	8.8	7.3	6.1	0.09
	1,020	FineLine BGA	10.2	8.2	6.8	5.7	0.10
	1,508	FineLine BGA	9.4	7.4	6.1	5.0	0.10
EP2S130	780	FineLine BGA	10.9	8.7	7.2	6.0	0.07
	1,020	FineLine BGA	10.1	8.1	6.7	5.5	0.07
	1,508	FineLine BGA	9.3	7.3	6.0	4.8	0.07
EP2S180	1,020	FineLine BGA	9.9	7.9	6.5	5.4	0.05
	1,508	FineLine BGA	9.1	7.1	5.8	4.7	0.05

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 Thermally Enhanced FineLine BGA Packaging

- All dimensions and tolerances conform to American National Standards Institute (ANSI) Y14.5M – 1994.
- Controlling dimension is in millimeters.
- Some devices have a chamfered corner at the A-1 ball location.
- M is the maximum solder ball matrix size.

Tables 10–3 and 10–4 show package information and package outline figure references, respectively, for the 484-pin thermally enhanced FineLine BGA packaging.

Table 10–3. Package Information for 484-Pin Thermally Enhanced FineLine BGA Packaging

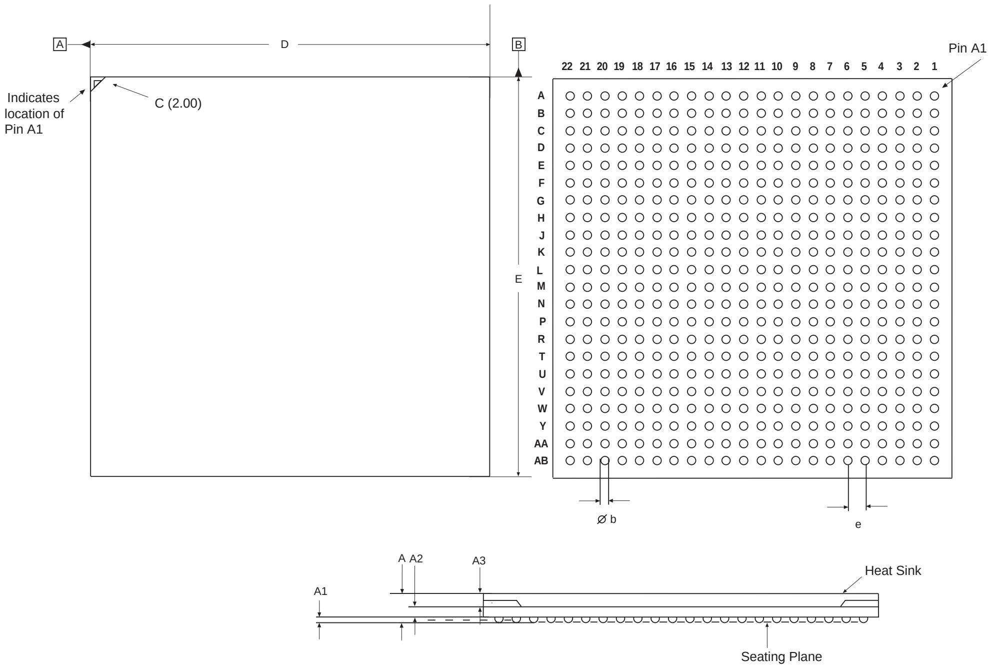
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Lead material	Tin-lead alloy (63/37)
Lead finish	N/A
JEDEC outline	MS-034
JEDEC option	AAJ-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	5.7 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 10–4. Package Outline Figure References for 484-Pin Thermally Enhanced FineLine BGA Packaging

Symbol	Min. (mm)	Nom. (mm)	Max. (mm)
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
D/E	23.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		
M	22		

Figure 10–1 shows a package outline for the 484-pin thermally enhanced FineLine BGA packaging.

Figure 10–1. Package Outline for 484-Pin Thermally Enhanced FineLine BGA Packaging



672-Pin Thermally Enhanced FineLine BGA Packaging

- All dimensions and tolerances conform to ANSI Y14.5M – 1994.
- Controlling dimension is in millimeters.
- Orientation of the package is shown by a chamfer and/or a pin 1 mark.

Tables 10–5 and 10–6 show package information and package outline figure references, respectively, for the 672-pin thermally enhanced FineLine BGA packaging.

Table 10–5. Package Information for 672-Pin Thermally Enhanced FineLine BGA Packaging

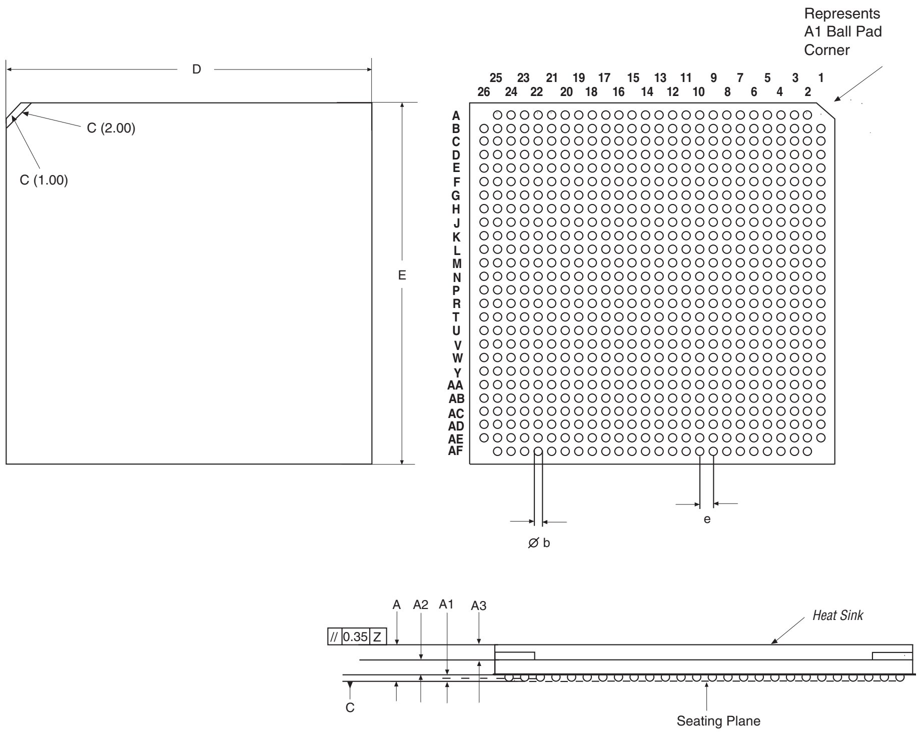
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Lead material	Tin-lead alloy (63/37)
Lead finish	N/A
JEDEC outline	MS-034
JEDEC option	AAL-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	7.7 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 10–6. Package Outline Figure References for 672-Pin Thermally Enhanced FineLine BGA Packaging

Symbol	Min. (mm)	Nom. (mm)	Max. (mm)
A	–	–	3.50
A1	0.30	–	–
A2	0.25	–	3.00
A3	–	–	2.50
b	0.50	0.60	0.70
e	1.00 BSC		
D/E	27		
M	26		

Figure 10–2 shows a package outline for the 672-pin thermally enhanced FineLine BGA packaging.

Figure 10–2. Package Outline for 672-Pin Thermally Enhanced FineLine BGA Packaging



1,020-Pin Thermally Enhanced FineLine BGA Packaging

- All dimensions and tolerances conform to ANSI Y14.5M – 1994.
- Controlling dimension is in millimeters.
- Orientation of the package is shown by a chamfer and/or a pin 1 mark.
- M is the maximum solder ball matrix size.

Tables 10–7 and 10–8 show package information and package outline figure references, respectively, for the 1,020-pin thermally enhanced FineLine BGA packaging.

Table 10–7. Package Information for 1,020-Pin Thermally Enhanced FineLine BGA Packaging

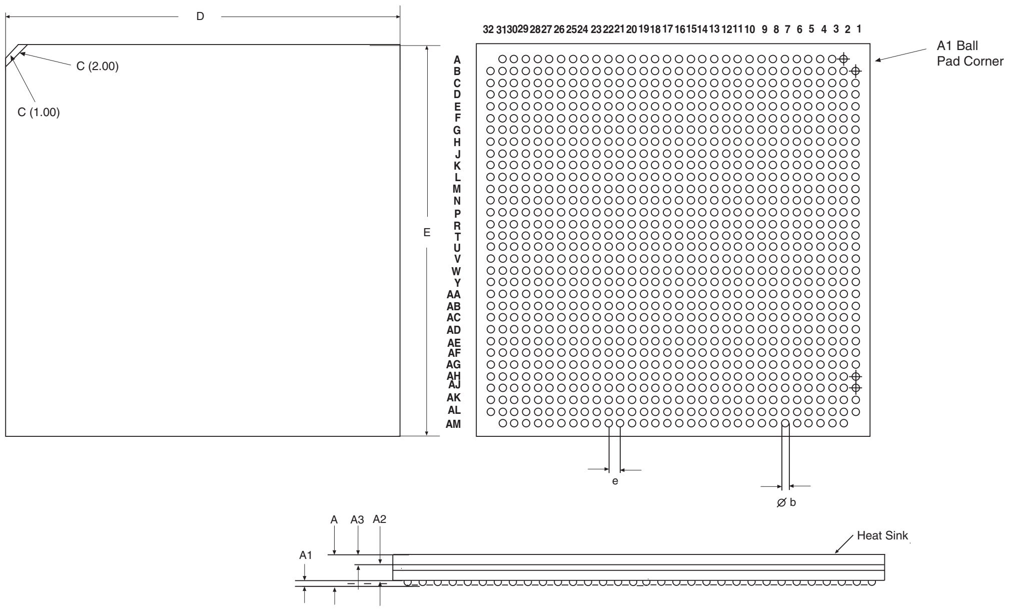
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Lead material	Tin-lead alloy (63/37)
Lead finish	N/A
JEDEC outline	MS-034
JEDEC option	AAP-1, depopulated
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	11.5 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 10–8. Package Outline Figure References for 1,020-Pin Thermally Enhanced FineLine BGA Packaging

Symbol	Min. (mm)	Nom. (mm)	Max. (m)
A	–	–	3.50
A1	0.35	–	–
A2	0.25	–	3.00
A3	–	–	2.50
b	0.50	0.60	0.70
e	1.00 BSC		
D/E	33.00 BSC		
M	32		

Figure 10–3 shows a package outline for the 1,020-pin thermally enhanced FineLine BGA packaging.

Figure 10–3. Package Outline for 1,020-Pin Thermally Enhanced FineLine BGA Packaging



1,508-Pin Thermally Enhanced FineLine BGA Packaging

- All dimensions and tolerances conform to ANSI Y14.5M – 1994.
- Controlling dimension is in millimeters.
- Orientation of the package is shown by a chamfer and/or a pin 1 mark.
- M is the maximum solder ball matrix size.

Tables 10–9 and 10–10 show package information and package outline figure references, respectively, for the 1,508-pin thermally enhanced FineLine BGA packaging.

Table 10–9. Package Information for 1,508-Pin Thermally Enhanced FineLine BGA Packaging

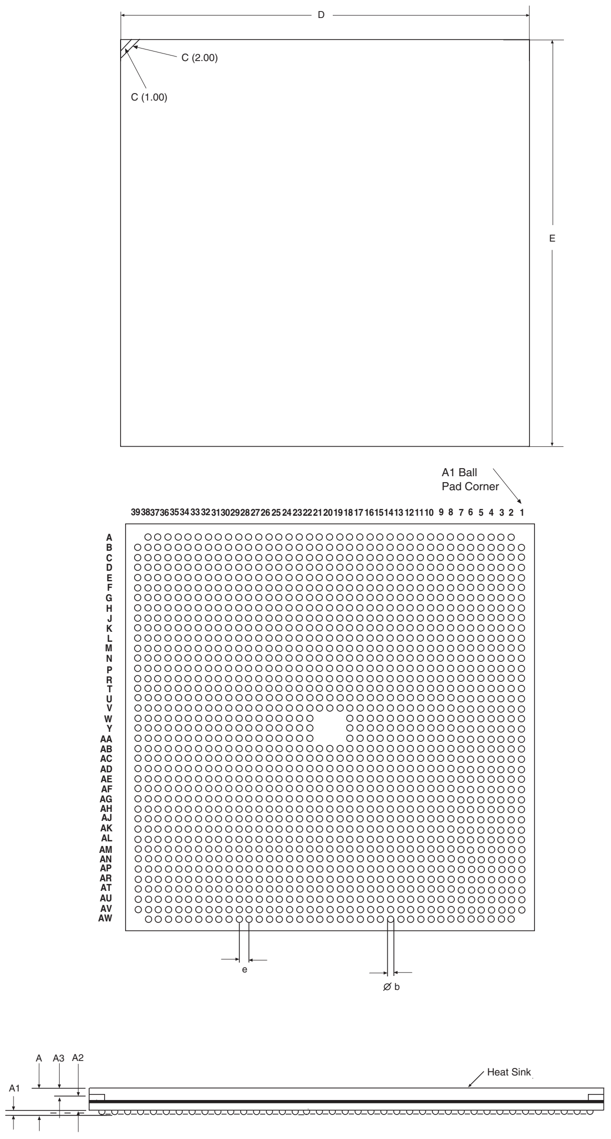
Description	Specification
Ordering code reference	F
Package acronym	FineLine BGA
Lead material	Tin-lead alloy (63/37)
Lead finish	N/A
JEDEC outline	MS-034
JEDEC option	AAU-1
Maximum lead coplanarity	0.008 inches (0.20 mm)
Weight	15.3 g
Moisture sensitivity level	Printed on moisture barrier bag

Table 10–10. Package Outline Figure References for 1,508-Pin Thermally Enhanced FineLine BGA Packaging

Symbol	Min. (mm)	Nom. (mm)	Max. (mm)
A	–	–	3.50
A1	0.35	–	–
A2	0.25	–	3.00
A3	–	–	2.50
b	0.50	0.60	0.70
e	1.00 BSC		
D/E	40.00 BSC		
M	39		

Figure 10–4 shows a package outline for the 1,508-pin thermally enhanced FineLine BGA packaging.

Figure 10–4. Package Outline for 1,508-Pin Thermally Enhanced FineLine BGA Packaging



11. High-Speed Board Layout Guidelines

SII52012-1.2

Introduction

Printed circuit board (PCB) layout becomes more complex as device pin density and system frequency increase. A successful high-speed board must effectively integrate devices and other elements while avoiding signal transmission problems associated with high-speed I/O standards. Because Altera® devices include a variety of high-speed features, including fast I/O pins and edge rates less than one hundred picoseconds, it is imperative that an effective design successfully:

- Reduces system noise by filtering and evenly distributing power to all devices
- Terminates the signal line to diminish signal reflection
- Minimizes crosstalk between parallel traces
- Reduces the effects of ground bounce
- Matches impedance

This chapter provides guidelines for effective high-speed board design using Altera devices and discusses the following issues:

- PCB material selection
- Transmission line layouts
- Routing schemes for minimizing crosstalk and maintaining signal integrity
- Termination schemes
- Simultaneous switching noise (SSN)
- Electromagnetic interference (EMI)
- Additional FPGA-specific board design/signal integrity information

PCB Material Selection

Fast edge rates contribute to noise and crosstalk, depending on the PCB dielectric construction material. Dielectric material can be assigned a dielectric constant (ε_r) that is related to the force of attraction between two opposite charges separated by a distance in a uniform medium as follows:

$$F = \frac{Q_1 Q_2}{4\pi\varepsilon r^2}$$

where:

Q_1, Q_2 = charges
r = distance between the charges (m)
F = force (N)
ε = permittivity of dielectric (F/m).

Each PCB substrate has a different relative dielectric constant. The dielectric constant is the ratio of the permittivity of a substance to that of free space, as follows:

$$\varepsilon_r = \frac{\varepsilon}{\varepsilon_o}$$

where:

ε_r = dielectric constant
ε_o = permittivity of empty space (F/m)
ε = permittivity (F/m)

The dielectric constant compares the effect of an insulator on the capacitance of a conductor pair, with the capacitance of the conductor pair in a vacuum. The dielectric constant affects the impedance of a transmission line. Signals can propagate faster in materials that have a lower dielectric constant.

A high-frequency signal that propagates through a long line on the PCB from driver to receiver is severely affected by the loss tangent of the dielectric material. A large loss tangent means higher dielectric absorption.

The most widely used dielectric material for PCBs is FR-4, a glass laminate with epoxy resin that meets a wide variety of processing conditions. The dielectric constant for FR-4 is between 4.1 and 4.5. GETEK is another material that can be used in high-speed boards. GETEK is composed of epoxy and resin (polyphenylene oxide) and has a dielectric constant between 3.6 and 4.2.

Table 11–1 shows the loss tangent value for FR-4 and GETEK materials.

Table 11–1. Loss Tangent Value of FR-4 & GETEK Materials

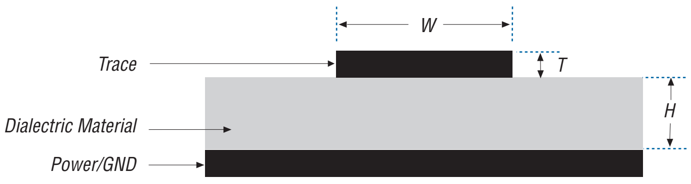
Manufacturer	Material	Loss Tangent Value
GE Electromaterials	GETEK	0.010 @ 1 MHz
Isola Laminate Systems	FR-4	0.019 @ 1 MHz

Transmission Line Layout

The transmission line is a trace and has a distributed mixture of resistance (R), inductance (L), and capacitance (C). There are two types of transmission line layouts, microstrip and stripline.

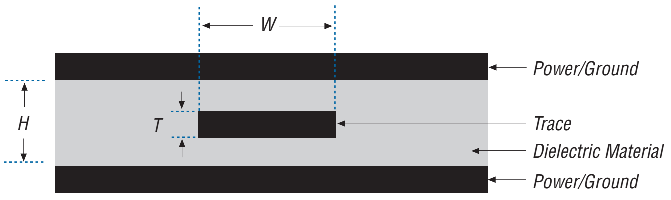
Figure 11–1 shows a microstrip transmission line layout, which refers to a trace routed as the top or bottom layer of a PCB and has one voltage-reference plane (power or ground). Figure 11–2 shows a stripline transmission line layout, which uses a trace routed on the inside layer of a PCB and has two voltage-reference planes (power and/or ground).

Figure 11–1. Microstrip Transmission Line Layout Note (1)



Note to Figure 11–1:
(1) *W* = width of trace, *T* = thickness of trace, and *H* = height between trace and reference plane.

Figure 11–2. Stripline Transmission Line Layout Note (1)



Note to Figure 11–2:
(1) *W* = width of trace, *T* = thickness of trace, and *H* = height between trace and two reference planes.

Impedance Calculation

Any circuit trace on the PCB has characteristic impedance associated with it. This impedance is dependent on the width (*W*) of the trace, the thickness (*T*) of the trace, the dielectric constant of the material used, and the height (*H*) between the trace and reference plane.

Microstrip Impedance

A circuit trace routed on an outside layer of the PCB with a reference plane (GND or V_{CC}) below it, constitutes a microstrip layout. Use the following microstrip impedance equation to calculate the impedance of a microstrip trace layout:

$$Z_0 = \frac{87}{\sqrt{\varepsilon_r + 1.41}} \ln\left(\frac{5.98 \times H}{0.8W + T}\right) \Omega$$

Using typical values of $W = 8$ mil, $H = 5$ mil, $T = 1.4$ mil, the dielectric constant, and (FR-4) = 4.1, with the microstrip impedance equation, solving for microstrip impedance (Z_o) yields:

$$Z_0 = \frac{87}{\sqrt{4.1 + 1.41}} \ln\left(\frac{5.98 \times (5)}{0.8(8) + 1.4}\right) \Omega$$

$$Z_0 \sim 50\ \Omega$$

☞ The measurement unit in the microstrip impedance equation is mils (i.e., 1 mil = 0.001 inches). Also, copper (Cu) trace thickness is usually measured in ounces (i.e., 1 oz = 1.4 mil).

Figure 11–3 shows microstrip trace impedance with changing trace width (*W*), using the values in the microstrip impedance equation, keeping dielectric height and trace thickness constant.

Figure 11–3. Microstrip Trace Impedance with Changing Trace Width

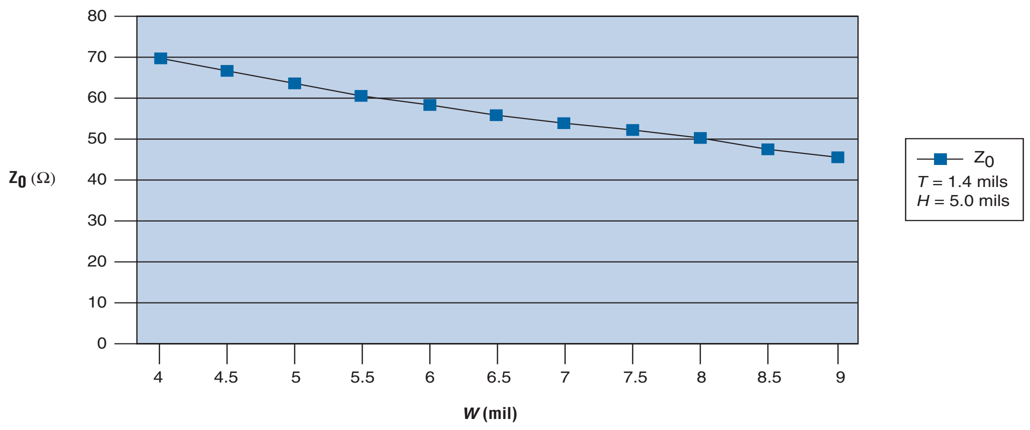
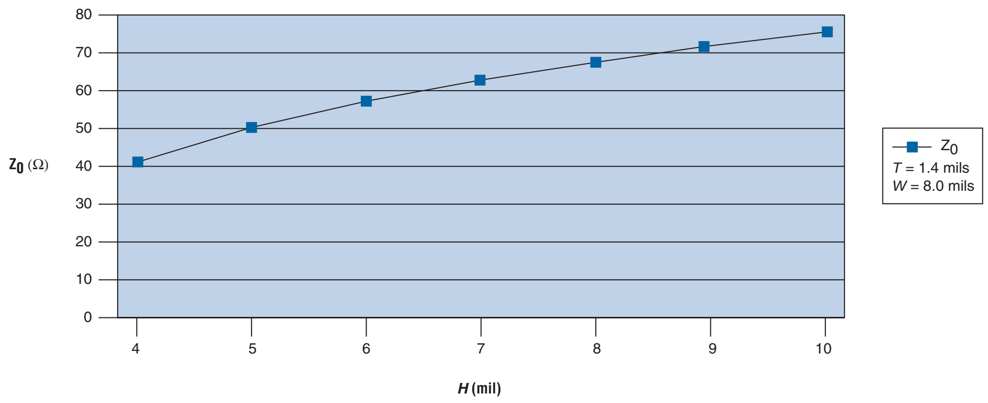


Figure 11–4 shows microstrip trace impedance with changing height, using the values in the microstrip impedance equation, keeping trace width and trace thickness constant.

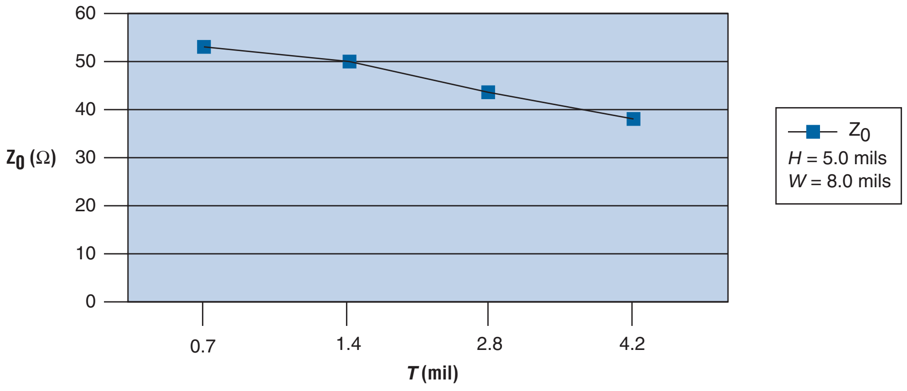
Figure 11–4. Microstrip Trace Impedance with Changing Height



The impedance graphs show that the change in impedance is inversely proportional to trace width and directly proportional to trace height above the ground plane.

Figure 11–5 plots microstrip trace impedance with changing trace thickness using the values in the microstrip impedance equation, keeping trace width and dielectric height constant. Figure 11–5 shows that as trace thickness increases, trace impedance decreases.

Figure 11–5. Microstrip Trace Impedance with Changing Trace Thickness



Stripline Impedance

A circuit trace routed on the inside layer of the PCB with two low-voltage reference planes (power and/or GND) constitutes a stripline layout. You can use the following stripline impedance equation to calculate the impedance of a stripline trace layout:

$$Z_o = \frac{60}{\sqrt{\varepsilon_r}} \ln\left(\frac{4H}{0.67\pi\,(T + 0.8W)}\right)\ \Omega$$

Using typical values of $W = 9$ mil, $H = 24$ mil, $T = 1.4$ mil, dielectric constant and (FR-4) = 4.1 with the stripline impedance equation and solving for stripline impedance (Z_o) yields:

$$Z_o = \frac{60}{\sqrt{4.1}} \ln\left(\frac{4\,(24)}{0.67\pi\,(1.4) + 0.8(9)}\right)\ \Omega$$

$$Z_o \sim 50\ \Omega$$

Figure 11–6 shows impedance with changing trace width using the stripline impedance equation, keeping height and thickness constant for stripline trace.

Figure 11–6. Stripline Trace Impedance with Changing Trace Width

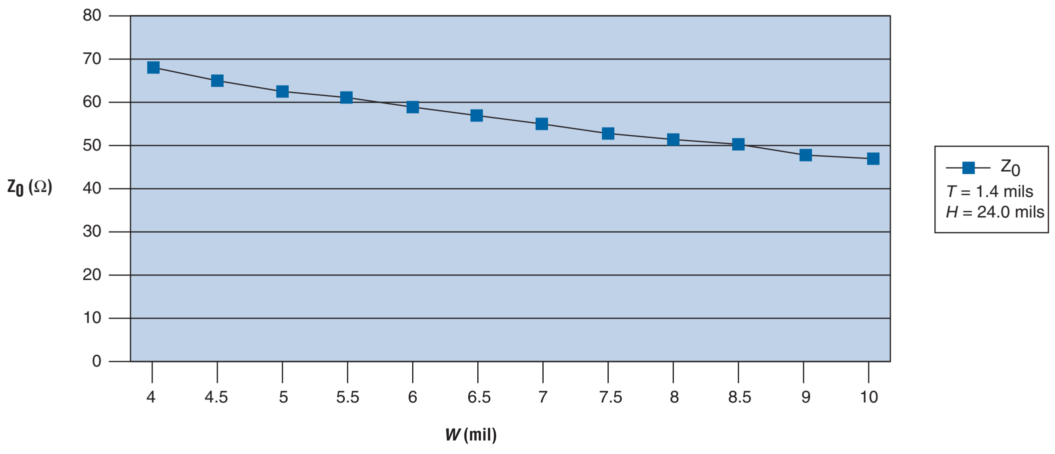
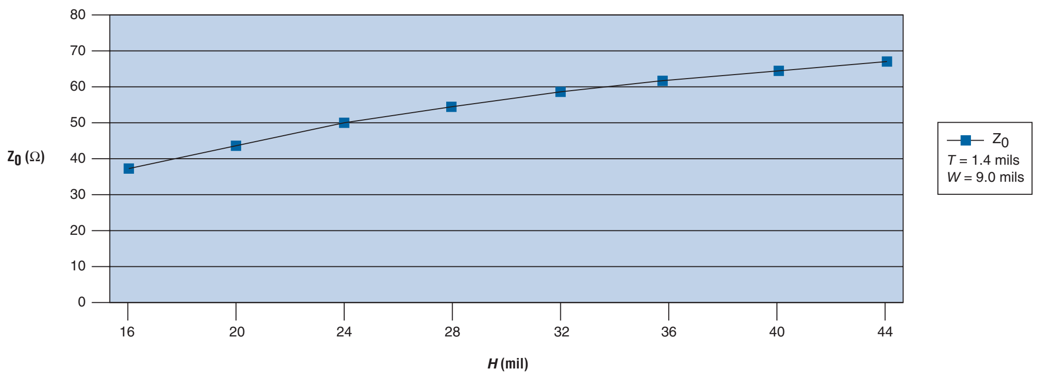


Figure 11–7 shows stripline trace impedance with changing dielectric height using the stripline impedance equation, keeping trace width and trace thickness constant.

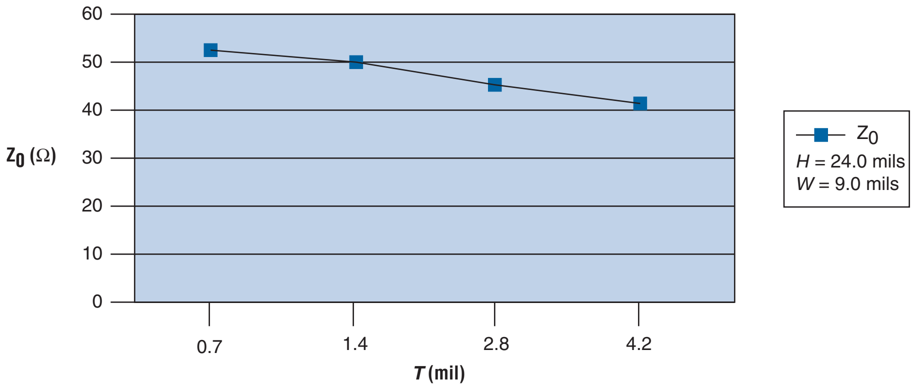
Figure 11–7. Stripline Trace Impedance with Changing Dielectric Height



As with the microstrip layout, the stripline layout impedance also changes inversely proportional to line width and directly proportional to height. However, the rate of change with trace height above GND is much slower in a stripline layout compared with a microstrip layout. A stripline layout has a signal sandwiched by FR-4 material, whereas a microstrip layout has one conductor open to air. This exposure causes a higher effective dielectric constant in stripline layouts compared with microstrip layouts. Thus, to achieve the same impedance, the dielectric span must be greater in stripline layouts compared with microstrip layouts. Therefore, stripline-layout PCBs with controlled impedance lines are thicker than microstrip-layout PCBs.

Figure 11–8 shows stripline trace impedance with changing trace thickness, using the stripline impedance equation, keeping trace width and dielectric height constant. Figure 11–8 shows that the characteristic impedance decreases as the trace thickness increases.

Figure 11–8. Stripline Trace Impedance with Changing Trace Thickness



Propagation Delay

Propagation delay (t_{PD}) is the time required for a signal to travel from one point to another. Transmission line propagation delay is a function of the dielectric constant of the material.

Microstrip Layout Propagation Delay

You can use the following equation to calculate the microstrip trace layout propagation delay:

$$t_{PD}\text{ (microstrip)} = 85\sqrt{0.475\varepsilon_r + 0.67}$$

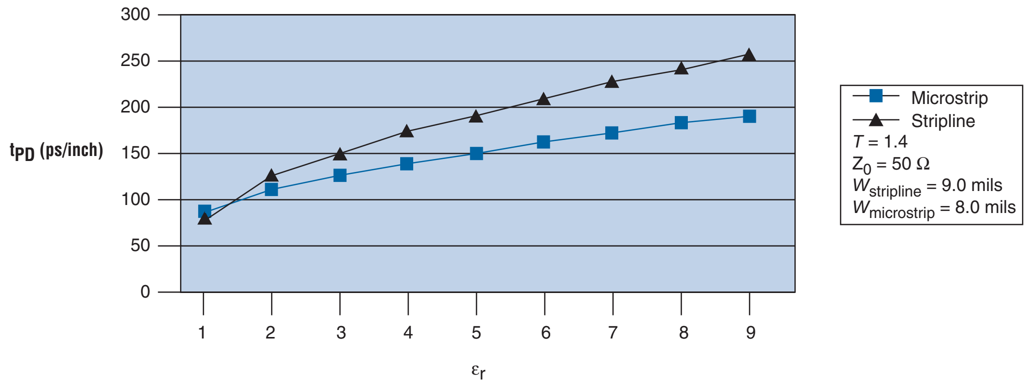
Stripline Layout Propagation Delay

You can use the following equation to calculate the stripline trace layout propagation delay.

$$t_{PD}\text{ (stripline)} = 85\sqrt{\varepsilon_r}$$

Figure 11–9 shows the propagation delay versus the dielectric constant for microstrip and stripline traces. As the dielectric constant increases, the propagation delay also increases.

Figure 11–9. Propagation Delay Versus Dielectric Constant for Microstrip & Stripline Traces



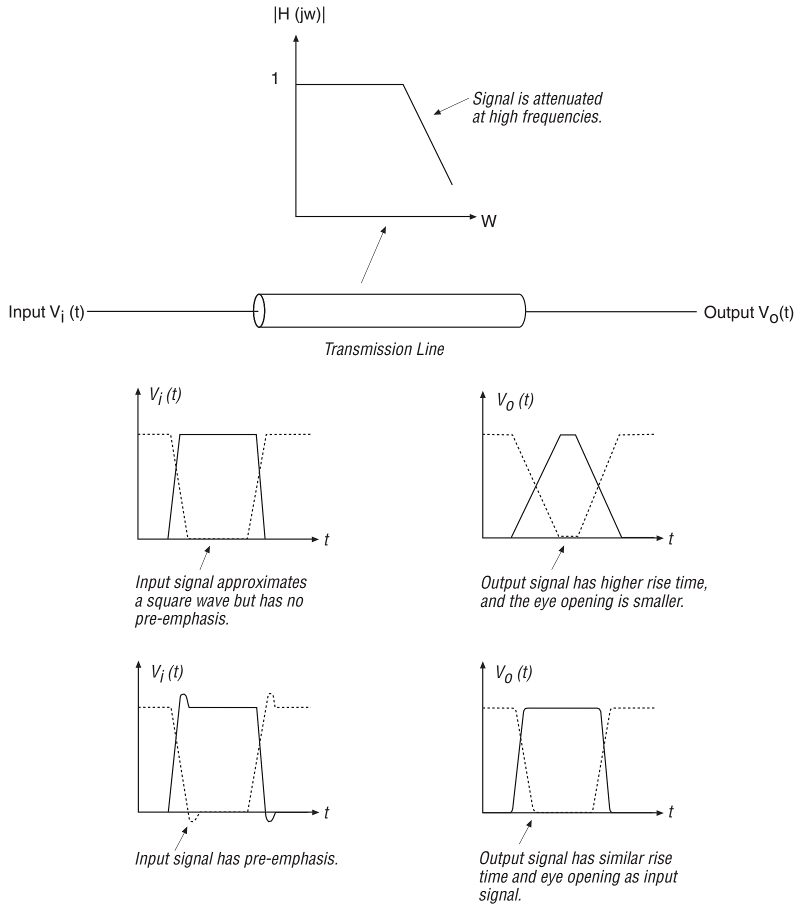
Pre-Emphasis

Typical transmission media like copper trace and coaxial cable have low-pass characteristics, so they attenuate higher frequencies more than lower frequencies. A typical digital signal that approximates a square wave contains high frequencies near the switching region and low frequencies in the constant region. When this signal travels through low-pass media, its higher frequencies are attenuated more than the lower frequencies, resulting in increased signal rise times. Consequently, the eye opening narrows and the probability of error increases.

The high-frequency content of a signal is also degraded by what is called the "skin effect." The cause of skin effect is the high-frequency current that flows primarily on the surface (skin) of a conductor. The changing current distribution causes the resistance to increase as a function of frequency.

You can use pre-emphasis to compensate for the skin effect. By Fourier analysis, a square wave signal contains an infinite number of frequencies. The high frequencies are located in the low-to-high and high-to-low transition regions and the low frequencies are located in the flat (constant) regions. Increasing the signal's amplitude near the transition region emphasizes higher frequencies more than the lower frequencies. When this pre-emphasized signal passes through low-pass media, it will come out with minimal distortion, if you apply the correct amount of pre-emphasis (see Figure 11–10).

Figure 11–10. Input & Output Signals with & without Pre-Emphasis


|H (jw)|
1
Signal is attenuated at high frequencies.
W
Input V_i (t)
Output V_o(t)
Transmission Line
V_i (t)
t
Input signal approximates a square wave but has no pre-emphasis.
V_O (t)
t
Output signal has higher rise time, and the eye opening is smaller.
V_i (t)
t
Input signal has pre-emphasis.
V_O (t)
t
Output signal has similar rise time and eye opening as input signal.

Stratix® II and Stratix GX devices provide programmable pre-emphasis to compensate for variable lengths of transmission media. You can set the pre-emphasis to between 5 and 25%, depending on the value of the output differential voltage (V_{OD}) in the Stratix GX device. Table 11–2 shows the available Stratix GX programmable pre-emphasis settings.

Table 11–2. Programmable Pre-Emphasis with Stratix GX Devices

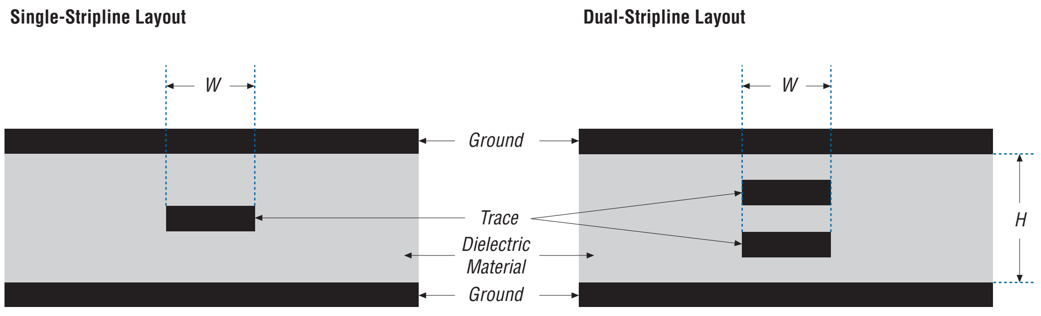
V_{OD}	Pre-emphasis Setting (%)				
	5	10	15	20	25
400	420	440	460	480	500
480	504	528	552	576	600
600	630	660	690	720	750
800	840	880	920	960	1,000
960	1,008	1,056	1,104	1,152	1,200
1,000	1,050	1,100	1,150	1,200	1,250
1,200	1,260	1,320	1,380	1,440	1,500
1,400	1,470	1,540	-	-	-
1,440	1,512	1,584	-	-	-
1,500	1,575	-	-	-	-
1,600	-	-	-	-	-

Routing Schemes for Minimizing Crosstalk & Maintaining Signal Integrity

Crosstalk is the unwanted coupling of signals between parallel traces. Proper routing and layer stack-up through microstrip and stripline layouts can minimize crosstalk.

To reduce crosstalk in dual-stripline layouts that have two signal layers next to each other, route all traces perpendicular, increase the distance between the two signal layers, and minimize the distance between the signal layer and the adjacent reference plane (see Figure 11–11).

Figure 11–11. Dual- and Single-Stripline Layouts

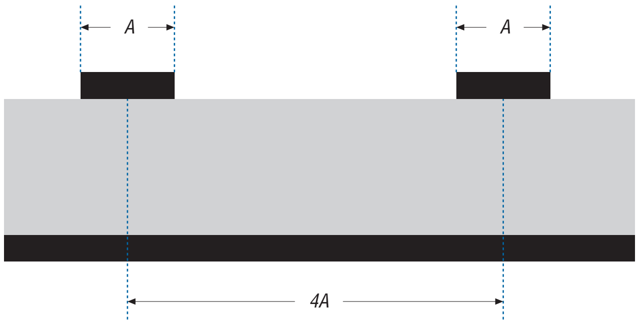


Take the following actions to reduce crosstalk in either microstrip or stripline layouts:

- Widen spacing between signal lines as much as routing restrictions will allow. Try not to bring traces closer than three times the dielectric height.
- Design the transmission line so that the conductor is as close to the ground plane as possible. This technique will couple the transmission line tightly to the ground plane and help decouple it from adjacent signals.
- Use differential routing techniques where possible, especially for critical nets (i.e., match the lengths as well as the turns that each trace goes through).
- If there is significant coupling, route single-ended signals on different layers orthogonal to each other.
- Minimize parallel run lengths between single-ended signals. Route with short parallel sections and minimize long, coupled sections between nets.

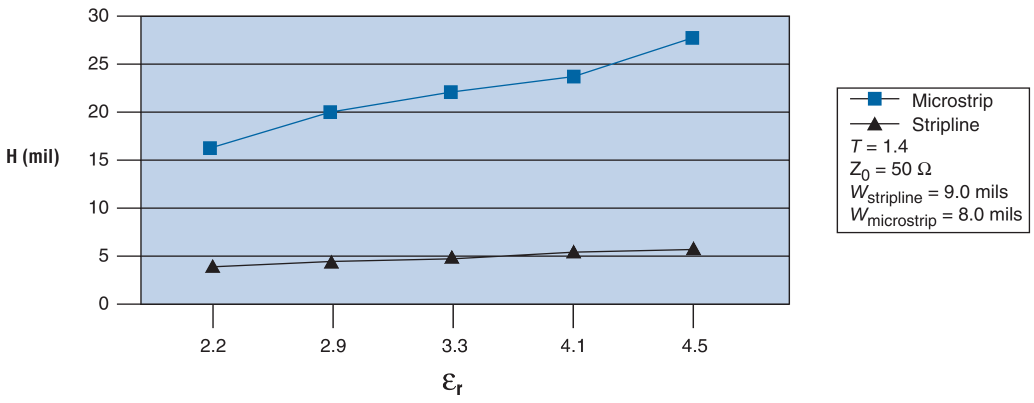
Crosstalk also increases when two or more single-ended traces run parallel and are not spaced far enough apart. The distance between the centers of two adjacent traces should be at least four times the trace width, as shown in Figure 11–12. To improve design performance, lower the distance between the trace and the ground plane to under 10 mils without changing the separation between the two traces.

Figure 11–12. Separating Traces for Crosstalk



Compared with high dielectric materials, low dielectric materials help reduce the thickness between the trace and ground plane while maintaining signal integrity. Figure 11–13 plots the relationship of height versus dielectric constant using the microstrip impedance and stripline impedance equations, keeping impedance, width, and thickness constant.

Figure 11–13. Height Versus Dielectric Constant



Signal Trace Routing

Proper routing helps to maintain signal integrity. To route a clean trace, you should perform simulation with good signal integrity tools. The following section describes the two different types of signal traces available for routing, single-ended traces, and differential pair traces.

Single-Ended Trace Routing

A single-ended trace connects the source and the load/receiver. Single-ended traces are used in general point-to-point routing, clock routing, low-speed, and non-critical I/O routing. This section discusses different routing schemes for clock signals. You can use the following types of routing to drive multiple devices with the same clock:

- Daisy chain routing
  - With stub
  - Without stub
- Star routing
- Serpentine routing

Use the following guidelines to improve the clock transmission line's signal integrity:

- Keep clock traces as straight as possible. Use arc-shaped traces instead of right-angle bends.
- Do not use multiple signal layers for clock signals.
- Do not use vias in clock transmission lines. Vias can cause impedance change and reflection.
- Place a ground plane next to the outer layer to minimize noise. If you use an inner layer to route the clock trace, sandwich the layer between reference planes.
- Terminate clock signals to minimize reflection.
- Use point-to-point clock traces as much as possible.

Daisy Chain Routing With Stubs

Daisy chain routing is a common practice in designing PCBs. One disadvantage of daisy chain routing is that stubs, or short traces, are usually necessary to connect devices to the main bus (see Figure 11–14). If a stub is too long, it will induce transmission line reflections and degrade signal quality. Therefore, the stub length should not exceed the following conditions:

$TD_{stub} < (T_{10\% \text{ to } 90\%})/3$

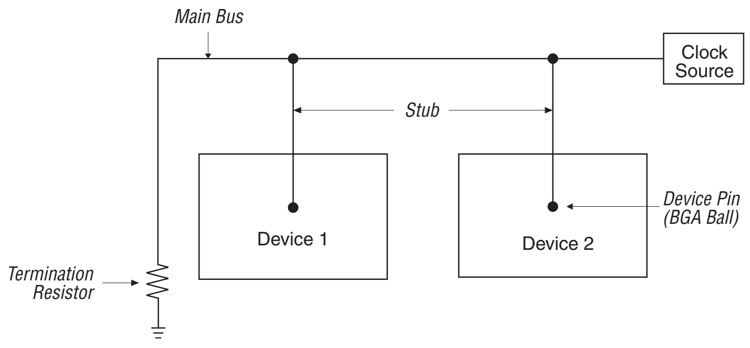
where TD_{stub} = Electrical delay of the stub

$T_{10\% \text{ to } 90\%}$ = Rise or fall time of signal edge

For a 1-ns rise-time edge, the stub length should be less than 0.5 inches (see the "References" section). If your design uses multiple devices, all stub lengths should be equal to minimize clock skew.

If possible, you should avoid using stubs in your PCB design. For high-speed designs, even very short stubs can create signal integrity problems.

Figure 11–14. Daisy Chain Routing with Stubs



Figures 11–15 through 11–17 show the SPICE simulation with different stub length. As the stub length decreases, there is less reflection noise, which causes the eye opening to increase.

Figure 11–15. Stub Length = 0.5 Inch

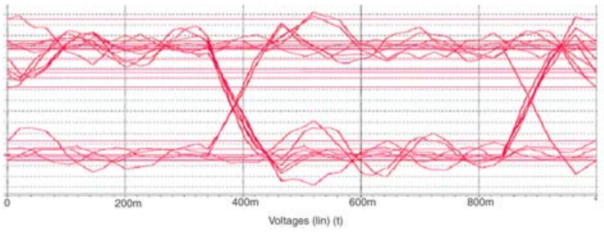


Figure 11–16. Stub Length = 0.25 Inch

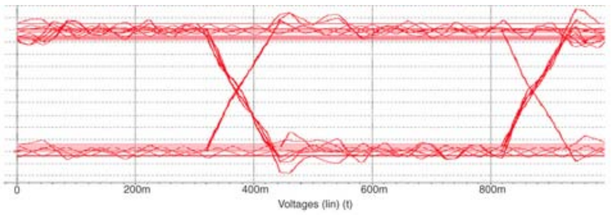
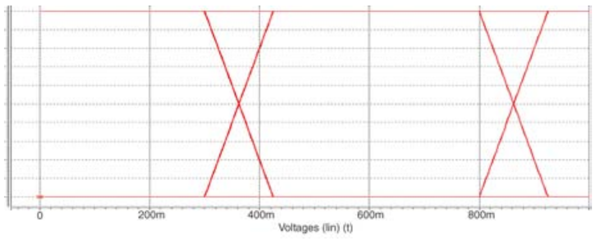


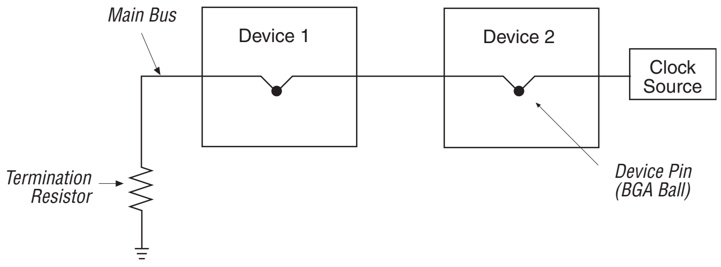
Figure 11–17. Stub Length = Zero Inches



Daisy Chain Routing without Stubs

Figure 11–18 shows daisy chain routing with the main bus running through the device pins, eliminating stubs. This layout removes the risk of impedance mismatch between the main bus and the stubs, minimizing signal integrity problems.

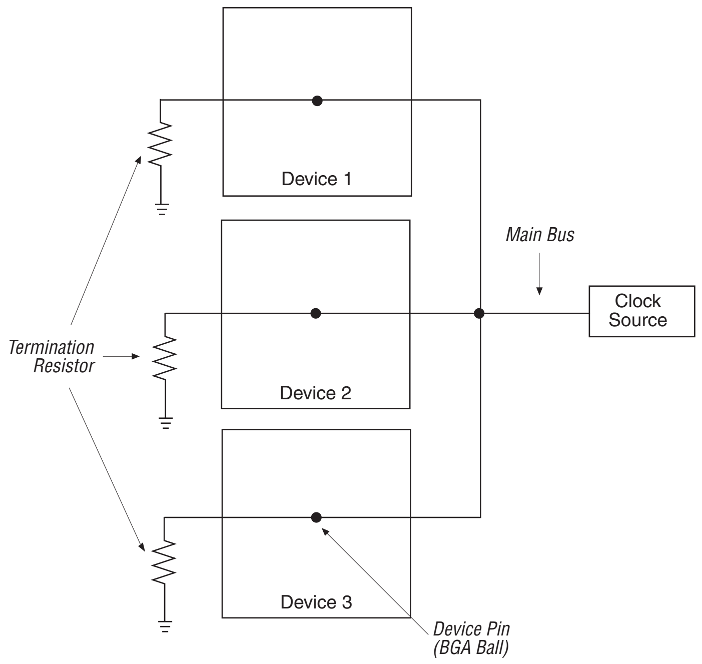
Figure 11–18. Daisy Chain Routing without Stubs



Star Routing

In star routing, the clock signal travels to all the devices at the same time (see Figure 11–19). Therefore, all trace lengths between the clock source and devices must be matched to minimize the clock skew. Each load should be identical to minimize signal integrity problems. In star routing, you must match the impedance of the main bus with the impedance of the long trace that connects to multiple devices.

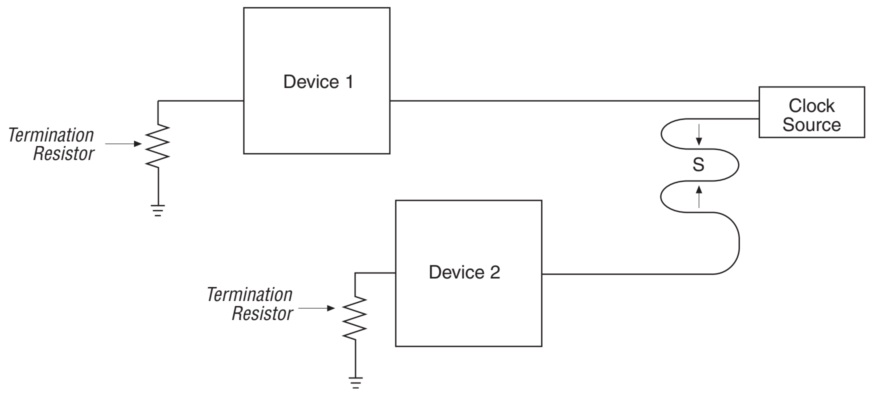
Figure 11–19. Star Routing



Serpentine Routing

When a design requires equal-length traces between the source and multiple loads, you can bend some traces to match trace lengths (see Figure 11–20). However, improper trace bending affects signal integrity and propagation delay. To minimize crosstalk, ensure that $S \geq 3 \times H$, where S is the spacing between the parallel sections and H is the height of the signal trace above the reference ground plane (see Figure 11–21).

Figure 11–20. Serpentine Routing

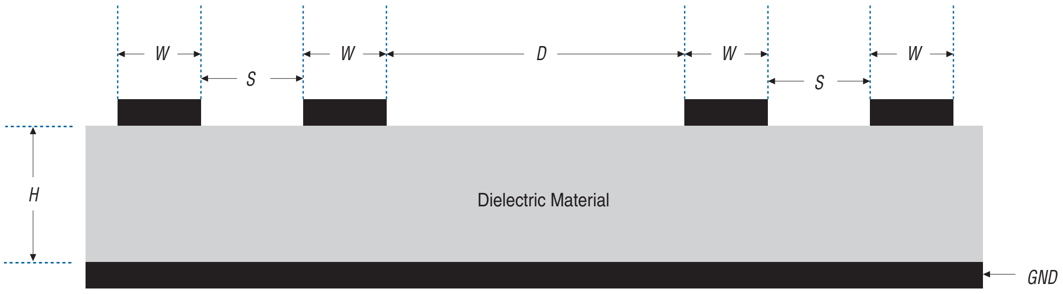


Altera recommends avoiding serpentine routing, if possible. Instead, use arcs to create equal-length traces.

Differential Trace Routing

To maximize signal integrity, proper routing techniques for differential signals are important for high-speed designs. Figure 11–21 shows two differential pairs using the microstrip layout.

Figure 11–21. Differential Trace Routing Note (1)



Note to Figure 11–21:

(1) *D* = distance between two differential pair signals; *W* = width of a trace in a differential pair; *S* = distance between the trace in a differential pair; and *H* = dielectric height above the group plane.

Use the following guidelines when using two differential pairs:

- Keep the distance between the differential traces (*S*) constant over the entire trace length.

- Ensure that $D > 2S$ to minimize the crosstalk between the two differential pairs.
- Place the differential traces $S = 3H$ as they leave the device to minimize reflection noise.
- Keep the length of the two differential traces the same to minimize the skew and phase difference.
- Avoid using multiple vias because they can cause impedance mismatch and inductance.

Termination Schemes

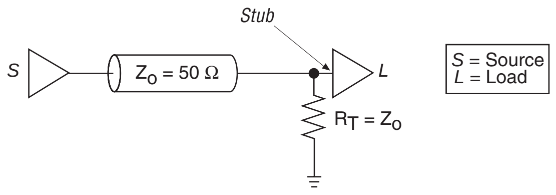
Mismatched impedance causes signals to reflect back and forth along the lines, which causes ringing at the load receiver. The ringing reduces the dynamic range of the receiver and can cause false triggering. To eliminate reflections, the impedance of the source (Z_S) must equal the impedance of the trace (Z_o), as well as the impedance of the load (Z_L). This section discusses the following signal termination schemes:

- Simple parallel termination
- Thevenin parallel termination
- Active parallel termination
- Series-RC parallel termination
- Series termination
- Differential pair termination

Simple Parallel Termination

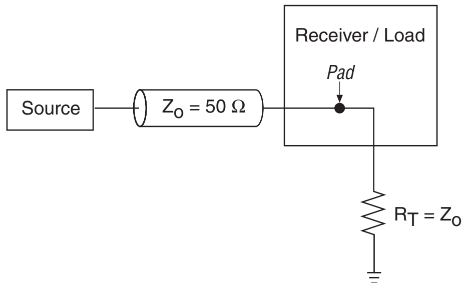
In a simple parallel termination scheme, the termination resistor (R_T) is equal to the line impedance. Place the R_T as close to the load as possible to be efficient (see Figure 11–22).

Figure 11–22. Simple Parallel Termination



The stub length from the R_T to the receiver pin and pads should be as small as possible. A long stub length causes reflections from the receiver pads, resulting in signal degradation. If your design requires a long termination line between the terminator and receiver, the placement of the resistor becomes important. For long termination line lengths, use fly-by termination (see Figure 11–23).

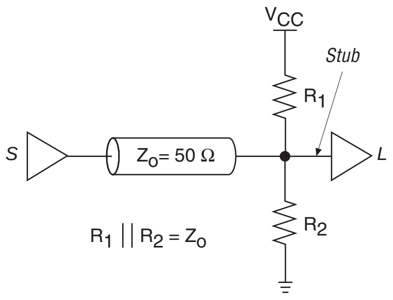
Figure 11–23. Simple Parallel Fly-By Termination



Thevenin Parallel Termination

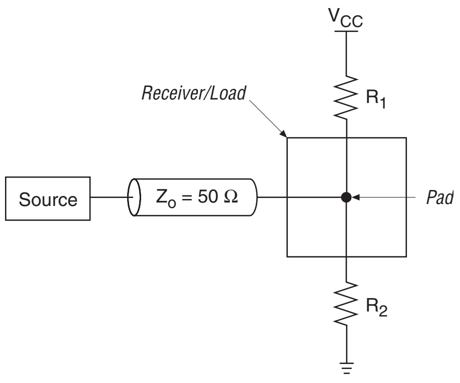
An alternative parallel termination scheme uses a Thevenin voltage divider (see Figure 11–24). The R_T is split between R_1 and R_2, which equals the line impedance when combined. Although this scheme reduces the current drawn from the source device, it adds current drawn from the power supply because the resistors are tied between V_{CC} and GND.

Figure 11–24. Thevenin Parallel Termination



As noted in the previous section, stub length is dependent on signal rise and fall time and should be kept to a minimum. If your design requires a long termination line between the terminator and receiver, use fly-by termination or Thevenin fly-by termination (see Figures 11–23 and 11–25).

Figure 11–25. Thevenin Parallel Fly-By Termination



Active Parallel Termination

Figure 11–26 shows an active parallel termination scheme, where the terminating resistor ($R_T = Z_o$) is tied to a bias voltage (V_{BIAS}). In this scheme, the voltage is selected so that the output drivers can draw current from the high- and low-level signals. However, this scheme requires a separate voltage source that can sink and source currents to match the output transfer rates.

Figure 11–26. Active Parallel Termination

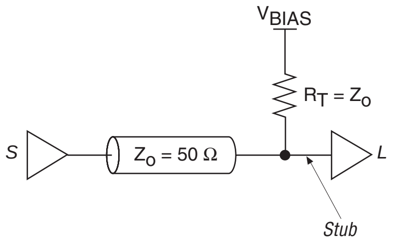
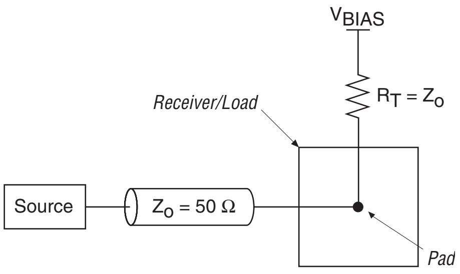


Figure 11–27 shows the active parallel fly-by termination scheme.

Figure 11–27. Active Parallel Fly-By Termination



Series-RC Parallel Termination

A series-RC parallel termination scheme uses a resistor and capacitor (series-RC) network as the terminating impedance. R_T is equal to Z_0. The capacitor must be large enough to filter the constant flow of DC current. However, if the capacitor is too large, it will delay the signal beyond the design threshold.

Capacitors smaller than 100 pF diminish the effectiveness of termination. The capacitor blocks low-frequency signals while passing high-frequency signals. Therefore, the DC loading effect of R_T does not have an impact on the driver, as there is no DC path to ground. The series-RC termination scheme requires balanced DC signaling, the signals spend half the time on and half the time off. AC termination is typically used if there is more than one load (see Figure 11–28).

Figure 11–28. Series-RC Parallel Termination

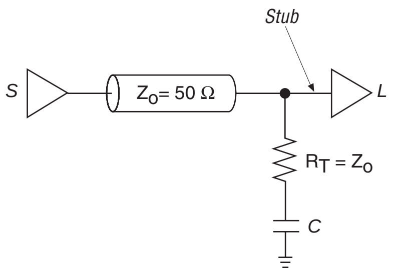
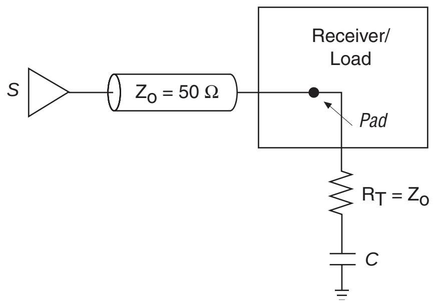


Figure 11–29 shows series-RC parallel fly-by termination.

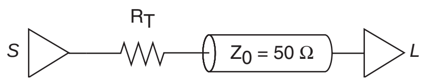
Figure 11–29. Series-RC Parallel Fly-By Termination



Series Termination

In a series termination scheme, the resistor matches the impedance at the signal source instead of matching the impedance at each load (see Figure 11–30). The sum of R_T and the impedance of the output driver should be equal to Z_0. Because Altera device output impedance is low, you should add a series resistor to match the signal source to the line impedance. The advantage of series termination is that it consumes little power. However, the disadvantage is that the rise time degrades because of the increased RC time constant. Therefore, for high-speed designs, you should perform the pre-layout signal integrity simulation with Altera I/O buffer information specification (IBIS) models before using the series termination scheme.

Figure 11–30. Series Termination



Differential Pair Termination

Differential signal I/O standards require an R_T between the signals at the receiving device (see Figure 11–31). For the low-voltage differential signal (LVDS) and low-voltage positive emitter-coupled logic (LVPECL) standard, the R_T should match the differential load impedance of the bus (typically 100 Ω).

Figure 11–31. Differential Pair (LVDS & LVPECL) Termination

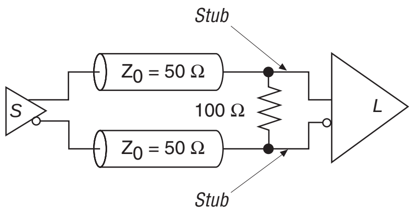
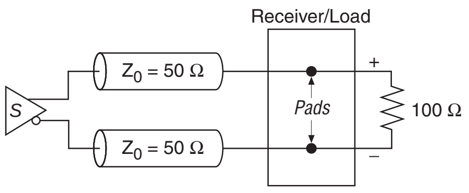


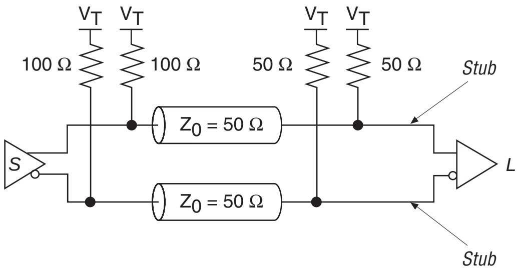
Figure 11–32 shows the differential pair fly-by termination scheme for the LVDS and LVPECL standard.

Figure 11–32. Differential Pair (LVDS & LVPECL) Fly-By Termination



3.3-V pseudo current mode logic (PCML) uses two parallel 100-Ω termination resistors at the transmitter and two parallel 50-Ω termination resistors at the receiver (see Figure 11–33). The termination voltage (V_T) is the same as the V_{CCIO} voltage (3.3 V).

Figure 11–33. Differential Pair (3.3-V PCML) Termination



See the *Board Design Guidelines for LVDS Systems* White Paper for more information on terminating differential signals.

Simultaneous Switching Noise

As digital devices become faster, their output switching times decrease. This causes higher transient currents in outputs as the devices discharge load capacitances. These higher transient currents result in a board-level phenomenon known as ground bounce.

Because many factors contribute to ground bounce, you cannot use a standard test method to predict its magnitude for all possible PCB environments. You can only test the device under a given set of conditions to determine the relative contributions of each condition and of the device itself. Load capacitance, socket inductance, and the number of switching outputs are the predominant factors that influence the magnitude of ground bounce in FPGAs.

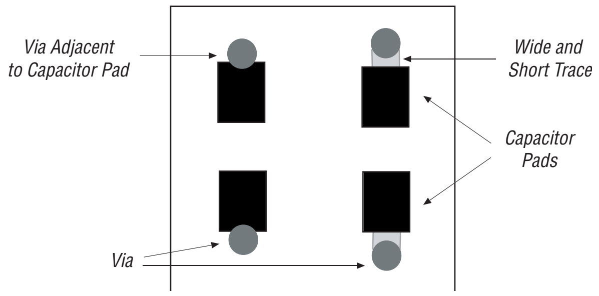
Altera requires 0.01- to 0.1-µF surface-mount capacitors in parallel to reduce ground bounce. Add an additional 0.001-µF capacitor in parallel to these capacitors to filter high-frequency noise (>100 MHz).

Altera recommends that you take the following action to reduce ground bounce and V_{CC} sag:

- Configure unused I/O pins as output pins, and drive the output low to reduce ground bounce. This configuration will act as a virtual ground.
- Configure the unused I/O pins as output, and drive high to prevent V_{CC} sag.
- Create a programmable ground or V_{CC} next to switching pins.
- Reduce the number of outputs that can switch simultaneously and distribute them evenly throughout the device.
- Manually assign ground pins in between I/O pins. (Separating I/O pins with ground pins prevents ground bounce.)
- Set the programmable drive strength feature with a weaker drive strength setting to slow down the edge rate.
- Eliminate sockets whenever possible. Sockets have inductance associated with them.
- Depending on the problem, move switching outputs close to either a package ground or VCC pin. Eliminate pull-up resistors, or use pull-down resistors.
- Use multi-layer PCBs that provide separate V_{CC} and ground planes to utilize the intrinsic capacitance of the V_{CC} /GND plane.
- Create synchronous designs that are not affected by momentarily switching pins.
- Add the recommended decoupling capacitors to V_{CC}/GND pairs.
- Place the decoupling capacitors as close as possible to the power and ground pins of the device.
- Connect the capacitor pad to the power and ground plane with larger vias to minimize the inductance in decoupling capacitors and allow for maximum current flow.

- Use wide, short traces between the vias and capacitor pads, or place the via adjacent to the capacitor pad (see Figure 11–34).

Figure 11–34. Suggested Via Location that Connects to Capacitor Pad



- Traces stretching from power pins to a power plane (or island, or a decoupling capacitor) should be as wide and as short as possible. This reduces series inductance, thereby reducing transient voltage drops from the power plane to the power pin which, in turn, decreases the possibility of ground bounce.
- Use surface-mount low effective series resistance (ESR) capacitors to minimize the lead inductance. The capacitors should have an ESR value as small as possible.
- Connect each ground pin or via to the ground plane individually. A daisy chain connection to the ground pins shares the ground path, which increases the return current loop and thus inductance.

Power Filtering & Distribution

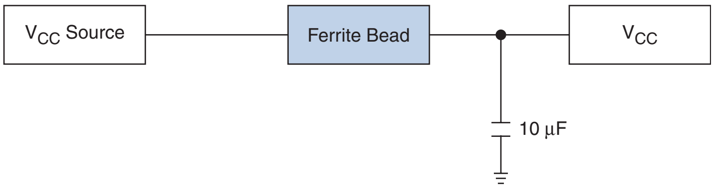
You can reduce system noise by providing clean, evenly distributed power to V_{CC} on all boards and devices. This section describes techniques for distributing and filtering power.

Filtering Noise

To decrease the low-frequency (< 1 kHz) noise caused by the power supply, filter the noise on power lines at the point where the power connects to the PCB and to each device. Place a 100-µF electrolytic capacitor where the power supply lines enter the PCB. If you use a voltage regulator, place the capacitor immediately after the pin that provides the VCC signal to the device(s). Capacitors not only filter low-frequency noise from the power supply, but also supply extra current when many outputs switch simultaneously in a circuit.

To filter power supply noise, use a non-resonant, surface-mount ferrite bead large enough to handle the current in series with the power supply. Place a 10- to 100-µF bypass capacitor next to the ferrite bead (see Figure 11–35). (If proper termination, layout, and filtering eliminate enough noise, you do not need to use a ferrite bead.) The ferrite bead acts as a short for high-frequency noise coming from the V_{CC} source. Any low-frequency noise is filtered by a large 10-µF capacitor after the ferrite bead.

Figure 11–35. Filtering Noise with a Ferrite Bead



Usually, elements on the PCB add high-frequency noise to the power plane. To filter the high-frequency noise at the device, place decoupling capacitors as close as possible to each V_{CC} and GND pair.

See the *Operating Requirements for Altera Devices* Data Sheet for more information on bypass capacitors.

Power Distribution

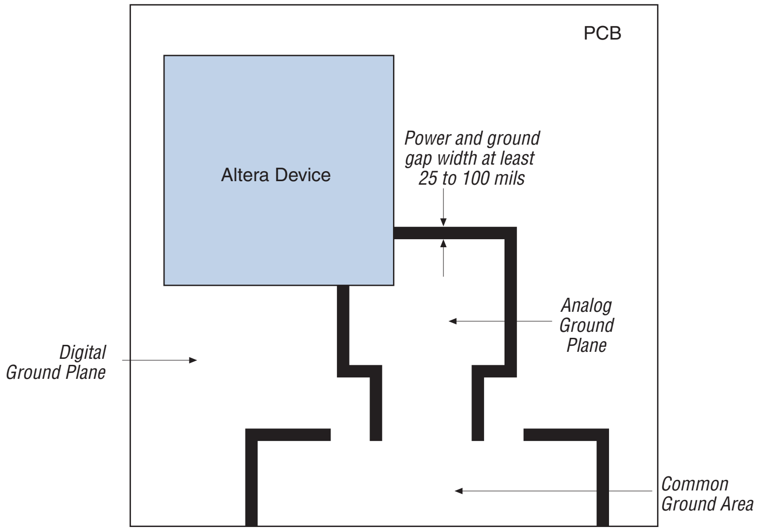
A system can distribute power throughout the PCB with either power planes or a power bus network.

You can use power planes on multi-layer PCBs that consist of two or more metal layers that carry V_{CC} and GND to the devices. Because the power plane covers the full area of the PCB, its DC resistance is very low. The power plane maintains V_{CC} and distributes it equally to all devices while providing very high current-sink capability, noise protection, and shielding for the logic signals on the PCB. Altera recommends using power planes to distribute power.

The power bus network—which consists of two or more wide-metal traces that carry V_{CC} and GND to devices—is often used on two-layer PCBs and is less expensive than power planes. When designing with power bus networks, be sure to keep the trace widths as wide as possible. The main drawback to using power bus networks is significant DC resistance.

Altera recommends using separate analog and digital power planes. For fully digital systems that do not already have a separate analog power plane, it can be expensive to add new power planes. However, you can create partitioned islands (split planes). Figure 11–36 shows an example board layout with phase-locked loop (PLL) ground islands.

Figure 11–36. Board Layout for General-Purpose PLL Ground Islands



If your system shares the same plane between analog and digital power supplies, there may be unwanted interaction between the two circuit types. The following suggestions will reduce noise:

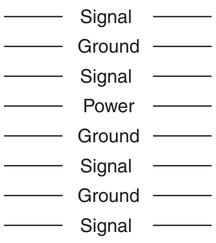
- For equal power distribution, use separate power planes for the analog (PLL) power supply. Avoid using trace or multiple signal layers to route the PLL power supply.
- Use a ground plane next to the PLL power supply plane to reduce power-generated noise.
- Place analog and digital components only over their respective ground planes.
- Use ferrite beads to isolate the PLL power supply from digital power supply.

Electromagnetic Interference (EMI)

Electromagnetic interference (EMI) is directly proportional to the change in current or voltage with respect to time. EMI is also directly proportional to the series inductance of the circuit. Every PCB generates EMI. Precautions such as minimizing crosstalk, proper grounding, and proper layer stack-up significantly reduce EMI problems.

Place each signal layer in between the ground plane and power plane. Inductance is directly proportional to the distance an electric charge has to cover from the source of an electric charge to ground. As the distance gets shorter, the inductance becomes smaller. Therefore, placing ground planes close to a signal source reduces inductance and helps contain EMI. Figure 11–37 shows an example of an eight-layer stack-up. In the stack-up, the stripline signal layers are the quietest because they are centered by power and GND planes. A solid ground plane next to the power plane creates a set of low ESR capacitors. With integrated circuit edge rates becoming faster and faster, these techniques help to contain EMI.

Figure 11–37. Example Eight-Layer Stack-Up



Component selection and proper placement on the board is important to controlling EMI.

The following guidelines can reduce EMI:

- Select low-inductance components, such as surface mount capacitors with low ESR, and effective series inductance.
- Use proper grounding for the shortest current return path.
- Use solid ground planes next to power planes.
- In unavoidable circumstances, use respective ground planes next to each segmented power plane for analog and digital circuits.

Additional FPGA-Specific Information

This section provides the following additional information recommended by Altera for board design and signal integrity: FPGA-specific configuration, Joint Test Action Group (JTAG) testing, and permanent test points.

Configuration

The DCLK signal is used in configuration devices and passive serial (PS) and passive parallel synchronous (PPS) configuration schemes. This signal drives edge-triggered pins in Altera devices. Therefore, any overshoot, undershoot, ringing, crosstalk, or other noise can affect configuration. Use the same guidelines for designing clock signals to route the DCLK trace (see the "Signal Trace Routing"section). If your design uses more than five configuration devices, Altera recommends using buffers to split the fan-out on the DCLK signal.

JTAG

As PCBs become more complex, testing becomes increasingly important. Advances in surface mount packaging and PCB manufacturing have resulted in smaller boards, making traditional test methods such as external test probes and "bed-of-nails" test fixtures harder to implement. As a result, cost savings from PCB space reductions can be offset by cost increases in traditional testing methods.

In addition to boundary scan testing (BST), you can use the IEEE Std. 1149.1 controller for in-system programming. JTAG consists of four required pins, test data input (TDI), test data output (TDO), test mode select (TMS), and test clock input (TCK) as well as an optional test reset input (TRST) pin.

Use the same guidelines for laying out clock signals to route TCK traces. Use multiple devices for long JTAG scan chains. Minimize the JTAG scan chain trace length that connects one device's TDO pins to another device's TDI pins to reduce delay.

See *Application Note 39: IEEE 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices* for additional details on BST.

Test Point

As device package pin density increases, it becomes more difficult to attach an oscilloscope or a logic analyzer probe on the device pin. Using a physical probe directly on to the device pin can damage the device. If the ball grid array (BGA) or FineLine BGA® package is mounted on top of the board, it is difficult to probe the other side of the board. Therefore, the PCB must have a permanent test point to probe. The test point can be a via that connects to the signal under test with a very short stub. However, placing a via on a trace for a signal under test can cause reflection and poor signal integrity.

Summary

You must carefully plan out a successful high-speed PCB. Factors such as noise generation, signal reflection, crosstalk, and ground bounce can interfere with a signal, especially with the high speeds that Altera devices transmit and receive. The signal routing, termination schemes, and power distribution techniques discussed in this chapter contribute to a more effectively designed PCB using high-speed Altera devices.

References

Johnson, H. W., and Graham, M., "*High-Speed Digital Design.*" Prentice Hall, 1993.

Hall, S. H., Hall, G. W., and McCall J. A., "*High-Speed Digital System Design.*" John Wiley & Sons, Inc. 2000.