



HARDCOPY™

HardCopy Series Handbook, Volume 1



101 Innovation Drive
San Jose, CA 95134
www.altera.com

Copyright © 2008 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. Mentor Graphics and ModelSim are registered trademarks of Mentor Graphics Corporation. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.





Chapter Revision Dates	ix
-------------------------------------	-----------

About this Handbook	xi
----------------------------------	-----------

How to Contact Altera	xi
-----------------------------	----

Typographic Conventions	xi
-------------------------------	----

Section I. HardCopy Stratix Device Family Data Sheet

Revision History	1-1
------------------------	-----

Chapter 1. Introduction to HardCopy Stratix Devices

Introduction	1-1
--------------------	-----

Features	1-2
----------------	-----

Document Revision History	1-4
---------------------------------	-----

Chapter 2. Description, Architecture, and Features

Introduction	2-1
--------------------	-----

HardCopy Stratix and Stratix FPGA Differences	2-2
---	-----

Logic Elements	2-4
----------------------	-----

Embedded Memory	2-4
-----------------------	-----

DSP Blocks	2-6
------------------	-----

PLLs and Clock Networks	2-6
-------------------------------	-----

I/O Structure and Features	2-6
----------------------------------	-----

Power-Up Modes in HardCopy Stratix Devices	2-7
--	-----

Hot Socketing	2-8
---------------------	-----

HARDCOPY_FPGA_PROTOTYPE Devices	2-9
---------------------------------------	-----

Document Revision History	2-10
---------------------------------	------

Chapter 3. Boundary-Scan Support

IEEE Std. 1149.1 (JTAG) Boundary-Scan Support	3-1
---	-----

Document Revision History	3-4
---------------------------------	-----

Chapter 4. Operating Conditions

Recommended Operating Conditions	4-1
--	-----

Power Consumption	4-15
-------------------------	------

Timing Closure	4-15
----------------------	------

External Timing Parameters	4-16
----------------------------------	------

HardCopy Stratix External I/O Timing	4-17
--	------

Maximum Input and Output Clock Rates	4-23
High-Speed I/O Specification	4-28
PLL Specifications	4-30
Electrostatic Discharge	4-33
Positive Voltage Zap	4-34
Negative Voltage Zap	4-35
Document Revision History	4-36

Chapter 5. Quartus II Support for HardCopy Stratix Devices

Introduction	5-1
Features	5-2
HARDCOPY_FPGA_PROTOTYPE, HardCopy Stratix and Stratix Devices	5-3
HardCopy Design Flow	5-5
The Design Flow Steps of the One Step Process	5-6
How to Design HardCopy Stratix Devices	5-7
Tcl Support for HardCopy Migration	5-11
Design Optimization and Performance Estimation	5-12
Design Optimization	5-12
Performance Estimation	5-12
Buffer Insertion	5-16
Placement Constraints	5-16
Location Constraints	5-17
LAB Assignments	5-17
LogicLock Assignments	5-18
Checking Designs for HardCopy Design Guidelines	5-19
Altera Recommended HDL Coding Guidelines	5-19
Design Assistant	5-19
Reports and Summary	5-20
Generating the HardCopy Design Database	5-21
Static Timing Analysis	5-23
Early Power Estimation	5-23
HardCopy Stratix Early Power Estimation	5-23
HardCopy APEX Early Power Estimation	5-24
Tcl Support for HardCopy Stratix	5-24
Targeting Designs to HardCopy APEX Devices	5-25
Conclusion	5-25
Related Documents	5-26
Document Revision History	5-26

Chapter 6. Design Guidelines for HardCopy Stratix Performance Improvement

Introduction	6-1
Background Information	6-1
Planning Stratix FPGA Design for HardCopy Stratix Design Conversion	6-2
Partitioning Your Design	6-2
Physical Synthesis Optimization	6-3
Using LogicLock Regions in HardCopy Stratix Designs	6-4
Recommended LogicLock Settings for HardCopy Stratix Designs	6-5

Using Design Space Explorer for HardCopy Stratix Designs	6–6
Recommended DSE Settings for HardCopy Stratix Designs	6–7
Performance Improvement Example	6–8
Initial Design Example Settings	6–8
Using Analysis and Synthesis Settings for Performance Improvement	6–11
Using Fitter Assignments and Physical Synthesis Optimizations for Performance Improvement	6–13
Design Space Explorer	6–15
Back-Annotation and Location Assignment Adjustments	6–17
Conclusion	6–21
Document Revision History	6–22

Section II. HardCopy APEX Device Family Data Sheet

Revision History	6–1
------------------------	-----

Chapter 7. Introduction to HardCopy APEX Devices

Introduction	7–1
Features... ..	7–1
...and More Features	7–2
Document Revision History	7–5

Chapter 8. Description, Architecture, and Features

Introduction	8–1
Differences Between HardCopy APEX and APEX 20K FPGAs	8–5
Power-up Mode and Configuration Emulation	8–5
Speed Grades	8–6
Quartus II-Generated Output Files	8–6
Document Revision History	8–7

Chapter 9. Boundary-Scan Support

IEEE Std. 1149.1 (JTAG) Boundary-Scan Support	9–1
Document Revision History	9–3

Chapter 10. Operating Conditions

Recommended Operating Conditions	10–1
Document Revision History	10–15

Section III. General HardCopy Series Design Considerations

Revision History	10–1
------------------------	------

Chapter 11. Design Guidelines for HardCopy Series Devices

Introduction	11–1
--------------------	------

Design Assistant Tool	11-1
Asynchronous Clock Domains	11-2
Transferring Data between Two Asynchronous Clock Domains	11-4
Gated Clocks	11-7
Preferred Clock Gating Circuit	11-7
Alternative Clock Gating Circuits	11-9
Inverted Clocks	11-11
Clocks Driving Non-Clock Pins	11-11
Clock Signals Should Use Dedicated Clock Resources	11-13
Mixing Clock Edges	11-14
Combinational Loops	11-16
Intentional Delays	11-18
Ripple Counters	11-20
Pulse Generators	11-21
Combinational Oscillator Circuits	11-24
Reset Circuitry	11-25
Gated Reset	11-25
Asynchronous Reset Synchronization	11-26
Synchronizing Reset Signals Across Clock Domains	11-27
Asynchronous RAM	11-30
Conclusion	11-31
Document Revision History	11-31

Chapter 12. Power-Up Modes and Configuration Emulation in HardCopy Series Devices

Introduction	12-1
HardCopy Power-Up Options	12-1
Instant On Options	12-2
Configuration Emulation of FPGA Configuration Sequence	12-9
Power-Up Options Summary When Designing With HardCopy Series Devices	12-15
Power-Up Option Selection and Examples	12-17
Replacing One FPGA With One HardCopy Series Device	12-18
Replacing One or More FPGAs With One or More HardCopy Series Devices in a Multiple-Device Configuration Chain	12-19
Replacing all FPGAs with HardCopy Series Devices in a Multiple-Device Configuration Chain	12-21
FPGA to HardCopy Configuration Migration Examples	12-21
HardCopy Series Device Replacing a Stand-Alone FPGA	12-21
HardCopy Series Device Replacing an FPGA in a Cascaded Configuration Chain	12-23
HardCopy Series Device Replacing an FPGA Configured Using a Microprocessor	12-25
HardCopy Stratix Device Replacing FPGA Configured in a JTAG Chain	12-28
HardCopy II Device Replacing Stratix II Device Configured With a Microprocessor	12-30
Conclusion	12-32
Document Revision History	12-33

Section IV. HardCopy Design Center Migration Process

Revision History	12-1
------------------------	------

Chapter 13. Back-End Design Flow for HardCopy Series Devices

Introduction	13-1
HardCopy II Back-End Design Flow	13-1
Device Netlist Generation	13-2
Design for Testability Insertion	13-3
Clock Tree and Global Signal Insertion	13-3
Formal Verification of the Processed Netlist	13-3
Timing and Signal Integrity Driven Place and Route	13-3
Parasitic Extraction and Timing Analysis	13-4
Layout Verification	13-4
Design Signoff	13-4
HardCopy Stratix and HardCopy APEX Migration Flow	13-5
Netlist Generation	13-6
Testability Audit	13-6
Placement	13-6
Test Vector Generation	13-7
Routing	13-7
Extracted Delay Calculation	13-7
Static Timing Analysis and Timing Closure	13-7
Formal Verification	13-8
Physical Verification	13-8
Manufacturing	13-8
Testing	13-9
Unused Resources	13-11
Conclusion	13-12
Document Revision History	13-12

Chapter 14. Back-End Timing Closure for HardCopy Series Devices

Introduction	14-1
Timing Analysis of HardCopy Prototype Device	14-1
Cell Structure	14-2
HardCopy II	14-2
HardCopy Stratix, HardCopy APEX	14-2
Clock Tree Structure	14-3
HardCopy II	14-3
HardCopy Stratix	14-3
HardCopy APEX	14-3
Importance of Timing Constraints	14-4
Correcting Timing Violations	14-4
Hold-Time Violations	14-5
Setup-Time Violations	14-10
Timing ECOs	14-15
Conclusion	14-17

Document Revision History 14–17



Chapter Revision Dates

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

Chapter 1. Introduction to HardCopy Stratix Devices

Revised: *September 2008*

Part number: *H51001-2.3*

Chapter 2. Description, Architecture, and Features

Revised: *September 2008*

Part number: *H51002-3.3*

Chapter 3. Boundary-Scan Support

Revised: *September 2008*

Part number: *H51004-3.3*

Chapter 4. Operating Conditions

Revised: *September 2008*

Part number: *H51005-3.3*

Chapter 5. Quartus II Support for HardCopy Stratix Devices

Revised: *September 2008*

Part number: *H51014-3.3*

Chapter 6. Design Guidelines for HardCopy Stratix Performance Improvement

Revised: *September 2008*

Part number: *H51027-1.3*

Chapter 7. Introduction to HardCopy APEX Devices

Revised: *September 2008*

Part number: *H51006-2.2*

Chapter 8. Description, Architecture, and Features

Revised: *September 2008*

Part number: *H51007-2.2*

Chapter 9. Boundary-Scan Support

Revised: *September 2008*

Part number: *H51009-2.2*

Chapter 10. Operating Conditions

Revised: *September 2008*

Part number: *H51010-2.2*

Chapter 11. Design Guidelines for HardCopy Series Devices

Revised: *September 2008*

Part number: *H51011-3.3*

Chapter 12. Power-Up Modes and Configuration Emulation in HardCopy Series Devices

Revised: *September 2008*

Part number: *H51012-2.4*

Chapter 13. Back-End Design Flow for HardCopy Series Devices

Revised: *September 2008*

Part number: *H51019-1.3*

Chapter 14. Back-End Timing Closure for HardCopy Series Devices

Revised: *September 2008*

Part number: *H51013-2.3*



About this Handbook

This handbook provides comprehensive information about the Altera® HardCopy® devices.

How to Contact Altera

For the most up-to-date information about Altera products, refer to the following table.

Contact	Contact Method	Address
Technical support	Website	www.altera.com/support/
Technical training	Website	www.altera.com/training
	Email	custrain@altera.com
Product literature	Website	www.altera.com/literature
Altera literature services	Email	literature@altera.com
Non-technical (General) (SoftwareLicensing)	Email	nacomp@altera.com
	Email	authorization@altera.com

Note to table:

(1) You can also contact your local Altera sales office or sales representative.

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: f_{MAX} , lqdesigns directory, d: drive, chiptrip.gdf file.
<i>Italic Type with Initial Capital Letters</i>	Document titles are shown in italic type with initial capital letters. Example: <i>AN 75: High-Speed Board Design</i> .

Visual Cue	Meaning
<i>Italic type</i>	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>.pdf 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.
	A caution calls attention to a condition or possible situation that can damage or destroy the product or the user's work.
	A warning calls attention to a condition or possible situation that can cause injury to the user.
	The angled arrow indicates you should press the Enter key.
	The feet direct you to more information on a particular topic.



Section I. HardCopy Stratix Device Family Data Sheet

This section provides designers with the data sheet specifications for HardCopy® Stratix structured ASICs. The chapters contain feature definitions of the internal architecture, JTAG boundary-scan testing information, DC operating conditions, AC timing parameters, and a reference to power consumption for HardCopy Stratix structured ASICs.

This section contains the following:

- [Chapter 1, Introduction to HardCopy Stratix Devices](#)
- [Chapter 2, Description, Architecture, and Features](#)
- [Chapter 3, Boundary-Scan Support](#)
- [Chapter 4, Operating Conditions](#)
- [Chapter 5, Quartus II Support for HardCopy Stratix Devices](#)
- [Chapter 6, Design Guidelines for HardCopy Stratix Performance Improvement](#)

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the complete handbook.



1. Introduction to HardCopy Stratix Devices

H51001-2.4

Introduction

HardCopy® Stratix® structured ASICs, Altera's second-generation HardCopy structured ASICs, are low-cost, high-performance devices with the same architecture as the high-density Stratix FPGAs. The combination of Stratix FPGAs for prototyping and design verification, HardCopy Stratix devices for high-volume production, and the Quartus® II design software beginning with version 3.0, provide a complete and powerful alternative to ASIC design and development.

HardCopy Stratix devices are architecturally equivalent and have the same features as the corresponding Stratix FPGA. They offer pin-to-pin compatibility using the same package as the corresponding Stratix FPGA prototype. Designers can prototype their design to verify functionality with Stratix FPGAs before seamlessly migrating the proven design to a HardCopy Stratix structured ASIC.

The Quartus II software provides a complete set of inexpensive and easy-to-use tools for designing HardCopy Stratix devices. Using the successful and proven methodology from HardCopy APEX™ devices, Stratix FPGA designs can be seamlessly and quickly migrated to a low-cost ASIC alternative. Designers can use the Quartus II software to design HardCopy Stratix devices to obtain an average of 50% higher performance and up to 40% lower power consumption than can be achieved in the corresponding Stratix FPGAs. The migration process is fully automated, requires minimal customer involvement, and takes approximately eight weeks to deliver fully tested HardCopy Stratix prototypes.

The HardCopy Stratix devices use the same base arrays across multiple designs for a given device density and are customized using the top two metal layers. The HardCopy Stratix family consists of the HC1S25, HC1S30, HC1S40, HC1S60, and HC1S80 devices. [Table 1-1](#) provides the details of the HardCopy Stratix devices.

Table 1–1. HardCopy Stratix Devices and Features

Device	LEs (1)	M512 Blocks	M4K Blocks	M-RAM Blocks	DSP Blocks (2)	PLLs (3)
HC1S25	25,660	224	138	2	10	6
HC1S30	32,470	295	171	2 (4)	12	6
HC1S40	41,250	384	183	2 (4)	14	6
HC1S60	57,120	574	292	6	18	12
HC1S80	79,040	767	364	6 (4)	22	12

Notes to Table 1–1:

- (1) LE: logic elements.
- (2) DSP: digital signal processing.
- (3) PLLs: phase-locked loops.
- (4) In HC1S30, HC1S40, and HC1S80 devices, there are fewer M-RAM blocks than in the equivalent Stratix FPGA. All other resources are identical to the Stratix counterpart.

Features

HardCopy Stratix devices are manufactured on the same 1.5-V, 0.13 μm all-layer-copper metal fabrication process (up to eight layers of metal) as the Stratix FPGAs.

- Preserves the functionality of a configured Stratix device
- Pin-compatible with the Stratix counterparts
- On average, 50% faster than their Stratix equivalents
- On average, 40% less power consumption than their Stratix equivalents
- 25,660 to 79,040 LEs
- Up to 5,658,408 RAM bits available
- TriMatrix memory architecture consisting of three RAM block sizes to implement true dual-port memory and first-in-first-out (FIFO) buffers
- Embedded high-speed DSP blocks provide dedicated implementation of multipliers, multiply-accumulate functions, and finite impulse response (FIR) filters
- Up to 12 PLLs (four enhanced PLLs and eight fast PLLs) per device which provide identical features as the FPGA counterparts, including spread spectrum, programmable bandwidth, clock switchover, real-time PLL reconfiguration, advanced multiplication, and phase shifting
- Supports numerous single-ended and differential I/O standards
- Supports high-speed networking and communications bus standards including RapidIO™, UTOPIA IV, CSIX, HyperTransport technology, 10G Ethernet XSBI, SPI-4 Phase 2 (POS-PHY Level 4), and SFI-4
- Differential on-chip termination support for LVDS

- Supports high-speed external memory, including zero bus turnaround (ZBT) SRAM, quad data rate (QDR and QDRII) SRAM, double data rate (DDR) SDRAM, DDR fast-cycle RAM (FCRAM), and single data rate (SDR) SDRAM
- Support for multiple intellectual property (IP) megafunctions from Altera® MegaCore® functions, and Altera Megafunction Partners Program (AMPPSM) megafunctions
- Available in space-saving flip-chip FineLine BGA® and wire-bond packages (Tables 1–2 and 1–3)
- Optional emulation of original FPGA configuration sequence
- Optional instant-on power-up



The actual performance and power consumption improvements over the Stratix equivalents mentioned in this data sheet are design-dependent.

Table 1–2. HardCopy Stratix Device Package Options and I/O Pin Counts
Note (1)

Device	672-Pin FineLine BGA (2)	780-Pin FineLine BGA (3)	1,020-Pin FineLine BGA (3)
HC1S25	473		
HC1S30		597	
HC1S40		613 (4)	
HC1S60			782
HC1S80			782

Notes to Table 1–2:

- (1) Quartus II I/O pin counts include one additional pin, PLENA, which is not a general-purpose I/O pin. PLENA can only be used to enable the PLLs.
- (2) This device uses a wire-bond package.
- (3) This device uses a flip-chip package.
- (4) In the Stratix EP1S40F780 FPGA, the I/O pins U12 and U18 are general-purpose I/O pins. In the FPGA prototype, EP1S40F780_HARDCOPY_FPGA_PROTOTYPE, and in the HardCopy Stratix HC1S40F780 device, U12 and U18 must be connected to ground. The EP1S40F780_HARDCOPY_FPGA_PROTOTYPE and HC1S40F780 pin-outs are identical.

Table 1–3. HardCopy Stratix Device Package Sizes

Device	672-Pin FineLine BGA	780-Pin FineLine BGA	1,020-Pin FineLine BGA
Pitch (mm)	1.00	1.00	1.00
Area (mm ²)	729	841	1,089
Length × width (mm × mm)	27 × 27	29 × 29	33 × 33

Document Revision History

Table 1–4 shows the revision history for this chapter.

Table 1–4. Document Revision History

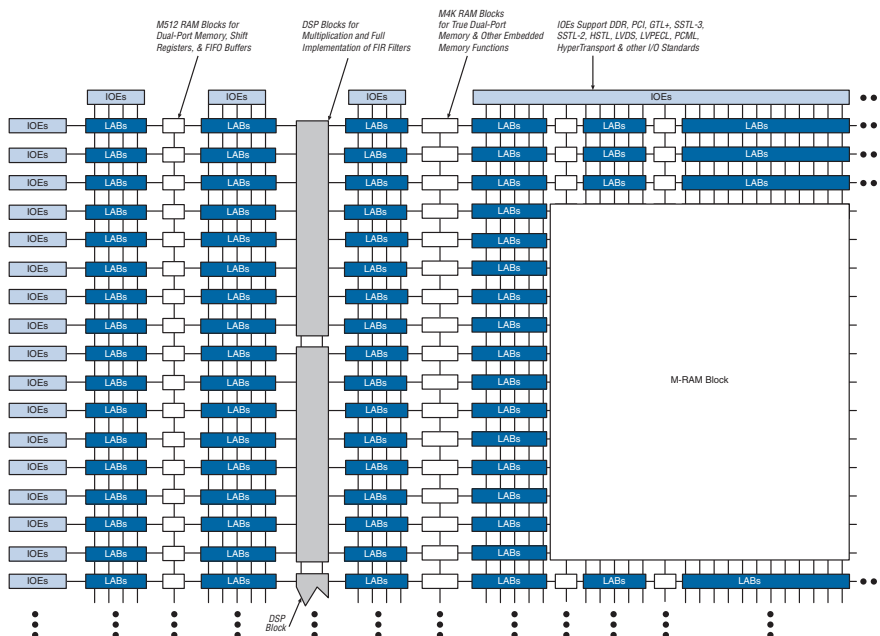
Date and Document Version	Changes Made	Summary of Changes
September 2008 v2.4	Revised chapter number and metadata.	—
June 2007 v2.3	Updated Introduction section. Updated Table 1–2.	—
December 2006 v2.2	Updated revision history.	—
March 2006	Formerly chapter 5; no content change.	—
October 2005 v2.1	Minor edits	—
January 2005 v2.0	Minor edits	—
June 2003 v1.0	Initial release of Chapter 5, <i>Introduction to HardCopy Stratix Devices</i> , in the <i>HardCopy Device Handbook</i> .	

Introduction

HardCopy® Stratix® structured ASICs provide a comprehensive alternative to ASICs. The HardCopy Stratix device family is fully supported by the Quartus® II design software, and, combined with a vast intellectual property (IP) portfolio, provides a complete path from prototype to volume production. Designers can now procure devices, tools, and Altera® IP for their high-volume applications.

As shown in Figure 2–1, HardCopy Stratix devices preserve their Stratix FPGA counterpart's architecture, but the programmability for logic, memory, and interconnect is removed. HardCopy Stratix devices are also manufactured in the same process technology and process voltage as Stratix FPGAs. Removing all configuration and programmable routing resources and replacing it with direct metal interconnect results in considerable die size reduction and the ensuing cost savings.

Figure 2–1. HardCopy Stratix Device Architecture



The HardCopy Stratix family consists of base arrays that are common to all designs for a particular device density. Design-specific customization is done within the top two metal layers. The base arrays use an area-efficient sea-of-logic-elements (SOLE) core and extend the flexibility of high-density Stratix FPGAs to a cost-effective, high-volume production solution. With a seamless migration process employed in numerous successful designs, functionality-verified Stratix FPGA designs can be migrated to fixed-function HardCopy Stratix devices with minimal risk and guaranteed first-time success.

The SRAM configuration cells of the original Stratix devices are replaced in HardCopy Stratix devices by metal connects, which define the function of each logic element (LE), digital signal processing (DSP) block, phase-locked loop (PLL), embedded memory, and I/O cell in the device. These resources are interconnected using metallization layers. Once a HardCopy Stratix device has been manufactured, the functionality of the device is fixed and no re-programming is possible. However, as is the case with Stratix FPGAs, the PLLs can be dynamically configured in HardCopy Stratix devices.

HardCopy Stratix and Stratix FPGA Differences

To ensure HardCopy Stratix device functionality and performance, designers should thoroughly test the original Stratix FPGA-based design for satisfactory results before committing the design for migration to a HardCopy Stratix device. Unlike Stratix FPGAs, HardCopy Stratix devices are customized at the time of manufacturing and therefore do not have programmability support.

Since HardCopy Stratix devices are customized within the top two metal layers, no configuration circuitry is required. Refer to [“Power-Up Modes in HardCopy Stratix Devices” on page 2–7](#) for more information.

Depending on the design, HardCopy Stratix devices can provide, on average, a 50% performance improvement over equivalent Stratix FPGAs. The performance improvement is achieved by die size reduction, metal interconnect optimization, and customized signal buffering. HardCopy Stratix devices consume, on average, 40% less power than their equivalent Stratix FPGAs.



Designers can use the Quartus II software to design HardCopy Stratix devices, estimate performance and power consumption, and maximize system throughput.

Table 2–1 illustrates the differences between HardCopy Stratix and Stratix devices.

Table 2–1. HardCopy Stratix and Stratix Device Comparison (Part 1 of 2)	
HardCopy Stratix	Stratix
Customized device. All reprogrammability support is removed and no configuration is required.	Re-programmable with configuration is required upon power-up.
Average of 50% performance improvement over corresponding FPGA (1).	High-performance FPGA.
Average of 40% less power consumption compared to corresponding FPGA (1).	Standard FPGA power consumption.
Contact Altera for information regarding specific IP support.	IP support for all devices is available.
Double data rate (DDR) SDRAM maximum operating frequency is pending characterization.	DDR SDRAM can operate at 200 MHz for -5 speed grade devices.
All routing connections are direct and all unused routing is removed.	MultiTrack™ routing stitches together routing resources to provide a path.
HC1S30 and HC1S40 devices have two M-RAM blocks. HC1S80 devices have six M-RAM blocks.	EP1S30 and EP1S40 devices have four M-RAM blocks. EP1S80 devices have nine M-RAM blocks.
It is not possible to initialize M512 and M4K RAM contents during power-up.	The contents of M512 and M4K RAM blocks can be preloaded during configuration with data specified in a memory initialization file (.mif).
The contents of memory output registers are unknown after power-on reset (POR).	The contents of memory output registers are initialized to '0' after POR.
HC1S30 and HC1S40 devices have six PLLs.	HC1S30 devices have 10 PLLs. HC1S40 devices have 12 PLLs.
PLL dynamic reconfiguration uses ROM for information. This reconfiguration is performed in the back-end and does not affect the migration flow.	PLL dynamic reconfiguration uses a MIF to initialize a RAM resource with information.
The I/O elements (IOEs) are equivalent but not identical to FPGA IOEs due to slight design optimizations for HardCopy devices.	The IOEs are optimized for the FPGA architecture.

Table 2–1. HardCopy Stratix and Stratix Device Comparison (Part 2 of 2)

HardCopy Stratix	Stratix
The I/O drive strength for single-ended I/O pins are slightly different and is modeled in the HardCopy Stratix IBIS models.	The I/O drive strength for single-ended I/O pins are found in Stratix IBIS models.
In the HC1S40 780-pin FineLine BGA® device, the I/O pins U12 and U18 must be connected to ground.	In the HC1S40 780-pin FineLine BGA device, the I/O pins U12 and U18 are available as general-purpose I/O pins.
The BSDL file describes re-ordered Joint Test Action Group (JTAG) boundary-scan chains.	The JTAG boundary-scan chain is defined in the BSDL file.

Note to Table 2–1:

- (1) Performance and power consumption are design dependant.

Logic Elements

Logic is implemented in HardCopy Stratix devices using the same architectural units as the Stratix device family. The basic unit is the logic element (LE) with logic array blocks (LAB) consisting of 10 LEs. The implementation of LEs and LABs is identical to the Stratix device family.

In the HardCopy Stratix device family, all extraneous routing resources not essential to the specific design are removed for performance and die size efficiency. Therefore, the MultiTrack interconnect for routing implementation between LABs and other device resources in the Stratix device family is no longer necessary in the HardCopy Stratix device family.

Table 2–2 illustrates the differences between HardCopy Stratix and Stratix logic.

Table 2–2. HardCopy Stratix and Stratix Logic Comparison

HardCopy Stratix	Stratix
All routing connections are direct and all unused routing is removed.	MultiTrack routing stitches routing resources together to provide a path.

Embedded Memory

TriMatrix™ memory blocks from Stratix devices, including M512, M4K, and M-RAM memory blocks, are available in HardCopy Stratix devices. Embedded memory is seamlessly implemented in the equivalent resource.

Although memory resource implementation is equivalent, the number of specific M-RAM blocks are not necessarily the same between corresponding Stratix and HardCopy Stratix devices. Table 2–3 shows the number of M-RAM blocks available in each device.

Table 2–3. HardCopy Stratix and Stratix M-RAM Block Comparison

HardCopy Stratix		Stratix	
Device	M-RAM Blocks	Device	M-RAM Blocks
HC1S25	2	EP1S25	2
HC1S30	2	EP1S30	4
HC1S40	2	EP1S40	4
HC1S60	6	EP1S60	6
HC1S830	6	EP1S830	9

In HardCopy Stratix devices, it is not possible to preload RAM contents using a MIF after powering up; the output registers of memory blocks will have unknown values. This occurs because there is no configuration process that is executed.



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

Table 2–4 illustrates the differences between HardCopy Stratix and Stratix memory.

Table 2–4. HardCopy Stratix and Stratix Memory Comparison

HardCopy Stratix	Stratix
HC1S30 and HC1S40 devices have two M-RAM blocks. HC1S80 devices have six M-RAM blocks.	EP1S30 and EP1S40 devices have four M-RAM blocks. EP1S80 devices have nine M-RAM blocks.
It is not possible to initialize M512 and M4k RAM contents during power-up.	The contents of M512 and M4K RAM blocks can be preloaded during configuration with data specified in a MIF.
The contents of memory output registers are unknown after POR.	The contents of memory output registers are initialized to '0' after POR.

DSP Blocks

DSP blocks in HardCopy Stratix devices are architecturally identical to those in Stratix devices. The number of DSP blocks available in HardCopy Stratix devices matches the number of DSP blocks available in the corresponding Stratix device.

PLLs and Clock Networks

The PLLs in HardCopy Stratix devices are identical to those in Stratix devices. The clock networks are also implemented exactly as they are in Stratix devices. The number of PLLs can vary between corresponding Stratix and HardCopy Stratix devices. Table 2–5 shows the number of PLLs available in each device.

Table 2–5. HardCopy Stratix and Stratix PLL Comparison			
HardCopy Stratix		Stratix	
Device	PLLs	Device	PLLs
HC1S25	6	EP1S25	6
HC1S30	6	EP1S30	10
HC1S40	6	EP1S40	12
HC1S60	12	EP1S60	12
EP1S830	12	EP1S830	12

Table 2–6 illustrates the differences between HardCopy Stratix and Stratix PLLs.

Table 2–6. HardCopy Stratix and Stratix PLL Differences	
HardCopy Stratix	Stratix
HC1S30 and HC1S40 devices have six PLLs.	HC1S30 devices have 10 PLLs. HC1S40 devices have 12 PLLs.
PLL dynamic reconfiguration uses ROM for information. This reconfiguration is performed in the back-end and does not affect the migration flow.	PLL dynamic reconfiguration uses a MIF to initialize a RAM resource with information.

I/O Structure and Features

The HardCopy Stratix IOEs are equivalent, but not identical to, the Stratix FPGA IOEs. This is due to the reduced die size, layout difference, and metal customization of the HardCopy Stratix device. The differences are minor but may be relevant to customers designing with tight DC and switching characteristics. However, no signal integrity concerns are introduced with HardCopy Stratix IOEs.

When designing with very tight timing constraints (for example, DDR or quad data rate [QDR]), or if using the programmable drive strength option, Altera recommends verifying final drive strength using updated IBIS models located on the Altera website at www.altera.com. Differential I/O standards are unaffected.

I/O pin placement and V_{REF} pin placement rules are identical between HardCopy Stratix and Stratix devices. Unused pin settings will carry over from Stratix device settings and are implemented as tri-stated outputs driving ground or outputs driving V_{CC} .

In Stratix EP1S40 780-pin FineLine BGA FPGAs, the I/O pins U12 and U18 are available as general-purpose I/O pins. In the FPGA prototype, EP1S40F780_HARDCOPY_FPGA_PROTOTYPE, and in the Hardcopy Stratix HC1S40 780-pin FineLine BGA device, the I/O pins U12 and U18 must be connected to ground. HC1S40 780-pin FineLine BGA and EP1S40F780_HARDCOPY_FPGA_PROTOTYPE pin-outs are identical.

Table 2–7 illustrates the differences between HardCopy Stratix and Stratix I/O pins.

Table 2–7. HardCopy Stratix and Stratix I/O Pin Comparison	
HardCopy Stratix	Stratix
The IOEs are equivalent, but not identical to, the FPGA IOEs due to slight design optimizations for HardCopy devices.	IOEs are optimized for the FPGA architecture.
The I/O drive strength for single-ended I/O pins are slightly different and are found in the HardCopy Stratix IBIS models.	The I/O drive strength for single-ended I/O pins are found in Stratix IBIS models.
In the HC1S40 780-pin FineLine BGA device, the I/O pins U12 and U18 must be connected to ground.	In the EP1S40 780-pin FineLine BGA device, the I/O pins U12 and U18 are available as general-purpose I/O pins.

Power-Up Modes in HardCopy Stratix Devices

Designers do not need to configure HardCopy Stratix devices, unlike their FPGA counterparts. However, to facilitate seamless migration, configuration can be emulated in HardCopy Stratix devices.

The modes in which a HardCopy Stratix device can be made ready for operation after power-up are: instant on, instant on after 50 ms, and configuration emulation. These modes are briefly described below.

- In instant on mode, the HardCopy Stratix device is available for use shortly after the device receives power. The on-chip POR circuit resets all registers. The `CONF_DONE` output is tri-stated once the POR has elapsed. No configuration device or configuration data is necessary.
- In instant on after 50 ms mode, the HardCopy Stratix device performs in a fashion similar to the instant on mode, except that there is an additional delay of 50 ms, during which time the device is held in reset stage. The `CONF_DONE` output is pulled low during this time, and then tri-stated after the 50 ms have elapsed. No configuration device or configuration data is necessary for this option.
- In configuration emulation mode, the HardCopy series device emulates the behavior of an APEX or Stratix FPGA during its configuration phase. When this mode is used, the HardCopy device uses a configuration emulation circuit to receive configuration bit streams. When all the configuration data is received, the HardCopy series device transitions into an initialization phase and releases the `CONF_DONE` pin to be pulled high. Pulling the `CONF_DONE` pin high signals that the HardCopy series device is ready for normal operation. If the optional open-drain `INIT_DONE` output is used, the normal operation is delayed until this signal is released by the HardCopy series device.



HardCopy II and some HardCopy Stratix devices do not support configuration emulation mode.

Instant on and instant on after 50 ms modes are the recommended power-up modes because these modes are similar to an ASIC's functionality upon power-up. No changes to the existing board design or the configuration software are required.

All three modes provide significant benefits to system designers. They enable seamless migration of the design from the FPGA device to the HardCopy device with no changes to the existing board design or the configuration software. The pull-up resistors on `nCONFIG`, `nSTATUS`, and `CONF_DONE` should be left on the printed circuit board.



For more information, refer to the *HardCopy Series Configuration Emulation* chapter in the *HardCopy Series Handbook*.

Hot Socketing

HardCopy Stratix devices support hot socketing without any external components. In a hot socketing situation, a device's output buffers are turned off during system power up or power down. To simplify board design, HardCopy Stratix devices support any power-up or power-down sequence (V_{CCIO} and V_{CCINT}). For mixed-voltage environments, you can

drive signals into the device before or during power up or power down without damaging the device. HardCopy Stratix devices do not drive out until they have attained proper operating conditions.

You can power up or power down the V_{CCIO} and V_{CCINT} pins in any sequence. The power supply ramp rates can range from 100 ns to 100 ms. During hot socketing, the I/O pin capacitance is less than 15 pF and the clock pin capacitance is less than 20 pF.

- The hot socketing DC specification is $|I_{IOPIN}| < 300 \mu A$.
- The hot socketing AC specification is $|I_{IOPIN}| < 8 \text{ mA}$ for 10 ns or less. This specification takes into account the pin capacitance only. Additional capacitance for trace, connector, and loading needs to be taken into consideration separately. I_{IOPIN} is the current at any user I/O pin on the device.



The DC specification applies when all V_{CC} supplies to the device are stable in the powered-up or powered-down conditions. For the AC specification, the peak current duration due to power-up transients is 10 ns or less.

HARDCOPY_ FPGA_ PROTOTYPE Devices

HARDCOPY_FPGA_PROTOTYPE devices are Stratix FPGAs available for designers to prototype their HardCopy Stratix designs and perform in-system verification before migration to a HardCopy Stratix device. The HARDCOPY_FPGA_PROTOTYPE devices have the same available resources as in the final HardCopy Stratix devices.

The Quartus II software version 4.1 and later contains the latest timing models. For designs with tight timing constraints, Altera strongly recommends compiling the design with the Quartus II software version 4.1 or later. To properly verify I/O features, it is important to design with the HARDCOPY_FPGA_PROTOTYPE device option prior to migrating to a HardCopy Stratix device.



Some `HARDCOPY_FPGA_PROTOTYPE` devices, as indicated in [Table 2–8](#), have fewer M-RAM blocks compared to the equivalent Stratix FPGAs. The selective removal of these resources provides a significant price benefit to designers using HardCopy Stratix devices.

Table 2–8. M-RAM Block Comparison Between Various Devices

Number of LEs	HARDCOPY_FPGA_PROTOTYPE Devices		HardCopy Stratix Devices		Stratix Devices	
	Device	M-RAM Blocks	Device	M-RAM Blocks	Device	M-RAM Blocks
25,660	EP1S25	2	HC1S25	2	EP1S25	2
32,470	EP1S30	2	HC1S30	2	EP1S30	4
41,250	EP1S40	2	HC1S40	2	EP1S40	4
57,120	EP1S60	6	HC1S60	6	EP1S60	6
79,040	EP1S830	6	HC1S830	6	EP1S830	9



For more information about how the various features in the Quartus II software can be used for designing HardCopy Stratix devices, refer to the *Quartus II Support for HardCopy Stratix Devices* chapter of the *HardCopy Series Handbook*.

`HARDCOPY_FPGA_PROTOTYPE` FPGA devices have the identical speed grade as the equivalent Stratix FPGAs. However, HardCopy Stratix devices are customized and do not have any speed grading. HardCopy Stratix devices, on an average, can be 50% faster than their equivalent `HARDCOPY_FPGA_PROTOTYPE` devices. The actual improvement is design-dependent.

Document Revision History

[Table 2–9](#) shows the revision history for this chapter.

Table 2–9. Document Revision History (Part 1 of 2)

Date and Document Version	Changes Made	Summary of Changes
September 2008 v3.4	Revised chapter number and metadata.	—
June 2007 v3.3	- Updated Table 2–1. - Added note to the “Embedded Memory” section. - Updated the “Hot Socketing” section.	—

Table 2–9. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
December 2006 v3.2	Updated revision history.	—
March 2006	Formerly chapter 6; no content change.	—
October 2005 v3.1	• Minor edits • Updated graphics	Minor edits.
May 2005 v3.0	• Added Table 6-1 • Added the Logic Elements section • Added the Embedded Memory section • Added the DSP Blocks section • Added the PLLs and Clock Networks section • Added the I/O Structure and Features section	Minor update.
January 2005 v2.0	• Added summary of I/O and timing differences between Stratix FPGAs and HardCopy Stratix devices • Removed section on Quartus II support of HardCopy Stratix devices • Added “Hot Socketing” section	Minor update.
August 2003 v1.1	Edited section headings’ hierarchy.	Minor edits.
June 2003 v1.0	Initial release of Chapter 6, Description, Architecture and Features, in the <i>HardCopy Device Handbook</i>	—

IEEE Std. 1149.1 (JTAG) Boundary-Scan Support

All HardCopy® Stratix® structured ASICs provide JTAG boundary-scan test (BST) circuitry that complies with the IEEE Std. 1149.1-1990 specification. The BST architecture offers the capability to efficiently test components on printed circuit boards (PCBs) with tight lead spacing by testing pin connections, without using physical test probes, and capturing functional data while a device is in normal operation. Boundary-scan cells in a device can force signals onto pins, or capture data from pin or core logic signals. Forced test data is serially shifted into the boundary-scan cells. Captured data is serially shifted out and externally compared to expected results.

A device using the JTAG interface uses four required pins, TDI, TDO, TMS, and TCK, and one optional pin, TRST. HardCopy Stratix devices support the JTAG instructions as shown in [Table 3-1](#).

Table 3-1. HardCopy Stratix JTAG Instructions (Part 1 of 2)

JTAG Instruction	Instruction Code	Description
SAMPLE/PRELOAD	00 0000 0101	Allows a snapshot of signals at the device pins to be captured and examined during normal device operation, and permits an initial data pattern to be output at the device pins.
EXTEST (1)	00 0000 0000	Allows the external circuitry and board-level interconnects to be tested by forcing a test pattern at the output pins and capturing test results at the input pins.
BYPASS	11 1111 1111	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation.
USERCODE	00 0000 0111	Selects the 32-bit USERCODE register and places it between the TDI and TDO pins, allowing the USERCODE to be serially shifted out of TDO.
IDCODE	00 0000 0110	Selects the IDCODE register and places it between TDI and TDO, allowing the IDCODE to be serially shifted out of TDO.
HIGHZ (1)	00 0000 1011	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation, while tri-stating all of the I/O pins.

Table 3–1. HardCopy Stratix JTAG Instructions (Part 2 of 2)

JTAG Instruction	Instruction Code	Description
CLAMP (1)	00 0000 1010	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation while holding I/O pins to a state defined by the data in the boundary-scan register.

Note to Table 3–1:

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



The boundary-scan description language (BSDL) files for HardCopy Stratix devices are different from the corresponding Stratix FPGAs. The BSDL files for HardCopy Stratix devices are available for download from the Altera website at www.altera.com.

The HardCopy Stratix device instruction register length is 10 bits; the USERCODE register length is 32 bits. The USERCODE registers are mask-programmed, so they are not re-programmable. The designer can choose an appropriate 32-bit sequence to program into the USERCODE registers.

Tables 3–2 and 3–3 show the boundary-scan register length and device IDCODE information for HardCopy Stratix devices.

Table 3–2. HardCopy Stratix Boundary-Scan Register Length

Device	Maximum Boundary-Scan Register Length
HC1S25 672-pin FineLine BGA	1,458
HC1S30 780-pin FineLine BGA	1,878
HC1S40 780-pin FineLine BGA	1,878
HC1S60 1,020-pin FineLine BGA	2,382
HC1S80 1,020-pin FineLine BGA	2,382

Table 3–3. 32-Bit HardCopy Stratix Device IDCODE

Device	IDCODE (32 Bits) (1)			
	Version (4 Bits)	Part Number (16 Bits)	Manufacturer Identity (11 Bits)	LSB (1 Bit) (2)
HC1S25	0000	0010 0000 0000 0011	000 0110 1110	1
HC1S30	0000	0010 0000 0000 0100	000 0110 1110	1
HC1S40	0000	0010 0000 0000 0101	000 0110 1110	1
HC1S60	0000	0010 0000 0000 0110	000 0110 1110	1
HC1S80	0000	0010 0000 0000 0111	000 0110 1110	1

Notes to Table 3–3:

- (1) The most significant bit (MSB) is on the left.
 (2) The IDCODE's least significant bit (LSB) is always 1.

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

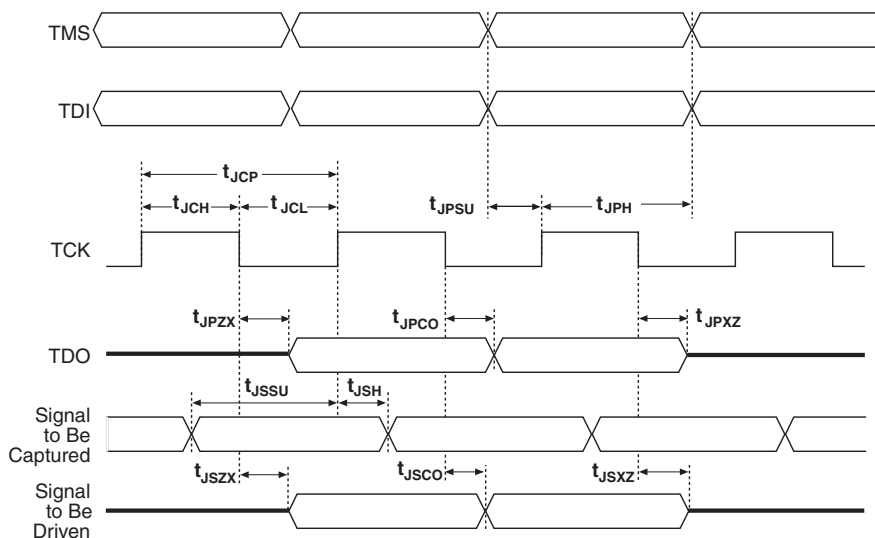
Figure 3–1. HardCopy Stratix JTAG Waveforms

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

Table 3–4. HardCopy Stratix JTAG Timing Parameters and Values				
Symbol	Parameter	Min	Max	Unit
t_{JCP}	TCK clock period	100		ns
t_{JCH}	TCK clock high time	50		ns
t_{JCL}	TCK clock low time	50		ns
t_{JPSU}	JTAG port setup time	20		ns
t_{JPH}	JTAG port hold time	45		ns
t_{JPCO}	JTAG port clock to output		25	ns
t_{JPZX}	JTAG port high impedance to valid output		25	ns
t_{JPXZ}	JTAG port valid output to high impedance		25	ns
t_{JSSU}	Capture register setup time	20		ns
t_{JSH}	Capture register hold time	45		ns
t_{JSCO}	Update register clock to output		35	ns
t_{JSZX}	Update register high impedance to valid output		35	ns
t_{JSXZ}	Update register valid output to high impedance		35	ns



For more information on JTAG, refer to *AN 39: IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices*.

Document Revision History

Table 3–5 shows the revision history for this chapter.

Table 3–5. Document Revision History (Part 1 of 2)		
Date and Document Version	Changes Made	Summary of Changes
September 2008 v3.4	Updated chapter number and metadata.	—
June 2007 v3.3	Updated Figure 3–1.	—
December 2006 v3.2	Updated revision history.	—
March 2006	Formerly chapter 7; no content change.	—

Table 3–5. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
October 2005 v3.1	• Minor edits • Graphic updates	—
May 2005 v3.0	Updated “IEEE Std. 1149.1 (JTAG) Boundary-Scan Support” section	
January 2005 v2.0	Added information about <code>USERCODE</code> registers	
June 2003 v1.0	Initial release of Chapter 7, Boundary-Scan Support, in the <i>HardCopy Device Handbook</i>	

Recommended Operating Conditions

Tables 4–1 through 4–3 provide information on absolute maximum ratings, recommended operating conditions, DC operating conditions, and capacitance for 1.5-V HardCopy® Stratix® devices.

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

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

Table 4–2. HardCopy Stratix Device Recommended Operating Conditions

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCINT}	Supply voltage for internal logic and input buffers	(4)	1.425	1.575	V
V_{CCIO}	Supply voltage for output buffers, 3.3-V operation	(4), (5)	3.00 (3.135)	3.60 (3.465)	V
	Supply voltage for output buffers, 2.5-V operation	(4)	2.375	2.625	V
	Supply voltage for output buffers, 1.8-V operation	(4)	1.71	1.89	V
	Supply voltage for output buffers, 1.5-V operation	(4)	1.4	1.6	V
V_I	Input voltage	(3), (6)	–0.5	4.1	V
V_O	Output voltage		0	V_{CCIO}	V
T_J	Operating junction temperature	For commercial use	0	85	°C
		For industrial use	–40	100	°C

Table 4–3. HardCopy Stratix Device DC Operating Conditions *Note (7)*

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
I_I	Input pin leakage current	$V_I = V_{CCIOmax}$ to 0 V (8)	–10		10	μA
I_{OZ}	Tri-stated I/O pin leakage current	$V_O = V_{CCIOmax}$ to 0 V (8)	–10		10	μA
I_{CC0}	V_{CC} supply current (standby) (All memory blocks in power-down mode)	V_I = ground, no load, no toggling inputs				mA
R_{CONF}	Value of I/O pin pull-up resistor before and during configuration	$V_i=0$; $V_{CCIO} = 3.3$ V (9)	15	25	50	$k\Omega$
		$V_i=0$; $V_{CCIO} = 2.5$ V (9)	20	45	70	$k\Omega$
		$V_i=0$; $V_{CCIO} = 1.8$ V (9)	30	65	100	$k\Omega$
		$V_i=0$; $V_{CCIO} = 1.5$ V (9)	50	100	150	$k\Omega$
	Recommended value of I/O pin external pull-down resistor before and during configuration			1	2	$k\Omega$

Notes to Tables 4–1 through 4–3:

- Refer to the *Operating Requirements for Altera Devices* Data Sheet.
- Conditions beyond those listed in Table 4–1 may cause permanent damage to a device. Additionally, device operation at the absolute maximum ratings for extended periods of time may have adverse effects on the device.
- Minimum DC input is –0.5 V. During transitions, the inputs may undershoot to –2 V or overshoot to 4.6 V for input currents less than 100 mA and periods shorter than 20 ns.
- Maximum V_{CC} rise time is 100 ms, and V_{CC} must rise monotonically.
- V_{CCIO} maximum and minimum conditions for LVPECL, LVDS, and 3.3-V PCML are shown in parentheses.
- All pins, including dedicated inputs, clock, I/O, and JTAG pins, may be driven before V_{CCINT} and V_{CCIO} are powered.
- Typical values are for $T_A = 25$ °C, $V_{CCINT} = 1.5$ V, and $V_{CCIO} = 1.5$ V, 1.8 V, 2.5 V, and 3.3 V.
- This value is specified for normal device operation. The value may vary during power up. This applies for all V_{CCIO} settings (3.3, 2.5, 1.8, and 1.5 V).
- Pin pull-up resistance values will be lower if an external source drives the pin higher than V_{CCIO} .

Tables 4–4 through 4–31 list the DC operating specifications for the supported I/O standards. These tables list minimal specifications only; HardCopy Stratix devices may exceed these specifications. Table 4–32 provides information on capacitance for 1.5-V HardCopy Stratix devices.

Table 4–4. LVTTTL Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.6	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
V_{OH}	High-level output voltage	$I_{OH} = -4$ to -24 mA (1)	2.4		V
V_{OL}	Low-level output voltage	$I_{OL} = 4$ to 24 mA (1)		0.45	V

Table 4–5. LVCMOS Specifications

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

Table 4–6. 2.5-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		2.375	2.625	V
V_{IH}	High-level input voltage		1.7	4.1	V
V_{IL}	Low-level input voltage		–0.5	0.7	V
V_{OH}	High-level output voltage	$I_{OH} = -0.1$ mA	2.1		V
		$I_{OH} = -1$ mA	2.0		V
		$I_{OH} = -2$ to -16 mA (1)	1.7		V
V_{OL}	Low-level output voltage	$I_{OL} = 0.1$ mA		0.2	V
		$I_{OL} = 1$ mA		0.4	V
		$I_{OL} = 2$ to 16 mA (1)		0.7	V

Table 4–7. 1.8-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.95	V
V_{IH}	High-level input voltage		$0.65 \times V_{CCIO}$	2.25	V
V_{IL}	Low-level input voltage		–0.3	$0.35 \times V_{CCIO}$	V
V_{OH}	High-level output voltage	$I_{OH} = -2$ to -8 mA (1)	$V_{CCIO} - 0.45$		V
V_{OL}	Low-level output voltage	$I_{OL} = 2$ to 8 mA (1)		0.45	V

Table 4–8. 1.5-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Unit
V_{CCIO}	Output supply voltage		1.4	1.6	V
V_{IH}	High-level input voltage		$0.65 \times V_{CCIO}$	$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		–0.3	$0.35 \times V_{CCIO}$	V
V_{OH}	High-level output voltage	$I_{OH} = -2$ mA (1)	$0.75 \times V_{CCIO}$		V
V_{OL}	Low-level output voltage	$I_{OL} = 2$ mA (1)		$0.25 \times V_{CCIO}$	V

Table 4–9. 3.3-V LVDS I/O Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	I/O supply voltage		3.135	3.3	3.465	V
V_{ID}	Input differential voltage swing	$0.1 \text{ V} < V_{CM} < 1.1 \text{ V}$ $J = 1$ through 10	300		1,000	mV
		$1.1 \text{ V} \leq V_{CM} \leq 1.6 \text{ V}$ $J = 1$	200		1,000	mV
		$1.1 \text{ V} \leq V_{CM} \leq 1.6 \text{ V}$ $J = 2$ through 10	100		1,000	mV
		$1.6 \text{ V} < V_{CM} < 1.8 \text{ V}$ $J = 1$ through 10	300		1,000	mV

Table 4–9. 3.3-V LVDS I/O Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{ICM}	Input common mode voltage	LVDS $0.3\text{ V} < V_{ID} < 1.0\text{ V}$ $J = 1$ through 10	100		1,100	mV
		LVDS $0.3\text{ V} < V_{ID} < 1.0\text{ V}$ $J = 1$ through 10	1,600		1,800	mV
		LVDS $0.2\text{ V} < V_{ID} < 1.0\text{ V}$ $J = 1$	1,100		1,600	mV
		LVDS $0.1\text{ V} < V_{ID} < 1.0\text{ V}$ $J = 2$ through 10	1,100		1,600	mV
$V_{OD} (2)$	Output differential voltage	$R_L = 100\ \Omega$	250	375	550	mV
ΔV_{OD}	Change in V_{OD} between high and low	$R_L = 100\ \Omega$			50	mV
V_{OCM}	Output common mode voltage	$R_L = 100\ \Omega$	1,125	1,200	1,375	mV
ΔV_{OCM}	Change in V_{OCM} between high and low	$R_L = 100\ \Omega$			50	mV
R_L	Receiver differential input resistor		90	100	110	Ω

Table 4–10. 3.3-V PCML Specifications

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

Table 4–11. LVPECL Specifications

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

Table 4–12. HyperTransport Technology Specifications

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

Table 4–13. 3.3-V PCI Specifications

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

Table 4–14. PCI-X 1.0 Specifications

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

Table 4–15. GTL+ I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{TT}	Termination voltage		1.35	1.5	1.65	V
V_{REF}	Reference voltage		0.88	1.0	1.12	V
V_{IH}	High-level input voltage		$V_{REF} + 0.1$			V
V_{IL}	Low-level input voltage				$V_{REF} - 0.1$	V
V_{OL}	Low-level output voltage	$I_{OL} = 34 \text{ mA}$ (1)			0.65	V

Table 4–16. GTL I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{TT}	Termination voltage		1.14	1.2	1.26	V
V_{REF}	Reference voltage		0.74	0.8	0.86	V
V_{IH}	High-level input voltage		$V_{REF} + 0.05$			V
V_{IL}	Low-level input voltage				$V_{REF} - 0.05$	V
V_{OL}	Low-level output voltage	$I_{OL} = 40 \text{ mA}$ (1)			0.4	V

Table 4–17. SSTL-18 Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.8	1.95	V
V_{REF}	Reference voltage		0.8	0.9	1.0	V
V_{TT}	Termination voltage		$V_{REF} - 0.04$	V_{REF}	$V_{REF} + 0.04$	V
$V_{IH(DC)}$	High-level DC input voltage		$V_{REF} + 0.125$			V
$V_{IL(DC)}$	Low-level DC input voltage				$V_{REF} - 0.125$	V
$V_{IH(AC)}$	High-level AC input voltage		$V_{REF} + 0.275$			V
$V_{IL(AC)}$	Low-level AC input voltage				$V_{REF} - 0.275$	V
V_{OH}	High-level output voltage	$I_{OH} = -6.7 \text{ mA}$ (1)	$V_{TT} + 0.475$			V
V_{OL}	Low-level output voltage	$I_{OL} = 6.7 \text{ mA}$ (1)			$V_{TT} - 0.475$	V

Table 4–18. SSTL-18 Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.8	1.95	V
V_{REF}	Reference voltage		0.8	0.9	1.0	V
V_{TT}	Termination voltage		$V_{REF} - 0.04$	V_{REF}	$V_{REF} + 0.04$	V
$V_{IH(DC)}$	High-level DC input voltage		$V_{REF} + 0.125$			V
$V_{IL(DC)}$	Low-level DC input voltage				$V_{REF} - 0.125$	V
$V_{IH(AC)}$	High-level AC input voltage		$V_{REF} + 0.275$			V
$V_{IL(AC)}$	Low-level AC input voltage				$V_{REF} - 0.275$	V
V_{OH}	High-level output voltage	$I_{OH} = -13.4 \text{ mA}$ (1)	$V_{TT} + 0.630$			V
V_{OL}	Low-level output voltage	$I_{OL} = 13.4 \text{ mA}$ (1)			$V_{TT} - 0.630$	V

Table 4–19. SSTL-2 Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		2.375	2.5	2.625	V
V_{TT}	Termination voltage		$V_{REF} - 0.04$	V_{REF}	$V_{REF} + 0.04$	V
V_{REF}	Reference voltage		1.15	1.25	1.35	V
$V_{IH(DC)}$	High-level DC input voltage		$V_{REF} + 0.18$		3.0	V
$V_{IL(DC)}$	Low-level DC input voltage		-0.3		$V_{REF} - 0.18$	V
$V_{IH(AC)}$	High-level AC input voltage		$V_{REF} + 0.35$			V
$V_{IL(AC)}$	Low-level AC input voltage				$V_{REF} - 0.35$	V
V_{OH}	High-level output voltage	$I_{OH} = -8.1 \text{ mA}$ (1)	$V_{TT} + 0.57$			V
V_{OL}	Low-level output voltage	$I_{OL} = 8.1 \text{ mA}$ (1)			$V_{TT} - 0.57$	V

Table 4–20. SSTL-2 Class II Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		2.375	2.5	2.625	V
V_{TT}	Termination voltage		$V_{REF} - 0.04$	V_{REF}	$V_{REF} + 0.04$	V

Table 4–20. SSTL-2 Class II Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{REF}	Reference voltage		1.15	1.25	1.35	V
$V_{IH(DC)}$	High-level DC input voltage		$V_{REF} + 0.18$		$V_{CCIO} + 0.3$	V
$V_{IL(DC)}$	Low-level DC input voltage		–0.3		$V_{REF} - 0.18$	V
$V_{IH(AC)}$	High-level AC input voltage		$V_{REF} + 0.35$			V
$V_{IL(AC)}$	Low-level AC input voltage				$V_{REF} - 0.35$	V
V_{OH}	High-level output voltage	$I_{OH} = -16.4 \text{ mA}$ (1)	$V_{TT} + 0.76$			V
V_{OL}	Low-level output voltage	$I_{OL} = 16.4 \text{ mA}$ (1)			$V_{TT} - 0.76$	V

Table 4–21. SSTL-3 Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.3	3.6	V
V_{TT}	Termination voltage		$V_{REF} - 0.05$	V_{REF}	$V_{REF} + 0.05$	V
V_{REF}	Reference voltage		1.3	1.5	1.7	V
$V_{IH(DC)}$	High-level DC input voltage		$V_{REF} + 0.2$		$V_{CCIO} + 0.3$	V
$V_{IL(DC)}$	Low-level DC input voltage		–0.3		$V_{REF} - 0.2$	V
$V_{IH(AC)}$	High-level AC input voltage		$V_{REF} + 0.4$			V
$V_{IL(AC)}$	Low-level AC input voltage				$V_{REF} - 0.4$	V
V_{OH}	High-level output voltage	$I_{OH} = -8 \text{ mA}$ (1)	$V_{TT} + 0.6$			V
V_{OL}	Low-level output voltage	$I_{OL} = 8 \text{ mA}$ (1)			$V_{TT} - 0.6$	V

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

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.0	3.3	3.6	V
V_{TT}	Termination voltage		$V_{REF} - 0.05$	V_{REF}	$V_{REF} + 0.05$	V
V_{REF}	Reference voltage		1.3	1.5	1.7	V

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

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{IH(DC)}$	High-level DC input voltage		$V_{REF} + 0.2$		$V_{CCIO} + 0.3$	V
$V_{IL(DC)}$	Low-level DC input voltage		-0.3		$V_{REF} - 0.2$	V
$V_{IH(AC)}$	High-level AC input voltage		$V_{REF} + 0.4$			V
$V_{IL(AC)}$	Low-level AC input voltage				$V_{REF} - 0.4$	V
V_{OH}	High-level output voltage	$I_{OH} = -16 \text{ mA}$ (1)	$V_{TT} + 0.8$			V
V_{OL}	Low-level output voltage	$I_{OL} = 16 \text{ mA}$ (1)			$V_{TT} - 0.8$	V

Table 4–23. 3.3-V AGP 2× Specifications

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

Table 4–24. 3.3-V AGP 1× Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		3.15	3.3	3.45	V
V_{IH}	High-level input voltage (4)		$0.5 \times V_{CCIO}$		$V_{CCIO} + 0.5$	V
V_{IL}	Low-level input voltage (4)				$0.3 \times V_{CCIO}$	V
V_{OH}	High-level output voltage	$I_{OUT} = -0.5 \text{ mA}$	$0.9 \times V_{CCIO}$		3.6	V
V_{OL}	Low-level output voltage	$I_{OUT} = 1.5 \text{ mA}$			$0.1 \times V_{CCIO}$	V

Table 4–25. 1.5-V HSTL Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.4	1.5	1.6	V
V_{REF}	Input reference voltage		0.68	0.75	0.9	V
V_{TT}	Termination voltage		0.7	0.75	0.8	V
V_{IH} (DC)	DC high-level input voltage		$V_{REF} + 0.1$			V
V_{IL} (DC)	DC low-level input voltage		–0.3		$V_{REF} - 0.1$	V
V_{IH} (AC)	AC high-level input voltage		$V_{REF} + 0.2$			V
V_{IL} (AC)	AC low-level input voltage				$V_{REF} - 0.2$	V
V_{OH}	High-level output voltage	$I_{OH} = 8 \text{ mA}$ (1)	$V_{CCIO} - 0.4$			V
V_{OL}	Low-level output voltage	$I_{OH} = -8 \text{ mA}$ (1)			0.4	V

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

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.4	1.5	1.6	V
V_{REF}	Input reference voltage		0.68	0.75	0.9	V
V_{TT}	Termination voltage		0.7	0.75	0.8	V
V_{IH} (DC)	DC high-level input voltage		$V_{REF} + 0.1$			V
V_{IL} (DC)	DC low-level input voltage		–0.3		$V_{REF} - 0.1$	V
V_{IH} (AC)	AC high-level input voltage		$V_{REF} + 0.2$			V
V_{IL} (AC)	AC low-level input voltage				$V_{REF} - 0.2$	V
V_{OH}	High-level output voltage	$I_{OH} = 16 \text{ mA}$ (1)	$V_{CCIO} - 0.4$			V
V_{OL}	Low-level output voltage	$I_{OH} = -16 \text{ mA}$ (1)			0.4	V

Table 4–27. 1.8-V HSTL Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.80	1.95	V
V_{REF}	Input reference voltage		0.70	0.90	0.95	V
V_{TT}	Termination voltage			$V_{CCIO} \times 0.5$		V
$V_{IH} (DC)$	DC high-level input voltage		$V_{REF} + 0.1$			V
$V_{IL} (DC)$	DC low-level input voltage		–0.5		$V_{REF} - 0.1$	V
$V_{IH} (AC)$	AC high-level input voltage		$V_{REF} + 0.2$			V
$V_{IL} (AC)$	AC low-level input voltage				$V_{REF} - 0.2$	V
V_{OH}	High-level output voltage	$I_{OH} = 8 \text{ mA}$ (1)	$V_{CCIO} - 0.4$			V
V_{OL}	Low-level output voltage	$I_{OH} = -8 \text{ mA}$ (1)			0.4	V

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

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
V_{CCIO}	Output supply voltage		1.65	1.80	1.95	V
V_{REF}	Input reference voltage		0.70	0.90	0.95	V
V_{TT}	Termination voltage			$V_{CCIO} \times 0.5$		V
$V_{IH} (DC)$	DC high-level input voltage		$V_{REF} + 0.1$			V
$V_{IL} (DC)$	DC low-level input voltage		–0.5		$V_{REF} - 0.1$	V
$V_{IH} (AC)$	AC high-level input voltage		$V_{REF} + 0.2$			V
$V_{IL} (AC)$	AC low-level input voltage				$V_{REF} - 0.2$	V
V_{OH}	High-level output voltage	$I_{OH} = 16 \text{ mA}$ (1)	$V_{CCIO} - 0.4$			V
V_{OL}	Low-level output voltage	$I_{OH} = -16 \text{ mA}$ (1)			0.4	V

Table 4–29. 1.5-V Differential HSTL Specifications

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

Table 4–30. CTT I/O Specifications

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

Table 4–31. Bus Hold Parameters

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

Table 4–32. Stratix Device Capacitance *Note (5)*

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

Notes to Tables 4–4 through 4–32:

- (1) Drive strength is programmable according to values in the *Stratix Architecture* chapter of the *Stratix Device Handbook*.
- (2) When the tx_outclock port of the alt1vds_tx megafunction is 717 MHz, V_{OD(min)} = 235 mV on the output clock pin.
- (3) Pin pull-up resistance values will lower if an external source drives the pin higher than V_{CCIO}.
- (4) V_{REF} specifies the center point of the switching range.
- (5) Capacitance is sample-tested only. Capacitance is measured using time-domain reflections (TDR). Measurement accuracy is within ±0.5 pF.

Power Consumption

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

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

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

Timing Closure

The timing numbers in Tables 4–34 to 4–43 are only provided as an indication of allowable timing for HardCopy Stratix devices. The Quartus II software provides preliminary timing information for HardCopy Stratix designs, which can be used as an estimation of the device performance.

The final timing numbers and actual performance for each HardCopy Stratix design is available when the design migration is complete and are subject to verification and approval by Altera and the designer during the HardCopy Design review process.

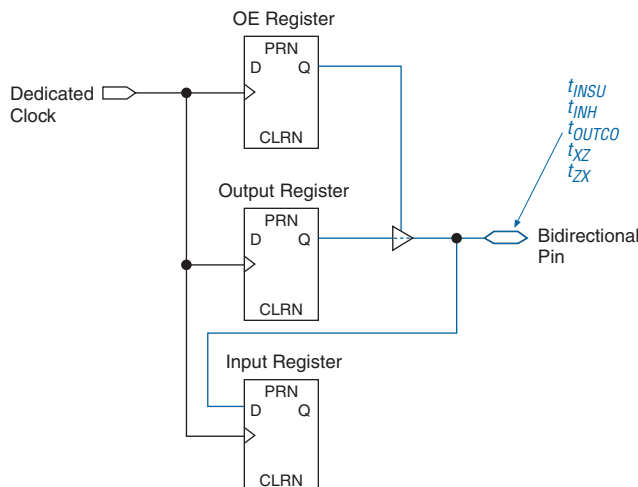


For more information, refer to the *HardCopy Series Back-End Timing Closure* chapter in the *HardCopy Series Handbook*.

External Timing Parameters

External timing parameters are specified by device density and speed grade. Figure 4–1 shows the pin-to-pin timing model for bidirectional IOE pin timing. All registers are within the IOE.

Figure 4–1. External Timing in HardCopy Stratix Devices



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

Table 4–33 shows the external I/O timing parameters when using global clock networks.

Table 4–33. HardCopy Stratix Global Clock External I/O Timing Parameters <i>Notes (1), (2)</i>	
Symbol	Parameter
t_{INSU}	Setup time for input or bidirectional pin using IOE input register with global clock fed by CLK pin
t_{INH}	Hold time for input or bidirectional pin using IOE input register with global clock fed by CLK pin
t_{OUTCO}	Clock-to-output delay output or bidirectional pin using IOE output register with global clock fed by CLK pin
t_{INSUPLL}	Setup time for input or bidirectional pin using IOE input register with global clock fed by Enhanced PLL with default phase setting
t_{INHPLL}	Hold time for input or bidirectional pin using IOE input register with global clock fed by Enhanced PLL with default phase setting
t_{OUTCOPLL}	Clock-to-output delay output or bidirectional pin using IOE output register with global clock Enhanced PLL with default phase setting
t_{XZPLL}	Synchronous IOE output enable register to output pin disable delay using global clock fed by Enhanced PLL with default phase setting
t_{ZXPLL}	Synchronous IOE output enable register to output pin enable delay using global clock fed by Enhanced PLL with default phase setting

Notes to Table 4–33:

- (1) These timing parameters are sample-tested only.
- (2) These timing parameters are for column and row IOE pins. Designers should use the Quartus II software to verify the external timing for any pin.

HardCopy Stratix External I/O Timing

These timing parameters are for both column IOE and row IOE pins. In HC1S30 devices and above, designers can decrease the t_{SU} time by using FPLLCLK , but may get positive hold time in HC1S60 and HC1S80 devices. Designers should use the Quartus II software to verify the external devices for any pin.

Tables 4–34 through 4–35 show the external timing parameters on column and row pins for HC1S25 devices.

Table 4–34. HC1S25 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	1.371		ns
t_{INH}	0.000		ns
t_{OUTCO}	2.809	7.155	ns
t_{xZ}	2.749	7.040	ns
t_{ZX}	2.749	7.040	ns
t_{INSUPLL}	1.271		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.124	2.602	ns
t_{xZPLL}	1.064	2.487	ns
t_{ZXPLL}	1.064	2.487	ns

Table 4–35. HC1S25 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	1.665		ns
t_{INH}	0.000		ns
t_{OUTCO}	2.834	7.194	ns
t_{xZ}	2.861	7.276	ns
t_{ZX}	2.861	7.276	ns
t_{INSUPLL}	1.538		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.164	2.653	ns
t_{xZPLL}	1.191	2.735	ns
t_{ZXPLL}	1.191	2.735	ns

Tables 4–36 through 4–37 show the external timing parameters on column and row pins for HC1S30 devices.

Table 4–36. HC1S30 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	1.935		ns
t_{INH}	0.000		ns
t_{OUTCO}	2.814	7.274	ns
t_{XZ}	2.754	7.159	ns
t_{ZX}	2.754	7.159	ns
t_{INSUPLL}	1.265		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.068	2.423	ns
t_{XZPLL}	1.008	2.308	ns
t_{ZXPLL}	1.008	2.308	ns

Table 4–37. HC1S30 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	1.995		ns
t_{INH}	0.000		ns
t_{OUTCO}	2.917	7.548	ns
t_{XZ}	2.944	7.630	ns
t_{ZX}	2.944	7.630	ns
t_{INSUPLL}	1.337		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.164	2.672	ns
t_{XZPLL}	1.191	2.754	ns
t_{ZXPLL}	1.191	2.754	ns

Tables 4–38 through 4–39 show the external timing parameters on column and row pins for HC1S40 devices.

Table 4–38. HC1S40 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	2.126		ns
t_{INH}	0.000		ns
t_{OUTCO}	2.856	7.253	ns
t_{XZ}	2.796	7.138	ns
t_{ZX}	2.796	7.138	ns
t_{INSUPLL}	1.466		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.092	2.473	ns
t_{XZPLL}	1.032	2.358	ns
t_{ZXPLL}	1.032	2.358	ns

Table 4–39. HC1S40 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	2.020		ns
t_{INH}	0.000		ns
t_{OUTCO}	2.912	7.480	ns
t_{XZ}	2.939	7.562	ns
t_{ZX}	2.939	7.562	ns
t_{INSUPLL}	1.370		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.144	2.693	ns
t_{XZPLL}	1.171	2.775	ns
t_{ZXPLL}	1.171	2.775	ns

Tables 4–40 through 4–41 show the external timing parameters on column and row pins for HC1S60 devices.

Table 4–40. HC1S60 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	2.000		ns
t_{INH}	0.000		ns
t_{OUTCO}	3.051	6.977	ns
t_{xZ}	2.991	6.853	ns
t_{ZX}	2.991	6.853	ns
t_{INSUPLL}	1.315		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.029	2.323	ns
t_{xZPLL}	0.969	2.199	ns
t_{ZXPLL}	0.969	2.199	ns

Table 4–41. HC1S60 External I/O Timing on Row Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	2.232		ns
t_{INH}	0.000		ns
t_{OUTCO}	3.182	7.286	ns
t_{xZ}	3.209	7.354	ns
t_{ZX}	3.209	7.354	ns
t_{INSUPLL}	1.651		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.154	2.622	ns
t_{xZPLL}	1.181	2.690	ns
t_{ZXPLL}	1.181	2.690	ns

Tables 4–42 through 4–43 show the external timing parameters on column and row pins for HC1S80 devices.

Table 4–42. HC1S80 External I/O Timing on Column Pins Using Global Clock Networks

Parameter	Performance		Unit
	Min	Max	
t_{INSU}	0.884		ns
t_{INH}	0.000		ns
t_{OUTCO}	3.267	7.415	ns
t_{XZ}	3.207	7.291	ns
t_{ZX}	3.207	7.291	ns
t_{INSUPLL}	0.506		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.635	2.828	ns
t_{XZPLL}	1.575	2.704	ns
t_{ZXPLL}	1.575	2.704	ns

Table 4–43. HC1S80 External I/O Timing on Rows Using Pin Global Clock Networks

Symbol	Performance		Unit
	Min	Max	
t_{INSU}	1.362		ns
t_{INH}	0.000		ns
t_{OUTCO}	3.457	7.859	ns
t_{XZ}	3.484	7.927	ns
t_{ZX}	3.484	7.927	ns
t_{INSUPLL}	0.994		ns
t_{INHPLL}	0.000		ns
t_{OUTCOPLL}	1.821	3.254	ns
t_{XZPLL}	1.848	3.322	ns
t_{ZXPLL}	1.848	3.322	ns

Maximum Input and Output Clock Rates

Tables 4–44 through 4–46 show the maximum input clock rate for column and row pins in HardCopy Stratix devices.

Table 4–44. HardCopy Stratix Maximum Input Clock Rate for CLK[7..4] and CLK[15..12] Pins

I/O Standard	Performance	Unit
LVTTL	422	MHz
2.5 V	422	MHz
1.8 V	422	MHz
1.5 V	422	MHz
LVC MOS	422	MHz
GTL	300	MHz
GTL+	300	MHz
SSTL-3 class I	400	MHz
SSTL-3 class II	400	MHz
SSTL-2 class I	400	MHz
SSTL-2 class II	400	MHz
SSTL-18 class I	400	MHz
SSTL-18 class II	400	MHz
1.5-V HSTL class I	400	MHz
1.5-V HSTL class II	400	MHz
1.8-V HSTL class I	400	MHz
1.8-V HSTL class II	400	MHz
3.3-V PCI	422	MHz
3.3-V PCI-X 1.0	422	MHz
Compact PCI	422	MHz
AGP 1×	422	MHz
AGP 2×	422	MHz
CTT	300	MHz
Differential HSTL	400	MHz
LVPECL (1)	645	MHz
PCML (1)	300	MHz
LVDS (1)	645	MHz
HyperTransport technology (1)	500	MHz

Table 4–45. HardCopy Stratix Maximum Input Clock Rate for CLK[0, 2, 9, 11] Pins and FPLL[10..7]CLK Pins

I/O Standard	Performance	Unit
LVTTL	422	MHz
2.5 V	422	MHz
1.8 V	422	MHz
1.5 V	422	MHz
LVC MOS	422	MHz
GTL	300	MHz
GTL+	300	MHz
SSTL-3 class I	400	MHz
SSTL-3 class II	400	MHz
SSTL-2 class I	400	MHz
SSTL-2 class II	400	MHz
SSTL-18 class I	400	MHz
SSTL-18 class II	400	MHz
1.5-V HSTL class I	400	MHz
1.5-V HSTL class II	400	MHz
1.8-V HSTL class I	400	MHz
1.8-V HSTL class II	400	MHz
3.3-V PCI	422	MHz
3.3-V PCI-X 1.0	422	MHz
Compact PCI	422	MHz
AGP 1×	422	MHz
AGP 2×	422	MHz
CTT	300	MHz
Differential HSTL	400	MHz
LVPECL (1)	717	MHz
PCML (1)	400	MHz
LVDS (1)	717	MHz
HyperTransport technology (1)	717	MHz

Table 4–46. HardCopy Stratix Maximum Input Clock Rate for CLK[1, 3, 8, 10] Pins

I/O Standard	Performance	Unit
LVTTL	422	MHz
2.5 V	422	MHz
1.8 V	422	MHz
1.5 V	422	MHz
LVC MOS	422	MHz
GTL	300	MHz
GTL+	300	MHz
SSTL-3 class I	400	MHz
SSTL-3 class II	400	MHz
SSTL-2 class I	400	MHz
SSTL-2 class II	400	MHz
SSTL-18 class I	400	MHz
SSTL-18 class II	400	MHz
1.5-V HSTL class I	400	MHz
1.5-V HSTL class II	400	MHz
1.8-V HSTL class I	400	MHz
1.8-V HSTL class II	400	MHz
3.3-V PCI	422	MHz
3.3-V PCI-X 1.0	422	MHz
Compact PCI	422	MHz
AGP 1×	422	MHz
AGP 2×	422	MHz
CTT	300	MHz
Differential HSTL	400	MHz
LVPECL (1)	645	MHz
PCML (1)	300	MHz
LVDS (1)	645	MHz
HyperTransport technology (1)	500	MHz

Note to Tables 4–44 through 4–46:

(1) These parameters are only available on row I/O pins.

Tables 4–47 through 4–48 show the maximum output clock rate for column and row pins in HardCopy Stratix devices.

Table 4–47. HardCopy Stratix Maximum Output Clock Rate for PLL[5, 6, 11, 12] Pins (Part 1 of 2)

I/O Standard	Performance	Unit
LVTTTL	350	MHz
2.5 V	350	MHz
1.8 V	250	MHz
1.5 V	225	MHz
LVC MOS	350	MHz
GTL	200	MHz
GTL+	200	MHz
SSTL-3 class I	200	MHz
SSTL-3 class II	200	MHz
SSTL-2 class I (3)	200	MHz
SSTL-2 class I (4)	200	MHz
SSTL-2 class I (5)	150	MHz
SSTL-2 class II (3)	200	MHz
SSTL-2 class II (4)	200	MHz
SSTL-2 class II (5)	150	MHz
SSTL-18 class I	150	MHz
SSTL-18 class II	150	MHz
1.5-V HSTL class I	250	MHz
1.5-V HSTL class II	225	MHz
1.8-V HSTL class I	250	MHz
1.8-V HSTL class II	225	MHz
3.3-V PCI	350	MHz
3.3-V PCI-X 1.0	350	MHz
Compact PCI	350	MHz
AGP 1×	350	MHz
AGP 2×	350	MHz
CTT	200	MHz
Differential HSTL	225	MHz
Differential SSTL-2 (6)	200	MHz
LVPECL (2)	500	MHz
PCML (2)	350	MHz

Table 4–47. HardCopy Stratix Maximum Output Clock Rate for PLL[5, 6, 11, 12] Pins (Part 2 of 2)

I/O Standard	Performance	Unit
LVDS (2)	500	MHz
HyperTransport technology (2)	350	MHz

Table 4–48. HardCopy Stratix Maximum Output Clock Rate (Using I/O Pins) for PLL[1, 2, 3, 4] Pins (Part 1 of 2)

I/O Standard	Performance	Unit
LVTTL	400	MHz
2.5 V	400	MHz
1.8 V	400	MHz
1.5 V	350	MHz
LVCMOS	400	MHz
GTL	200	MHz
GTL+	200	MHz
SSTL-3 class I	167	MHz
SSTL-3 class II	167	MHz
SSTL-2 class I	150	MHz
SSTL-2 class II	150	MHz
SSTL-18 class I	150	MHz
SSTL-18 class II	150	MHz
1.5-V HSTL class I	250	MHz
1.5-V HSTL class II	225	MHz
1.8-V HSTL class I	250	MHz
1.8-V HSTL class II	225	MHz
3.3-V PCI	250	MHz
3.3-V PCI-X 1.0	225	MHz
Compact PCI	400	MHz
AGP 1×	400	MHz
AGP 2×	400	MHz
CTT	300	MHz
Differential HSTL	225	MHz
LVPECL (2)	717	MHz
PCML (2)	420	MHz

Table 4–48. HardCopy Stratix Maximum Output Clock Rate (Using I/O Pins) for PLL[1, 2, 3, 4] Pins (Part 2 of 2)

I/O Standard	Performance	Unit
LVDS (2)	717	MHz
HyperTransport technology (2)	420	MHz

Notes to Tables 4–47 through 4–48:

- (1) Differential SSTL-2 outputs are only available on column clock pins.
- (2) These parameters are only available on row I/O pins.
- (3) SSTL-2 in maximum drive strength condition.
- (4) SSTL-2 in minimum drive strength with $\leq 10\text{pF}$ output load condition.
- (5) SSTL-2 in minimum drive strength with $> 10\text{pF}$ output load condition.
- (6) Differential SSTL-2 outputs are only supported on column clock pins.

High-Speed I/O Specification

Table 4–49 provides high-speed timing specifications definitions.

Table 4–49. High-Speed Timing Specifications and Terminology

High-Speed Timing Specification	Terminology
t_C	High-speed receiver/transmitter input and output clock period.
f_{HCLK}	High-speed receiver/transmitter input and output clock frequency.
t_{RISE}	Low-to-high transmission time.
t_{FALL}	High-to-low transmission time.
Timing unit interval (TUI)	The timing budget allowed for skew, propagation delays, and data sampling window. $(TUI = 1/(\text{Receiver Input Clock Frequency} \times \text{Multiplication Factor}) = t_C/w)$.
f_{HSDR}	Maximum LVDS data transfer rate ($f_{HSDR} = 1/TUI$).
Channel-to-channel skew (TCCS)	The timing difference between the fastest and slowest output edges, including t_{CO} variation and clock skew. The clock is included in the TCCS measurement.
Sampling window (SW)	The period of time during which the data must be valid to be captured correctly. The setup and hold times determine the ideal strobe position within the sampling window. $SW = t_{SW}(\text{max}) - t_{SW}(\text{min})$.
Input jitter (peak-to-peak)	Peak-to-peak input jitter on high-speed PLLs.
Output jitter (peak-to-peak)	Peak-to-peak output jitter on high-speed PLLs.
t_{DUTY}	Duty cycle on high-speed transmitter output clock.
t_{LOCK}	Lock time for high-speed transmitter and receiver PLLs.

Table 4–50 shows the high-speed I/O timing for HardCopy Stratix devices.

Table 4–50. High-Speed I/O Specifications (Part 1 of 2) Notes (1), (2)

Symbol	Conditions	Performance			Unit
		Min	Typ	Max	
f_{HSCLK} (Clock frequency) (LVDS, LVPECL, HyperTransport technology) $f_{\text{HSCLK}} = f_{\text{HSDR}} / W$	$W = 4$ to 30 (Serdes used)	10		210	MHz
	$W = 2$ (Serdes bypass)	50		231	MHz
	$W = 2$ (Serdes used)	150		420	MHz
	$W = 1$ (Serdes bypass)	100		462	MHz
	$W = 1$ (Serdes used)	300		717	MHz
f_{HSDR} Device operation (LVDS, LVPECL, HyperTransport technology)	$J = 10$	300		840	Mbps
	$J = 8$	300		840	Mbps
	$J = 7$	300		840	Mbps
	$J = 4$	300		840	Mbps
	$J = 2$	100		462	Mbps
	$J = 1$ (LVDS and LVPECL only)	100		462	Mbps
f_{HSCLK} (Clock frequency) (PCML) $f_{\text{HSCLK}} = f_{\text{HSDR}} / W$	$W = 4$ to 30 (Serdes used)	10		100	MHz
	$W = 2$ (Serdes bypass)	50		200	MHz
	$W = 2$ (Serdes used)	150		200	MHz
	$W = 1$ (Serdes bypass)	100		250	MHz
	$W = 1$ (Serdes used)	300		400	MHz
f_{HSDR} Device operation (PCML)	$J = 10$	300		400	Mbps
	$J = 8$	300		400	Mbps
	$J = 7$	300		400	Mbps
	$J = 4$	300		400	Mbps
	$J = 2$	100		400	Mbps
	$J = 1$	100		250	Mbps
TCCS	All			200	ps
SW	PCML ($J = 4, 7, 8, 10$)	750			ps
	PCML ($J = 2$)	900			ps
	PCML ($J = 1$)	1,500			ps
	LVDS and LVPECL ($J = 1$)	500			ps
	LVDS, LVPECL, HyperTransport technology ($J = 2$ through 10)	440			ps

Table 4–50. High-Speed I/O Specifications (Part 2 of 2) Notes (1), (2)

Symbol	Conditions	Performance			Unit
		Min	Typ	Max	
Input jitter tolerance (peak-to-peak)	All			250	ps
Output jitter (peak-to-peak)	All			160	ps
Output t_{RISE}	LVDS	80	110	120	ps
	HyperTransport technology	110	170	200	ps
	LVPECL	90	130	150	ps
	PCML	80	110	135	ps
Output t_{FALL}	LVDS	80	110	120	ps
	HyperTransport technology	110	170	200	ps
	LVPECL	90	130	160	ps
	PCML	105	140	175	ps
t_{DUTY}	LVDS ($J = 2$ through 10)	47.5	50	52.5	%
	LVDS ($J = 1$) and LVPECL, PCML, HyperTransport technology	45	50	55	%
t_{LOCK}	All			100	μ s

Notes to Table 4–50:

- (1) When $J = 4, 7, 8$, and 10, the SERDES block is used.
- (2) When $J = 2$ or $J = 1$, the SERDES is bypassed.

PLL Specifications

Table 4–51 describes the HardCopy Stratix device enhanced PLL specifications.

Table 4–51. Enhanced PLL Specifications (Part 1 of 3)

Symbol	Parameter	Min	Typ	Max	Unit
f_{IN}	Input clock frequency	3 (1)		684	MHz
f_{INDUTY}	Input clock duty cycle	40		60	%
$f_{EINDUTY}$	External feedback clock input duty cycle	40		60	%
$t_{INJITTER}$	Input clock period jitter			± 200 (2)	ps
$t_{EINJITTER}$	External feedback clock period jitter			± 200 (2)	ps
t_{FCOMP}	External feedback clock compensation time (3)			6	ns

Table 4–51. Enhanced PLL Specifications (Part 2 of 3)

Symbol	Parameter	Min	Typ	Max	Unit
f_{OUT}	Output frequency for internal global or regional clock	0.3		500	MHz
f_{OUT_EXT}	Output frequency for external clock (2)	0.3		526	MHz
$t_{OUTDUTY}$	Duty cycle for external clock output (when set to 50%)	45		55	%
t_{JITTER}	Period jitter for external clock output (5)			± 100 ps for >200 MHz $outclk$ ± 20 mUI for <200 MHz $outclk$	ps or mUI
$t_{CONFIG5,6}$	Time required to reconfigure the scan chains for PLLs 5 and 6			$289/f_{SCANCLK}$	
$t_{CONFIG11,12}$	Time required to reconfigure the scan chains for PLLs 11 and 12			$193/f_{SCANCLK}$	
$t_{SCANCLK}$	scanclk frequency (4)			22	MHz
t_{DLOCK}	Time required to lock dynamically (after switchover or reconfiguring any non-post-scale counters/delays) (6)	(8)		100	μ s
t_{LOCK}	Time required to lock from end of device configuration	10		400	μ s
f_{VCO}	PLL internal VCO operating range	300		800 (7)	MHz
t_{LSKEW}	Clock skew between two external clock outputs driven by the same counter		± 50		ps
t_{SKEW}	Clock skew between two external clock outputs driven by the different counters with the same settings		± 75		ps
f_{SS}	Spread spectrum modulation frequency	30		150	kHz
% spread	Percentage spread for spread spectrum frequency (9)	0.4	0.5	0.6	%

Table 4–51. Enhanced PLL Specifications (Part 3 of 3)

Symbol	Parameter	Min	Typ	Max	Unit
t_{ARESET}	Minimum pulse width on ARESET signal	10 (11)			ns
		500 (12)			ns

Notes to Table 4–51:

- (1) The minimum input clock frequency to the PFD (f_{IN}/N) must be at least 3 MHz for HardCopy Stratix device enhanced PLLs.
- (2) Refer to “Maximum Input and Output Clock Rates”.
- (3) t_{FCOMP} can also equal 50% of the input clock period multiplied by the pre-scale divider n (whichever is less).
- (4) This parameter is timing analyzed by the Quartus II software because the `scanc1k` and `scandata` ports can be driven by the logic array.
- (5) Actual jitter performance may vary based on the system configuration.
- (6) Total required time to reconfigure and lock is equal to $t_{\text{DLOCK}} + t_{\text{CONFIG}}$. If only post-scale counters and delays are changed, then t_{DLOCK} is equal to 0.
- (7) The VCO range is limited to 500 to 800 MHz when the spread spectrum feature is selected.
- (8) Lock time is a function of PLL configuration and may be significantly faster depending on bandwidth settings or feedback counter change increment.
- (9) Exact, user-controllable value depends on the PLL settings.
- (10) The LOCK circuit on HardCopy Stratix PLLs does not work for industrial devices below -20°C unless the PFD frequency > 200 MHz. Refer to the *Stratix FPGA Errata Sheet* for more information on the PLL.
- (11) Applicable when the PLL input clock has been running continuously for at least 10 μs .
- (12) Applicable when the PLL input clock has stopped toggling or has been running continuously for less than 10 μs .

Table 4–52 describes the HardCopy Stratix device fast PLL specifications.

Table 4–52. Fast PLL Specifications

Symbol	Parameter	Min	Max	Unit
f_{IN}	CLKIN frequency (for $m = 1$) (1), (2)	300	717	MHz
	CLKIN frequency (for $m = 2$ to 19)	300/ m	1,000/ m	MHz
	CLKIN frequency (for $m = 20$ to 32)	10	1,000/ m	MHz
f_{OUT}	Output frequency for internal global or regional clock (3)	9.4	420	MHz
f_{OUT_EXT}	Output frequency for external clock (2)	9.375	717	MHz
f_{VCO}	VCO operating frequency	300	1,000	MHz
t_{INDUTY}	CLKIN duty cycle	40	60	%
$t_{INJITTER}$	Period jitter for CLKIN pin		±200	ps
t_{DUTY}	Duty cycle for DFFIO 1 × CLKOUT pin (4)	45	55	%
t_{JITTER}	Period jitter for DFFIO clock out (4)		±80	ps
	Period jitter for internal global or regional clock		±100 ps for >200-MHz outclk ±20 mUI for <200-MHz outclk	ps or mUI
t_{LOCK}	Time required for PLL to acquire lock	10	100	μs
m	Multiplication factors for m counter (4)	1	32	Integer
l_0, l_1, g_0	Multiplication factors for l_0, l_1 , and g_0 counter (5), (6)	1	32	Integer
t_{ARESET}	Minimum pulse width on areset signal	10		ns

Notes to Table 4–52:

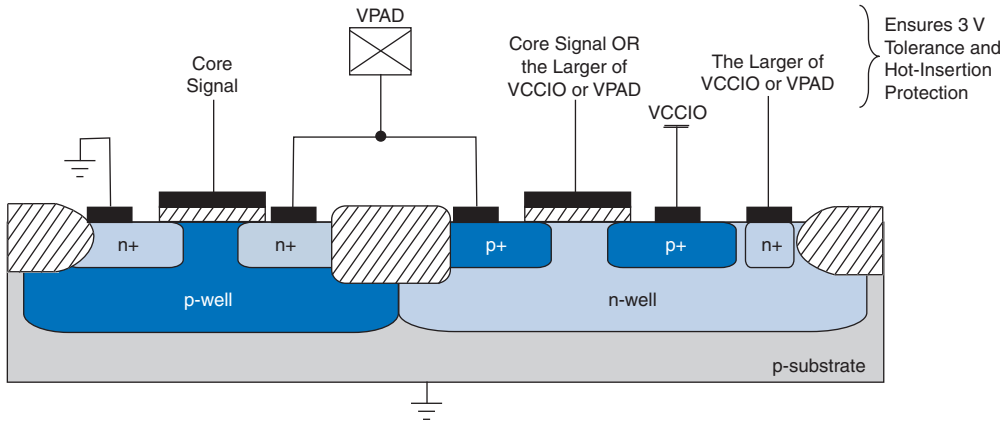
- (1) Refer to “Maximum Input and Output Clock Rates” on page 4–23 for more information.
- (2) PLLs 7, 8, 9, and 10 in the HC1S80 device support up to 717-MHz input and output.
- (3) When using the SERDES, high-speed differential I/O mode supports a maximum output frequency of 210 MHz to the global or regional clocks (for example, the maximum data rate 840 Mbps divided by the smallest SERDES J factor of 4).
- (4) This parameter is for high-speed differential I/O mode only.
- (5) These counters have a maximum of 32 if programmed for 50/50 duty cycle. Otherwise, they have a maximum of 16.
- (6) High-speed differential I/O mode supports $W = 1$ to 16 and $J = 4, 7, 8$, or 10.

Electrostatic Discharge

Electrostatic discharge (ESD) protection is a design practice that is integrated in Altera FPGAs and Structured ASIC devices. HardCopy Stratix devices are no exception, and they are designed with ESD protection on all I/O and power pins.

Figure 4–2 shows a transistor level cross section of the HardCopy Stratix CMOS I/O buffer structure which will be used to explain ESD protection.

Figure 4–2. Transistor-Level Cross Section of the HardCopy Stratix Device I/O Buffers



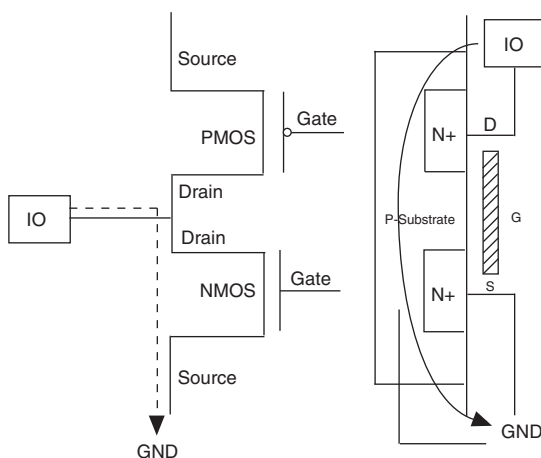
The CMOS output drivers in the I/O pins intrinsically provide electrostatic discharge protection. There are two cases to consider for ESD voltage strikes: positive voltage zap and negative voltage zap.

Positive Voltage Zap

A positive ESD voltage zap occurs when a positive voltage is present on an I/O pin due to an ESD charge event. This can cause the N+ (Drain)/P-Substrate) junction of the N-channel drain to break down and the N+ (Drain)/P-Substrate/N+ (Source) intrinsic bipolar transistor turns ON to discharge ESD current from I/O pin to GND.

The dashed line (Figure 4-3) shows the ESD current discharge path during a positive voltage zap.

Figure 4-3. ESD Protection During Positive Voltage Zap

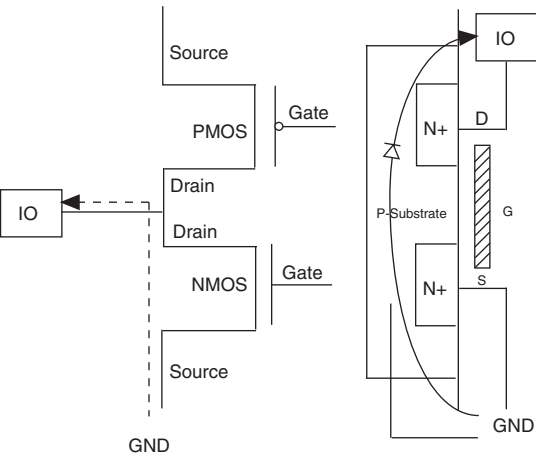


Negative Voltage Zap

When the I/O pin receives a negative ESD zap at the pin that is less than -0.7 V (0.7 V is the voltage drop across a diode), the intrinsic PSubstrate/N+ drain diode is forward biased. Hence, the discharge ESD current path is from GND to the I/O pin, as shown in Figure 4-4.

The dashed line (Figure 4-4) shows the ESD current discharge path during a negative voltage zap.

Figure 4-4. ESD Protection During Negative Voltage Zap



- Details of ESD protection are also outlined in the *Hot-Socketing and Power-Sequencing Feature and Testing for Altera Devices* white paper located on the Altera website at www.altera.com.
- For information on ESD results of Altera products, see the Reliability Report on the Altera website at www.altera.com.

Document
Revision History

Table 4-53 shows the revision history for this chapter.

Table 4-53. Document Revision History (Part 1 of 2)		
Date and Document Version	Changes Made	Summary of Changes
September 2008 v3.4	Updated the revision history.	—
June 2007 v3.3	Updated R _{CONF} section of Table 4-3. Added the “Electrostatic Discharge” section.	—

Table 4–53. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
December 2006 v3.2	Updated chapter number and metadata.	—
March 2006	Formerly chapter 8; no content change.	—
October 2005 v3.1	• Minor edits • Graphic updates	
May 2005 v3.0	• Updated SSTL-2 and SSTL-3 specifications in Tables 8–19 through 8–22 • Updated CTT I/O specifications in Table 8–30 • Updated bus hold parameters in Table 8–31. • Added the External Timing Parameters, HardCopy Stratix External I/O Timing, and Maximum Input and Output Clock Rates sections • Added the High-Speed I/O Specification, and PLL Specifications sections	—
January 2005 v2.0	Removed recommended maximum rise and fall times (t_R and t_F) for input signals	—
June 2003 v1.0	Initial release of Chapter 8, Operating Conditions, in the <i>HardCopy Device Handbook</i>	

Introduction

Altera® HardCopy devices provide a comprehensive alternative to ASICs. HardCopy structured ASICs offer a complete solution from prototype to high-volume production, and maintain the powerful features and high-performance architecture of their equivalent FPGAs with the programmability removed. You can use the Quartus II design software to design HardCopy devices in a manner similar to the traditional ASIC design flow and you can prototype with Altera's high density Stratix, APEX 20KC, and APEX 20KE FPGAs before seamlessly migrating to the corresponding HardCopy device for high-volume production.

HardCopy structured ASICs provide the following key benefits:

- Improves performance, on the average, by 40% over the corresponding -6 speed grade FPGA device
- Lowers power consumption, on the average, by 40% over the corresponding FPGA
- Preserves the FPGA architecture and features and minimizes risk
- Guarantees first-silicon success through a proven, seamless migration process from the FPGA to the equivalent HardCopy device
- Offers a quick turnaround of the FPGA design to a structured ASIC device—samples are available in about eight weeks

Altera's Quartus II software has built-in support for HardCopy Stratix devices. The HardCopy design flow in Quartus II software offers the following advantages:

- Unified design flow from prototype to production
- Performance estimation of the HardCopy Stratix device allows you to design systems for maximum throughput
- Easy-to-use and inexpensive design tools from a single vendor
- An integrated design methodology that enables system-on-a-chip designs

This section discusses the following areas:

- How to design HardCopy Stratix and HardCopy APEX structured ASICs using the Quartus II software
- An explanation of what the `HARDCOPY_FPGA_PROTOTYPE` devices are and how to target designs to these devices
- Performance and power estimation of HardCopy Stratix devices
- How to generate the HardCopy design database for submitting HardCopy Stratix and HardCopy APEX designs to the HardCopy Design Center

Features

Beginning with version 4.2, the Quartus II software contains several powerful features that facilitate design of HardCopy Stratix and HardCopy APEX devices:

- **HARDCOPY_FPGA_PROTOTYPE Devices**
These are virtual Stratix FPGA devices with features identical to HardCopy Stratix devices. You must use these FPGA devices to prototype your designs and verify the functionality in silicon.
- **HardCopy Timing Optimization Wizard**
Using this feature, you can target your design to HardCopy Stratix devices, providing an estimate of the design's performance in a HardCopy Stratix device.
- **HardCopy Stratix Floorplans and Timing Models**
The Quartus II software supports post-migration HardCopy Stratix device floorplans and timing models and facilitates design optimization for design performance.
- **Placement Constraints**
Location and LogicLock constraints are supported at the HardCopy Stratix floorplan level to improve overall performance.
- **Improved Timing Estimation**
Beginning with version 4.2, the Quartus II software determines routing and associated buffer insertion for HardCopy Stratix designs, and provides the Timing Analyzer with more accurate information about the delays than was possible in previous versions of the Quartus II software. The Quartus II Archive File automatically receives buffer insertion information, which greatly enhances the timing closure process in the back-end migration of your HardCopy Stratix device.

■ Design Assistant

This feature checks your design for compliance with all HardCopy device design rules and establishes a seamless migration path in the quickest time.

■ HardCopy Files Wizard

This wizard allows you to deliver to Altera the design database and all the deliverables required for migration. This feature is used for HardCopy Stratix and HardCopy APEX devices.



The HardCopy Stratix and HardCopy APEX PowerPlay Early Power Estimator is available on the Altera website at www.altera.com.

HARDCOPY_FPGA_PROTOTYPE, HardCopy Stratix and Stratix Devices

You must use the HARDCOPY_FPGA_PROTOTYPE virtual devices available in the Quartus II software to target your designs to the actual resources and package options available in the equivalent post-migration HardCopy Stratix device. The programming file generated for the HARDCOPY_FPGA_PROTOTYPE can be used in the corresponding Stratix FPGA device.

The purpose of the HARDCOPY_FPGA_PROTOTYPE is to guarantee seamless migration to HardCopy by making sure that your design only uses resources in the FPGA that can be used in the HardCopy device after migration. You can use the equivalent Stratix FPGAs to verify the design's functionality in-system, then generate the design database necessary to migrate to a HardCopy device. This process ensures the seamless migration of the design from a prototyping device to a production device in high volume. It also minimizes risk, assures samples in about eight weeks, and guarantees first-silicon success.



HARDCOPY_FPGA_PROTOTYPE devices are only available for HardCopy Stratix devices and are not available for the HardCopy II or HardCopy APEX device families.

Table 5–1 compares HARDCOPY_FPGA_PROTOTYPE devices, Stratix devices, and HardCopy Stratix devices.

Table 5–1. Qualitative Comparison of HARDCOPY_FPGA_PROTOTYPE to Stratix and HardCopy Stratix Devices (Part 1 of 2)

Stratix Device	HARDCOPY_FPGA_PROTOTYPE Device	HardCopy Stratix Device
FPGA	Virtual FPGA	Structured ASIC
FPGA	Architecture identical to Stratix FPGA	Architecture identical to Stratix FPGA

Table 5–1. Qualitative Comparison of *HARDCOPY_FPGA_PROTOTYPE* to Stratix and HardCopy Stratix Devices (Part 2 of 2)

Stratix Device	<i>HARDCOPY_FPGA_PROTOTYPE</i> Device	HardCopy Stratix Device
FPGA	Resources identical to HardCopy Stratix device	M-RAM resources different than Stratix FPGA in some devices
Ordered through Altera part number	Cannot be ordered, use the Altera Stratix FPGA part number	Ordered by Altera part number

Table 5–2 lists the resources available in each of the HardCopy Stratix devices.

Table 5–2. HardCopy Stratix Device Physical Resources

Device	LEs	ASIC Equivalent Gates (K) ⁽¹⁾	M512 Blocks	M4K Blocks	M-RAM Blocks	DSP Blocks	PLLs	Maximum User I/O Pins
HC1S25F672	25,660	250	224	138	2	10	6	473
HC1S30F780	32,470	325	295	171	2 ⁽²⁾	12	6	597
HC1S40F780	41,250	410	384	183	2 ⁽²⁾	14	6	615
HC1S60F1020	57,120	570	574	292	6	18	12	773
HC1S80F1020	79,040	800	767	364	6 ⁽²⁾	22	12	773

Notes to Table 5–2:

- (1) Combinational and registered logic do not include digital signal processing (DSP) blocks, on-chip RAM, or phase-locked loops (PLLs).
- (2) The M-RAM resources for these HardCopy devices differ from the corresponding Stratix FPGA.

For a given device, the number of available M-RAM blocks in HardCopy Stratix devices is identical with the corresponding *HARDCOPY_FPGA_PROTOTYPE* devices, but may be different from the corresponding Stratix devices. Maintaining the identical resources between *HARDCOPY_FPGA_PROTOTYPE* and HardCopy Stratix devices facilitates seamless migration from the FPGA to the structured ASIC device.



For more information about HardCopy Stratix devices, refer to the *HardCopy Stratix Device Family Data Sheet* section in volume 1 of the *HardCopy Series Handbook*.

The three devices, Stratix FPGA, *HARDCOPY_FPGA_PROTOTYPE*, and HardCopy device, are distinct devices in the Quartus II software. The *HARDCOPY_FPGA_PROTOTYPE* programming files are used in the

Stratix FPGA for your design. The three devices are tied together with the same netlist, thus a single SRAM Object File (.sof) can be used to achieve the various goals at each stage. The same SRAM Object File is generated in the HARDCOPY_FPGA_PROTOTYPE design, and is used to program the Stratix FPGA device, the same way that it is used to generate the HardCopy Stratix device, guaranteeing a seamless migration.



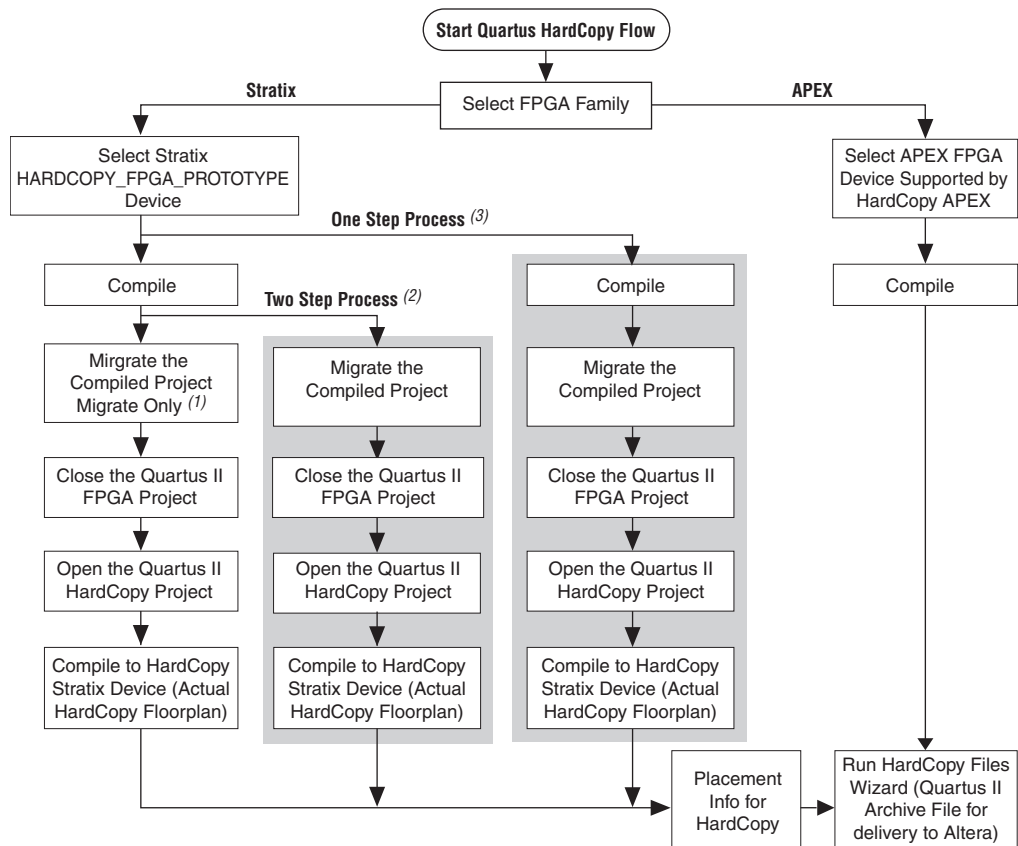
For more information about the SRAM Object File and programming Stratix FPGA devices, refer to the *Programming and Configuration* chapter of the *Introduction to Quartus II Manual*.

HardCopy Design Flow

Figure 5–1 shows a HardCopy design flow diagram. The design steps are explained in detail in the following sections of this chapter. The HardCopy Stratix design flow utilizes the HardCopy Timing Optimization Wizard to automate the migration process into a one-step process. The remainder of this section explains the tasks performed by this automated process.



For a detailed description of the HardCopy Timing Optimization Wizard and HardCopy Files Wizard, refer to “HardCopy Timing Optimization Wizard Summary” and “Generating the HardCopy Design Database”.

Figure 5–1. HardCopy Stratix and HardCopy APEX Design Flow Diagram**Notes to Figure 5–1:**

- (1) Migrate Only Process: The displayed flow is completed manually.
- (2) Two Step Process: Migration and Compilation are done automatically (shaded area).
- (3) One Step Process: Full HardCopy Compilation. The entire process is completed automatically (shaded area).

The Design Flow Steps of the One Step Process

The following sections describe each step of the full HardCopy compilation (the One Step Process), [Figure 5–1](#).

Compile the Design for an FPGA

This step compiles the design for a HARDCOPY_FPGA_PROTOTYPE device and gives you the resource utilization and performance of the FPGA.

Migrate the Compiled Project

This step generates the Quartus II Project File (.qpf) and the other files required for HardCopy implementation. The Quartus II software also assigns the appropriate HardCopy Stratix device for the design migration.

Close the Quartus FPGA Project

Because you must compile the project for a HardCopy Stratix device, you must close the existing project which you have targeted your design to a HARDCOPY_FPGA_PROTOTYPE device.

Open the Quartus HardCopy Project

Open the Quartus II project that you created in the “[Migrate the Compiled Project](#)” step. The selected device is one of the devices from the HardCopy Stratix family that was assigned during that step.

Compile for HardCopy Stratix Device

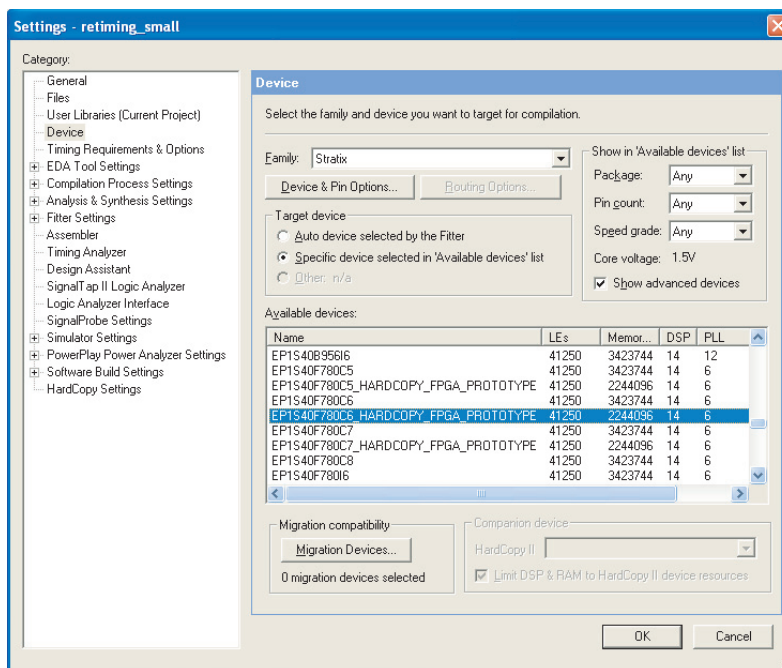
Compile the design for a HardCopy Stratix device. After successful compilation, the Timing Analysis section of the compilation report shows the performance of the design implemented in the HardCopy device.

How to Design HardCopy Stratix Devices

This section describes the process for designing for a HardCopy Stratix device using the HARDCOPY_FPGA_PROTOTYPE as your initial selected device. In order to use the HardCopy Timing Optimization Wizard, you must first design with the HARDCOPY_FPGA_PROTOTYPE in order for the design to migrate to a HardCopy Stratix device.

To target a design to a HardCopy Stratix device in the Quartus II software, follow these steps:

1. If you have not yet done so, create a new project or open an existing project.
2. On the Assignments menu, click **Settings**. In the **Category** list, select **Device**.
3. On the **Device** page, in the **Family** list, select **Stratix**. Select the desired HARDCOPY_FPGA_PROTOTYPE device in the **Available Devices** list ([Figure 5–2](#)).

Figure 5–2. Selecting a **HARDCOPY_FPGA_PROTOTYPE** Device

By choosing the **HARDCOPY_FPGA_PROTOTYPE** device, all the design information, available resources, package option, and pin assignments are constrained to guarantee a seamless migration of your project to the HardCopy Stratix device. The netlist resulting from the **HARDCOPY_FPGA_PROTOTYPE** device compilation contains information about the electrical connectivity, resources used, I/O placements, and the unused resources in the FPGA device.

4. On the Assignments menu, click **Settings**. In the **Category** list, select **HardCopy Settings** and specify the input transition timing to be modeled for both clock and data input pins. These transition times are used in static timing analysis during back-end timing closure of the HardCopy device.
5. Add constraints to your **HARDCOPY_FPGA_PROTOTYPE** device, and on the Processing menu, click **Start Compilation** to compile the design.

HardCopy Timing Optimization Wizard

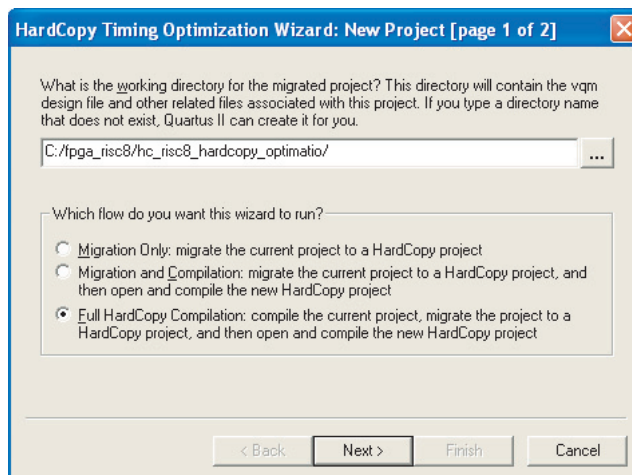
After you have successfully compiled your design in the `HARDCOPY_FPGA_PROTOTYPE`, you must migrate the design to the HardCopy Stratix device to get a performance estimation of the HardCopy Stratix device. This migration is required before submitting the design to Altera for the HardCopy Stratix device implementation. To perform the required migration, on the Project menu, point to HardCopy Utilities and click **HardCopy Timing Optimization Wizard**.

At this point, you are presented with the following three choices to target the designs to HardCopy Stratix devices ([Figure 5-3](#)).

- **Migration Only:** You can select this option after compiling the `HARDCOPY_FPGA_PROTOTYPE` project to migrate the project to a HardCopy Stratix project.

You can now perform the following tasks manually to target the design to a HardCopy Stratix device. Refer to [“Performance Estimation” on page 5-12](#) for additional information about how to perform these tasks.

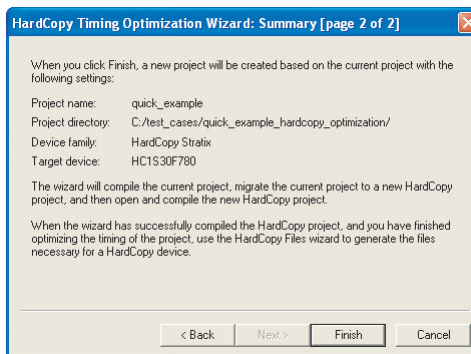
- Close the existing project
 - Open the migrated HardCopy Stratix project
 - Compile the HardCopy Stratix project for a HardCopy Stratix device
- **Migration and Compilation:** You can select this option after compiling the project. This option results in the following actions:
 - Migrating the project to a HardCopy Stratix project
 - Opening the migrated HardCopy Stratix project and compiling the project for a HardCopy Stratix device
- **Full HardCopy Compilation:** Selecting this option results in the following actions:
 - Compiling the existing `HARDCOPY_FPGA_PROTOTYPE` project
 - Migrating the project to a HardCopy Stratix project
 - Opening the migrated HardCopy Stratix project and compiling it for a HardCopy Stratix device

Figure 5–3. HardCopy Timing Optimization Wizard Options

The main benefit of the HardCopy Timing Wizard's three options is flexibility of the conversion process automation. The first time you migrate your `HARDCOPY_FPGA_PROTOTYPE` project to a HardCopy Stratix device, you may want to use Migration Only, and then work on the HardCopy Stratix project in the Quartus II software. As your prototype FPGA project and HardCopy Stratix project constraints stabilize and you have fewer changes, the Full HardCopy Compilation is ideal for one-click compiling of your `HARDCOPY_FPGA_PROTOTYPE` and HardCopy Stratix projects.

After selecting the wizard you want to run, the “HardCopy Timing Optimization Wizard: Summary” page shows you details about the settings you made in the Wizard, as shown in (Figure 5–4).

Figure 5–4. HardCopy Timing Optimization Wizard Summary Page



When either of the second two options in Figure 5–4 are selected (**Migration and Compilation** or **Full HardCopy Compilation**), designs are targeted to HardCopy Stratix devices and optimized using the HardCopy Stratix placement and timing analysis to estimate performance. For details on the performance optimization and estimation steps, refer to “**Performance Estimation**” on page 5–12. If the performance requirement is not met, you can modify your RTL source, optimize the FPGA design, and estimate timing until you reach timing closure.

Tcl Support for HardCopy Migration

To complement the GUI features for HardCopy migration, the Quartus II software provides the following command-line executables (which provide the tool command language (Tcl) shell to run the `--flow` Tcl command) to migrate the `HARDCOPY_FPGA_PROTOTYPE` project to HardCopy Stratix devices:

- `quartus_sh --flow migrate_to_hardcopy <project_name> [-c <revision>]` ↵

This command migrates the project compiled for the `HARDCOPY_FPGA_PROTOTYPE` device to a HardCopy Stratix device.

```
■ quartus_sh --flow hardcopy_full_compile <project_name>  
  [-c <revision>] ←
```

This command performs the following tasks:

- Compiles the existing project for a `HARDCOPY_FPGA_PROTOTYPE` device.
- Migrates the project to a HardCopy Stratix project.
- Opens the migrated HardCopy Stratix project and compiles it for a HardCopy Stratix device.

Design Optimization and Performance Estimation

The HardCopy Timing Optimization Wizard creates the HardCopy Stratix project in the Quartus II software, where you can perform design optimization and performance estimation of your HardCopy Stratix device.

Design Optimization

Beginning with version 4.2, the Quartus II software supports HardCopy Stratix design optimization by providing floorplans for placement optimization and HardCopy Stratix timing models. These features allow you to refine placement of logic array blocks (LAB) and optimize the HardCopy design further than the FPGA performance. Customized routing and buffer insertion done in the Quartus II software are then used to estimate the design's performance in the migrated device. The HardCopy device floorplan, routing, and timing estimates in the Quartus II software reflect the actual placement of the design in the HardCopy Stratix device, and can be used to see the available resources, and the location of the resources in the actual device.

Performance Estimation

Figure 5–5 illustrates the design flow for estimating performance and optimizing your design. You can target your designs to `HARDCOPY_FPGA_PROTOTYPE` devices, migrate the design to the HardCopy Stratix device, and get placement optimization and timing estimation of your HardCopy Stratix device.

In the event that the required performance is not met, you can:

- Work to improve LAB placement in the HardCopy Stratix project.

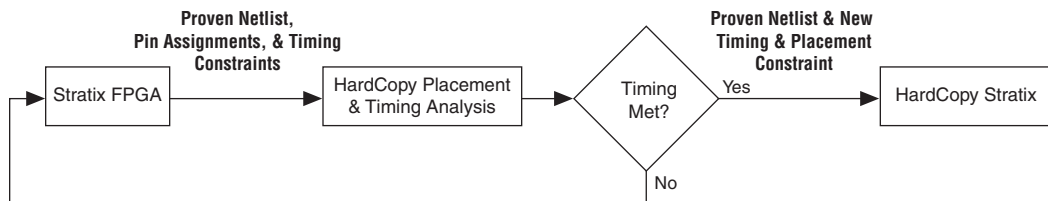
or

- Go back to the `HARDCOPY_FPGA_PROTOTYPE` project and optimize that design, modify your RTL source code, repeat the migration to the HardCopy Stratix device, and perform the optimization and timing estimation steps.



On average, HardCopy Stratix devices are 40% faster than the equivalent -6 speed grade Stratix FPGA device. These performance numbers are highly design dependent, and you must obtain final performance numbers from Altera.

Figure 5–5. Obtaining a HardCopy Performance Estimation



To perform Timing Analysis for a HardCopy Stratix device, follow these steps:

1. Open an existing project compiled for a `HARDCOPY_FPGA_PROTOTYPE` device.
2. On the Project menu, point to HardCopy Utilities and click **HardCopy Timing Optimization Wizard**.
3. Select a destination directory for the migrated project and complete the HardCopy Timing Optimization Wizard process.

On completion of the HardCopy Timing Optimization Wizard, the destination directory created contains the Quartus II project file, and all files required for HardCopy Stratix implementation. At this stage, the design is copied from the `HARDCOPY_FPGA_PROTOTYPE` project directory to a new directory to perform the timing analysis. This two-project directory structure enables you to move back and forth between the `HARDCOPY_FPGA_PROTOTYPE` design database and the HardCopy Stratix design database. The Quartus II software creates the `<project name>_hardcopy_optimization` directory.

You do not have to select the HardCopy Stratix device while performing performance estimation. When you run the HardCopy Timing Optimization Wizard, the Quartus II software selects the

HardCopy Stratix device corresponding to the specified `HARDCOPY_FPGA_PROTOTYPE` FPGA. Thus, the information necessary for the HardCopy Stratix device is available from the earlier `HARDCOPY_FPGA_PROTOTYPE` device selection.

All constraints related to the design are also transferred to the new project directory. You can modify these constraints, if necessary, in your optimized design environment to achieve the necessary timing closure. However, if the design is optimized at the `HARDCOPY_FPGA_PROTOTYPE` device level by modifying the RTL code or the device constraints, you must migrate the project with the HardCopy Timing Optimization Wizard.



If an existing project directory is selected when the HardCopy Timing Optimization Wizard is run, the existing information is overwritten with the new compile results.

The project directory is the directory that you chose for the migrated project. A snapshot of the files inside the `<project name>_hardcopy_optimization` directory is shown in Table 5–3.

Table 5–3. Directory Structure Generated by the HardCopy Timing Optimization Wizard

```

<project name>_hardcopy_optimization\
  <project name>.qsf
  <project name>.qpf
  <project name>.sof
  <project name>.macr
  <project name>.gclk
  db\
  hardcopy_fpga_prototype\
    fpga_<project name>_violations.datasheet
    fpga_<project name>_target.datasheet
    fpga_<project name>_rba_pt_hcpy_v.tcl
    fpga_<project name>_pt_hcpy_v.tcl
    fpga_<project name>_hcpy_v.sdo
    fpga_<project name>_hcpy.vo
    fpga_<project name>_cpld.datasheet
    fpga_<project name>_cksum.datasheet
    fpga_<project name>.tan.rpt
    fpga_<project name>.map.rpt
    fpga_<project name>.map.atm
    fpga_<project name>.fit.rpt
    fpga_<project name>.db_info
    fpga_<project name>.cmp.xml
    fpga_<project name>.cmp.rcf
    fpga_<project name>.cmp.atm
    fpga_<project name>.asm.rpt
    fpga_<project name>.qarlog
    fpga_<project name>.qar
    fpga_<project name>.qsf
    fpga_<project name>.pin
    fpga_<project name>.qpf
  db_export\
    <project name>.map.atm
    <project name>.map.hdbx
    <project name>.db_info

```

4. Open the migrated Quartus II project created in Step 3.
5. Perform a full compilation.

After successful compilation, the Timing Analysis section of the Compilation Report shows the performance of the design.



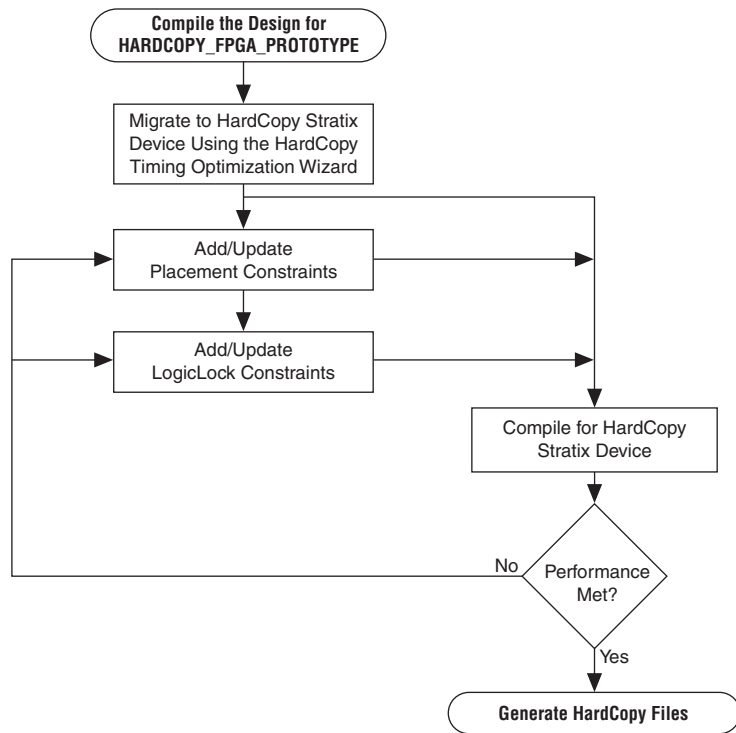
Performance estimation is not supported for HardCopy APEX devices in the Quartus II software. Your design can be optimized by modifying the RTL code or the FPGA design and the constraints. You should contact Altera to discuss any desired performance improvements with HardCopy APEX devices.

Buffer Insertion

Beginning with version 4.2, the Quartus II software provides improved HardCopy Stratix device timing closure and estimation, to more accurately reflect the results expected after back-end migration. The Quartus II software performs the necessary buffer insertion in your HardCopy Stratix device during the Fitter process, and stores the location of these buffers and necessary routing information in the Quartus II Archive File. This buffer insertion improves the estimation of the Quartus II Timing Analyzer for the HardCopy Stratix device.

Placement Constraints

Beginning with version 4.2, the Quartus II software supports placement constraints and LogicLock regions for HardCopy Stratix devices. [Figure 5–6](#) shows an iterative process to modify the placement constraints until the best placement for the HardCopy Stratix device is achieved.

Figure 5–6. Placement Constraints Flow for HardCopy Stratix Devices

Location Constraints

This section provides information about HardCopy Stratix logic location constraints.

LAB Assignments

Logic placement in HardCopy Stratix is limited to LAB placement and optimization of the interconnecting signals between them. In a Stratix FPGA, individual logic elements (LE) are placed by the Quartus II Fitter into LABs. The HardCopy Stratix migration process requires that LAB contents cannot change after the Timing Optimization Wizard task is done. Therefore, you can only make LAB-level placement optimization and location assignments after migrating the `HARDCOPY_FPGA_PROTOTYPE` project to the HardCopy Stratix device.

The Quartus II software supports these LAB location constraints for HardCopy Stratix devices. The entire contents of a LAB is moved to an empty LAB when using LAB location assignments. If you want to move the logic contents of LAB A to LAB B, the entire contents of LAB A are moved to an empty LAB B. For example, the logic contents of LAB_X33_Y65 can be moved to an empty LAB at LAB_X43_Y56 but individual logic cell LC_X33_Y65_N1 can not be moved by itself in the HardCopy Stratix Timing Closure Floorplan.

LogicLock Assignments

The LogicLock feature of the Quartus II software provides a block-based design approach. Using this technique you can partition your design and create each block of logic independently, optimize placement and area, and integrate all blocks into the top level design.



To learn more about this methodology, refer to the *Quartus II Analyzing and Optimizing Design Floorplan* chapter in volume 2 of the *Quartus II Handbook*.

LogicLock constraints are supported when you migrate the project from a HARDCOPY_FPGA_PROTOTYPE project to a HardCopy Stratix project. If the LogicLock region was specified as “Size=Fixed” and “Location=Locked” in the HARDCOPY_FPGA_PROTOTYPE project, it is converted to have “Size=Auto” and “Location=Floating” as shown in the following LogicLock examples. This modification is necessary because the floorplan of a HardCopy Stratix device is different from that of the Stratix device, and the assigned coordinates in the HARDCOPY_FPGA_PROTOTYPE do not match the HardCopy Stratix floorplan. If this modification did not occur, LogicLock assignments would lead to incorrect placement in the Quartus II Fitter. Making the regions auto-size and floating, maintains your LogicLock assignments, allowing you to easily adjust the LogicLock regions as required and lock their locations again after HardCopy Stratix placement.

The following are two examples of LogicLock assignments.

LogicLock Region Definition in the HARDCOPY_FPGA_PROTOTYPE Quartus II Settings File

```
set_global_assignment -name LL_HEIGHT 15 -entity risc8 -section_id test
set_global_assignment -name LL_WIDTH 15 -entity risc8 -section_id test
set_global_assignment -name LL_STATE LOCKED -entity risc8 -section_id test
set_global_assignment -name LL_AUTO_SIZE OFF -entity risc8 -section_id test
```

LogicLock Region Definition in the Migrated HardCopy Stratix Quartus II Settings File

```
set_global_assignment -name LL_HEIGHT 15 -entity risc8 -section_id test
set_global_assignment -name LL_WIDTH 15 -entity risc8 -section_id test
set_global_assignment -name LL_STATE FLOATING -entity risc8 -section_id test
set_global_assignment -name LL_AUTO_SIZE ON -entity risc8 -section_id test
```

Checking Designs for HardCopy Design Guidelines

When you develop a design with HardCopy migration in mind, you must follow Altera-recommended design practices that ensure a straightforward migration process or the design will not be able to be implemented in a HardCopy device. Prior to starting migration of the design to a HardCopy device, you must review the design and identify and address all the design issues. Any design issues that have not been addressed can jeopardize silicon success.

Altera Recommended HDL Coding Guidelines

Designing for Altera PLD, FPGA, and HardCopy structured ASIC devices requires certain specific design guidelines and hardware description language (HDL) coding style recommendations be followed.



For more information about design recommendations and HDL coding styles, refer to the *Design Guidelines* section in volume 1 of the *Quartus II Handbook*.

Design Assistant

The Quartus II software includes the Design Assistant feature to check your design against the HardCopy design guidelines. Some of the design rule checks performed by the Design Assistant include the following rules:

- Design should not contain combinational loops
- Design should not contain delay chains
- Design should not contain latches

To use the Design Assistant, you must run Analysis and Synthesis on the design in the Quartus II software. Altera recommends that you run the Design Assistant to check for compliance with the HardCopy design guidelines early in the design process and after every compilation.

Design Assistant Settings

You must select the design rules in the **Design Assistant** page prior to running the design. On the Assignments menu, click **Settings**. In the **Settings** dialog box, in the Category list, select **Design Assistant** and turn on **Run Design Assistant during compilation**. Altera recommends enabling this feature to run the Design Assistant automatically during compilation of your design.

Running Design Assistant

To run Design Assistant independently of other Quartus II features, on the Processing menu, point to Start and click **Start Design Assistant**.

The Design Assistant automatically runs in the background of the Quartus II software when the HardCopy Timing Optimization Wizard is launched, and does not display the Design Assistant results immediately to the display. The design is checked before the Quartus II software migrates the design and creates a new project directory for performing timing analysis.

Also, the Design Assistant runs automatically whenever you generate the HardCopy design database with the HardCopy Files Wizard. The Design Assistant report generated is used by the Altera HardCopy Design Center to review your design.

Reports and Summary

The results of running the Design Assistant on your design are available in the Design Assistant Results section of the Compilation Report. The Design Assistant also generates the summary report in the *<project name>\hardcopy* subdirectory of the project directory. This report file is titled *<project name>_violations.datasheet*. Reports include the settings, run summary, results summary, and details of the results and messages. The Design Assistant report indicates the rule name, severity of the violation, and the circuit path where any violation occurred.



To learn about the design rules and standard design practices to comply with HardCopy design rules, refer to the Quartus II Help and the *HardCopy Series Design Guidelines* chapter in volume 1 of the *HardCopy Series Handbook*.

Generating the HardCopy Design Database

You can use the HardCopy Files Wizard to generate the complete set of deliverables required for migrating the design to a HardCopy device in a single click. The HardCopy Files Wizard asks questions related to the design and archives your design, settings, results, and database files for delivery to Altera. Your responses to the design details are stored in `<project name>_hardcopy_optimization\<project name>.hps.txt`.

You can generate the archive of the HardCopy design database only after compiling the design to a HardCopy Stratix device. The Quartus II Archive File is generated at the same directory level as the targeted project, either before or after optimization.



The Design Assistant automatically runs when the HardCopy Files Wizard is started.

Figure 5–4 shows the archive directory structure and files collected by the HardCopy Files Wizard.

Table 5–4. HardCopy Stratix Design Files Collected by the HardCopy Files Wizard

```

<project name>_hardcopy_optimization\
  <project name>.flow.rpt
  <project name>.qpf
  <project name>.asm.rpt
  <project name>.blf
  <project name>.fit.rpt
  <project name>.gclk
  <project name>.hps.txt
  <project name>.macr
  <project name>.pin
  <project name>.qsf
  <project name>.sof
  <project name>.tan.rpt

  hardcopy\
    <project name>.apc
    <project name>_cksum.datasheet
    <project name>_cpld.datasheet
    <project name>_hcpy.vo
    <project name>_hcpy_v.sdo
    <project name>_pt_hcpy_v.tcl
    <project name>_rba_pt_hcpy_v.tcl
    <project name>_target.datasheet
    <project name>_violations.datasheet

  hardcopy_fpga_prototype\
    fpga_<project name>.asm.rpt
    fpga_<project name>.cmp.rcf
    fpga_<project name>.cmp.xml
    fpga_<project name>.db_info
    fpga_<project name>.fit.rpt
    fpga_<project name>.map.atm
    fpga_<project name>.map.rpt
    fpga_<project name>.pin
    fpga_<project name>.qsf
    fpga_<project name>.tan.rpt
    fpga_<project name>_cksum.datasheet
    fpga_<project name>_cpld.datasheet
    fpga_<project name>_hcpy.vo
    fpga_<project name>_hcpy_v.sdo
    fpga_<project name>_pt_hcpy_v.tcl
    fpga_<project name>_rba_pt_hcpy_v.tcl
    fpga_<project name>_target.datasheet
    fpga_<project name>_violations.datasheet

  db_export\
    <project name>.db_info
    <project name>.map.atm
    <project name>.map.hdbx

```

After creating the migration database with the HardCopy Timing Optimization Wizard, you must compile the design before generating the project archive. You will receive an error if you create the archive before compiling the design.

Static Timing Analysis

In addition to performing timing analysis, the Quartus II software also provides all of the requisite netlists and Tcl scripts to perform static timing analysis (STA) using the Synopsys STA tool, PrimeTime. The following files, necessary for timing analysis with the PrimeTime tool, are generated by the HardCopy Files Wizard:

- `<project name>_hcpy.vo`—Verilog HDL output format
- `<project name>_hcpy_v.sdo`—Standard Delay Format Output File
- `<project name>_pt_hcpy_v.tcl`—Tcl script

These files are available in the `<project name>\hardcopy` directory. PrimeTime libraries for the HardCopy Stratix and Stratix devices are included with the Quartus II software.



Use the HardCopy Stratix libraries for PrimeTime to perform STA during timing analysis of designs targeted to `HARDCOPY_FPGA_PROTOTYPE` device.



For more information about static timing analysis, refer to the *Classic Timing Analyzer* and the *Synopsys PrimeTime Support* chapters in volume 3 of the *Quartus II Handbook*.

Early Power Estimation

You can use PowerPlay Early Power Estimation to estimate the amount of power your HardCopy Stratix or HardCopy APEX device will consume. This tool is available on the Altera website. Using the Early Power Estimator requires some knowledge of your design resources and specifications, including:

- Target device and package
- Clock networks used in the design
- Resource usage for LEs, DSP blocks, PLL, and RAM blocks
- High speed differential interfaces (HSDI), general I/O power consumption requirements, and pin counts
- Environmental and thermal conditions

HardCopy Stratix Early Power Estimation

The PowerPlay Early Power Estimator provides an initial estimate of I_{CC} for any HardCopy Stratix device based on typical conditions. This calculation saves significant time and effort in gaining a quick understanding of the power requirements for the device. No stimulus vectors are necessary for power estimation, which is established by the clock frequency and toggle rate in each clock domain.

This calculation should only be used as an estimation of power, not as a specification. The actual I_{CC} should be verified during operation because this estimate is sensitive to the actual logic in the device and the environmental operating conditions.



For more information about simulation-based power estimation, refer to the *Power Estimation and Analysis* Section in volume 3 of the *Quartus II Handbook*.



On average, HardCopy Stratix devices are expected to consume 40% less power than the equivalent FPGA.

HardCopy APEX Early Power Estimation

The PowerPlay Early Power Estimator can be run from the Altera website in the device support section (<http://www.altera.com/support/devices/dvs-index.html>). You cannot open this feature in the Quartus II software.

With the HardCopy APEX PowerPlay Early Power Estimator, you can estimate the power consumed by HardCopy APEX devices and design systems with the appropriate power budget. Refer to the web page for instructions on using the HardCopy APEX PowerPlay Early Power Estimator.



HardCopy APEX devices are generally expected to consume about 40% less power than the equivalent APEX 20KE or APEX 20KC FPGA devices.

Tcl Support for HardCopy Stratix

The Quartus II software also supports the HardCopy Stratix design flow at the command prompt using Tcl scripts.



For details on Quartus II support for Tcl scripting, refer to the *Tcl Scripting* chapter in volume 2 of the *Quartus II Handbook*.

Targeting Designs to HardCopy APEX Devices

Beginning with version 4.2, the Quartus II software supports targeting designs to HardCopy APEX device families. After compiling your design for one of the APEX 20KC or APEX 20KE FPGA devices supported by a HardCopy APEX device, run the HardCopy Files Wizard to generate the necessary set of files for HardCopy migration.

The HardCopy APEX device requires a different set of design files for migration than HardCopy Stratix. Table 5–5 shows the files collected for HardCopy APEX by the HardCopy Files Wizard.

Table 5–5. HardCopy APEX Files Collected by the HardCopy Files Wizard
<project name>.tan.rpt <project name>.asm.rpt <project name>.fit.rpt <project name>.hps.txt <project name>.map.rpt <project name>.pin <project name>.sof <project name>.qsf <project name>_cksum.datasheet <project name>_cpld.datasheet <project name>_hcpy.vo <project name>_hcpy_v.sdo <project name>_pt_hcpy_v.tcl <project name>_rba_pt_hcpy_v.tcl <project name>_target.datasheet <project name>_violations.datasheet

Refer to “Generating the HardCopy Design Database” on page 5–21 for information about generating the complete set of deliverables required for migrating the design to a HardCopy APEX device. After you have successfully run the HardCopy Files Wizard, you can submit your design archive to Altera to implement your design in a HardCopy device. You should contact Altera for more information about this process.

Conclusion

The methodology for designing HardCopy Stratix devices using the Quartus II software is the same as that for designing the Stratix FPGA equivalent. You can use the familiar Quartus II software tools and design flow, target designs to HardCopy Stratix devices, optimize designs for higher performance and lower power consumption than the Stratix FPGAs, and deliver the design database for migration to a HardCopy Stratix device. Compatible APEX FPGA designs can migrate to HardCopy APEX after compilation using the HardCopy Files Wizard to archive the design files. Submit the files to the HardCopy Design Center to complete the back-end migration.

Related Documents

For more information, refer to the following documentation:

- The *HardCopy Series Design Guidelines* chapter in volume 1 of the *HardCopy Series Handbook*.
- The *HardCopy Series Back-End Timing Closure* chapter in volume 1 of the *HardCopy Series Handbook*.

Document Revision History

Table 5–6 shows the revision history for this chapter.

Table 5–6. Document Revision History		
Date and Document Version	Changes Made	Summary of Changes
September 2008 v3.4	Updated chapter number and metadata.	—
June 2007 v3.3	Updated with the current Quartus II software version 7.1 information.	—
December 2006 v3.2	Updated revision history.	—
March 2006	Formerly chapter 20; no content change.	—
October 2005 v3.1	● Updated for technical contents for Quartus II 5.1 release ● Minor edits	Minor edits.
May 2005 v3.0	Added PowerPlay early Power estimator information.	—
January 2005 v2.0	This revision was previously the <i>Quartus® II Support for HardCopy Devices</i> chapter in the <i>Quartus II Development Software Handbook, v4.1</i> .	—
August 2003 v1.1	Overall edit; added Tcl script appendix.	—
June 2003 v1.0	Initial release of Chapter 20, Quartus II Support for HardCopy Stratix Devices.	—

Introduction

Advanced design techniques using Altera® HardCopy® Stratix® devices can yield tremendous performance improvements over the design implemented in a Stratix FPGA device. After you verify your Stratix FPGA design in system operation and are ready to migrate to a HardCopy Stratix device, additional device performance is possible through the migration. This chapter focuses on Quartus® II software advanced design techniques that apply to both Stratix FPGA devices and HardCopy Stratix devices. Use these techniques to increase your maximum clock frequency, improve input and output pin timing, and improve timing closure in HardCopy Stratix designs.



Every design is different. The techniques described in this chapter may not apply to every design, and may not yield the same level of improvement.

This document discusses the following topics:

- Planning Stratix FPGA design for HardCopy Stratix design conversion
- Using LogicLock™ regions in HardCopy Stratix designs
- Using Design Space Explorer (DSE) on HardCopy Stratix designs
- Design performance improvement example

Background Information

To understand the Quartus II software and device architecture, and to use the advanced design techniques described in this chapter, Altera recommends reading the *HardCopy Series Handbook* and the following chapters in the *Quartus II Software Handbook*:

- *Design Recommendations for Altera Devices and the Quartus II Design Assistant*
- *Design Optimization for Altera Devices*
- *Design Space Explorer*
- *Analyzing and Optimizing the Design Floorplan*
- *Netlist Optimizations and Physical Synthesis*

Planning Stratix FPGA Design for HardCopy Stratix Design Conversion

In order to achieve greater performance improvement in your HardCopy Stratix device, additional Quartus II software constraints and placement techniques in the `HARDCOPY_FPGA_PROTOTYPE` design project may be necessary. This does not mean changing the source hardware description language (HDL) code or functionality, but providing additional constraints in the Quartus II software that specifically impact HardCopy Stratix timing optimization.

Planning ahead for migration to the HardCopy design, while still modifying the `HARDCOPY_FPGA_PROTOTYPE` design, can improve design performance results. You must anticipate how portions of your FPGA design are placed and connected in the HardCopy device floorplan. The HardCopy device floorplan is smaller than the FPGA device floorplan, allowing use of the customized metal routing in HardCopy Stratix devices.

Partitioning Your Design

Partitioning your design into functional blocks is essential in multi-million gate designs. With a HardCopy Stratix device, you can implement approximately one million ASIC gates of logic. Therefore, Altera recommends hierarchical-design partitioning based on system functions.

When using a hierarchical- or incremental-design methodology, you must consider how your design is partitioned to achieve good results. Altera recommends the following practices for partitioning designs as documented in the *Design Recommendations for Altera Devices* chapter in volume 1 of the *Quartus II Development Software Handbook*:

- Partition your design at functional boundaries.
- Minimize the I/O connections between different partitions.
- Register all inputs and outputs of each block. This makes logic synchronous and avoids glitches and any delay penalty on signals that cross between partitions. Registering I/O pins typically eliminates the need to specify timing requirements for signals that connect between different blocks.
- Do not use glue logic or connection logic between hierarchical blocks. When you preserve hierarchy boundaries, glue logic is not merged with hierarchical blocks. Your synthesis software may optimize glue logic separately, which can degrade synthesis results and is not efficient when used with the LogicLock design methodology.
- Logic is not synthesized or optimized across partition boundaries. Any constant values (for example, signals set to GND), are not propagated across partitions.

- Do not use tri-state signals or bidirectional ports on hierarchical boundaries. If you use tri-state boundaries in a lower-level block, synthesis pushes the tri-state signals through the hierarchy to the top-level. This takes advantage of the tri-state drivers on the output pins of the Altera device. Since this requires optimizing through hierarchies, lower-level boundary tri-state signals are not supported with a block-level design methodology.
- Limit clocks to one per block. Partitioning your design into clock domains makes synthesis and timing analysis easier.
- Place state machines in separate blocks to speed optimization and provide greater encoding control.
- Separate timing-critical functions from non-timing-critical functions.
- Limit the critical timing path to one hierarchical block. Group the logic from several design blocks to ensure the critical path resides in one block.

These guidelines apply to all Altera device architectures including HardCopy Stratix devices. Partitioning functional boundaries to have all outputs immediately registered is crucial to using LogicLock regions effectively in HardCopy devices. With registered outputs, you allow the signals to leave a function block at the start of the clock period. This gives the signals more set-up time to reach their endpoints in the clock period. In large designs that are partitioned into multiple function blocks, the block-to-block interconnects are often the limiting factor for f_{MAX} performance. Registered outputs give the Quartus II Fitter the optimal place-and-route flexibility for interconnects between major function blocks.

Physical Synthesis Optimization

All physical synthesis settings in the Quartus II software can be used in the `HARDCOPY_FPGA_PROTOTYPE` design. These settings are found in the **Physical Synthesis Optimizations** section of the **Fitter Settings** dialog box (Assignments menu) and include the following settings:

- Physical synthesis for combinational logic
- Register duplication
- Register retiming

These settings can improve FPGA performance while developing the `HARDCOPY_FPGA_PROTOTYPE`. All modifications are passed along into the HardCopy Stratix project when you run the HardCopy Timing Optimization wizard. After running the HardCopy Timing Optimization wizard and subsequently opening the HardCopy project in the Quartus II software, these physical synthesis optimizations are disabled. No further modifications to the netlist are made.

Altera recommends physical synthesis optimizations for the `HARDCOPY_FPGA_PROTOTYPE`. The work done in the prototype enhances performance in the HardCopy Stratix device after migration. Duplicating combinational logic and registers can increase area utilization, which limits placement flexibility when designs exceed 95% logic element (LE) utilization. However, duplicating combinational logic and registers can help with performance by allowing critical paths to be duplicated when their endpoints must reach different areas of the device floorplan.



For more information on netlist and design optimization, refer to *Area Optimization and Timing Closure* in volume 2 of the *Quartus II Development Software Handbook*.

Using LogicLock Regions in HardCopy Stratix Designs

Create LogicLock regions in the `HARDCOPY_FPGA_PROTOTYPE` project and migrate the regions into the HardCopy Stratix optimization project using the Quartus II software. LogicLock regions can provide significant benefits in design performance by carefully isolating critical blocks of logic, including:

- MegaCore® IP functions
- I/O interfaces
- Reset or other critical logic feeding global clock lines
- Partitioned function blocks

You must compile your design initially without LogicLock regions present and review the timing analysis reports to determine if additional constraints or LogicLock regions are necessary. This process allows you to determine which function blocks or data paths require LogicLock regions.

Create LogicLock regions in the `HARDCOPY_FPGA_PROTOTYPE` design project in the Quartus II software. This transfers the LogicLock regions to the HardCopy design project after the HardCopy Timing Optimization Wizard is run. Although the Quartus II software transfers the contents of the LogicLock region, the area, location, and soft boundary settings revert to their default settings in the HardCopy project immediately after the HardCopy Timing Optimization Wizard is run.

If you are using LogicLock regions, Altera recommends you use the **Migration Only** setting in the HardCopy Timing Optimization Wizard to create the HardCopy design project. You should not compile your design automatically using the **Full Compilation** or **Migrate and Compile** options in the wizard. Open the HardCopy design project and verify that the LogicLock region properties meet your desired settings before compiling the HardCopy optimization project. LogicLock soft regions are

turned on by default in the HardCopy Stratix design. While this does allow the Fitter to place all logic in your design with fewer restrictions, it is not optimal for performance improvement in the HardCopy Stratix design.

Recommended LogicLock Settings for HardCopy Stratix Designs

Altera recommends the following LogicLock region settings for the `HARDCOPY_FPGA_PROTOTYPE`:

- Turn on **Reserve Unused Logic**
- Turn off **Soft Region**
- Select either **Auto** or **Fixed** as the **Size** (design-dependent)
- Select either **Floating** or **Locked** as the **Location** (design-dependent)

When using the **Reserve Unused Logic** setting in a design with high resource utilization (> 95% LE utilization), and a large number of LogicLock regions, the design may not fit in the device. Turning off **Reserve Unused Logic** in less critical LogicLock regions can help Fitter placement. The LEs allowed to float in placement and be packed into unused LEs of LogicLock regions may not be placed optimally after migration to the HardCopy Stratix device since they are merged with other LogicLock regions.

After running the HardCopy Timing Optimization Wizard, the LogicLock region properties are reset to their default conditions. This allows a successful and immediate placement of your design in the Quartus II software. You can further refine the LogicLock region properties for additional benefits.

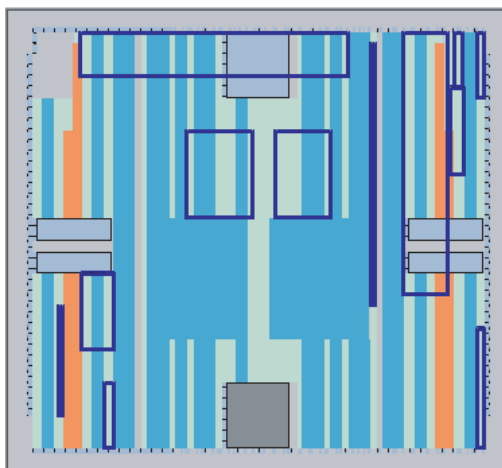
Altera recommends using the following properties for LogicLock regions in the HardCopy design project:

- Turn off **Soft Region**
- Select either **Auto** or **Fixed** as the **Size** after you are satisfied with the placement and timing result of a LogicLock region in a successful HardCopy Stratix compilation
- Select either **Floating** or **Locked** as the **Location** after you are satisfied with the placement and timing results
- **Reserve Unused Logic** is not applicable in the HardCopy Stratix device placement because logic array block (LAB) contents can not be changed after the HardCopy Timing Optimization Wizard is run

An example of a well partitioned design using LogicLock regions effectively for some portions of the design is shown in [Figure 6–1](#). Only the most critical logic functions required are placed in LogicLock regions in order to achieve the desired performance in the HardCopy Stratix

device. The dark blue rectangles shown in [Figure 6–1](#) are the user-assigned LogicLock regions that have fixed locations. In this example, the design needed to be constrained by LogicLock regions first inside the `HARDCOPY_FPGA_PROTOTYPE` with **Reserve Unused Logic** turned off in **Properties** in LogicLock regions. This selection allows the Quartus II software to isolate and compact the logic of these blocks in the `HARDCOPY_FPGA_PROTOTYPE` such that the placement is tightly controlled in the HardCopy Stratix device.

Figure 6–1. A Well Partitioned Design



In the example shown in [Figure 6–1](#), once suitable locations were identified for LogicLock regions, the LogicLock region properties were changed from floating to locked. The Quartus II software can then reproduce their placement in subsequent compilations, while focusing attention on fixing other portions of the design.

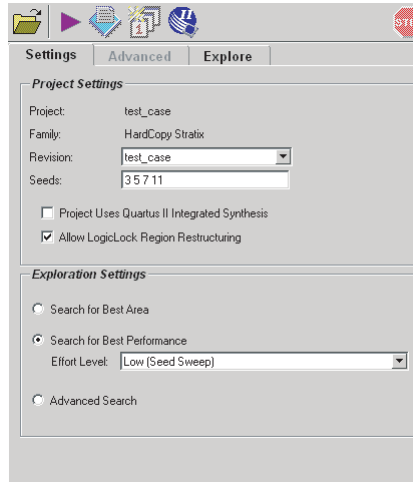
Using Design Space Explorer for HardCopy Stratix Designs

The DSE feature in the Quartus II software allows you to evaluate various compilation settings to achieve the best results for your FPGA designs. DSE can also be used in the HardCopy Stratix project after running the HardCopy Timing Optimization wizard.

Only some of the DSE settings affect HardCopy Stratix designs because HDL synthesis and physical optimization have been completed on the FPGA. No logic restructuring can occur after using the HardCopy Timing Optimization wizard. When you compile your design, the placement of LABs is optimized in the HardCopy Stratix device. To access the DSE GUI

in your open project in the Quartus II software, select **Launch Design Space Explorer** (Tools menu). An example of the DSE GUI and DSE Settings window for the HardCopy Stratix device is shown in [Figure 6–2](#).

Figure 6–2. DSE Settings Window in the DSE GUI



Recommended DSE Settings for HardCopy Stratix Designs

The HardCopy Stratix design does not require all advanced settings or effort-level settings in DSE. Altera recommends using the following settings in DSE for HardCopy Stratix designs:

- In the **Settings** tab ([Figure 6–2](#)), make the following selections:
 - Under **Project Settings**, enter several seed numbers in the **Seeds** box. Each seed number requires one full compile of the HardCopy Stratix project.
 - Under **Project Settings**, select **Allow LogicLock Region Restructuring**.
 - Under **Exploration Settings**, select **Search for Best Performance**, and select **Low (Seed Sweep)** from the **Effort Level** menu.
- Turn on **Archive all Compilations** (Options menu).

After running DSE with the seed sweep setting, view the results and identify which seed settings produced the best compilation results. Use the archive of the identified seed, or merge the compilation settings and seed number from the DSE archived project into your primary HardCopy Stratix project.

Performance Improvement Example

With the design used for the performance improvement example in this section, the designer was seeking performance improvement on an HC1S30F780 design for an intellectual property (IP) core consisting of approximately 5200 LEs, 75,000 bits of memory, and two digital signal processing (DSP) multiplier accumulators (MACs). The final application needed to fit in a reserved portion of the HC1S30 device floorplan, so the entire block of IP was initially bounded in a single LogicLock region. The IP block was evaluated as a stand-alone block.

Initial Design Example Settings

The default settings in the Quartus II software version 4.2 were used, with the following initial constraints added:

- The device was set to the target Stratix FPGA device which is the prototype for the HC1S30F780 device:

```
set_global_assignment -name DEVICE  
EP1S30F780C6_HARDCOPY_FPGA_PROTOTYPE
```
- A LogicLock region was created for the block to bound it in the reserved region.
- The LogicLock region properties were set to **Auto Size** and **Floating Location**, and **Reserve Unused Logic** was turned on:

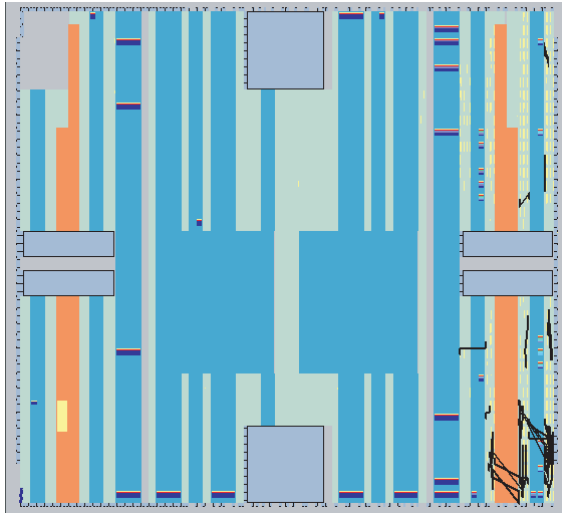
```
set_global_assignment -name LL_STATE FLOATING  
set_global_assignment -name LL_AUTO_SIZE ON  
set_global_assignment -name LL_RESERVED OFF  
set_global_assignment -name LL_SOFT OFF
```
- Virtual I/O pins were used for the ports of the core since this core does not interface to pins in the parent design, and the I/O pins were placed outside the LogicLock region and are represented as registers in LEs.

The initial compilation results yielded 65.30-MHz f_{MAX} in the FPGA. The block was constrained through virtual I/O pins and a LogicLock region to keep the logic from spreading throughout the floorplan.

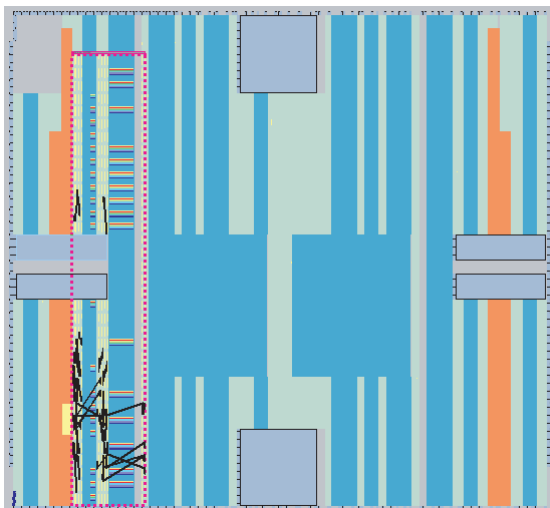
The initial compile-relevant statistics for this example are provided in [Table 6–1](#).

<i>Table 6–1. Initial Compilation Statistics</i>	
Result Type	Results
f_{MAX}	65.30 MHz
Total logic elements (LEs)	5,187/32,470 (15%)
Total LABs	564/3,247 (17%)
M512 blocks	20/295 (6%)
M4K blocks	16/171 (9%)
M-RAM blocks	0/2 (0%)
Total memory bits	74,752/2,137,536 (3%)
Total RAM block bits	85,248/2,137,536 (3%)
DSP block 9-bit elements	2/96 (2%)

The design project was migrated to the HardCopy device using the HardCopy Timing Optimization wizard and was compiled. The default settings of the LogicLock region in a HardCopy Stratix project in the Quartus II software have the **Soft Region** option turned on. With this setting, the HardCopy Stratix compilation yields an f_{MAX} of 66.48 MHz, mainly due to the Fitter placement being scattered in an open design ([Figure 6–3](#)). Because the **Soft Region** is set to on, the LogicLock region is not bounded. This is not an optimal placement in the HardCopy Stratix design and is not the best possible performance.

Figure 6–3. HardCopy Stratix Device Floorplan with Soft Region On

To keep the LogicLock region contents bounded in the final placement in the HardCopy Stratix device floorplan, turn off the **Soft Region** option. After turning off the **Soft Region** option and compiling the HardCopy Stratix design, the result is an f_{MAX} of 88.14 MHz—a gain of 33% over the Stratix FPGA device performance. The bounded placement in the LogicLock region helps to achieve performance improvement in well-partitioned design blocks by taking advantage of the smaller die size and custom metal routing interconnect of the HardCopy Stratix device. The floorplan of the bounded LogicLock region is visible in [Figure 6–4](#). In this figure, you can see the difference in disabling the Soft Region setting in the HardCopy Stratix design.

Figure 6–4. HardCopy Stratix Device Floorplan with Soft Region Off

Using Analysis and Synthesis Settings for Performance Improvement

After establishing the baseline for improvement for this design of 65.30 MHz FPGA/88.14 MHz HardCopy, you can gain additional performance improvement in the Stratix FPGA and HardCopy Stratix devices using the available features in the Quartus II software.

Changing the **Analysis & Synthesis Effort** from **Balanced** to **Speed** yields additional benefit in performance, but at the cost of additional LE resources. The Tcl command for this assignment is as follows:

```
set_global_assignment -name  
STRATIX_OPTIMIZATION_TECHNIQUE SPEED
```

The relevant compilation results of the FPGA are provided in [Table 6–2](#).

Table 6–2. Relevant Compile Results	
Result Type	Results
f_{MAX}	68.88 MHz
Total logic elements	5,508/32,470 (16%)
Total LABs	598/3,247 (18%)
M512 blocks	20/295 (6%)
M4K blocks	16/171 (9%)
M-RAM blocks	0/2 (0%)
Total memory bits	74,752/2,137,536 (3%)
Total RAM block bits	85,248/2,137,536 (3%)
DSP block 9-bit elements	2/96 (2%)

Increasing the LE resources by 6% only yielded an additional 3 MHz in performance in the FPGA, without using additional settings. However, after migrating this design to the HardCopy Stratix design and compiling it, the performance did not improve over the previous HardCopy Stratix design compile, and was slightly worse in performance at 87.34 MHz. This shows that the Quartus II software synthesis was very effective with the **Synthesis Effort Level** set to **Balanced**, and there was only marginal improvement in the FPGA when this option was set to **Speed**.

The next settings activated in this example were the **Synthesis Netlist Optimizations** shown below in Tcl format for WYSIWYG synthesis remapping and gate-level retiming after synthesis mapping:

```
set_global_assignment -name
ADV_NETLIST_OPT_SYNTH_WYSIWYG_REMAP ON

set_global_assignment -name
ADV_NETLIST_OPT_SYNTH_GATE_RETIME ON
```

Making these settings in the FPGA while leaving **Analysis & Synthesis Effort** set to **Speed** yielded some additional improvement in the FPGA as shown in [Table 6–3](#).

<i>Table 6–3. Results of Analysis & Synthesis Effort Set to Speed</i>	
Result Type	Results
f_{MAX}	70.28 MHz
Total logic elements	5,515/32,470 (16%)
Total LABs	597/3,247 (18%)

The WYSIWYG resynthesis added a minimal increase in LEs over the speed setting, and the design performance improved by 2 MHz in the FPGA. Using the HardCopy Timing Optimization wizard to migrate the design to HardCopy and subsequently compiling the HardCopy Stratix design, we find that performance is not improved beyond previous compiles, with an f_{MAX} of 86.58 MHz.

The Quartus II software automatically optimizes state machines and restructures multiplexers when these settings are set to **Auto** in the **Analysis & Synthesis** settings. Changing these options from **Auto** usually does not yield performance improvement.

For example, changing the multiplexer restructuring and state machine processing settings from both set to **Auto**, to **On** and **One-Hot**, respectively, actually hurt performance, not allowing the Quartus II software to determine the optimization on a case-by-case basis. With these settings, the FPGA compiled to an f_{MAX} of 65.99 MHz, and the HardCopy Stratix design only performed at 83.77 MHz. For this design example, it is better to leave these settings to **Auto** as seen in the Tcl assignments in the [“Using Fitter Assignments and Physical Synthesis Optimizations for Performance Improvement”](#) section, and allow the Quartus II software to determine when to use these features.

Using Fitter Assignments and Physical Synthesis Optimizations for Performance Improvement

After exploring the Analysis & Synthesis optimization settings in the Quartus II software, you can use the Fitter Settings and Physical Synthesis Optimization features to gain further performance improvement in your Stratix FPGA and HardCopy Stratix devices. In this design example, multiplexer and state machine restructuring settings have been set to **Auto**, and the **Synthesis Optimization Technique** is set

for **Speed**. The **Fitter effort** is set to **Standard Fit (highest effort)**. The next features enabled are the **Physical Synthesis Optimizations** as seen in the Tcl assignments below and in [Figure 6–5](#):

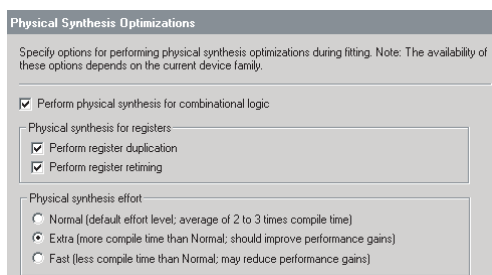
```
set_global_assignment -name
PHYSICAL_SYNTHESIS_COMBO_LOGIC ON

set_global_assignment -name
PHYSICAL_SYNTHESIS_REGISTER_DUPLICATION ON

set_global_assignment -name
PHYSICAL_SYNTHESIS_REGISTER_RETIMING ON

set_global_assignment -name PHYSICAL_SYNTHESIS_EFFORT
EXTRA
```

Figure 6–5. Physical Synthesis Optimization Settings



The compiled design shows a performance increase in the FPGA, running at an f_{MAX} of 74.34 MHz, requiring additional LE resources as a result of the physical synthesis and logic duplication. In this example, you can see how performance can be increased in the Stratix FPGA device at the expense of additional LE resources, as this design's LE resources grew almost 12% over the beginning compilation. The compiled FPGA design's statistics are provided in [Table 6–4](#).

Table 6–4. Compiled FPGA Design Statistics

Result Type	Results
f_{MAX}	74.34 MHz
Total logic elements	5,781/32,470 (17%)
Total LABs	610/3,247 (18%)

Running the HardCopy Timing Optimization wizard on this design and compiling the HardCopy Stratix project yields an f_{MAX} of 92.01 MHz, a 24% improvement over the FPGA timing.

Design Space Explorer

The available Fitter Settings produce an additional performance improvement. The DSE feature is used on the Stratix FPGA device to run through the various seeds in the design and select the best seed point to use for future compiles. This can often yield additional performance benefits as the Quartus II software further refines placement of the LEs and performs clustering of associated logic together.

For this design example, DSE was run with high effort (physical synthesis) and multiple placement seeds. Table 6–5 shows the DSE results. The base compile matches the fifth compile in the DSE variations, showing that the work already done on the design before DSE was optimal. The FPGA project was optimized before running DSE.

Table 6–5. DSE Results		
Compile Point	Clock Period: CLK	Logic Cells
Base (Best)	13.451 ns (74.34 MHz)	5,781
1	13.954 ns	5,703
2	13.712 ns	6,447
3	14.615 ns	5,777
4	13.911 ns	5,742
5	13.451 ns	5,781
6	14.838 ns	5,407
7	14.177 ns	5,751
8	14.479 ns	5,827
9	14.863 ns	5,596
10	14.662 ns	5,605
11	14.250 ns	5,710
12	14.016 ns	5,708
13	13.840 ns	5,802
14	13.681 ns	5,788
15	14.829 ns	5,644

Additional correlation is seen inside the `<project>.dse.rpt` file, showing the summary of assignments used for each compile inside the Quartus II software. The base compile settings and the fifth compile settings show good correlation, as shown in [Table 6–6](#). The `MUX_RESTRUCTURE` setting did not have any effect on the design performance. This may be due to an already efficient HDL coding for multiplexer structures, requiring no optimization.

Table 6–6. Base Compile and Fifth Compile Correlation

Setting	New Value	Base Value
PHYSICAL_SYNTHESIS_REGISTER_RETIMING	ON	ON
SEED	1	1
STATE_MACHINE_PROCESSING	AUTO	AUTO
MUX_RESTRUCTURE	OFF	AUTO
PHYSICAL_SYNTHESIS_COMBO_LOGIC	ON	ON
FITTER_EFFORT	STANDARD FIT	STANDARD FIT
AUTO_PACKED_REGISTERS_STRATIX	NORMAL	NORMAL
PHYSICAL_SYNTHESIS_REGISTER_DUPLICATION	ON	ON
ADV_NETLIST_OPT_SYNTH_GATE_RETIME	ON	ON
STRATIX_OPTIMIZATION_TECHNIQUE	SPEED	SPEED
PHYSICAL_SYNTHESIS_EFFORT	EXTRA	EXTRA

The information presented in [Table 6–6](#) confirms that the FPGA Prototype device has been optimized as much as possible without manual floorplan adjustments.

Design Space Explorer for HardCopy Stratix Devices

Migrating this compiled design to the HardCopy Stratix project and compiling the HardCopy Stratix design optimization, results in a design performance of 92.01 MHz. The next task is to run DSE on the HardCopy Stratix project using **Low Effort (Seed Sweep)** in the **Exploration Settings**, and entering a range of seed numbers with which to compile the project.

The results of the DSE run with the **Seed Sweep** option are summarized in [Table 6–7](#).

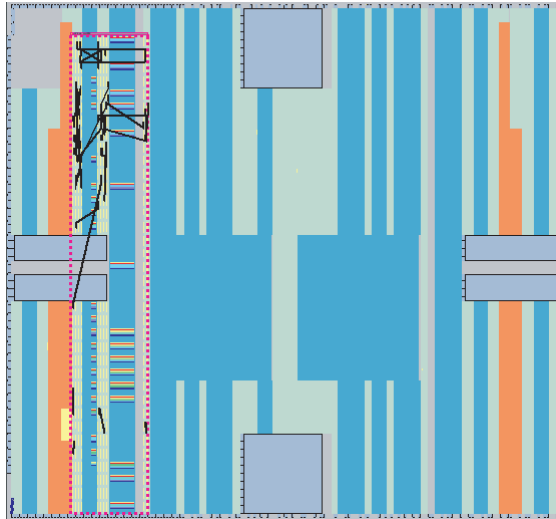
<i>Table 6–7. DSE Results Run with Seed Sweep</i>	
Compile Point	Clock Period: CLK
Base (Best)	10.868 ns
1	11.710 ns
2	11.040 ns
3	10.790 ns
4	10.945 ns
5	11.154 ns
6	11.707 ns
7	11.648 ns
8	11.476 ns
9	11.423 ns
10	11.449 ns

The results in [Table 6–7](#) illustrate how the **Seed Sweep** option in DSE provides additional improvement in the HardCopy Stratix design, even after DSE has been run on the Stratix FPGA project. In this example, compile point 3 using seed value = 4 turns out to be slightly beneficial over other seeds in the Fitter Placement. The HardCopy Stratix device has an f_{MAX} of 92.71 MHz.

Back-Annotation and Location Assignment Adjustments

Another technique available for improving performance in the HardCopy Stratix design is manually adjusting placement and back-annotating location assignments from the placement results. These techniques should be one of the last steps taken for design optimization of HardCopy Stratix devices.

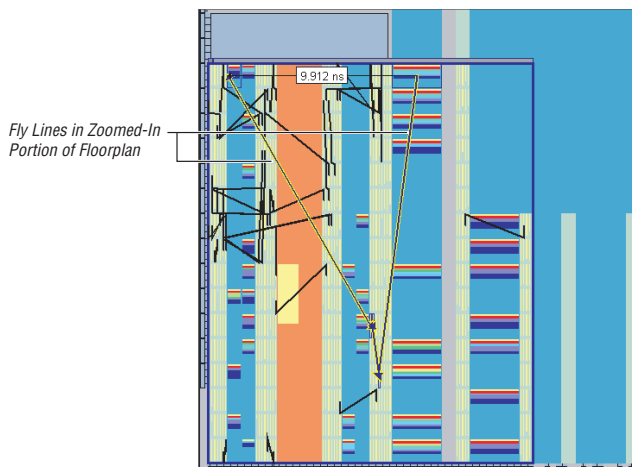
Observing the floorplan of the 92.71 MHz compile ([Figure 6–6](#)), the placement of the LogicLock region is stretched vertically, and additional improvement is possible if the aspect ratio of the LogicLock region is defined, and placement in it is refined.

Figure 6–6. Vertically Stretched LogicLock Region

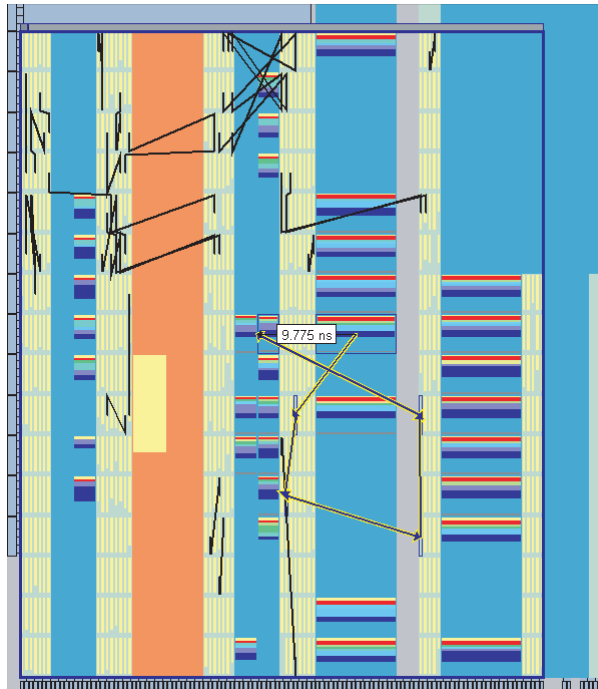
This floorplan would be better optimized if the LogicLock region had a more square shape, helping the paths that go from memory-to-memory, by containing the M4K and M512 memory blocks in a smaller space, and allowing LAB placement to be adjusted by the Fitter. In the HardCopy Stratix device, signals are routed between LABs, DSP blocks, and memory blocks using the customized metal layers. The reconfigurable routing tracks in the Stratix FPGA device limit the routing paths and delays between elements in the HardCopy Stratix device. This flexibility allows for aspect ratio changes in LogicLock regions, so the raw distance between points becomes the critical factor, and not the usage of available routing resources in the FPGA.

For the final placement optimization in this example, the LogicLock region was fixed in a square region that encompassed two columns of M4K blocks, four columns of M512 blocks, two columns of DSP blocks, and enough LABs to fit the remaining resources required. After compiling the design with these new LogicLock assignments, the performance increased to 93.46 MHz in the HardCopy Stratix device. The critical path and LogicLock region location can be seen in the zoomed-in area of the floorplan (Figure 6–7).

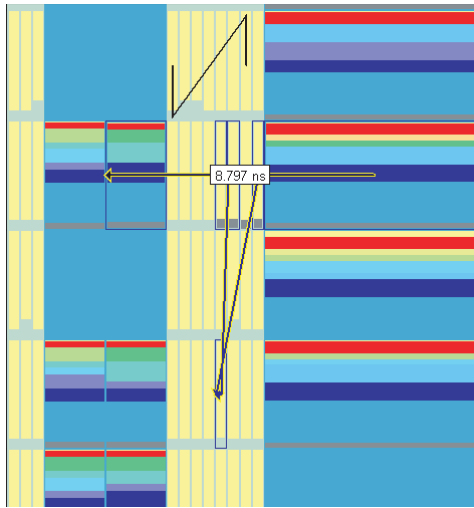
You can see in Figure 6–7 that the critical path shown is from an M4K block to an M512 block through several levels of logic. The placement of the memory blocks can be optimized manually, since the LogicLock region contains more memory blocks than necessary.

Figure 6–7. Critical Path and LogicLock Region

Using the critical path “fly lines” as a guide for placement optimization, manual location assignments were made for some of the M512 and M4K instances used in the design. The resulting compile improved the f_{MAX} to 94.67 MHz. The new critical path (Figure 6–8) shows how placement of all path elements are confined to a much smaller area. As a result, the routing distances and delays are smaller through the path.

Figure 6–8. New Critical Path

Examining this new critical path placement, you can see that there is room for further performance improvement through additional location assignments. The current slowest path is 9.775 ns of delay. Manually moving the LABs in this critical path and placing them between the M4K and M512 endpoints, and subsequently recompiling, shows improved results not only for this path, but for several other paths, as this path contained a major timing bottleneck. The critical path between this start and endpoint was reduced to 8.797 ns (Figure 6–9). However, the entire design only improved to 100.30 MHz because other paths are now the slowest paths in the design. This illustrates that fixing one major bottleneck path can raise the entire design performance since one high fanout node can affect multiple timing paths, as was the case in this example.

Figure 6–9. Improved Results

In summary, this design example started with 65.30 MHz in the Stratix FPGA device, and was improved to 74.34 MHz. It was then taken from the Stratix FPGA device compile and improved to 100.30 MHz in the HardCopy Stratix design, for a performance improvement of 35%.

Conclusion

Using performance-optimization techniques specifically for HardCopy Stratix devices can achieve significant performance improvement over the Stratix FPGA prototype device. Many of these changes must be incorporated up-front in the `HARDCOPY_FPGA_PROTOTYPE` so that your design is properly prepared for performance improvement after running the HardCopy Timing Optimization wizard.

The example discussed in this chapter demonstrates the process for performance improvement and various features in the Quartus II software available for use when optimizing your Stratix FPGA prototype and HardCopy Stratix device. It also demonstrates the importance of planning ahead for the HardCopy Stratix design implementation while continuing to work in the `HARDCOPY_FPGA_PROTOTYPE` design if you are going to seek performance improvement in the HardCopy Stratix device.

Document Revision History

Table 6–8 shows the revision history for this chapter.

<i>Table 6–8. Document Revision History</i>		
Date and Document Version	Changes Made	Summary of Changes
September 2008 v1.4	Updated chapter number and metadata.	—
June 2007 v1.3	• Updated the “Background Information” section. • Completed minor typographical updates.	—
December 2006 v1.2	Updated revision history.	—
March 2006	Formerly chapter 21; no content change.	—
October 2005 v1.1	• Updated graphics • Minor edits	—
July 2005 v1.0	Initial release of Chapter 21, Design Guidelines for HardCopy Stratix Performance Improvement.	—



Section II. HardCopy APEX Device Family Data Sheet

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

This section contains the following:

- Chapter 7, Introduction to HardCopy APEX Devices
- Chapter 8, Description, Architecture, and Features
- Chapter 9, Boundary-Scan Support
- Chapter 10, Operating Conditions

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the complete handbook.

Introduction

HardCopy® APEX™ devices enable high-density APEX 20KE device technology to be used in high-volume applications where significant cost reduction is desired. HardCopy APEX devices are physically and functionally compatible with APEX 20KC and APEX 20KE devices. They combine the time-to-market advantage, performance, and flexibility of APEX 20KE devices with the ability to move to high-volume, low-cost devices for production. The migration process from an APEX 20KE device to a HardCopy APEX device is fully automated, with designer involvement limited to providing a few Quartus® II software-generated output files.

Features...

HardCopy APEX devices are manufactured using an 0.18-μm CMOS six-layer-metal process technology:

- Preserves functionality of a configured APEX 20KC or APEX 20KE device
- Pin-compatible with APEX 20KC or APEX 20KE devices
- Meets or exceeds timing of configured APEX 20KE and APEX 20KC devices
- Optional emulation of original programmable logic device (PLD) programming sequence
- High-performance, low-power device
- MultiCore architecture integrating embedded memory and look-up table (LUT) logic used for register-intensive functions
- Embedded system blocks (ESBs) used to implement memory functions, including first-in first-out (FIFO) buffers, dual-port RAM, and content-addressable memory (CAM)
- Customization performed through metallization layers

High-density architecture:

- 400,000 to 1.5 million typical gates (Table 7-1)
- Up to 51,840 logic elements (LEs)
- Up to 442,368 RAM bits that can be used without reducing available logic

Table 7-1. HardCopy APEX Device Features *Note (1)*

Feature	HC20K400	HC20K600	HC20K1000	HC20K1500
Maximum system gates	1,052,000	1,537,000	1,772,000	2,392,000
Typical gates	400,000	600,000	1,000,000	1,500,000
LEs	16,640	24,320	38,400	51,840
ESBs	104	152	160	216
Maximum RAM bits	212,992	311,296	327,680	442,368
Phase-locked loops (PLLs)	4	4	4	4
Maximum macrocells	1,664	2,432	2,560	3,456
Maximum user I/O pins	488	588	708	808

Note to Table 7-1:

- (1) The embedded IEEE Std. 1149.1 Joint Test Action Group (JTAG) boundary-scan circuitry contributes up to 57,000 additional gates.

...and More Features

Low-power operation:

- 1.8-V supply voltage (Table 7-2)
- MultiVolt I/O support for 1.8-, 2.5-, and 3.3-V interfaces
- ESBs offering power-saving mode

Flexible clock management circuitry with up to four phase-locked loops (PLLs):

- Built-in low-skew clock tree
- Up to eight global clock signals
- ClockLock feature reducing clock delay and skew
- ClockBoost feature providing clock multiplication and division
- ClockShift feature providing clock phase and delay shifting

Powerful I/O features:

- Compliant with peripheral component interconnect Special Interest Group (PCI SIG) *PCI Local Bus Specification, Revision 2.2* for 3.3-V operation at 33 or 66 MHz and 32 or 64 bits

- Support for high-speed external memories, including double-data rate (DDR), synchronous dynamic RAM (SDRAM), and zero-bus-turnaround (ZBT) static RAM (SRAM)
- 16 input and 16 output LVDS channels
- Fast t_{CO} and t_{SU} times for complex logic
- MultiVolt I/O support for 1.8-V, 2.5-V, and 3.3-V interfaces
- Individual tri-state output enable control for each pin
- Output slew-rate control to reduce switching noise
- Support for advanced I/O standards, including LVDS, LVPECL, PCI-X, AGP, CTT, SSTL-3 and SSTL-2, GTL+, and HSTL Class I
- Supports hot-socketing operation

Table 7–2. HardCopy APEX Device Supply Voltages

Feature	Voltage
Internal supply voltage (V_{CCINT})	1.8 V
MultiVolt I/O interface voltage levels (V_{CCIO})	1.8 V, 2.5 V, 3.3 V, 5.0 V (1)

Note to Table 7–2:

- (1) HardCopy APEX devices can be 5.0-V tolerant by using an external resistor.

HardCopy APEX device implementation features:

- Customized interconnect for each design
- HardCopy APEX devices preserve APEX 20K device MegaLAB structure, LEs, ESBs, I/O element (IOE), PLLs, and LVDS circuitry
- Up to four metal layers customizable for customer designs
- Completely automated proprietary design migration flow
 - Testability analysis and fix
 - Automatic test pattern generation (ATPG)
 - Automatic place and route
 - Static timing analysis
 - Static functional verification
 - Physical verification

Tables 7–3 through 7–6 show the HardCopy APEX device ball-grid array (BGA) and FineLine BGA package options, I/O counts, and sizes.

Table 7–3. HardCopy APEX Device BGA Package Options and I/O Count
Note (1)

Device	652-Pin BGA
HC20K400	488
HC20K600	488
HC20K1000	488
HC20K1500	488

Table 7–4. HardCopy APEX Device FineLine BGA Package Options and I/O Count
Note (1)

Device	672-Pin	1,020-Pin
HC20K400	488	—
HC20K600	508	588
HC20K1000	508	708
HC20K1500	—	808

Note to Tables 7–3 and 7–4:

(1) I/O counts include dedicated input and clock pins.

Table 7–5. HardCopy APEX Device BGA Package Sizes

Feature	652-Pin BGA
Pitch (mm)	1.27
Area (mm ²)	2,025
Length × width (mm × mm)	45.0 × 45.0

Table 7–6. HardCopy APEX Device FineLine BGA Package Sizes

Feature	672-Pin	1,020-Pin
Pitch (mm)	1.00	1.00
Area (mm ²)	729	1,089
Length × width (mm × mm)	27 × 27	33 × 33

Document Revision History

Table 7–7 shows the revision history for this chapter.

<i>Table 7–7. Document Revision History</i>		
Date and Document Version	Changes Made	Summary of Changes
September 2008, v2.3	Updated chapter number and metadata.	—
June 2007, v2.2	Minor text edits.	—
December 2006 v2.1	Updated revision history.	—
March 2006	Formerly chapter 9; no content change.	—
January 2005 v2.0	Update device names and other minor textual changes	—
June 2003 v1.0	Initial release of Chapter 9, <i>Introduction to HardCopy APEX Devices</i> , in the HardCopy Device Handbook	—

Introduction

HardCopy® APEX™ devices extend the flexibility of high-density FPGAs to a cost-effective, high-volume production solution. The migration process from an Altera® FPGA to a HardCopy APEX device offers seamless migration of a high-density system-on-a-programmable-chip (SOPC) design to a low-cost alternative device with minimal risk. Using HardCopy APEX devices, Altera's SOPC solutions can be leveraged from prototype to production, while reducing costs and speeding time-to-market.

A significant benefit of HardCopy devices is that customers do not need to be involved in the device migration process. Unlike application-specific integrated circuit (ASIC) development, the HardCopy design flow does not require generation of test benches, test vectors, or timing and functional simulation. The HardCopy migration process only requires the Quartus® II software-generated output files from a fully functional APEX 20KE or APEX 20KC device. Altera performs the migration and delivers functional prototypes in as few as seven weeks.

A risk-free alternative to ASICs, HardCopy APEX devices are customizable, full-featured devices created by Altera's proprietary design migration methodology. They are based on Altera's industry-leading high-density device architecture and use an area-efficient sea-of-logic-elements (SOLE) core.

HardCopy APEX devices retain all the same features as the APEX 20KE and APEX 20KC devices, which combine the strength of LUT-based and product-term-based devices in conjunction with the same embedded memory structures. All routing resources that were programmable in the APEX 20K device family are replaced by custom interconnect, resulting in a considerable die size reduction and subsequent cost saving.

The SRAM configuration cells of the original FPGA are replaced in HardCopy APEX devices by metal elements, which define the function of each logic element (LE), embedded memory, and I/O cell in the device. These resources are connected to each other using the same metallization layers. Once a HardCopy APEX device has been manufactured, the functionality of the device is fixed and no programming is possible. Altera performs the migration of the original FPGA design to an equivalent HardCopy APEX device using a proprietary design migration flow.

The migration of a FPGA to a HardCopy APEX device begins with a user design that has been implemented in an APEX 20KE or APEX 20KC device. [Table 8–1](#) shows the device equivalence for HardCopy and APEX 20KE or APEX 20KC devices.

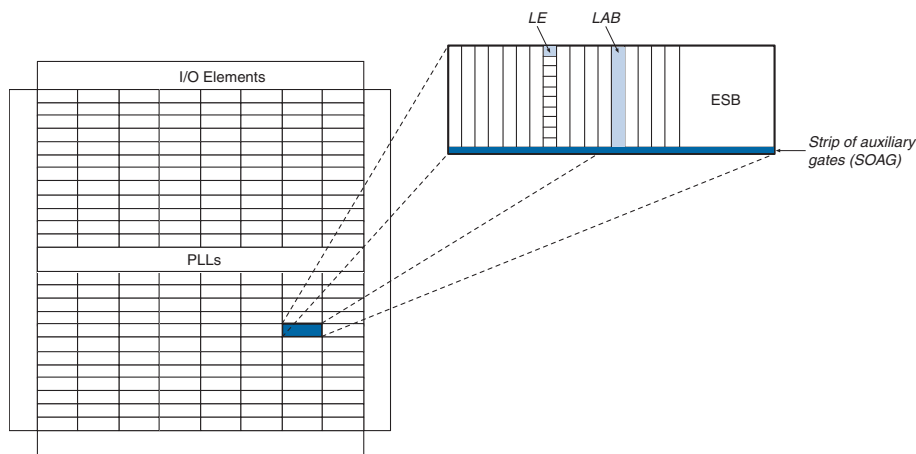
Table 8–1. HardCopy and APEX 20KE or APEX 20C Device Equivalence

HardCopy APEX Device	APEX 20KE Device	APEX 20KC Device
HC20K1500	EP20K1500E	EP20K1500C
HC20K1000	EP20K1000E	EP20K1000C
HC20K600	EP20K600E	EP20K600C
HC20K400	EP20K400E	EP20K400C



To ensure HardCopy device performance and functionality, the APEX 20K design must be completely debugged before committing the design to HardCopy device migration.

HardCopy APEX device implementation begins with extracting the Quartus II software-generated SRAM Object File (.sof) and converting its connectivity information into a structural Verilog HDL netlist. This netlist is then placed and routed in a similar fashion to a gate array. There are no dedicated routing channels. The router can exploit all available metal layers (up to four) and route over LE cells and other functional blocks. Altera's proprietary architecture and design methodology will guarantee virtually 100% routing of any APEX 20KE or APEX 20KC design compiled and fitted successfully using the Quartus II software. Place and route is timing-driven and will comply with the timing constraints of the original FPGA design as specified in the Quartus II software. [Figure 8–1](#) shows a diagram of the HardCopy APEX device architecture.

Figure 8–1. HardCopy APEX Device Architecture

The strip of auxiliary gates (SOAG) is an Altera proprietary feature designed into the HardCopy APEX device and is used during the HardCopy device implementation process. The SOAG structures can be configured into several different types of functions through the use of metallization. For example, high fanout signals require adequate buffering, so buffers are built out of SOAG cells for this purpose.

HardCopy APEX devices include the same advanced features as the APEX 20KE and APEX 20KC devices, such as enhanced I/O standard support, content-addressable memory (CAM), additional global clocks, and enhanced ClockLock circuitry. [Table 8–2](#) lists the features included in HardCopy APEX devices.

Table 8–2. HardCopy APEX Device Features (Part 1 of 2)

Feature	HardCopy Devices
MultiCore system integration	Full support
Hot-socketing support	Full support
32-/64-bit, 33-MHz PCI	Full compliance
32-/64-bit, 66-MHz PCI	Full compliance
MultiVolt I/O operation	1.8-V, 2.5-V, or 3.3-V V_{CCIO} V_{CCIO} selected bank by bank 5.0-V tolerant with use of external resistor

Table 8–2. HardCopy APEX Device Features (Part 2 of 2)

Feature	HardCopy Devices
ClockLock support	Clock delay reduction $m/(n \times v)$ clock multiplication Drive ClockLock output off-chip External clock feedback ClockShift circuitry LVDS support Up to four PLLs ClockShift, clock phase adjustment
Dedicated clock and input pins	Eight
I/O standard support	1.8-V, 2.5-V, 3.3-V, 5.0-V I/O 3.3-V PCI and PCI-X 3.3-V AGP CTT GTL+ LVCMOS LVTTTL True-LVDS and LVPECL data pins LVDS and LVPECL clock pins HSTL class I PCI-X SSTL-2 class I and II SSTL-3 class I and II
Memory support	CAM Dual-port RAM FIFO RAM ROM

All HardCopy APEX devices are tested using automatic test pattern generation (ATPG) vectors prior to shipment. For fully synchronous designs near 100%, fault coverage can be achieved through the built-in full-scan architecture. ATPG vectors allow the designer to focus on simulation and design verification.

Because the configuration of HardCopy APEX devices is built-in during manufacture, they cannot be configured in-system. However, if the APEX 20KE or APEX 20KC device configuration sequence must be emulated, the HardCopy APEX device has this capability.



All of the device features of APEX 20KE and APEX 20KC devices are available in HardCopy APEX devices. For a detailed description of these device features, refer to the *APEX 20K Programmable Logic Device Family Data Sheet* and the *APEX 20KC Programmable Logic Device Family Data Sheet*.

Differences Between HardCopy APEX and APEX 20K FPGAs

Several differences must be considered before a design is ready for implementation in HardCopy technology:

- HardCopy APEX devices are only customizable at the time they are manufactured. Make sure that the original APEX 20KE or APEX 20KC device has undergone thorough testing in the end-system before deciding to proceed with migration to a HardCopy APEX device, because no changes can be made to the HardCopy APEX device after it has been manufactured.
- ESBs that are configured as RAM or CAM will power-up un-initialized in the HardCopy APEX device. In the FPGA it is possible to configure, or “pre-load,” the ESB memory as part of the configuration sequence, then overwrite it when the device is in normal functional mode. This pre-loaded memory feature of the FPGA is not available in HardCopy devices. If a design contains RAM or CAM with assumed data values at power-up, then the HardCopy APEX device will not operate as expected. If a design uses this feature, it should be re-compiled without the memory pre-load. ESBs configured as ROM are fully supported.
- The JTAG boundary scan order in the HardCopy APEX device is different compared to the APEX 20K device. A HardCopy BSDL file that describes the re-ordered boundary scan chain should be used.



The BSDL files for HardCopy APEX devices are different from the corresponding APEX 20KE or APEX 20KC devices. Download the correct HardCopy BSDL file from Altera’s website at www.altera.com.

- The advanced 0.18- μ m aluminum metal process is used to support both APEX 20KE and APEX 20KC devices. The performance improvement achieved by the die size reduction and metal interconnect optimization more than offsets the need for copper in this case. Altera guarantees that a target HardCopy APEX device will provide the same or better performance as in the corresponding APEX 20KE or APEX 20KC device.

Power-up Mode and Configuration Emulation

Unlike their FPGA counterparts, HardCopy APEX devices do not need to be configured. However, to facilitate seamless migration, configuration can be emulated in these devices. There are three modes in which a

HardCopy APEX device can be prepared for operation after power up: instant on, instant on after 50 ms, and configuration emulation. Each mode is described below.

- In instant on mode, the HardCopy APEX device is available for use shortly after the device receives power. The on-chip power-on-reset (POR) circuit will set or reset all registers. The `CONF_DONE` output will be tri-stated once the power-on reset has elapsed. No configuration device or configuration input signals are necessary.
- In instant on after 50 ms mode, the HardCopy APEX device performs in a similar fashion to the Instant On mode, except that there is an additional delay of 50 ms (nominal), during which time the device is held in reset stage. The `CONF_DONE` output is pulled low during this time and then tri-stated after the 50 ms have elapsed. No configuration devices or configuration input signals are necessary for this option.
- In configuration emulation mode, the HardCopy APEX device undergoes an emulation of a full configuration sequence as if configured by an external processor or an EPC device. In this mode, the `CONF_DONE` signal is tri-stated after the correct number of clock cycles. This mode may be useful where there is some dependency on the configuration sequence (for example, multi-device configuration or processor initialization). In this mode, the device expects to see all configuration control and data input signals.

Speed Grades

Because HardCopy APEX devices are customized, no speed grading is performed. All HardCopy APEX devices will meet the timing requirements of the original FPGA of the fastest speed grade. Generally, HardCopy APEX devices will have a higher f_{MAX} than the corresponding FPGA, but the speed increase will vary on a design-by-design basis.

Quartus II-Generated Output Files

The HardCopy migration process requires several Quartus II software-generated files. These key output files are listed and explained below.

- The SRAM Object File (**.sof**) contains all of the necessary information needed to configure a FPGA
- The Compiler Report File (**.csf.rpt**) is parsed to extract useful information about the design
- The Verilog atom-based netlist file (**.vo**) is used to check the HardCopy netlist
- The pin out information file (**.pin**) contains user signal names and I/O configuration information

- The Delay Information File (.sdo) is used to check the original FPGA timing
- A completed HardCopy timing requirements file describes all necessary timing information on the design. A template of this text file is available for download from the Altera website at www.altera.com.

The migration process consists of several steps. First, a netlist is constructed from the SOF. Then, the netlist is checked to ensure that the built-in scan test structures will operate correctly. The netlist is then fed into a place-and-route engine, and the design interconnect is generated. Static timing analysis ensures that all timing constraints are met, and static functional verification techniques are employed to ensure correct device migration. After successfully completing these stages, physical verification of the device takes place, and the metal mask layers are taped out to fabricate HardCopy APEX devices.

Document Revision History

Table 8–3 shows the revision history for this chapter.

Table 8–3. Document Revision History		
Date and Document Version	Changes Made	Summary of Changes
September 2008, v2.3	Updated chapter number and metadata.	—
June 2007, v2.2	Minor text edits.	—
December 2006 v2.1	Updated revision history.	—
March 2006	Formerly chapter 10; no content change.	—
January 2005 v2.0	Update device names and other minor textual changes	—
June 2003 v1.0	Initial release of Chapter 10, <i>Description, Architecture and Features</i> , in the HardCopy Device Handbook	—

IEEE Std. 1149.1 (JTAG) Boundary-Scan Support

All HardCopy devices provide JTAG boundary-scan test (BST) circuitry that complies with the IEEE Std. 1149.1-1990 specification. HardCopy® APEX™ devices support the JTAG instructions shown in Table 9–1.



The BSDL files for HardCopy devices are different from the corresponding APEX 20KE or APEX 20KC parts. Download the correct HardCopy BSDL file from Altera's website at www.altera.com.

Table 9–1. HardCopy APEX JTAG Instructions

JTAG Instruction	Description
SAMPLE/PRELOAD	SAMPLE/PRELOAD allows a snapshot of signals at the device pins to be captured and examined during normal device operation and permits an initial data pattern to be output at the device pins. It is also used by the SignalTap® embedded logic analyzer.
EXTEST	Allows the external circuitry and board-level interconnections to be tested by forcing a test pattern at the output pins and capturing test results at the input pins.
BYPASS	Places the 1-bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through selected devices to adjacent devices during normal device operation.
USERCODE	Selects the 32-bit USERCODE register and places it between the TDI and TDO pins, allowing the USERCODE to be serially shifted out of TDO.
IDCODE	Selects the IDCODE register and places it between the TDI and TDO pins, allowing the IDCODE to be serially shifted out of TDO.

HardCopy APEX devices instruction register length is 10 bits; the USERCODE register length is 32 bits. Tables 9–2 and 9–3 show the boundary-scan register length and device IDCODE information for HardCopy devices.

Table 9–2. HardCopy APEX Boundary-Scan Register Length

Device	Boundary-Scan Register Length
HC20K400	1,506
HC20K600	1,806
HC20K1000	2,190
HC20K1500	2,502

Table 9–3. 32-Bit HardCopy APEX Device IDCODE

Device	IDCODE (32 Bits) <i>Note (1)</i>			
	Version (4 Bits)	Part Number (16 Bits)	Manufacturer Identity (11 Bits)	1 (1 Bit) <i>(2)</i>
HC20K400	0000	1000 0100 0000 0000	000 0110 1110	1
HC20K600	0000	1000 0110 0000 0000	000 0110 1110	1
HC20K1000	0000	1001 0000 0000 0000	000 0110 1110	1
HC20K1500	0000	1001 0101 0000 0000	000 0110 1110	1

Notes to Table 9–3:

- (1) The most significant bit (MSB) is on the left.
 (2) The IDCODE's least significant bit (LSB) is always 1.

Figure 9–1 shows the timing requirements for the JTAG signals.

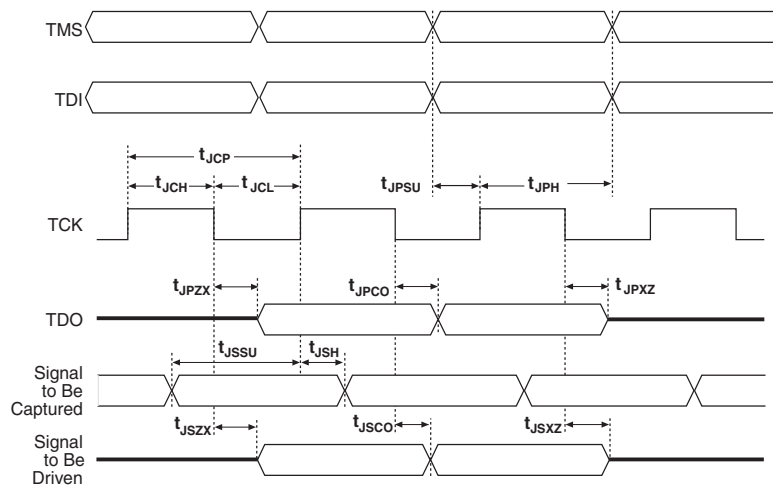
Figure 9–1. HardCopy JTAG Waveforms

Table 9–4 shows the JTAG timing parameters and values for HardCopy devices.

Table 9–4. HardCopy APEX JTAG Timing Parameters and Values				
Symbol	Parameter	Min	Max	Unit
t_{JCP}	TCK clock period	100		ns
t_{JCH}	TCK clock high time	50		ns
t_{JCL}	TCK clock low time	50		ns
t_{JPSU}	JTAG port setup time	20		ns
t_{JPH}	JTAG port hold time	45		ns
t_{JPCO}	JTAG port clock to output		25	ns
t_{JPZX}	JTAG port high impedance to valid output		25	ns
t_{JPXZ}	JTAG port valid output to high impedance		25	ns
t_{JSSU}	Capture register setup time	20		ns
t_{JSH}	Capture register hold time	45		ns
t_{JSCO}	Update register clock to output		35	ns
t_{JSZX}	Update register high impedance to valid output		35	ns
t_{JSXZ}	Update register valid output to high impedance		35	ns



For more information about using JTAG BST circuitry in Altera devices, refer to *Application Note 39 (IEEE Std. 1149.1 (JTAG) Boundary-Scan Testing in Altera Devices)*.

Document Revision History

Table 9–5 shows the revision history for this chapter.

Table 9–5. Document Revision History (Part 1 of 2)		
Date and Document Version	Changes Made	Summary of Changes
September 2008, v2.3	Updated chapter number and metadata.	—
June 2007, v2.2	Minor text edits.	—
December 2006 v2.1	Updated revision history.	Updated revision history.
March 2006	Formerly chapter 11; no content change.	

Table 9–5. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
January 2005 v2.0	Update device names and other minor textual changes.	—
June 2003 v1.0	Initial release of <i>Boundary-Scan Support</i> in the HardCopy Device Handbook.	—

Recommended Operating Conditions

Tables 10–1 through 10–4 provide information on absolute maximum ratings, recommended operating conditions, DC operating conditions, and capacitance for 1.8-V HardCopy® APEX™ devices.

Table 10–1. HardCopy APEX Device Absolute Maximum Ratings *Note (1)*

Symbol	Parameter	Conditions	Min	Max	Unit
V_{CCINT}	Supply voltage	With respect to ground (2)	–0.5	2.5	V
V_{CCIO}			–0.5	4.6	V
V_I			–0.5	4.6	V
I_{OUT}	DC output current, per pin		–25	25	mA
T_{STG}	Storage temperature	No bias	–65	150	°C
T_{AMB}	Ambient temperature	Under bias	–65	135	°C
T_J	Junction temperature	BGA packages, under bias		135	°C

Table 10–2. HardCopy APEX Device Recommended Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
V_{CCINT}	Supply voltage for internal logic and input buffers	(3), (4)	1.71 (1.71)	1.89 (1.89)	V
V_{CCIO}	Supply voltage for output buffers, 3.3-V operation	(3), (4)	3.00 (3.00)	3.60 (3.60)	V
	Supply voltage for output buffers, 2.5-V operation	(3), (4)	2.375 (2.375)	2.625 (2.625)	V
V_I	Input voltage	(2), (5)	–0.5	4.1	V
V_O	Output voltage		0	V_{CCIO}	V
T_J	Junction temperature	For commercial use	0	85	°C
		For industrial use	–40	100	°C
t_R	Input rise time (10% to 90%)			40	ns
t_F	Input fall time (90% to 10%)			40	ns

Table 10–3. HardCopy APEX Device DC Operating Conditions (Part 1 of 2) Notes (6), (7), (8)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{IH}	High-level LVTTTL, CMOS, or 3.3-V PCI input voltage		1.7, $0.5 \times V_{CCIO}$ (8)		4.1	V
V_{IL}	Low-level LVTTTL, CMOS, or 3.3-V PCI input voltage		–0.5		0.8, $0.3 \times V_{CCIO}$ (8)	V
V_{OH}	3.3-V high-level LVTTTL output voltage	$I_{OH} = -12$ mA DC, $V_{CCIO} = 3.00$ V (9)	2.4			V
	3.3-V high-level LVCMOS output voltage	$I_{OH} = -0.1$ mA DC, $V_{CCIO} = 3.00$ V (9)	$V_{CCIO} - 0.2$			V
	3.3-V high-level PCI output voltage	$I_{OH} = -0.5$ mA DC, $V_{CCIO} = 3.00$ to 3.60 V (9)	$0.9 \times V_{CCIO}$			V
	2.5-V high-level output voltage	$I_{OH} = -0.1$ mA DC, $V_{CCIO} = 2.30$ V (9)	2.1			V
		$I_{OH} = -1$ mA DC, $V_{CCIO} = 2.30$ V (9)	2.0			V
		$I_{OH} = -2$ mA DC, $V_{CCIO} = 2.30$ V (9)	1.7			V
V_{OL}	3.3-V low-level LVTTTL output voltage	$I_{OL} = 12$ mA DC, $V_{CCIO} = 3.00$ V (10)			0.4	V
	3.3-V low-level LVCMOS output voltage	$I_{OL} = 0.1$ mA DC, $V_{CCIO} = 3.00$ V (10)			0.2	V
	3.3-V low-level PCI output voltage	$I_{OL} = 1.5$ mA DC, $V_{CCIO} = 3.00$ to 3.60 V (10)			$0.1 \times V_{CCIO}$	V
	2.5-V low-level output voltage	$I_{OL} = 0.1$ mA DC, $V_{CCIO} = 2.30$ V (10)			0.2	V
		$I_{OL} = 1$ mA DC, $V_{CCIO} = 2.30$ V (10)			0.4	V
		$I_{OL} = 2$ mA DC, $V_{CCIO} = 2.30$ V (10)			0.7	V
I_I	Input pin leakage current (11)	$V_I = 4.1$ to -0.5 V	–10		10	μ A
I_{OZ}	Tri-stated I/O pin leakage current (11)	$V_O = 4.1$ to -0.5 V	–10		10	μ A
I_{CC0}	V_{CC} supply current (standby) (All ESBs in power-down mode)	$V_I =$ ground, no load, no toggling inputs, -7 speed grade		10		mA
		$V_I =$ ground, no load, no toggling inputs, -8, -9 speed grades		5		mA

Table 10–3. HardCopy APEX Device DC Operating Conditions (Part 2 of 2) Notes (6), (7), (8)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
R_{CONF}	Value of I/O pin pull-up resistor before and during configuration emulation	$V_{CCIO} = 3.0 \text{ V}$ (12)	20		50	k Ω
		$V_{CCIO} = 2.375 \text{ V}$ (12)	30		80	k Ω
		$V_{CCIO} = 1.71 \text{ V}$ (12)	60		150	k Ω

Table 10–4. HardCopy APEX Device Capacitance Note (13)

Symbol	Parameter	Conditions	Min	Typ	Max
C_{IN}	Input capacitance	$V_{IN} = 0 \text{ V}$, $f = 1.0 \text{ MHz}$		8	pF
C_{INCLK}	Input capacitance on dedicated clock pin	$V_{IN} = 0 \text{ V}$, $f = 1.0 \text{ MHz}$		12	pF
C_{OUT}	Output capacitance	$V_{OUT} = 0 \text{ V}$, $f = 1.0 \text{ MHz}$		8	pF

Notes to Table 10–1 through 10–4:

- (1) Refer to the *Operating Requirements for Altera Devices Data Sheet*.
- (2) Minimum DC input is -0.5 V . During transitions, the inputs may undershoot to -0.5 V or overshoot to 4.6 V for input currents less than 100 mA and periods shorter than 20 ns .
- (3) Numbers in parentheses are for industrial-temperature-range devices.
- (4) Maximum V_{CC} rise time is 100 ms , and V_{CC} must rise monotonically.
- (5) All pins (including dedicated inputs, clock, I/O, and JTAG pins) may be driven before V_{CCINT} and V_{CCIO} are powered.
- (6) Typical values are for $T_A = 25^\circ \text{C}$, $V_{CCINT} = 1.8 \text{ V}$, and $V_{CCIO} = 1.8 \text{ V}$, 2.5 V , or 3.3 V .
- (7) These values are specified under the HardCopy device recommended operating conditions, as shown in [Table 10–2 on page 10–1](#).
- (8) Refer to *AN 117: Using Selectable I/O Standards in Altera Devices* for the V_{IH} , V_{IL} , V_{OH} , V_{OL} , and I_I parameters when $V_{CCIO} = 1.8 \text{ V}$.
- (9) The APEX 20KE input buffers are compatible with 1.8-V , 2.5-V and 3.3-V (LVTTTL and LVC MOS) signals. Additionally, the input buffers are 3.3-V PCI compliant. Input buffers also meet specifications for GTL+, CTT, AGP, SSTL-2, SSTL-3, and HSTL.
- (10) The I_{OH} parameter refers to high-level TTL, PCI, or CMOS output current.
- (11) This value is specified for normal device operation. The value may vary during power-up.
- (12) Pin pull-up resistance values will be lower if an external source drives the pin higher than V_{CCIO} .
- (13) Capacitance is sample-tested only.

Tables 10–5 through 10–20 list the DC operating specifications for the supported I/O standards. These tables list minimal specifications only; HardCopy devices may exceed these specifications.

Table 10–5. LVTTTL I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Output supply voltage		3.0	3.6	V
V_{IH}	High-level input voltage		2.0	$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		–0.3	0.8	V
I_I	Input pin leakage current	$V_{IN} = 0\text{ V or }3.3\text{ V}$	–10	10	μA
V_{OH}	High-level output voltage	$I_{OH} = -12\text{ mA}$, $V_{CCIO} = 3.0\text{ V}$ (1)	2.4		V
V_{OL}	Low-level output voltage	$I_{OL} = 12\text{ mA}$, $V_{CCIO} = 3.0\text{ V}$ (2)		0.4	V

Table 10–6. LVCMOS I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Power supply voltage range		3.0	3.6	V
V_{IH}	High-level input voltage		2.0	$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		–0.3	0.8	V
I_I	Input pin leakage current	$V_{IN} = 0\text{ V or }3.3\text{ V}$	–10	10	μA
V_{OH}	High-level output voltage	$V_{CCIO} = 3.0\text{ V}$ $I_{OH} = -0.1\text{ mA}$ (1)	$V_{CCIO} - 0.2$		V
V_{OL}	Low-level output voltage	$V_{CCIO} = 3.0\text{ V}$ $I_{OL} = 0.1\text{ mA}$ (2)		0.2	V

Table 10–7. 2.5-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Output supply voltage		2.375	2.625	V
V_{IH}	High-level input voltage		1.7	$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		–0.3	0.7	V
I_I	Input pin leakage current	$V_{IN} = 0\text{ V or }3.3\text{ V}$	–10	10	μA
V_{OH}	High-level output voltage	$I_{OH} = -0.1\text{ mA (1)}$	2.1		V
		$I_{OH} = -1\text{ mA (1)}$	2.0		V
		$I_{OH} = -2\text{ mA (1)}$	1.7		V
V_{OL}	Low-level output voltage	$I_{OL} = 0.1\text{ mA (2)}$		0.2	V
		$I_{OL} = 1\text{ mA (2)}$		0.4	V
		$I_{OL} = 2\text{ mA (2)}$		0.7	V

Table 10–8. 1.8-V I/O Specifications

Symbol	Parameter	Conditions	Minimum	Maximum	Units
V_{CCIO}	Output supply voltage		1.7	1.9	V
V_{IH}	High-level input voltage		$0.65 \times V_{CCIO}$	$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage			$0.35 \times V_{CCIO}$	V
I_I	Input pin leakage current	$V_{IN} = 0\text{ V or }3.3\text{ V}$	–10	10	μA
V_{OH}	High-level output voltage	$I_{OH} = -2\text{ mA (1)}$	$V_{CCIO} - 0.45$		V
V_{OL}	Low-level output voltage	$I_{OL} = 2\text{ mA (2)}$		0.45	V

Table 10–9. 3.3-V PCI Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		3.0	3.3	3.6	V
V_{IH}	High-level input voltage		$0.5 \times V_{CCIO}$		$V_{CCIO} + 0.5$	V

Table 10–9. 3.3-V PCI Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{IL}	Low-level input voltage		–0.5		$0.3 \times V_{CCIO}$	V
I_I	Input pin leakage current	$0 < V_{IN} < V_{CCIO}$	–10		10	μA
V_{OH}	High-level output voltage	$I_{OUT} = -500 \mu A$	$0.9 \times V_{CCIO}$			V
V_{OL}	Low-level output voltage	$I_{OUT} = 1,500 \mu A$			$0.1 \times V_{CCIO}$	V

Table 10–10. 3.3-V PCI-X Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	Output supply voltage		3.0	3.3	3.6	V
V_{IH}	High-level input voltage		$0.5 \times V_{CCIO}$		$V_{CCIO} + 0.5$	V
V_{IL}	Low-level input voltage		–0.5		$0.35 \times V_{CCIO}$	V
V_{IPU}	Input pull-up voltage		$0.7 \times V_{CCIO}$			V
I_{IL}	Input pin leakage current	$0 < V_{IN} < V_{CCIO}$	–10.0		10.0	μA
V_{OH}	High-level output voltage	$I_{OUT} = -500 \mu A$	$0.9 \times V_{CCIO}$			V
V_{OL}	Low-level output voltage	$I_{OUT} = 1500 \mu A$			$0.1 \times V_{CCIO}$	V
L_{PIN}	Pin Inductance				15.0	nH

Table 10–11. 3.3-V LVDS I/O Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		3.135	3.3	3.465	V
V_{OD}	Differential output voltage	$R_L = 100 \Omega$	250		450	mV
V_{OD}	Change in VOD between high and low	$R_L = 100 \Omega$			50	mV
V_{OS}	Output offset voltage	$R_L = 100 \Omega$	1.125	1.25	1.375	V

Table 10–11. 3.3-V LVDS I/O Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{OS}	Change in VOS between high and low	$R_L = 100\ \Omega$			50	mV
V_{TH}	Differential input threshold	$V_{CM} = 1.2\ V$	–100		100	mV
V_{IN}	Receiver input voltage range		0.0		2.4	V
R_L	Receiver differential input resistor (external to APEX 20K devices)		90	100	110	Ω

Table 10–12. GTL+ I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{TT}	Termination voltage		1.35	1.5	1.65	V
V_{REF}	Reference voltage		0.88	1.0	1.12	V
V_{IH}	High-level input voltage		$V_{REF} + 0.1$			V
V_{IL}	Low-level input voltage				$V_{REF} - 0.1$	V
V_{OL}	Low-level output voltage	$I_{OL} = 36\ \text{mA}$ (2)			0.65	V

Table 10–13. SSTL-2 Class I Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		2.375	2.5	2.625	V
V_{TT}	Termination voltage		$V_{REF} - 0.04$	V_{REF}	$V_{REF} + 0.04$	V
V_{REF}	Reference voltage		1.15	1.25	1.35	V
V_{IH}	High-level input voltage		$V_{REF} + 0.18$		$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		–0.3		$V_{REF} - 0.18$	V

Table 10–13. SSTL-2 Class I Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{OH}	High-level output voltage	$I_{OH} = -7.6 \text{ mA}$ (1)	$V_{TT} + 0.57$			V
V_{OL}	Low-level output voltage	$I_{OL} = 7.6 \text{ mA}$ (2)			$V_{TT} - 0.57$	V

Table 10–14. SSTL-2 Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		2.375	2.5	2.625	V
V_{TT}	Termination voltage		$V_{REF} - 0.04$	V_{REF}	$V_{REF} + 0.04$	V
V_{REF}	Reference voltage		1.15	1.25	1.35	V
V_{IH}	High-level input voltage		$V_{REF} + 0.18$		$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		-0.3		$V_{REF} - 0.18$	V
V_{OH}	High-level output voltage	$I_{OH} = -15.2 \text{ mA}$ (1)	$V_{TT} + 0.76$			V
V_{OL}	Low-level output voltage	$I_{OL} = 15.2 \text{ mA}$ (2)			$V_{TT} - 0.76$	V

Table 10–15. SSTL-3 Class I Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		3.0	3.3	3.6	V
V_{TT}	Termination voltage		$V_{REF} - 0.05$	V_{REF}	$V_{REF} + 0.05$	V
V_{REF}	Reference voltage		1.3	1.5	1.7	V
V_{IH}	High-level input voltage		$V_{REF} + 0.2$		$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		-0.3		$V_{REF} - 0.2$	V
V_{OH}	High-level output voltage	$I_{OH} = -8 \text{ mA}$ (1)	$V_{TT} + 0.6$			V
V_{OL}	Low-level output voltage	$I_{OL} = 8 \text{ mA}$ (2)			$V_{TT} - 0.6$	V

Table 10–16. SSTL-3 Class II Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		3.0	3.3	3.6	V
V_{TT}	Termination voltage		$V_{REF} - 0.05$	V_{REF}	$V_{REF} + 0.05$	V
V_{REF}	Reference voltage		1.3	1.5	1.7	V
V_{IH}	High-level input voltage		$V_{REF} + 0.2$		$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		-0.3		$V_{REF} - 0.2$	V
V_{OH}	High-level output voltage	$I_{OH} = -16 \text{ mA}$ (1)	$V_{TT} + 0.8$			V
V_{OL}	Low-level output voltage	$I_{OL} = 16 \text{ mA}$ (2)			$V_{TT} - 0.8$	V

Table 10–17. HSTL Class I I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		1.71	1.8	1.89	V
V_{TT}	Termination voltage		$V_{REF} - 0.05$	V_{REF}	$V_{REF} + 0.05$	V
V_{REF}	Reference voltage		0.68	0.75	0.90	V
V_{IH}	High-level input voltage		$V_{REF} + 0.1$		$V_{CCIO} + 0.3$	V
V_{IL}	Low-level input voltage		-0.3		$V_{REF} - 0.1$	V
V_{OH}	High-level output voltage	$I_{OH} = -8 \text{ mA}$ (1)	$V_{CCIO} - 0.4$			V
V_{OL}	Low-level output voltage	$I_{OL} = 8 \text{ mA}$ (2)			0.4	V

Table 10–18. LVPECL Specifications (Part 1 of 2)

Symbol	Parameter	Minimum	Typical	Maximum	Units
V_{CCIO}	Output Supply Voltage	3.135	3.3	3.465	V
V_{IH}	Low-level input voltage	1,300		1,700	mV
V_{IL}	High-level input voltage	2,100		2,600	mV
V_{OH}	Low-level output voltage	1,450		1,650	mV

Table 10–18. LVPECL Specifications (Part 2 of 2)

Symbol	Parameter	Minimum	Typical	Maximum	Units
V_{OL}	High-level output voltage	2,275		2,420	mV
V_{ID}	Input voltage differential	400	600	950	mV
V_{OD}	Output voltage differential	625	800	950	mV
t_r, t_f	Rise and fall time (20 to 80%)	85		325	ps
t_{DSKEW}	Differential skew			25	ps
t_O	Output load		150		Ω
R_L	Receiver differential input resistor		100		Ω

Table 10–19. 3.3-V AGP I/O Specifications

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		3.15	3.3	3.45	V
V_{REF}	Reference voltage		$0.39 \times V_{CCIO}$		$0.41 \times V_{CCIO}$	V
V_{IH}	High-level input voltage		$0.5 \times V_{CCIO}$		$V_{CCIO} + 0.5$	V
V_{IL}	Low-level input voltage				$0.3 \times V_{CCIO}$	V
V_{OH}	High-level output voltage	$I_{OUT} = -500 \mu A$	$0.9 \times V_{CCIO}$		3.6	V
V_{OL}	Low-level output voltage	$I_{OUT} = 1500 \mu A$			$0.1 \times V_{CCIO}$	V
I_I	Input pin leakage current	$0 < V_{IN} < V_{CCIO}$	-10		10	μA

Table 10–20. CTT I/O Specifications (Part 1 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{CCIO}	I/O supply voltage		3.0	3.3	3.6	V
V_{TT}/V_{REF} (3)	Termination and reference voltage		1.35	1.5	1.65	V
V_{IH}	High-level input voltage		$V_{REF} + 0.2$			V

Table 10–20. CTT I/O Specifications (Part 2 of 2)

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Units
V_{IL}	Low-level input voltage				$V_{REF} - 0.2$	V
I_I	Input pin leakage current	$0 < V_{IN} < V_{CCIO}$	–10		10	μA
V_{OH}	High-level output voltage	$I_{OH} = -8 \text{ mA}$ (1)	$V_{REF} + 0.4$			V
V_{OL}	Low-level output voltage	$I_{OL} = 8 \text{ mA}$ (2)			$V_{REF} - 0.4$	V
I_O	Output leakage current (when output is high Z)	$GND \leq V_{OUT} \leq V_{CCIO}$	–10		10	μA

Notes to Tables 10–5 through 10–20:

- (1) The I_{OH} parameter refers to high-level output current.
- (2) The I_{OL} parameter refers to low-level output current. This parameter applies to open-drain pins as well as output pins.
- (3) V_{REF} specifies center point of switching range.

Figure 10–1 shows the output drive characteristics of HardCopy APEX devices.

Figure 10–1. Output Drive Characteristics of HardCopy APEX Devices

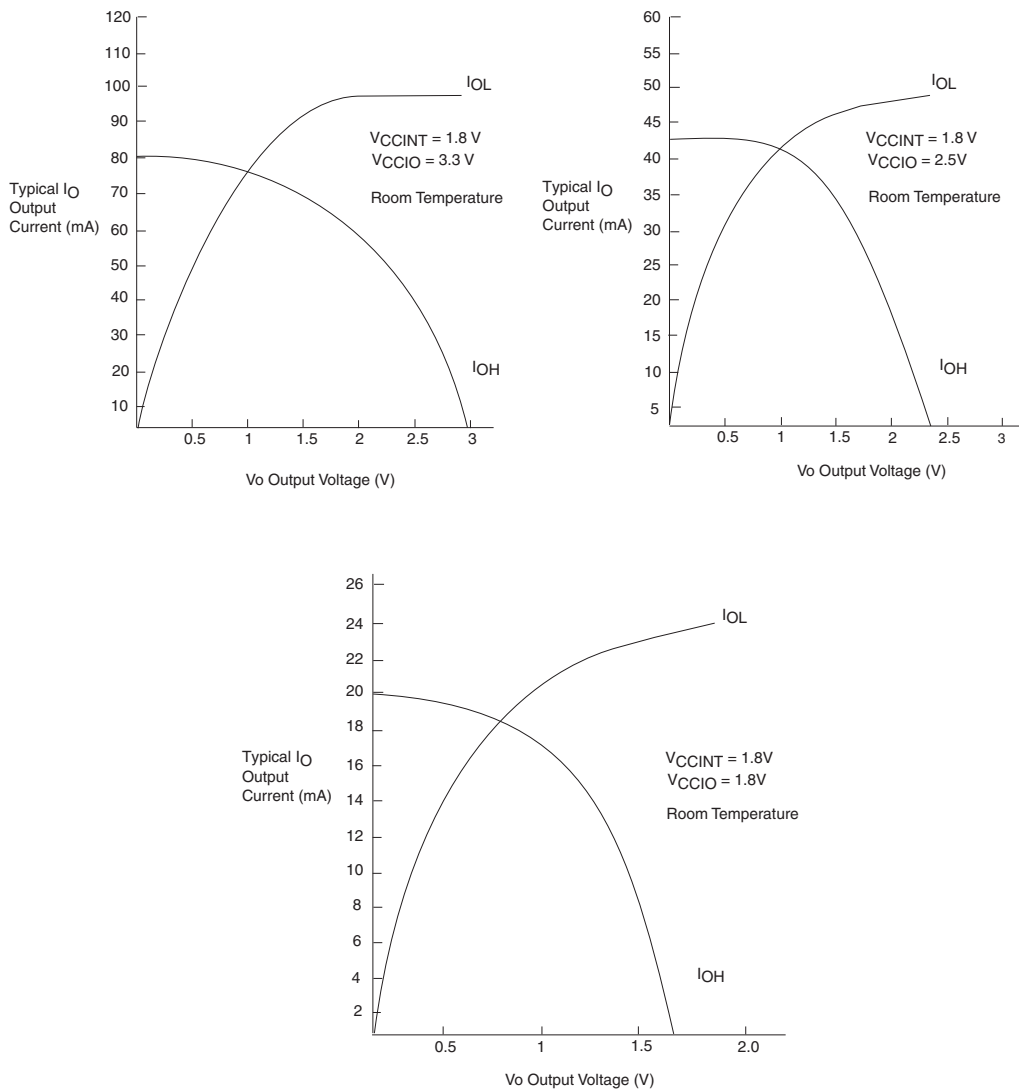
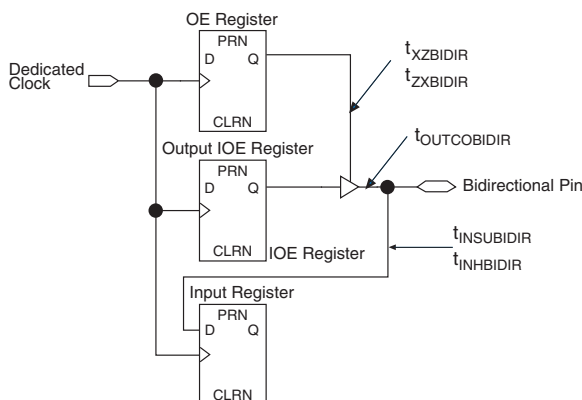


Figure 10–2 shows the timing model for bidirectional I/O pin timing.

Figure 10–2. Synchronous Bidirectional Pin External Timing



Tables 10–21 and 10–22 describe HardCopy APEX device external timing parameters.

Table 10–21. HardCopy APEX Device External Timing Parameters *Note (1)*

Symbol	Clock Parameter	Conditions
t_{INSU}	Setup time with global clock at IOE register	
t_{INH}	Hold time with global clock at IOE register	
t_{OUTCO}	Clock-to-output delay with global clock at IOE output register	C1 = 35 pF
$t_{INSUPLL}$	Setup time with PLL clock at IOE input register	
t_{INHPLL}	Hold time with PLL clock at IOE input register	
$t_{OUTCOPLL}$	Clock-to-output delay with PLL clock at IOE output register	C1 = 35 pF

Table 10–22. HardCopy APEX Device External Bidirectional Timing Parameters (Part 1 of 2) *Note (1)*

Symbol	Parameter	Condition
$t_{INSUBIDIR}$	Setup time for bidirectional pins with global clock at LAB-adjacent input register	
$t_{INHBIDIR}$	Hold time for bidirectional pins with global clock at LAB-adjacent input register	
$t_{OUTCOBIDIR}$	Clock-to-output delay for bidirectional pins with global clock at IOE register	C1 = 35 pF
$t_{XZBIDIR}$	Synchronous output enable register to output buffer disable delay	C1 = 35 pF

Table 10–22. HardCopy APEX Device External Bidirectional Timing Parameters (Part 2 of 2) *Note (1)*

Symbol	Parameter	Condition
t_{ZXBIDIR}	Synchronous output enable register to output buffer enable delay	C1 = 35 pF
$t_{\text{INSUBIDIRPLL}}$	Setup time for bidirectional pins with PLL clock at LAB-adjacent input register	
$t_{\text{INHIDIRPLL}}$	Hold time for bidirectional pins with PLL clock at LAB-adjacent input register	
$t_{\text{OUTCOBIDIRPLL}}$	Clock-to-output delay for bidirectional pins with PLL clock at IOE register	C1 = 35 pF
$t_{\text{XZBIDIRPLL}}$	Synchronous output enable register to output buffer disable delay with PLL	C1 = 35 pF
$t_{\text{ZXBIDIRPLL}}$	Synchronous output enable register to output buffer enable delay with PLL	C1 = 35 pF

Note to Tables 10–21 and 10–22:

(1) These timing parameters are sample-tested only.

Tables 10–23 and 10–24 show the external timing parameters for HC20K1500 devices.

Table 10–23. HC20K1500 External Timing Parameters *Note (1)*

Symbol	Min	Max	Unit
t_{INSU}	2.0		ns
t_{INH}	0.0		ns
t_{OUTCO}	2.0	5.0	ns
t_{INSUPLL}	3.3		ns
t_{INHPLL}	0.0		ns
t_{OUTCOPLL}	0.5	2.1	ns

Table 10–24. HC20K1500 External Bidirectional Timing Parameters (Part 1 of 2) *Note (1)*

Symbol	Min	Max	Unit
$t_{\text{INSUBIDIR}}$	1.9		ns
t_{INHIDIR}	0.0		ns
$t_{\text{OUTCOBIDIR}}$	2.0	5.0	ns
t_{XZBIDIR}		7.1	ns
t_{ZXBIDIR}		7.1	ns
$t_{\text{INSUBIDIRPLL}}$	3.9		ns

Table 10–24. HC20K1500 External Bidirectional Timing Parameters
(Part 2 of 2) *Note (1)*

Symbol	Min	Max	Unit
$t_{INHDIRPLL}$	0.0		ns
$t_{OUTCOBIDIRPLL}$	0.5	2.1	ns
$t_{XZBIDIRPLL}$		4.2	ns
$t_{ZXBIDIRPLL}$		4.2	ns

Note to Tables 10–23 and 10–24:

- (1) Timing information is preliminary. Final timing information will be available in a future version of this data sheet.

Document Revision History

Table 10–25 shows the revision history for this chapter.

Table 10–25. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
September 2008, v2.3	Updated chapter number and metadata.	—
June 2007, v2.2	Minor text edits.	—
December 2006 v2.1	Updated revision history.	—
March 2006	Formerly chapter 12; no content change.	—
January 2005 v2.0	Update device names and other minor textual changes.	—
June 2003 v1.0	Initial release of <i>Operating Conditions</i> , in the HardCopy Device Handbook	—



Section III. General HardCopy Series Design Considerations

This section provides information on hardware design considerations for HardCopy® series devices.

This section contains the following:

- [Chapter 11, Design Guidelines for HardCopy Series Devices](#)
- [Chapter 12, Power-Up Modes and Configuration Emulation in HardCopy Series Devices](#)

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the complete handbook.

Introduction

HardCopy® series devices provide dramatic cost savings, performance improvement, and reduced power consumption over their programmable counterparts. In order to ensure the smoothest possible transfer from the FPGA device to the equivalent HardCopy series device, you must meet certain design rules while the FPGA implementation is still in progress. A design that meets standard, accepted coding styles for FPGAs, adheres easier to recommended guidelines. This chapter describes some common situations that you should avoid. It also provides alternatives on how to design in these situations.

Design Assistant Tool

The Design Assistant tool in the Quartus® II software allows you to check for any potential design problems early in the design process. The Design Assistant is a design-rule checking tool that checks the compiled design for adherence to Altera® recommended design guidelines. It provides a summary of the violated rules that exist in a design together with explicit details of each violation instance. You can customize the set of rules that the tool checks to allow some rule violations in your design. This is useful if it is known that the design violates a particular rule that is not critical. However, for HardCopy design, you must enable all of the Design Assistant rules. All Design Assistant rules are enabled and run by default in the Quartus II software when using the HardCopy Timing Optimization Wizard in the **HardCopy Utilities** (Project menu). The HardCopy Advisor in the Quartus II software also checks to see if the Design Assistant is enabled.

The Design Assistant classifies messages using the four severity levels described in [Table 11-1](#).

Table 11-1. Design Assistant Message Severity Levels (Part 1 of 2)

Severity Level	Description
Critical	The rule violation described in the message critically affects the reliability of the design. Altera cannot migrate the design successfully to a HardCopy device without closely reviewing these violations.
High	The rule violation described in the message affects the reliability of the design. Altera must review the violation before the design is migrated to a HardCopy device.

Table 11–1. Design Assistant Message Severity Levels (Part 2 of 2)

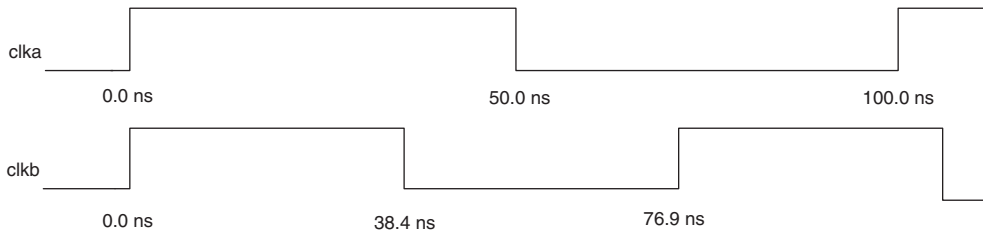
Severity Level	Description
Medium	The rule violation described in the message may result in implementation complexity. The violation may impact the schedule or effort required to migrate the design to a HardCopy series device.
Information only	The message contains information regarding a design rule.

A design that adheres to Altera recommended design guidelines does not produce any critical, high, or medium level Design Assistant messages. If the Design Assistant generates these kinds of messages, Altera's HardCopy Design Center (which performs the migration) carefully reviews each message before considering implementing the FPGA design into a HardCopy design. After reviewing these messages with your design team, Altera may be able to implement the design in a HardCopy device. Informational messages are primarily for the benefit of the Altera HardCopy Design Center and are used to gather information about your design for the migration process from FPGA prototype to HardCopy production device.

Asynchronous Clock Domains

A design contains several clock sources, each driving a subsection of the design. A design subsection, driven by a single clock source is called a clock domain. The frequency and phase of each clock source can be different from the rest.

The timing diagram in [Figure 11–1](#) shows two free-running clocks used to describe the nature of asynchronous clock domains. If the two clock signals do not have a synchronous, or fixed, relationship, they are asynchronous to each other. An example of asynchronous signals are two clock signals running at frequencies that have no obvious harmonic relationship.

Figure 11–1. Two Asynchronous Clock Signals Notes (1), (2)**Notes to Figure 11–1:**

- (1) `clka` = 10 MHz; `clkb` = 13 MHz.
- (2) Both clocks have 50% duty cycles.

In Figure 11–1, the `clka` signal is defined with a rising edge at 0.0 ns, a falling edge at 50 ns ($1/10 \text{ MHz} = 100 \text{ ns}$). Subsequent rising edges of `clka` are at 200 ns, 300 ns, 400 ns, and so on.

The `clkb` signal is defined with a rising edge at 0.0 ns, a falling edge at 38.45 ns, and the next rising edge at 76.9 ns. The subsequent rising edges of `clkb` are at 153.8 ns, 230.7 ns, 307.6 ns, 384.5 ns, and so on.

Not until the thousandth clock edge of `clkb` ($1000 \times 76.9 = 76,900 \text{ ns}$) or the 7,690th clock edge of `clka` ($7,690 \times 100 = 769,000 \text{ ns}$), does `clka` and `clkb` have coincident edges. It is very unlikely that these two clocks are intended to synchronize with each other every 76,900 ns, so these two clock domains are considered asynchronous to each other.

A more subtle case of asynchronous clock domains occurs when two clock domains have a very obvious frequency and phase relationship, especially when one is a multiple of the other. Consider a system with clocks running at 100 MHz and 50 MHz. The edges of one of these clocks are always a fixed distance away, in time, from the edges of the other clock. In this case, the clock domains may or may not be asynchronous, depending on what your original intention was regarding the interactions of these two clock domains.

Similarly, two clocks running at the same nominal frequency may be asynchronous to each other if there is no synchronization mechanism between them. For example, two crystal oscillators, each running at 100 MHz on a PC board, have some frequency variations due to temperature fluctuations, and this may be different for each oscillator. This results in the two independent clock signals drifting in and out of phase with each other.

Transferring Data between Two Asynchronous Clock Domains

If two asynchronous clock domains need to communicate with each other, you need to consider how to reliably perform this operation. The following three examples show how to transfer data between two asynchronous clock domains.:

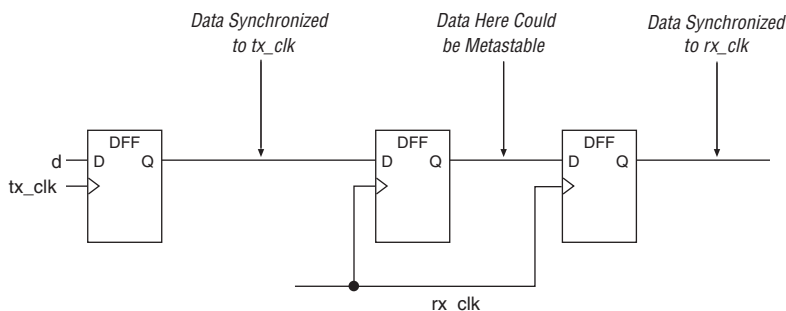
- Using a double synchronizer
- Using a first-in first-out (FIFO) buffer
- Using a handshake protocol

The choice of which to use depends on the particular application, the number of asynchronous signals crossing clock boundaries, and the resources available to perform the cross-domain transfers.

Using a Double Synchronizer for Single-Bit Data Transfer

Figure 11–2 shows a double synchronizer for single-bit data transfer consisting of a 2-bit shift register structure clocked by the receiving clock. The second stage of the shift register reduces the probability of metastability (unknown state) on the data output from the first register propagating through to the output of the second register. The data from the transmitting clock domain should come directly from a register. This technique is recommended only if single-data signals (for example, non-data buses) need to be transferred across clock domains. This is because it is possible that some bits of a data bus are captured in one clock cycle while other bits get captured in the next. More than two stages of the synchronizer circuit can be used at the expense of increased latency. The benefit of more stages is that the mean time between failures (MTBF) is increased with each additional stage.

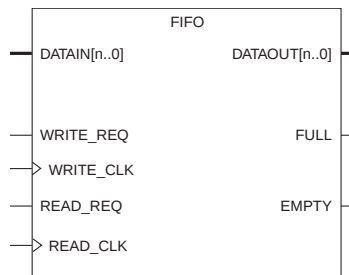
Figure 11–2. A Double Synchronizer Circuit



Using a FIFO Buffer

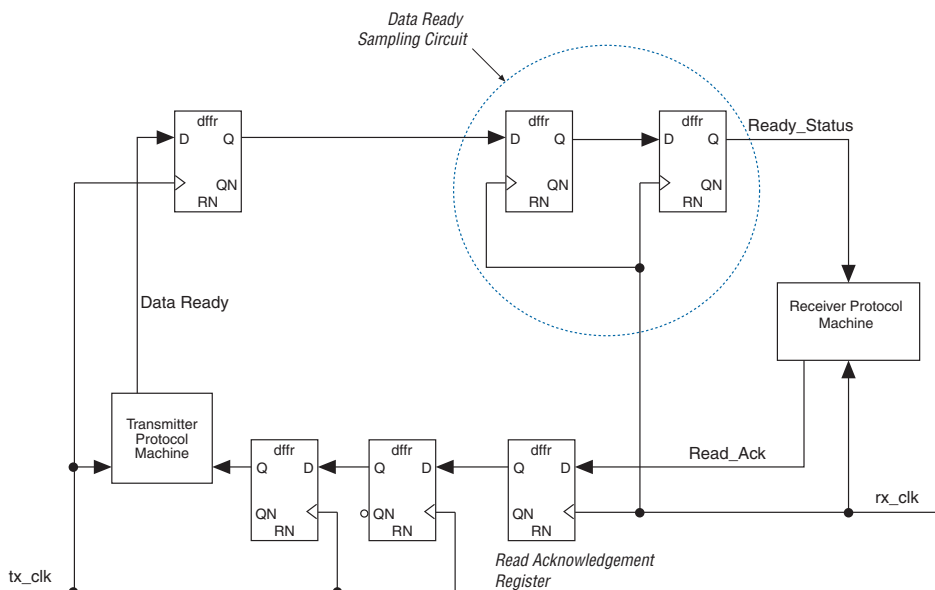
The advantage of using a FIFO buffer, shown in [Figure 11–3](#), is that Altera’s MegaWizard® Plug-In Manager makes it very easy to design a FIFO buffer. A FIFO buffer is useful when you need to transfer a data bus signal across an asynchronous clock domain, and it is beneficial to temporary storage of this data. A FIFO buffer circuit should not generate any Design Assistant warnings unless an asynchronous clear is used in the circuit. An asynchronous clear in the FIFO buffer circuit results in a warning stating that a reset signal generated in one clock domain is not being synchronized before being used in another clock domain. This occurs because a dual-clock FIFO megafunction only has one `acclr` pin to reset the entire FIFO buffer circuit. You cannot remove this warning in the case of a dual-clock FIFO buffer circuit. As a safeguard, Altera recommends using a reset signal that is synchronous to the clock domain of the write side of the FIFO buffer circuit.

Figure 11–3. A FIFO Buffer



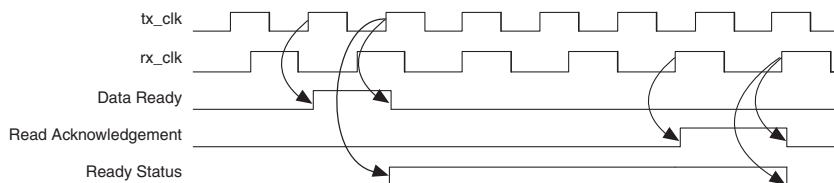
Using a Handshake Protocol

A handshake protocol circuit uses a small quantity of logic cells to implement and guarantee that all bits of a data bus crossing asynchronous clock domains are registered by the same clock edge in the receiving clock domain. This circuit, shown in [Figure 11–4](#), is best used in cases where there is no memory available to be used as FIFO buffers, and the design has many data buses to transfer between clock domains.

Figure 11–4. A Handshake Protocol Circuit

This circuit is initiated by a data ready signal going high in the transmitting clock domain tx_clk . This is clocked into the data ready sampling registers and causes the **Ready_Status** signal to go high. The **Data Ready** signal must be long enough in duration so that it is successfully sampled in the receiver domain. This is important if the rx_clk signal is slower than tx_clk .

At this point, the receiving clock domain rx_clk can read the data from the transmitting clock domain tx_clk . After this read operation has finished, the receiving clock domain (rx_clk) generates a synchronous **Read_Ack** signal, which gets registered by the read acknowledge register. This registered signal is sampled by the **Read_Ack** sampling circuit in the transmitter domain. The **Read_Ack** signal must be long enough in duration so that it is successfully sampled in the transmitter domain. This is important if the transmitter clock is slower than the receiver clock. After this event, the data transfer between the two asynchronous domains is complete, as shown by the timing diagram in [Figure 11–5](#).

Figure 11–5. Data Transfer Between Two Asynchronous Clock Domains

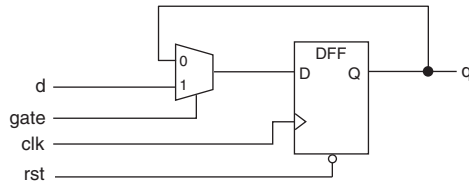
Gated Clocks

Clock gating is sometimes used to “turn off” parts of a circuit to reduce the total power consumption of a device. The gated clock signal prevents any of the logic driven by it from switching so the logic does not consume any power. This works best if the gating is done at the root of the clock tree. If the clock is gated at the leaf-cell level (for example, immediately before the input to the register), the device does not save much power because the whole clock network still toggles. The disadvantage in using this type of circuit is that it can lead to unexpected glitches on the resultant gated clock signal if certain rules are not adhered to. Rules are provided in the following subsections:

- Preferred Clock Gating Circuit
- Alternative Clock Gating Circuits
- Inverted Clocks
- Clocks Driving Non-Clock Pins
- Clock Signals Should Use Dedicated Clock Resources
- Mixing Clock Edges

Preferred Clock Gating Circuit

The preferred way to gate a clock signal is to use a purely synchronous circuit, as shown in [Figure 11–6](#). In this implementation, the clock is not gated at all. Rather, the data signal into a register is gated. This circuit is sometimes represented as a register with a clock enable (CE) pin. This circuit is not sensitive to any glitches on the gate signal, so it gets generated directly from a register or any complex combinational function. The constraints on the gate or clock enable signal are exactly the same as those on the ‘d’ input of the gating multiplexer. Both of these signals must meet the setup and hold times of the register that they feed into.

Figure 11–6. Preferred Clock-Gating Circuit

This circuit only takes a few lines of VHDL or Verilog hardware description language (HDL) to describe.

The following is a VHDL code fragment for a synchronous clock gating circuit.

```

architecture rtl of vhdl_enable is
begin
  process (rst, clk)
  begin
    if (rst = '0') then
      q <= '0';
    elsif clk'event and clk = '1' then
      if (gate = '1') then
        q <= d;
      end if;
    end if;
  end process;
end rtl;

```

The following is a Verilog HDL code fragment for a synchronous clock gating circuit.

```

always @ (posedge clk or negedge rst)
begin
  if (!rst)
    q <= 1'b0;
  else if (gate)
    q <= d;
  else
    q <= q;
end

```

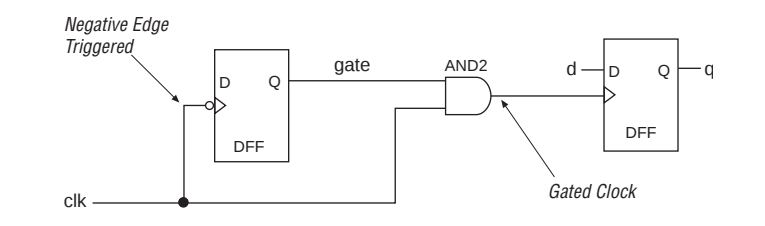
Alternative Clock Gating Circuits

If a clock gating circuit is absolutely necessary in the design, one of the following two circuits may also be used. The Design Assistant does not flag a violation for these circuits.

Clock Gating Circuit Using an AND Gate

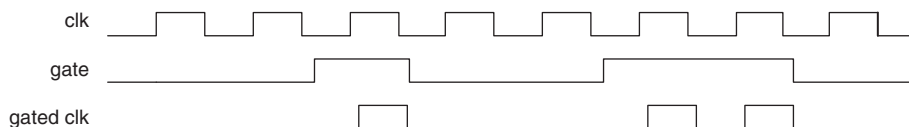
Designs can use a two-input AND gate for a gated clock signal that feeds into positive-edge-triggered registers. One input to the AND gate is the original clock signal. The other input to the AND gate is the gating signal, which should be driven directly from a register clocked by the negative edge of the same original clock signal. Figure 11–7 shows this type of circuit.

Figure 11–7. Clock Gating Circuit Using an AND Gate



Because the register that generates the gate signal is triggered off of the negative edge of the same clock, the effect of using both edges of the same clock in the design should be considered. The timing diagram in Figure 11–8 shows the operation of this circuit. The *gate* signal occurs after the negative edge of the clock and comes directly from a register. The logical AND of this *gate* signal, with the original un-inverted clock, generates a clean clock signal.

Figure 11–8. Timing Diagram for Clock Gating Circuit Using an AND Gate

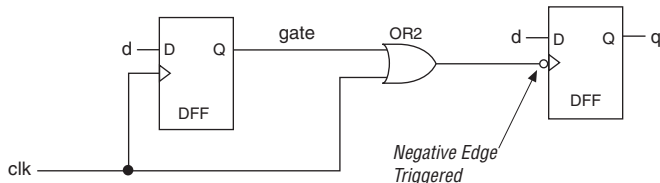


If the delay between the register that generates the *gate* signal and the *gate* input to the AND gate is greater than the low period of the clock, (one half of the clock period for a 50% duty cycle clock), the clock pulse width is narrowed.

Clock Gating Circuit Using an OR Gate

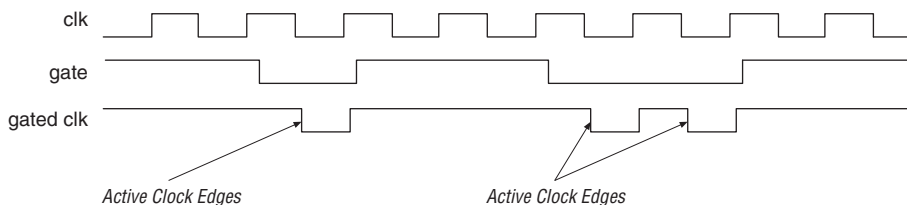
Use a two-input OR gate for a gated clock signal that feeds into a negative-edge-triggered register. One input to the OR gate is the original clock signal. The other input to the OR gate is the gating signal, which should be driven directly from a register clocked by the positive edge of the same original clock signal. Figure 11–9 shows this circuit.

Figure 11–9. Clock Gating Circuit Using an OR Gate



Because the register that generates the gate signal is triggered off the positive edge of the same clock, you need to consider the effect of using both edges of the same clock in your design. The timing diagram in Figure 11–10 shows the operation of this circuit. The *gate* signal occurs after the positive edge of the clock, and comes directly from a register. The logical OR of this *gate* signal with the original, un-inverted clock generates a clean clock signal. This clean, gated clock signal should only feed registers that use the negative edge of the same clock.

Figure 11–10. Timing Diagram for Clock Gating Circuit Using an OR Gate



If the delay between the register that generates the *gate* signal and the *gate* input to the AND gate is greater than the low period of the clock, (one half of the clock period for a 50% duty cycle clock), the clock pulse width is narrowed.



Altera recommends using a synchronous clock gating circuit because it is the only way to guarantee the duty cycle of the clock and to align the clock to the data.

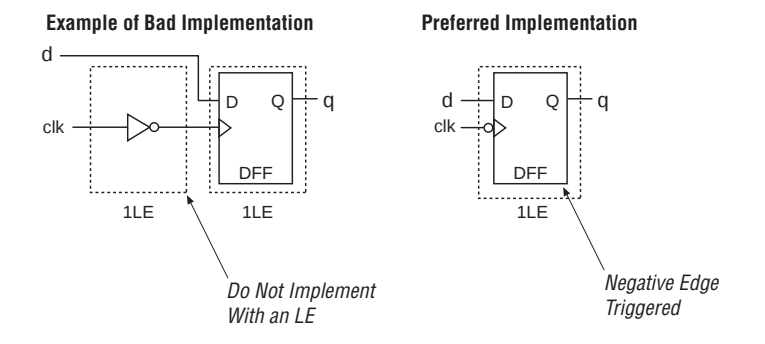
Inverted Clocks

A design may require both the positive edge and negative edge of a clock, as shown in [Figure 11-11](#). In Altera FPGAs, each logic element (LE) has a programmable clock inversion feature. Use this feature to generate an inverted clock.



Do not instantiate a LE look-up-table (LUT) configured as an inverter to generate the inverted clock signal.

Figure 11-11. An LE LUT Configured as an Inverter



Using a LUT to perform the clock inversion may lead to a clock insertion delay and skew, which poses a significant challenge to timing closure of the design. It also consumes more device resources than are necessary. Refer to [“Mixing Clock Edges” on page 11-14](#) for more information on this topic.



Do not generate schematics or register transfer level (RTL) code that instantiates LEs used to invert clocks. Instead, let the synthesis tool decide on the implementation of inverted clocks.

Clocks Driving Non-Clock Pins

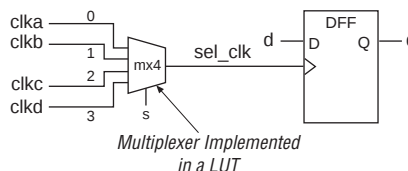
As a general guideline, clock sources should only be used to drive the register clock pins. There are exceptions to this rule, but every effort should be taken to minimize these exceptions or remove them altogether.

One category of exception is for various gated clocks, which are described in [“Preferred Clock Gating Circuit” on page 11-7](#).

You should avoid another exception, when possible, in which you use a clock multiplexer circuit to select one clock from a number of different clock sources, to drive non-clock pins. This type of circuit introduces

complexity into the static timing analysis of HardCopy and FPGA implementations. For example, as shown in [Figure 11–12](#), in order to investigate the timing of the `sel_clk` clock signal, it is necessary to make a clock assignment on the multiplexer output pin, which has a specific name. This name may change during the course of the design unless you preserve the node name in the Quartus II software settings. Refer to the Quartus II Help for more information on preserving node names.

Figure 11–12. A Circuit Showing a Multiplexer Implemented in a LUT



In the FPGA, a clock multiplexing circuit is built out of one or more LUTs, and the resulting multiplexer output clock may possibly no longer use one of the dedicated clock resources. Consequently, the skew and insertion delay of this multiplexed clock is potentially large, adversely impacting performance. The Quartus II Design Assistant traces clocks to their destination and, if it encounters a combinational gate, it issues a gated clock warning.

If the design requires this type of functionality, ensure that the multiplexer output drives one of the global routing resources in the FPGA. For example, this output should drive a fast line in an APEX™ 20KE device, or a global or regional clock in a Stratix® or Stratix II device.

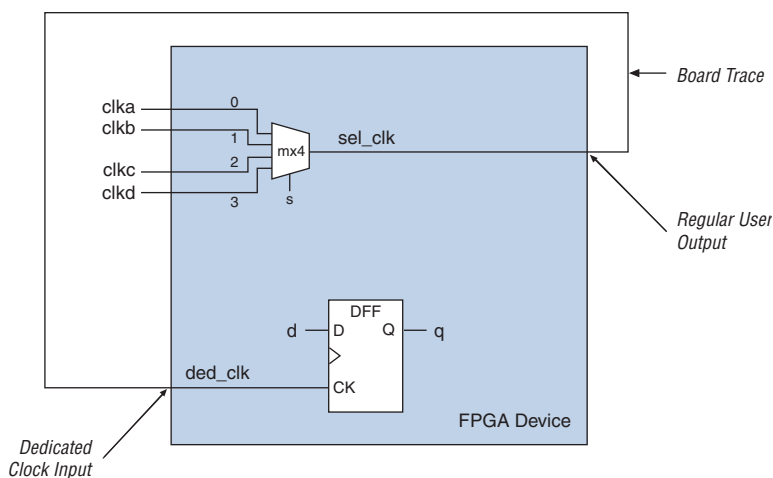
Enhanced PLL Clock Switchover

Clock source multiplexing can be done using the enhanced PLL clock switchover feature in Stratix and Stratix II FPGAs, and in HardCopy Stratix and HardCopy II structured ASICs. The clock switchover feature allows multiple clock sources to be used as the reference clock of the enhanced PLL. The clock source switchover can be controlled by an input pin or internal logic. This generally eliminates the need for routing a multiplexed clock signal out to a board trace and bringing it back into the device, as shown in [Figure 11–13](#).

Routing a multiplexed clock signal, as shown in [Figure 11–13](#), is only intended for APEX 20K FPGA and HardCopy APEX devices. This alternative to a clock multiplexing circuit ensures that a global clock resource is used to distribute the clock signal over the entire device by

routing the multiplexed clock signal to a primary output pin. Outside of the device, this output pin then drives one of the dedicated clock inputs of the same device, possibly through a phase-locked loop (PLL) to reduce the clock insertion delay. Although there is a large delay through the multiplexing circuit and external board trace, the resulting clock skew is very small because the design uses the dedicated clock resource for the selected clock signal. The advantage that this circuit has over the other implementations is that the timing analysis becomes very simple, with only a single-clock domain to analyze, whose source is a primary input pin to the APEX 20K FPGA or HardCopy APEX device.

Figure 11–13. Routing a Multiplexed Clock Signal to a Primary Output Pin



Clock Signals Should Use Dedicated Clock Resources

All clock signals in a design should be assigned to the global clock networks that exist in the target FPGA. Clock signals that are mapped to use non-dedicated clock networks can negatively affect the performance of the design. This is because the clock must be distributed using regular FPGA routing resources, which can be slower and have a larger skew than the dedicated clock networks. If your design has more clocks than are available in the target FPGA, you should consider reducing the number of clocks, so that only dedicated clock resources are used in the FPGA for clock distribution. If you need to exceed the number of dedicated clock resources, implement the clock with the lowest fan-out with regular (non-clock network) routing resources. Give priority to the fastest clock signals when deciding how to allocate dedicated clock resources.

In the Quartus II software, you can use the **Global Signal Logic** option to specify that a clock signal is a global signal. You can also use the auto **Global Clock Logic** option to allow the Fitter to automatically choose clock signals as global signals.



Altera recommends using the FPGA's built-in clock networks because they are pre-routed for low skew and for short insertion delay.

Mixing Clock Edges

You can use both edges of a single clock in a design. An example where both edges of a clock must be used in order to get the desired functionality is with a double data rate (DDR) memory interface. In Stratix II, Stratix, HardCopy II, and HardCopy Stratix devices, this interface logic is built into the I/O cell of the device, and rigorous simulation and characterization is performed on this interface to ensure its robustness. Consequently, this circuitry is an exception to the rule of using both edges of a clock. However, for general data transfers using generic logic resources, the design should only use a single edge of the clock. A circuit needs to use both edges of a single clock, then the duty cycle of the clock has to be accurately described to the Static Timing Analysis tool, otherwise inaccurate timing analysis could result.

Figure 11-14 shows two clock waveforms. One has a 50% duty-cycle, the other has a 10% duty cycle.

Figure 11-14. Clock Waveforms with 50% and 10% Duty Cycles



Figure 11–15 shows a circuit that uses only the positive edge of the clock. The distance between successive positive clock edges is always the same; for example, the clock period. For this circuit, the duty cycle of the clock has no effect on the performance of the circuit.

Figure 11–15. Circuit Using the Positive Edge of a Clock

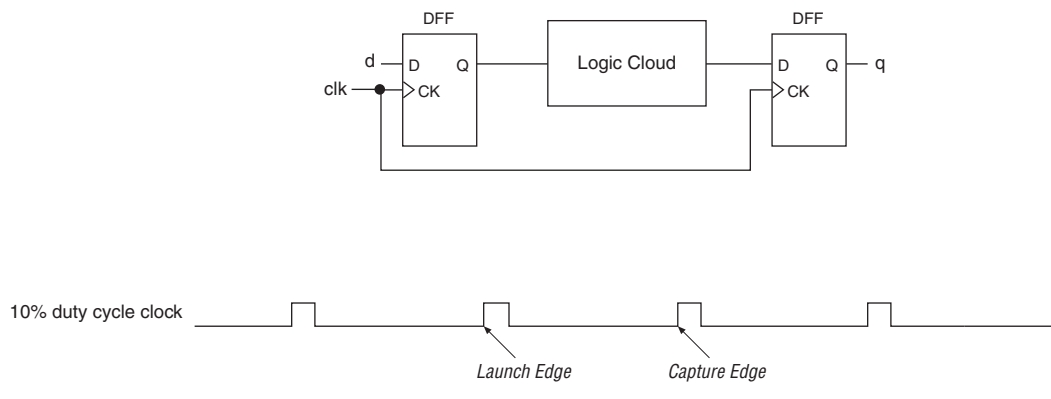
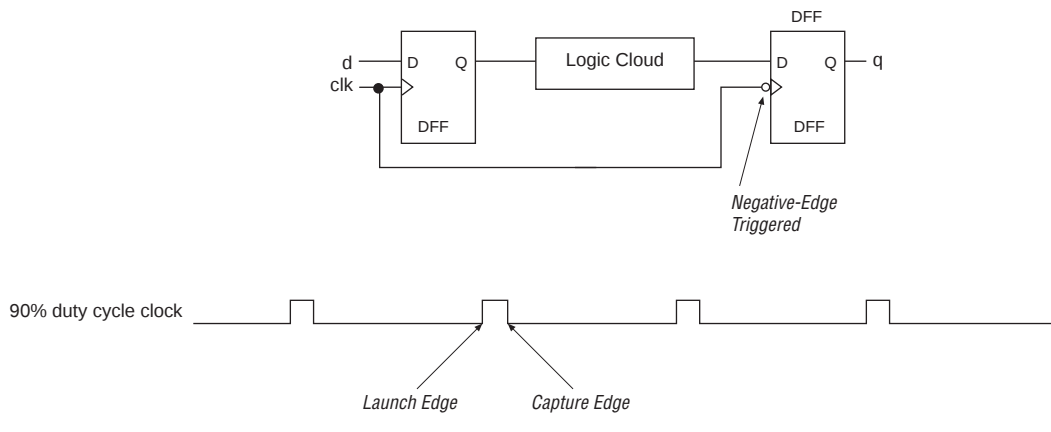


Figure 11–16 shows a circuit that used the positive clock edge to launch data and the negative clock edge to capture this data. Since this particular clock has a 10% duty cycle, the amount of time between the launch edge and capture edge is small. This small gap makes it difficult for the synthesis tool to optimize the cloud of logic so that no setup-time violations occur at the capture register.

Figure 11–16. Circuit Using the Positive and Negative Edges of a Clock



If you design a circuit that uses both clock edges, you could get the Design Assistant warning “Registers are Triggered by Different Edges of Same Clock.” You do not get this warning under the following conditions:

- If the opposite clock edge is used in a clock gating circuit
- A double data rate memory interface circuit is used

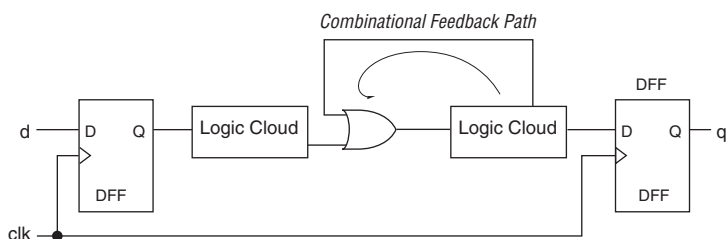


Try to only use a single edge of a clock in a design.

Combinational Loops

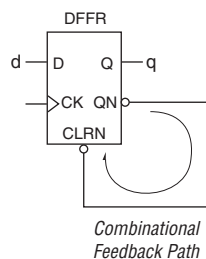
A combinational loop exists (Figure 11–17) if the output of a logic gate (or gates) feeds back to the input of the same gate without first encountering a register. A design should not contain any combinational loops.

Figure 11–17. A Circuit Using a Combinational Loop



It is also possible to generate a combinational loop using a register (Figure 11–18) if the register output pin drives the reset pin of the same register.

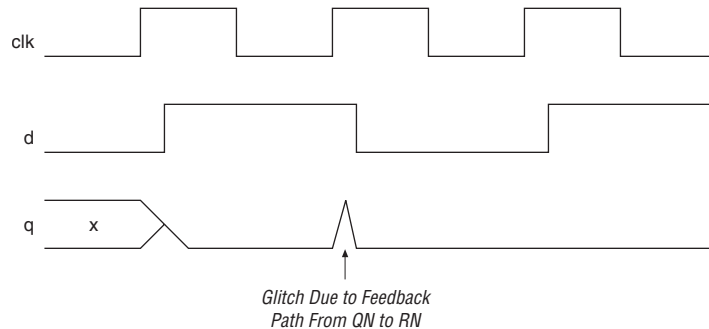
Figure 11–18. Generation of a Combinational Loop Using a Register



The timing diagram for this circuit is shown in Figure 11–19. When a logic 1 value on the register D input is clocked in, the logic 1 value appears on the Q output pin after the rising clock edge. The same clock event causes the QN output pin to go low, which in turn, causes the

register to be reset through RN. The Q register output consequently goes low. This circuit may not operate if there isn't sufficient delay in the QN-to-RN path, and is not recommended.

Figure 11-19. Timing Diagram for the Circuit Shown in Figure 11-18



Combinational feedback loops are either intentionally or unintentionally introduced into a design. Intentional feedback loops are typically introduced in the form of instantiated latches. An instantiated latch is an example of a combinational feedback loop in Altera FPGAs because its function has to be built out of a LUT, and there are no latch primitives in the FPGA logic fabric. Unintentional combinational feedback loops usually exist due to partially specified IF-THEN or CASE constructs in the register transfer level (RTL). The Design Assistant checks your design for these circuit structures. If any are discovered, you should investigate and implement a fix to your RTL to remove unintended latches, or re-design the circuit so that no latch instantiation is required. In Altera FPGAs, many registers are available, so there should never be any need to use a latch.

Combinational loops can cause significant stability and reliability problems in a design because the behavior of a combinational loop often depends on the relative propagation delays of the loop's logic. This combinational loop circuit structure behaves differently under different operation conditions. A combinational loop is asynchronous in nature, and EDA tools operate best with synchronous circuits.

A storage element such as a level-sensitive latch or an edge-triggered register has particular timing checks associated with it. For example, there is a setup-and-hold requirement for the data input of an edge-triggered register. Similarly, there is also a setup-and-hold timing requirement for the data to be stable in a transparent latch when the gate signal turns the latch from transparent to opaque. When latches are built

out of combinational gates, these timing checks do not exist, so the static timing analysis tool is not able to perform the necessary checks on these latch circuits.



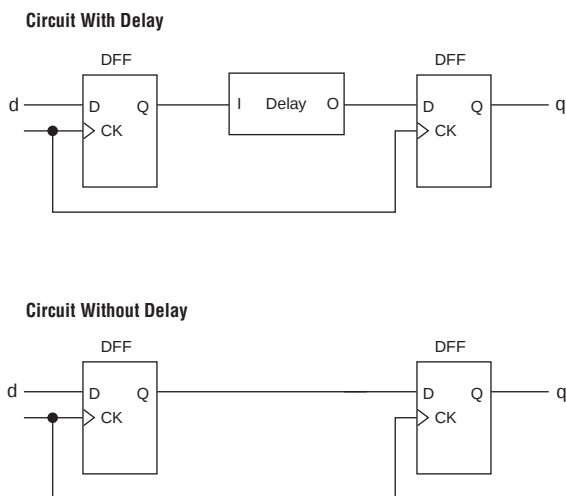
Check your design for intentional and unintentional combinational loops, and remove them.

Intentional Delays

Altera does not recommend instantiating a cell that does not benefit a design. This type of cell only delays the signal. For a synchronous circuit that uses a dedicated clock in the FPGA (Figure 11–20), this delay cell is not needed. In an ASIC, a delay cell is used to fix hold-time violations that occur due to the clock skew between two registers, being larger than the data path delay between those same two registers. The FPGA is designed with the clock skew and the clock-to-Q time of the FPGA registers in mind, to ensure that there is no need for a delay cell.

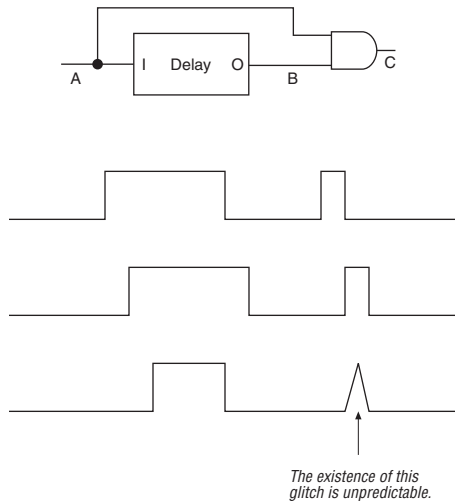
Figure 11–20 shows two versions of the same shift registers. Both circuits operate identically. The first version has a delay cell, possibly implemented using a LUT, in the data path from the Q output of the first register to the D input of the second register. The function of the delay cell is a non-inverting buffer. The second version of this circuit also shows a shift register function, but there is no delay cell in the data path. Both circuits operate identically.

Figure 11–20. Shift Register With and Without an Intentional Delay



If delay chains exist in a design, they are possibly symptomatic of an asynchronous circuit. One such case is shown in the circuit in [Figure 11–21](#). This circuit relies on the delay between two inputs of an AND gate to generate a pulse on the AND gate output. The pulse may or may not be generated, depending on the shape of the waveform on the A input pin.

Figure 11–21. A Circuit and Corresponding Timing Diagram Showing a Delay Chain



Using delay chains can cause various design problems, including an increase in a design's sensitivity to operating conditions and a decrease in design reliability.

Be aware that not all cases of delay chains in a design are due to asynchronous circuitry. If the Design Assistant report states that you have delay chains that you are unaware of (or are not expecting), the delay chains may be a result of using pre-built intellectual property (IP) functions. Pre-built IP functions may contain delay chains which the Design Assistant reports. These functions are usually parameterizable, and have thousands of different combinations of parameter settings. The synthesis tool may not remove all unused LEs from these functions when particular parameter settings are used, but the resulting circuit is still synchronous. Check all Design Assistant delay chain warnings carefully.

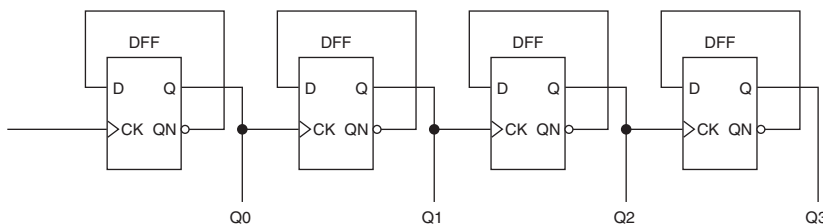


Avoid designing circuits that rely on the use of delay chains, and always carefully check any Design Assistant delay chain warnings.

Ripple Counters

Designs should not contain ripple counters. A ripple counter, shown in [Figure 11–22](#), is a circuit structure where the Q output of the first counter stage drives into the clock input of the following counter stage. Each counter stage consists of a register with the inverted QN output pin feeding back into the D input of the same register.

Figure 11–22. A Typical Ripple Counter



This type of structure is used to make a counter out of the smallest amount of logic possible. However, the LE structure in Altera FPGA devices allows you to construct a counter using one LE per counter-bit, so there is no logic savings in using the ripple counter structure. Each stage of the counter in a ripple counter contributes some phase delay, which is cumulative in successive stages of the counter. [Figure 11–23](#) shows the phase delay of the circuit in [Figure 11–22](#).

Figure 11–23. Timing Diagram Showing Phase Delay of Circuit Shown in [Figure 11–22](#)

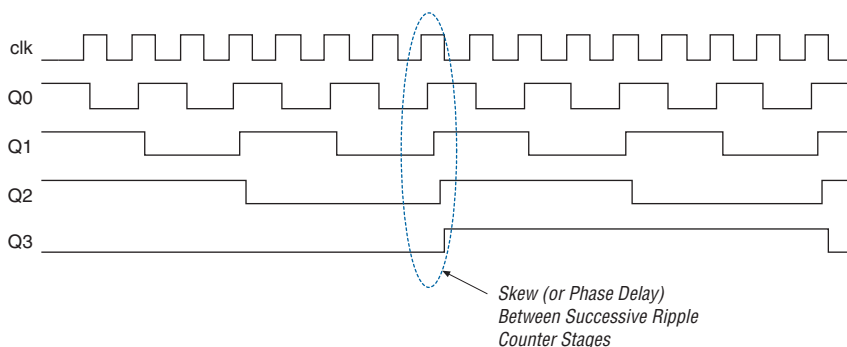
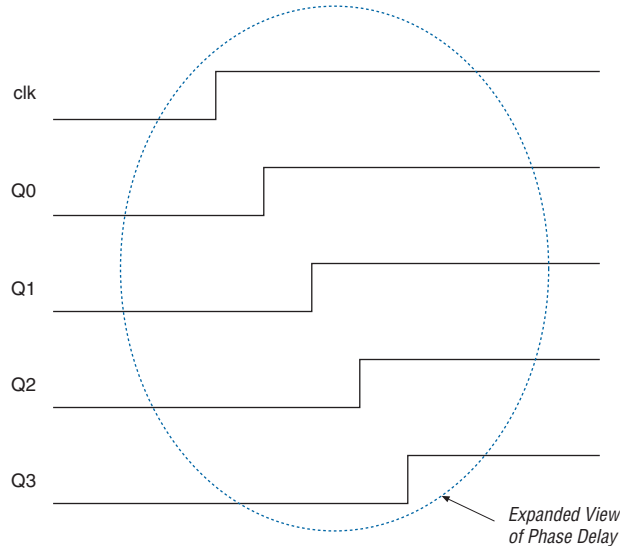


Figure 11-24 shows detailed view of the phase delay shown in Figure 11-23.

Figure 11-24. Detailed View of the Phase Delay Shown in Figure 11-23



This phase delay is problematic if the ripple counter outputs are used as clock signals for other circuits. Those other circuits are clocked by signals that have large skews.

Ripple counters are particularly challenging for static timing analysis tools to analyze as each stage in the ripple counter causes a new clock domain to be defined. The more clock domains that the static timing analysis tool has to deal with, the more complex and time-consuming the process becomes.



Altera recommends that you avoid using ripple counters under any circumstances.

Pulse Generators

A pulse generator is a circuit that generates a signal that has two or more transitions within a single clock period. Figure 11-25 shows an example of a pulse generator waveform.

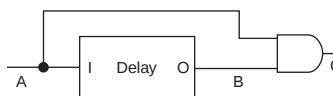


For more information on pulse generators, refer to “Intentional Delays” on page 11-18.

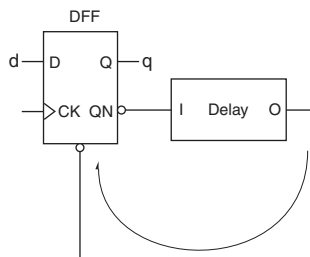
Figure 11–25. Example of a Pulse Generator Waveform

Creating Pulse Generators

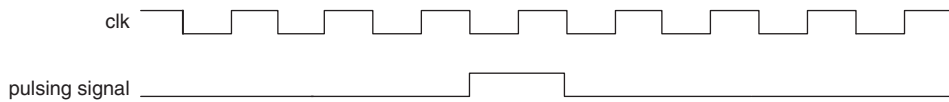
Pulse generators can be created in two ways. The first way to create a pulse generator is to increase the width of a glitch using a 2-input AND, NAND, OR, or NOR gate, where the source for the two gate inputs are the same, but the design delays the source for one of the gate inputs, as shown in [Figure 11–26](#).

Figure 11–26. A Pulse Generator Circuit Using a 2-Input AND

The second way to create a pulse generator is by using a register where the register output drives its own asynchronous reset signal through a delay chain, as shown in [Figure 11–27](#).

Figure 11–27. Pulse Generator Circuit Using a Register Output to Drive a Reset Signal Through a Delay Chain

These pulse generators are asynchronous in nature and are detected by the Design Assistant as unacceptable circuit structures. If you need to generate a pulsed signal, you should do it in a purely synchronous manner. That is, where the duration of the pulse is equal to one or more clock periods, as shown in [Figure 11–28](#).

Figure 11–28. An Example of a Synchronous Pulse Generator

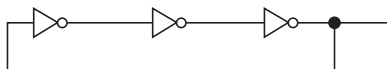
A synchronous pulse generator can be created with a simple section of Verilog HDL or VHDL code. The following is a Verilog HDL code fragment for a synchronous pulse generator circuit.

```
reg [2:0] count;
reg pulse;
always @ (posedge clk or negedge rst)
begin
    if (!rst)
        begin
            count[2:0] <= 3'b000;
            pulse <= 1'b0;
        end
    else
        begin
            count[2:0] <= count[2:0] + 1'b1;
            if (count == 3'b000)
                begin
                    pulse <= 1'b1;
                end
            else
                begin
                    pulse <= 1'b0;
                end
        end
    end
end
end
```

Combinational Oscillator Circuits

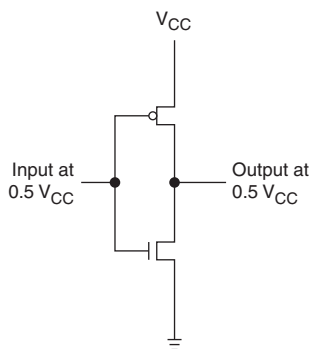
The circuit shown in [Figure 11–29 on page 11–24](#) consists of a combinational logic gate whose inverted output feeds back to one of the inputs of the same gate. This feedback path causes the output to change state and; therefore, oscillate.

Figure 11–29. A Combinational Ring Oscillator Circuit



This circuit is sometimes built out of a series of cascaded inverters in a structure known as a ring oscillator. The frequency at which this circuit oscillates depends on the temperature, voltage, and process operating conditions of the device, and is completely asynchronous to any of the other clock domains in the device. Worse, the circuit may fail to oscillate at all, and the output of the inverter goes to a stable voltage at half of the supply voltage, as shown in [Figure 11–30](#). This causes both the PMOS and NMOS transistors in the inverter chain to be switched on concurrently with a path from V_{CC} to GND, with no inverter function and consuming static current.

Figure 11–30. An Inverter Biased at $0.5 V_{CC}$



Avoid implementing any kind of combinational feedback oscillator circuit.

Reset Circuitry

Reset signals are control signals that synchronously or asynchronously affect the state of registers in a design. The special consideration given to clock signals also needs to be given to reset signals. Only the term “reset” is used in this document, but the information described here also applies to “set,” “preset,” and “clear” signals. Reset signals should only be used to put a circuit into a known initial condition. Also, both the `set` and `reset` pins of the same register should never be used together. If the signals driving them are both activated at the same time, the logic state of the register may be indeterminate.

Gated Reset

A gated reset is generated when combinational logic feeds into the asynchronous reset pin of a register. The gated reset signal may have glitches on it, causing unintentional resetting of the destination register. Figure 11–31 shows a gated reset circuit where the signal driving into the register reset pin has glitches on it causing unintentional resetting.

Figure 11–31. A Gated Reset Circuit and its Associated Timing Diagram

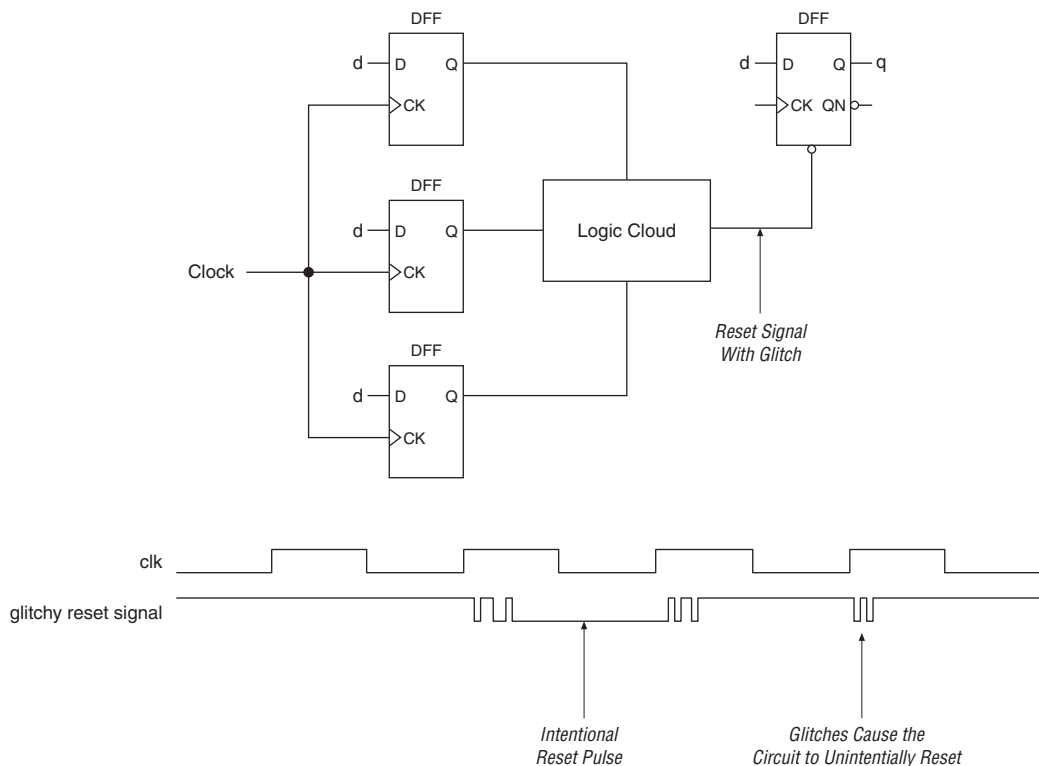
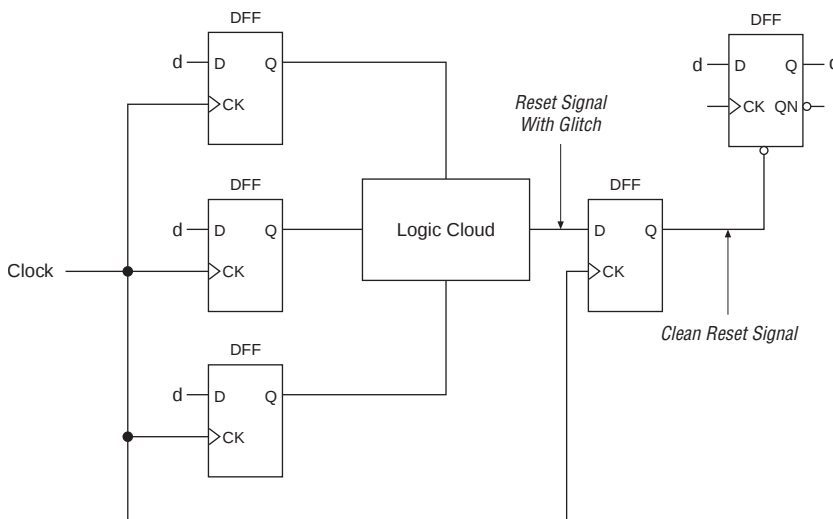


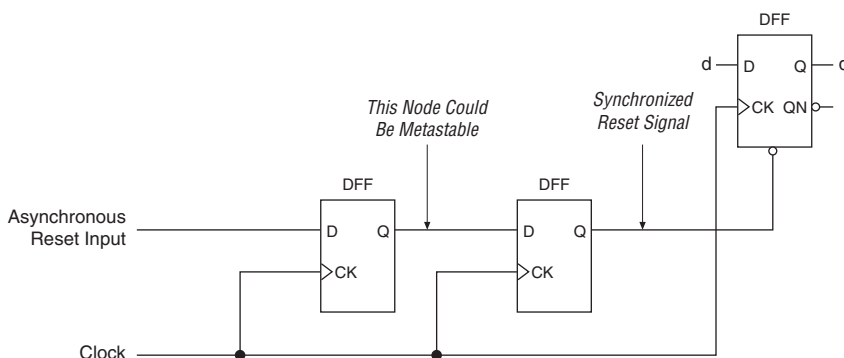
Figure 11–32 shows a better approach to implement a gated reset circuit, by placing a register on the output of the reset-gating logic, thereby synchronizing it to a clock. The register output then becomes a glitch-free reset signal that drives the rest of the design. However, the resulting reset signal is delayed by an extra clock cycle.

Figure 11–32. A Better Approach to the Gated Reset Circuit in Figure 11–31



Asynchronous Reset Synchronization

If the design needs to be put into a reset state in the absence of a clock signal, the only way to achieve this is through the use of an asynchronous reset. However, it is possible to generate a synchronous reset signal from an asynchronous one by using a double-buffer circuit, as shown in Figure 11–33.

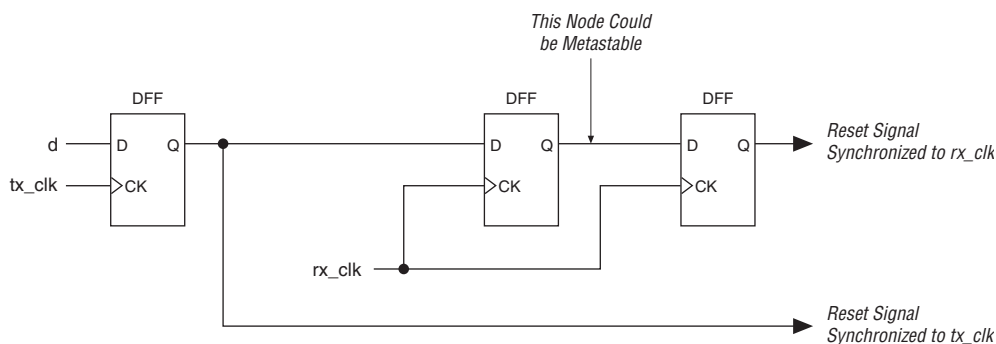
Figure 11–33. A Double-Buffer Circuit

Synchronizing Reset Signals Across Clock Domains

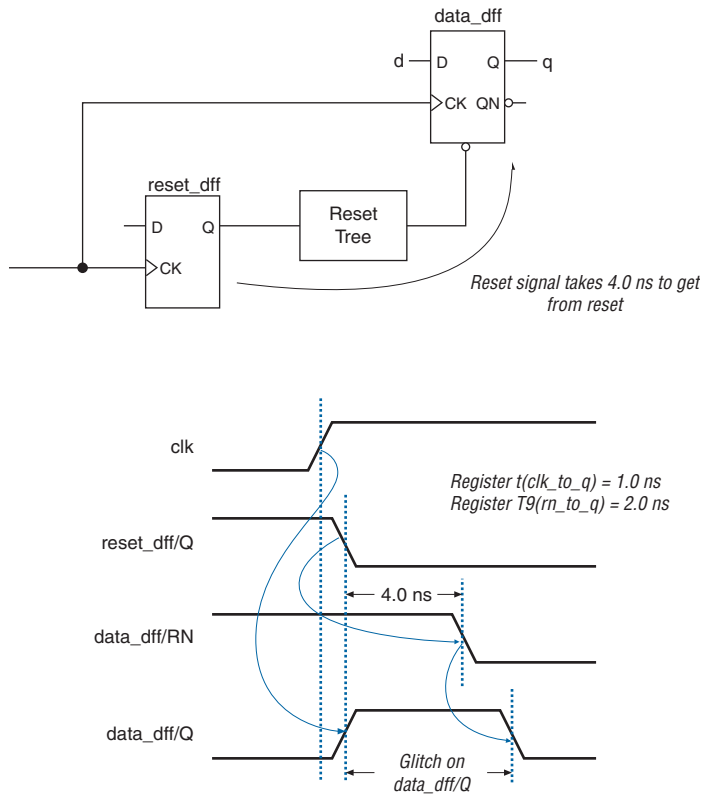
In a design, an internally generated reset signal that is generated in one clock domain, and used in one or more other asynchronous clock domains, should be synchronized. A reset signal that is not synchronized can cause metastability problems.

The synchronization of the gated reset should follow these guidelines, as shown in [Figure 11–34](#).

- The reset signal should be synchronized with two or more cascading registers in the receiving asynchronous clock domain.
- The cascading registers should be triggered on the same clock edge.
- There should be no logic between the output of the transmitting clock domain and the cascaded registers in the receiving asynchronous clock domain.

Figure 11–34. Circuit for a Synchronized Reset Signal Across Two Clock Domains

With either of the reset synchronization circuits described in [Figures 11–33](#) and [11–34](#), when the reset is applied, the Q output of the registers in the design may send a wrong signal, momentarily causing some primary output pins to also send wrong signals. The circuit and its associated timing diagram, shown in [Figure 11–35](#), demonstrate this phenomenon.

Figure 11–35. Common Problem with Reset Synchronization Circuits

A purely synchronous reset circuit does not exhibit this behavior. The following Verilog HDL RTL code shows how to do this.

```
always @ (posedge clk)
begin
    if (!rst)
        q <= 1'b0;
    else
        q <= d; end
```

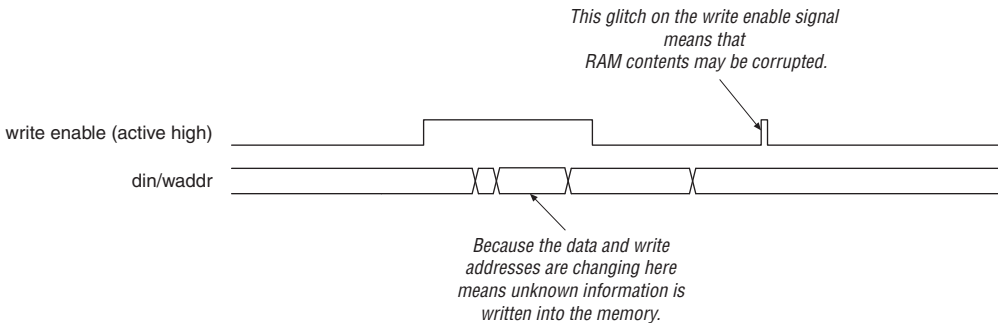


Avoid using reset signals for anything other than circuit initialization, and be aware of the reset signal timing if reset-synchronizing circuitry is used.

Asynchronous RAM

Altera FPGA devices contain flexible embedded memory structures that can be configured into many different modes. One possible mode is asynchronous RAM. The definition of an asynchronous RAM circuit is one where the write-enable signal driving into the RAM causes data to be written into it, without a clock being required, as shown in Figure 11–36. This means that the RAM is sensitive to corruption if any glitches exist on the write-enable signal. Also, the data and write address ports of the RAM should be stable before the write pulse is asserted, and must remain stable until the write pulse is de-asserted. These limitations in using memory structures in this asynchronous mode imply that synchronous memories are always preferred. Synchronous memories also provide higher design performance.

Figure 11–36. Potential Problems of Using Asynchronous RAM Structures



Stratix, Stratix II, HardCopy Stratix, and HardCopy II device architectures do not support asynchronous RAM behavior. These devices always use synchronous RAM input registers. Altera recommends using RAM output registering; this is optional, however, not using output registering degrades performance.

APEX 20K FPGA and HardCopy APEX support both synchronous and asynchronous RAM using the embedded system block (ESB). Altera recommends using synchronous RAM structures. Immediately registering both input and output RAM interfaces improves performance and timing closure.

Conclusion

Most issues described in this document can be easily avoided while a design is still in its early stages. These issues not only apply to HardCopy devices, but to any digital logic integrated circuit design, whether it is a standard cell ASIC, gate array, or FPGA.

Sometimes, violating one or more of the above guidelines is unavoidable, but understanding the implications of doing so is very important. One must be prepared to justify to Altera the need to break those rules in this case, and to support it with as much documentation as possible.

Following the guidelines outlined in this document can ultimately lead to the design being more robust, quicker to implement, easier to debug, and fitted more easily into the target architecture, increasing the likelihood of success.

Document Revision History

Table 11–2 shows the revision history for this chapter.

Table 11–2. Document Revision History (Part 1 of 2)		
Date and Document Version	Changes Made	Summary of Changes
September 2008, v3.4	Updated chapter number and metadata.	—
June 2007, v3.3	Minor text edits.	—
December 2006 v3.2	Added revision history.	Added revision history.
March 2006	Formerly chapter 14; no content change.	—
October 2005, v3.1	- Graphic updates - Minor edits	—
May 2005, v3.0	Updated the Using a FIFO Buffer section.	—
January 2005, v2.0	- Chapter title changed to <i>Design Guidelines for HardCopy Series Devices</i>. - Updated <i>Quartus® II Software Supported Versions</i> - Updated <i>HardCopy® Design Center Support</i> - Updated heading <i>Using a Double Synchronizer for Single-Bit Data Transfer</i> - Added <i>Stratix® II support for a global or regional clock</i> - Added <i>Support for Stratix II and HardCopy II to Mixing Clock Edges</i>	—

Table 11–2. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
August 2003, v1.1	Edited hierarchy of section headings.	—
May 2003, v1.0	Initial release	—

Introduction

Configuring an FPGA is the process of loading the design data into the device. Altera's SRAM-based Stratix® II, Stratix, APEX™ 20KC, and APEX 20KE FPGAs require configuration each time the device is powered up. After the device is powered down, the configuration data within the Stratix II, Stratix, or APEX device is lost and must be loaded again on power up.

There are several ways to configure these FPGAs. The details on the various configuration schemes available for these FPGAs are explained in the *Configuration Handbook*.

HardCopy® series devices are mask-programmed and cannot be configured. However, in addition to the capability of being instantly on upon power up (like a traditional ASIC device), these devices can mimic the behavior of the FPGA during the configuration process if necessary.

This chapter addresses various power-up options for HardCopy series devices. This chapter also discusses how configuration is emulated in HardCopy series devices while retaining the benefits of seamless migration and provides examples of how to replace the FPGAs in the system with HardCopy series devices.

HardCopy Power-Up Options

HardCopy series devices feature three variations of instant on power-up modes and a configuration emulation power-up mode. They are as follows:

- Instant on
- Instant on after 50 ms
- Configuration emulation of an FPGA configuration sequence



You must choose the power-up option when submitting the design database to Altera for migrating to a HardCopy series device. Once the HardCopy series devices are manufactured, the power-up option cannot be changed.



HardCopy II and some HardCopy Stratix devices do not support configuration emulation. Refer to [“Configuration Emulation of FPGA Configuration Sequence” on page 12-9](#) for more information.



HardCopy II and HardCopy Stratix devices retain the functionality of VCCSEL and PORSEL pins from the prototyping Stratix and Stratix II FPGAs. The signals can affect the HardCopy series power-up behavior using any power up option. Refer to the *Stratix Device Handbook* or the *Stratix II Device Handbook* for proper use of these additional signals.

Instant On Options

Instant on is the traditional power-up scheme of most ASIC and non-volatile devices. The instant on mode is the fastest power-up option of a HardCopy series device and is used when the HardCopy series device powers up independently while other components on the board still require initialization and configuration. Therefore, you must verify all signals that propagate to and from the HardCopy series device (for example, reference clocks and other input pins) are stable or do not affect the HardCopy series device operation.

There are two variations of instant on power-up modes available on all HardCopy devices.

- Instant on (no added delay)
- Instant on after 50 ms (additional delay)

Instant On (No Added Delay)

In the instant on power-up mode, once the power supplies ramp up above the HardCopy series device's power-on reset (POR) trip point, the device initiates an internal POR sequence. When this sequence is complete, the HardCopy series device transitions to an initialization phase, which releases the CONF_DONE signal to be pulled high. Pulling the CONF_DONE signal high indicates that the HardCopy series device is ready for normal operation. Figures 12-1 to 12-3 show the instant on timing waveform relationships of the configuration signals, V_{CC}, and user I/O pins with respect to the HardCopy series device's normal operation mode.

During the power-up sequence, internal weak pull-up resistors can pull the user I/O pins high. Once POR and the initialization phase is complete, the I/O pins are released. Similar to the FPGA, if the nIO_pullup pin transitions high, the weak pull-up resistors are disabled. Refer to the table that provides recommended operating conditions in the handbook for the specific device.

The value of the internal weak pull-up resistors on the I/O pins is in the Operating Conditions table of the specific FPGA's device handbook.

Instant On After 50-ms Delay

The instant on after 50-ms delay power-up mode is similar to the instant on power-up mode. However, in this case, the device waits an additional 50 ms following the end of the internal POR sequence before releasing the CONF_DONE pin. This option is useful if other devices on the board (such as a microprocessor) must be initialized prior to the normal operation of the HardCopy series device.

An on-chip oscillator generates the 50-ms delay after the power-up sequence. During the POR sequence and delay period, all user I/O pins can be driven high by internal, weak pull-up resistors. Just like the instant on mode, these pull-up resistors are affected by the nIO_pullup pin.

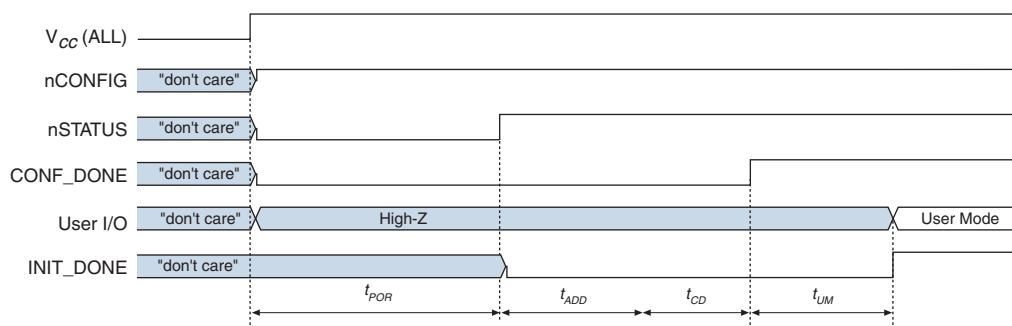


Similar to APEX 20K FPGAs, HardCopy APEX devices do not have an nIO_pullup function. Their internal, weak pull-up resistors are enabled during the power-up and initialization phase.

On the FPGA, an initialization phase occurs immediately after configuration where registers are reset, any PLLs used are initialized, and any I/O pins used are enabled as the device transitions into user mode. When the HardCopy series device uses instant on and instant on after 50-ms modes, a configuration sequence is not necessary, so the HardCopy series device transitions into the initialization phase after a power-up sequence immediately or after a 50-ms delay.

Figures 12–1 to 12–3 show instant on timing waveform relationships of the configuration signals, V_{CC}, and user I/O pins with respect to the HardCopy series device's normal operation mode. Tables 12–1 to 12–3 define the timing parameters for each of the HardCopy series device waveforms, and also show the effect of the PORSEL pin on power up. The nCE pin must be driven low externally for these waveforms to apply.

Figure 12–1 shows an instant on power-up waveform, where the HardCopy device is powered up, and the nCONFIG, nSTATUS, and CONF_DONE are not driven low externally.

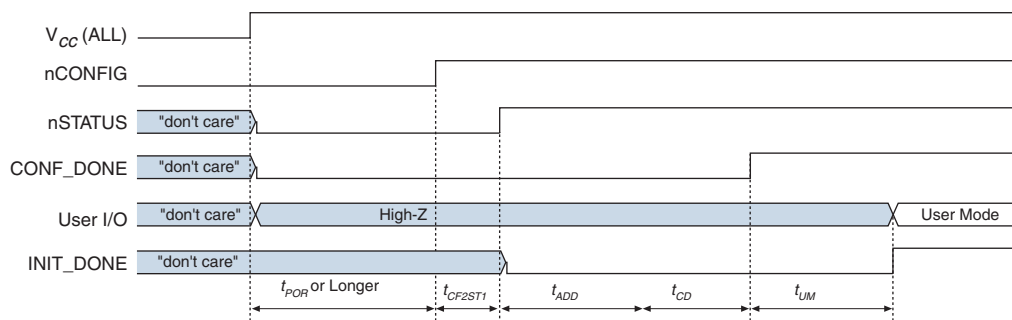
Figure 12–1. Timing Waveform for Instant On Option Notes (1), (2), (3), (4), (5)**Notes to Figure 12–1:**

- (1) $V_{CC} (ALL)$ represents either all of the power pins or the last power pin powered up to specified operating conditions. All HardCopy power pins must be powered within specifications as described under *Hot Socketing* sections.
- (2) $nCONFIG$, $nSTATUS$, and $CONF_DONE$ must not be driven low externally for this waveform to apply.
- (3) User I/O pins may be tri-stated or driven before and during power up. See the *Hot Socketing* sections for more details. The nIO_pullup pin can affect the state of the user I/O pins during the initialization phase.
- (4) $INIT_DONE$ is an optional pin that can be enabled on the FPGA using the Quartus II software. HardCopy series devices carry over the $INIT_DONE$ functionality from the prototyped FPGA design.
- (5) The $nCEO$ pin is asserted about the same time the $CONF_DONE$ pin is released. However, the nCE pin must be driven low externally for this waveform to apply.

An alternative to the power-up waveform in Figure 12–1 is if the $nCONFIG$ pin is externally held low longer than the $PORSEL$ delay. This delays the initialization sequence by a small amount as indicated in Figure 12–2.

In addition, Figure 12–2 is an instant on power-up waveform where $nCONFIG$ is momentarily held low and $nSTATUS$ and $CONF_DONE$ are not driven low externally.

Figure 12–2. Timing Waveform for Instant On Option Where nCONFIG is Held Low After Power Up Notes (1), (2), (3), (4), (5), (6)

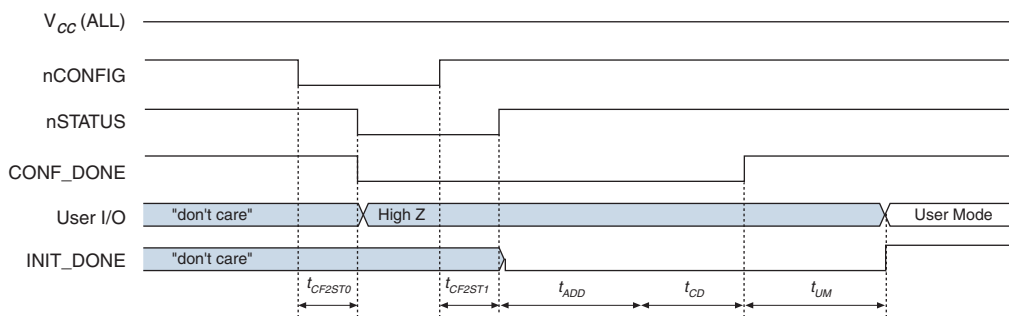


Notes to Figure 12–2:

- (1) This waveform applies if nCONFIG is held low longer than t_{POR} delay.
- (2) V_{CC} (ALL) represents either all of the power pins or the last power pin powered up to specified operating conditions. All HardCopy power pins must be powered within specifications as described under *Hot Socketing* sections.
- (3) nCONFIG, nSTATUS, and CONF_DONE must not be driven low externally for this waveform to apply.
- (4) User I/O pins may be tri-stated or driven before and during power up. See the *Hot Socketing* sections for more details. The nIO_pullup pin can affect the state of the user I/O pins during the initialization phase.
- (5) INIT_DONE is an optional pin that can be enabled on the FPGA using the Quartus II software. HardCopy devices carry over the INIT_DONE functionality from the prototyped FPGA design.
- (6) The nCEO pin is also asserted about the same time the CONF_DONE pin is released. However, the nCE pin must be driven low externally for this waveform to apply.

Pulsing the nCONFIG signal on an FPGA re-initializes the configuration sequence. The nCONFIG signal on a HardCopy series device also restarts the initialization sequence.

Figure 12–3 shows the instant on behavior of the configuration signals and user I/O pins if the nCONFIG pin is pulsed while the V_{CC} supplies are already powered up and stable.

Figure 12–3. Timing Waveform for Instant On Option When Pulsing NConfig Notes (1), (2), (3), (4), (5)**Notes to Figure 12–3:**

- (1) V_{CC} (ALL) represents either all of the power pins or the last power pin powered up to specified operating conditions. All HardCopy power pins must be powered within specifications as described under *Hot Socketing* sections.
- (2) nSTATUS and CONF_DONE must not be driven low externally for this waveform to apply.
- (3) The nIO_pullup pin can affect the state of the user I/O pins during the initialization phase.
- (4) INIT_DONE is an optional pin that can be enabled on the FPGA using the Quartus II software. HardCopy devices carry over the INIT_DONE functionality from the prototyped FPGA design.
- (5) The nCEO pin is also asserted about the same time the CONF_DONE pin is released. However, the nCE pin must be driven low externally for this waveform to apply.



In the FPGA, the INIT_DONE signal remains high for several clock cycles after the nCONFIG signal is asserted, after which time INIT_DONE goes low. In the HardCopy series device, the INIT_DONE signal starts low, as shown in Figure 12–3, regardless of the logic state of the nCONFIG signal. The INIT_DONE signal transitions high only after the CONF_DONE signal transitions high.

Tables 12–1 through 12–3 show the timing parameters for the instant on mode. These tables also show the time taken for completing the instant on power-up sequence in Figure 12–1 on page 12–4 for HardCopy series devices. This option is typical of an ASIC’s functionality.

Table 12–1. Timing Parameters for Instant On Mode in HardCopy II Devices

Parameter	Description	Condition	Min	Typical	Max	Units
t_{POR}	PORSEL delay (1)	12		12		ms
		100		100		ms
t_{CF2ST0}	nCONFIG low to nSTATUS low (1)				800	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high (1)				100	μs
t_{ADD}	Additional delay	Instant on	33		60	μs
		After 50 ms added delay	50		90	ms
t_{CD}	CONF_DONE delay		600		1100	ns
t_{UM}	User mode delay		25		55	μs

Note to Table 12–1:

- (1) This parameter is similar to the Stratix II FPGA specifications. Refer to the *Configuration Handbook* for more information.

Table 12–2. Timing Parameters for Instant On Mode in HardCopy Stratix Devices

Parameter	Description	Condition	Min	Typical	Max	Units
t_{POR}	PORSEL delay	2	1	2		ms
		100	70	100		ms
t_{CF2ST0}	nCONFIG low to nSTATUS low (1)				800	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high (1)				40	μs
t_{ADD}	Additional delay	Instant on	4		8	ms
		After 50 ms added delay	25	50	75	ms
t_{CD}	CONF_DONE delay		0.5		3	μs
t_{UM}	User mode delay		6.0		28	μs

Note to Table 12–2

- (1) This parameter is similar to the Stratix FPGA specifications. Refer to the *Configuration Handbook* for more information.

Table 12–3. Timing Parameters for Instant On Mode in HardCopy APEX Devices

Parameter	Description	Condition	Min	Typical	Max	Units
t_{POR}	POR delay			5		μs
t_{CF2ST0}	nCONFIG low to nSTATUS low (1)				200	ns
t_{CF2ST1}	nCONFIG high to nSTATUS high (1)				1	μs
t_{ADD}	Additional delay	Instant on		0		μs
		After 50 ms added delay		50		ms
t_{CD}	CONF_DONE delay		0.5		3	μs
t_{UM}	User mode delay		2.5		8	μs

Note to Table 12–3:

- (1) This parameter is similar to the APEX FPGA specifications. Refer to the *Configuration Handbook* for more information.

For correct operation of a HardCopy series device using the instant on option, pull the nSTATUS, nCONFIG, and CONF_DONE pins to V_{CC} . In the HardCopy series devices, these pins are designed with weak internal resistors pulled up to V_{CC} . Many FPGA configuration schemes require pull-up resistors on these I/O pins, so they may already be present on the board. In some HardCopy series device applications, you can remove these external pull-up resistors.

Altera recommends leaving external pull-up resistors on the board if one of the following conditions exists.



For more information, refer to the *Designing with 1.5-V Devices* chapter in the *Stratix Device Handbook*.

- There is more than one HardCopy series and/or FPGA on the board
- The HardCopy design uses configuration emulation
- The design uses MultiVolt I/O configurations

In the FPGA, you can enable the `INIT_DONE` pin in the Quartus II software. If you used the `INIT_DONE` pin on the FPGA prototype, the HardCopy series device retains its function.

- In HardCopy series devices, the `INIT_DONE` settings option is masked-programmed into the device. You must submit these settings to Altera with the final design prior to migrating to a HardCopy series device. The use of the `INIT_DONE` option and other option pins (for example, `DEV_CLRn` and `DEV_OE`) are available in the Fitter Device Options sections of the Quartus II report file.
- For HardCopy II and HardCopy Stratix devices, the `PORSEL` pin setting delays the POR sequence similar to the prototyping FPGA. For more information on `PORSEL` settings for the FPGA, refer to the *Configuration Handbook*.

In some FPGA configuration schemes, inputs `DCLK` and `DATA[7..0]` float if the configuration device is removed from the board. In the HardCopy series devices, these I/O pins are designed with weak, internal pull-up resistors, so the pins can be left unconnected on the board.

Configuration Emulation of FPGA Configuration Sequence

In configuration emulation mode, the HardCopy series device emulates the behavior of an APEX or Stratix FPGA during its configuration phase. When this mode is used, the HardCopy device uses a configuration emulation circuit to receive configuration bit streams. When all the configuration data is received, the HardCopy series device transitions into an initialization phase and releases the `CONF_DONE` pin to be pulled high. Pulling the `CONF_DONE` pin high signals that the HardCopy series device is ready for normal operation. If the optional open-drain `INIT_DONE` output is used, the normal operation is delayed until this signal is released by the HardCopy series device.



HardCopy II and some HardCopy Stratix devices do not support configuration emulation mode.

During the emulation sequence, the user I/O pins can be pulled high by internal, weak pull-up resistors. Once the configuration emulation and initialization phase is completed, the I/O pins are released. Similar to the FPGA, if the `nIO_pullup` pin is driven high, the weak pull-up resistors are disabled. The value of the internal weak pull-up resistors on the I/O pins can be found in the Operating Conditions table of the specific FPGA's device handbook.



Similar to APEX 20K FPGAs, HardCopy APEX devices do not have an `nIO_pullup` function. Their internal weak pull-up resistors are enabled during the power up and initialization phase.

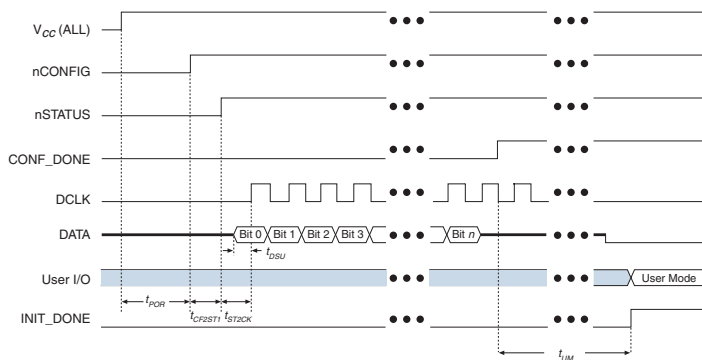


Similar to Stratix or APEX FPGAs, HardCopy Stratix or HardCopy APEX devices enter initialization phase immediately after a successful configuration sequence. At this time, registers are reset, any PLLs used are initialized, and any I/O pins used are enabled as the device transitions into user mode.

One application of the configuration emulation mode occurs when multiple programmable devices are cascaded in a configuration chain and only one device is replaced with a HardCopy series device. In this case, programming control signals and clock signals used to program the FPGA must also be used for the HardCopy series device. If this is not done, the HardCopy series device remains in the configuration emulation phase, the emulation sequence never ends, and the HardCopy `CONF_DONE` pin remains de-asserted. The proper configuration data stream and data clock is necessary so the HardCopy series device has the accurate emulation behavior.

Figure 12–4 shows a waveform of the configuration signals and the user I/O signals using configuration emulation mode.

Figure 12–4. Timing Waveform for Configuration Emulation Mode Notes (1), (2), (3), (4), (5)



Notes to Figures 12–4:

- (1) V_{CC} (ALL) represents either all of the power pins or the last power pin powered up to specified operating conditions. All HardCopy power pins must be powered within specifications as described under *Hot Socketing* sections.
- (2) $nCONFIG$, $nSTATUS$, and $CONF_DONE$ must not be driven low externally for this waveform to apply.
- (3) User I/O pins may be tri-stated or driven before and during power up. See the *Hot Socketing* sections for more details. The nIO_pullup pin can affect the state of the user I/O pins during the initialization phase.
- (4) $INIT_DONE$ is an optional pin that can be enabled on the FPGA using the Quartus II software. HardCopy devices will carry over the $INIT_DONE$ functionality from the prototyped FPGA design.
- (5) The $nCEO$ pin is also asserted about the same time the $CONF_DONE$ pin is released. However, the nCE pin must be driven low externally for this waveform to apply.

Configuration Emulation Timing Parameters

Tables 12–4 and 12–5 provide the timing parameters for the configuration emulation mode.

Table 12–4. Timing Parameters for Configuration Emulation Mode in HardCopy Stratix Devices *Note (1)*

Parameter	Description (2)	Condition	Min	Typ	Max	Units
t_{POR}	PORSEL delay	2	1	2		ms
		100	70	100		ms
t_{DSU}	Data setup time		7			ns
t_{CF2ST1}	nCONFIG high to nSTATUS				40	μ s
t_{ST2CK}	nSTATUS to DCLK		1			μ s
t_{UM}	User mode delay		6.0		28	μ s

Notes to Table 12–4:

- (1) HC1S80, HC1S60, and HC1S25 devices do not support emulation mode.
- (2) These parameters are similar to the Stratix FPGA specifications. Refer to the *Configuration Handbook* for more information.

Table 12–5. Timing Parameters for Configuration Emulation Mode in HardCopy APEX Devices

Parameter	Description (1)	Min	Typical	Max	Units
t_{POR}	POR delay		5		μ s
t_{DSU}	Data setup time	10			ns
t_{CF2ST1}	nCONFIG high to nSTATUS			1	μ s
t_{ST2CK}	nSTATUS to DCLK	1		3	μ s
t_{UM}	User mode delay	2		8	μ s

Notes to Table 12–5:

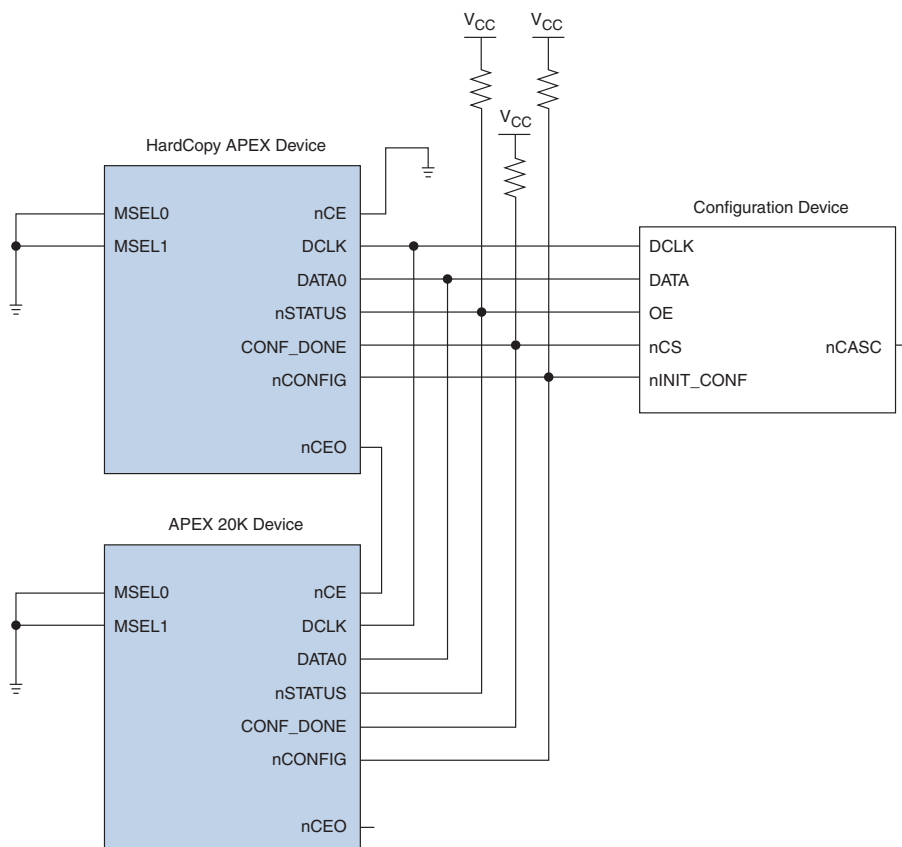
- (1) These parameters are similar to the APEX FPGA specifications. Refer to the *Configuration Handbook* for more information.

Benefits of Configuration Emulation

Configuration emulation in HardCopy series devices provides several advantages, including the following:

- Removes any necessity for changes to software, especially if the FPGA is configured using a microprocessor. Not having to change the software benefits the designer because microprocessor software changes demand significant system verification and qualification efforts, which also impact development time.
- Allows HardCopy series devices to co-exist with other FPGAs in a cascaded chain. None of the components need to be modified or added, and no design changes to the board are required. Additionally, no configuration software changes need to be made.
- Supports all configuration options available for the FPGA.

In this example, a single configuration device originally configured two APEX FPGAs. In [Figure 12-5](#), a HardCopy APEX device replaces an APEX FPGA.

Figure 12–5. Emulation of Configuration Sequence

A HardCopy series device in configuration emulation mode requires the same configuration control signals as the FPGA that was replaced. In configuration emulation mode, the HardCopy series device responds in exactly the same way as the FPGA. The `CONF_DONE` signal of the HardCopy series device is asserted at exactly the same time as the FPGA.

Power-Up Options Summary When Designing With HardCopy Series Devices

When designing a board for the prototyping FPGA with the intent of eventually replacing it with a HardCopy device, there are three power-up options that you should consider.

- Instant on
- Instant on after 50 ms
- Configuration emulation of an FPGA configuration sequence

You must choose the power-up option when submitting the design database to Altera for migrating to a HardCopy series device. Once the HardCopy series devices are manufactured, the power-up option cannot be changed.



HardCopy II and some HardCopy Stratix devices do not support configuration emulation mode.

HardCopy II and HardCopy Stratix devices retain the functionality of the `VCCSEL` and `PORSEL` pins from the prototyping Stratix II or Stratix FPGAs. For HardCopy II and HardCopy Stratix devices, the `PORSEL` pin setting delays the POR sequence similar to the prototyping FPGA.



For more information on `PORSEL` settings for the FPGA, refer to the *Configuration Handbook*.

The `nCE` and `nCEO` pins are functional in HardCopy series devices. The `nCE` pin must be held low for proper operation of the `nCEO` pin. If the `nCE` pin is driven low, the `nCEO` pin will be asserted after the initialization is completed and the `CONF_DONE` pin is released.

On the HardCopy II device, the `nCE` pin delays the initialization if it is not driven low. Like in the Stratix II device, `nCEO` and `TDO` of the HardCopy II device are powered by `VCCIO`.

If you used the `INIT_DONE` pin on the FPGA prototype, the HardCopy series device retains its function. In HardCopy series devices, the `INIT_DONE` settings option is masked-programmed into the device. These settings must be submitted to Altera with the final design prior to migrating to a HardCopy series device. The use of the `INIT_DONE` option and other option pins (for example, `DEV_CLRn` and `DEV_OE`) are available in the Fitter Device Options sections of the Quartus II report file.

HardCopy II devices do not support the user-supplied start-up clock option available for Stratix II devices. The HardCopy II device uses its own internal clock for power-up circuitry. The startup clock selection is an option for configuring the FPGA, which you can set in the Quartus II software under Device and Pin Options.

HardCopy devices support device-wide reset (`DEV_CLRn`) and device-wide output enable (`DEV_OE`). The HardCopy settings follow the prototyping FPGA setting, which you set in the Quartus II software under Device and Pin Options.

For correct operation of a HardCopy series device using the instant on option, pull the `nSTATUS`, `nCONFIG`, and `CONF_DONE` pins to V_{CC} . In the HardCopy series devices, these pins are designed with weak, internal resistors pulled up to V_{CC} . Many FPGA configuration schemes require pull-up resistors on these I/O pins, so they may already be present on the board. In some HardCopy series device applications, you can remove these external pullup resistors.

Altera recommends leaving external pull-up resistors on the board if one of the following conditions exists:

- There is more than one HardCopy series and/or FPGA on the board
- The HardCopy design uses configuration emulation
- The design uses MultiVolt™ I/O configurations



For more information, refer to the *Designing with 1.5-V Devices* chapter in the *Stratix Device Handbook*.

In some FPGA configuration schemes, inputs `DCLK` and `DATA[7..0]` float if the configuration device is removed from the board. In the HardCopy series devices, these I/O pins are designed with weak internal pull-up resistors, so the pins can be left unconnected on the board.

When designing a board with a Stratix II prototype device and its companion HardCopy II device, most configuration pins required by the Stratix II device are not required by the HardCopy II device. To maximize I/O pin counts with HardCopy II device utilization, Altera recommends minimizing power-up and configuration pins that do not carry over from a Stratix II device into a HardCopy II device. More information can be found on the *Migrating Stratix II Device Resources to HardCopy II Devices* chapter.

HardCopy devices support the MSEL settings used on the FPGA. You are not required to change these settings on the board when replacing the prototyping FPGA with the HardCopy series device.

HardCopy II devices do not use MSEL pins and these pin locations are not connected in the package. It is acceptable to drive these pins to V_{CC} or GND as required by the prototyping Stratix II device.

Pulsing the `nCONFIG` signal on an FPGA re-initializes the configuration sequence. The `nCONFIG` signal on a HardCopy series device also restarts the initialization sequence.

The HardCopy device JTAG pin locations match their corresponding FPGA prototypes. Like the FPGAs, the JTAG pins have internal weak pull ups or pull downs on the four input pins TMS, TCK, TDI, and TRST. There is no requirement to change the JTAG connections on the board when replacing the prototyping FPGA with the HardCopy series device. More information on JTAG pins is the corresponding *Boundary-Scan Support* chapter for each device.

Power-Up Option Selection and Examples

The HardCopy series device power-up option is mask-programmed. Therefore, it is important that the board design is verified to ensure that the HardCopy series device power-up option chosen will work properly. This section provides recommendations on selecting a power-up option and provides some examples.

Table 12–6 shows a comparison of applicable FPGA and HardCopy power up options.

Table 12–6. FPGA Configuration Modes and HardCopy Series Power-Up Schemes (Part 1 of 2)

Power Up Scheme	Device Family					
	Stratix II	Stratix	APEX 20K APEX 20KE APEX 20KC	HardCopy II (1)	HardCopy Stratix (2)	HardCopy APEX
Instant on				✓	✓	✓
Instant on after 50 ms				✓	✓	✓
Passive serial (PS)	✓	✓	✓		✓	✓
Active serial (AS)	✓					
Fast passive parallel (FPP)	✓	✓			✓	
Passive parallel synchronous (PPS)			✓			✓
Passive parallel asynchronous (PPA)	✓	✓	✓		✓	✓
Joint Test Action Group (JTAG)	✓	✓	✓		✓	✓
Remote local update FPP (3)	✓	✓				

Table 12–6. FPGA Configuration Modes and HardCopy Series Power-Up Schemes (Part 2 of 2)

Power Up Scheme	Device Family					
	Stratix II	Stratix	APEX 20K APEX 20KE APEX 20KC	HardCopy II (1)	HardCopy Stratix (2)	HardCopy APEX
Remote local update PPA (3)	✓	✓				
Remote local update PS (3)		✓				

Notes to Table 12–6:

- (1) HardCopy II devices do not support emulation mode.
 (2) HC1S80, HC1S60, and HC1S25 devices do not support emulation mode.
 (3) The remote/local update feature of Stratix devices is not supported in HardCopy Stratix devices.

Power-up option recommendations depend on the following board configurations:

- Single HardCopy series device replacing a single FPGA on the board
- One or more HardCopy series devices replacing one or more FPGA of a multiple-device configuration chain
- All HardCopy series devices replacing all FPGAs of a multiple-device configuration chain

In a multiple-device configuration chain, more than one FPGA on a board obtains configuration data from the same source.

Replacing One FPGA With One HardCopy Series Device

Altera recommends using the instant on or instant on after 50 ms mode when replacing an FPGA with a HardCopy series device regardless of the board configuration scheme. Table 12–7 gives a summary of HardCopy series device power-up options when a single HardCopy series device replaces a single FPGA on the board.



Table 12–7 does not include HardCopy II options because HardCopy II devices only support instant on and instant on after 50 ms modes.

Table 12–7. Summary of Power-Up Options for One HardCopy Series Device Replacing One FPGA

Configuration Scheme	HardCopy APEX Options	HardCopy Stratix Options	Comments
PS with configuration device(s) or download cable (1)	- Instant on - Instant on after 50 ms	- Instant on - Instant on after 50 ms	The configuration device(s) must be removed from the board.
FPP with enhanced configuration devices	Not available	- Instant on - Instant on after 50 ms	The configuration device(s) must be removed from the board.
PS, PPA, PPS, FPP, with a microprocessor (2)	Emulation	Emulation (3)	If the microprocessor code can be changed, the design should use the instant on or instant on after 50 ms mode. However, the microprocessor still needs to drive a logic ‘1’ value on the HardCopy nCONFIG pin
JTAG configuration	- Instant on after 50 ms - Emulation	- Instant on after 50 ms - Emulation (3)	Configuration emulation mode can be used but delays the initialization of the board or device.

Notes to Table 12–7:

- (1) Download cable used may be either MasterBlaster™, USB Blaster, ByteBlaster™ II, or ByteBlasterMV™ hardware.
- (2) For parallel programming modes, DATA[7..1] pins have weak pull up resistors on the HardCopy series device, which can be optionally enabled or disabled through metallization. DCLK and DATA[0] pins have internal weak pull-up resistors.
- (3) HC1S80, HC1S60, and HC1S25 devices do not support emulation mode.

Replacing One or More FPGAs With One or More HardCopy Series Devices in a Multiple-Device Configuration Chain

Altera recommends using the instant on or instant on after 50 ms mode when replacing an FPGA with a HardCopy series device, regardless of configuration scheme. Table 12–8 gives a summary of HardCopy series device power-up options when a single HardCopy series device replaces a single FPGA of a multiple-device configuration chain.



When using the instant on or instant on after 50 ms mode, the HardCopy series device could be in user-mode and ready before other configured devices on the board. It is important to verify that any signals that communicate to and from the HardCopy series device are stable or will not affect the HardCopy series device or other device operation while the devices are still in the power up or configuration stage. For example, if the HardCopy series design used a PLL reference clock that is not available until after other devices are fully powered up, the HardCopy series device PLL will not operate properly unless the PLLs are reset.



Table 12–8 does not include HardCopy II options because HardCopy II devices only support instant on and instant on after 50 ms modes.

Table 12–8. Power-Up Options for One or More HardCopy Series Devices Replacing FPGAs in a Multiple-Device Configuration Chain (Part 1 of 2)

Configuration Scheme	HardCopy APEX Options	HardCopy Stratix Options	Comments
PS with configuration device(s) or download cable (1) FPP with enhanced configuration device (4)	- Emulation - Instant on (3) - Instant on after 50 ms (3)	- Emulation (2) - Instant on (3) - Instant on after 50 ms (3)	Instant on or instant on after 50 ms modes can be used if the <code>nCE</code> pin of the following APEX or Stratix device can be tied to logic 0 on the board and the configuration data is modified to remove the HardCopy series device configuration data. The configuration sequence then skips the HardCopy series device.
PS, PPA, PPS, FPP, with a microprocessor (4)	Emulation	Emulation (2)	If the microprocessor code can be changed, the design should use the instant on or instant on after 50 ms mode. However, the microprocessor still needs to drive a logic '1' value on the HardCopy series device <code>nCONFIG</code> pin.

Table 12–8. Power-Up Options for One or More HardCopy Series Devices Replacing FPGAs in a Multiple-Device Configuration Chain (Part 2 of 2)

Configuration Scheme	HardCopy APEX Options	HardCopy Stratix Options	Comments
JTAG configuration	● Emulation	● Emulation (2)	If the HardCopy series device is put in BYPASS mode and the JTAG programming data is modified to remove the HardCopy configuration information, instant on or instant on after 50 ms modes can be used.

Notes to Table 12–8:

- (1) Download cable used may be either MasterBlaster, USB Blaster, ByteBlaster II, or ByteBlasterMV hardware.
- (2) HC1S80, HC1S60, and HC1S25 devices do not support emulation mode.
- (3) If the HardCopy series device is the last device in the configuration chain, Altera recommends using instant on modes.
- (4) For parallel programming modes, DATA [7 . . 1] pins have weak pull up resistors on the HardCopy series device, which can be optionally enabled or disabled through metallization. DCLK and DATA [0] pins also have weak pull-up resistors.

Replacing all FPGAs with HardCopy Series Devices in a Multiple-Device Configuration Chain

When all Stratix II, Stratix, and APEX FPGAs are replaced by HardCopy II, HardCopy Stratix, and HardCopy APEX devices, respectively, Altera recommends using the instant on or instant on after 50 ms mode, regardless of configuration scheme.

Once the HardCopy series devices replace the FPGAs, any configuration devices used to configure the FPGAs should be removed from the board. Microprocessor code, if applicable, should be changed to account for the HardCopy series device power-up scheme. You can use the JTAG chain to perform other JTAG operations except configuration.

FPGA to HardCopy Configuration Migration Examples

The following are examples of how HardCopy series devices replace FPGAs that use different FPGA configuration schemes.

HardCopy Series Device Replacing a Stand-Alone FPGA

In this example, the single HardCopy series device uses the instant on power-up option, as shown in Figure 12–7. The configuration device, now redundant, is removed, and no further board changes are necessary. The pull-up resistors on the nCONFIG, nSTATUS, and CONF_DONE pins can be removed, but should be left on the board if configuration emulation or multiple-voltage I/O standards are used. You could also use the instant on after 50 ms power-up mode in this example.

Figures 12–6 and 12–7 show how a HardCopy series device replaces an FPGA previously configured with an Altera configuration device.

Figure 12–6. Configuration of a Stand-Alone FPGA *Note (1)*

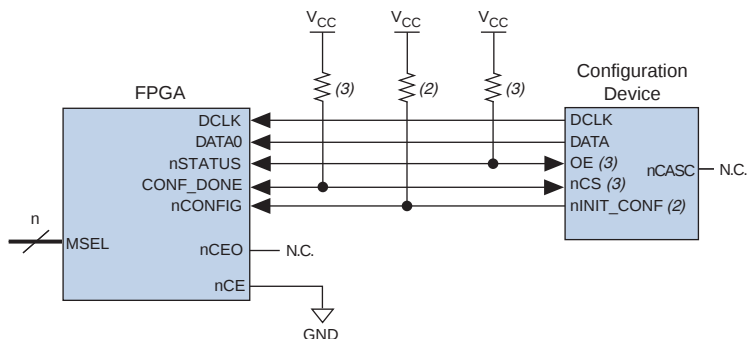
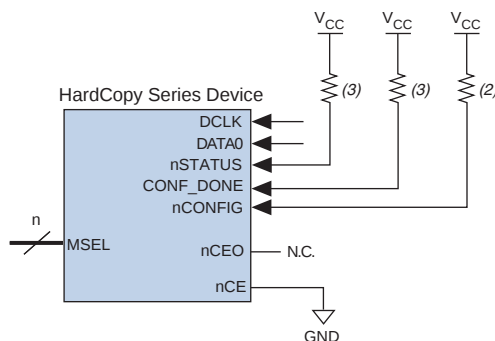


Figure 12–7. HardCopy Series Device Replacing Stand-Alone FPGA *Note (1)*



Notes to Figures 12–6 and 12–7:

- (1) For details on configuration interface connections, refer to the *Configuration Handbook*. The handbook includes information on MSEL pins set to PS mode.
- (2) The nINIT_CONF pin (available on enhanced configuration and EPC2 devices) has an internal pull-up resistor that is always active. Therefore, the nINIT_CONF/nCONFIG line does not require an external pull-up resistor. 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, use a resistor to pull the nCONFIG pin to V_{CC}.
- (3) Enhanced configuration and EPC2 devices have internal programmable pull-up resistors on OE and nCS pins. Refer to the *Configuration Handbook* for more details of this application in FPGAs. HardCopy series devices have internal weak pull-up resistors on nSTATUS, nCONFIG, and CONF_DONE pins.

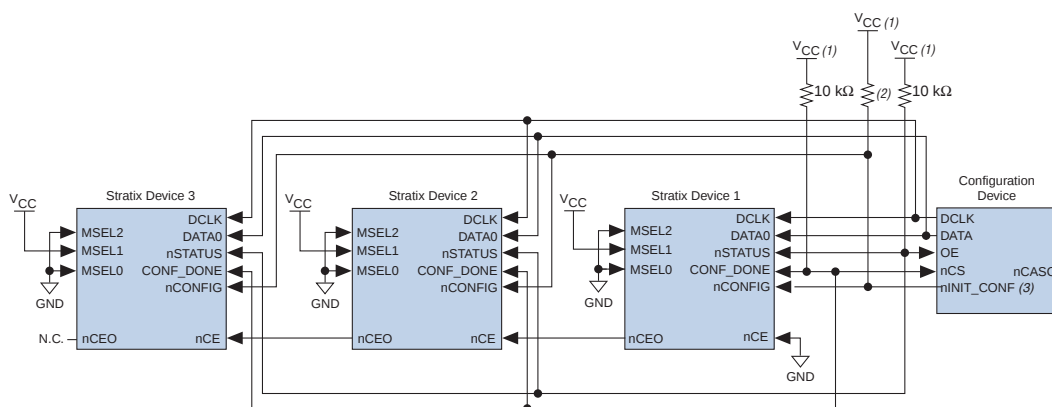
HardCopy Series Device Replacing an FPGA in a Cascaded Configuration Chain

Figure 12–8 shows a design where the configuration data for the Stratix devices is stored in a single configuration device, and the FPGAs are connected in a multiple-device configuration chain. The second device in the chain is replaced with a HardCopy Stratix device, as shown in Figure 12–9.



For more information on Stratix FPGA configuration schemes, refer to the *Configuration Handbook*.

Figure 12–8. Configuration of Multiple FPGAs in a Cascade Chain

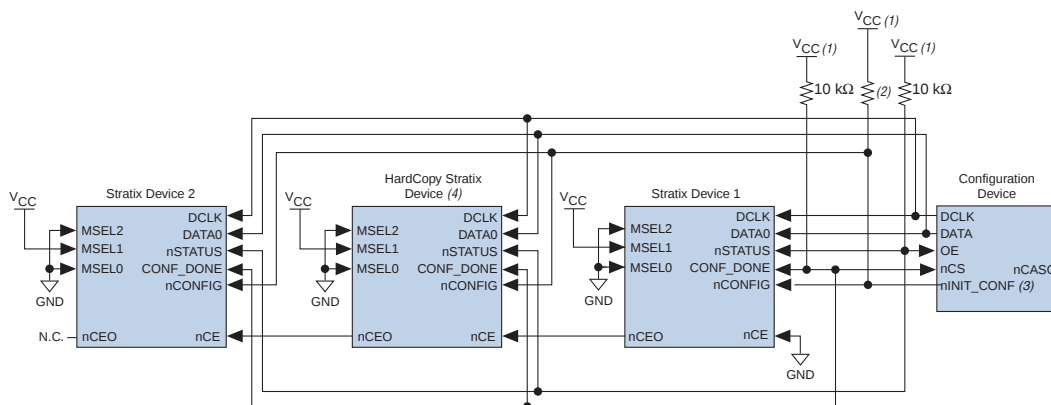


Notes to Figure 12–8:

- (1) The pull-up resistors are connected to the same supply voltage as the configuration device.
- (2) The enhanced configuration devices and EPC2 devices have internal programmable pull-up resistors on the OE and nCS pins. Refer to the *Configuration Handbook* for more details.
- (3) The nINIT_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices. Refer to the *Configuration Handbook* for more details.

Configuration with the HardCopy Series Device in the Cascade Chain

Figure 12–9 shows the same cascade chain as Figure 12–8, but the second FPGA in the chain has been replaced with a HardCopy Stratix device.

Figure 12–9. Replacing an FPGA with a HardCopy Equivalent in the Cascade Chain**Notes to Figure 12–9:**

- (1) The pull-up resistors are connected to the same supply voltage as the configuration device.
- (2) The enhanced configuration devices and EPC2 devices have internal programmable pull-up resistors on the OE and nCS pins. Refer to the *Configuration Handbook* for more details.
- (3) The nINIT_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices. Refer to the *Configuration Handbook* for more information.
- (4) HC1S80, HC1S60, and HC1S25 devices do not support emulation mode and cannot be used in this method.

In this example, the HardCopy Stratix device can only be configured using the configuration emulation mode. The configuration device cannot be removed, as it is still required by other Stratix devices in the chain. While the HardCopy Stratix device does not need the data stored in the configuration device, the data in the configuration device is not modified to reflect this. The emulation mode ensures that the HardCopy series device nCEO pin is asserted correctly after the emulation of the configuration sequence. The nCEO pin enables the next device in the chain to receive the correct configuration data from the configuration device. Additionally, with the configuration emulation mode, you do not need to make any changes to the board.

Configuration With the HardCopy Series Device Removed From the Cascade Chain

An alternative method to configure FPGAs on a board with both HardCopy series devices and FPGAs is to remove the HardCopy series device from the cascade chain. Figure 12–10 shows how the devices are connected with the HardCopy series device removed from the chain.

The data in the configuration device should be modified to exclude the HardCopy series device configuration data. The HardCopy series device can use any of the three power-up options.



- (1) The pull-up resistors are connected to the same supply voltage as the configuration device.
- (2) The enhanced configuration devices and EPC2 devices have internal programmable pull-up resistors on the OE and nCS pins. Refer to the *Configuration Handbook* for more details.
- (3) The nINIT_CONF pin is available on EPC16, EPC8, EPC4, and EPC2 devices. Refer to the *Configuration Handbook* for more information.
- (4) HC1S80, HC1S60, and HC1S25 devices do not support emulation mode and cannot be used in this method.

- The **nCE** pin of the HardCopy series device must be tied to GND.
- The **nCE** pin of the FPGA that was driven by the HardCopy series **nCEO** pin must now be driven by the **nCEO** pin of the FPGA that precedes the HardCopy series device in the chain.

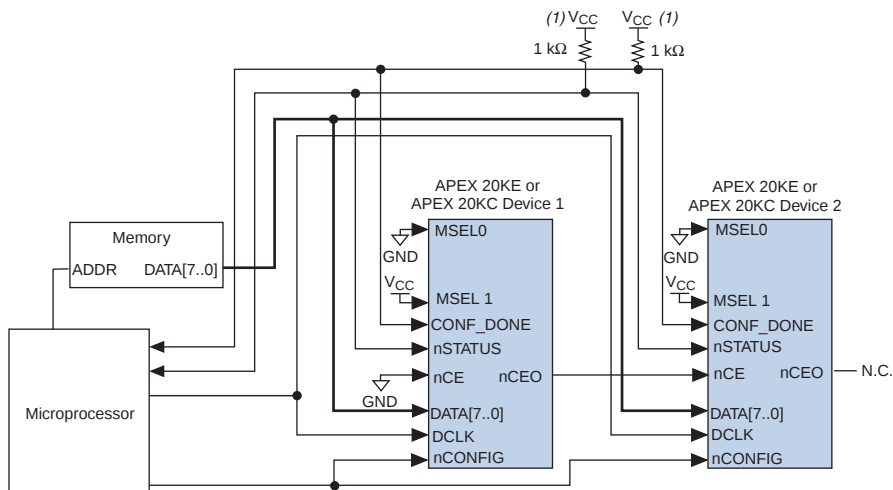
HardCopy Series Device Replacing an FPGA Configured Using a Microprocessor

The HardCopy series device can replace FPGAs that are configured using a microprocessor, as shown in [Figures 12-12](#) and [12-13](#). While the instant on mode is the most efficient, designers can also use the instant on after 50 ms and configuration emulation mode.

Figure 12-11 shows an application where APEX FPGAs are configured using a microprocessor in the PPS configuration scheme.



For more information on the PPS configuration scheme, refer to the *Configuration Handbook*.

Figure 12–11. Configuring FPGAs Using a Microprocessor**Note to Figure 12–11:**

- (1) Connect the pull-up resistors to a supply that provides an acceptable input signal for all devices in the chain.

When the HardCopy series device replaces the last FPGA of the configuration sequence (as shown in Figure 12–12), use the instant on or instant on after 50 ms mode. However, you must modify the microprocessor code to eliminate the configuration data for the last FPGA of the configuration chain.

Figures 12–12 and 12–13 show the HardCopy APEX device replacing APEX FPGAs either first or last in the configuration chain.

Figure 12–12. Replacement of Last FPGA in the Chain With a HardCopy Series Device

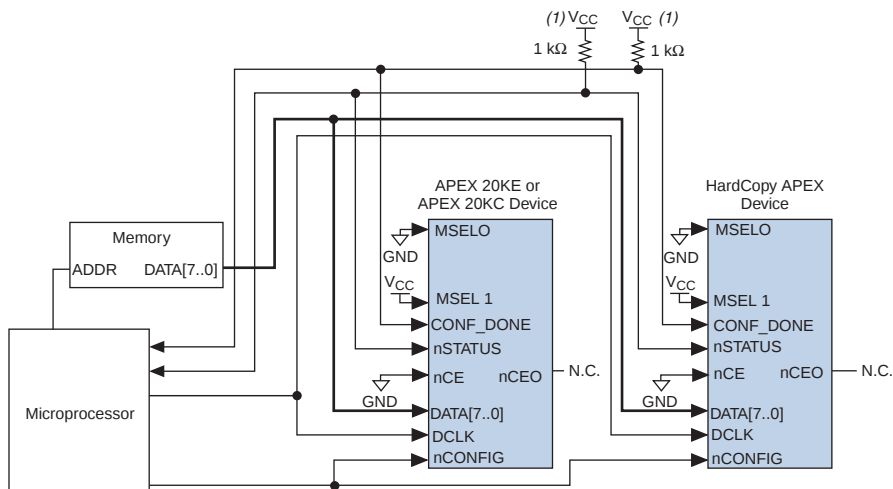
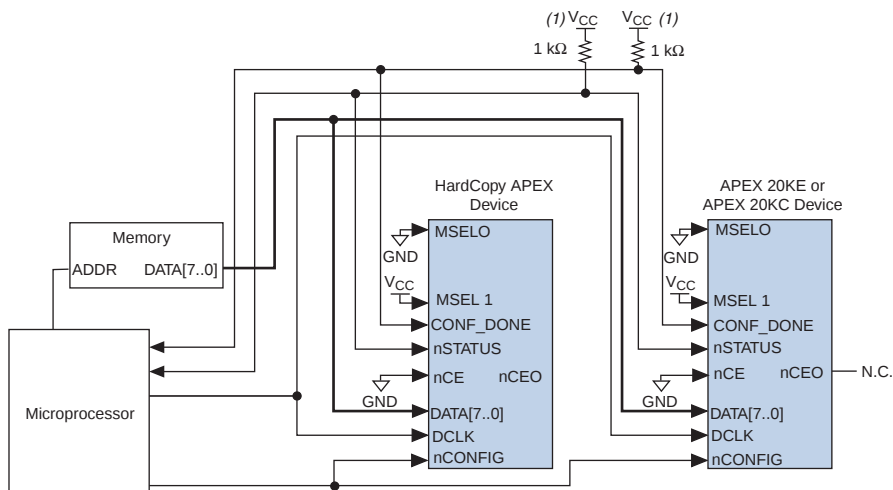


Figure 12–13. Replacement of First FPGA in the Chain With a HardCopy Series Device



Note to Figures 12–12 and 12–13:

(1) Connect the pull-up resistors to a supply that provides an acceptable input signal for all devices in the chain.

If the HardCopy series device is the first device in the chain as opposed to the second (as shown in [Figure 12–13](#)), you must take the following into consideration, depending on the HardCopy power-up option used.

- Instant on mode—The microprocessor program code must be modified to remove the configuration code relevant to the HardCopy series device. The microprocessor must delay sending the first configuration data word to the FPGA until the `nCEO` pin on the HardCopy series device is asserted. The microprocessor then loads the first configuration data word into the FPGA.
- Instant on after 50 ms mode—The boot-up time of the microprocessor must be greater than 50 ms. The HardCopy series device asserts the `nCEO` pin after the 50-ms delay which, in turn, enables the following FPGA. The microprocessor can send the first configuration data word to the FPGA after the FPGA is enabled.
- Emulation mode—This option should be used if the microprocessor code pertaining to the configuration of the above devices cannot be modified.

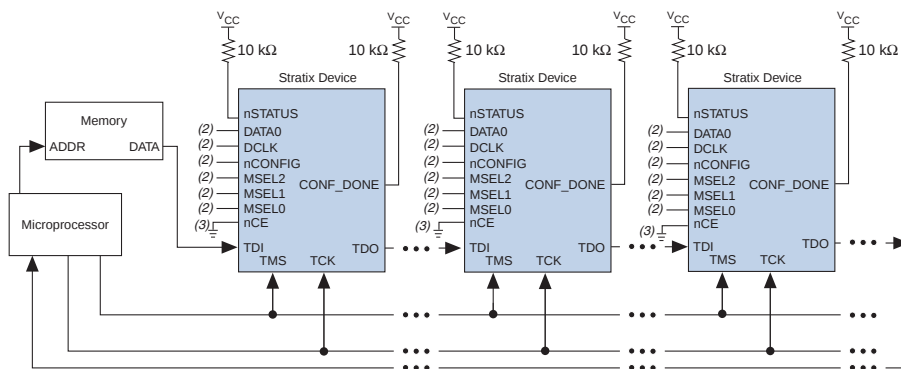
HardCopy Stratix Device Replacing FPGA Configured in a JTAG Chain

In this example, the circuit connectivity is maintained and there are no changes made to the board. The HardCopy series device can use either of the following power-up options when applicable.

- Instant on mode—Use the instant on power up mode if the microprocessor code can be modified so that it treats the HardCopy series device as a non-configurable device. The microprocessor can achieve this by issuing a `BYPASS` instruction to the HardCopy series device. With the HardCopy series device in `BYPASS` mode, the configuration data passes through it to the downstream FPGAs.
- Configuration emulation mode—Use the configuration emulation power up mode if the microprocessor code pertaining to the configuration of the above devices cannot be modified. HC1S80, HC1S60, and HC1S25 devices do not support this mode.

Figure 12–14 shows an example where there are multiple Stratix FPGAs. These devices are connected using the JTAG I/O pins for each device, and programmed using the JTAG port. An on-board microprocessor generates the configuration data.

Figure 12–14. Configuring FPGAs in a JTAG Chain Using a Microprocessor *Note (1)*

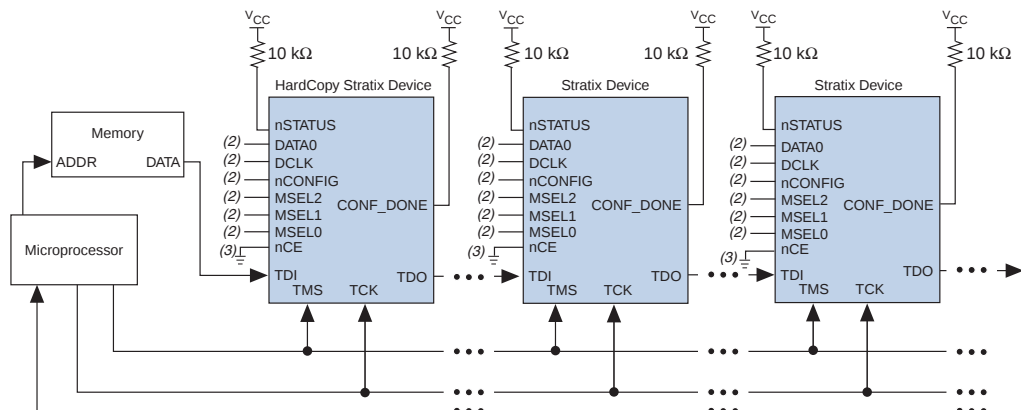


Notes to Figure 12–14:

- (1) Stratix II, Stratix, and APEX 20K devices can be placed within the same JTAG chain for device programming and configuration.
- (2) Connect the nCONFIG, MSEL0, MSEL1, and MSEL2 pins to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL0, MSEL1, and MSEL2 to ground. Pull DATA0 and DCLK to either high or low.
- (3) nCE must be connected to GND or driven low for successful JTAG configuration.

Figure 12–15 shows an example where the first Stratix device in the JTAG chain is replaced by a HardCopy Stratix device.

Figure 12–15. Replacement of the First FPGA in the JTAG Chain With a HardCopy Series Device *Note (1)*



Notes to Figure 12–15:

- (1) Stratix II, Stratix, and APEX 20K devices can be placed within the same JTAG chain for device programming and configuration.
- (2) Connect the nCONFIG, MSEL0, MSEL1, and MSEL2 pins to support a non-JTAG configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL0, MSEL1, and MSEL2 to ground. Pull DATA0 and DCLK to either high or low.
- (3) nCE must be connected to GND or driven low for successful JTAG configuration.

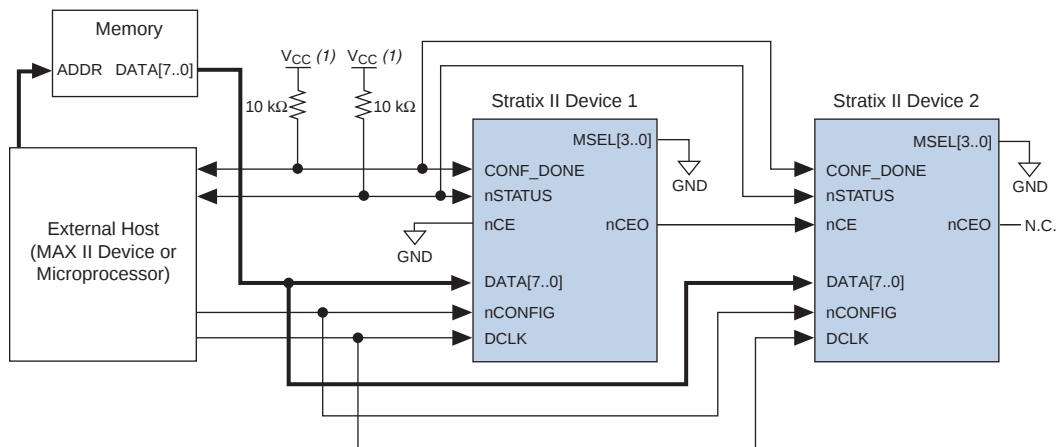
HardCopy II Device Replacing Stratix II Device Configured With a Microprocessor

When replacing a Stratix II FPGA with a HardCopy II device, the HardCopy II device can only use the instant on and instant on after 50 ms modes. This example does not require any changes to the board. However, the microprocessor code must be modified to treat the HardCopy II device as a non-configurable device.

Figure 12–16 shows an example with two Stratix II devices configured using a microprocessor or MAX® II device and the FPP configuration scheme.

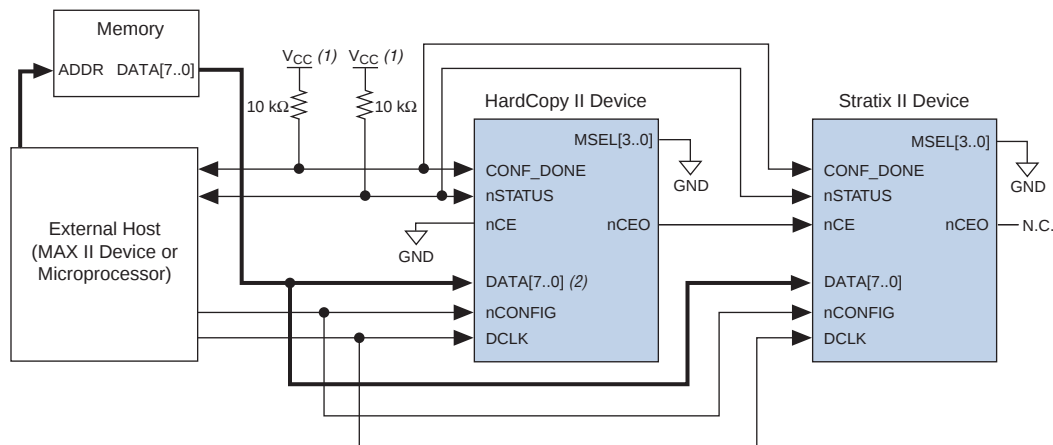


For more information on Stratix II configuration, refer to the *Configuration Handbook*.

Figure 12–16. Multiple-Device FPP Configuration Using a Microprocessor or MAX II Device**Note to Figure 12–16:**

- (1) Connect the pull-up resistor to a supply that provides an acceptable input signal for all devices in the chain. The V_{CC} voltage meets the I/O standard's V_{IH} specification on the device and the external host.

Figure 12–17 shows how the first Stratix II device is replaced by a HardCopy II device. In this case, the microprocessor code must be modified to send configuration data only to the second device (the Stratix II device) of the configuration chain. The microprocessor can only send this data after its nCE pin is asserted by the first device (the HardCopy II device).

Figure 12–17. Replacement of the First FPGA in the FPP Configuration Chain With a HardCopy Series Device**Notes to Figure 12–17:**

- (1) Connect the pull-up resistor to a supply that provides an acceptable input signal for all devices in the chain. The V_{CC} voltage meets the I/O standard's V_{IH} specification on the device and the external host.
- (2) The $DATA[7..0]$ pins are not used on the HardCopy II device, but they preserve the pin assignment and direction from the Stratix II device, allowing drop-in replacement.

Conclusion

HardCopy series devices can emulate a configuration sequence while maintaining the seamless migration benefits of the HardCopy methodology. Instant on mode, which is the simplest of the available options, provides ASIC-like operation at power on. This mode can be used in most cases without regard to the original FPGA configuration mode and without any hardware and/or software changes.

In some cases, however, a software revision and/or a board re-design may be necessary to guarantee that correct configuration data is sent to the remaining programmable devices. Such modifications are easily made in the early stages of the board design process if it is determined that one or more of the FPGAs will be replaced with an equivalent HardCopy series device. Board-design techniques like jumper connectors and 0-Ω resistors enable such modifications without the necessity to re-design the board.

The instant on after 50 ms mode is suitable in cases where a delay is necessary to accommodate the configuration device to become operational, or to allow one or more pre-determined events to be completed before the HardCopy series device asserts its `CONF_DONE` pin.

Finally, the emulation mode is the option to choose if software or hardware modifications are not possible. In such cases, the HardCopy series device co-exists with other FPGAs.

Document Revision History

Table 12–9 shows the revision history for this chapter.

Table 12–9. Document Revision History (Part 1 of 2)		
Date and Document Version	Changes Made	Summary of Changes
September 2008, v2.5	Updated chapter number and metadata.	—
June 2007, v2.4	Minor text edits.	—
December 2006 v2.3	Added revision history.	—
May 2006, v2.2	Updated Tables 20-1, 20-3, and 2-5.	—
March 2006, v2.1	- Formerly chapter 16. - Re-organized <i>HardCopy Power-Up Options</i> section to eliminate redundancy. - Updated Figures 20-1, 20-2, and 20-3. - Updated Tables 20-1 to 20-5, and Table 20-7. - Added <i>Power Up Options Summary When Designing With HardCopy Series Devices</i> section.	—
October 2005, v2.0	Moved from Chapter 15 to Chapter 16 in Hardcopy Series Device Handbook 3.2	—

Table 12–9. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
January 2005, v2.0	- Chapter title changed to <i>Power-Up Modes and Configuration Emulation in HardCopy Series Devices</i>. - Added HardCopy II device information. - Updated external resistor requirements depending on chip configuration. - Added reference to some control and option pins that carry over functions from the FPGA design and affect the HardCopy power up. - Updated information on which HardCopy devices do not support emulation mode. - Added Table 15–9 which lists what power up options are supported by FPGAs and their HardCopy counterpart. - Added “Replacing One FPGA With One HardCopy Series Device”, “Replacing One or More FPGAs With One or More HardCopy Series Devices in a Multiple-Device Configuration Chain”, and “Replacing all FPGAs with HardCopy Series Devices in a Multiple-Device Configuration Chain” sections, including Tables 15-10 and 15-11, highlighting power up recommendations for each HardCopy series family.	—
June 2003, v1.0	Initial release of Chapter 15, Power-Up Modes and Configuration Emulation in HardCopy Series Devices.	—



Section IV. HardCopy Design Center Migration Process

This section provides information on the software support for HardCopy® Stratix® devices.

This section contains the following:

- [Chapter 13, Back-End Design Flow for HardCopy Series Devices](#)
- [Chapter 14, Back-End Timing Closure for HardCopy Series Devices](#)

Revision History

Refer to each chapter for its own specific revision history. For information on when each chapter was updated, refer to the Chapter Revision Dates section, which appears in the complete handbook.

Introduction

This chapter discusses the back-end design flow executed by the HardCopy® Design Center when developing your HardCopy series device. The chapter is divided into two sections:

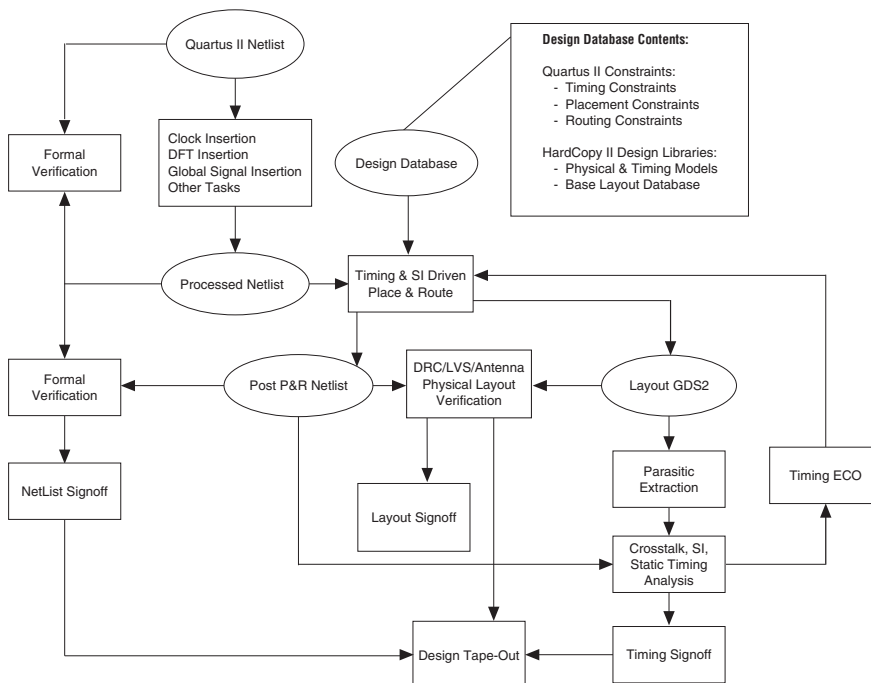
- HardCopy II Back-End Design Flow
- HardCopy Stratix® and HardCopy APEX™ Back-End Design Flow



For more information on the HardCopy II, HardCopy Stratix, and HardCopy APEX families, refer to the respective sections for these families in the *HardCopy Series Handbook*.

HardCopy II Back-End Design Flow

This section outlines the back-end design process for HardCopy II devices, which occurs in several steps. [Figure 13–1](#) illustrates these steps. The design process uses both proprietary and third-party EDA tools. The HardCopy II device design flow is different from that of previous HardCopy families (HardCopy Stratix and HardCopy APEX devices). The following sections outline these differences.

Figure 13–1. HardCopy II Back-End Design Flow

Device Netlist Generation

For HardCopy II designs, the Quartus® II software generates a complete Verilog gate-level netlist of your design. The HardCopy Design Center uses the netlist to start the migration process. HardCopy Stratix and HardCopy APEX designs use the SRAM Object file (.sof) to program the FPGA, as the primary starting point for generating the HardCopy device netlist.

HardCopy Stratix and HardCopy APEX designs use the .sof file to program the FPGA, as the primary starting point for generating the HardCopy device netlist. In addition to the Verilog gate level netlist and the .sof file, the Quartus II software generates additional information as part of the design database submitted to the HardCopy Design Center. This information includes timing constraints, placement constraints, global routing information, and much more. Generation of this database provides the HardCopy Design Center with the necessary information to complete the design of your HardCopy II device.

Design for Testability Insertion

The HardCopy Design Center inserts the necessary test structures into the HardCopy II Verilog netlist. These test structures include full-scan capable registers and scan chains, JTAG, and memory testing. After adding the test structures, the modified netlist is verified using third-party EDA formal verification software against the original Verilog netlist to ensure that the test structures have not broken your netlist functionality. The “[Formal Verification of the Processed Netlist](#)” section explains the formal verification process.

Clock Tree and Global Signal Insertion

Along with adding testability, the HardCopy Design Center adds an additional local layer of clock tree buffering to connect the global clock resources to the locally placed registers in the design. Global signals with high fan-out may also use dedicated Global Clock Resources built into the base layers of all HardCopy II devices. The HardCopy Design Center does local buffering.

Formal Verification of the Processed Netlist

After all design-for-testability logic, clock tree buffering, and global signal buffering are added to the processed netlist, the HardCopy Design Center uses third-party EDA formal verification software to compare the processed netlist with your submitted Verilog netlist generated by the Quartus II software. Added test structures are constrained to bypass mode during formal verification to verify that your design’s intended functionality was not broken.

Timing and Signal Integrity Driven Place and Route

Placement and global signal routing is principally done in the Quartus II software before submitting the HardCopy II design to the HardCopy Design Center. Using the Quartus II software, you control the placement and timing driven placement optimization of your design. The Quartus II software also does global routing of your signal nets, and passes this information in the design database to the HardCopy Design Center to do the final routing. After submitting the design to the HardCopy Design Center, Altera® engineers use the placement and global routing information provided in the design database to do final routing and timing closure and to perform signal integrity and crosstalk analysis. This may require buffer and delay cell insertion in the design through an engineering change order (ECO). The resulting post-place and route netlist is verified again with the source netlist and the processed netlist to guarantee that functionality was not altered in the process.

Parasitic Extraction and Timing Analysis

After doing placement and routing on the design by the HardCopy Design Center, it generates the `gds2` design file and extracts the parasitic resistance and capacitance values for timing analysis. Parasitic extraction uses the physical layout of the design stored in a `.gds2` file to extract these resistance and capacitance values for all signal nets in the design. The HardCopy Design Center uses these parasitic values to calculate the path delays through the design for static timing analysis and crosstalk analysis.

Layout Verification

When the Timing Analysis reports that all timing requirements are met, the design layout goes into the final stage of verification for manufacturability. The HardCopy Design Center performs physical Design Rule Checking (DRC), antenna checking of long traces of signals in the layout, and a comparison of layout to the design netlist, commonly referred to as Layout Versus Schematic (LVS). These tasks guarantee that the layout contains the exact logic represented in the place-and-route netlist, and the physical layout conforms to 90-nm manufacturing rules.

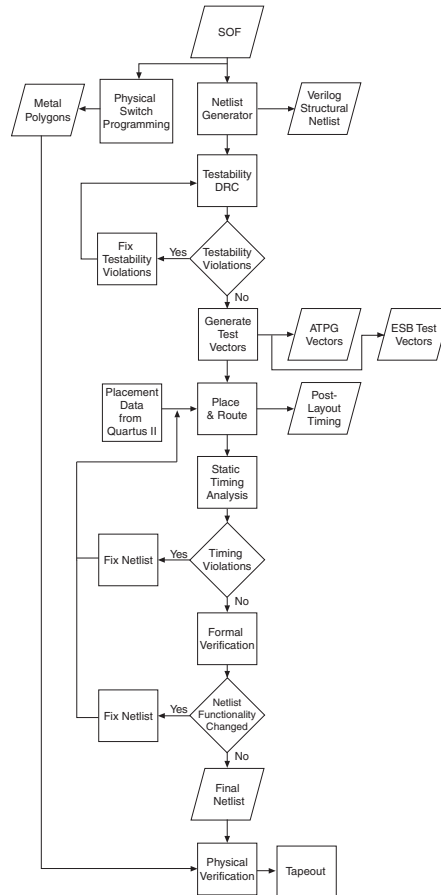
Design Signoff

The Altera HardCopy II back-end design methodology has a thorough verification and signoff process, guaranteeing your design's functionality. Signoff occurs after confirming the final place-and-route netlist functional verification, confirming layout verification for manufacturability, and the timing analysis reports meeting all requirements. After achieving all three signoff points, Altera begins the manufacturing of the HardCopy II devices.

HardCopy Stratix and HardCopy APEX Migration Flow

Design migration for HardCopy Stratix and HardCopy APEX devices occurs in several steps, outlined in this section and shown in [Figure 13–2](#). The migration process uses both proprietary and third-party EDA tools.

Figure 13–2. HardCopy Stratix and HardCopy APEX Migration Flow Diagram



Netlist Generation

For HardCopy Stratix and HardCopy APEX designs, Altera migrates the Quartus II software-generated **.sof** file to a Verilog HDL structural netlist that describes how the following structural elements are configured in the design and how each structural element is connected to other structural elements:

- Logic element (LE)
- Phase-locked loop (PLL)
- Digital signal processing (DSP) block
- Memory block
- Input/output element (IOE)

The information that describes the structural element configuration is converted into a physical coordinate format so that metal elements can be implemented on top of the pre-defined HardCopy series device-base array. Using the **.sof** file for netlist extraction helps ensure that the HardCopy series device contains the same functional implementation that was used in the FPGA version of the design.

Testability Audit

The Design Center performs an audit for testability violations when the Verilog HDL netlist is available. This audit ensures that all built-in scan chain structures will work reliably while testing the HardCopy series devices. Certain circuit structures, such as gated clocks, gated resets, oscillators, pulse generators, or other types of asynchronous circuit structures makes the performance of scan chain structures unreliable. During the testability audit, all such circuit structures are detected and disabled when the device is put into test mode.

Placement

Beginning with version 4.2, the Quartus II software supports all HardCopy series devices. The HardCopy Timing Optimization Wizard in the Quartus II software is used for HardCopy Stratix devices and generates placement information of the design when it is mapped to the HardCopy Stratix base array. This placement information is read in and directly used by the place-and-route tool during migration to the equivalent HardCopy Stratix device.



For more information on how to use the HardCopy Timing Optimization Wizard, refer to the *Quartus II Support for HardCopy Stratix Devices* chapter. For more information on Quartus II features for HardCopy II devices, refer to the *Quartus II Support for HardCopy II Devices* chapter.

To generate placement data, the Quartus II software uses the **.sof** file to generate the netlist, as described in [“Netlist Generation” on page 13–6](#). The netlist is then read into a place-and-route tool. The placement optimization is based on the netlist connectivity and the design’s timing constraints. The placement of all IOEs is fixed. After placement is complete, the Quartus II software generates the scan chain ordering information so the scan paths can be connected.

Test Vector Generation

Memory test vectors and memory built-in self-test (BIST) circuitry ensure that all memory bits function correctly. Automatic test pattern generation (ATPG) vectors test all LE, DSP, and IOE logic. These vectors ensure that a high stuck-at-fault coverage is achieved. The target fault coverage for all HardCopy Stratix devices is near 100%.

When the testability audit is successfully completed and the scan chains have been re-ordered, the Design Center can generate memory and ATPG test vectors. When test vector generations are complete, they are simulated to verify their correctness.

Routing

Routing involves generating the physical interconnect between every element in the design. At this stage, physical design rule violations are fixed. For example, nodes with large fan-outs need to be buffered. Otherwise, these signal transition times are too slow, and the device’s power consumption increases. All other types of physical design rule violations are also fixed during this stage, such as antenna violations, crosstalk violations, and metal spacing violations.

Extracted Delay Calculation

Interconnect parasitic capacitance and resistance information is generated after the routing is complete. This information is then converted into a Standard Delay File (**.sdf**) with a delay calculation tool, and timing is generated for minimum and maximum delays.

Static Timing Analysis and Timing Closure

The design timing is checked and corrected after place and route using the post-layout generated **.sdf** file. Setup time violations are corrected in two ways. First, extra buffers can be inserted to speed up slow signals. Second, if buffer insertion does not completely fix the setup violation, the placement can be re-optimized.

Setup time violations are rare in HardCopy II and HardCopy Stratix devices because the die sizes are considerably smaller than the equivalent Stratix II and Stratix devices. Statistically, the interconnect loading and distance is much smaller in HardCopy Stratix devices, so the device operates at a higher clock frequency. Hold-time violations are fixed by inserting delay elements into fast data paths.

As part of the timing analysis process, crosstalk analysis is also performed to remove any crosstalk effects that could be encountered in the device after it has been manufactured. This ensures signal integrity in the device resulting in proper functionality and satisfactory performance.

After implementing all timing violation corrections in the netlist, the place and route is updated to reflect the changes. This process is repeated until all timing violations are removed. Typically, only a single iteration is required after the initial place and route. Finally, static functional verification is tested after this stage to double-check the netlist integrity.

Formal Verification

After any change to the netlist, you must verify its integrity through static functional verification (or formal verification) techniques. These techniques show whether two versions of a design are functionally identical when certain constraints are applied. For example, after test fixes, the netlist must be logically equivalent to the netlist state before test fixes, when the test mode is disabled. This technique does not rely on any customer-supplied functional simulation vectors. Altera uses third-party formal verification software to confirm that the back-end implementation matches the netlist generated from the FPGA's .sof programming file.

Physical Verification

Before manufacturing the metal customization layers, the physical programming information must be verified. This stage involves cross-checking for physical design rule violations in the layout database, and also checking that the circuit was physically implemented correctly. These processes are commonly known as running design rule check and layout-versus-schematic verification.

Manufacturing

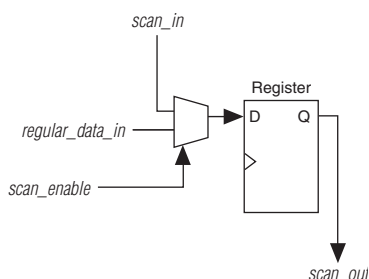
Metallization masks are created to manufacture HardCopy series devices. After manufacturing, the parts are tested using the test vectors that were developed as part of the implementation process.

Testing

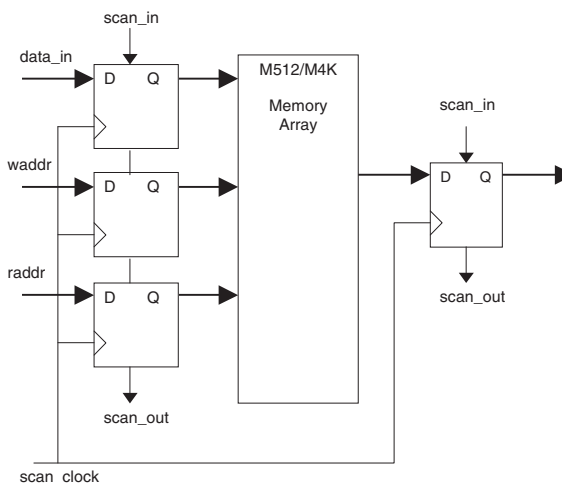
HardCopy series devices are fully tested as part of the manufacturing process. Testing does not require user-specific simulation vectors, because every HardCopy series device utilizes full scan path technology. This means that every node inside the device is both controllable and observable through one or more of the package pins of the device. The scan paths, or “scan chains,” are exercised through ATPG. This ensures a high-confidence level in the detection of all manufacturing defects.

Every register in the HardCopy series device belongs to a scan chain. Scan chains are test features that exist in ASICs to ensure that there is access to all internal nodes of a design. With scan chains, defective parts can be screened out during the manufacturing process. Scan chain registers are constructed by combining the original FPGA register with a 2-to-1 multiplexer. In normal user mode, the multiplexer is transparent to the user. In scan mode, the registers in the device are connected into a long-shift register so that automatic test pattern generation vectors can be scanned into and out of the device. Several independent scan chains exist in the HardCopy series device to keep scan chain lengths short, and are run in parallel to keep tester time per device short. Figure 13–3 shows a diagram of a scan register.

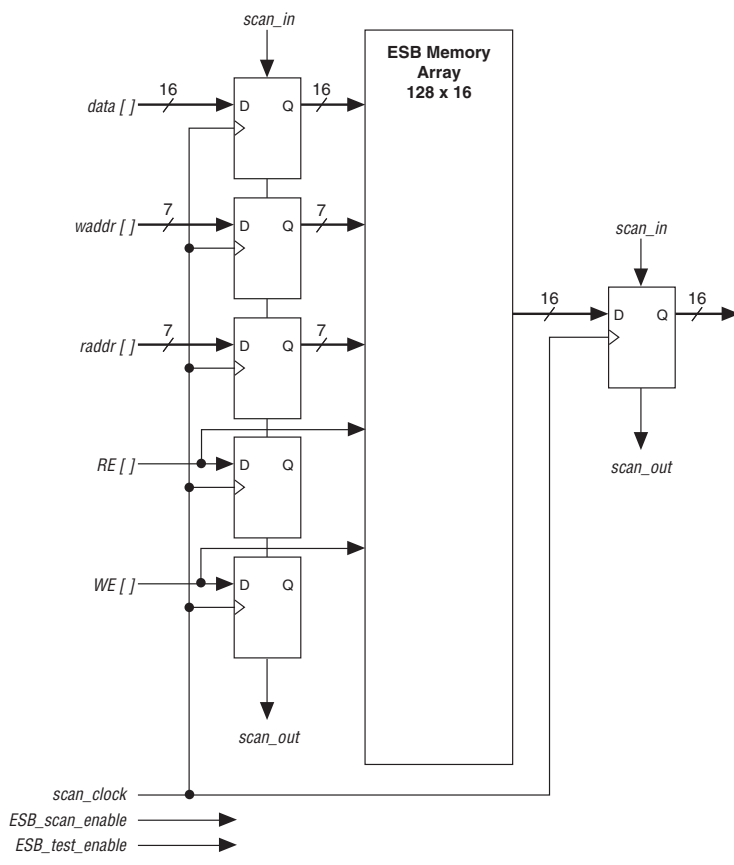
Figure 13–3. HardCopy Stratix Scan Chain Circuitry



In addition to the scan circuitry (Figure 13–3), which is designed to test all LEs and IOEs, both M512 and M4K blocks (Figure 13–4) have the same scan chain structure so that all bits inside the memory array are tested for correct operation. The M512 and M4K RAM bits are tested by scanning data into the M512 and M4K blocks’ `data_in`, write address (`waddr`), and read address (`raddr`) registers. After each vector has been scanned into the HardCopy Stratix device, a write enable (`WE`) pulse is generated to write the data into the M512 and M4K blocks. A read enable (`RE`) pulse is also generated to read data out of the M512 and M4K blocks. The data read back from the M512 and M4K blocks are scanned out of the device via the `data_out` registers. Figure 13–4 shows the M512 and M4K blocks’ scan chain connectivity.

Figure 13–4. HardCopy Stratix M512 and M4K Block Scan Chain Connectivity

For HardCopy APEX devices, every embedded system block (ESB) contains dedicated test circuitry so that all bits inside the memory array are tested for correct operation. Access to the ESB memory is also facilitated through scan chains. The ESB also offers an ESB test mode in which the ESB is reconfigured into a 128×16 RAM block. In this mode, data is scanned into the ESB I/O registers and written into the ESB memory. For ESBs configured as product-term logic or ROM, the write-enable signal has no effect on the ESB memory array data. When the test mode is disabled (the default), the ESB reverts to the desired user functionality. [Figure 13–5](#) shows the ESB test mode configuration.

Figure 13–5. HardCopy APEX ESB Test Mode Configuration

PLLs and M-RAM blocks are tested with BIST circuitry and test point additions. All test circuitry is disabled once the device is installed into the end user system so that the device then behaves in the expected normal functional mode.

Unused Resources

Unused resources in a customer design still exist in the HardCopy base. However, these resources are configured into a “parked” state. This is a state where all input pins of an unused resource are tied off to V_{CC} or GND so that the resource is in a low-power state. This is achieved using the same metal layers that are used to configure and connect all resources used in the design.

Conclusion

The HardCopy series back-end design methodology ensures that your design seamlessly migrates from your prototype FPGA to a HardCopy device. This methodology, matched with Altera's unique FPGA prototyping and migration process, provides an excellent way for you to develop your design for production.



For more information about how to start building your HardCopy series design, contact your Altera Field Applications Engineer.



For more information on HardCopy products and solutions, refer to the *HardCopy Series Handbook*.

Document Revision History

Table 13–1 shows the revision history for this chapter.

Table 13–1. Document Revision History		
Date and Document Version	Changes Made	Summary of Changes
September 2008, v1.4	Updated chapter number and revision history.	—
June 2007, v1.3	Minor text edits.	—
December 2006 v1.2	Added revision history.	—
March 2006	Formerly chapter 13; no content change.	—
October 2005 v1.1	• Graphic updates • Minor edits	—
January 2005 v1.0	Initial release of Chapter 13, Back-End Design Flow for HardCopy Series Devices.	—

Introduction

Back-end implementation of HardCopy® series devices meet design requirements through a timing closure process similar to the methodology used for today's standard cell ASICs.

The Quartus® II software provides a pre-layout estimation of your HardCopy design performance and then the Altera® HardCopy Design Center uses industry leading EDA software to complete the back-end layout and extract the final timing results prior to tape-out.



For more information on the HardCopy back-end design flow, refer to the *HardCopy Series Back-End Design Flow* chapter in the *HardCopy Series Device Handbook*.

This chapter describes how Altera ensures that HardCopy series devices meet their required timing performance.

Timing Analysis of HardCopy Prototype Device

You should perform timing analysis on the FPGA prototype implementation of the design before migrating to HardCopy. For HardCopy II designs, timing analysis should also be performed after successfully fitting the design in a HardCopy II device with Quartus II software. Timing analysis determines whether the design's performance meets the required timing goals.

The timing analysis must be done for both setup and hold time checks on all design paths, including internal paths and input and output paths. Measuring these parameters against performance goals ensures that the FPGA design functions as planned in the end-target system.



For more information on timing analysis of Altera devices, refer to the *Timing Analysis* section in volume 3 of the *Quartus II Handbook*.

After the FPGA design is stabilized, fully tested in-system and satisfies the HardCopy series design rules, the design can be migrated to a HardCopy series device. Altera performs rigorous timing analysis on the HardCopy series device during its implementation, ensuring that it meets the required timing goals. Because the critical timing paths of the HardCopy version of a design may be different from the corresponding paths in the FPGA version, meeting the required timing goals constrained in the Quartus II software is particularly important. Additional

performance gains are design dependent, and the percentage of performance improvement can be different for each clock domain of your design.

Timing differences between the FPGA design and the equivalent HardCopy series device can exist for several reasons. While maintaining the same set of features as the corresponding FPGA, HardCopy series devices have a highly optimized die size to make them as small as possible. Because of the customized interconnect structure that makes this optimization possible, the delay through each signal path is different from the original FPGA design.

Cell Structure

Meeting system timing goals in an ASIC design can be very challenging and can easily consume many months of engineering effort. The slower development process exists because, in today's silicon technology (0.18 μm , 0.13 μm , and 90 nm), the delay associated with interconnect dominates the delay associated with the transistors used to make the logic gates. Consequently, ASIC performance is sensitive to the physical placement and routing of the logic blocks that make up the design.

HardCopy II

HardCopy II devices use timing constraints to drive placement and routing of logic into the fabric of HCells. Each Stratix II Adaptive Look-up Table (ALUT) is implemented in HCell Macros in the HardCopy II device. HCell Macros are pre-defined and characterized libraries built out of HCells. The Quartus II software performs the placement and global routing of all HCell Macros and this information is forward-annotated to the HardCopy Design Center for final back-end implementation and timing closure.

HardCopy Stratix, HardCopy APEX

HardCopy Stratix® and HardCopy APEX™ are structurally identical to their respective FPGA counterparts. There is no re-synthesis or library re-mapping required. Since the interconnect lengths are much smaller in the HardCopy series device than they are in the FPGA, the place-and-route engine compiling the HardCopy series design has a considerably less difficult task than it does in an equivalent ASIC development. Coupled with detailed timing constraints, the place-and-route is timing driven.

Clock Tree Structure

The following section describes the clock tree structure for the HardCopy device family.

HardCopy II

HardCopy II devices offer a fine-grained architecture of HCells which are used to build HCell Macros for standard logic functions. The pre-built metal layers of HardCopy II devices contain the same global clock tree resources as those available in Stratix II devices, though they are smaller in HardCopy II devices because of the difference in die size. The top levels of the dedicated global clock networks in HardCopy II are pre-routed in the non-custom metal layers. The lowest level of clock tree buffering and routing is done using custom metal routing. Local buffering can be done using HCell Macros to fix any clock skew issues. HCell Macros are used to create registers, and local custom routing is needed to connect the clock networks to these HCell Macro registers. These tasks are performed as part of the HardCopy Design Center process.

HardCopy Stratix

HardCopy Stratix devices have the same global clock tree resources as Stratix FPGA devices. The construction of non-customizable layers of silicon minimizes global clock tree skew. HardCopy Stratix devices with clock trees using global clock resources have smaller clock insertion delay than Stratix FPGA devices because the HardCopy Stratix devices have a smaller die area. The use of clock tree synthesis to build small localized clock trees using the existing buffer resources in HardCopy Stratix devices automatically implements clock trees using fast regional clock resources in Stratix FPGA devices.

HardCopy APEX

The HardCopy APEX device architecture is based on the APEX 20KE and APEX 20KC devices. The same dedicated clock trees (CLK [3 . . 0]) that exist in APEX 20KE and APEX 20KC devices also exist in the corresponding HardCopy APEX device. These clock trees are carefully designed and optimized to minimize the clock skew over the entire device. The clock tree is balanced by maintaining the same loading at the end of each point of the clock tree, regardless of what resources (logic elements [LEs], embedded system blocks [ESBs], and input/output elements [IOEs]) are used in any design. The insertion delay of the HardCopy APEX dedicated clock trees is marginally faster than in the corresponding APEX 20KE or APEX 20KC FPGA device because of the smaller footprint of the HardCopy device silicon. This difference is less than 1 ns.

Because there is a large area overhead for the global signals that may not be used on every design, the FAST bidirectional pins (FAST[3..0]) do not have dedicated pre-built clock or buffer trees in HardCopy APEX devices. If any of the FAST signals are used as clocks, the place-and-route tool synthesizes a clock tree after the placement of the design has occurred. The skew and insertion delay of these synthesized clock trees is carefully controlled, ensuring that the timing requirements of the design are met. You can also use the FAST signals as high fan-out reset or enable signals. For these cases, skew is usually less important than insertion delay. To reiterate, a buffer tree is synthesized after the design placement.

The clock or buffer trees that are synthesized for the FAST pins are built out of special cells in the HardCopy APEX base design. These cells do not exist in the FPGA, and they are used in the HardCopy APEX design exclusively to meet timing and testing goals. They are not available to make any logical changes to the design as implemented in the FPGA. These resources are called the strip of auxiliary gates (SOAG). There is one strip per MegaLAB™ structure in HardCopy APEX devices. Each SOAG consists of a number of primitive cells, and there are approximately 10 SOAG primitive cells per logic array block (LAB). Several SOAG primitives can be combined to form more complex logic, but the majority of SOAG resources are used for buffer tree, clock tree, and delay cell generation.



For detailed information on the HardCopy APEX series device architecture, including SOAG resources, refer to the *HardCopy APEX Device Family Data Sheet* section in volume 1 of the *HardCopy Series Handbook*.

Importance of Timing Constraints

After capturing the information, Altera directly checks all timing of the HardCopy series device before tape-out occurs. It is important to constrain the FPGA and HardCopy devices for the exact timing requirements that need to be achieved. Timing violations seen in the Quartus II project or in the HardCopy Design Center migration process must be fixed or waived prior to the design being manufactured.

Correcting Timing Violations

After generating the customized metal interconnect for the HardCopy series device, Altera checks the design timing with a static timing analysis tool. The static timing analysis tool reports timing violations and then the HardCopy Design Center corrects the violations.

Hold-Time Violations

Because the interconnect in a HardCopy series device is customized for a particular application, it is possible that hold-time (tH) violations exist in the HardCopy series device after place-and-route occurs. A hold violation exists if the sum of the delay in the clock path between two registers plus the micro hold time of the destination register is greater than the delay of the data path from the source register to the destination register. The following equation describes this relationship.

$$tH \text{ slack} = \text{data delay} - \text{clock delay} - \mu tH$$

If a negative slack value exists, a hold-time violation exists. Any hold-time violation present in the HardCopy series design database after the interconnect data is generated is removed by inserting the appropriate delay in the data path. The inserted delay is large enough to guarantee no hold violation under fast, low-temperature, high-voltage conditions.

An Example HardCopy APEX Hold-Time Violation Fix

Table 14–1 shows an example report of a Synopsys PrimeTime static timing analysis of a HardCopy APEX design. The first report shows that the circuit has a hold-time violation and a negative slack value. The second result shows the timing report for the same path after fixing the hold violation. Part of the HardCopy implementation process is to generate the instance and cell names shown in these reports. The physical location of those elements in the device determines the generation of the names.

Table 14–1. HardCopy APEX Static Timing Analysis Before Hold-Time Violation Fix

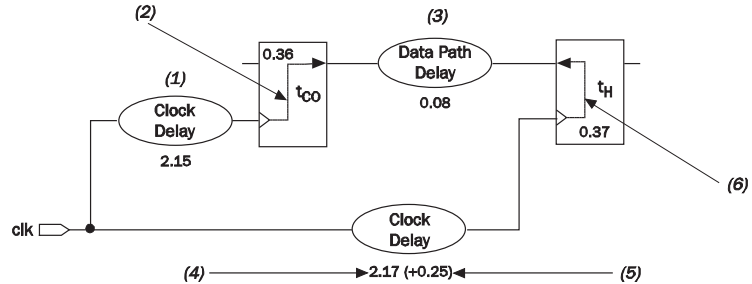
Startpoint: GR23_GC0_L19_LE1/um6 (falling edge-triggered flip-flop clocked by CLK0')			
Endpoint: GR23_GC0_L20_LE8/um6 (falling edge-triggered flip-flop clocked by CLK0')			
Path Group: CLK0			
Path Type: min			
Point	Incr	Path	Reference Point (1)
clock CLK0' (fall edge)	0.00	0.00	
clock network delay (propagated)	2.15	2.15	(1)
GR23_GC0_L19_LE1/um6/clk (c1110)	0.00	2.15 f	(2)
GR23_GC0_L19_LE1/um6/regout (c1110)	0.36 *	2.52 r	(2)
GR23_GC0_L19_LE1/REGOUT (c1000_2d7a8)	0.00	2.52 r	(2)
GR23_GC0_L20_LE8/LUTD (c1000_56502)	0.00	2.52 r	(3)
GR23_GC0_L20_LE8/um1/datad (indsim)	0.01 *	2.52 r	(3)
GR23_GC0_L20_LE8/um1/ndsim (indsim)	0.01 *	2.53 f	(3)
GR23_GC0_L20_LE8/um5/ndsim (mxcascout)	0.00 *	2.53 f	(3)
GR23_GC0_L20_LE8/um5/cascout	0.06 *	2.59 f	(3)
GR23_GC0_L20_LE8/um6/dcout (c1110)	0.00 *	2.59 f	(3)
data arrival time		2.59	
clock CLK0' (fall edge)	0.00	0.00	
clock network delay (propagated)	2.17	2.17	(4)
clock uncertainty	0.25	2.42	(5)
GR23_GC0_L20_LE8/um6/clk (c1110)		2.42 f	(6)
library hold time	0.37 *	2.79	
data required time		2.79	
data arrival time		2.59	
data required time		-2.79	
slack (VIOLATED)		-0.20	

Note to Table 14–1:

- (1) This column does not exist in the actual report. It is included in this document to provide corresponding reference points to Figure 14–1.

Figure 14–1 shows the circuit described by the Table 14–1 static timing analysis report.

Figure 14–1. Circuit With a Hold-Time Violation



Placing the values from the static timing analysis report into the hold-time slack equation results in the following:

$$t_H \text{ slack} = \text{data delay} - \text{clock delay} - \mu t_H$$

$$t_H \text{ slack} = (2.15 + 0.36 + 0.08) - (2.17 + 0.25) - 0.37$$

$$t_H \text{ slack} = -0.20 \text{ ns}$$

This result shows that there is negative slack in this path, meaning that there is a hold-time violation of 0.20 ns.

After fixing the hold violation, the timing report for the same path is re-generated (Table 14–2). The netlist changes are in *bold italic* type.

Table 14–2. HardCopy APEX Static Timing Analysis After Hold-Time Violation Fix

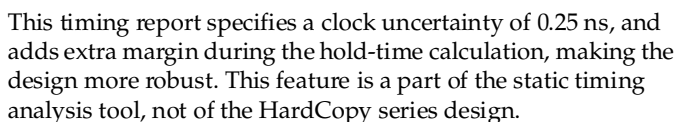
Startpoint: GR23_GC0_L19_LE1/um6
(falling edge-triggered flip-flop clocked by CLK0')
Endpoint: GR23_GC0_L20_LE8/um6
(falling edge-triggered flip-flop clocked by CLK0')
Path Group: CLK0
Path Type: min
Static Timing Analysis After Hold-Time Violation Fix

Point	Incr	Path	Reference Point (1)
clock CLK0' (fall edge)	0.00	0.00	(1)
clock network delay (propagated)	2.15	2.15	(1)
GR23_GC0_L19_LE1/um6/c1k (c1110)	0.00	2.15 f	(2)
GR23_GC0_L19_LE1/um6/regout (c1110)	0.36 *	2.52 r	(2)
GR23_GC0_L19_LE1/REGOUT (c1000_2d7a8)	0.00	2.52 r	(2)
thc_916/A (de105)	0.01 *	2.52 r	(3)
thc_916/Z (de105)	0.25 *	2.78 r	(3)
GR23_GC0_L20_LE8/LUTD (c1000_56502)	0.00	2.78 r	(3)
GR23_GC0_L20_LE8/um1/datad (indsim)	0.01 *	2.78 r	(3)
GR23_GC0_L20_LE8/um1/ndsim (indsim)	0.01 *	2.79 f	(3)
GR23_GC0_L20_LE8/um5/ndsim (mxcascout)	0.00 *	2.79 f	(3)
GR23_GC0_L20_LE8/um5/cascout (mxcascout)	0.06 *	2.85 f	(3)
GR23_GC0_L20_LE8/um6/dcout (c1110)	0.00 *	2.85 f	(3)
data arrival time		2.85	
clock CLK0' (fall edge)	0.00	0.00	
clock network delay (propagated)	2.17	2.17	(4)
clock uncertainty	0.25	2.42	(5)
GR23_GC0_L20_LE8/um6/c1k (c1110)		2.42 f	(6)
library hold time	0.37 *	2.79	
data required time		2.79	
data arrival time		2.85	
data required time		-2.79	
slack (MET)		0.06	

Note to Table 14–2:

- (1) This column does not exist in the actual report. It is included in this document to provide corresponding reference points to Figure 14–2.

$$t_H \text{ slack} = + 0.06 \text{ ns}$$



Setup-Time Violations

A setup violation exists if the sum of the delay in the data path between two registers plus the micro setup time (t_{SU}) of the destination register is greater than the sum of the clock period and the clock delay at the destination register. The following equation describes this relationship:

$$t_{SU} \text{ slack} = \text{clock period} + \text{clock delay} - (\text{data delay} + \mu t_{SU})$$

If there is a negative slack value, a setup-time violation exists. Several potential mechanisms can cause a setup-time violation. The first is when the synthesis tool is unable to meet the required timing goals. However, a HardCopy series design does not rely on any re-synthesis to a new cell library; synthesis results are generated as part of the original FPGA design, meaning that the HardCopy implementation of a design uses exactly the same structural netlist as its FPGA counterpart. For example, if you used a particular synthesis option to ensure that a particular path only contain a certain number of logic levels, the HardCopy series design contains exactly the same number of logic levels for that path. Consequently, if the FPGA was free of setup-time violations, no setup-time violations will occur in the HardCopy series device due to the netlist structure.

The second mechanism that can cause setup-time violations is differing placement of the resources in the netlist for the HardCopy series device compared to the original FPGA. This scenario is extremely unlikely as the place-and-route tool used during the HardCopy implementation performs timing-driven placement. In extreme cases, some manual placement modifications are necessary. The placement is performed at the LAB and ESB level, meaning that the placement of logic cells inside each LAB is fixed, and is identical to the placement of the FPGA. IOEs have fixed placement to maintain the pin and package compatibility of the original FPGA.

The third, and most likely, mechanism for setup-time violations occurring in the HardCopy series device is a signal with a high fan-out. In the FPGA, high fan-out signals are buffered by large drivers that are integral parts of the programmable interconnect structure. Consequently, a signal that was fast in the FPGA can be initially slower in the HardCopy version. The place-and-route tool detects these signals and automatically creates buffer trees using SOAG resources, ensuring that the heavily loaded, high fan-out signal is fast enough to meet performance requirements.

An Example HardCopy APEX Setup-Time Violation Fix

Table 14–3 shows the timing report for a path in a HardCopy APEX design that contains a high fan-out signal before the place-and-route process. Table 14–4 shows the timing report for a path that contains a high fan-out signal after the place-and-route process. Before the place-and-route process, there is a large delay on the high fan-out net driven by the pin GR12_GC0_L2_LE4/REGOUT. This delay is due to the large capacitive load that the pin has to drive. Figure 14–3 shows the timing report information.

Table 14–3. HardCopy APEX Timing Report Before Place-and-Route Process

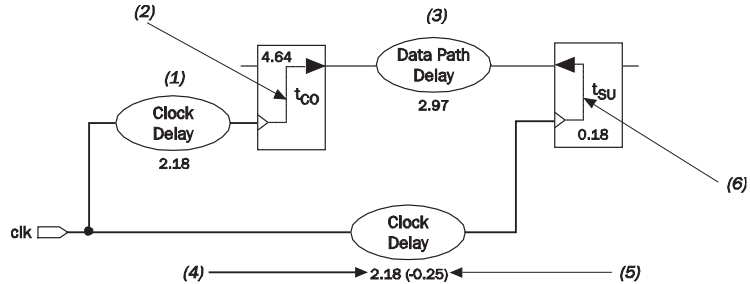
Startpoint: GR12_GC0_L2_LE4/um6 (falling edge-triggered flip-flop clocked by clkx')			
Endpoint: GR4_GC0_L5_LE2/um6 (falling edge-triggered flip-flop clocked by clkx')			
Path Group: clkx			
Path Type: max			
Point	Incr	Path	Reference Point (1)
clock clkx' (fall edge)	0.00	0.00	(1)
clock network delay (propagated)	2.18	2.18	(1)
GR12_GC0_L2_LE4/um6/clk (c1110)	0.00	2.18 f	(2)
GR12_GC0_L2_LE4/um6/regout (c1110)			(2)
GR12_GC0_L2_LE4/REGOUT (c1000_7f802) <-			(2)
GR4_GC0_L5_LE0/LUTC (c1000_0029a)			(3)
GR4_GC0_L5_LE0/um4/ltb (lt53b)	2.36	9.18 f	(3)
GR4_GC0_L5_LE0/um5/cascout (mxascout)	0.07	9.24 f	(3)
GR4_GC0_L5_LE0/um2/COMBOUT (icombout)	0.09	9.34 r	(3)
GR4_GC0_L5_LE0/COMBOUT (c1000_0029a)	0.00	9.34 r	(3)
GR4_GC0_L5_LE2/LUTC (c1000_0381a)	0.00	9.34 r	(3)
GR4_GC0_L5_LE2/um4/ltb (lt03b)	0.40	9.73 r	(3)
GR4_GC0_L5_LE2/um5/cascout (mxascout)	0.05	9.78 r	(3)
GR4_GC0_L5_LE2/um6/dcout (c1110)	0.00	9.78 r	(3)
data arrival time		9.79	(3)
clock clkx' (fall edge)	7.41	7.41	
clock network delay (propagated)	2.18	9.59	(4)
clock uncertainty	-0.25	9.34	(5)
GR4_GC0_L5_LE2/um6/clk (c1110)		9.34 f	
Point	Incr	Path	Reference Point (1)
library setup time	-0.18	9.16	(6)
data required time		9.16	
data required time		9.16	
data arrival time		-9.79	
slack (VIOLATED)		-0.63	

Note to Table 14–3:

- (1) This column does not exist in the actual report. It is included in this document to provide corresponding reference points to Figure 14–3.

Figure 14–3 shows the circuit that Table 14–3 static timing analysis report describes.

Figure 14–3. Circuit That Has a Setup-Time Violation



The timing numbers in this report are based on pre-layout estimated delays.

Placing the values from the static timing analysis report into the set-up time slack equation, results in the following.

$$t_{SU} \text{ slack} = \text{clock period} + \text{clock delay} - (\text{data delay} + \mu t_{SU})$$

$$t_{SU} \text{ slack} = 7.41 + (2.18 - 0.25) - (2.18 + 4.64 + 2.97 + 0.18)$$

$$t_{SU} \text{ slack} = -0.63 \text{ ns}$$

This result shows that there is negative slack for this path, meaning that there is a setup-time violation of 0.63 ns.

After place-and-route, a buffer tree is constructed on the high fan-out net and the setup-time violation is fixed. Table 14–4 shows the timing report for the same path. The changes to the netlist are in **bold italic** type.

Figure 14–4 shows more information on this timing report.

Table 14–4. HardCopy APEX Timing Report After the Place-and-Route Process

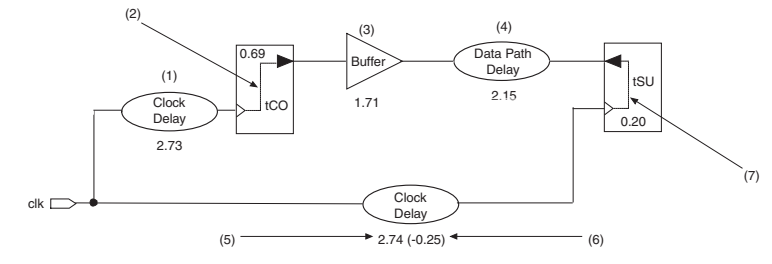
Startpoint: GR12_GC0_L2_LE4/um6 (falling edge-triggered flip-flop clocked by clkx')			
Endpoint: GR4_GC0_L5_LE2/um6 (falling edge-triggered flip-flop clocked by clkx')			
Path Group: clkx			
Path Type: max			
Point	Incr	Path	Reference Point (1)
clock clkx' (fall edge)	0.00	0.00	
clock network delay (propagated)	2.73	2.73	(1)
GR12_GC0_L2_LE4/um6/clk (c1110)	0.00	2.73 f	(2)
GR12_GC0_L2_LE4/um6/regout (c1110)	0.69 *	3.42 r	(2)
GR12_GC0_L2_LE4/REGOUT (c1000_7f802) <-	0.00	3.42 r	(2)
N1188_iv06_1_0/Z (iv06)	0.06 *	3.49 f	(3)
N1188_iv06_2_0/Z (iv06)	0.19 *	3.68 r	(3)
N1188_iv06_3_0/Z (iv06)	0.12 *	3.80 f	(3)
N1188_iv06_4_0/Z (iv06)	0.10 *	3.90 r	(3)
N1188_iv06_5_0/Z (iv06)	0.08 *	3.97 f	(3)
N1188_iv06_6_2/Z (iv06)	1.16 *	5.13 r	(3)
GR4_GC0_L5_LE0/LUTC (c1000_0029a)	0.00	5.13 r	(4)
GR4_GC0_L5_LE0/um4/lrb (lt53b)	1.55 *	6.68 f	(4)
GR4_GC0_L5_LE0/um5/cascout (mxscascout)	0.06 *	6.74 f	(4)
GR4_GC0_L5_LE0/um2/COMBOUT (icombout)	0.09 *	6.84 r	(4)
GR4_GC0_L5_LE0/COMBOUT (c1000_0029a)	0.00	6.84 r	(4)
GR4_GC0_L5_LE2/LUTC (c1000_0381a)	0.00	6.84 r	(4)
GR4_GC0_L5_LE2/um4/lrb (lt03b)	0.40 *	7.24 r	(4)
GR4_GC0_L5_LE2/um5/cascout (mxscascout)	0.05 *	7.28 r	(4)
GR4_GC0_L5_LE2/um6/dcout (c1110)	0.00 *	7.28 r	(4)
<u>data arrival time</u>		7.28	(4)
Point	Incr	Path	Reference Point (1)
clock clkx' (fall edge)	7.41	7.41	
clock network delay (propagated)	2.74	10.15	(5)
clock uncertainty	-0.25	9.90	(6)
GR4_GC0_L5_LE2/um6/clk (c1110)		9.90 f	
library setup time	-0.20 *	9.70	(7)
<u>data required time</u>		9.70	
data required time		9.70	
<u>data arrival time</u>		-7.28	
<u>slack (MET)</u>		2.42	

Note to Table 14–4:

- (1) This column does not exist in the actual report. It is included in this document to provide corresponding reference points to Figure 14–4.

The GR12_GC0_L2_LE4/REGOUT pin now has the loading on it reduced by the introduction of several levels of buffering (in this case, six levels of inverters). The inverters have instance names similar to N1188_iv06_1_0, and are of type iv06, as shown in the static timing analysis report. As a result, the original setup-time violation of -0.63 ns turned into a slack of $+2.42$ ns, meaning the setup-time violation is fixed. Figure 14-4 illustrates the circuit that the static timing analysis report shows. The buffer tree (buffer) is shown as a single cell.

Figure 14-4. Circuit Post Place-and-Route



Placing the values from the static timing analysis report into the setup-time slack equation, results in the following:

$$t_{SU} \text{ slack} = \text{clock period} + \text{clock delay} - (\text{data delay} + \mu t_{SU})$$

$$t_{SU} \text{ slack} = 7.41 + (2.74 - 0.25) - (2.73 + 0.69 + 1.71 + 2.15 + 0.20)$$

$$t_{SU} \text{ Slack} = +2.42 \text{ ns}$$

This result shows that there is positive slack for this path, meaning that there is now no setup-time violation.

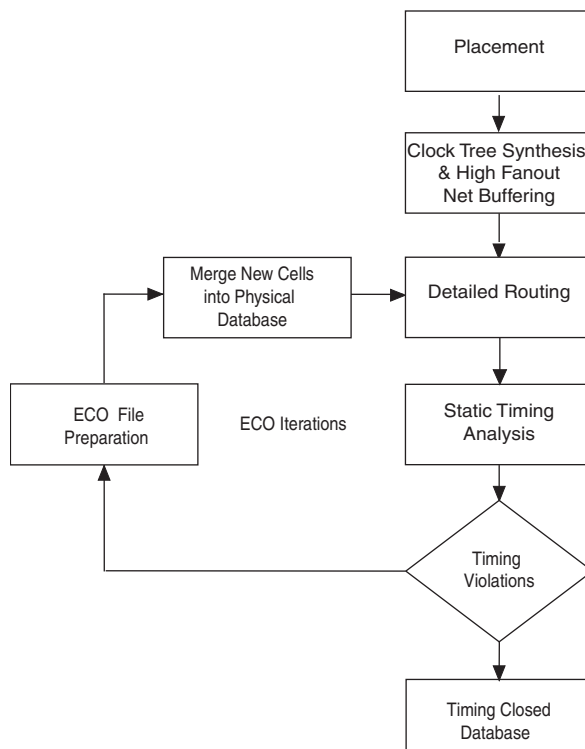
Timing ECOs

In an ASIC, small incremental changes to a design database are termed engineering change orders (ECOs). In the HardCopy series design flow, ECOs are performed after the initial post-layout timing data is available.

You run static timing analysis on the design, which generates a list of paths with timing violations. An automatically updated netlist reflects changes that correct these timing violations (for example, the addition of delay cells to fix hold-time violations). After the netlist update, the updated place-and-route database reflects the netlist changes. The impact to this database is made minimal by maintaining all of the pre-existing placement and routing, and only changing the routing of newly inserted cells.

The parasitic (undesirable, but unavoidable) resistances and capacitances of the customized interconnect are extracted, and are used in conjunction with the static timing analysis tool to re-check the timing of the design. Detected crosstalk violations on signals are fixed by adding additional buffering to increase the setup or hold margin on victim signals. In-line buffering and small buffer tree insertion is done for signals with high fan-out, high transition times, or high capacitive loading. Figure 14–5 shows this flow in more detail.

Figure 14–5. ECO Flow Diagram



The back-end flow in HardCopy produces the final sign-off timing for your HardCopy device. The Quartus II software produces the timing report for HardCopy based on a global route and does not factor in exact physical parasitics of the routed nets, nor does it factor in the crosstalk effect that neighboring nets can have on interconnect capacitance.

Conclusion

It is critical that you fully constrain your HardCopy series design for timing. Although HardCopy series devices are functionally equivalent to their FPGA prototype companion, they have inevitable timing differences. Fully constrained timing paths are a cornerstone of designing for HardCopy series devices.

Consult with Altera if you have questions on what areas to concentrate your efforts in to achieve timing closure within the Quartus fitter for HardCopy design submission.

Document Revision History

Table 14–5 shows the revision history for this chapter.

Table 14–5. Document Revision History (Part 1 of 2)

Date and Document Version	Changes Made	Summary of Changes
September 2008, v2.4	Updated chapter number and metadata.	—
June 2007, v2.3	Minor text edits.	—
December 2006 v2.2	• Minor updates for the Quartus II software version 6.1.0 • Moved <i>Checking the HardCopy Series Device Timing</i> section to Chapter 7	A minor update to the chapter, due to changes in the Quartus II software version 6.1 release; also, <i>Checking the HardCopy Series Device Timing</i> section moved to Chapter 7.
March 2006	Formerly chapter 17; no content change.	—
October 2005 v2.1	• Moved <i>Chapter 16 Back-End Timing Closure for Hardcopy Series Devices</i> to Chapter 17 in <i>HardCopy Series Device Handbook</i> release 3.2 • Updated graphics • Minor edits	—

Table 14–5. Document Revision History (Part 2 of 2)

Date and Document Version	Changes Made	Summary of Changes
January 2005 v2.0	• Chapter title changed to <i>Back-End Timing Closure for HardCopy Series Devices</i>. • Sizes of silicon technology updated in “Timing Closure” on page 17–2. • HardCopy® Stratix® and HardCopy APEX™ equivalence to their respective FPGA is updated on page 17–2. • Stratix II migration added. • Updated Table 17–2 on page 17–12. • Updated last paragraph in “Timing ECOs” on page 17–18.	—
June 2003 v1.0	Initial release of Chapter 17, Back-End Timing Closure for HardCopy Series Devices.	—