Cyclone III Design Guidelines

AN-466-2.2 Application Note

This document summarizes the various aspects of the Cyclone III device, and highlights the Quartus® II software features that you should consider when you are designing with the Cyclone III devices. With good design practice and clear understanding of the design flow of the Cyclone III device, your design flow will be much easier.

This document covers the following topics:

Device Selection

This section provides the information for you to consider when selecting a Cyclone III device for your system.

Device Feature Consideration

The Cyclone III device family offers from over 5,000 logic elements (LEs) to nearly 120,000 LEs and is suitable for a wide range of applications.

Always choose a device that has more LEs than the estimated count for your design requirements:

- You will have extra LEs if you want to upgrade or expand your design.
- Consider having additional LEs and memories for debugging purposes.
- Additional resources allow more flexibility with the Quartus II software for optimizing placement and routing for maximum performance, lower power consumption or both.

For the number of LEs and other resource counts of the Cyclone III devices, refer to the *Cyclone III Device Family Overview* chapter in the *Cyclone III Device Handbook.*

I/O Pin Count, Package Offering and Vertical Migration

The Cyclone III device family offers up to a maximum of 535 user I/O pins.

Depending on your application and board layout, you can select the available package options:

- Quad Flat Pack (QFP)
- FineLine Ball Grid Array (FBGA) with a 1.0 mm ball pitch
- Ultra FBGA package (UBGA) with an 0.8 mm ball pitch—the smallest in the Cyclone III device family and saves board space.

Cyclone III devices support vertical migration within the same package. For a given package, the devices across different densities have the same locations for the power pins, configuration pins and dedicated pins. This allows future upgrade or changes to your Cyclone III design without having to change the board layout as you can replace the Cyclone III device in your board with another Cyclone III device of a different density.

For best results, you can specify the migration device before compiling your initial design with the Quartus II software to ensure that only pins that are available in the same locations on both devices are used in the design for seamless migration to a larger or smaller device.

The number of differential channels may vary across device density.

For information about the number of user I/O pins and package offerings for the Cyclone III family across device density, refer to the *Cyclone III Device Family Overview* chapter in the *Cyclone III Device Handbook*.

For more information about the number of differential channels available for different densities and packages of the Cyclone III device family, refer to the *High-Speed Differential Interfaces in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Speed Grade

Device speed grade affects the timing closure of the device. Depending on the density and package, Cyclone III devices are available in three different speed grades:

- –6 (fastest),
- –7 and,
- –8

When migrating to a device of different speed grade, check the timing report from the timing analysis to ensure that there is no timing violation between different blocks within the Cyclone III device, between the Cyclone III device and other devices on the board.

Always design with a sufficient timing margin so that your design can work on devices of different speed grades. Generally, the difference of one speed grade can mean a core f_{MAX} or I/O performance difference of up to 20%.

Early System Planning

It is important to know your system requirement and what your Cyclone III device can offer at the early stage of the design cycle.

Early Power Estimation

As FPGAs have increased in logic capacity and performance, the power consumption must be accurately estimated for appropriate power supply, decoupling and thermal solution planning. The power supply must be able to provide enough current for the device operation and the thermal solution must be able to cool the device junction temperature to within the specification.

Altera provides a power estimation tool called the Early Power Estimator (EPE) to help you estimate the power consumption of your design during the system planning phase. The EPE allows you to estimate the power consumption, current drawn from the power supply and device junction temperature based on the device resources that you are going to use in your design, along with the information about the ambient temperature, heat sink, air flow and board thermal model.

You can either enter the design information manually into the spreadsheet or import a power estimator file of a fully or partially completed design from the Quartus II software. After importing a file, you can edit some of the input parameters such as the ambient temperature, airflow, clock frequency and toggle percentage to suit your system requirements.

For more information about the EPE and ways generate and import the power estimator file, refer to the *PowerPlay Early Power Estimator User Guide for Cyclone III FPGAs.*

I/O Support

This section discusses the general Cyclone III device I/O support. Before starting with the I/O planning, take some time to review the support offered by the Cyclone III I/Os.

Selectable I/O Standards

Cyclone III devices support a wide range of industry I/O standards, including single-ended, voltage-referenced and differential I/O standards. Selection factors are driven by performance versus cost. Table 1 simplifies the selection choice for each I/O signaling type.

Table 1. Selection Criteria for Each I/O Signaling Type

I/O Signaling Type	Selection Criteria	
	Performance	**Cost**
Single-ended	Slow speed rail-to-rail interface, limited by large voltage swing and noise.	Fairly low, unless reflection causes signal integrity concerns, whereby termination is required.
Voltage-reference (single-ended and differential) (1)	Reduced Simultaneous Switching Output (SSO) effects from large number of pins changing levels at the same time. Improved logic transition rate with reduced voltage swing and minimized noise caused by reflections due to termination requirements of the I/O standard.	High, with extra components due to termination requirement and additional clean reference voltage, V_{TT}.
Differential (2), (3)	Superior speed (up to 840 Mbps), lower swing voltage and increased noise immunity with common mode noise rejection capability.	Low, with reduced physical traces and I/O resources by implementing serialization/deserialization (SERDES) logic to replace parallel data transmission.

Notes to Table 1:

(1) Differential voltage-referenced standards can only be used for clocking. These I/O standards are only supported on the `GCLK` and `PLL_OUT` pins.

(2) Side I/O banks support dedicated differential buffers. Top and bottom I/O banks support differential signaling with an additional resistor network on the transmitter side.

(3) For details on the guidelines and considerations of the high-speed **LVDS** interface implementation in a Cyclone III device, refer to *AN 479: Design Guidelines for Implementing LVDS Interfaces in Cyclone Series Devices.*

Flexible I/O Banks

Simultaneous support of various I/O standards is possible with efficient I/O groupings in banks. Each bank must be supplied with one V_{CCIO} level. Each I/O bank is powered up individually by the `VCCIO` pins of that particular bank and is independent of the V_{CCIO} of other I/O banks. Eight I/O banks are offered for increased flexibility to be used with multi-voltage systems.

An output buffer does not meet the configured I/O standard specification if the V_{CCIO} is out of the recommended operating range specified in the Cyclone III device datasheet for the I/O standard.

Although there can only be one V_{CCIO} voltage per I/O bank, Cyclone III devices permit additional input signaling capabilities.

For more information about the acceptable input and output levels, refer to the *I/O Features in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Each Cyclone III I/O bank has a `VREF` bus to accommodate voltage-referenced I/O standards. Multiple `VREF` pins within an I/O bank feed the common `VREF` bus. Each bank can only have a single V_{CCIO} voltage level and a single V_{REF} voltage level at a given time. V_{REF} is used as a reference voltage for voltage-referenced inputs (**SSTL** and **HSTL** I/O standards) to determine logic threshold. Because of this, it is important for `VREF` to be noise-free.

Follow pad placement guidelines in the *I/O Features in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook* to minimize noise coupling onto the reference voltage.

Voltage deviation on the `VREF` pin can affect the threshold sensitivity for the input operation. If a voltage referenced input is not utilized for a V_{REF} group, the `VREF` pin is released automatically by the Quartus II software for use as an I/O pin.

However, if the `VREF` pin is used as an I/O pin, it will have a higher pin capacitance due to the power bus loading effects. The I/O edge rate will be slower than regular I/O pins, which can affect the timing of the signal. For best results, do not use `VREF` pins as I/O pins when the `VREF` pins are part of a common bus or pin group that requires similar timing characteristics. Similarly, do not use the `VREF` pins for clocks.

External Memory Interface

Cyclone III devices support interfaces to the DDR2 SDRAM, DDR SDRAM, and QDRII SRAM. Depending on device density and package, specific side of I/O bank may support up to x36 mode of memory interface. Supported modes in the Cyclone III devices are x8, x9, x16, x18, x32, and x36 modes. Use the Pin Planner tool to assist you in determining and making pin assignments for the memory interface.

In general, choose the top or bottom I/O banks instead of the side I/O banks to achieve a higher clock rate for the external memory interfaces. The Cyclone III devices support external memory interfaces up to 200 MHz.

For more information about the DDR/DDR2 and QDRII pads placement, refer to the *Cyclone III Device I/O Feature* chapter in the *Cyclone III Device Handbook.*

For more information about the recommended design flow to implement the DDR2 SDRAM memory interface with Cyclone III devices, refer to *External Memory Interface Handbook Volume 5, Section I. ALTMEMPHY Design*.

For a complete table of the maximum clock rate support across all speed grades for every memory standard, refer to the *External Memory Interfaces in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Pin-Out Files

The Cyclone III pin-out files contain information about the location for all the pins of the devices, according to package:

- For the I/O pins, you can find out which I/O bank and the V_{REF} group the pins belong to.
- The pin-out files contain the description for the dedicated and multi-purpose pins.
- The pin-out files help designer to determine the I/O pins to be used when creating the design as well as when designing the board.

Apart from I/O pins, the location of dedicated and multi-purpose pins is also important during the board design stage.

To obtain device pin-outs for Cyclone III devices, refer to the *Device Pin-Outs* page of the *Literature* section of the Altera website (www.altera.com).

Planning and Selecting Configuration Scheme

Select your device configuration method early to allow system and board designers to determine if any additional devices are required for your system. Your board layout depends on the configuration method that you plan to use for the programmable device, because different schemes require different connections. For board design guidelines related to configuration pins, refer to "Board Design Considerations" on page 16.

Cyclone III devices offer configuration data decompression and real-time remote system upgrades to save configuration memory space and time. Support for these configuration features varies depending on your configuration scheme.

Cyclone III devices also include optional configuration pins and a reconfiguration option that you must determine at the start and set up in the Quartus II software, so that you have all the information required for your board and system design. For the Quartus II software settings and pins related to the configuration options that affect your board and system design, refer to "Design and Compilation" on page 28.

Depending on the device densities and package options, you can configure Cyclone III devices using one of the following five configuration schemes:

- Active serial (AS)
- Active parallel (AP)
- Passive serial (PS)
- Fast passive parallel (FPP)
- Joint Test Action Group (JTAG)

A selection of configuration scheme with different configuration voltage standards is selected by driving the Cyclone III device's `MSEL` pins either high or low. Depending on the `MSEL` pin settings, you can either select a fast power-on reset (POR) time or a standard POR time. The fast POR time supports fast wake-up time applications, where it may be necessary for a device to wake up quickly to begin operation.

In Cyclone III devices, the supported configuration schemes differ for different device densities and package options. For example, the EP3C16 device's E144 package offers the AS, PS, and JTAG schemes while its U484 package offers the AS, PS, FPP, AP, and JTAG schemes.

For more information about the supported configuration schemes, refer to the Cyclone III devices' Configuration Center on the Altera website.

For complete information about the following items, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*:

- Supported configuration schemes across device densities and package options
- Configuration voltage standards and POR time
- Methods to execute the required configuration schemes and all the necessary option pin settings, including the `MSEL` pin settings

For new user of Altera device configuration schemes, the available choice of configuration schemes and methods in which the configuration schemes can be set up may be overwhelming. In general, Altera configuration schemes are categorized into the following configuration schemes:

- Active configuration scheme—provides a one-chip solution configuration to build a simple system, where only a serial configuration device is required for the AS configuration scheme or a parallel flash memory for the AP configuration scheme.
- Passive configuration scheme—provides a two-chip solution configuration. If your system already contains an external intelligent host or a microprocessor, you can utilize it with a flash memory to perform the passive configuration scheme such as the PS or FPP.

For more information on the configuration scheme selection, refer to the Configuration Center on the Altera website. This web page includes a link to the configuration guidelines that provide an overview of Altera FPGA configuration schemes and a general comparison of the schemes to guide you in choosing the one that best suits your design requirements.

All configuration schemes use a configuration device, a download cable or an external controller (for example, a MAX® II device or microprocessor):

- The AS and AP schemes use an external flash memory, such as serial configuration device or a supported flash memory, respectively.
- The PS and JTAG schemes use either an external controller or a download cable.
- The FPP scheme uses an external controller.

When choosing the configuration scheme that best suits your system requirements, you may also want to consider the configuration time. Configuration time varies for different configuration schemes and depends on the configuration file size, configuration data width, frequency of the driving clock, and flash access time.

The AP configuration scheme offers the fastest configuration time among the supported configuration schemes in a Cyclone III device. The speedup in the configuration time is mainly due to the 16-bit wide parallel data bus which is used to retrieve data from the flash. However, a 24-bit address bus is required to connect the Cyclone III device to the address bus of the flash memory. In other words, 40 pins of your Cyclone III device and the flash memory would be occupied for the address bus and data transfer.

Comparatively, the FPP takes only 8 pins of your Cyclone III device and the flash memory device for the data transfer and does not require an address bus. Although the configuration of the FPP is not relatively as fast as the configuration of the AP, this scheme provides faster configuration time compared to any serial configuration scheme such as the AS and PS.

For the active configuration schemes such as the AS and AP, the `DCLK` is an output from the Cyclone III device that provides the timing for the configuration interface. The maximum `DCLK` frequency for the AS and AP is 40 MHz.

In the PS and FPP configuration, the DCLK is the clock input that is used to clock data from an external source into the target Cyclone III device. Data is latched into the device on the rising edge of the DCLK. In other words, you are able to monitor the DCLK frequency to control the configuration time in the passive configuration scheme by varying the clock from the external source. For the PS configuration, the maximum DCLK frequency is 133 MHz and for the FPP configuration, the maximum DCLK frequency is 100 MHz.

Table 2 simplifies the general comparison of the supported configuration schemes in Cyclone III.

Table 2. General Comparison for Supported Configuration Schemes in Cyclone III

Configuration Scheme	Serial/Parallel Configuration	Configuration Method	Width of Data bus (in bits)	Width of Address bus (in bits)	Relative configuration time	Supported configuration feature
Active Serial (AS)	Serial	Serial Configuration (EPCS) device	1	—	Moderate	Decompression, Remote System Upgrade
Active Parallel (AP)	Parallel	Supported parallel flash memory	16	24	Fast	Remote System Upgrade
Passive Serial (PS)	Serial	Download cable	1	—	Moderate	Decompression
		MAX II or microprocessor with flash memory	1	—	Moderate	Decompression
Fast Passive Serial (FPP)	Parallel	MAX II or microprocessor with flash memory	8	—	Fast	—
(JTAG)	Serial	Download cable	1	—	Slow	—
		Microprocessor with flash memory	1	—	Slow	—

Configuration Devices

You can use the Altera's serial configuration devices (EPCS) in the AS configuration scheme. Supported commodity parallel flash families are used in the AP configuration scheme. Check whether the configuration device supports the configuration bitstream file size of your Cyclone III device. In the PS and FPP configuration schemes, you can use a MAX II device or a microprocessor with a flash memory configuration method. In the AS and PS configuration schemes, you can use the compression feature to reduce the configuration file size of a large Cyclone III device.

For more information on the EPCS devices, refer to the *Serial Configuration (EPCS) Devices Datasheet*.

For more information on the supported families for the commodity parallel flash, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

Download Cables

The Quartus II programmer supports configuring Cyclone III devices directly using PS or JTAG interfaces via Altera programming download cables. You can download design changes directly to the device with the Altera download cables, making prototyping easy and enabling you to make multiple design iterations in quick succession. You can use the same download cable to program configuration devices on the board and use JTAG debugging tools such as the SignalTap® II Logic Analyzer.

For more information about using Altera's download cables, refer to the following documents:

- *USB-Blaster Download Cable User Guide*
- *ByteBlaster II Download Cable User Guide*
- *EthernetBlaster Communications Cable User Guide*

Serial FlashLoader

When you use an AS configuration scheme, the conventional method to program an EPCS device is via the active serial (AS) programming interface.

However, with the Serial FlashLoader (SFL), you can program the EPCS device in-system via the JTAG interface. The SFL solution reduces the effort to have a separate programming interface for the Cyclone III and EPCS device because you are able to configure the Cyclone III and EPCS device using the same JTAG interface.

The SFL for the Cyclone III device is available with the Quartus II software, version 7.1 and higher. SFL Megafunction is available with the Quartus II software, version 6.0 and higher.

For more information about the SFL, refer to *AN 370: Using the Serial FlashLoader with the Quartus II Software.*

MAX II Parallel Flash Loader

If your system already contains the common flash interface (CFI) flash memory, you can use it for the Cyclone III device configuration storage as well. The parallel flash loader (PFL) feature with MAX II devices allows you to program the CFI flash memory devices through the JTAG interface. It also provides the logic to control the configuration from the flash memory device to the Cyclone III device and supports compression to reduce the size of your configuration data.

Both the PS and FPP configuration modes are supported using the PFL feature. If you choose this configuration method, you should check the list of supported flash devices early in your system design cycle and plan accordingly.

For more information about the MAX II PFL, refer to *Parallel Flash Loader Megafunction User Guide*.

FPGA-Based Parallel Flash Loader

The AP configuration scheme configures the Cyclone III device using the Intel StrataFlash® Embedded Memory P30 flash family and the Intel StrataFlash Embedded Memory P33 flash family, which are two industry standard flash families.

If your system contains a parallel flash device, you can use it to store configuration data. The parallel flash device reduces the configuration time through the parallel interface and provides a higher memory capacity to store configuration data. However, the parallel flash device does not support direct device programming through JTAG.

Using the FPGA-based PFL, you can use the Cyclone III JTAG interface to perform in-system programming for the parallel flash device. The PFL enables you to program the flash device indirectly before configuring the Cyclone III with the AP configuration scheme.

For more information about the FPGA-based PFL, refer to *AN 478: Using FPGA-Based Parallel Flash Loader with the Quartus II Software.*

Configuration Features

This section describes the Cyclone III device's configuration features, specifically the data compression and remote system upgrade (RSU) and how they affect your design process.

For the supported features in each configuration scheme, refer to Table 2 on page 8.

Data Compression

If you enable data compression, the Quartus II software generates configuration files with compressed configuration data. This compressed file reduces the storage requirements in the configuration device or flash memory and decreases the time needed to transmit the bitstream to the Cyclone III device. The time required by a Cyclone III device to decompress a configuration file is less than the time needed to transmit the configuration data to the device.

Cyclone III devices support decompression in the PS (when you use a MAX II device/microprocessor + flash) and AS configuration schemes. The Cyclone III decompression feature is not available in the AP, FPP, or JTAG configuration scheme.

To enable compression before compilation, enable **Generate compressed bitstreams** on the **Configuration** tab of the **Device & Pin Options** dialog box. You can also enable compression when creating programming files from the **Convert Programming Files** window. Open the **Properties** dialog box for the programming file, and turn on **Compression**.

For more information about data compression, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

Remote System Upgrade

Cyclone III devices support remote update in the AS and AP configuration schemes. You can enable or disable remote update mode with an option setting in the Quartus II software. You can implement remote update in conjunction with real-time decompression of configuration data if you must save configuration memory space in the serial configuration device with AS configuration. To implement the remote system upgrade interface, you can use the altremote_update megafunction.

For more information about making remote system upgrade, refer to the *Configuration, Design Security, and Remote System Upgrades in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

For more information about the altremote_update megafunction, refer to the *Remote System Upgrade (ALTREMOTE_UPDATE) Megafunction User Guide.*

Phase-Locked Loop

Cyclone III PLLs have a number of advanced features available including clock multiplication and division, phase shifting, programmable duty cycles, clock switchover, PLL cascading, PLL dynamic reconfiguration, dynamic phase shifting, spread-spectrum clocking, external clock outputs and control signals.

For more information about the PLL features, refer to the *Clock Networks and PLLs in Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Previously, PLLs in the Cyclone series devices' FPGAs were designed to be configured for a specific input frequency. The newly added voltage-controlled oscillator (VCO) range detector in the Cyclone III PLLs, together with the dynamic phase reconfiguration and PLL reconfiguration, enable the support for advanced display applications where the PLL input frequency may not be known ahead of time or may change.

For information about the VCO range detector and the reference design to support the unknown F_{ref} video applications, refer to *Supporting Unknown F_{REF} Video Applications with PLLs* white paper.

PLL Applications

In general, you can use the PLLs for frequency synthesis and clock management. The PLL usage helps to compensate the clock delay for large clock networks to improve performance. You can also use the PLLs to recover clocks and clean jitter caused by the transmission signal line. The programmability of the FPGA makes it easy to change the parameters such as the frequency, bandwidth and duty cycle.

Before you start your design using a PLL, make sure you define the correct applications for the PLL.

Define PLL Settings

Based on the features that are available in the Cyclone III device that you are targeting, define the settings of the PLL based on the system requirement. The following guidelines help you to decide on the settings.

PLL Input and Output Frequencies

Based on your system requirement, define the input frequencies and output frequencies for the PLL. Cyclone III PLLs can operate in a particular bandwidth. If the input frequencies and output frequencies do not meet that bandwidth, you can cascade the PLLs. Cyclone III PLL clock inputs can only be fed by dedicated clock input pins or outputs from another PLL. PLL clock inputs cannot be fed by internally generated logic or clock source that originates from the general purpose I/O pins. PLL clock outputs can drive the dedicated clock output pin or global clock networks.

To find out whether the input frequencies and output frequencies can be implemented in one PLL, enter your settings when instantiating the altpll megafunction through the Quartus II software.

To obtain the input and output clock frequency range specifications for the different device speed grades, refer to the *Cyclone III Device Datasheet* chapter in the *Cyclone III Device Handbook*

Number of Clock Inputs

Cyclone III PLLs can have a maximum of two clock inputs, where only one clock input functions at a time. You need two clock inputs for your PLL for applications such as clock redundancy or dual clock domain. Clock redundancy application ensures there is a back-up input clock in case the current clock is not present. A dual clock application enables the PLL to change to another input clock frequency during operation. This feature is called clock switchover.

Cyclone III devices have automatic and manual clock switchover. Automatic switchover enables the PLL to change to another clock input after the current clock input becomes unavailable. Manual clock switchover enables the user to control the switch of the input clocks.

If you do not use the PLL for these applications, one clock input is sufficient.

You can enable the clock switchover feature from the altpll megafunction.

Number of Clock Outputs

Cyclone III PLLs can have a maximum of five clock outputs (c0-c4). You can connect the c0 clock output to the dedicated external clock output pin (recommended as this minimizes the clock jitter), normal user I/O or dedicated global clock network. The c1, c2, c3, and c4 clock outputs can be connected to the user I/O pins or dedicated global clock networks.

Clock Input and Output I/O Standard

Dedicated clock input pins can support all I/O standards supported by the Cyclone III device, except **PPDS**, **RSDS**, and **mini-LVDS**. Dedicated external clock output pins can support all I/O standards supported by the Cyclone III device.

For more information about the I/O standard support, refer to the *I/O Features in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

PLL Design

After defining the requirements, these are the items that you must consider while designing the PLL according to your application.

Selecting the Right Compensation Mode

Table 3 shows the four compensation modes that Cyclone III devices support.

Table 3. Compensation Modes

Modes	Description
Source-synchronous Mode	Select this mode if you want to maintain the same phase relationship between the clock and data from the input pins to the I/O element (IOE) register driven by the PLL.
No Compensation Mode	Select this mode if you want a better jitter performance at the PLL output.
Normal Mode	Select this mode if you want the clock edge at an IOE or Logic Element (LE) register driven by the PLL to be phase-aligned with the clock signal at the clock input pin.
Zero Delay Buffer (ZDB) Mode	Select this mode if you want the external clock output pin to be phase-aligned with the clock input pin for zero delay through the device.

PLL Dynamic Reconfiguration and Phase Stepping

You can use this feature if you want to change the PLL settings without needing to reconfigure the entire device. Two main applications for this feature change between two output frequencies to suit the design and to adjust the clock-to-output (t_{CO}) delays in real time without the need to regenerate a configuration file with new PLL settings.

You can enable the dynamic reconfiguration feature by instantiating the altpll megafunction. Then, you can ease the usage of this feature by instantiating the altpll_reconfig megafunction.

To obtain the maximum reconfiguration clock frequency (`fscanclk`) and typical time required to reconfigure the scan chains (`tconfigpll`), refer to the *Cyclone III Device Datasheet* in the *Cyclone III Device Handbook*.

PLL Control Signals

There are three main control signals, which are essential during PLL usage, depending on the application.

Table 4. Compensation Modes

Signal	Description
`areset`	When enabled, this control pin is an input reset pin for each PLL. You must enable this control pin if you enable the dynamic reconfiguration and clock switchover feature. You should also enable the automatic reset upon loss of lock feature using the altpll megafunction so that the PLL resets automatically when the locked signal goes low. Refer to the *Cyclone III Device Datasheet* in the *Cyclone III Device Handbook* to obtain the minimum pulse width on the `areset` signal (t_{ARESET}).
`locked`	When enabled, this control pin is an output pin for each PLL. When it is high, the PLL has locked onto the reference clock and the PLL clock outputs are operating at the desired phase and frequency. You should create a design to monitor this signal to ensure that the system using the PLL clock outputs reacts according to the validity of clock output.
`pfdena`	When enabled, this control pin is an input pin for each PLL. You should enable this pin if your system requires a certain clock frequency from the PLL even though the input clock is disabled, so that it will have time to store its current setting before shutting down.

Planning for On-Chip Debugging

The Quartus II software includes various debugging tools. The following is the list of debugging tools and their use as well as the requirement for using the tools. Using these tools in your design flow is optional, but is recommended.

SignalProbe Incremental Routing

The SignalProbe Incremental Routing feature allows you to route internal signals to I/O pins without affecting the routing of the original design. Starting with a fully routed design, you can select and route the signals for debugging to either previously reserved or currently unused I/O pins. To use the SignalProbe feature, you must have additional I/Os in your device for the signals to be routed to. These I/Os should be routed out on the board in order for you to be able to probe the signals. You can use the SignalProbe feature to monitor synchronous or asynchronous signals.

For more information about ways to use the SignalProbe feature for debugging, refer to the *Quick Design Debugging Using SignalProbe* chapter in the *Quartus II Handbook.*

SignalTap II Logic Analyzer

The SignalTap II Logic Analyzer captures and displays real-time behavior of your FPGA design's internal and I/O signals while the design is running at full speed, without the need for additional I/O pins or external probes.

You can use the SignalTap II Logic Analyzer by instantiating the SignalTap II Logic Analyzer megafunction and includes that in your design or by creating a SignalTap II Logic Analyzer (**.stp**) file so that you do not need to modify your design. The SignalTap II Logic Analyzer is suitable for capturing synchronous signals and needs a clock signal from your design to control data acquisition. The SignalTap II Logic Analyzer requires JTAG connection and communicates with the device through an Altera download cable. Additional LEs and the M9K memory are needed as well.

To minimize changes to your Cyclone III device's performance and reduce compilation time when you use the SignalTap II Logic Analyzer, back-annotate your design and use incremental compilation.

For more information about how to use the SignalTap II Logic Analyzer for debugging, refer to the *Design Debugging Using the SignalTap II Logic Analyzer* chapter in the *Quartus II Handbook.*

Logic Analyzer Interface

Logic Analyzer Interface enables you to connect and transmit the internal FPGA signals to an external logic analyzer or a mixed signal oscilloscope for analysis. You can use this feature to connect a large set of internal device signals to a small number of output pins for debugging purposes. This feature functions as a multiplexer and you can select which of the signal groups to be connected to the output pins for monitoring on the fly, without the need to recompile the design or reconfigure the FPGA.

The Logic Analyzer Interface requires JTAG connection and communicates with the device through an Altera download cable. Additional LEs and I/Os are needed as well.

For more information about ways to use the Logic Analyzer Interface feature for debugging, refer to the *In-System Debugging Using External Logic Analyzers* chapter in the *Quartus II Handbook*.

In-System Memory Content Editor

The In-System Memory Content Editor feature allows you to modify the content of the memory or constant through the JTAG interface. This is useful in debugging for design examples that perform read and write operation on the memory so that you can check whether the design writes or reads the correct data to or from the memory. Changes to the memory or constant can be done without interrupting the device's functionality.

Specify that the memory or constant are modifiable when instantiating the megafunction for the memory or constant. The In-System Memory Content Editor requires JTAG connection and communicates with the device through an Altera download cable. Additional LEs are needed when this feature is turned on.

For more information on ways to use the In-System Memory Content Editor to access the on-chip memory, refer to the *In-System Updating of Memory and Constants* chapter in the *Quartus II Handbook*.

In-System Sources and Probes

This feature allows you to input simple virtual stimuli and capture the current value of internal nodes in your design dynamically without the use of any external test equipment. You can force trigger conditions set up using the SignalTap II Logic Analyzer. The feature needs the altsource_probe megafunction instantiated in your design before compilation and requires JTAG connection for the communication through an Altera download cable. Additional LEs are needed when this feature is used.

For more information on ways to use the In-System Source and Probes for debugging, refer to the *Design Debugging Using In-System Sources and Probes* chapter in the *Quartus II Handbook*.

Virtual JTAG Megafunction

Similar to the in-System Sources and Probes, the Virtual JTAG megafunction allows you to apply virtual stimuli and capture the current value of internal nodes in your design dynamically without the use of any external test equipment, albeit the Virtual JTAG megafunction gives you a greater level of control but at the cost of greater complexity. The feature needs the sld_virtual_jtag megafunction instantiated in your design before compilation and requires JTAG connection for the communication through an Altera download cable. Additional LEs are needed when this feature is used.

For more information on ways to use the sld_virtual_jtag megafunction for debugging, refer to the *Virtual JTAG (sld_virtual_jtag) Megafunction User Guide*.

Board Design Considerations

This section contains the information for your consideration when designing the board.

I/O Consideration

3.3/3.0/2.5V LVTTL/LVCMOS Interface

Cyclone III devices are designed to support interface voltage levels from 1.2 V up to 3.3 V to accommodate the need for flexible I/O interface implementation. You can use Cyclone III devices to directly drive out 3.3-V **LVTTL** at up to 8 mA and 3.3-V **LVCMOS** at up to 2 mA. When using Cyclone III devices as a receiver in 3.3/3.0/2.5-V **LVTTL**/**LVCMOS** systems, you must follow the operating conditions including the Cyclone III absolute maximum DC input voltage and maximum allowed overshoot/undershoot voltage conditions.

For the values of the absolute maximum DC input voltage and maximum allowed overshoot/undershoot voltage, refer to the *Cyclone III Device Datasheet* in the *Cyclone III Device Handbook.*

Follow the guidelines in *AN 447: Interfacing Cyclone III and Cyclone IV Devices with 3.3/3.0/2.5 V LVTTL/LVCMOS I/O Systems* to ensure the device reliability when interfacing Cyclone III with 3.3/3.0/2.5 V voltage levels.

Pad Placement Consideration

The V_{CCIO} supply for a bank is susceptible to noise from switching outputs in the bank. To maintain an acceptable noise level on the V_{CCIO} supply, there are restrictions on placement of single-ended I/O pads in relation to differential pads. The Quartus II software automatically checks for these restrictions.

When single-ended voltage-referenced inputs are used in a bank, the Quartus II software automatically checks for restrictions about the placement of outputs in relation to V_{REF} pads and supply pairs (V_{CCIO} and GND). The restriction is in place to maintain acceptable noise level on the V_{CCIO} supply and prevent output switching noise from shifting the V_{REF} rail.

For placement guidelines for single-ended pads with respect to differential pads for each supported differential I/O standards, and for input, output and bidirectional pads when voltage referenced input pads exist in a bank, refer to the "Pad Placement and DC Guidelines" section in the *I/O Features in the Cyclone III Device Family* chapter of the *Cyclone III Device Handbook*.

In specific applications, you can relax the restriction checks in the Quartus II software. For example, if you have a non-toggling single-ended pin, you can place it closer to a differential pin safely—bypassing the pin placement checks. To set this in the Quartus II software, assign 0 MHz toggle rate to Toggle Rate assignments for the pin in the Assignment Editor.

The Output Enable Group assignment is another useful setting especially in external memory interfaces to allow efficient placement of output or bidirectional pins in a V_{REF} group when a voltage referenced input is used in the group.

For details about implementing these features, refer to the *I/O Management* chapter in the *Quartus II Handbook*.

Minimizing Simultaneous Switching Noise

Simultaneous switching noise (SSN) becomes a concern when too many pins within close proximity change levels at the same time and cause V_{CC} sag or ground bounce on the quiet pins nearby. Noise generated by SSN can reduce noise margin and cause incorrect switching.

When creating your design, try to separate the pins that switch simultaneously. If possible, distribute the switching pins to different I/O banks. Set the unused I/O pins nearby to V_{CC} to minimize V_{CC} sag, or to ground to minimize ground bounce. You can also turn on the slow slew rate feature and use a lower drive strength for the switching pins. Proper termination on the switching I/O pins also helps to reduce reflection and the SSN effect on the quiet pins.

For details about the sources of the SSN, ways to mitigate SSN and guidelines on a PCB design for the general high speed digital designs, refer to *AN 508: Cyclone III Simultaneous Switching Noise (SSN) Design Guidelines.*

For board design guidelines, refer to *AN 224: High-Speed Board Layout Guidelines* and *AN 315: Guidelines for Designing High-Speed FPGA PCBs.*

Unused Pin Connection

The Quartus II software generates the pin report file (**.pin**) when you compile your design. This report file specifies how you should connect the unused pins of your device. For a Cyclone III device, depending on how you set the unused pins in the Quartus II software, unused I/O pins are marked in the report file as any one of the following items:

- `GND*`
- `RESERVED`
- `RESERVED_INPUT`
- `RESERVED_INPUT_WITH_WEAK_PULLUP`
- `RESERVED_INPUT_WITH_BUS_HOLD`

All I/O pins specified as `GND*` can be connected to ground to improve the device's immunity to noise, or left unconnected. Leave all `RESERVED` I/O pins unconnected on your board because these I/O pins drive out unspecified signals. Tying a `RESERVED` I/O pin to V_{CC}, ground, or another signal source can create contention that can damage the output driver of the device.

`RESERVED_INPUT` I/O pins can be connected to a high or low signal on the board while `RESERVED_INPUT_WITH_WEAK_PULLUP` and `RESERVED_INPUT_WITH_BUS_HOLD` pins can be left unconnected.

Termination Schemes

Correct termination is important to prevent signal reflection on the signal lines. Depending on the I/O standards used and the direction of the signals travel, the termination scheme you should use on the boards may be different:

- You can use series and parallel termination.
- In general, series termination is normally used at the transmitter while parallel termination is used at the receiver.
- The value of the termination resistors should match the trace impedance.
- You can perform board-level simulation to obtain the suitable resistance values based on your termination scheme.

The Cyclone III device has on-chip series termination (with and without calibration) that you can use to replace the series termination resistors on your board.

For information about the various termination schemes, refer to the following documents:

- *I/O Features in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*
- *AN 224: High-Speed Board Layout Guidelines*
- *AN 315: Guidelines for Designing High-Speed FPGA PCBs*

Board-Level Simulation

To ensure that the selected I/O signaling meets receiver threshold levels with the board setup, perform simulation with third-party board-level simulation tools using the Cyclone III IBIS or HSPICE model.

Both the IBIS and HSPICE models describe the behavior of I/O buffers, but in a different way. The IBIS models describe the I/O buffers with voltage-current and voltage-time data curves while the HSPICE models describe the I/O buffers by their physical properties, such as transistor characteristics, parasitic capacitance and their connections to one another.

You can download the IBIS and HSPICE models from www.altera.com. You can also use the Quartus II software to create custom IBIS and HSPICE models for your design and perform simulation with the models to check the effect of the termination scheme on your board signals.

For information about performing simulation with IBIS and HSPICE models, refer to the *Signal Integrity Analysis with Third-Party Tools* chapter in the *Quartus II Handbook*.

Power Consideration

Table 5 lists the external power supplies required to power Cyclone III devices.

Table 5. Cyclone III Power Supply Requirements

Power Supply	Description
V_{CCINT}	Core voltage power supply
V_{CCIO}	I/O power supply to the input and output buffers in Bank 1 to Bank 8.
V_{CCA}	Analog power supply for PLL.
V_{CCD_PLL}	Digital power supply for PLL.
V_{REF}	Input reference voltage for voltage-reference I/O standards.

For the possible values of each power supply and the recommendation operation conditions, refer to the *Cyclone III Device Datasheet* chapter in the *Cyclone III Device Handbook*.

The V_{CCIO} pin connections depend on the design's I/O standards, and support 1.2, 1.5, 1.8, 2.5, 3.0 and 3.3 V. Each bank can support a different voltage level. The V_{REF} pin serves as an input reference voltage for the voltage reference I/O standards and is used mainly for a voltage bias and does not source or sink much current. The voltage can be created with a regulator or a resistor divider network. If voltage reference I/O standards are not used in the bank, the `VREF` pins are available as user I/O pins

Altera suggests that you use a linear regulator to power the V_{CCA} pins because they power the analog circuitry. You can power the digital voltage rails with the linear or switching regulators depending on the efficiency or cost considerations.

To reduce system noise, it is important to ensure that the power supply is clean. Place a ferrite bead and tantalum parallel capacitors where the power supply enters the board's power plane to filter out the noise to the power plane. Generally, the tantalum capacitors are used for circuits which demand high stability in the capacitance values.

The ferrite bead should be connected in series between the power supply and the power plane, while the capacitors are connected between the power plane and ground, in parallel with each other. Decoupling depends on the design decoupling requirements of the specific board.

For information on minimizing noise for power supplies, refer to the following documents:

- *AN 224: High-Speed Board Layout Guidelines*
- *AN 315: Guidelines for Designing High-Speed FPGA PCBs.*

The PLL contains analog components embedded in a digital device. Consider the following items for designing the PLL power supply and minimizing jitter:

- Run a thick trace (at least 20 mils) from the power supply to each `VCCA` pin.
- Connect all `VCCD_Pll` power pins to the quietest digital supply on the board.
- Ensure that all `VCCA` and `VCCD_PLL` power pins are connected to a 2.5-V and 1.2-V power supply respectively, even if you are not using any PLL in the device.
- Use an isolated linear regulator to power the `VCCA` pin.
- Connect the `VCCD_PLL` power pin to the quietest digital supply on the board.
- Filter each `VCCA` and `VCCD_PLL` power pin with a decoupling circuit. Refer to the Cyclone III device pin-out for recommendation.
- Connect the `GNDA` pins to an isolated analog ground plane on the board.

For more information about the decoupling strategy for Cyclone III power supplies, refer to *Cyclone III Device Family Pin Connection Guidelines.*

Device Power-up

In many applications, knowing the power-up behavior of your device is critical, especially when you are inserting or removing the board in a system during system operation. Among the considerations are I/O behaviors, and the conditions of supplies and timing.

Review the conditions and considerations discussed in this section before you begin with board design and configuration setup to ensure a successful implementation of Cyclone III devices in your design.

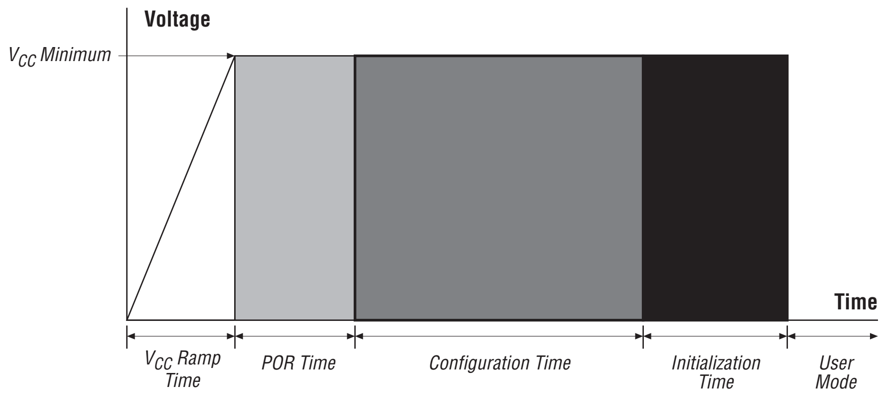
Technology

Cyclone III devices feature hot-socketing and power-on reset (POR) circuitry that regulates the power-up sequencing of the device before configuration:

- The circuitries ensure that the Cyclone III devices are ready for configuration while taking care of uncertainties on device interfaces during these stages
- The circuitries enable the Cyclone III device to be hot-swapped without the use of any external devices.

Figure 1 shows the power-up stages of a Cyclone III device.

Figure 1. Cyclone III Power-Up Stages



Note to Figure 1:

(1) V_{CC} ramp must be monotonic.

For more details about hot socketing and power-on reset circuit operation, refer to the *Hot Socketing and Power-On Reset in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Requirements and Behavior

The requirement and behavior of a Cyclone III device are listed in Table 6 for each stage before configuration.

Table 6. Requirement and Behavior of Device During Various Power-up Stages

Stages	Setup	
	Requirement	**Behavior**
Pre-Power Up	■ Signals can be driven into Cyclone III I/O pins, dedicated inputs and dedicated clock pins without damaging the device. ■ Determine the required POR time for your device; fast POR (3 ms to 9 ms) or standard POR (50 ms to 200 ms). Selection is determined by `MSEL` pins setting.	Not applicable
V_{CC} Ramp	■ Can support any power-up sequence of supplies. ■ V_{CCIO}, V_{CCINT}, and V_{CCA} are required. V_{CCIO} for banks with configuration pins (1, 6, 7, and 8) are required. ■ Supplies must meet ramp rate according to required POR; 50 µs to 3 ms for fast POR or 50 ms for standard POR. Fast POR is used when Cyclone III device is required to wake up quickly to begin operation. Select POR using `MSEL` pin settings—refer to the *Configuration, Design Security, and Remote System Upgrades in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*. ■ V_{CC} ramp must be monotonic.	■ Output buffers tri-stated for all conditions, for example: ■ when V_{CCIO} is powered before V_{CCINT} ■ if the I/O pad voltage is higher than V_{CCIO} ■ sudden voltage spikes (overshoot) during hot-socketing. ■ Exception for configuration pins, which are expected to drive out during power-up cycle. No hot socketing circuit for these pins. ■ No current path exists from I/O pin to V_{CCINT} or V_{CCIO}, as hot-socketing circuit is enabled. Refer to the *Hot Socketing and Power-On Reset in the Cyclone II Device Family* chapter in the *Cyclone III Device Handbook* for leakage or driving current during hot-socketing. ■ Safe from latch-up, no low-impedance path from V_{CC} to GND that may result in large current passing through the path.
POR	■ Maintain V_{CC} ramp to desired operating voltage level. ■ If maximum V_{CC} ramp time cannot be met, use an external component to hold `nCONFIG` low until the power supplies have reached their minimum recommended operating levels. Else, the device may not configure properly and enter user mode.	■ Output buffers remain tri-stated. ■ POR circuit keeps device in reset state until all V_{CC} supplies have stabilized and reached acceptable levels before configuration is triggered.

After the Cyclone III device enters user mode, the POR circuit continues to monitor the `VCCINT` and `VCCA` pins so that a brown-out condition during user mode can be detected. If the V_{CCINT} and V_{CCA} voltage sags below the POR trip point during user mode, the POR circuit resets the device. The POR circuit does not reset the device if the V_{CCIO} voltage sags during user mode.

For more information about power-on reset, refer to the *Hot Socketing and Power-On Reset in Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Configuration Pin Connections

Depending on your configuration scheme, different pull-up or pull-down resistor or signal integrity requirements may apply. Some configuration pins also have specific requirements if unused. It is important to correctly connect the configuration pins. This section provides guidelines to address common issues.

The internal PCI clamping diode for the dual-purpose configuration pin is turned off if the pin is part of the selected configuration scheme in your Quartus II project.

For a list of the dedicated and dual-purpose configuration pins, and a description of the function and connection guidelines, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

Configuration and JTAG Pins I/O Voltage Requirements

When using a serial configuration device in the AS configuration scheme, you must connect a 25-Ω series resistor at the near end of the serial configuration device for the DATA[0]. When cascading Cyclone III devices in multi-device configuration, you must connect repeater buffers between the Cyclone III master and slave device for DATA and DCLK. The output resistance of the repeater buffers has to fit the maximum overshoot equation given by $0.8Z_O = R_E = 1.8Z_O$. In this equation, Z_O is the transmission line impedance and R_E is the equivalent resistance of the output buffer.

For information about configuration requirements, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

DCLK/TCK Signal Integrity

Adopt good design techniques to ensure that the DCLK and TCK traces on your board produce clean signals with no overshoot, undershoot, or ringing. When designing the board, lay out the DCLK and TCK traces using the same techniques used to lay out a clock line. A noisy DCLK signal could affect configuration and cause an nSTATUS error. Ensure that the TCK trace produces a clean signal with no overshoot, undershoot, or ringing. For a chain of Cyclone III devices, noise on any of the DCLK or TCK pins of each device in the chain could cause configuration or JTAG programming to fail for the whole chain.

For more information about connecting devices in a chain, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

JTAG/Configuration Pin Pull-up/Down

Noise at the JTAG pins, whether the device is in ISP or user mode, or during power up, can cause the device to go into an undefined state or mode. Altera recommends pulling the TCK pin low and the TMS pin high through resistors.

The JTAG circuitry is activated when V_{CCINT} is powered up. If the `TMS` and `TCK` pins are connected to V_{CCIO} and the V_{CCIO} is not powered up, the JTAG signals are left floating. Any transition on the `TCK` pin can cause the JTAG state machine to transition to an unknown state, leading to incorrect operation when V_{CCIO} is finally powered up. To disable the JTAG state machine during power-up, the `TCK` pin should be pulled low to ensure that an inadvertent rising edge does not occur on `TCK`.

JTAG/Configuration Chain Connection

Connect the JTAG pins of the device to the download cable header correctly. If you have more than one device in the chain, connect the `TDO` pin of a device to the `TDI` pin of the next device in the chain.

All I/O inputs must maintain a maximum AC voltage of 4.1 V. Because JTAG pins do not have the internal PCI clamping diodes to prevent voltage overshoot when using V_{CCIO} of 2.5 V, 3.0 V or 3.3 V, you must power up the download cable's V_{CC} with a 2.5 V supply from V_{CCA}. When using device V_{CCIO} of 1.2 V, 1.5 V, or 1.8 V, you can power up the download cable's V_{CC} with the supply from V_{CCIO}.

For more information on JTAG configuration chain, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*. You can also cascade the FPGA devices in chain in other configuration schemes such as AS, AP, PS, and FPP.

Multi Device Multi VCCIO (Level Shifter)

The operating voltage supplied to the Altera download cable by the target board through the 10-pin header determines the I/O voltage level of the download cable. The JTAG pins for all Cyclone III devices reside in Bank 1 and their I/O standard support is controlled by the V_{CCIO} setting for Bank 1.

As the download cable interfaces with the JTAG pins of your device, ensure that the download cable I/O voltage and the JTAG pin voltage are compatible. Table 7 shows the I/O voltages supported by different download cables.

Table 7. Download Cable I/O Voltage

Download Cable	I/O Voltage		
	2.5 V V_{CCA} (1)	1.8 V V_{CCIO} (2)	1.5 V V_{CCIO} (2)
USB-Blaster™	✓	✓	—
ByteBlaster™ II	✓	✓	✓
ByteBlasterMV™	✓	—	—
MasterBlaster™	✓	—	—

Notes to Table 7:

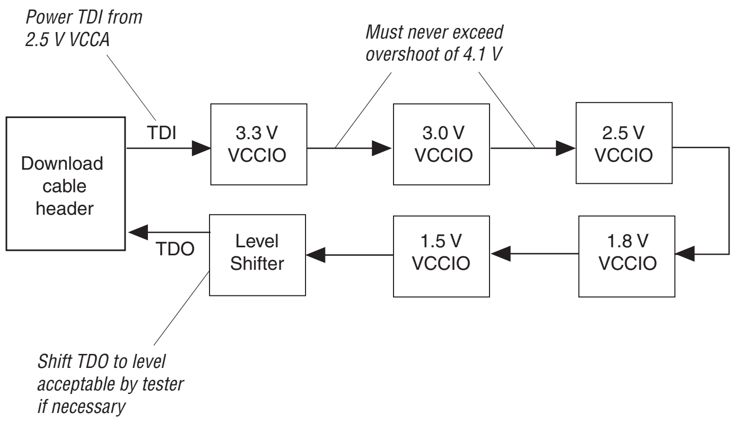
(1) When interfacing the download cables with the Cyclone III device, you must ensure that all I/O inputs to the device maintain a maximum AC voltage of 4.1 V. Altera recommends powering up the download cables with V_{CCA} of 2.5 V when using V_{CCIO} of 2.5 V, 3.0 V, or 3.3 V on your Cyclone III devices.

(2) When using device V_{CCIO} of 1.2 V, 1.5 V, or 1.8 V, you can power up the download cable's V_{CC} with the supply from V_{CCIO}. However, Altera download cables do not support a target supply voltage of 1.2 V.

For more information about JTAG configuration with Altera download cables, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*, and the appropriate download cable user guide.

In a JTAG chain containing devices with different V_{CCIO}, the devices with a higher V_{CCIO} level should drive the devices with the same or lower V_{CCIO} level. All I/O inputs must maintain a maximum AC voltage of 4.1 V. To prevent voltage overshoot when using V_{CCIO} of 2.5 V, 3.0 V, or 3.3 V, you must power up the download cable's V_{CC} with a 2.5 V supply from V_{CCA}. When using device V_{CCIO} of 1.2 V, 1.5 V, or 1.8 V, you can power up the download cable's V_{CC} with the supply from V_{CCIO}. However, Altera download cables do not support a target supply voltage of 1.2 V. You only need one level shifter at the end of the chain with this device arrangement. If this arrangement is not possible, you have to add more level shifters into the chain. Figure 2 shows the JTAG chain that contains devices with different V_{CCIO}.

Figure 2. Devices with Different VCCIO in a JTAG Chain


Power TDI from 2.5 V VCCA
Must never exceed overshoot of 4.1 V
Download cable header
TDI
3.3 V VCCIO
3.0 V VCCIO
2.5 V VCCIO
TDO
Level Shifter
1.5 V VCCIO
1.8 V VCCIO
Shift TDO to level acceptable by tester if necessary

JTAG Signal Buffering

The JTAG signal integrity determines the need to buffer a JTAG chain. Pay particular attention to the `TCK` signal because it is the JTAG clock and is the fastest switching signal compared to the other JTAG signals. Altera recommends buffering the signals at the connector because cables and board connectors tend to act as poor transmission lines and introduce noise to the signals. After this initial buffer at the connector, add buffers as the chain gets longer or whenever the signals must cross a board connector.

At any given time, when a cable must drive three or more devices, buffer the signal at the cable connector to prevent signal deterioration. The decision to buffer the signal also depends on the board layout, loads, connectors, jumpers, and switches on the board. Anything added to the board that affects the inductance or capacitance of the JTAG signals increases the likelihood of a buffer being added to the chain.

For the `TCK` and `TMS` signals that drive in parallel, each buffer should drive no greater than eight loads. If jumper or switches are added to the path, decrease the number of loads.

Unused JTAG Pins Connection

If you are not using the JTAG interface, make sure the JTAG pins on the Cyclone III device are not left floating and are connected to a stable level. Because JTAG configuration takes precedence over all other configuration methods, these pins should not be left floating or toggling during configuration.

MSEL Configuration Mode Pins

Select the configuration scheme by driving the Cyclone III device `MSEL` pins either high or low. The `MSEL` pins are powered by the V_{CCINT} power supply of the bank in which they reside. The `MSEL[3..0]` pins have 5-kΩ internal pull-down resistors that are always active. During power-on reset (POR) and during reconfiguration, the `MSEL` pins have to be at least **LVTTL** V_{IL} or V_{IH} levels to be considered a logic low or logic high, respectively. To avoid any problems with detecting an incorrect configuration scheme, hard wire the `MSEL` pins to V_{CCA} and GND without any pull-up or pull-down resistors. Alternatively, set up your board so that you can connect each pin to either V_{CCA} or GND with a 0-Ω resistor to change between configuration modes during testing or debugging. Altera recommends not to drive the `MSEL` pins with a microprocessor or another device.

Do not leave the `MSEL` pins floating.

For more information about configuration, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*. Also refer to the Configuration Center, which includes links to troubleshooters that you can use to help debug configuration problems. To help you debug JTAG programming issues, refer to the JTAG Configuration & ISP Troubleshooter. To debug FPGA configuration issues, refer to the FPGA Configuration Troubleshooter.

Configuration Devices

The EPCS devices (EPCS4, EPCS16, EPCS64, and EPCS128) are used in the AS configuration scheme for Cyclone III devices.

For information about the EPCS devices, refer to the *Serial Configuration (EPCS) Devices Datasheet*.

EPCS Connection

The four-pin interface of the EPCS device consists of a serial clock input (`DCLK`), serial data output (`DATA`), AS data input (`ASDI`), and an active-low chip select (`nCS`). This four-pin interface connects to the Cyclone III device pins, which are the `DCLK`, `DATA[0]`, `DATA[1]`, and `FLASH_nCE`, respectively. The Cyclone III device's `FLASH_nCE` pin functions as the `nCEO` pin in AS configuration scheme, and `DATA[1]` pin functions as the `ASDO` pin in AS configuration scheme.

When connecting a serial configuration device to the Cyclone III device in single device AS configuration, you must connect a 25-Ω series resistor at the near end of the serial configuration device for the `DATA[0]`. In single device AS configuration, the board trace length between the serial configuration device to the Cyclone III device should not exceed 10 inches.

For more information on the connections for AS configuration scheme, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

If you use the AP configuration scheme for Cyclone III devices, the V_{CCIO} of I/O Banks 1, 6, 7 and 8 must be the same and must be either 1.8, 2.5, 3.0 or 3.3 V. Altera recommends not to use level shifters between a configuration device and the Cyclone III device in any active (AS or AP) configuration scheme.

EPCS Support for Cyclone III Devices

In Cyclone III devices, the active master clock frequency runs at a maximum of 40 MHz, and at a typical 30 MHz. Cyclone III devices work only with EPCS devices that support up to 40 MHz. Existing batches of EPCS4 manufactured on 0.15 µm process geometry support AS configuration in Cyclone III devices up to 40 MHz. However, batches of EPCS4 manufactured on 0.18 µm process geometry support only up to 20 MHz and do not support AS configuration in Cyclone III devices. The EPCS16 and EPCS64 serial configuration devices are not affected.

For information about product traceability and transition date to differentiate between 0.15 µm process geometry and 0.18 µm process geometry EPCS4 serial configuration devices, refer to the *PCN 0514 Manufacturing Changes on EPCS Family* process change notification on the Altera website at www.altera.com.

Flash Pins and other Pin Connections

For information on the flash pin connections for the AP configurations scheme, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

Flash for AP

In the AP configuration scheme, the commodity parallel flash is used as configuration memory. The AP configuration controller in Cyclone III devices is designed to interface with the Intel StrataFlash® Embedded Memory (P30 and P33) flash families. Unlike the EPCS devices, the P30 and P33 flash families supported in the AP configuration scheme is designed to interface with microprocessors. By configuring from an industry standard microprocessor flash which allows access to flash after it is in user mode, the AP configuration scheme allows you to combine configuration data and user data (microprocessor boot code) on the same flash memory.

The Intel P30 and P33 flash families support a continuous synchronous burst read mode at 40 MHz `DCLK` frequency for reading data from the flash. You must refer to the respective flash datasheet to check for the supported speed grades and package options. For example, the Intel P30 and P33 families have only a single speed grade at 40 MHz; however, they do not support 40 MHz on the TSOP package. Therefore, the P30 and P33 FBGA packages are supported for the AP configuration scheme, while the TSOP package is not supported.

For more information about the active parallel configuration scheme and its supported parallel flash, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook*.

Design and Compilation

This section covers topics you should look into during the design process, which includes design entry consideration and recommendation, power consideration, information on the Cyclone III I/O pins and PLLs, timing consideration as well as information on performing simulation on your design.

Design Entry

The Quartus II software allows you to create your design through schematic/block diagram or hardware description language (HDL) coding. The commonly used HDL formats supported are Verilog and VHDL. For simple designs, using schematic or block diagrams make your task of creating the design easier. However, for more complex designs, for example designs with state machines, using HDL coding gives you the flexibility you need, and in some cases, makes your design more efficient.

If you use HDL as the design entry for your design, pay attention to the coding style. HDL coding has a significant effect on the quality of results in terms of logic utilization and performance that you achieve for your designs. Effective coding helps the synthesis tool to perform better when synthesizing your design.

For the specific HDL coding examples and recommendations, refer to the *Recommended HDL Coding Styles* chapter in the *Quartus II Handbook*. Refer to your synthesis tool's documentation for any additional tool-specific guidelines. In the Quartus II software, you can use the HDL examples in the Language Templates available from the right-click menu in the text editor.

The Quartus II software is able to generate the symbol or HDL files for the megafunctions you create and allows you to integrate the megafunction into your design, regardless of the design entry method. You can also generate the symbol file for certain blocks in your design written in HDL. With that, your design can have a mixture of HDL and schematic entry. You can infer certain device resources, for example the memory, through HDL as well.

Selecting a Synthesis Tool

The Quartus II software can synthesize design entries in Verilog HDL, VHDL, Altera hardware description language (AHDL) and schematics. You can also use third-party EDA synthesis tools to synthesize your Verilog or VHDL design, and then use the Quartus II software to perform the placement and routing based on the generated netlist. Specify the third-party synthesis tool in the New Project Wizard or the **EDA Tools Settings** page of the **Settings** dialog box to use the correct Library Mapping File for your synthesis netlist.

Your synthesis tool might offer the capability to create a Quartus II project and pass constraints such as the EDA tool setting, device selection, and timing requirements that you specified in your synthesis project. You can use this capability to save time when setting up your Quartus II project for placement and routing.

Altera recommends using the most recent version of the third-party synthesis tools, because tool vendors are continuously adding new features, fixing tool issues, and enhancing performance for Altera devices. The EDA tools that can generate the netlist for the Quartus II Fitter include Synplicity Synplify and SynplifyPro software, Synopsys Design Compiler, and Mentor Graphics® Precision RTL Synthesis software, and LeonardoSpectrum™ software. The EDA tools normally generate the design netlist file in either EDIF Input File (**.edf**) or Verilog Quartus Mapped (**.vqm**) format.

For more information on the supported synthesis tools, refer to the appropriate chapter in *Volume 1: Design and Synthesis* of the *Quartus II Handbook*. The Quartus II Release Notes list the version of each synthesis tool that is officially supported by that version of the Quartus II software.

Qsys System Integration Tool

Qsys is a system integration tool that is included as part of the Quartus II software. Qsys captures system-level hardware designs at a high level of abstraction and automates the task of defining and integrating customized HDL components, which may include IP cores, verification IP, and other design modules.

Qsys facilitates design reuse by packaging and making available your custom components and systems, and integrates your custom components with Altera® and third-party developer components.

During system generation, Qsys automatically creates interconnect logic from the connectivity options you specify, eliminating the error-prone and time-consuming task of writing HDL to specify the system-level connections.

For more information about Qsys, refer to the Qsys System Integration Tool Support page on the Altera website.

For guidelines to migrate from SOPC Builder to Qsys, refer to *AN 632: SOPC Builder to Qsys Migration Guidelines*.

Intellectual Property

Altera and its third-party intellectual property (IP) partners offer a wide range of IP cores. You can use these fully parameterized and performance-optimized IP cores in your design to save time on designing and testing as these IP cores are fully tested and their functionality is proven. These IP cores are available for evaluation, allowing you to verify the functionality and timing of the IP cores prior to purchasing a license.

For systems that have the Nios II embedded processor, you can write your C/C++ code for the processor. This is especially useful for system developers who are more familiar in the C/C++ programming language than the hardware description language.

For more information on the IP cores offered by Altera and its third-party IP partners, refer to the Intellectual Property & Reference Designs page on the Altera website.

Using Megafunctions

Altera provides parameterized megafunctions that are optimized for Altera device architectures. You can save design time with megafunctions instead of coding your own logic. Additionally, the megafunctions provided by Altera may offer more efficient logic synthesis and device implementation. You can scale the megafunction's size and set various options with the parameters. Megafunctions include the library of parameterized modules (LPM) and Altera device-specific megafunctions. You can also take advantage of Altera and third-party IP and reference designs to save design time.

The Quartus II MegaWizard Plug-In Manager provides an easy user interface to customize megafunctions. You should build or change megafunction parameters using the wizard to ensure that you set all the ports and parameters correctly.

For detailed information about specific megafunctions, refer to Quartus II Help or the megafunction user guides on the User Guides literature page.

Design Recommendations

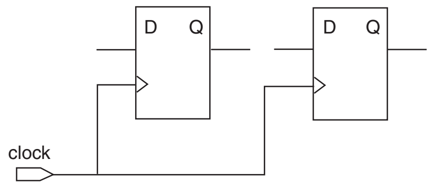
Good design practices especially those related to the clocks are essential in ensuring that your design functions as intended.

Synchronous Design

In a synchronous design, the clock signal triggers signal transitions. On every active edge of the clock, the data inputs of registers are sampled and transferred to outputs.

Use a single clock source to clock the registers in your design, as shown in Figure 3. If two cascaded registers are triggered on different clock sources or edges, there is a risk that the second register will not have enough time to resolve the metastable output from the first register because of setup time violation at the second register, and thus will clock in an incorrect value.

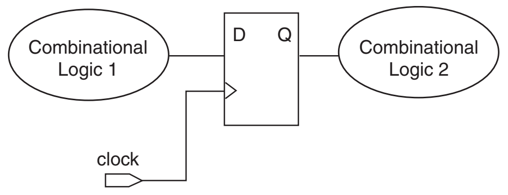
Figure 3. Synchronous Design



If the combinational logic output from your design feeds to another part of the design, have the signal go through a register, as illustrated in Table 2. This applies if you are using the combinational logic output as a clock signal or as an asynchronous reset signal. Due to propagation delays through the combinational logic, the signal may go through a number of transitions before the output settles to a new value. This means changes to the combinational output can trigger a period of instability.

As the input of the register is only sampled and transferred to the design (for example, the combinational logic 2 shown in Figure 4) on every active edge of the clock, transitions taking place on data inputs of the register do not affect the register output or input to the other part of the design until the next active clock edge. As long as the setup and hold time of the register is not violated, the register effectively isolates any glitches or instable input signals from other logics.

Figure 4. Registering Combinational Output Signal



Altera recommends that you register the input signals to the device as well to filter out any glitches. The Cyclone III I/Os have input registers for this purpose. Using the input registers for your input signals allows faster setup times compared to using the LE registers. This can be done with the **Fast Input Register** assignment.

Also, your design should not rely on delay paths within the architecture of a device, for example using the LCELL primitives to increase the delay of certain signals in the design because factors such as temperature, voltage, process change, or placement and routing change could affect the timing of logic paths in the device. Any change in timing to a specific path could cause unwanted functional changes and affect the design functionality. Synchronization eliminates the unwanted functional changes.

For more information about synchronizing your design, refer to the *Recommended Design Practices* chapter in the *Quartus II Handbook* or the Quartus II Help.

Clocks

Clocks are important as they have a large effect on your design's timing accuracy, performance, and reliability. Problems associated with the clocks can cause functional and timing problems in the design. Pay attention to the following sub-sections that are related to clocks when you are in the designing stage.

Global Clock Pins and Clock Networks

Use dedicated clock pins and global clock routing for clock signal input to your design. The dedicated clock pins drive the clock network directly, ensuring lower skew compared to using normal I/O pins. High clock skew might cause hold time violation. The clock network drives the entire Cyclone III device and you can use the global clock to drive all the features in the device, for example the M9K memory and logic array blocks. Using the clock network ensures you have a predictable and more constant delay for signals that have high fan-out. You can also use the clock pins and clock network to drive control signals such as asynchronous reset.

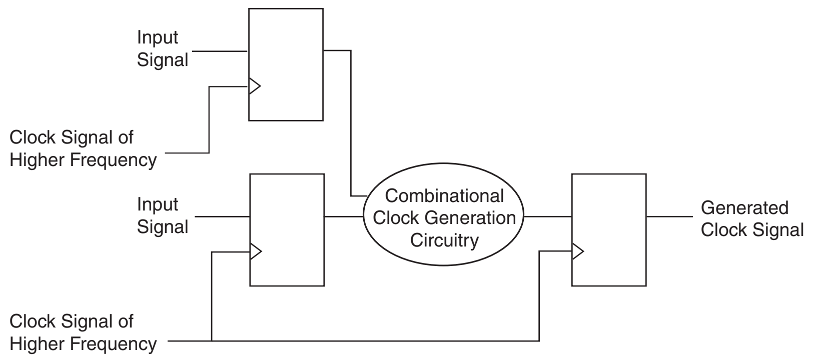
For more information about the global clock pins and clock networks, refer to the *Clock Networks and PLLs in Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Internally Generated Clocks

Combinational logic output is not immune to glitches as it depends on the inputs to the circuit. The input signals may arrive at the combinational logic circuit at different times and cause glitches. Moreover, the input signals may have glitches and the combinational logic circuitry does not filter out these glitches. If you use this combinational logic output signal as a clock to other circuitries, these glitches can cause functional problems.

Always register the input signals to the combinational circuitry and the output signal from that circuitry that is used as a clock (Figure 5) if you have a clock signal with a higher frequency than the clock generated internally. Registering the input signals to the combinational circuitry ensures the arrival of the signals to the circuit is controlled by the clock, and any glitches are filtered in the input signals. By registering the output signals from the circuit, you can filter out any glitches generated by the circuitry.

Figure 5. Registering the Input and Output Signals


Input Signal
Clock Signal of Higher Frequency
Input Signal
Combinational Clock Generation Circuitry
Generated Clock Signal
Clock Signal of Higher Frequency

Clock Division

To obtain a lower frequency clock from a higher frequency clock, use a PLL to divide the clock. PLL has a predictable output delay, and you can also change the phase of the output signal to compensate for any delay. Using cascaded registers to divide the clock introduces delay to the clock. As the ripple carry register chain propagates the delay through the chain, there will be phase shift between the input clock and the output clocks, and between different output clocks that are generated. Also, the delay depends on the placement of the registers.

Inverted Clock

Use a PLL to generate an inverted clock, instead of using a `NOT` gate implemented with an LE. The PLL allows you to set the phase shift of the output signal and compensate for any delay between the input and output signals.

Multiplexed Clock

Some user designs require different clock source selection. Use the dedicated clock control block or the PLL clock switchover feature of the Cyclone III device to multiplex clock inputs. The clock control block allows up to four different clock source selections, either from clock pins or PLL outputs as the source for the clock network. You can dynamically select the clock source from two clock input pins and two PLL outputs through the altclkctrl megafunction.

The PLL clock switchover feature also allows you to dynamically select between two clock sources, either from clock input pins or output clocks from another PLL.

For more information, refer to the following documents:

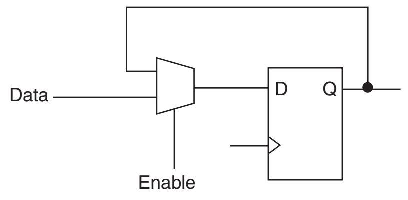
- *Clock Networks and PLLs in Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*
- *Clock Control Block (ALTCLKCTRL) Megafunction User Guide*
- *Phase-Locked Loop (ALTPLL) Megafunction User Guide*

Gated Clock

Gated clock allows users to turn off the clock signal. However, the recommended way to turn off the clock in the Cyclone III device is to use the clock control block. The clock control block allows you to power down the clock network. Use the altclkctrl megafunction to instantiate the clock control block.

If the purpose of turning off the clock is to stop clocking registers to prevent the register output change, you can control the input to the register instead, as illustrated in Figure 6. The enable signal controls the input to the register, while the clock signal keeps toggling. This ensures the register output is synchronized with other register outputs that use the same clock. Gating the clock to the register introduces additional delay to the clock and the register output may not be synchronized with other signals.

Figure 6. Synchronous Clock Enable



If you must implement gated clocks using logic resources, register the gating signal that enables or disables the clock to filter out any glitches that might appear in the gating signal. Otherwise, glitches from the gating signal can propagate to the clock signal. Also, gating the clock signal at the source minimizes the clock delay differences between different blocks within the design that are clocked by the same clock signal.

For more information, refer to the following documents:

- *Clock Networks and PLLs in Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*
- *Clock Control Block (ALTCLKCTRL) Megafunction User Guide*

Chip-Wide Reset

A Cyclone III device supports the chip-wide reset that clears all registers in the device, including registers of the M9K blocks. This reset overrides all other control signals of the registers and allows you to re-initialize the device at any given time.

To enable the chip-wide reset from the Quartus II software, on the **Assignments** menu, click **Settings**. Under **Device**, click **Device & Pin Options**. Then, on the General window, turn on **Enable device-wide reset (DEV_CLRn)**. Enable this feature before compiling your design. When the chip-wide reset option is turned on, holding the `DEV_CLRn` pin low resets all the registers. When this feature is turned off, the `DEV_CLRn` pin functions as a normal user I/O pin.

Register Power-Up Level

You can select the power-up level of the registers in the Cyclone III to be either high or low. By default, when the device powers up, all registers in the device are cleared and the registered output signals drive low. Setting the register to power up high prevents the activation of another device's active-low input when the Cyclone III device is powered up. The register power-up high feature can also be used as a system reset signal when the Cyclone III device is powered up. If a register is set to power-up high, the register output will remain high until the asynchronous clear signal is asserted or a low data signal is clocked in. You can use the asynchronous preset signal to ensure that the register output remains high after the device is powered up. Set the register in the Quartus II design to power up high before design compilation.

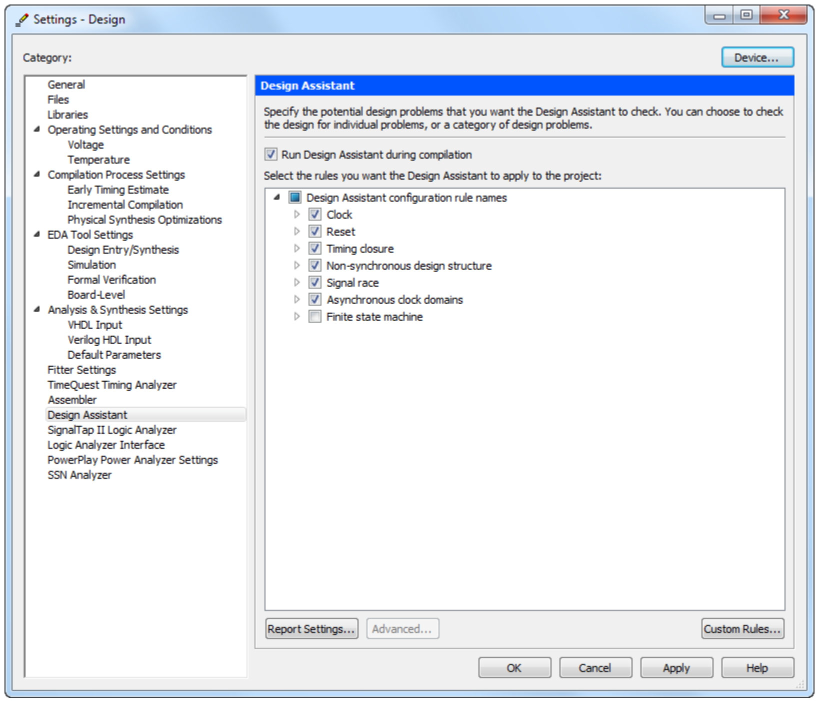
In the Quartus II integrated synthesis, you can apply the **Power-Up Level** logic option in the **Assignment Editor**, with a Tcl assignment, or create an `altera_attribute` assignment in your source code.

Design Assistant

The Quartus II Design Assistant analyzes the reliability of a design based on Altera-recommended design guidelines or design rules during design compilation. The Design Assistant checks rules related to areas such as clocks, resets, timing closure, and non-synchronous design structure. You can select the areas you want the Design Assistant to check and the Design Assistant will report any design violation based on the settings you specified. The HardCopy rules are not applicable to designs targeting Cyclone III devices.

To turn on the Design Assistant, on the Assignments menu, click **Settings**. Select the design rules under **Design Assistant**. Figure 7 shows the areas the Design Assistant checks.

Figure 7. Design Assistant Rules



Settings - Design

Category:

- General
- Files
- Libraries
- Operating Settings and Conditions
  - Voltage
  - Temperature
- Compilation Process Settings
  - Early Timing Estimate
  - Incremental Compilation
  - Physical Synthesis Optimizations
- EDA Tool Settings
  - Design Entry/Synthesis
  - Simulation
  - Formal Verification
  - Board-Level
- Analysis & Synthesis Settings
  - VHDL Input
  - Verilog HDL Input
  - Default Parameters
- Fitter Settings
- TimeQuest Timing Analyzer
- Assembler
- Design Assistant
- SignalTap II Logic Analyzer
- Logic Analyzer Interface
- PowerPlay Power Analyzer Settings
- SSN Analyzer

Design Assistant

Specify the potential design problems that you want the Design Assistant to check. You can choose to check the design for individual problems, or a category of design problems.

[✓] Run Design Assistant during compilation

Select the rules you want the Design Assistant to apply to the project:

- Design Assistant configuration rule names
  - [✓] Clock
  - [✓] Reset
  - [✓] Timing closure
  - [✓] Non-synchronous design structure
  - [✓] Signal race
  - [✓] Asynchronous clock domains
  - [ ] Finite state machine

For more information about the Design Assistant, refer to the *Recommended Design Practices* chapter in the *Quartus II Handbook*.

Planning for Hierarchical and Team-Based Design

The Quartus II incremental compilation feature preserves the results and performance of the unchanged logic in your design as you make changes elsewhere, allowing you to perform more design iterations, achieve timing closure efficiently and reduce compilation time when changes are made to certain partitions of the design.

In an incremental compilation flow, a large design is split into smaller partitions that can be designed separately and independently to simplify the design process and reduce compilation time. Good partition and floorplan design helps lower-level design blocks meet top-level design requirements, reducing the time spent integrating and verifying the timing of the top-level design.

For more information about using the incremental compilation flows in the Quartus II software, as well as important guidelines for creating design partitions and a design floorplan, refer to the *Quartus II Incremental Compilation for Hierarchical and Team-Based Design* chapter in the *Quartus II Handbook.*

Design Partitions

When partitioning a design for an FPGA, register the partition of the I/O boundaries to keep critical timing paths inside one partition that can be optimized independently. When the design partitions are specified, you can use the **Incremental Compilation Advisor** to ensure that the partitions meet Altera's recommendations.

Having the source code for each design block in a separate file allows you to make changes to the block separately. If you use a third-party synthesis tool, create a separate **.vqm** or **.edf** netlist for each design partition in your synthesis tool. Refer to your synthesis tool documentation for information on the support for the Quartus II incremental compilation. Use the hierarchy in your design to provide more flexibility when partitioning. The top level of the hierarchy should have very little logic, and the lower-level design blocks contain the most logic.

Timing Budget and Resource Allocation

In a team-based design, if you optimize the lower-level partitions separately, any unregistered path that crosses between the partitions are not optimized as an entire path when the partitions are integrated. For each unregistered timing path that crosses between the partitions, make the timing assignments on the corresponding I/O path in each partition to constrain both ends of the path to the budgeted timing delay. The software optimizes the paths appropriately so the paths can meet the top-level design requirements when the partitions are connected together in the top-level design.

When performing incremental compilation, the software synthesizes each partition separately with no data about the resources used in other partitions. Allocating resource utilization among the design partitions for example logic, memory, multipliers, PLLs and global routing signals avoids any problems with conflicting resources when all the partitions are integrated.

Planning in Bottom-Up and Team-Based Flows

In bottom-up design flows, top-level project information, such as the pin locations, physical constraints, and timing requirements, should be communicated to the designers of the lower-level partitions before they start their design to avoid problems during system integration.

The system architect can plan the design partitions at the top level and use the Quartus II incremental compilation to communicate information to lower-level designers through automatically-generated scripts. The Quartus II software's **Generate bottom-up design partition scripts** option automates the process of transferring top-level project information to lower-level modules. The software provides a project manager interface for managing project information in the top-level design.

Creating a Design Floorplan

Create a design floorplan to avoid resource and location conflicts between the design partitions. Floorplan assignments are recommended for timing-critical partitions in top-down flows. You can use the Quartus II Chip Planner to create a design floorplan using LogicLock region assignments for each design partition. The floorplan editor enables you to view connections between regions, estimate physical timing delays on the chip, and move regions around the device floorplan. When you have compiled the full design, you can also view logic placement and locate areas of routing congestion to improve the floorplan assignments.

For more information about creating placement assignments in the design floorplan, refer to the *Analyzing and Optimizing the Design Floorplan with the Chip Planner* chapter in the *Quartus II Handbook.*

Power Optimization

This section describes the various Quartus II software features and design techniques that help to reduce your design power consumption.

Quartus II Power Optimization Features

The Quartus II software offers power-driven compilation to reduce your design core dynamic power. Depending on the design, power-optimized synthesis and fitting can help reduce dynamic power by an average of up to 16%. Altera recommends activating both the options to achieve minimal power consumption. However, you should prioritize your design timing constraint requirements over power optimization.

The Quartus II software includes the Power Optimization Advisor to provide recommendations to reduce power based on the current design project settings and assignments. Each recommendation includes a description, a summary of the effect and the actions required to make the appropriate settings. The recommendations are split into stages to show the order in which you should apply the settings. After making any of the recommended changes, recompile your design and run the PowerPlay Power Analyzer to check the change in your power results.

The Design Space Explorer (DSE) is an utility in the Quartus II software that can be used to find the optimal Quartus II software settings to minimize your design power consumption.

For more information on power-driven compilation, Power Optimization Advisor, and DES, refer to the *Power Optimization* chapter in the *Quartus II Handbook*, and the Quartus II Help.

Design Power Optimization Techniques

This section provides guidelines on design techniques that affect overall design power. The results of these techniques may be different from design to design.

Clock Power Management

Clock routing power is automatically optimized by the Quartus II software, which only enables those portions of the clock network that are required to feed downstream registers. Alternately, you can use clock control blocks to implement the clock enable signal. When a clock network is powered down, all the logic fed by that clock network does not toggle, thereby reducing the overall power consumption of the device.

For more information about how to use the clock control block, refer to the *Clock Control Block (ALTCLKCTRL) Megafunction User Guide*.

Reducing Memory Power Consumption

The key to reducing memory power consumption is to reduce the number of memory clocking events. This can be achieved through the use of the clock control blocks on the clock network or on per memory basis through the use of the clock enable signals on the memory port. The clock enable signal enables the memory only when necessary and shuts it down for the rest of the time, reducing overall memory. You can use the Quartus II MegaWizard Plug-In Manager to create the enable signals by selecting the **Clock enable signal** option when generating the memory block function.

Reducing I/O Power

The dynamic power consumed in the I/O buffer is proportional to the load capacitance, output transition frequency, and output voltage swing. Lower the load capacitance to reduce dynamic power consumption.

Non-terminated I/O standards such as the **LVTTL** and **LVCMOS** have a rail-to-rail output voltage swing equal to the V_{CCIO} supply voltage. Use lower voltage I/O standard to reduce dynamic power. For high speed applications, use voltage-reference I/O standards, such as the **SSTL**. The output voltage swings by an amount smaller than V_{CCIO} around the reference voltage, so dynamic power is lower than that of **LVTTL** or **LVCMOS** under the same conditions.

However, voltage-reference I/O standards dissipate significant static power because current is constantly driven into the termination network. Use the lowest drive strength that meets your speed and waveform requirements to minimize static power. On the contrary, **LVTTL** and **LVCMOS** consume little static power.

Note that the power dissipated in the external termination network is not included in the EPE or PowerPlay Power Analyzer calculations. Make sure that you include it separately in your system power calculations.

Pipelining and Retiming

A design that has many glitches consumes more power because of faster switching activity. Pipelining by inserting flip flops into long combinational paths can reduce design glitches. Flip flops do not allow glitches to propagate through combinational paths, resulting in reduced power dissipation in combinational logic. However, if there are not many glitches in your design, pipelining may increase power consumption due to the addition of unnecessary registers.

Architectural Optimization

Use specific device architecture features to reduce power consumption. For example, use the dedicated DSP block available in the Cyclone III device in place of LEs to perform arithmetic-related functions; build large shift registers from RAM-based FIFO buffers instead of building the shift registers from the LE registers.

I/O Considerations

Cyclone III devices offer increased system integration with enhancements in I/O flexibility and improvements in features. I/O capabilities of the FPGA device, board layout guidelines, and signal integrity concerns significantly influence pin location decisions and other types of assignments for each of your design pins. You can optimize I/O resources for performance and cost while avoiding undesirable signal issues with the tools and taking into account considerations described in this section.

I/O Tools

I/O design flow includes creating pin-related assignments and validating them against pin placement guidelines. Use the Pin Planner and Assignment Editor tools in the Quartus II software to help you through the pin-related assignment creation and editing. I/O Assignment Analysis helps you to check the legality of the pin assignments. The following subsections discuss the tools used to create and edit pin-related assignments.

Pin Planner

The Pin Planner is used to create and edit pin-related assignments. Use the Pin Planner Package View to make pin location and other assignments using a device package view instead of pin numbers. With the Pin Planner, you can visually identify I/O banks, V_{REF} groups, and differential pin pairings. The visual representation of pins in the Package View helps you to minimize signal breakout congestion on board layout when making pin location assignments.

Complementing the Pin Planner is the Pad View window, which displays the pads in the order around the silicon die. Package pins are connected to pads located on the perimeter of the top metal layer of the silicon die. Note the pad location where your pins were assigned, because some pin placement rules describe pad placement restrictions. Use the Pad View window to guide your pin placement decisions to maintain good signal integrity for the interface in your design.

The Pin Migration View in the Pin Planner tool shows the pins that will change function in a migration device if you select one or more migration devices for your project. It helps you to identify the differences in pins which can exist between migration devices.

For details about using the Package View, Pad View, and Pin Migration View in the Pin Planner tool, refer to the *Using the Pin Planner* section of the *I/O Management* chapter in the *Quartus II Handbook.*

Assignment Editor

While the Pin Planner provides an intuitive graphical representation of the targeted device, the Assignment Editor provides a spreadsheet-like interface that allows you to create and change all pin-related assignments.

Pin-Related Assignments

Selectable I/O capabilities increase the flexibility of Cyclone III devices to connect to other devices in many low-cost applications. Integration of various I/O-related features further reduces the cost of any application with Cyclone III devices. Notable features include current strength control, slew rate control, open-drain, bus hold, PCI clamp diode, weak pull-up, series on-chip termination, and dedicated output buffer with programmable pre-emphasis. The following sub-sections discuss the considerations involved when you use some of the features.

For more information about each I/O features, refer to the *I/O Features in the Cyclone III Device Family* chapter of the *Cyclone III Device Handbook*.

Current Strength and Slew Rate Control

Each I/O standard supports a range of programmable current strength. Select a suitable current strength setting to meet the performance target for your I/O. Note that higher current strength increases not just I/O performance, but also noise on the interface. Ensure that the output buffer current strength is sufficiently high without causing excessive overshoot or undershoot that violates voltage threshold parameters for the I/O standard. At the same time, the setting should not be too low to cause stair-step response, which increases edge rate significantly and results in incorrect clocking if the pin is used as a clock source.

Run simulation with Cyclone III IBIS models to determine if your current strength setting meets the desired performance and the receiver's input specification.

Cyclone III devices observe a maximum AC voltage of 4.1 V on all the inputs. To ensure the device reliability, maximum voltage overshoot seen at the Cyclone III receiver should not exceed the specification.

For more details, refer to *AN 447: Interfacing Cyclone III and Cyclone IV Devices with 3.3/3.0/2.5-V LVTTL/LVCMOS I/O Systems.*

If a large data bus is used in which pins may switch simultaneously, Altera recommends turning on the slew rate control to reduce simultaneous switching output (SSO) effects such as crosstalk and ground bounce. Slew rate control is only available in single-ended I/O standards with 8 mA current strength or higher.

Series On-Chip Termination

Series On-Chip Termination (OCT) in Cyclone III devices can be implemented in two ways; OCT without calibration, and OCT with calibration. Selection factor is mainly driven by driver impedance accuracy versus cost of additional external components. OCT without calibration uses driver output capabilities to match its impedance to the impedance of the transmission line. For higher impedance matching accuracy, implement OCT with calibration, which takes into account the voltage and temperature variation. The calibration process starts at the end of configuration and completes before user mode operation. While providing better accuracy, you must connect two external resistors of 25 Ω ±1% or 50 Ω ±1% to a pair of R_{UP} and R_{DN} pins for the calibration block. A calibration block is available on each side of the device. Calibration to other impedance values other than the recommended 25 Ω or 50 Ω is possible, which is limited only by range of buffer impedance.

For more information on the support and implementation, refer to the *On-Chip Termination Support* section in the *I/O Features in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook.*

Dedicated Differential Output Buffer with Programmable Pre-Emphasis

Cyclone III devices provide dedicated differential output buffers on left and right I/O banks. These buffers are capable of transmitting **LVDS** signals at up to 840 Mbps without requiring external resistors. Similarly, you can transmit **RSDS**, **mini-LVDS**, and **PPDS** with dedicated transmitter buffers with no external resistors required at different speeds. Only a differential termination resistor is required on the receiver. The top and bottom I/O banks can transmit differential I/O standards, but require an additional resistor network on the transmitter. You can implement the serializer and deserializer (SERDES) for **LVDS** using the altlvds megafunction in the Quartus II software. SERDES is implemented in core logic as Cyclone III devices do not have a dedicated SERDES circuit.

For more information on the support and implementation, refer to the *High-Speed Differential Interfaces in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Programmable pre-emphasis is supported on dedicated differential output buffers to maximize the data eye opening at the far-end receiver. High speed interfaces that use long traces typically suffer high frequency signal attenuation in the transmission media due to skin effect and dielectric loss. Programmable pre-emphasis reduces the attenuation, by boosting high frequency component at each transition in the data stream. To enable this in the Quartus II software, set Programmable Pre-emphasis assignment for the pin to **On** in the Assignment Editor.

PLL Considerations

Post PLL Design

After designing the PLL, consider the following topics before designing the board.

Choosing the PLL

For Cyclone III devices, there are four identical PLLs (except EP3C5 and EP3C10, which have only two identical PLLs). You can choose to use a specific PLL or let the Quartus II software choose the best PLL to use. For the first option, you must choose the PLL where you have available suitable dedicated input and output clock pins. The PLL input clock must be either dedicated input clock pins or another PLL output. The PLL input clock cannot be driven from internal logic. The PLL output clock should be connected to its dedicated clock output pin for optimum routing (only applies to c0), user I/O, or global clock (GCLK).

To obtain the locations of the PLLs and their corresponding pins, refer to the corresponding device pin-outs and *Clock Networks and PLLs in Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Pin Assignment

The following are the recommended pin assignments in the Quartus II software for the PLL input and output ports.

- `inclk[1,0]` – dedicated clock input pin (`CLK[15..0]`)
- `clkswitch`, `areset`, `pfdena`, `scandata`, `scanclk`, `configupdate`, `scanclkena`, `phasecounterselect[2..0]`, `phaseupdown`, `phasestep` – User I/O or logic array signal
- `c[4..0]` – dedicated PLL clock output pin (only c0) or user I/O or logic array signal (c0-c4)
- `clkbad[1,0]`, `locked`, `activeclock`, `scandone`, `scandataout`, `phasedone` – User I/O or logic array signal

To find the correct pin, refer to the Device Pin-Outs page of the Literature section of the Altera website (www.altera.com).

Simulation

The altpll megafunction supports behavioral and timing simulation.

For more information about simulation support and methods, refer to “Section I. Simulation” in *Volume 3: Verification* of the *Quartus II Handbook*.

Configuration Software Settings

This section covers several configuration options that you can set in the Quartus II software before you compile to generate configuration or programming files. Your board and system design is affected by these settings and pins.

Optional Configuration Pins

Prior to generating the configuration files, you should take the following optional configuration pins into your consideration according to your board and system design. You can enable the following optional configuration pins on the **General** tab of the **Device and Pin Options** dialog box.

CLKUSR

By default, the initialization clock source is from the 10 MHz (typical) internal oscillator. The advantage of using the internal oscillator is that you do not need to send additional clock cycles from an external source to the `CLKUSR` pin during the initialization stage. However, if you have to synchronize the initialization of the multiple devices or to delay the initialization, a Cyclone III device provides the flexibility to control when your device enters user mode using `CLKUSR` pin. The **Enable user-supplied start-up clock** (`CLKUSR`) option in the Quartus II software allows you to select which clock source is used for initialization; either the internal oscillator or external clocks provided on the `CLKUSR` pin. Supplying an external clock on `CLKUSR` does not affect the configuration process. After all the configuration data is accepted and the `CONF_DONE` goes high, Cyclone III devices require 3,185 clock cycles to initialize properly and enter user mode. Cyclone III devices support a `CLKUSR` f_{MAX} of 133 MHz. When you are not using the `CLKUSR` pin to supply external clock, you can use it as a user I/O pin.

INIT_DONE

To check if the Cyclone III device has completed initialization and is in user mode, you can monitor the `INIT_DONE` pin. A low-to-high transition in this `INIT_DONE` pin indicates the end of initialization and the start of user mode. The `INIT_DONE` pin is an optional pin and can be turned on in the Quartus II software through the **Enable INIT_DONE output** option. The `INIT_DONE` pin is an open-drain output and requires an external 10 KΩ pull up resistor to V_{CC}.

Auto Restart Configuration

If you would like to begin a reconfiguration to the Cyclone III device when configuration error occurs, you can enable the **auto-restart configuration after error** option in Quartus II software. In the case where configuration error occurs, the Cyclone III device drives `nSTATUS` low, which resets itself internally. The Cyclone III device releases its `nSTATUS` pin after a reset time-out period. The `nSTATUS` pin is then pulled to V_{CC} by a pull-up resistor, indicating that reconfiguration can begin.

Estimating Configuration File Size

The configuration file size could be one of your main concern in the configuration process. To estimate the configuration file size of Cyclone III devices, you may convert your configuration file in uncompressed raw binary file (**.rbf**) which provides the approximate uncompressed configuration file sizes for Cyclone III devices. To calculate the amount of storage space required for multiple device configuration, add the file size of each device together. Cyclone III devices can receive a compressed configuration bitstream and decompress this data in real-time, reducing storage requirements and configuration time in active serial and passive serial configuration schemes.

For more information on the uncompressed raw binary file (**.rbf**) sizes for Cyclone III devices, refer to the *Configuration, Design Security, and Remote System Upgrades* in the Cyclone III Device Family chapter in the *Cyclone III Device Handbook.*

Use the data on the uncompressed raw binary file size only to estimate the file size before design compilation. Different configuration file formats, such as Hexadecimal (**.hex**) or Tabular Text File (**.ttf**) format, have different file sizes. However, for any specific version of the Quartus II software, any design targeted for the same device has the same uncompressed configuration file size. If you are using compression, the file size can vary after each compilation because the compression ratio is dependent on the design.

Converting Programming File

To store the Cyclone III data into configuration devices, you can convert the default SRAM Object File (**.sof**) data into a different file format and program the configuration device. You can set up the software to generate additional programming files during compilation in the **Device and Pin Options** dialog box. To convert the **.sof** file, on the File menu, click **Convert Programming Files**.

The Quartus II software supports data conversion into the following format:

- Programmer Object File (**.pof**),
- Hexadecimal (Intel-Format) Output File (**.hexout**),
- Raw Binary File (**.rbf**),
- Tabular Text File (**.ttf**),
- Raw Programming Data (**.rpd**) file, and
- JTAG Indirect Configuration File (**.jic**).

The **.pof** and **.jic** files are used with the Quartus II Programmer to program the configuration devices while the **.hexout**, **.rbf**, **.ttf**, and **.rpd** files are used with other programmers.

When working with multi-device configuration chains, you should combine each device's **.sof** file into one configuration file in the **Convert Programming Files** dialog box. When generating the configuration file, ensure that the configuration files are in the same order as the devices on the board.

For more information about setting device configuration options or creating configuration files, refer to relevant documents at the Documentation: Configuration Devices page on the Altera website.

For more information about programming file formats and using the programmer, refer to the *Quartus II Programmer* chapter in the *Quartus II Handbook*.

Validating Pin Placement

After creating pin-related assignments, you must validate the assignments against rules specific to the Cyclone III device. This section discusses the validation and considerations about pin placement.

I/O Assignment Analysis

The **Start I/O Assignment Analysis** command allows you to check the legality of your I/O assignments before, during, or after you compile your design. If design files are available, you can use this command to perform thorough legality checks on your design's I/O pins and surrounding logic. These checks include proper reference voltage pin usage, valid pin location assignments, and acceptable mixed I/O standards. Run the analysis each time you add or modify a pin-related assignment.

For more information about the **Start I/O Assignment Analysis** command and its design flow, refer to the *I/O Management* chapter in the *Quartus II Handbook*.

DC Guidelines

Sourcing or sinking large amounts of steady current from output pins can damage the device due to electromigration, where the movement of the conducting metal's atoms is caused by the large amount of electric current flowing through it. Some examples pull a high output pin to ground or connect a low output pin to V_{CC} directly. The maximum DC current a Cyclone III I/O pin can sink is 25 mA and source is 40 mA. If certain pins have to be pulled high or low, pull them through external resistors.

In addition, there is a limit for total current sink or source for a set of consecutive output pads. As this relates to the location of pads, use the Quartus II software to check for this violation by providing the pin’s maximum DC current. To set this in the Quartus II software, go to the Assignment Editor and enter the maximum DC current to Electromigration Current assignment for the pin. To view the DC restrictions in the Quartus II software, on the Assignments menu, click **Device**. Click **Device & Pin Options** and select the **Pin Placement** tab. The restrictions are listed in **Electromigration** window.

For the specification on current limit, refer to the *Pad Placement and DC Guidelines* section in the *I/O Features in the Cyclone III Device Family* chapter in the *Cyclone III Device Handbook*.

Verification

This section provides information about verifying your design in the areas of timing, power consumption, and simulation.

Timing Closure and Verification

After compiling the completed logic design, check the device utilization and verify that the design meets its timing requirements. Analyze the messages generated during the compilation to check for any potential problems. If required, you can use the Quartus II software to optimize the design's resource utilization and achieve timing closure, preserve the performance of unchanged design blocks, and reduce compilation time for future iterations. You can also verify the design functionality with simulation. This section provides the guidelines for these stages of the compilation flow.

Timing Constraints and Analysis

Timing constraints are critical to ensure that the designs meet their timing requirements, as timing requirements represent actual design requirements that must be met for the device to operate correctly.

The Quartus II software includes the Quartus II TimeQuest Timing Analyzer, a powerful ASIC-style timing analysis tool that validates the timing performance of all logic in your design. It supports the industry-standard Synopsys Design Constraints (SDC) format timing constraints, and has an easy-to-use GUI with interactive timing reports. It is ideal for constraining high-speed source-synchronous interfaces and clock multiplexing design structures.

The software also supports static timing analysis in the industry-standard Synopsys PrimeTime software. Specify the tool in the **New Project Wizard** or the **EDA Tools Settings** page of the **Settings** dialog box to generate the required timing netlist.

A comprehensive static timing analysis includes analysis of register-to-register, I/O, and asynchronous reset paths. It is important to specify the frequencies and relationships for all clocks in your design. Use input and output delay constraints to specify external device or board timing parameters. The Timing Analyzer performs static timing analysis on the entire system, using data required times, data arrival times, and clock arrival times to verify circuit performance and detect possible timing violations. You can use the `report_datasheet` command to generate a datasheet report that summarizes the I/O timing characteristics of the entire design. For asynchronous control signals; for example clear, reset and load signal of a register, perform the recovery and removal timing analysis. Use the `-recovery` and `-removal` options together with the `report_timing` command for the TimeQuest Timing Analyzer.

Multicycle paths are data paths that require more than one clock cycle to latch data at the destination register. For example, a register only captures data on every second or third rising clock edge. For multicycle paths, the default setup and hold relationships can be modified with the `set_multicycle_path` command to accommodate the system requirement.

For more information about timing analysis, refer to the following documents:

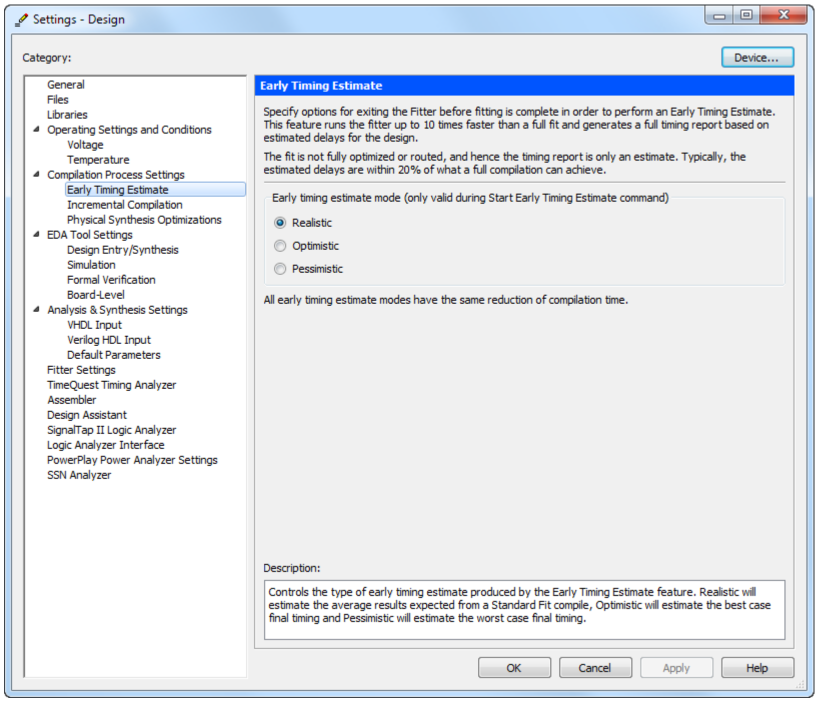
- *Timing Analysis Overview* chapter in the Quartus II Handbook
- *The Quartus II TimeQuest Timing Analyzer* chapter in the *Quartus II Handbook.*

Early Timing Estimation

You can estimate your design's timing with the Quartus II Early Timing Estimation. The Early Timing Estimation provides preliminary timing information without performing a full compilation of your design. To save compilation time, the fitter does not perform a full optimization, thus the timing information is only an estimate. On the Processing menu, point to **Start** and click **Start Early Timing Estimate** to generate initial compilation results after you have run analysis and synthesis.

You can set the **Early Timing Estimation Level** to **Realistic**, **Optimistic**, or **Pessimistic,** as shown in Figure 8. Typically, the estimated delays are within 10% of those obtained with a full fit when the realistic setting is used. Early Timing Estimate generates a full timing report based on the early placement and routing delays.

Figure 8. Early Timing Estimate



Specify options for exiting the Fitter before fitting is complete in order to perform an Early Timing Estimate. This feature runs the fitter up to 10 times faster than a full fit and generates a full timing report based on estimated delays for the design.

The fit is not fully optimized or routed, and hence the timing report is only an estimate. Typically, the estimated delays are within 20% of what a full compilation can achieve.

Early timing estimate mode (only valid during Start Early Timing Estimate command)

[✓ Realistic] Optimistic Pessimistic

All early timing estimate modes have the same reduction of compilation time.

Description:

Controls the type of early timing estimate produced by the Early Timing Estimate feature. Realistic will estimate the average results expected from a Standard Fit compile, Optimistic will estimate the best case final timing and Pessimistic will estimate the worst case final timing.

Checking Design Timing

To ensure that your design works, check the timing report to ensure that there is no timing violation. It is important that the internal and I/O timings are met. The f_{MAX} from the timing report indicates the maximum frequency at which your design can operate. To ensure that the Cyclone III device interfaces correctly with other devices on the board, check the t_{SU}, t_{CO}, t_H, and t_{PD} values from the timing report. For example, when the Cyclone III device receives data from another device, make sure that the t_{SU} and t_H are not violated when the data is being clocked into the Cyclone III device register. Violating t_{SU} or t_H can cause the register output to become metastable. Metastable output can become incorrect for short durations, thus affecting the design functionality.

When the Cyclone III device transmits registered data to another device, check the t_{CO} of the Cyclone III device to estimate the data arrival time for the external device.

For more information about the TimeQuest Timing Analyzer, refer to *The Quartus II TimeQuest Timing Analyzer* chapter in the *Quartus II Handbook*.

For more information about metastability, refer to *Understanding Metastability in FPGAs* white paper.

I/O Timing Analysis

Perform I/O analysis with the TimeQuest Timing Analyzer to ensure that the Cyclone III device meets the timing requirement when interfacing with external devices on the board. If you know the maximum or minimum delay of a signal from a register of an external device to a specified input or bidirectional pin on the Cyclone III device relative to a specified clock source, use the **Input Delay** assignment.

For the registered output from the Cyclone III device, you can make the **Output Delay** assignment for maximum or minimum delay of a signal from the registered output of the Cyclone III device to the registered input of the external device, relative to a specified clock source.

Skew Management

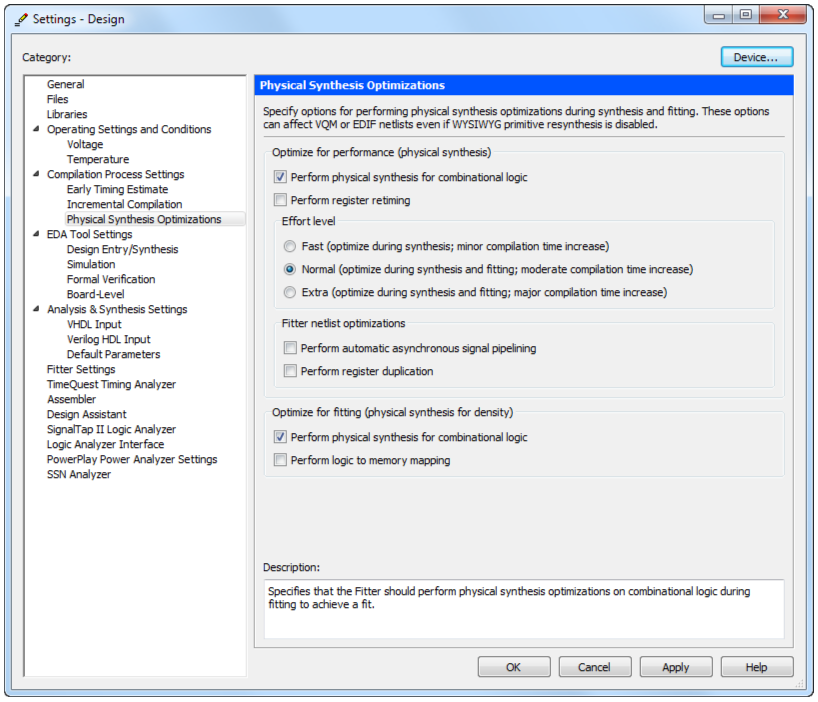
Skew refers to the arrival time difference of a signal at two different destinations. For synchronous designs, we have clock and data skew due to the different path length of the clock and data signals. The effect of the skew is more significant on high speed signals, due to the short time period of the signals. Using the clock network for these high speed signals results in lower clock and data skew.

Use the Quartus II **Maximum Clock Arrival Skew** assignment to specify the maximum allowable clock arrival skew between a clock signal and various destination registers. For data signals, use the **Maximum Data Arrival Skew** assignment to specify the maximum allowable data arrival skew to various destination registers or pins. When these assignments are used, the Quartus II software determines the timing difference between the longest clock or data path, and the shortest clock or data path so that the Fitter attempts to meet the requirement.

Area and Timing Optimization

Physical synthesis optimizations make placement-specific changes to the netlist that improve results for a specific Altera device. You can select the options under the **Optimize for performance (physical synthesis)** and **Optimize for fitting (physical synthesis for density)** sections in the Physical Synthesis Optimizations page under the Compilation Process Settings category, as shown in Figure 9.

Figure 9. Physical Synthesis Optimizations



Physical Synthesis Optimizations

Specify options for performing physical synthesis optimizations during synthesis and fitting. These options can affect VQM or EDIF netlists even if WYSIWYG primitive resynthesis is disabled.

Optimize for performance (physical synthesis)

- [✓] Perform physical synthesis for combinational logic
- [ ] Perform register retiming

Effort level

- ○ Fast (optimize during synthesis; minor compilation time increase)
- ● Normal (optimize during synthesis and fitting; moderate compilation time increase)
- ○ Extra (optimize during synthesis and fitting; major compilation time increase)

Fitter netlist optimizations

- [ ] Perform automatic asynchronous signal pipelining
- [ ] Perform register duplication

Optimize for fitting (physical synthesis for density)

- [✓] Perform physical synthesis for combinational logic
- [ ] Perform logic to memory mapping

Description:

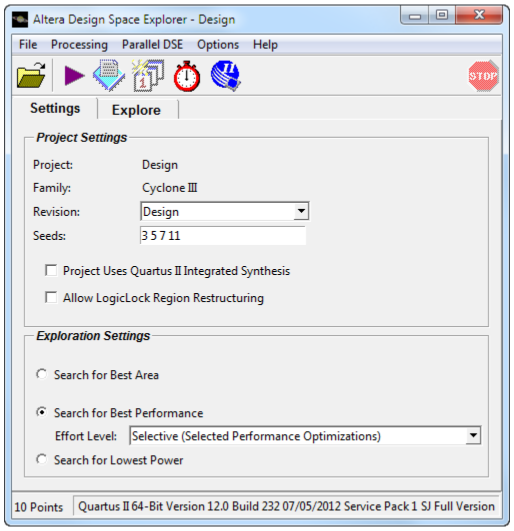
Specifies that the Fitter should perform physical synthesis optimizations on combinational logic during fitting to achieve a fit.

These options typically increase compilation time significantly but can provide significant improvements to the compilation result. If you turn on these options, ensure that they do improve the results for your design.

For more information, refer to the *Netlist Optimizations and Physical Synthesis* chapter in the *Quartus II Handbook.*

The Design Space Explorer (DSE) is a utility that automates the process of finding the optimal collection of Quartus II software settings for your design. The **Search for Best Performance** and **Search for Best Area** options under **Exploration Settings**, as in Figure 10, use a predefined exploration space to target design performance or area improvements with multiple compilations.

Figure 10. Design Space Explorer Window



Project Settings

Project: Design
Family: Cyclone III
Revision: Design
Seeds: 3 5 7 11

Project Uses Quartus II Integrated Synthesis
Allow LogicLock Region Restructuring

Exploration Settings

Search for Best Area
[✓ Search for Best Performance]
Effort Level: Selective (Selected Performance Optimizations)
Search for Lowest Power

10 Points | Quartus II 64-Bit Version 12.0 Build 232 07/05/2012 Service Pack 1 SJ Full Version

For more information, refer to the *Power Optimization* chapter in the *Quartus II Handbook*.

The Optimization Advisors provide guidance in making settings that optimize your design. On the Tools menu, point to **Advisors**, and click **Resource Optimization Advisor**, or **Timing Optimization Advisor**. Evaluate the options and choose the settings that best suit the requirements.

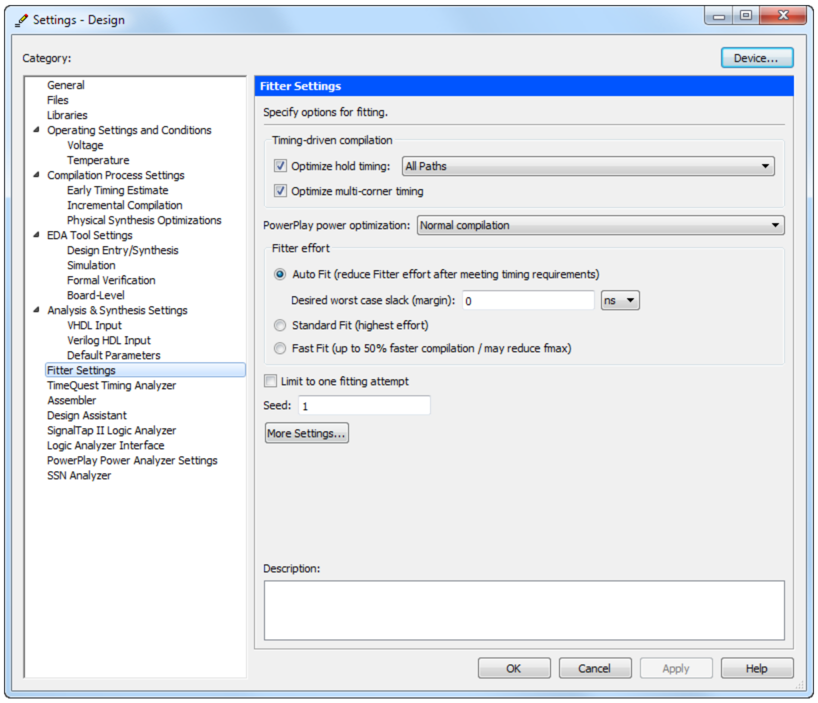
For more information about the various timing optimization settings in the Quartus II software as well as the timing optimization techniques, refer to the *Quartus II Integrated Synthesis* chapter in the *Quartus II Handbook* and the *Area and Timing Optimization* chapter in the *Quartus II Handbook*.

Recommended Timing Optimization and Analysis Assignments

The assignments and settings described in this section are not turned on in the software by default for all designs, but are important for large designs.

Turn on **Optimize fast-corner timing** on the **Fitter Settings** page in the **Settings** dialog box, as in Figure 11. When this option is on, the design is optimized to meet its timing requirements at the **Fast Timing** process corner and operating condition, as well as at the **Slow Timing** corner. Therefore, turning on this option helps create a design implementation that is more robust across process, temperature, and voltage variations.

Figure 11. Settings Dialog Box



Fitter Settings

Specify options for fitting.

Timing-driven compilation

[✓] Optimize hold timing: All Paths

[✓] Optimize multi-corner timing

PowerPlay power optimization: Normal compilation

Fitter effort

[●] Auto Fit (reduce Fitter effort after meeting timing requirements)

Desired worst case slack (margin): 0 ns

[○] Standard Fit (highest effort)

[○] Fast Fit (up to 50% faster compilation / may reduce fmax)

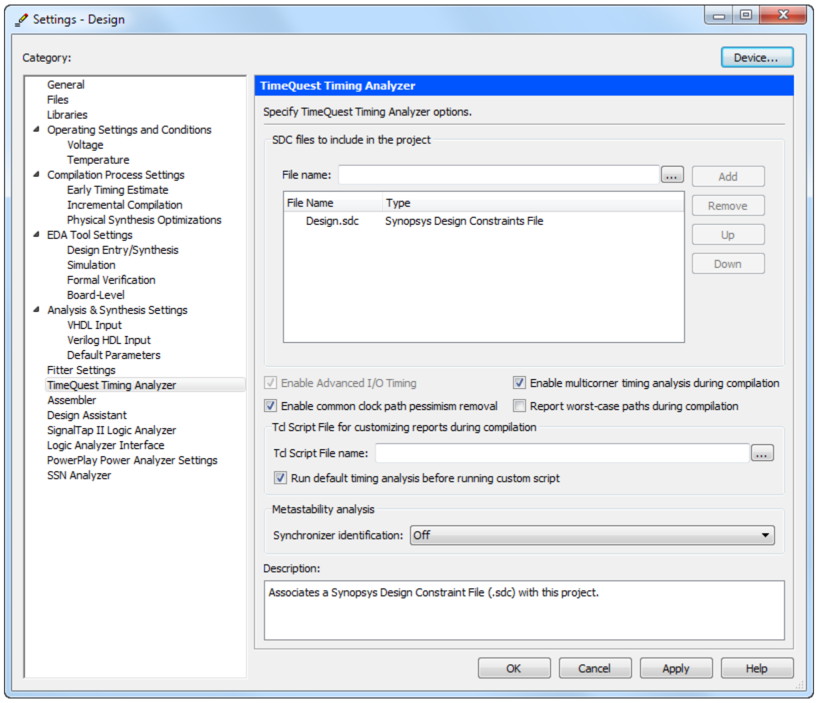
[ ] Limit to one fitting attempt

Seed: 1

If your design has problems meeting hold times, there may be design problems such as gated clocks adding delay in your clock lines. Otherwise, set the **Optimize hold timing** option to **All paths** to optimize the register-to-register hold time paths.

Turn on **Enable multicorner timing analysis** on the **TimeQuest Timing Analyzer** page under **Timing Analysis Settings** in the **Settings** dialog box, as in Figure 11, or use the `--multicorner` command line option for TimeQuest. This option directs the TimeQuest Timing Analyzer to analyze the design and generate slack reports for the slow and fast corners.

Figure 12. TimeQuest Timing Analyzer Settings



TimeQuest Timing Analyzer

Specify TimeQuest Timing Analyzer options.

SDC files to include in the project

File name:

File Name	Type
Design.sdc	Synopsys Design Constraints File

[✓] Enable Advanced I/O Timing
[✓] Enable multicorner timing analysis during compilation
[✓] Enable common clock path pessimism removal
[ ] Report worst-case paths during compilation

Tcl Script File for customizing reports during compilation

Tcl Script File name:

[✓] Run default timing analysis before running custom script

Metastability analysis

Synchronizer identification: Off

Description:

Associates a Synopsys Design Constraint File (.sdc) with this project.

In your TimeQuest SDC constraints file, use the following recommended constraints as applicable to your design:

- `create_clock`, `create_generated_clock` – specify the frequencies and relationships for all clocks in your design.
- `derive_pll_clocks` - creates generated clocks for all PLL outputs, according to the settings in the PLL megafunctions. Specify multicycle relationships for **LVDS** transmitters or receiver deserialization factors.
- `set_input_delay`, `set_output_delay` - specify external device or board timing parameters.
- `derive_clock_uncertainty` - automatically applies inter-clock, intra-clock, and I/O interface uncertainties. You can also user the `set_clock_uncertainty` constraint to specify clock uncertainty or skew for clocks or clock-to-clock transfers.
- `check_timing` - generates a report on any problem with the design or applied constraints, including missing constraints.

For more information on the SDC commands and clock uncertainties, refer to the *Quartus II TimeQuest Timing Analyzer* chapter in the *Quartus II Handbook*.

Device Resource Utilization Reports

After compilation, review the device resource utilization information to determine that there is enough logic for any potential design change in future. If your compilation results in a no-fit error, check the messages generated to understand what causes the no-fit.

To determine resource usage, refer to the **Flow Summary** section of the Compilation Report for a percentage representing the total logic utilization, which includes an estimation of resources that cannot be used due to existing connections or logic use.

More detailed resource information is available by viewing the reports under **Resource Section** in the **Fitter** section of the Compilation Report. There are also reports that describe some of the optimizations that occurred during compilation. For example, if you are using Quartus II integrated synthesis, the reports under the **Optimization Results** folder in the **Analysis & Synthesis** section describe information including registers that were removed during synthesis. This report can be useful when estimating device resource utilization for a partial design, to ensure that registers were not removed due to missing connections with other parts of the design.

Quartus II Messages

Each stage of the compilation flow generates messages, including informational notes, warnings, critical warnings and errors. Review these messages to check for any potential design problems. To understand the meaning of the messages, right click on the message and select **Help**. The Quartus II Help shows the meaning of the message.

For more information about messages and message suppression, refer to the *Managing Quartus II Projects* chapter in the *Quartus II Handbook.*

Preserving Performance

You can use the incremental compilation feature to preserve unchanged parts of your design, thus preserving performance and allowing you to reach timing closure more efficiently. For guidelines and references, refer to "Planning for Hierarchical and Team-Based Design" on page 35.

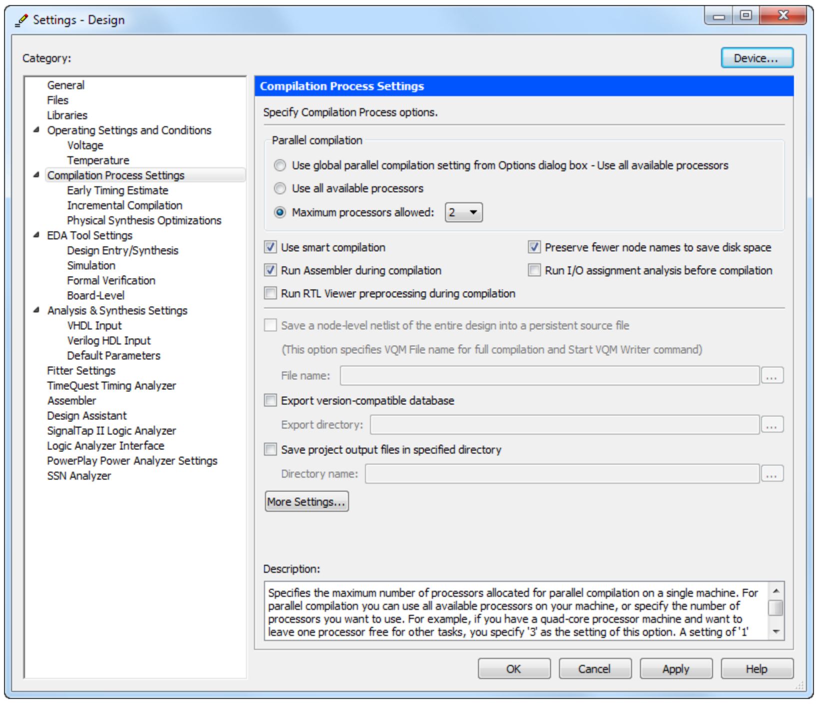
Reducing Compilation Time

You can speed up design iteration time by an average of 60% when making changes to the design with the incremental compilation feature.

For guidelines and references, refer to the "Planning for Hierarchical and Team-Based Design" on page 35.

The Quartus II software can run some algorithms in parallel to take advantage of multiple processors and reduce compilation time when more than one processor is available to compile the design. To set the number of processors available for a Quartus II compilation, specify the **Maximum processors allows for parallel compilation** on the **Compilation Process Settings** page of the **Settings** dialog box, as in Figure 13. The default value for the number of processors is 1, which disables parallel compilation.

Figure 13. Specifying the Number of Processors



Settings - Design

Category:

- General
- Files
- Libraries
- Operating Settings and Conditions
  - Voltage
  - Temperature
- Compilation Process Settings
  - Early Timing Estimate
  - Incremental Compilation
  - Physical Synthesis Optimizations
- EDA Tool Settings
  - Design Entry/Synthesis
  - Simulation
  - Formal Verification
  - Board-Level
- Analysis & Synthesis Settings
  - VHDL Input
  - Verilog HDL Input
  - Default Parameters
- Fitter Settings
- TimeQuest Timing Analyzer
- Assembler
- Design Assistant
- SignalTap II Logic Analyzer
- Logic Analyzer Interface
- PowerPlay Power Analyzer Settings
- SSN Analyzer

Compilation Process Settings

Specify Compilation Process options.

Parallel compilation

- ○ Use global parallel compilation setting from Options dialog box - Use all available processors
- ○ Use all available processors
- ● Maximum processors allowed: 2

- ✓ Use smart compilation
- ✓ Preserve fewer node names to save disk space
- ✓ Run Assembler during compilation
- ☐ Run I/O assignment analysis before compilation
- ☐ Run RTL Viewer preprocessing during compilation
- ☐ Save a node-level netlist of the entire design into a persistent source file

(This option specifies VQM File name for full compilation and Start VQM Writer command)

File name:

- ☐ Export version-compatible database

Export directory:

- ☐ Save project output files in specified directory

Directory name:

Description:

Specifies the maximum number of processors allocated for parallel compilation on a single machine. For parallel compilation you can use all available processors on your machine, or specify the number of processors you want to use. For example, if you have a quad-core processor machine and want to leave one processor free for other tasks, you specify '3' as the setting of this option. A setting of '1'

The Compilation Time Advisor provides guidance in making settings that reduce your design compilation time. On the Tools menu, point to **Advisors**, and click **Compilation Time Advisor**. Using some of these techniques to reduce compilation time can reduce the overall quality of results (QoR).

For more suggestions, refer to the *Area and Timing Optimization* chapter in the *Quartus II Handbook.*

Power Analysis

When your design is completed, use Quartus II PowerPlay Power Analyzer to calculate the power consumption of your design to ensure that the thermal and power supply budgets are not violated. You must successfully compile your design to use the PowerPlay Power Analyzer. Information about the design resources, placement and routing and I/O standard assigned to each I/O pin allow PowerPlay Power Analyzer to provide accurate power estimation.

To run power analysis, from the Processing menu, click **PowerPlay Power Analyzer Tool**. You must specify the source of input data and operating conditions.

The input data can derive from either signal activity data from simulation results or a user-defined default toggle rate and vectorless estimation. The signal activities used for the analysis must be representative of the actual operating behavior over realistic time period. For the most accurate power estimation, use gate-level simulation results with a Value Change Dump (**.vcd**) output file from the ModelSim®-Altera or a third-party simulation tool. Use the recommended simulator settings (such as glitch filtering) to ensure good results.

The operating conditions, including core voltage, device power characteristic, ambient and junction temperature, cooling solution and board thermal model can be set in the **Operating Settings and Conditions** page in the **Settings** dialog box.

The PowerPlay Power Analyzer tool calculates the core and I/O dynamic power, static power and current consumed from the power supplies. The tool also provides a summary of the signal activities used for analysis and a confidence metric of the data sources for the signal activities.

☞ Note that the report is a power estimate based on the data provided, and is not a power specification. Always refer to the data sheet for your device.

For more information about the PowerPlay Power Analyzer tool, refer to the *PowerPlay Power Analysis* chapter in the *Quartus II Handbook*.

Simulation

Perform simulation to verify your design works correctly. For designs that consist of multiple modules, simulation ensures that each module is functionally working before being combined together. Debugging your design at lower level saves you debugging time. You can monitor pins or registered signals through simulation.

You can perform simulation on your Cyclone III design using various simulation tools. Among the commonly used third-party simulation tools are the Mentor Graphics ModelSim®, Synopsys VCS and Cadence NC-Sim. You can also use the ModelSim-Altera Edition or ModelSim-Altera Starter Edition. Both ModelSim-Altera Edition and ModelSim-Altera Starter Edition support the Cyclone III family.

These third-party simulation tools support functional RTL, post-synthesis and gate-level simulation. Perform functional RTL simulation before gate-level simulation or post-synthesis simulation as functional RTL simulation verifies the functionality of your design before synthesis and place-and-route.

Post-synthesis simulation verifies the functionality of a design after synthesis has been performed. You can create a post-synthesis netlist in the Quartus II software and use this netlist to perform post-synthesis simulation with the third-party simulation tools. After the post-synthesis version of the design is verified, you can then proceed with the place-and-route for your design in the target Cyclone III device using the Quartus II Fitter. Use the EDA Netlist Writer of the Quartus II software to generate the simulation netlist for the specific third-party simulation tool.

Gate-level timing simulation is a post place-and-route simulation to verify the operation of the design after the worst-case timing delays have been calculated.

Similarly, the ModelSim-Altera software supports both functional and timing simulation. Perform functional simulation at the beginning of your design flow to check the functionality or logical behavior of your design. You do not need to compile your design, simply generate the functional simulation netlist of your design which does not contain timing information to perform functional simulation.

Timing simulation uses the timing netlist generated by the Timing Analyzer when you compile your design. Timing simulation takes the delay of different device blocks as well as place-and-route information into consideration, thus is more accurate than the functional simulation. Perform timing simulation at the top-level design, at the end of your design flow to ensure that your design works in the targeted device.

For more information about simulating your design with simulation tools, refer to "Section I. Simulation" in *Volume 3: Verification* of the *Quartus II Handbook*.

Design Debugging

Using incremental compilation when performing on-chip debugging reduces the compilation time as changes are usually made only to certain parts of the design.

For more information on the on-chip debugging features available in the Quartus II software, refer to the "Planning for On-Chip Debugging" section under "Early System Planning" on page 3.

Testing

Boundary-Scan Test

One of the most common board-level tests on the devices in production environment is the boundary-scan test. The Cyclone III device fully supports IEEE 1149.1 (JTAG) boundary-scan testing. A boundary-scan test allows you to test pin connections at board level without using physical test probes while the device is operating normally. To perform the boundary-scan test, you must have the boundary-scan description language (BSDL) file of the device.

Use the BSDL files from the Altera website to perform boundary-scan test on pre-configured Cyclone III devices. For post-configured devices, you must modify the BSDL file according to the design. Altera provides the BSDLCustomizer tool for you to regenerate a BSDL file for your post-configured device. The BSDLCustomizer is a Tool Command Language (Tcl) script that uses the **.pin** file of the design generated by the Quartus II software and the BSDL file of the pre-configured device available from the Altera website to generate another BSDL file with modified port definitions and boundary-scan cell groups' attributes.

Other Considerations

Reliability

Take note of some basic rules when dealing with semiconductor devices to maintain the reliability of the devices.

CAUTION When handling an In-System Sources and Probes semiconductor device, beware of the electrostatic discharge (ESD) that can cause immediate functional failure, degrade I/O performance or decrease long-term reliability of the device.

WARNING Exercise care when handling the device by storing the device in an ESD-free environment and wearing grounds straps.

To ensure reliable device functionality, always operate the device within the recommended operating conditions, as specified in the *Cyclone III Device Datasheet: DC and Switching Characteristics* chapter in the *Cyclone III Device Handbook*. Device operation at the absolute maximum ratings for extended period of time may damage the device and result in the device not operating properly.

FPGA Starter Kit

To help you simplify the design process and reduce time-to-market, you can use the Cyclone III FPGA starter kit as a test and debug platform. The starter kit is easy to use and includes software, reference designs, cables, and programming hardware.

For more information about the Cyclone III starter kit, refer to the *Cyclone III FPGA Starter Kit*.

Conclusion

This design guideline covers various aspects of using the Cyclone III device that an FPGA designer should know when designing with Cyclone III FPGA. This guideline also makes the design flow easier. Using the right tools and good design practice allows you to design with Cyclone III devices with ease and save time.

Design Checklist

This checklist provides a summary of the guidelines described in this document. Use the checklist to verify that you have followed the guidelines for each stage of your design.

Project Name:
Date:

"Device Selection" on page 1

	Done	N/A	
1.	☐	☐	Select a device based on the logic/memory/multiplier density, device features such as PLLs, I/O pin count, package offering, and additional resources for debugging future development.
2.	☐	☐	Consider vertical device migration requirements. Consider availability of speed grades. Ensure sufficient timing margin if you plan to use a lower speed grade device in the future.

"Early System Planning" on page 3

	Done	N/A	
3.	☐	☐	Estimate power consumption and heat dissipation with the Early Power Estimator spreadsheet to plan the cooling solution and power supplies.
4.	☐	☐	Reduce I/O power consumption with lower I/O capacitance, lower voltage I/O standards, and resistively terminated I/O standards for high frequency signals.
5.	☐	☐	Select suitable I/O standard (single-ended, voltage-referenced, or differential) for your design.
6.	☐	☐	Place I/O pins that share the same V_{CCIO} and V_{REF} in the same I/O bank.
7.	☐	☐	Select the configuration scheme based on the Cyclone III device package, the resources available for the configuration, and the configuration time required by your system.
8.	☐	☐	For AS, AP, and PS configuration schemes, you can select either fast or standard POR time.
9.	☐	☐	For AS configuration scheme, make sure the selected EPCS device supports the configuration bitstream file size for the selected Cyclone III device.
10.	☐	☐	For AS configuration scheme, use SFL to reduce the effort to have separate programming interface for your Cyclone III and EPCS device.
11.	☐	☐	When using MAX II PFL for PS or FPP configuration scheme, or when using FPGA-based PFL for AP configuration scheme, ensure that you select the appropriate flash device according to the list of supported flash devices.
12.	☐	☐	Use the compression feature to reduce the configuration file size for AS and PS configuration schemes.
13.	☐	☐	Use the Cyclone III PLL for frequency synthesis and clock management.
14.	☐	☐	Ensure that the PLL input, output, and VCO frequency are within the specification in the datasheet.
15.	☐	☐	Cascade the PLLs to obtain the frequency you need, if the m, n, c counter or VCO frequency does not allow you to obtain the desired frequency when one PLL is used.
16.	☐	☐	Use the clock switchover feature of the PLL, if you need a backup input clock or to change the input clock source in user mode.
17.	☐	☐	Select the PLL compensation mode that best fits your design requirement.

18.	☐	☐	Use the PLL reconfiguration feature, if you need to change the PLL settings on the fly in user mode without reconfiguring the entire Cyclone III device.
19.	☐	☐	Use the `areset`, `locked` or `pfdena` control signals as required by your design.
20.	☐	☐	Depending on the on-chip debugging method used, reserve additional resources such as LE, memory and I/O, and route the JTAG pins out.

"Board Design Considerations" on page 16

	Done	**N/A**	
21.	☐	☐	Use the correct hardware setup when interfacing the Cyclone III device with 3.3/3.0/2.5-V **LVTTL**/**LVCMOS** I/O systems.
22.	☐	☐	Use the Quartus II software to check the restriction on the pin placement.
23.	☐	☐	Minimize the simultaneous switching noise effect on the board with the following methods: ■ Distributing simultaneous switching I/O to different I/O banks ■ Setting unused I/O pins to V_{CC} or ground ■ Turning on the slow slew rate feature for the switching pins ■ Using a lower drive strength for the switching pins ■ Using the proper termination scheme for the switching pins
24.	☐	☐	Connect the device unused pins to the board according to the behavior of the pins.
25.	☐	☐	Terminate the signals according to the I/O standard and signal travel direction.
26.	☐	☐	Perform board-level IBIS or HSPICE simulation.
27.	☐	☐	Minimize the noise on the power planes.
28.	☐	☐	Consider the power-up requirement and I/O behavior of the Cyclone III device during power up.
29.	☐	☐	Add the correct series resistance to the appropriate configuration signals based on the configuration scheme used.
30.	☐	☐	Use repeater buffers for the configuration signals when cascading Cyclone III devices in multi-device configuration.
31.	☐	☐	Ensure that the `DCLK` or `TCK` signals are clean.
32.	☐	☐	Pull the configuration pins high or low through resistors with the recommended values.
33.	☐	☐	Ensure that the devices in a configuration chain are connected properly.
34.	☐	☐	To prevent the signals to the configuration pins from overshooting to more than 4.1 V, use series resistors and buffers according to the setup.
35.	☐	☐	Do not leave the `JTAG` pins unconnected.
36.	☐	☐	Connect the `MSEL` pins to V_{CCA} or ground directly based on the configuration scheme used.
37.	☐	☐	For AP configuration, select the flash memory from the Intel P30 and P33 families that can support 40-MHz clock.
38.	☐	☐	Check the pin connection for other pins of the configuration device or flash memory.

"Board Design Considerations" on page 16

	Done	**N/A**	
39.	☐	☐	Use the correct hardware setup when interfacing the Cyclone III device with 3.3/3.0/2.5-V **LVTTL**/**LVCMOS** I/O systems.
40.	☐	☐	Use the Quartus II software to check the restriction on the pin placement.

41.	☐	☐	Minimize the simultaneous switching noise effect on the board with the following methods: ■ Distributing simultaneous switching I/O to different I/O banks ■ Setting unused I/O pins to V_{CC} or ground ■ Turning on the slow slew rate feature for the switching pins ■ Using a lower drive strength for the switching pins ■ Using the proper termination scheme for the switching pins
42.	☐	☐	Connect the device unused pins to the board according to the behavior of the pins.
43.	☐	☐	Terminate the signals according to the I/O standard and signal travel direction.
44.	☐	☐	Perform board-level IBIS or HSPICE simulation.
45.	☐	☐	Minimize the noise on the power planes.
46.	☐	☐	Consider the power-up requirement and I/O behavior of the Cyclone III device during power up.
47.	☐	☐	Add the correct series resistance to the appropriate configuration signals based on the configuration scheme used.
48.	☐	☐	Use repeater buffers for the configuration signals when cascading Cyclone III devices in multi-device configuration.
49.	☐	☐	Ensure that the DCLK or TCK signals are clean.
50.	☐	☐	Pull the configuration pins high or low through resistors with the recommended values.
51.	☐	☐	Ensure that the devices in a configuration chain are connected properly.
52.	☐	☐	To prevent the signals to the configuration pins from overshooting to more than 4.1 V, use series resistors and buffers according to the setup.
53.	☐	☐	Do not leave the JTAG pins unconnected.
54.	☐	☐	Connect the MSEL pins to V_{CCA} or ground directly based on the configuration scheme used.
55.	☐	☐	For AP configuration, select the flash memory from the Intel P30 and P33 families that can support 40-MHz clock.
56.	☐	☐	Check the pin connection for other pins of the configuration device or flash memory.

"Design and Compilation" on page 28

	Done	N/A	
57.	☐	☐	Consider the trade-off between schematics or HDL for your design entry based on your design's complexity.
58.	☐	☐	Other than the Quartus II software, consider building your design with third-party EDA tools, Qsys, or IP cores.
59.	☐	☐	If you are using HDL for design entry, use the recommended coding styles.
60.	☐	☐	If you are using third-party synthesis tools to synthesize your design, use the netlist generated by the synthesis tools for the Quartus II Fitter to do the place-and-route procedure.
61.	☐	☐	Use synchronous design. Use a single clock source to clock the registers in your design.
62.	☐	☐	Register your combinational logic outputs and inputs to filter out glitches.
63.	☐	☐	Pay attention to the clocks in your design. Use clock pins, the global clock network, clock control blocks, and the PLL for your clock signals.
64.	☐	☐	Chip-wide reset can be used to clear the registers.
65.	☐	☐	If you need the registers to be powered up high, set the register power-up level.

66.	☐	☐	Use the Quartus II Design Assistant to check design reliability.
67.	☐	☐	Use incremental compilation to preserve results and performance of unchanged logic in your design, achieve timing closure more efficiently and reduce compilation time when you make changes to your design, especially for bottom-up or top-down design flow.
68.	☐	☐	Take advantage of the various Quartus II software settings to optimize power consumption, including the Design Space Explorer, power-driven compilation, gate-level register retiming, minimum area-synthesis, and Power Optimization Advisor.
69.	☐	☐	Split large designs into smaller partitions and register the partition I/O boundary.
70.	☐	☐	Separate the source code of each block into different files.
71.	☐	☐	Allocate the resources among the design partitions to avoid conflict.
72.	☐	☐	Use the Quartus II Chip Planner to create a design floorplan using LogicLock region assignments.
73.	☐	☐	Pay attention to the design techniques to optimize power consumption, including clock power management, memory power reduction, pipelining and retiming, and architectural optimization.
74.	☐	☐	Use the Quartus II Pin Planner to create and edit pin related assignments.
75.	☐	☐	Use the Pin Planner Package View to visually identify the locations of the pins, I/O banks, V_{REF} groups, and the differential pin pairing.
76.	☐	☐	Use the Pin Migration View in the Pin Planner to identify the pins that change functions between migration devices.
77.	☐	☐	Use the correct current strength and slew rate on output or bidirectional pins to prevent signal overshoot or undershoot for signal noise reduction, or to prevent stair-step output from the pins.
78.	☐	☐	Use the Cyclone III series on-chip termination feature for I/O impedance matching and saving board space.
79.	☐	☐	Use the programmable pre-emphasis for high-frequency signals to reduce attenuation to high-frequency signals during transmission.
80.	☐	☐	Use the dedicated differential output buffers to implement the differential I/O standards without requiring additional resistors.
81.	☐	☐	Determine the location of the PLL you want to use based on the location of the clock pins your design uses, or let the Quartus II software determine the PLL location based on your pin assignment for the clock pins.
82.	☐	☐	Assign the ports of the altpll megafunction and the altpll_reconfig megafunction to the recommended pins.
83.	☐	☐	Perform timing or functional simulation on your PLL design to check the functionality.
84.	☐	☐	Use the `CLKUSR` pin to control the initialization upon the completion of the configuration.
85.	☐	☐	Use the `INIT_DONE` pin to show the completion of the initialization process after configuration.
86.	☐	☐	Use the auto-restart configuration after error option to force the Cyclone III device reconfiguration if there is any error during configuration.
87.	☐	☐	Use compressed configuration file for configuring the Cyclone III device to reduce storage requirement and configuration time.
88.	☐	☐	Use the RBF file to estimate the amount of storage space required for configuration data.

89.	☐	☐	Perform I/O assignment analysis with the Quartus II software. The Quartus II software also checks the location of I/O pads for any restrictions.
90.	☐	☐	Do not sink or source current through the I/O pins above the recommended specification. You can also set the limit in the Quartus II software.

"Verification" on page 45

	Done	N/A	
91.	☐	☐	Use the TimeQuest Timing Analyzer to perform timing analysis. Specify the frequencies and relationships for all clocks in your design. Use input and output delay constraints to specify external device or board timing parameters.
92.	☐	☐	Perform the Early Timing Estimation if you need a timing estimation before running a full compilation.
93.	☐	☐	Check the timing report and make sure there is no t_{SU}, t_H, t_{CO}, t_{PD}, or f_{MAX} violation.
94.	☐	☐	Use the Quartus II assignment to specify the input and output delay.
95.	☐	☐	Use the Quartus II assignment to specify the data and clock skew.
96.	☐	☐	Use the **physical synthesis** option or the Design Space Explorer utility to optimize the fitting for performance or area.
97.	☐	☐	Perform recovery and removal timing analysis for asynchronous control signals
98.	☐	☐	Use the `set_multicycle_path` command for multicycle path timing analysis.
99.	☐	☐	Use the Optimization Advisors as guidance to make settings to optimize your design on resources or timing.
100.	☐	☐	Turn on the **Optimize fast-corner timing** option to increase the timing robustness of your design.
101.	☐	☐	Check the **Analysis & Synthesis** section on the compilation report to ensure that the registers are not removed unintentionally.
102.	☐	☐	Turn on **Optimize hold timing** option to **All paths** to optimize the register-to-register hold time paths.
103.	☐	☐	Turn on **Enable multicorner timing analysis** option for the TimeQuest Timing Analyzer to analyze the design and generate slack reports for the slow and fast corners.
104.	☐	☐	Use the `derive_pll_clocks` constraint to create generated clocks for PLLs.
105.	☐	☐	Use the `derive_clock_uncertainty` constraint to apply inter-clock, intra-clock, and I/O interface uncertainties.
106.	☐	☐	Use the `check_timing` constraint to generate a report on any problem with the design or applied constraints.
107.	☐	☐	Review the Quartus II messages for any potential problem to your design. Use the Quartus II Help to understand the messages.
108.	☐	☐	Use parallel processing or incremental compilation in the Quartus II software to reduce compilation time.
109.	☐	☐	Use the PowerPlay Power Analyzer to perform power estimation on your design.
110.	☐	☐	Perform timing or functional simulation with the ModelSim-Altera or other third-party simulation tools to verify that your design is working correctly.

"Design Debugging" on page 56

	Done	N/A	
111.	☐	☐	Use incremental compilation to reduce compilation time during debugging.

"Testing" on page 56

	Done	N/A	
112.	☐	☐	Perform boundary-scan test to check the pin connections on your board.
113.	☐	☐	Use the BSDLCustomizer tool to regenerate the BSDL file based on your design.

"Other Considerations" on page 57

	Done	N/A	
114.	☐	☐	Be cautious of ESD when handling the device.
115.	☐	☐	Operate the device within the recommended operating conditions.
116.	☐	☐	Use the Cyclone III FPGA Starter Kit to test out various features of the Cyclone III device.

Document Revision History

Table 8 lists the revision history for this document.

Table 8. Document Revision History

Date	Version	Changes
August 2013	2.2	Updated the note in the "EPCS Connection" section.
July 2012	2.1	Updated "Flexible I/O Banks" with information about higher pin capacitance when `VREF` pins are used as I/O.
July 2012	2.0	■ Removed device resources and vertical migration information that are available in the *Cyclone III Device Handbook* and added links to the relevant chapters in the handbook. ■ Replaced the "SOPC Builder" section with "Qsys System Integration Tool", and updated references to SOPC Builder with Qsys. ■ Updated references to ModelSim-Altera Web Edition and to ModelSim-Altera Starter Edition. ■ Updated references to the Quartus II Simulator to ModelSim-Altera. ■ Removed references to the Quartus II Classic Timing Analyzer. ■ Updated reference links to new and updated documents. ■ Removed links to obsolete documents. ■ Updated Figure 7, Figure 8, Figure 9, Figure 10, Figure 11, Figure 12, and Figure 13. ■ Updated "Area and Timing Optimization" ■ Removed the "Referenced Documents" section.
November 2008	1.2	Updated "Power Consideration" section.
July 2008	1.1	■ Added "Planning for On-Chip Debugging", "Selecting a Synthesis Tool", "Using Megafunctions", and "Planning for Hierarchical and Team-Based Design" sections. ■ Updated "Design Entry" and "Verification" sections. ■ Added Table 1, Table 7, and Table 4. ■ Inserted Figure 8, Figure 9, Figure 10, Figure 11, Figure 12 and Figure 13.
August 2007	1.0	Initial release.