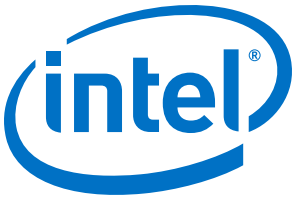


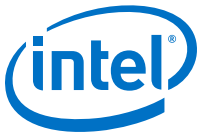
Intel® Cyclone® 10 LP Device Design Guidelines

AN-800
2017.05.08


Subscribe

Send Feedback

Contents



Intel® Cyclone® 10 LP Device Design Guidelines

It is important to follow Intel design guidelines and recommendations throughout the design process for Cyclone 10 LP designs. The design guidelines also assist you with planning the FPGA and system early in the design process, which is crucial to successfully meet design requirements.

Note: For more information about Cyclone 10 LP devices and features, refer to the *Cyclone 10 LP Core Fabric and General Purpose I/Os Handbook*.

The material references the Cyclone 10 LP device architecture as well as aspects of the Quartus® Prime software and third-party tools that you might use in your design. These guidelines can improve productivity and avoid common design pitfalls.

Related Links

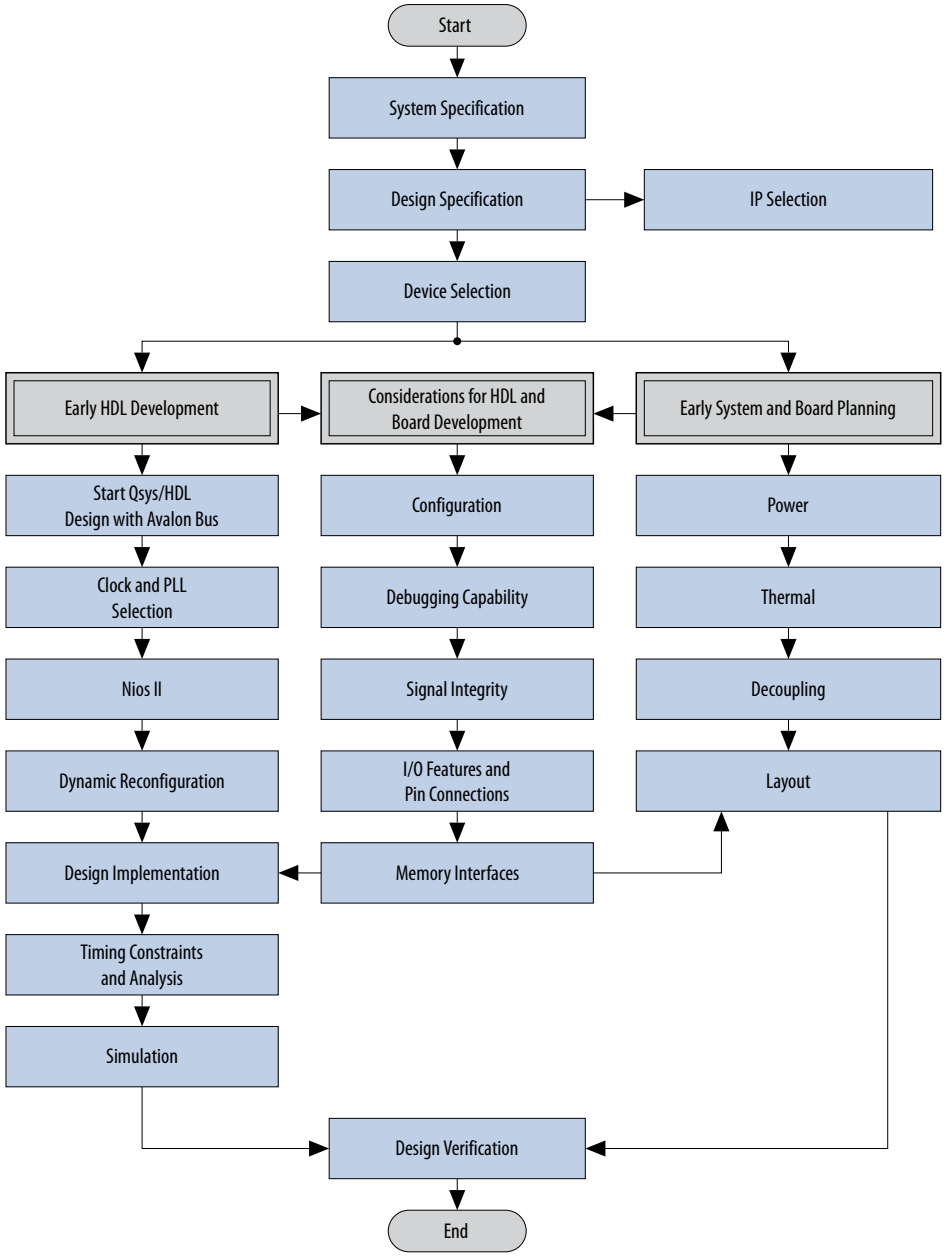
Cyclone 10 LP Device Handbook
Provides more information about the Cyclone 10 LP devices.

Design Flow

Table 1. Summary of the Design Flow Stage and Guideline Topics

Stages of the Design Flow	Guidelines
System Specification	Planning design specifications, IP selection
Device Selection	Device information, determining device variant and density, package offerings, migration, speed grade
Early System and Board Planning	Early power estimation, thermal management option, planning for configuration scheme, planning for on-chip debugging
Pin Connection Considerations for Board Design	Power-up, power pins, PLL connections, decoupling capacitors, configuration pins, signal integrity, board-level verification
I/O and Clock Planning	Pin assignments, early pin planning, I/O features and connections, memory interfaces, clock and PLL selection, simultaneous switching noise (SSN)
Design Entry	Coding styles and design recommendations, Qsys, planning for hierarchical or team-based design
Design Implementation, Analysis, Optimization, and Verification	Synthesis tool, device utilization, messages, timing constraints and analysis, area and timing optimization, compilation time, verification, power analysis and optimization

Figure 1. Cyclone 10 LP Device Design Flow



System Specification

In systems that contain a Cyclone 10 LP device, the FPGA typically plays a large role in the overall system and affects the rest of the system design. It is important to start the design process by creating detailed design specifications for the system and the FPGA, and determining the FPGA input and output interfaces to the rest of the system.

Design Specifications

Table 2. Design Specifications Checklist

Number	Done?	Checklist Item
1		Create detailed design specifications and a test plan if appropriate.
2		Plan clock domains, clock resources, and I/O interfaces early with a block diagram.

Create detailed design specifications that define the system before you create your logic design or complete your system design, by performing the following:

- Specify the I/O interfaces for the FPGA
- Identify the different clock domains
- Include a block diagram of basic design functions
- Include intellectual property (IP) blocks

  Note: Taking the time to create these specifications improves design efficiency, but this stage is often skipped by FPGA designers.
- Create a functional verification/test plan
- Consider a common design directory structure

Create a functional verification plan to ensure the team knows how to verify the system. Creating a test plan at this stage can also help you design for testability and design for manufacturability. For example, do you want to perform built-in-self test (BIST) functions to drive interfaces? If so, you could use a UART interface with a Nios® processor inside the FPGA device. You might require the ability to validate all the design interfaces.

If your design includes multiple designers, it is useful to consider a common design directory structure. This eases the design integration stages.

Related Links

- Planning for On-Chip Debugging on page 14
  Provides the guidelines related to analyzing and debugging the device after the device is in the system.
- IP Selection on page 6

IP Selection

Table 3. IP Selection Checklist

Number	Done?	Checklist Item
1		Select IP that affects system design, especially I/O interfaces.
2		If you plan to use the OpenCore Plus tethered mode for IP, ensure that your board design supports this mode of operation.

Intel and its third-party IP partners offer a large selection of off-the-shelf IP cores optimized for Intel devices. You can easily implement these parameterized blocks of IP in your design, reducing your system implementation and verification time, and allowing you to concentrate on adding proprietary value.

IP selection often affects system design, especially if the FPGA interfaces with other devices in the system. Consider which I/O interfaces or other blocks in your system design can be implemented using IP cores, and plan to incorporate these cores in your FPGA design.

The OpenCore Plus feature available for many IP cores allows you to program the FPGA to verify your design in hardware before you purchase the IP license. The evaluation supports an untethered mode, in which the design runs for a limited time, or a tethered mode. The tethered mode requires an Intel serial JTAG cable connected between the JTAG port on your board and a host computer running the Quartus Prime Programmer for the duration of the hardware evaluation period.

Related Links

Intellectual Property
Provides more descriptions of available IP cores.

Qsys

Table 4. Qsys Checklist

Number	Done?	Checklist Item
1		Take advantage of Qsys for system and processor designs.

Qsys is a system integration tool included as part of the Quartus Prime software. Qsys captures system-level hardware designs at a high level of abstraction and automates the task of defining and integrating customized Hardware Description Language (HDL) components. These components include IP cores, verification IP, and other design modules. Qsys facilitates design reuse by packaging and integrating your custom components with Intel and third-party IP components. Qsys automatically creates interconnect logic from the high-level connectivity you specify, thereby eliminating the error-prone and time-consuming task of writing HDL to specify system-level connections.

Qsys is more powerful if you design your custom components using standard interfaces. By using standard interfaces, your components inter-operate with the components in the Qsys Library. In addition, you can take advantage of bus functional models (BFMs), monitors, and other verification IP to verify your design.

Related Links

Creating a System With Qsys chapter, Quartus Prime Standard Edition Handbook Volume 1: Design and Synthesis
Provides more information about Qsys.

Device Selection

Device selection is the first step in the Cyclone 10 LP design process—choosing the device family variant, device density, features, package, and speed grade that best suit your design requirements.

Related Links

Cyclone 10 LP Device Overview

Provides more information about the features available in each device density, including logic, multipliers, and phase-locked loops (PLLs), as well as the various package offerings and I/O pin counts.

Device Variant

Table 5. Device Variant Checklist

Number	Done?	Checklist Item
1		Consider the available device variants.
2		Select a device based on transceivers, I/O pin count, LVDS channels, package offering, logic/ memory/multiplier density, PLLs, clock routing, and speed grade.

The Cyclone 10 LP devices offer low power with the lowest cost. If your application requires high-speed transceivers, refer to the *Cyclone 10 GX Device Design Guidelines*.

PLLs and Clock Routing

Table 6. PLLs and Clock Routing Checklist

Number	Done?	Checklist Item
1		Verify the number of PLLs and clock routing resources.

Verify that your chosen device density package combination includes enough PLLs and clock routing resources for your design.

Logic, Memory, and Multiplier Density

Table 7. Logic, Memory, and Multiplier Density Checklist

Number	Done?	Checklist Item
1		Estimate the required logic, memory, and multiplier density.
2		Reserve device resources for future development and debugging.

Cyclone 10 LP devices offer a range of densities that provide different amounts of device logic resources, including LEs, memory, and multipliers. Determining the required logic density can be a challenging part of the design planning process. Devices with more logic resources can implement larger and potentially more complex designs, but generally have a higher cost. Smaller devices have lower static power utilization. Cyclone 10 LP devices support vertical migration, which provides flexibility.

Many next-generation designs use a current design as a starting point. If you have other designs that target an Intel device, you can use their resource utilization as an estimate for your new design. Review the resource utilization to find out which device density fits the design. Consider that the coding style, device architecture, and optimization options used in the Quartus Prime software can significantly affect a design's resource utilization and timing performance.

Select a device that meets your design requirements with some safety margin in case you want to add more logic later in the design cycle, upgrade, or expand your design. You might also want additional space in the device to ease design floorplan creation for an incremental or team-based design. Consider reserving resources for debugging.

Related Links

- Vertical Device Migration on page 10
- Device Resource Utilization Reports on page 33
- Documentation: IP and Megafunctions
  Provides more information about resource utilization estimates for certain IP design configurations.
- Planning for On-Chip Debugging on page 14
  Provides more information about reserving resources for debugging.

I/O Pin Count, LVDS Channels, and Package Offering

Table 8. I/O Pin Count, LVDS Channels, and Package Offering Checklist

Number	Done?	Checklist Item
1		Estimate the number of I/O pins that you require.
2		Consider the I/O pins you need to reserve for debugging.
3		Verify that the number of LVDS channels are enough.

Determine the required number of I/O pins for your application, considering the design's interface requirements with other system blocks.

Larger densities and package pin counts offer more full-duplex LVDS channels for different signaling; ensure that your device density-package combination includes enough LVDS channels. Other factors can also affect the number of I/O pins required for a design, including simultaneous switching noise (SSN) concerns, pin placement guidelines, pins used as dedicated inputs, I/O standard availability for each I/O bank, differences between I/O standards and speed for row and column I/O banks, and package migration options.

You can compile any existing designs in the Quartus Prime software to determine how many I/O pins are used. Also consider reserving I/O pins for debugging.

Speed Grade

Table 9. Speed Grade Checklist

Number	Done?	Checklist Item
1		Determine the device speed grade that you require.

The device speed grade affects the device timing performance and timing closure, as well as power utilization. One way to determine which speed grade your design requires is to consider the supported clock rates for specific I/O interfaces.

You can use the fastest speed grade during prototyping to reduce compilation time (because less time is spent optimizing the design to meet timing requirements), and then move to a slower speed grade for production to reduce cost if the design meets its timing requirements.

Vertical Device Migration

Table 10. Vertical Device Migration Checklist

Number	Done?	Checklist Item
1		Consider vertical device migration availability and requirements.

Cyclone 10 LP devices support vertical migration within the same package, which enables you to migrate to different density devices whose dedicated input pins, configuration pins, and power pins are the same for a given package. This feature allows future upgrades or changes to your design without any changes to the board layout, because you can replace the FPGA on the board with a different density Cyclone 10 LP device.

Determine whether you want the option of migrating your design to another device density. Choose your device density and package to accommodate any possible future device migration to allow flexibility when the design nears completion. You should specify any potential migration options in the Quartus Prime software at the beginning of your design cycle or as soon as the device migration selection is possible in the Quartus Prime software. Selecting a migration device can impact the design's pin placement, because the Fitter ensures your design is compatible with the selected device(s). It is possible to add migration devices later in the design cycle, but it requires extra effort to check pin assignments, and can require design or board layout changes to fit into the new target device. It is easier to consider these issues early in the design cycle than at the end, when the design is near completion and ready for migration.

The Quartus Prime Pin Planner highlights pins that change function in the migration device when compared to the currently selected device.

Related Links

Making FPGA Pin Assignments on page 24

Early System and Board Planning

System information related to the FPGA should be planned early in the design process, before designers have completed the design in the Quartus Prime software. Early planning allows the FPGA team to provide early information to PCB board and system designers.

Early Power Estimation

Table 11. Early Power Estimation Checklist

Number	Done?	Checklist Item
1		Estimate power consumption with the Early Power Estimator (EPE) spreadsheet to plan the cooling solution and power supplies before the logic design is complete.

FPGA power consumption is an important design consideration and must be estimated accurately to develop an appropriate power budget to design the power supplies, voltage regulators, decouplers, heat sink, and cooling system. Power estimation and analysis have two significant planning requirements:

- Thermal planning—The cooling solution must sufficiently dissipate the heat generated by the device. In particular, the computed junction temperature must fall within normal device specifications.
- Power supply planning—The power supplies must provide adequate current to support device operation.

Power consumption in FPGA devices is dependent on the logic design. This dependence can make power estimation challenging during the early board specification and layout stages. The Intel EPE tool allows you to estimate power utilization before the design is complete by processing information about the device and the device resources that will be used in the design, as well as the operating frequency, toggle rates, and environmental considerations. You can use the tool to perform thermal analysis, including calculation of device junction temperatures derived from the ambient temperature and device power consumption. The EPE then calculates the power, current estimates, and thermal analysis for the design.

If you do not have an existing design, estimate the number of device resources used in your design and enter it manually. The EPE tool accuracy depends on your inputs and your estimation of the device resources. If this information changes (during or after your design is complete), your power estimation results are less accurate. If you have an existing design or a partially-completed compiled design, use the `Generate Early Power Estimator File` command in the Quartus Prime software to provide input to the spreadsheet.

The EPE spreadsheet includes the Import Data macro, which parses the information in the Quartus Prime-generated power estimation file (`.csv`), or alternatively from an older version of the EPE, and transfers it into the EPE tool. If you do not want to use the macro, you can transfer the data into the EPE tool manually. You should enter additional resources to be used in the final design manually if the existing Quartus Prime project represents only a portion of your full design. You can edit the inputs to the EPE tool and add additional device resources or adjust the parameters after importing the power estimation file information.

When the design is complete, the Power Analyzer tool in the Quartus Prime software provides more accurate estimation of power, ensuring that thermal and supply budgets are not violated. For the most accurate power estimation, use gate-level simulation results with an output file (`.vcd`) from a third-party simulation tool.

Related Links

- Power Analysis on page 38
- Early Power Estimators (EPE) and Power Analyzer Page
  Provides more information about using the EPE spreadsheet.

Planning for Device Configuration

Table 12. Planning for Device Configuration Checklist

Number	Done?	Checklist Item
1		Consider whether you require multiple configuration schemes.

Cyclone 10 LP devices are based on SRAM cells. You must download configuration data to the Cyclone 10 LP device each time the device powers up, because SRAM is volatile. Consider whether you require multiple configuration schemes, such as one for debugging or testing and another for the production environment.

Choosing the device configuration method early allows system and board designers to determine what companion devices, if any, are required for the system. Your board layout also depends on the configuration method you plan to use for the programmable device, because different schemes require different connections.

In addition, Cyclone 10 LP devices offer advanced configuration features, depending on your configuration scheme. Cyclone 10 LP devices also include optional configuration pins and a reconfiguration option that you should choose early in the design process (and set up in the Quartus Prime software), so you have all the information required for your board and system design.

Related Links

- Pin Connection Considerations for Board Design on page 16
- Configuration, Design Security, Remote System Upgrades in Cyclone LP 10 Devices
  Provides more information Cyclone 10 LP devices.

Configuration Scheme Selection

Table 13. Configuration Scheme Selection Checklist

Number	Done?	Checklist Item
1		Select a configuration scheme to plan companion devices and board connections.

Cyclone 10 LP devices offer several configuration schemes.

You can enable any specific configuration scheme by driving the Cyclone 10 LP device MSEL pins to specific values on the board.

All configuration schemes use a configuration device, a download cable, or an external controller (for example, MAX® (MAX II, MAX V, MAX 10) devices or microprocessor).

Serial Configuration Devices

Table 14. Serial Configuration Devices Checklist

Number	Done?	Checklist Item
1		If you want to use the AS configuration mode with large device densities, confirm there is a configuration device available that is large enough for your target FPGA density.

Serial configuration devices can be programmed using an Intel FPGA Download Cable or Intel FPGA Ethernet Cable with the Quartus Prime software through the active serial interface.

Alternatively, you can use supported third-party programmers such as BP Microsystems and System General, or a microprocessor with the SRunner software driver. SRunner is a software driver developed for embedded serial configuration device programming that designers can customize to fit in different embedded systems.

Serial configuration devices do not directly support the JTAG interface; however, you can program the device with JTAG download cables using the Cyclone 10 LP FPGA as a bridge between the JTAG interface and the configuration device, allowing both devices to use the same JTAG interface.

Download Cables

Table 15. Download Cables Checklist

Number	Done?	Checklist Item
1		Use download cables for device configuration.

The Quartus Prime programmer supports configuration of the Cyclone 10 LP devices directly using JTAG interfaces with Intel programming download cables. You can download design changes directly to the device with Intel 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 Signal Tap Embedded Logic Analyzer.

Configuration Features

Table 16. Configuration Features Checklist

Number	Done?	Checklist Item
1		Ensure your configuration scheme and board support the required features: design security, remote upgrades, single event upset (SEU) mitigation.

Cyclone 10 LP devices have a built-in dedicated circuitry for error detection and correction. When enabled, this feature checks for SEUs continuously and automatically. This allows you to confirm that the configuration data stored in a Cyclone 10 LP device is correct and alerts the system to a configuration error.

To take advantage of the SEU mitigation features, turn on **Enable error detection CRC** in the Device and Pin Options dialog box of the Quartus Prime software. Use the CRC_ERROR pin to flag errors and design your system to take the appropriate action. If not enabled for the CRC function, the CRC_ERROR pin is available as a user I/O pin.

Note: The SEU mitigation feature is available in all Cyclone 10 LP devices with 1.2-V core voltage.

Quartus Prime Configuration Settings

Table 17. Quartus Prime Configuration Settings Checklist

Number	Done?	Checklist Item
1		Consider the Quartus Prime configuration options when you plan your board and system design.

There are several configuration options that you can set in the Quartus Prime software before compilation to generate configuration or programming files. Your board and system design are affected by these settings and pins, so consider them in the planning stages. Set the options on the General category of the **Device and Pin Options** dialog box.

Optional Configuration Pins

Table 18. Optional Configuration Pins Checklist

Number	Done?	Checklist Item
1		Plan the board design to support optional configuration pins as required.

You can enable the following optional configuration pins:

- CLKUSR
- INIT_DONE

Planning for On-Chip Debugging

On-chip debugging is an optional step in the design flow, and different debugging tools work better for different systems and different designers. Evaluate on-chip debugging options early in your design process to ensure that your system board, Quartus Prime project, and design are able to support the appropriate options. Planning can reduce time spent debugging, and eliminates design changes later to accommodate your preferred debugging methodologies. Adding debug pins might not be enough, because of internal signal accessibility and I/O pin accessibility on the device. First, select your preferred debugging tool(s).

On-Chip Debugging Tools

Table 19. On-Chip Debugging Tools Checklist

Number	Done?	Checklist Item
1		Take advantage of on-chip debugging features to analyze internal signals and perform advanced debugging techniques.

The Quartus Prime portfolio of verification tools includes the following in-system debugging features:

- Signal Probe incremental routing—Quickly routes 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 signals for debugging to either previously reserved or currently unused I/O pins.
- Signal Tap Embedded Logic Analyzer—Probes the state of internal and I/O signals without the use of external equipment or extra I/O pins, while the design is running at full speed in an FPGA device. Defining custom trigger-condition logic provides greater accuracy and improves the ability to isolate problems. It does not require external probes or changes to the design files to capture the state of the internal nodes or I/O pins in the design; all captured signal data is stored in the device memory until you are ready to read and analyze the data. The Signal Tap Embedded Logic Analyzer works best for synchronous interfaces. For debugging asynchronous interfaces, consider using Signal Probe or an external logic analyzer to view the signals more accurately.
- Logic Analyzer Interface—Enables you to connect and transmit internal FPGA signals to an external logic analyzer for analysis, allowing you to take advantage of advanced features in your external logic analyzer or mixed signal oscilloscope. You can use this feature to connect a large set of internal device signals to a small number of output pins for debugging purposes and it can multiplex signals with design I/O pins if required.
- In-System Memory Content Editor—Provides read and write access to in-system FPGA memories and constants through the JTAG interface, so you can test changes to memory content and constant values in the FPGA while the device is functioning in the system.
- In-System Sources and Probes—Sets up customized register chains to drive or sample the instrumented nodes in your logic design, providing an easy way to input simple virtual stimuli and capture the current value of instrumented nodes.
- Virtual JTAG IP core—Enables you to build your own system-level debugging infrastructure, including both processor-based debugging solutions and debugging tools in the software for system-level debugging. You can instantiate the SLD_VIRTUAL_JTAG IP core directly in your HDL code to provide one or more transparent communication channels to access parts of your FPGA design using the JTAG interface of the device.

Planning Guidelines for Debugging Tools

Table 20. Planning Guidelines for Debugging Tools Checklist

Number	Done?	Checklist Item
1		Select on-chip debugging scheme(s) early to plan memory and logic requirements, I/O pin connections, and board connections.
2		If you want to use Signal Probe incremental routing, the Signal Tap Embedded Logic Analyzer, Logic Analyzer Interface, In-System Memory Content Editor, In-System Sources and Probes, or Virtual JTAG IP core, plan your system and board with JTAG connections that are available for debugging.
3		Plan for the small amount of additional logic resources used to implement the JTAG hub logic for JTAG debugging features.
4		For debugging with the Signal Tap Embedded Logic Analyzer, reserve device memory resources to capture data during system operation.

continued...

Number	Done?	Checklist Item
5		Reserve I/O pins for debugging with Signal Probe or the Logic Analyzer Interface so you do not have to change the design or board to accommodate debugging signals later.
6		Ensure the board supports a debugging mode where debugging signals do not affect system operation.
7		Incorporate a pin header or mictor connector as required for an external logic analyzer or mixed signal oscilloscope.
8		To use debug tools incrementally and reduce compilation time, ensure incremental compilation is on so you do not have to recompile the design to modify the debug tool.
9		To use the Virtual JTAG IP core for custom debugging applications, instantiate it in the HDL code as part of the design process.
10		To use the In-System Sources and Probes feature, instantiate the IP core in the HDL code.
11		To use the In-System Memory Content Editor for RAM or ROM blocks or the LPM_CONSTANT IP core, turn on the **Allow In-System Memory Content Editor** to capture and update content independently of the system clock option for the memory block in the IP catalog.

If you intend to use any of the on-chip debugging tools, plan for the tool(s) when developing the system board, Quartus Prime project, and design.

Pin Connection Considerations for Board Design

When designing the interfaces to the Cyclone 10 LP device, various factors can affect the PCB design.

Device Power-Up

Table 21. Device Power-Up Checklist

Number	Done?	Checklist Item
1		Design board for power-up: All Cyclone 10 LP GPIO pins are tri-stated until the device is configured and the configuration pins drive out.
2		Design voltage power supply ramps to be monotonic.
3		Set POR time to ensure power supplies are stable.
4		Design power sequencing and voltage regulators for best device reliability. Connect the GND between boards before connecting the power supplies.

Cyclone 10 LP device I/O pins are hot-socketing compliant without external components. You can insert or remove a Cyclone 10 LP device from a powered-up system board without damaging or interfering with normal system and board operation.

You can drive signals into the I/O pins before or during power up or power down without damaging the device. Cyclone 10 LP devices support power up or power down of the VCCINT, VCCA, and VCCIO pins in any sequence to simplify the system level design. The individual power supply ramp-up and ramp-down rates can range from 50 µs to 50 ms. The power ramp must be monotonic.

In a hot-socketing situation, the Cyclone 10 LP device's output buffers are turned off during system power up or power down. Also, the Cyclone 10 LP device does not drive out until the device is configured and working within the recommended operating conditions.

Hot-socketing circuitry is not available on CONF_DONE, nCEO, and nSTATUS configuration pins because they are required during configuration. Therefore, it is expected behavior for these pins to drive out during power-up and power-down sequences.

The POR circuit keeps the entire system in reset until the power supply voltage levels have stabilized after power up. After power up, the device does not release nSTATUS until V_{CCINT}, V_{CCA}, and V_{CCIO} for the I/O banks that contain configuration pins are above the POR trip point of the device. After power down, brown-out occurs if the VCCINT or V_{CCA} voltage sags below the POR trip point. In Cyclone 10 LP devices, you can select a fast or standard POR time, depending on the MSEL pin settings. The fast POR time is 3 ms < T_{POR} < 9 ms for a fast configuration time. The standard POR time is 50 ms < T_{POR} < 200 ms, which has a lower power-ramp rate.

When power is applied to a Cyclone 10 LP device, a POR event occurs if the power supply reaches the recommended operating range within a certain period of time (specified as a maximum power supply ramp time; t_{RAMP}). The maximum power supply ramp time for Cyclone 10 LP devices is 50 ms for standard POR or 3 ms for fast POR, while the minimum power supply ramp time is 50 µs. Although power sequencing is not a requirement for correct operation, you should consider the power-up timing for each rail to prevent problems with long-term device reliability when designing a multi-rail powered system. You can reduce the device in-rush current with proper sequencing and voltage regulator design.

Note: Intel recommends that you connect the GNDs between boards before connecting the power supplies to ensure device reliability and compliance to the hot-socketing specification.

Power Pin Connections and Power Supplies

Table 22. Power Pin Connections and Power Supplies Checklist

Number	Done?	Checklist Item
1		Connect all power pins correctly as specified in the *Cyclone 10 LP Device Family Pin Connection Guidelines*.
2		Connect VCCIO pins and VREF pins to support each bank's I/O standards.
3		Explore unique requirements for FPGA power pins or other power pins on your board, and determine which devices on your board can share a power rail.
4		Follow the suggested power supply sharing and isolation guidance, and the specific guidelines for each pin in the *Cyclone 10 LP Device Family Pin Connection Guidelines*.

Cyclone 10 LP devices support a wide range of industry I/O standards. The device output pins do not meet the I/O standard specifications if the V_{CCIO} level is out of the recommended operating range for the I/O standard.

Voltage reference (VREF) pins serve as voltage references for certain I/O standards. The VREF pin 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.

Decoupling Capacitors

Table 23. Decoupling Capacitors Checklist

Number	Done?	Checklist Item
1		Use the PDN tool to plan your power distribution netlist and decoupling capacitors.

Board decoupling is important for improving overall power supply integrity while ensuring the rated device performance.

Cyclone 10 LP devices include on-die decoupling capacitors to provide high-frequency decoupling. These low-inductance capacitors suppress power noise for excellent power integrity performance, and reduce the number of external PCB decoupling capacitors, saving board space, reducing cost, and greatly simplifying PCB design.

For each power supply, PDN designers must choose a network of bulk and decoupling capacitors. While SPICE simulation could be used to simulate the circuit, the PDN design tool provides a fast, accurate, and interactive way to determine the right number of decoupling capacitors for optimal cost and performance trade-offs.

PLL Board Design Guidelines

Table 24. PLL Board Design Guidelines Checklist

Number	Done?	Checklist Item
1		Connect all PLL power pins to reduce noise even if the design does not use all the PLLs.
2		Power supply nets should be provided by an isolated power plane, a power plane cut out, or thick trace of at least 20 mils.

Plan your board design when you design a power system for PLL usage and to minimize jitter, because PLLs contain analog components embedded in a digital device.

Configuration Pin Connections

Table 25. Configuration Pin Connections Checklist

Number	Done?	Checklist Item
1		Check that all configuration pin connections and pull-up/pull-down resistors are set correctly for your configuration scheme(s).

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

Clock Trace Signal Integrity

Table 26. Clock Trace Signal Integrity Checklist

Number	Done?	Checklist Item
1		Design configuration clock traces to be noise-free.

Board trace for clocks used in configuration, for example TCK and DCLK clock input, should produce clean signals with no overshoot, undershoot, or ringing. When designing the board, lay out the configuration clock traces with the same techniques used to lay out a clock line. Any overshoot, undershoot, ringing, or other noise on the clock signal can cause configuration failure.

JTAG Pins

Table 27. JTAG Pins Checklist

Number	Done?	Checklist Item
1		Connect JTAG pins to a stable voltage level if not in use.

Because JTAG configuration takes precedence over all other configuration methods, the JTAG pins should not be left floating or toggling during configuration if you do not use the JTAG interface. If you are using the JTAG interface, adhere to the following guidelines.

JTAG Pin Connections

Table 28. JTAG Pin Connections Checklist

Number	Done?	Checklist Item
1		Connect JTAG pins correctly to the download cable header. Ensure the pin order is not reversed.
2		To disable the JTAG state machine during power-up, pull the TCK pin low through a resistor to ensure that an unexpected rising edge does not occur on the TCK pin.
3		Pull the TMS and TDI pins high through a resistor.

A device operating in JTAG mode uses four required pins—TDI, TDO, TMS, and TCK. The TCK pin has an internal weak pull-down resistor, while the TDI and TMS pins have weak internal pull-up resistors.

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.

Noise on the JTAG pins during configuration, user mode, or power-up can cause the device to go into an undefined state or mode.

Download Cable Operating Voltage

Table 29. Download Cable Operating Voltage Checklist

Number	Done?	Checklist Item
1		Ensure the download cable and JTAG pin voltages are compatible because the download cable interfaces with the JTAG pins of your device.

The operating voltage supplied to the Intel download cable by the target board through the 10-pin header determines the operating voltage level of the download cable.

JTAG pins in the Cyclone 10 LP device are powered up by V_{CCIO}. In a JTAG chain containing devices with different voltage levels, devices with a higher voltage must drive devices with the same or lower voltage level. Level shifter might be required between devices to meet the voltage specifications of the devices input pin.

JTAG Signal Buffering

Table 30. JTAG Signal Buffering Checklist

Number	Done?	Checklist Item
1		Buffer JTAG signals per the recommendations, especially for connectors or if the cable drives more than three devices.
2		If your device is in a configuration chain, ensure all devices in the chain are connected properly.

You might have to add buffers to a JTAG chain, depending on the JTAG signal integrity, especially the TCK signal, because it is the JTAG clock and the fastest switching JTAG signal. Intel recommends buffering the signals at the connector because cables and board connectors tend to make bad 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 cross a board connector.

If a cable drives three or more devices, buffer the JTAG signal at the cable connector to prevent signal deterioration. This 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 that a buffer should be added to the chain.

Each buffer should drive no more than eight loads for the TCK and TMS signals, which drive in parallel. If jumpers or switches are added to the path, decrease the number of loads.

MSEL Configuration Mode Pins

Table 31. MSEL Configuration Mode Pins Checklist

Number	Done?	Checklist Item
1		Connect the MSEL pins to select the configuration scheme; do not leave them floating.

Select the configuration scheme by driving the MSEL pins high or low. JTAG configuration is always available, regardless of the MSEL settings. The MSEL pins are powered by the VCCINT power supply.

To avoid problems detecting an incorrect configuration scheme, hardwire the MSEL pins to VCCA or GND without pull-up or pull-down resistors. You must not drive the MSEL pins with a microprocessor or another device.

Other Configuration Pins

Table 32. Other Configuration Pins Checklist

Number	Done?	Checklist Item
1		Hold nCE (chip enable) low during configuration, initialization, and user mode.

In a single-device `configuration` or JTAG programming, tie nCE low. In a multi-device configuration, tie nCE low on the first device and connect its nCEO pin to the nCE pin on the next device in the chain.

Board-Related Quartus Prime Settings

Table 33. Board-Related Quartus Prime Settings Checklist

Number	Done?	Checklist Item
1		Set the options for the FPGA I/O pins correctly and plan for the functionality during board design.

The Quartus Prime software provides options for the FPGA I/O pins that you should consider during board design. Ensure that these options are set correctly when the Quartus Prime project is created, and plan for the functionality during board design.

Device-Wide Output Enable Pin

Table 34. Device-Wide Output Enable Pin Checklist

Number	Done?	Checklist Item
1		Turn on the device-wide output enable option, if required.

Cyclone 10 LP devices support an optional chip-wide output enable that allows you to override all tri-states on the device I/Os. When this DEV_0E pin is driven low, all I/O pins are tri-stated; when this pin is driven high, all pins behave as programmed. To use this chip-wide output enable, turn on **Enable device-wide output enable (DEV_OE)** in the Quartus Prime software before compiling your design in the **General** category of the **Device and Pin Options** dialog box. Ensure this pin is driven to a valid logic level on your board if you enable this pin in the Quartus Prime software. Do not leave this pin floating.

Unused Pins

Table 35. Unused Pins Checklist

Number	Done?	Checklist Item
1		Specify the reserved state for unused I/O pins.
2		Carefully check the pin connections in the Quartus Prime software-generated `.pin` file. Do not connect RESERVED pins.

You can specify the state of unused pins in the Quartus Prime software to allow flexibility in the board design by choosing one of the five allowable states for **Reserve all unused pins** on the **Unused Pins** category in the **Device and Pin Options** dialog box:

- **As inputs tri-stated**
- **As output driving ground**
- **As outputs driving an unspecified signal**
- **As input tri-stated with bus-hold circuitry**
- **As input tri-stated with weak pull-up**

The common setting is to set unused pins **As inputs tri-stated with weak pull-up**. To improve signal integrity, set the unused pins to **As output driving ground**. Doing this reduces inductance by creating a shorter return path and reduces noise on the neighboring I/Os. This approach should not be used if this results in many via paths causing congestion for signals under the device.

To reduce power dissipation, set clock pins and other unused I/O pins **As inputs tri-stated**, and tie them to ground.

Signal Integrity Considerations

Signal integrity considerations include detailed board design guidelines, as well as a few guidelines related to VREF pins, SSN, and I/O termination.

Voltage Reference Pins

Table 36. Voltage Reference Pins Checklist

Number	Done?	Checklist Item
1		Design VREF pins to be noise-free.

Voltage deviation on a VREF pin can affect the threshold sensitivity for inputs.

Simultaneous Switching Noise

Table 37. Simultaneous Switching Noise Checklist

Number	Done?	Checklist Item
1		Break out large bus signals on board layers close to the device to reduce cross talk.
2		Route traces orthogonally if two signal layers are next to each other, if possible. Use a separation of two to three times the trace width.

SSN is a concern when too many pins (in close proximity) change voltage levels at the same time. Noise generated by SSN can reduce the noise margin and cause incorrect switching. Although SSN is dominant on the device package, plan the board layout according to the board layout recommendations in the PCB guidelines can help with noise reduction.

I/O Termination

Table 38. I/O Termination Checklist

Number	Done?	Checklist Item
1		Check I/O termination and impedance matching for chosen I/O standards, especially for voltage-referenced standards.

Voltage-referenced I/O standards require both an V_{REF} and a termination voltage (V_{TT}). The reference voltage of the receiving device tracks the termination voltage of the transmitting device. Each voltage-referenced I/O standard requires a unique termination setup.

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

Cyclone 10 LP on-chip series termination provides the convenience of no external components. Alternatively, you can use external pull-up resistors to terminate the voltage-referenced I/O standards such as SSTL and HSTL.

Differential I/O standards typically require a termination resistor between the two signals at the receiver. The termination resistor must match the differential load impedance of the signal line.

Board-Level Simulation and Advanced I/O Timing Analysis

Table 39. Board-Level Simulation and Advanced I/O Timing Analysis Checklist

Number	Done?	Checklist Item
1		Perform board-level simulation using IBIS models (when available).
2		Configure board trace models for Quartus Prime advanced I/O timing analysis.

To ensure that the I/O signaling meets receiver threshold levels on your board setup, perform full board routing simulation with third-party board-level simulation tools using an IBIS model.

When this feature is available in the Quartus Prime software, select **IBIS** under **Board-level signal integrity analysis** on the **Board-Level** page in **EDA Tool Settings** of the **Settings** dialog box.

When you include an FPGA device with high-speed interfaces in a board design, knowing the signal integrity and board routing propagation delay is vital for proper system operation. You should analyze board level timing as part of the I/O and board planning, especially for high-speed designs.

You can configure board trace models of selected I/O standards and generate "board-aware" signal integrity reports with the Quartus Prime software. When **Enable Advanced I/O Timing** is turned on (**TimeQuest Timing Analyzer** page in the **Settings** dialog box), the TimeQuest Timing Analyzer uses simulation results for the I/O buffer, package, and the board trace model to generate more accurate I/O delays and extra reports to give insight into signal behavior at the system level. You can use these advanced timing reports as a guide to make changes to the I/O assignments and board design to improve timing and signal integrity.

I/O and Clock Planning

Planning and allocating I/O and clock resources is an important task with the high pin counts and advanced clock management features in Cyclone 10 LP devices. Various considerations are important to effectively plan the available I/O resources to maximize utilization and prevent issues related to signal integrity. Good clock management systems are also crucial to the performance of an FPGA design.

The I/O and clock connections of your FPGA affect the rest of your system and board design, so it is important to plan these connections early in your design cycle.

Related Links

- Cyclone 10 LP Device Package Planning
  Provides more information about package types.
- I/O and High Speed I/O in Cyclone 10 LP Devices
  Provides more information about the I/O pin counts.

Making FPGA Pin Assignments

Table 40. Making FPGA Pin Assignments Checklist

Number	Done?	Checklist Item
1		Use the Quartus Prime Pin Planner to make pin assignments.
2		Use Quartus Prime Fitter messages and reports for sign-off of pin assignments.
3		Verify that the Quartus Prime pin assignments match those in the schematic and board layout tools.
4		Plan interfaces and device periphery using the BluePrint Platform Designer. After design synthesis, use BluePrint to rapidly define a legal device floorplan. BluePrint planning involves initialization of BluePrint, reconciliation of project assignments, placement of periphery elements and clocks, and export of plan constraints to your Quartus Prime project.

With the Quartus Prime Pin Planner GUI, you can identify I/O banks, VREF groups, and differential pin pairings to help you through the I/O planning process. Right-click in the Pin Planner spreadsheet interface and click the **Pin Finder** to search for specific pins. If migration devices are selected, the Pin Migration view highlights pins that change function in the migration device when compared to the currently selected device.

You have the option of importing a Microsoft Excel spreadsheet into the Quartus Prime software to start the I/O planning process if you normally use a spreadsheet in your design flow. You can also export a spreadsheet compatible (`.csv`) file containing your I/O assignments when all pins are assigned.

When you compile your design in the Quartus Prime software, I/O Assignment Analysis in the Fitter validates that the assignments meet all the device requirements and generates messages if there are any problems.

Quartus Prime designers can then pass the pin location information to PCB designers. Pin assignments between the Quartus Prime software and your schematic and board layout tools must match to ensure the design works correctly on the board where it is placed, especially if changes to the pin-out must be made. The Pin Planner is integrated with certain PCB design EDA tools and can read pin location changes from these tools to check the suggested changes. When you compile your design, the Quartus Prime software generates the `.pin` file. You can use this file to verify that each pin is correctly connected in the board schematics.

Early Pin Planning and I/O Assignment Analysis

Table 41. Early Pin Planning and I/O Assignment Analysis Checklist

Number	Done?	Checklist Item
1		Use the **Create Top-Level Design File** command with I/O Assignment Analysis to check the I/O assignments before the design is complete.

In many design environments, FPGA designers want to plan top-level FPGA I/O pins early so that board designers can start developing the PCB design and layout. The FPGA device's I/O capabilities and board layout guidelines influence pin locations and other types of assignments. In cases where the board design team specifies an FPGA pin-out, it is crucial that you verify pin locations in the FPGA place-and-route software as soon as possible to avoid board design changes.

Starting FPGA pin planning early improves the confidence in early board layouts, reduces the chance of error, and improves the design's overall time to market. You can create a preliminary pin-out for an Intel FPGA using the Quartus Prime Pin Planner before the source code is designed.

Early in the design process, the system architect typically has information about the standard I/O interfaces (such as memory and bus interfaces), IP cores to be used in the design, and any other I/O-related assignments defined by system requirements.

The Pin Planner Create/Import IP Core feature interfaces with the IP catalog, and enables you to create or import custom IP cores that use I/O interfaces. Enter PLL and LVDS blocks, including options such as dynamic phase alignment (DPA), because options affect the pin placement rules. When you have entered as much I/O-related information as possible, generate a top-level design netlist file using the **Create Top-Level Design File** command in the Pin Planner. You can use the I/O analysis results to change pin assignments or IP parameters and repeat the checking process until the I/O interface meets your design requirements and passes the pin checks in the Quartus Prime software.

When planning is complete, the preliminary pin location information can be passed to PCB designers. When the design is complete, use the reports and messages generated by the Quartus Prime Fitter for the final sign-off of the pin assignments.

I/O Features and Pin Connections

Cyclone 10 LP I/O pins are designed for ease of use and rapid system integration, while simultaneously providing high bandwidth. Independent modular I/O banks with a common bank structure for vertical migration lend efficiency and flexibility to the high speed I/O.

The following guidelines provide information pertaining to I/O features and pin connections.

I/O Signaling Type

Table 42. I/O Signaling Type Checklist

Number	Done?	Checklist Item
1		Plan the I/O signaling type based on the system requirements.
2		Allow the software to assign locations for the negative pin in differential pin pairs.

Cyclone 10 LP devices support a wide range of industry I/O standards, including single-ended, voltage-referenced single-ended, and differential I/O standards. Follow these general guidelines when you select a signaling type.

Single-ended I/O signaling provides a simple rail-to-rail interface. Its speed is limited by the large voltage swing and noise. Single-ended I/Os do not require termination, unless reflection in the system causes undesirable effects.

Voltage-referenced signaling reduces the effects of simultaneous switching outputs (SSO) from pins changing voltage levels at the same time. Voltage-referenced signaling also provides an improved logic transition rate with a reduced voltage swing, and minimizes noise caused by reflection with a termination requirement. However, additional termination components are required for the reference voltage source (V_{TT}).

Differential signaling eliminates the interface performance barrier of single-ended and voltage-referenced signaling, with superior speed using an additional inverted closely-coupled data pair. Differential signaling also avoids the requirement for a clean reference voltage. This is possible because of a lower swing voltage and noise immunity with a common mode noise rejection capability. Considerations for this implementation include the requirements for a dedicated PLL to generate a sampling clock, and matched trace lengths to eliminate the phase difference between an inverted and non-inverted pair.

Cyclone 10 LP I/O pins are organized in pairs to support differential I/O standards. Each I/O pin pair can support differential input or output operations, with the exception of certain clock pins that support differential input operations only. In your design source code, define just one pin to represent a differential pair, and make a pin assignment for this positive end of the pair. When you specify a differential I/O standard, the Quartus Prime software automatically places the corresponding negative pin.

Selectable Standards and Flexible I/O Banks

Table 43. Selectable Standards and Flexible I/O Banks Checklist

Number	Done?	Checklist Item
1		Select a suitable signaling type and I/O standard for each I/O pin.
2		Ensure that the appropriate I/O standard support is supported in the targeted I/O bank.
3		Place I/O pins that share voltage levels in the same I/O bank.
4		Verify that all output signals in each I/O bank are intended to drive out at the bank's V_{CCIO} voltage level.
5		Verify that all voltage-referenced signals in each I/O bank are intended to use the bank's V_{REF} voltage level.
6		Check the I/O bank support for LVDS features. Different I/O banks include different support for LVDS signaling.

Cyclone 10 LP devices have eight I/O banks. All differential I/O standards are supported in all banks. The only exception is HSTL-12 Class II, which is only supported in column I/O banks.

You can assign I/O standards and make other I/O-related settings in the Pin Planner.

The board must supply each bank with one V_{CCIO} voltage level for every VCCIO pin in the bank. Each I/O bank is powered by the VCCIO pins of that particular bank and is independent of the VCCIO power supply of other I/O banks. A single I/O bank supports output signals that are driving at the same voltage as the VCCIO power supply. An I/O bank can simultaneously support any number of input signals with different I/O standards, with some exceptions for voltage-referenced inputs. Voltage-referenced standards are supported in an I/O bank using any number of single-ended or differential standards, as long as they use the same VREF and VCCIO values.

When you use the VREF pins as regular I/Os, they have higher pin capacitance than regular user I/O pins. This has an impact on the timing if the pins are used as inputs and outputs.

Dual-Purpose and Special Pin Connections

Table 44. Dual-Purpose and Special Pin Connections Checklist

Number	Done?	Checklist Item
1		Make dual-purpose pin settings and check for any restrictions when using these pins as regular I/O.

Cyclone 10 LP devices allow I/O flexibility with dual-purpose configuration pins. You can use dual-purpose configuration pins as general I/Os after device configuration is complete. Select the desired setting for each of the dual-purpose pins on the **Dual-Purpose Pins** category of the **Device and Pin Options** dialog box. Depending on the configuration scheme, these pins can be reserved as regular I/O pins, as inputs that are tri-stated, as outputs that drive ground, or as outputs that drive an unspecified signal.

You can also use dedicated clock inputs, which drive the programmable clock routing networks, as general-purpose input pins if they are not used as clock pins. When you use the clock inputs as general inputs, I/O registers use LE-based registers because the clock input pins do not include dedicated I/O registers.

The device-wide reset and clear pins are available as design I/Os if they are not enabled.

Cyclone 10 LP I/O Features

Table 45. Cyclone 10 LP I/O Features Checklist

Number	Done?	Checklist Item
1		Check available device I/O features that can help I/O interfaces: current strength, slew rate, I/O delays, open-drain, bus hold, programmable pull-up resistors, PCI clamping diodes, programmable pre-emphasis, and V_{OD}.
2		Consider on-chip termination (OCT) features to save board space.
3		Verify that the required termination scheme is supported for all pin locations.

The Cyclone 10 LP IOE offers a range of programmable features for an I/O pin. These features increase the flexibility of I/O utilization and provide an alternative to reduce the usage of external discrete components to on-chip, such as a pull-up resistor and diode.

Intel recommends performing an IBIS or SPICE simulations to optimize your design settings.

Related Links

IBIS Models

Clock and PLL Selection

Table 46. Clock and PLL Selection Checklist

Number	Done?	Checklist Item
1		Use the correct dedicated clock pins and routing signals for clock and global control signals.
2		Use the device PLLs for clock management.
3		Analyze input and output routing connections for each PLL and clock pin. Ensure PLL inputs come from the dedicated clock pins or from another PLL.

The first stage in planning your clocking scheme is to determine your system clock requirements. Understand your device's available clock resources and correspondingly plan the design clocking scheme. Consider your requirements for timing performance, and how much logic is driven by a particular clock.

Cyclone 10 LP devices provide dedicated low-skew and high fan-out routing networks.

The dedicated clock pins drive the clock network directly, ensuring lower skew than other I/O pins. Use the dedicated routing network to have a predictable delay with less skew for high fan-out signals. You can also use the clock pins and clock network to drive control signals like asynchronous reset.

Connect clock inputs to specific PLLs to drive specific low-skew routing networks. Analyze the global resource availability for each PLL and the PLL availability for each clock input pin.

Use the following information to help determine which clock networks are appropriate for the clock signals in your design:

- The GCLK networks can drive throughout the entire device, serving as low-skew clock sources for device logic. This clock region has the maximum delay when compared with to other clock regions but allows the signal to reach everywhere within the device. This option is good for routing global reset and clear signals or routing clocks throughout the device.
- IOEs and internal logic can also drive GCLKs to create internally generated GCLKs and other high fan-out control signals; for example, synchronous or asynchronous clears and clock enables.
- PLLs cannot be driven by internally generated GCLKs. The input clock to the PLL must come from dedicated clock input pins, fed by a pin-driven dedicated GCLK, or through a clock control block if the clock control block is fed by an output from another PLL or a pin-driven dedicated GCLK.

If your system requires more clock or control signals than are available in the target device, consider cases where the dedicated clock resource could be spared, particularly low fan-out and low-frequency signals where clock delay and clock skew do not have a significant impact on the design performance. Use the **Global Signal** assignment in the Quartus Prime Assignment Editor to select the type of global routing, or set the assignment to **Off** to specify that the signal should not use any global routing resources.

PLL Feature Guidelines

Table 47. PLL Feature Guidelines Checklist

Number	Done?	Checklist Item
1		Enable PLL features and check settings in the parameter editor.

Based on your system requirements, define the required clock frequencies for your FPGA design, and the input frequencies available to the FPGA. Use these specifications to determine your PLL scheme. Use the Quartus Prime parameter editor to enter your settings in ALTPLL IP core, and check the results to verify whether particular features and input/output frequencies can be implemented in a particular PLL.

Cyclone 10 LP device PLLs support several features for general-purpose clock management including clock feedback modes and switchover.

Clock Feedback Mode

Table 48. Clock Feedback Mode Checklist

Number	Done?	Checklist Item
1		Ensure you select the correct PLL feedback compensation mode.

Cyclone 10 LP PLLs support five different clock feedback modes. Each mode compensates for different clock networks and delays so the clocks are aligned differently.

Clock Outputs

Table 49. Clock Outputs Checklist

Number	Done?	Checklist Item
1		Check that the PLL offers the required number of clock outputs and use dedicated clock output pins.

You can connect clock outputs to dedicated clock output pins or dedicated clock networks. There is no dedicated clock out pin for fractional PLL. I/O PLL can connect to a clock network or a dedicated clock pin.

Clock Control Features

Table 50. Clock Control Features Checklist

Number	Done?	Checklist Item
1		Use the clock control block for clock selection and power-down.

In Cyclone 10 LP devices, dedicated clock input pins, PLL counter outputs, dual-purpose clock I/O inputs, and internal logic can all feed the clock control block for each GCLK. The output from the clock control block, in turn feeds the corresponding GCLK. The GCLK can drive the PLL input if the clock control block inputs are outputs of another PLL or dedicated clock input pins.

I/O Simultaneous Switching Noise

Table 51. I/O Simultaneous Switching Noise Checklist

Number	Done?	Checklist Item
1		Reduce the number of pins that switch the voltage level at exactly the same time whenever possible.
2		Use differential I/O standards and lower-voltage standards for high-switching I/Os.
3		Use lower drive strengths for high-switching I/Os. The default drive strength setting might be higher than your design requires.
4		Reduce the number of simultaneously switching output pins within each bank. Spread output pins across multiple banks if possible.
5		Spread switching I/Os evenly throughout the bank to reduce the number of aggressors in a given area to reduce SSN (when bank usage is substantially below 100%).
6		Separate simultaneously switching pins from input pins that are susceptible to SSN.
7		Place important clock and asynchronous control signals near ground signals and away from large switching buses.
8		Avoid using I/O pins one or two pins away from PLL power supply pins for high-switching or high-drive strength pins.
9		Use staggered output delays to shift the output signals through time, or use adjustable slew rate settings.

SSN is a concern when too many I/Os (in close proximity) change voltage levels at the same time. Plan the I/O and clock connections according to the recommendations.

Design Entry

In complex FPGA design development, design practices, coding styles, and IP cores use have an enormous impact on your device's timing performance, logic utilization, and system reliability. In addition, while planning and creating the design, plan for a hierarchical or team-based design to improve design productivity.

Design Recommendations

Table 52. Design Recommendations Checklist

Number	Done?	Checklist Item
1		Use synchronous design practices. Pay attention to clock signals.
2		Use the Quartus Prime Design Assistant to check design reliability.

In a synchronous design, a clock signal triggers all events. When all of the registers' timing requirements are met, a synchronous design behaves in a predictable and reliable manner for all process, voltage, and temperature (PVT) conditions. You can easily target synchronous designs to different device families or speed grades.

Problems with asynchronous design techniques include reliance on propagation delays in a device, incomplete timing analysis, and possible glitches. Pay particular attention to your clock signals, because they have a large effect on your design's timing accuracy, performance, and reliability. Problems with clock signals can cause functional and timing problems in your design. Use dedicated clock pins and clock routing for best results. For clock inversion, multiplication, and division, use the device PLLs. For

clock multiplexing and gating, use the dedicated clock control block or PLL clock switchover feature instead of combinational logic. If you must use internally generated clock signals, register the output of any combinational logic used as a clock signal to reduce glitches. For example, if you divide a clock using combinational logic, clock the final stage with the clock signal that was used to clock the divider circuit.

The Design Assistant in the Quartus Prime software is a design-rule checking tool that enables you to check for design issues early in the design flow. The Design Assistant checks your design for adherence to Intel recommended design guidelines or design rules. To run the Design Assistant, on the Processing menu, point to **Start** and click **Start Design Assistant**. To set the Design Assistant to run automatically during compilation, turn on **Run Design Assistant during compilation** in the **Settings** dialog box. You can also use third-party "lint" tools to check your coding styles.

Using IP Cores

Table 53. Using IP Cores Checklist

Number	Done?	Checklist Item
1		Use IP cores with the parameter editor.

Intel provides parameterizable IP cores that are optimized for Intel device architectures. You can save design time by using IP cores instead of coding your own logic. Additionally, the Intel-provided IP cores can offer more efficient logic synthesis and device implementation. You can scale the IP core's size and set various options with parameters. IP cores include the library of parameterized modules (LPM) and Intel device-specific IP cores. You can also take advantage of Intel and third-party IP cores and reference designs to save design time. The Quartus Prime IP catalog provides a user interface to customize IP cores. You should build or change IP core parameters using the parameter editor to ensure you set all ports and parameters correctly.

Recommended HDL Coding Styles

Table 54. Recommended HDL Coding Styles Checklist

Number	Done?	Checklist Item
1		Follow recommended coding styles, especially for inferring device dedicated logic such as memory and DSP blocks.

HDL coding styles can have a significant effect on the quality of results for programmable logic designs. Use Intel's recommended coding styles to achieve optimal synthesis results. When designing memory and digital system processing (DSP) functions, understand the device architecture so you can take advantage of the dedicated logic block sizes and configurations.

Register Power-Up Levels and Control Signals

Table 55. Register Power-Up Levels and Control Signals Checklist

Number	Done?	Checklist Item
1		Enable the chip-wide reset to clear all registers if required.
2		Consider resources available for register power-up and control signals. Do not apply both reset and preset signals to a register.

Cyclone 10 LP devices support an optional chip-wide reset that enables you to override all clears on all device registers, including the registers of the memory blocks (but not the memory contents itself). When this DEV_CLRn pin is driven low, all registers are cleared or reset to 0. If synthesis performs an optimization called NOT-gate-push back due to register control signals, the affected registers behave as though they are preset to a high value when DEV_CLRn is driven low. When the DEV_CLRn pin is driven high, all registers behave as programmed. To use this chip-wide reset, turn on **Enable device-wide reset (DEV_CLRn)** in the Quartus Prime software on the **General** category of the **Device and Pin Options** dialog box before compiling your design.

Each Cyclone 10 LP logic array block (LAB) also contains dedicated logic for driving register control signals to its LEs. Register control signals restrict how registers are packed into LABs because signals are shared within the LAB. It is important that control signals use the dedicated control signals in the device architecture, so in some cases you might be required to limit the number of different control signals used in your design.

If the clock signal is not available when reset is asserted, an asynchronous reset is typically used to reset the logic. The recommended reset architecture allows the reset signal to be asserted asynchronously and de-asserted synchronously. The source of the reset signal is then connected to the asynchronous port of the registers, which can be directly connected to global routing resources. The synchronous de-assertion allows all state machines and registers to start at the same time. It also avoids the possibility that an asynchronous reset signal is released at or near the active clock edge of a flipflop, in which case the output of the flipflop could go to a metastable unknown state.

By default, the Quartus Prime integrated synthesis enables the logic option called **Power-Up Don't Care**, which assumes your design does not depend on the power-up state of the device architecture and allows the software to remove registers that become stuck high. Other synthesis tools might use similar assumptions.

Designers typically use an explicit reset signal for the design that forces all registers into their appropriate values after reset but not necessarily at power-up. You can create your design such that the asynchronous reset allows the board to operate in a safe condition. You can then bring up the design with the reset active. Thus, you do not have to depend on the power-up conditions of the device.

If you force a particular power-up condition for your design, use the synthesis options available in your synthesis tool.

Some synthesis tools can also read the default or initial values for registered signals in your source code and implement this behavior in the device. For example, the Quartus Prime integrated synthesis converts HDL default and initial values for registered signals into **Power-Up Level** settings. That way, the synthesized behavior matches the power-up state of the HDL code during a functional simulation.

Registers in the device core always power up to a low (0) logic level in the physical device architecture. If you specify a high power-up level or a non-zero reset value (often called a preset signal), synthesis tools typically use the clear signals available on the registers and perform an optimization referred to as NOT-gate push back.

If you assign a high power-up level to a register that is reset low, or assign a low power-up value to a register that is preset high, synthesis tools cannot use the NOT-gate push back optimization technique and might ignore the power-up conditions.

To implement a reset and preset signal on the same register, synthesis tools emulate the controls with logic and latches that can be prone to glitches because of the different delays between the different paths to the register. In addition, the power-up value is undefined for these registers.

Design Implementation, Analysis, Optimization, and Verification

After you create your design source code and apply constraints including the device selection and timing requirements, your synthesis tool processes the code and maps it to elements of the device architecture. The Quartus Prime Fitter then performs placement and routing to implement the design elements in specific device resources. If required, you can use the Quartus Prime 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 or formal verification. This section provides guidelines for these stages of the compilation flow.

Selecting a Synthesis Tool

Table 56. Selecting a Synthesis Tool Checklist

Number	Done?	Checklist Item
1		Specify your synthesis tool and use the correct supported version.

The Quartus Prime software includes advanced and easy-to-use integrated synthesis that fully supports Verilog HDL and VHDL, as well as the Intel hardware description language (AHDL) and schematic design entry. You can also use industry-leading third-party EDA synthesis tools to synthesize your Verilog HDL or VHDL design, and then compile the resulting output netlist file in the Quartus Prime software. Specify a 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 (`.lmf`) for your synthesis netlist.

Intel recommends using the most recent version of third-party synthesis tools, because tool vendors are continuously adding new features, fixing tool issues, and enhancing performance for Intel devices.

Different synthesis tools can give different results. If you want to select the best-performing tool for your application, you can experiment by synthesizing typical designs for your application and coding style and comparing the results. Be sure to perform placement and routing in the Quartus Prime software to get accurate timing analysis and logic utilization results.

Your synthesis tool might offer the capability to create a Quartus Prime 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 Prime project for placement and routing.

Device Resource Utilization Reports

Table 57. Device Resource Utilization Reports Checklist

Number	Done?	Checklist Item
1		Review resource utilization reports after compilation.

After compilation in the Quartus Prime software, review the device resource utilization information to determine whether the future addition of extra logic or other design changes introduce fitting difficulties. If your compilation results in a no-fit error, resource utilization information is important so you can analyze the fitting problems in your design.

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. The Fitter **Resource Usage Summary** report breaks down the logic utilization information and indicates the usage of LEs, and provides other resource information including the number of bits in each type of memory block. There are also reports that describe some of the optimizations that occurred during compilation. For example, if you use the Quartus Prime integrated synthesis, the reports under the **Optimization Results** folder in the **Analysis & Synthesis** section provide information, including registers that were removed during synthesis. Use this report to estimate 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 Prime Messages

Table 58. Quartus Prime Messages Checklist

Number	Done?	Checklist Item
1		Review all Quartus Prime messages, especially warning or error messages.

Each stage of the compilation flow generates messages, including informational notes, warnings, and critical warnings. Review these messages to check for any design problems. Ensure that you understand the significance of any warning messages, and make changes to the design or settings if required. In the Quartus Prime user interface, you can use the **Message** window tabs to look at only certain types of messages, and you can suppress messages if you have determined that they do not require any action from you.

Timing Constraints and Analysis

Table 59. Design Specifications Checklist

Number	Done?	Checklist Item
1		Ensure timing constraints are complete and accurate, including all clock signals and I/O delays.
2		Review the TimeQuest Timing Analyzer reports after compilation to ensure there are no timing violations.
3		Ensure that the input I/O times are not violated when data is provided to the Cyclone 10 LP device.

In an FPGA design flow, accurate timing constraints allow timing-driven synthesis software and place-and-route software to obtain optimal results. Timing constraints are critical to ensure designs meet their timing requirements, which represent actual design requirements that must be met for the device to operate correctly. The Quartus Prime software optimizes and analyzes your design using different timing models for

each device speed grade, so you must perform timing analysis for the correct speed grade. The final programmed device might not operate as expected if the timing paths are not fully constrained, analyzed, and verified to meet requirements.

The Quartus Prime software includes the Quartus Prime 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. Specify accurate timing requirements for external interfacing components to reflect the exact system intent.

The TimeQuest 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. It determines the timing relationships that must be met for the design to correctly function.

You can use the `report_datasheet` command to generate a datasheet report that summarizes the I/O timing characteristics of the entire design.

Recommended Timing Optimization and Analysis Assignments

Table 60. Recommended Timing Optimization and Analysis Assignments Checklist

Number	Done?	Checklist Item
1		Turn on **Optimize multi-corner timing** on the **Fitter Settings** page in the **Settings** dialog box.
2		Use `create_clock` and `create_generated_clock` to specify the frequencies and relationships for all clocks in your design.
3		Use `set_input_delay` and `set_output_delay` to specify the external device or board timing parameters.
4		Use `derive_pll_clocks` to create generated clocks for all PLL outputs, according to the settings in the PLL IP cores. Specify multicycle relationships for LVDS transmitters or receiver deserialization factors.
5		Use `derive_clock_uncertainty` to automatically apply inter-clock, intra-clock, and I/O interface uncertainties.
6		Use `check_timing` to generate a report on any problem with the design or applied constraints, including missing constraints.

These assignments and settings are important for large designs such as those in Cyclone 10 LP devices.

When you turn on the **Optimize multi-corner timing** option, the design is optimized to meet its timing requirements at all timing process corners and operating conditions. Therefore, turning on this option helps create a design implementation that is more robust across PVT variations.

In your TimeQuest Timing Analyzer .sdc constraints file, apply the recommended constraints to your design.

Area and Timing Optimization

Table 61. Area and Timing Optimization Checklist

Number	Done?	Checklist Item
1		Perform Early Timing Estimation if you want timing estimates before running a full compilation.
2		Use Quartus Prime optimization features to achieve timing closure or improve the resource utilization.
3		Use the Timing and Area Optimization Advisors to suggest optimization settings.

This section highlights some of the features offered in the Quartus Prime software to help optimize area (or resource utilization) and timing performance. If the timing analysis reports that your design requirements were not met, you must make changes to your design or settings and recompile the design to achieve timing closure. If your compilation results in no-fit messages, you must make changes to get successful placement and routing.

You can use the Early Timing Estimation feature to estimate your design's timing results before the software performs full placement and routing. 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.

Physical synthesis optimizations make placement-specific changes to the netlist that improve results for a specific Intel device. You can optimize for performance and fitting in the **Physical Synthesis Optimizations** page of the **Settings** dialog box. The options in the **Physical Synthesis Optimizations** page typically increase compilation time significantly but can provide significant improvements to the quality of results with push-button optimizations. If you turn on these options, ensure that they do improve the results for your design. If you do not require these options to meet your design timing requirements, turn off the options to reduce the compilation time.

The Design Space Explorer (DSE) is a utility that automates the process of finding the optimal collection of the Quartus Prime software settings for your design. The **Search for Best Performance** and **Search for Best Area** options under **Exploration Settings** use a predefined exploration space to target design performance or area improvements with multiple compilations. You can also set the **Optimization Goal** to **Optimize for Speed** or **Optimize for Area** using the **Advanced** tab in the DSE window.

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 your requirements.

Preserving Performance and Reducing Compilation Time

Table 62. Preserving Performance and Reducing Compilation Time Checklist

Number	Done?	Checklist Item
1		Use incremental compilation to preserve performance for unchanged blocks in your design and to reduce compilation times.
2		Ensure parallel compilation is enabled if you have multiple processors available for compilation.
3		Use the Compilation Time Advisor to suggest settings that reduce compilation time.

Use the incremental compilation feature to preserve logic in unchanged parts of your design, preserve timing performance, and reach timing closure more efficiently. You can speed up design iteration time by an average of 60% when making changes to the design with the incremental compilation feature.

The Quartus Prime 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. Set the **Parallel compilation** option on the **Compilation Process Settings** page of the **Settings** dialog box, or change the default setting in the **Options** dialog box in the **Processing** page from the Tools menu.

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.

Simulation

Table 63. Simulation Checklist

Number	Done?	Checklist Item
1		Specify your simulation tool, and use the correct supported version and simulation models.

The Quartus Prime software supports both RTL and gate level functional simulations. Perform functional simulation at the beginning of your design flow to check the design functionality or logical behavior of each design block. You do not have to fully compile your design; you can generate a functional simulation netlist that does not contain timing information.

Intel provides the ModelSim® - Intel FPGA Starter Edition and offers the higher-performance ModelSim - Intel FPGA Edition, which enable you to take advantage of advanced testbench capabilities and other features. In addition, the Quartus Prime EDA Netlist Writer can generate timing netlist files to support other third-party simulation tools such as Synopsys VCS, Cadence NC-Sim, and Aldec Active-HDL. Specify your simulation tool in the **EDA Tools Settings** page of the **Settings** dialog box to generate the appropriate output simulation netlist. The software can also generate scripts to help you setup libraries in your tool with NativeLink integration.

If you use a third-party simulation tool, use the software version that is supported with your Quartus Prime software version. The Quartus Prime Software Release Notes list the version of each simulation tool that is officially supported with that particular version of the Quartus Prime software. Use the model libraries provided with your

Quartus Prime software version, because libraries can change between versions, which might cause a mismatch with your simulation netlist. To create a testbench, on the Processing menu, point to **Start** and click **Start Testbench Template Writer**.

Power Analysis

Table 64. Power Analysis Checklist

Number	Done?	Checklist Item
1		After compilation, analyze power consumption and heat dissipation in the Power Analyzer.
2		Provide accurate signal activities, preferably with a gate-level simulation .vcd, to get accurate power analysis results.
3		Specify the correct operating conditions for power analysis.

Before design completion, estimate power consumption using the EPE spreadsheet. After compiling your design, analyze the power consumption and heat dissipation with the Quartus Prime Power Analyzer to ensure the design has not violated power supply and thermal budgets.

You must compile a design (to provide information about design resources, placement and routing, and I/O standards) and provide signal activity data (toggle rates and static probabilities) to use the Power Analyzer. You can derive signal activity data from simulation results or a user-defined default toggle rate and vectorless estimation. The signal activities used for analysis must be representative of the actual operating behavior. For the most accurate power estimation, use gate-level simulation results with a .vcd output file from a third-party simulation tool. The simulation activity should include typical input vectors over a realistic time period and not the corner cases often used during functional verification. Use the recommended simulator settings (such as glitch filtering) to ensure good results.

You must also specify operating conditions, including the core voltage, device power characteristics, ambient and junction temperature, cooling solution, and the board thermal model. Select the appropriate settings on the **Operating Settings and Conditions** page in the **Settings** dialog box.

To calculate the dynamic, static, and I/O thermal power consumption, on the Processing menu, click **Power Analyzer Tool**. The tool also provides a summary of the signal activities used for analysis and a confidence metric that reflects the overall quality of the data sources for signal activities.

The report is a power estimate based on the data provided, and is not a power specification. Always refer to the datasheet for the power specification of your device.

Power Optimization

Cyclone 10 LP devices use architectural power reduction techniques to minimize power and deliver high performance. To reduce dynamic power consumption in Cyclone 10 LP devices, you can use various design and software techniques to optimize your design.

Power optimization in the Quartus Prime software depends on accurate power analysis results. Use the guidelines in the previous section to ensure the software optimizes the power utilization correctly for the design's operating behavior and conditions.

Device and Design Power Optimization Techniques

Table 65. Device and Design Power Optimization Techniques Checklist

Number	Done?	Checklist Item
1		Use recommended design techniques and Quartus Prime options to optimize your design for power consumption, if required.
2		Use the Power Optimization Advisor to suggest optimization settings.

Clock Power Management

Table 66. Clock Power Management Checklist

Number	Done?	Checklist Item
1		Optimize the clock power management.

Clocks represent a significant portion of dynamic power consumption, because of their high switching activity and long paths. The Quartus Prime software automatically optimizes clock routing power by enabling only the portions of a clock network that are required to feed downstream registers. You can also use clock control features to dynamically enable or disable the clock network. 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.

To reduce LAB-wide clock power consumption without disabling the entire clock tree, use the LAB-wide clock enable signal to gate the LAB-wide clock. The Quartus Prime software automatically promotes register-level clock enable signals to the LAB level.

Memory Power Reduction

Table 67. Memory Power Reduction Checklist

Number	Done?	Checklist Item
1		Reduce the number of memory clocking events.

Reduce the number of memory clocking events to reduce memory power consumption. You can use clock gating or the clock enable signals in the memory ports.

I/O Power Guidelines

Table 68. I/O Power Guidelines Checklist

Number	Done?	Checklist Item
1		Review the I/O power guidelines.

The dynamic power consumed in the I/O buffer is proportional to the total load capacitance; therefore, lower capacitance reduces power consumption.

Non-terminated I/O standards such as LVTTL and LVCMOS have a rail to-rail output swing equal to the V_{CCIO} supply voltage. Because dynamic power is proportional to the square of the voltage, use lower voltage I/O standards to reduce dynamic power. These I/O standards consume little static power.

Because dynamic power is also proportional to the output transition frequency, use resistively-terminated I/O standards such as SSTL for high-frequency applications. The output load voltage swings by an amount smaller than the V_{CCIO} around a bias point; therefore, dynamic power is lower than for non-terminated I/O under similar conditions.

Resistively-terminated 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 when using resistively terminated I/O standards.

The power used by external devices is not included in the EPE calculations, so be sure to include it separately in your system power calculations.

Quartus Prime Power Optimization Techniques

Table 69. Quartus Prime Power Optimization Techniques Checklist

Number	Done?	Checklist Item
1		Review recommended design techniques and Quartus Prime options to optimize power consumption.
2		Use the Power Optimization Advisor to suggest optimization settings.

The Quartus Prime software offers power-optimized synthesis and fitting to reduce core dynamic power.

Optimizing your design for area saves power because fewer logic blocks are used; therefore typically less switching activity. You can use the DSE and Power Optimization Advisor to provide additional suggestions to reduce power.

The Quartus Prime software includes the Power Optimization Advisor, which provides specific power optimization advice and recommendations based on the current design project settings and assignments. On the Tools menu, point to **Advisors** and click **Power Optimization Advisor**. After making any of the recommended changes, recompile your design and run the Power Analyzer to check the change in your power results.

Revision History

Date	Version	Changes
May 2017	2017.05.08	Initial release.