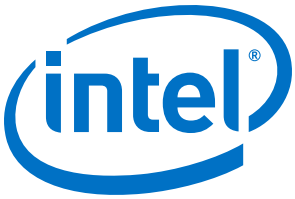


Intel® MAX® 10 FPGA Design Guidelines

Subscribe
Send Feedback

M10-GUIDELINES | 2017.05.03
Latest document on the web: PDF | HTML

Contents

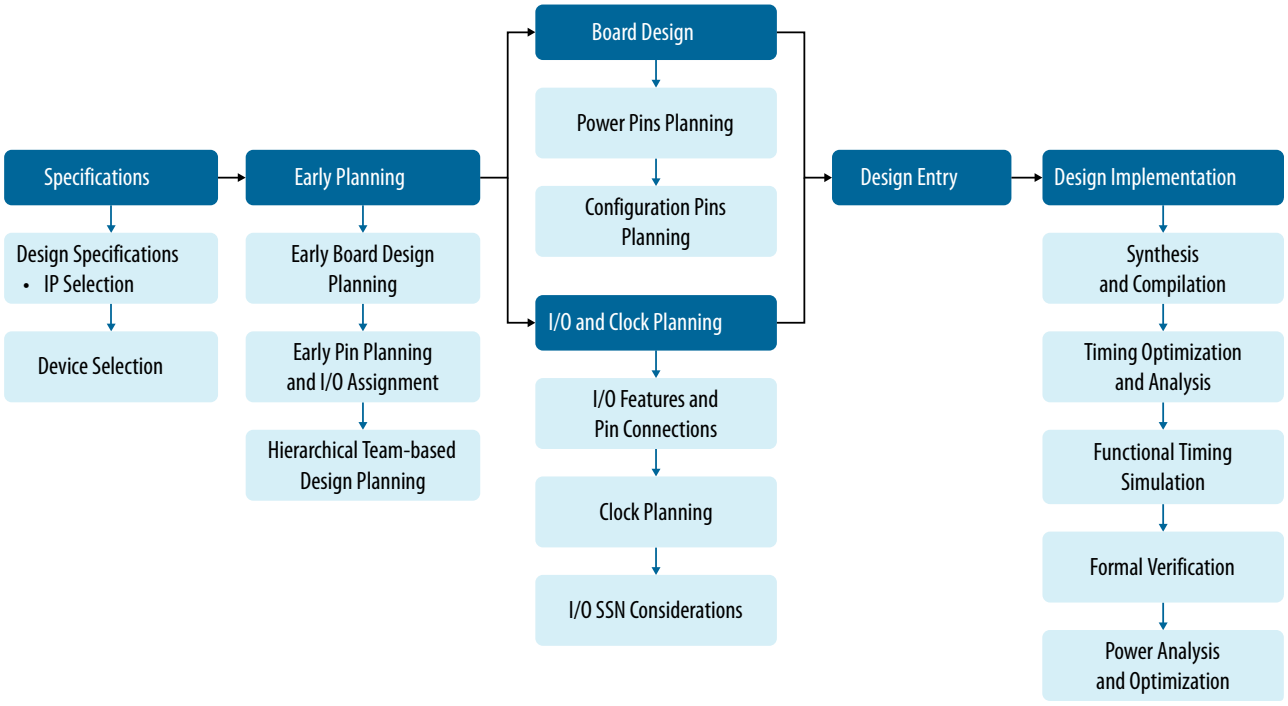
1. Intel® MAX® 10 FPGA Design Guidelines

1.1. Intel® MAX® 10 FPGA Design Guidelines

This application note provides a set of checklists that consist of design guidelines, recommendations, and factors to consider when you create designs using MAX® 10 FPGAs.

- Use this document to help you plan the FPGA and system early in the design process, which is crucial for a successful design.
- Follow Intel's recommendations throughout the design process to achieve good results, avoid common issues, and improve your design productivity.

Figure 1. MAX 10Design Flow



1.2. Before You Begin

Before you begin planning and designing your FPGA system, familiarize yourself with the FPGA device features, and the design tools and IP that are available for the MAX 10 device family.

1.2.1. Read through the Device Overview of the FPGA

The Device Overview provides an overview of the capabilities and options available for a device family. Read through the document to familiarize yourself with the device family offerings and general features.

Related Information

MAX 10 FPGA Device Overview

1.2.2. Estimate design requirements

Create a rough estimate of the design in the following terms:

- Basic functions of the product
- Similar previous designs
- General device requirements

1.2.3. Review available design tools

Consider the available design, estimators, system builders, and verification tools. The following items are some of the available tools provided by Intel:

- Quartus® Prime software for design, synthesis, simulation, and programming; including integration with Qsys, simulation tools, and verification tools.
- Qsys system integration tool—next-generation tool that automatically generates interconnect logic to connect intellectual property (IP) functions and subsystems.
- Mentor Graphics* ModelSim* - Intel FPGA Edition simulation software.
- TimeQuest Timing Analyzer for static timing analysis with support for Synopsys* Design Constraints (SDC) format.
- PowerPlay® Power Analyzer for power analysis and optimization.
- Signal Probe and Signal Tap II Logic Analyzer debugging tools.
- External Memory Interface Toolkit available in the Quartus Prime software.

Related Information

- Design Tools Services
  For more information design tools
- Design Software Support
  For more information on software support
- AN 632: SOPC Builder to Qsys Migration Guidelines
  For a guideline to migrate from SOPC Builder to Qsys

1.2.4. Review available IP

Intel and its FPGA IP partners offer a large selection of parameterized blocks of IP cores optimized for Intel® FPGAs that you can implement to reduce your implementation and verification time.

Related Information

All Intellectual Property

1.3. Design Specifications

Typically, the FPGA is an important part of the overall system and affects the rest of the system design. Use the following checklist to start your design process.

1.3.1. Create detailed design specifications

Before you create your logic design or complete your system design, perform the following:

- Specify the I/O interfaces for the FPGA
- Identify the different clock domains
- Include a block diagram of basic design functions
- Consider a common design directory structure—if your design includes multiple designers, a common design directory structure eases the design integration stages.
- When performing any UFM write or erase operation, make sure you provide stable power connection. Loss of power supply during a write or erase operation can cause damage to the device.

1.3.2. Create detailed functional verification or test plan

A functional verification plan ensures the team knows how to verify the system. Creating a test plan at an early stage helps you design for testability and manufacturability.

For example, if you plan to perform built-in-self test (BIST) functions to drive interfaces, you can plan to use a UART interface with a Nios® II processor inside the FPGA device.

Related Information

Review available on-chip debugging tools on page 11

1.3.3. Select IP that affects system design, especially I/O interfaces

Include intellectual property (IP) blocks in your detailed design specifications. Taking the time to create these specifications improves design efficiency.

Related Information

All Intellectual Property
For a list of available IP offered by Intel and Intel FPGA IP partners

1.3.4. Ensure your board design supports the OpenCore Plus tethered mode

You can program your FPGA and verify your design in hardware before you purchase an IP license by using the OpenCore Plus feature available for many IP cores. OpenCore Plus supports the following modes:

- Untethered—your design runs for a limited time.
- Tethered—your design runs for the duration of the hardware evaluation period. This mode requires an Intel FPGA download cable connected to the JTAG port on your board and a host computer that runs the Quartus Prime Programmer. If you plan to use this mode, ensure that your board design supports this mode.

1.3.5. Review available system development tools

Intel provides a complete suite of development tools for every stage of your design.

Whether you are creating a complex FPGA design as a hardware engineer, writing software for an embedded processor as a software developer, modeling a digital signal processing (DSP) algorithm, or focusing on system design, Intel has a tool that can help.

Related Information

- Design Tools Services.
- Design Software Support.

1.4. Device Selection

Use the following checklist to determine the device variant, density, and package combination that is suitable for your design.

1.4.1. Consider the available device variants

The MAX 10 FPGA family consist of several device variants that are optimized for different application requirements.

Select a device based on I/O pin count, LVDS channels, package offering, logic/memory/multiplier density, single/dual supply device options, PLLs, clock routing, and speed grade.

Consider the following feature options:

- Compact
- Flash
- Analog-to-digital converter (ADC)

Related Information

MAX 10 FPGA Device Overview

1.4.2. Estimate the required logic, memory, and multiplier density

MAX 10 devices offer a range of densities that provide different amounts of device logic resources. 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.

Related Information

MAX 10 Embedded Multipliers User Guide

1.4.3. Consider vertical device migration availability and requirements

Determine whether you want the flexibility 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.

To verify the pin migration compatibility, use the **Pin Migration View** window in the Quartus Prime software Pin Planner. The **Pin Migration View** window helps you identify the difference in pins that can exist between migration devices:

- If one device has pins for connection to V_{CC} or GND but are I/O pins on a different device, the Quartus Prime software ensures these pins are not used for I/O. For migration, ensure that these pins are connected to the correct PCB plane.
- If you are migrating between two devices in the same package, connect the pins that are not connected to the smaller die to V_{CC} or GND on the larger die in your original design.

Related Information

I/O Management
For more information about verifying the pin migration compatibility

1.4.4. Review resource utilization reports of similar designs

If you have other designs that target an Intel device, you can use their resource utilization as an estimate for your new design. Coding style, device architecture, and optimization options used in the Quartus Prime software can significantly affect resource utilization and timing performance of a design.

To estimate resource utilization for certain configurations of Intel's FPGA IP designs, refer to the respective MAX 10 user guides.

1.4.5. Reserve device resources for future development and debugging

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 may also want additional space in the device to ease design floorplan creation for an incremental or team-based design.

Related Information

Consider reserving resources for debugging on page 11

1.4.6. Estimate the number of I/O pins that you require

Determine the required number of I/O pins for your application, considering the design's interface requirements with other system blocks. You can compile any existing designs in the Quartus Prime software to determine how many I/O pins are used.

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.

Related Information

- I/O and Clock Planning on page 21
- Board Design on page 10

1.4.7. Consider the I/O pins you need to reserve for debugging

Intel provides a complete design debugging environment that easily adapts to your specific design requirements. When planning to debugging, you should decide which I/O pins you need to reserve for debugging.

Related Information

Consider the guidelines to plan for debugging tools on page 11

1.4.8. Verify that the number of LVDS channels are enough

Larger densities and package pin counts offer more full-duplex LVDS channels for differential signaling. Ensure that your device density-package combination includes enough LVDS channels.

1.4.9. 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. GCLK resources are shared between certain PLLs, which can affect the inputs that are available for use.

Related Information

I/O and Clock Planning on page 21
For more details and references regarding clock pins and global routing resources

1.4.10. 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 while prototyping to reduce compilation time because less time is spent optimizing the design to meet timing requirements. If the design meets the timing requirements, you can then move to a slower speed grade for production to reduce cost.

When migrating to a device of different speed grade, check the timing report from the timing analysis to ensure that there is no timing violation between different blocks within the MAX 10 device, between MAX 10 devices and other devices on the board.

Always design with a sufficient timing margin so that your design can work on devices of different speed grades.

Related Information

- External Memory Interface Spec Estimator
  For information about supported clock rates for memory interfaces using I/O pins on different sides of the device in different device speed grades
- MAX 10 FPGA Device Datasheet
  For information about the available speed grades

1.4.11. Determine the number of images supported for the device

Select a device that support dual configuration images, two FPGA bitstreams, if dual configuration feature is needed in your design. All MAX 10 devices support the dual configuration feature, except 10M02 device.

1.5. Board Design

1.5.1. Early Board Design

Early planning allows the FPGA team to provide early information to PCB board and system designers.

1.5.1.1. Select a configuration scheme

Intel offers a wide range of configuration solutions to configure MAX 10 devices.

Related Information

MAX 10 FPGA Configuration User Guide
For information about the internal configuration scheme, and the necessary and optional pin settings

1.5.1.2. Ensure board support the required features:

- Data decompression—if you enable data compression, the storage requirement and the programming time (writing to flash) are reduced. The configuration time (writing to CRAM) is increased.
- Design security—this feature utilizes a 128-bit security key to protect the designs from unauthorized copying, reverse engineering, and tampering. The devices can decrypt configuration bitstreams using the AES algorithm. Design security is not available for the JTAG configuration scheme.
- Dual configuration—this feature is supported only in self-download mode.
- SEU mitigation—dedicated circuitry in the devices perform cyclic redundancy check (CRC) error detection and check for SEU errors automatically. To detect SEU errors, use the CRC_ERROR pin to flag errors and design your system to take appropriate action. If you do not enable the CRC error detection feature, you can also use the CRC_ERROR pin as a design I/O pin.

Related Information

MAX 10 FPGA Configuration User Guide

1.5.1.3. Plan for the Auto-restart after configuration error option

To reset the device internally by driving the nSTATUS pin low when a configuration error occurs, enable the **Auto-restart after configuration error** option. The device releases its nSTATUS pin after the reset time-out period. This behavior allows you to re-initiate the configuration cycle. The nSTATUS pin requires an external 10-kΩ pull-up resistor to V_{CCIO}.

1.5.1.4. Estimating configuration file size

To estimate the configuration file size, convert your configuration file in uncompressed Raw Binary File (**.rbf**). The **.rbf** file size provides the approximate uncompressed configuration file sizes.

Use uncompressed **.rbf** size only to estimate the file size before design compilation. Different configuration file formats, such as Hexadecimal (Intel-Format) File (**.hex**) or Tabular Text File (**.ttf**) format, have different file sizes.

Related Information

MAX 10 FPGA Configuration User Guide
For more information on the uncompressed .rbf sizes for MAX 10 devices

1.5.1.5. Review available on-chip debugging tools

Take advantage of on-chip debugging features to analyze internal signals and perform advanced debugging techniques.

Different debugging tools work better for different systems and different designers. Early planning can reduce the time spent debugging, and eliminates design changes later to accommodate your preferred debugging methodologies. Adding debug pins may not be enough, because of internal signal and I/O pin accessibility on the device.

Related Information

- System Debugging Tools Overview
  For more information about in-system debugging tools in the Quartus Prime software
- Virtual JTAG (sld_virtual_jtag) Megafunction User Guide
  For more information about virtual JTAG

1.5.1.6. Consider the guidelines to plan for debugging tools

- Select on-chip debugging schemes early to plan memory and logic requirements, I/O pin connections, and board connections.
- If you want to use Signal Probe incremental routing, the Signal Tap II 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.
- Plan for the small amount of additional logic resources used to implement the JTAG hub logic for JTAG debugging features.

- For debugging with the Signal Tap II Embedded Logic Analyzer, reserve device memory resources to capture data during system operation.
- Reserve I/O pins for debugging with SignalProbe or the Logic Analyzer Interface so that you do not have to change the design or board to accommodate debugging signals later.
- Ensure the board supports a debugging mode where debugging signals do not affect system operation.
- Incorporate a pin header or micro connector as required for an external logic analyzer or mixed signal oscilloscope.
- 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.
- To use the Virtual JTAG IP core for custom debugging applications, instantiate it in the HDL code as part of the design process.
- To use the In-System Sources and Probes feature, instantiate the IP core in the HDL code.
- 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** option to capture and update content independently of the system clock option for the memory block in the parameter editor.

Related Information

- Quick Design Debugging Using Signal Probe
  For more information about debugging tools and methods
- Design Debugging Using the Signal Tap II Logic Analyzer
  For more information about debugging withSignal Tap II
- In-System Debugging Using External Logic Analyzers
  For more information about debugging with logic analyzers
- In-System Updating of Memory and Constants
  For more information about debugging memory
- Design Debugging Using In-System Sources and Probes
  For more information about debugging sources and probes
- Virtual JTAG (sld_virtual_jtag

1.5.1.7. Use the PowerPlay Early Power Estimator (EPE) to estimate power supplies and cooling solution

FPGA power consumption depends on logic design and is challenging to estimate during early board specification and layout. However, it is an important design consideration and must be estimated accurately to develop an appropriate power budget to design the power supplies, voltage regulators, decoupling capacitors, heat sink, and cooling system.

Use the PowerPlay EPE spreadsheet to estimate power, current, and device junction temperature before you have a complete design. The EPE calculates the estimated information based on device information, planned device resources, operating frequency, toggle rates, ambient temperature, heat sinks information, air flow, board thermal model, and other environmental considerations.

- If you have an existing or partially-completed and compiled design—use the **Generate PowerPlay Early Power Estimator File** command in the Quartus Prime software to provide input to the EPE spreadsheet.
- If you do not have an existing design—estimate manually the number of device resources used in your design and input into the EPE spreadsheet. If the device resources information changes during or after the design phase, your power estimation results will be less accurate.

Related Information

- PowerPlay Early Power Estimators (EPE)
- Use power distribution network (PDN) tool to plan for power distribution and decoupling capacitor selection on page 15

1.5.2. Power Pin Connections

The MAX 10 devices require various voltage supplies depending on your design requirements. Use the following checklist to design the board for the FPGA power pin connections.

1.5.2.1. Design the board for power-up

MAX 10 devices support hot socketing (hot plug-in/hot swap) and power sequencing without the use of external devices. Consider the following guidelines:

- During power-up, the output buffers are tri-stated and the internal weak pull-up resistors are disabled by default. You can enable the internal weak pull-up resistors through the Quartus Prime software.
- with weak pull-up resistors enabled until the device is configured and configuration pins drive out.
- Design the voltage power supply ramps to be monotonic—ensure that the minimum current requirement for the power-on-reset (POR) supplies is available during device power up. The following are the POR monitored power supplies:

  V_{CC} or V_{CC_ONE} (after regulated down)

  V_{CCIO} of bank 1B and bank 8

  V_{CCA}
- Set the POR delay in the Quartus Prime software to ensure power supplies are stable. You can extend the POR delay by using an external component to assert the nSTATUS pin low. To ensure the device configures properly and enters user mode, extend the POR delay if the board cannot meet the maximum power ramp time specifications.

- Design power sequencing and voltage regulators for the best device reliability—although power sequencing is not required for correct operation, consider the power-up timing of each rail to prevent problems with long-term device reliability if you are designing a multi-rail powered system.
- Take advantage of the power up sequence for instant-on feature. With instant-on, the device can directly enter user mode with the shortest time after power supplies reach the required level. During power up, the control block reads the POR delay value and instant-on setting bits. If the instant-on is set, the device directly enters the initialization phase. If the instant-on feature is not selected, the POR delay value delays the POR signal. Clear the DSM if you want to change the setting.
- Connect the GND between boards before connecting the power supplies—Intel uses GND as a reference for hot-socketing operations and I/O buffer designs. Connecting the GND between boards before connecting the power supplies prevents the GND on your board from being pulled up inadvertently by a path to power through other components on your board. A pulled up GND could otherwise cause an out-of-specification I/O voltage or current condition with the device.

Related Information

- MAX 10 Power Management User Guide.
- MAX 10 FPGA Device Family Pin Connection Guidelines
- MAX 10 FPGA Configuration User Guide

1.5.2.2. Review the list of required supply voltages and the power supply options

MAX 10 devices offer single and dual supply device options.

Related Information

MAX 10 FPGA Device Datasheet
For a list of the required supply voltages and the recommended operating conditions

1.5.2.3. Ensure I/O power pin compatibility with I/O standards

The output pins of the devices will not conform to the I/O standard specifications if the V_{CCIO} level is out of the recommended operating range of the I/O standard.

Related Information

MAX 10 General Purpose I/O User Guide
For a complete list of the supported I/O standards and VCCIO voltages

1.5.2.4. Ensure correct power pin connections

- Connect all power pins correctly.
- Connect V_{CCIO} pins and V_{REF} pins to support the I/O standards of each bank.
- For unused supplies, consider whether there is a need to ground, open, or retain the power connection.

Related Information

MAX 10 FPGA Device Family Pin Connection Guidelines
For more information on connecting power pins

1.5.2.5. Determine power rail sharing

- 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. It is especially important for you to consider the power supply sharing ability of devices from different device families.
- Follow the suggested power supply sharing and isolation guidance, and the specific guidelines for each pin.

Related Information

- MAX 10 FPGA Device Family Pin Connection Guidelines.
  For more information about power sharing
- AN 583: Designing Power Isolation Filters with Ferrite Beads for Altera FPGAs.
  For more information on isolation guidance

1.5.2.6. Use power distribution network (PDN) tool to plan for power distribution and decoupling capacitor selection

MAX 10 devices include on-die decoupling capacitors to provide high-frequency decoupling.

To plan power distribution and return currents from the voltage regulating module to the FPGA power supplies, you can use the PDN design tool that optimizes the board-level PDN graphically. Although you can use SPICE simulation 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-off.

Related Information

- AN 574: Printed Circuit Board (PCB) Power Delivery Network (PDN) Design Methodology
- Power Delivery Network (PDN) Tool User Guide

1.5.2.7. Review the following guidelines for PLL board design

- Connect all PLL power pins to power supplies to reduce noise even if the design does not use all the PLLs.
- Power supply nets should be provided by an isolated power plane, a power plane cut out, or a thick trace.

Related Information

MAX 10 FPGA Device Family Pin Connection Guidelines
For pin voltage requirements

1.5.3. Configuration Pin Connections

Depending on your configuration scheme, different pull-up or pull-down resistor, signal integrity, and specific pin requirements apply. Connecting the configuration pins correctly is important. Use the following checklist to address common issues.

1.5.3.1. Verify configuration pin connections and pull-up or pull-down resistors are correct for your configuration schemes

Intel provides pin connection guidelines to help you plan your design These guidelines describe each pin and give guidelines for their use.

Related Information

MAX 10 FPGA Device Family Pin Connection Guidelines
For specifics about each configuration pin

1.5.3.2. Design configuration TCK pin using the same technique as in designing high-speed signal or system clock

- Noise on the TCK signal can affect JTAG configuration.
- For a chain of devices, noise on the TCK pin in the chain can cause JTAG programming or configuration to fail for the entire chain.
- For a chain of devices, ensure all devices in the JTAG chain are powered on during JTAG programming or configuration.

1.5.3.3. Verify the JTAG pins are connected to a stable voltage level if not in use

JTAG configuration takes precedence over all configuration methods. If you do not use the JTAG interface, do not leave the JTAG pins floating or toggling during configuration.

To disable the JTAG circuitry, connect TCK pin to GND through a 1-kΩ resistor. Connect TMS and TDI pins to V_{CCIO} through a 1-kΩ resistor. Leave TDO unconnected.

1.5.3.4. Verify the JTAG pin connections to the download cable header

A device operating in JTAG mode uses the required TDI, TDO, TMS, and TCK pins. The TCK pin does not support internal weak pull-down.

Connect the TCK pin to an external 1-kΩ to 10-kΩ pull-down resistor. The TDI and TMS pins have weak internal pull-up resistors. The JTAG output pin (TDO) and all JTAG input pins are powered by V_{CCIO}. The voltage range is 1.5 V to 3.3 V.

The download cable must be powered at 2.5 V when V_{CCIO} of the JTAG pins are powered at 2.5 V to 3.3 V to prevent voltage overshoot because JTAG pins do not have internal PCI clamping diodes. The TCK pin must be pulled to ground. If the V_{CCIO} of the JTAG pins are powered at 1.5 V or 1.8 V, the download cable should be powered by the same V_{CCIO}.

1.5.3.5. Review the following JTAG pin connections guidelines:

- If you have multiple devices 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.
- To disable the JTAG state machine during power-up, pull the TCK pin low through a 1-kΩ resistor to ensure that an unexpected rising edge does not occur on TCK.
- Pull TMS and TDI high through a 1-kΩ to 10-kΩ resistor.

1.5.3.6. Ensure the download cable and JTAG pin voltages are compatible

The download cable interfaces with the JTAG pins of your device. The operating voltage supplied to the Intel FPGA download cable by the target board through the 10-pin header determines the operating voltage level of the download cable. The JTAG pins are powered by V_{CCIO}.

In a JTAG chain containing devices with different V_{CCIO} levels, the devices with a higher V_{CCIO} level should drive the devices with the same or lower V_{CCIO} level. A one-level shifter is required at the end of the chain with this device arrangement. If this arrangement is not possible, you have to add more level shifters into the chain.

Related Information

JTAG Boundary-Scan Testing User Guide
For recommendations about connecting a JTAG chain with multiple voltages across the devices in the chain

1.5.3.7. Buffer the JTAG signal according to the following guidelines:

- If a cable drives three or more devices, buffer the JTAG signal at the cable connector to prevent signal deterioration.
- 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.

Related Information

MAX 10 JTAG Boundary-Scan Testing User Guide
For more information about JTAG pin sharing

1.5.3.8. Ensure all devices in the chain are connected properly

If your device is in a configuration chain, ensure all devices in the chain are connected properly and powered on.

1.5.3.9. Determine if you need to turn on device-wide output enable

The MAX 10 device supports an optional chip-wide output enable that allows you to override all tri-states on the device I/Os. When the DEV_OE pin is driven low, all I/O pins are tri-stated; when this pin is driven high, all pins behave as programmed.

To use the chip-wide output enable feature:

- Turn on **Enable device-wide output enable (DEV_OE)** under the **General** category of the **Device and Pin Options** dialog box in the Quartus Prime software before compiling your design
- Ensure that the DEV_OE pin is driven to a valid logic level on your board
- Do not leave the DEV_OE pin floating

1.5.4. General I/O Pin Connections

Use the following checklist to plan your general I/O pin connections and to improve signal integrity.

1.5.4.1. Specify the state of unused I/O pins

- To reduce power dissipation, set clock pins and other unused I/O pins **As inputs tri-stated**. By default, the Quartus Prime software set the input pins tri-stated with weak pull-up resistor enabled.
- To improve signal integrity, in the **Reserve all unused pins** option under the **Unused Pins** category of the **Device and Pin Options** dialog box of the Quartus Prime software, set the unused pins **As output driving ground**. This setting reduces inductance by creating a shorter return path and reduces noise on the neighboring I/Os. However, do not use this approach if it results in many via paths that causes congestion for signals under the device.
- Carefully check the pin connections in the Pin-Out File (`.pin`) generated by the Quartus Prime software when you compile your design. The `.pin` file specifies how you should connect the device pins. I/O pins specified as GND can be left unconnected or connected to ground for improved noise immunity. Do not connect RESERVED pins.

1.5.4.2. Refer to the Board Design Resource Center

If your design has high-speed signals the board design has a major impact on the signal integrity in the system.

Related Information

- Board Design Resource Center
  For more information about signal integrity and board design
- AN 528: PCB Dielectric Material Selection and Fiber Weave Effect on High-Speed Channel Routing.
  For more information about high-speed board stack-up
- AN 529: Via Optimization Techniques for High-Speed Channel Designs.
  For more information about high-speed designs
- AN 530: Optimizing Impedance Discontinuity Caused by Surface Mount Pads for High-Speed Channel Designs.
  For more information about signal routing layers
- I/O Management, Board Development Support, and Signal Integrity Analysis Resource Center
  For more information on board-level signal integrity information related to the Quartus Prime software

1.5.4.3. Design VREF pins to be noise free

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

Related Information

I/O Features and Pin Connections on page 22
For more information about VREF pins and I/O standards

1.5.4.4. Refer to the Board Design Guideline Solution Center

Noise generated by SSN—when too many pins in close proximity change voltage levels at the same time—can reduce the noise margin and cause incorrect switching. For example, consider these board layout recommendations:

- Break out large bus signals on board layers close to the device to reduce cross talk.
- If possible, route traces orthogonally if two signal layers are next to each other, and use a separation of two to three times the trace width.

Related Information

- Board Design Guidelines Solution Center
  For more board layout recommendations that can help with noise reduction
- I/O Simultaneous Switching Noise on page 28
  For a list of recommendations for I/O and clock connections

1.5.4.5. Verify I/O termination and impedance matching

Voltage-referenced I/O standards require both a VREF and a termination voltage (V_{TT}). The reference voltage of the receiving device tracks the termination voltage of the transmitting device. Consider the following items:

- Each voltage-referenced I/O standard requires a unique termination setup. For example, a proper resistive signal termination scheme is critical in **SSTL-2** standards to produce a reliable DDR memory system with superior noise margin.
- Although single-ended, non-voltage-referenced I/O standards do not require termination, impedance matching is necessary to reduce reflections and improve signal integrity.
- 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.

  The MAX 10 on-chip series termination provides the convenience of no external components. You can also use external pull-up resistors to terminate the voltage-referenced I/O standards such as SSTL and HSTL.

Related Information

MAX 10 General Purpose I/O User Guide
For a complete list of on-chip termination (OCT) support for each I/O standard

1.5.4.6. Perform full board routing simulation using IBIS models

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.

To select the IBIS output in the Quartus Prime software, on the Assignments menu, click **Settings**. Navigate to the **Board-Level** page of the **EDA Tool Settings** category. Under the **Board-level signal integrity analysis** section, in the **Format** option, select **IBIS**.

Related Information

Signal Integrity Analysis with Third-Party Tools

1.5.4.7. Configure board trace models for Quartus Prime advanced timing analysis

For a system to operate properly, signal integrity and board routing propagation delays must be taken into consideration. If you use an FPGA with high-speed interfaces in your board design, analyze the board level timing as part of the I/O and board planning.

Differential I/Os at the top left corner are located in the low speed region.

To generate a more accurate I/O delays and extra reports to gain better insights into the signal behavior at the system level, turn on **Enable Advanced I/O Timing** under the **TimeQuest Timing Analyzer** category in the **Settings** dialog box of your Quartus Prime project. With this option turned on, the TimeQuest Timing Analyzer uses simulation results for the I/O buffer, package, and board trace model to generate the I/O delays.

You can use the advanced timing reports as a guide to make changes to the I/O assignments and board design to improve timing and signal integrity.

Related Information

- MAX 10 Device Pin-Outs.
  For more information about the performance of these I/O pins
- MAX 10 FPGA Device Datasheet.
  For more information about I/O pins

1.5.4.8. Review your pin connections

Intel provides schematic review worksheets based on the device Pin Connection Guidelines and other board-level pin connections literature that you need to consider when you finalize your schematics.

Related Information

MAX 10 Device Schematic Review Worksheet
For more information on finding mistakes in schematics and adhering to design guidelines

1.6. I/O and Clock Planning

1.6.1. Early Pin Planning and I/O Assignment Analysis

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.

1.6.1.1. Verify pin locations early in the FPGA place-and-route software

The FPGA I/O capabilities and board layout guidelines influence pin locations and other types of assignments. Starting FPGA pin planning early improves the confidence in early board layouts, reduces the chance of error, and improves the overall time-to-market.

1.6.1.2. Use the Quartus Prime Pin Planner for I/O pin planning, assignments, and validation

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.

You can use the Quartus Prime Pin Planner for I/O pin assignment planning, assignment, and validation:

- The Quartus Prime **Start I/O Assignment Analysis** command checks that pin locations and assignments are supported in the target FPGA architecture. Checks include reference voltage pin usage, pin location assignments, and mixing of I/O standards.
- You can use I/O Assignment Analysis to validate I/O-related assignments that you make or modify throughout the design process.
- The Create/Import IP core feature of the Pin Planner interfaces with the parameter editor, and enables you to create or import custom IP cores that use I/O interfaces.
- Enter PLL and LVDS blocks. Then, use the **Create Top-Level Design File** command to generate a top-level design netlist file.
- 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.

  After planning is complete, you can pass the preliminary pin location information to PCB designers.

  After the design is complete, you can use the reports and messages generated by the Quartus Prime Fitter for the final sign-off of the pin assignments.

Related Information

- MAX 10 General Purpose I/O User Guide
  For more information about the I/O restrictions guidelines
- Managing Device I/O Pinse
  For more information about I/O assignment and analysis

1.6.1.3. Check I/O restrictions related to ADC usage

The Quartus Prime software uses physics-based rules to define the number of I/O pins allowed in a particular bank based on the I/O's drive strength. These rules are based on noise calculation to analyze accurately the impact of I/O placement on the ADC performance. If you use the ADC block in your design, Intel recommends that you follow the guidelines.

Related Information

MAX 10 General Purpose I/O User Guide
For more information about ADC I/O restriction guidelines

1.6.2. I/O Features and Pin Connections

Use the checklist in this section for guidelines related to I/O features, I/O signal types, I/O standards, I/O banks memory interfaces, pad placements, and special pin connections.

Related Information

- MAX 10 FPGA Device Family Pin Connection Guidelines
- MAX 10 Device Pin-Outs

1.6.2.1. Determine if your system requires single-ended I/O signaling

- Single-ended I/O signaling provides a simple rail-to-rail interface.
- The 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.

Related Information

MAX 10 General Purpose I/O User Guide
For more information about the I/O restrictions guidelines

1.6.2.2. Determine if your system requires voltage-referenced signaling

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

Related Information

MAX 10 General Purpose I/O User Guide
For more information about the VREF restrictions guidelines

1.6.2.3. Determine if your system requires differential signaling

- 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 avoids the requirement for a clean reference voltage. This is possible because of 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.
- Allow the software to assign locations for the negative pin in differential pin pairs. You only need to assign the positive pins.

Related Information

MAX 10 General Purpose I/O User Guide
For more information about the differential I/O restrictions guidelines

1.6.2.4. Select a suitable signaling type and I/O standard for each I/O pin

Ensure that the appropriate I/O standard support is supported in the targeted I/O bank.

Place I/O pins that share voltage levels in the same I/O bank

- Certain I/O banks can support different I/O standards and voltage levels.
- You can assign I/O standards and make other I/O-related settings in the Pin Planner.
- Use the correct dedicated pin inputs for signals such as clocks and global control signals.

Related Information

- MAX 10 General Purpose I/O User Guide.
- MAX 10 High-Speed LVDS SERDES User Guide.

1.6.2.5. Verify that all output signals in each I/O bank are intended to drive out at the bank's assigned VCCIO voltage level

- The board must supply each bank with one V_{CCIO} voltage level for every VCCIO pin in a bank.
- Each I/O bank is powered by the VCCIO pins of that particular bank and is independent of the V_{CCIO} of other I/O banks.
- A single I/O bank supports output signals that are driving at the same voltage as the V_{CCIO}.
- An I/O bank can simultaneously support any number of input signals with different I/O standards.

Related Information

MAX 10 General Purpose I/O User Guide

1.6.2.6. Verify that all voltage-referenced signals in each I/O bank are intended to use the bank's VREF voltage (for devices that support VREF pins)

- To accommodate voltage-referenced I/O standards, each I/O bank supports multiple VREF pins feeding a common VREF bus. Set the VREF pins to the correct voltage for the I/O standards in the bank.
- Each I/O bank can only have a single V_{CCIO} voltage level and a single VREF voltage level at a given time. If the VREF pins are not used as voltage references, the pins cannot be used as generic I/O pins and must be tied to the V_{CCIO} of that same bank or GND.
- An I/O bank, including single-ended or differential standards, can support voltage-referenced standards as long as all voltage-referenced standards use the same VREF setting.
- For performance reasons, voltage-referenced input standards use their own V_{CCIO} level as the power source. You can place voltage-referenced input signals in a bank with a V_{CCIO} of 2.5 V or below.
- Voltage-referenced bidirectional and output signals must drive out at the V_{CCIO} voltage level of the I/O bank.

1.6.2.7. Check the I/O bank support for LVDS features

Different I/O banks include different support for LVDS signaling. Some banks have lower speed performance. Allocate the pins according to your data rate requirement.

Related Information

MAX 10 High-Speed LVDS SERDES User Guide

1.6.2.8. Verify the usage of the VREF pins that are used as regular I/Os

VREF pins have higher pin capacitance that results in a different I/O timing:

- Do not use these pins in a grouped interface such as a bus.
- Do not use these pins for high edge rate signals such as clocks.

1.6.2.9. Test pin connections with boundary-scan test

A boundary-scan test allows you to test pin connections at board level without using physical test probes while the device is operating normally. To perform the boundary-scan test, you must have the boundary-scan description language (BSDL) file of the device.

Use the BSDL files from www.altera.com to perform boundary-scan test on pre-configured MAX 10 devices. For post-configured devices, you must modify the BSDL file according to the design.

Related Information

BSDL Support
For information on performing boundary-scan test after configuration, BSDL generation tool, and guidelines

1.6.2.10. Use the UNIPHY IP core for each memory interface, and follow connection guidelines

The self-calibrating UNIPHY IP core is optimized to take advantage of the MAX 10 structure. The UNIPHY IP core allows you to set external memory interface features and helps set up the physical interface (PHY) best suited for your system. When you use the Intel memory controller IP core functions, the UNIPHY IP core is instantiated automatically.

If you design multiple memory interfaces into the device using Intel FPGA IP, generate a unique interface for each instance to ensure good results instead of designing it once and instantiating it multiple times.

Related Information

Planning Pin and FPGA Resources

1.6.2.11. Use dedicated DQ/DQS pins and DQ groups for memory interfaces

The data strobe DQS and data DQ pin locations are fixed in MAX 10 devices. Before you design your device pin-out, refer to the memory interface guidelines for details and important restrictions related to the connections for these and other memory related signals.

Related Information

- Volume 2: Design Guidelines
  For more information about external memory guidelines
- MAX 10 External Memory Interface User Guide
  For more information about MAX 10 external memory interfaces
- External Memory Solutions Center
  For more information about the memory spec estimator

1.6.2.12. Make dual-purpose pin settings and check for any restrictions when using these pins as regular I/O

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.

For configuration pins that used as general purpose I/Os, take note of the limitations of the pins when operating in user mode.

You can also use dedicated clock inputs, which drive the GCLK networks, as general purpose input pins if they are not used as clock pins. If you use the clock inputs as general inputs, the I/O registers use arithmetic logic module (ALM)-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.

Related Information

- MAX 10 General Purpose I/O User Guide.
- MAX 10 FPGA Configuration User Guide.
- Determine if you need to turn on device-wide output enable on page 17
- Enable the chip-wide reset to clear all registers if required on page 31

1.6.2.13. Review available device I/O features that can help I/O interfaces

Check the available I/O features and consider the following guidelines:

- Programmable current strength—ensure that the output buffer current strength is sufficiently high, but does not cause excessive overshoot or undershoot that violates voltage threshold parameters for the I/O standard. Intel recommends performing an IBIS or SPICE simulations to determine the right current strength setting for your specific application.
- Programmable slew rate—confirm that your interface meets its performance requirements if you use slower slew rates. Intel recommends performing IBIS or SPICE simulations to determine the right slew rate setting for your specific application.
- Programmable input/output element (IOE) delays—helps read and time margins by minimizing the uncertainties between signals in the bus. For delay specifications, refer to the relevant device datasheet.
- Open-drain output—if configured as an open-drain, the logic value of the output is either high-Z or 0. This feature is used in system-level control signals that can be asserted by multiple devices in the system. Typically, an external pull-up resistor is required to provide logic high.
- Bus hold—If the bus-hold feature is enabled, you cannot use the programmable pull-up option. Disable the bus-hold feature if the I/O pin is configured for differential signals. For the specific sustaining current driven through this resistor and the overdrive current used to identify the next driven input and level for each V_{CCIO} voltage, refer to the relevant device datasheet.
- Programmable pull-up resistors—weakly holds the I/O to the V_{CCIO} level when in user mode. This feature can be used with the open-drain output to eliminate the need for an external pull-up resistor. If the programmable pull-up option is enabled, you cannot use the bus-hold feature.
- Programmable pre-emphasis—increases the amplitude of the high frequency component of the output signal, and thus helps to compensate for the frequency-dependent attenuation along the transmission line.

Related Information

MAX 10 General Purpose I/O User Guide

1.6.2.14. Consider OCT features to save board space and verify that the required termination scheme is supported for all pin locations

Driver-impedance matching provides the I/O driver with controlled output impedance that closely matches the impedance of the transmission line to significantly reduce reflections. OCT maintains signal quality, saves board space, and reduces external component costs.

- OCT R_S are supported in the same I/O bank for different I/O standards if they use the same V_{CCIO} supply voltage
- Each I/O in an I/O bank can be independently configured to support OCT R_S or programmable current strength
- You cannot configure both OCT R_S and programmable current strength or slew rate control for the same I/O buffer

Related Information

- MAX 10 General Purpose I/O User Guide
  For more information about the support and implementation of OCT
- MAX 10 High-Speed LVDS SERDES User Guide.
  For more information about high-speed design

1.6.3. Clock Planning

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.
- 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.

1.6.3.1. Use the device PLLs for clock management

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 descriptions 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.
- 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 or from another pin/PLL-fed GCLK.

1.6.3.2. Ensure that you select the correct PLL feedback compensation mode

MAX 10 PLLs support four different clock feedback modes.

Related Information

MAX 10 Clocking and PLL User Guide
For more information about clock feedback modes

1.6.3.3. 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 clock networks.

MAX 10 PLL only allows one clock output per PLL block. If your device have 4 PLLs, there are 4 clock outputs from the PLLs.

1.6.3.4. Use the clock control block for clock selection and power-down

Every GCLK network has its own clock control block. The control block provides the following features that you can use to select different clock input signals or power-down clock networks to reduce power consumption without using any combinational logic in your design:

- Clock source selection (with dynamic selection)
- GCLK multiplexing
- Clock power down (with static or dynamic clock enable or disable)

  In MAX 10 devices, the `clkena` signals are supported at the clock network level instead of at the PLL output counter level. This allows you to gate off the clock even when you are not using a PLL. You can also use the `clkena` signals to control the dedicated external clocks from the PLLs.

Related Information

MAX 10 Clocking and PLL User Guide

1.6.3.5. Instantiate PLL with ADC

You can only use the PLL `C0` counter to connect to the ADC reference clock pin.

1.6.4. I/O Simultaneous Switching Noise

SSN is a concern when too many I/Os (in close proximity) change voltage levels at the same time. Use the checklist in this section for recommendations to plan I/O and clock connections.

1.6.4.1. Consider the following recommendations to mitigate I/O simultaneous switching noise:

- Analyze the design for possible SSN problems.
- Reduce the number of pins that switch the voltage level at exactly the same time whenever possible.
- Use differential I/O standards and lower-voltage standards for high-switching I/Os.
- Use lower drive strengths for high-switching I/Os. The default drive strength setting might be higher than your design requires.
- Reduce the number of simultaneously switching output pins within each bank. Spread output pins across multiple banks if possible.
- Spread the switching I/Os evenly throughout the bank to reduce the number of aggressors in a given area to reduce SSN if bank usage is substantially below 100%.

- Separate simultaneously switching pins from input pins that are susceptible to SSN.
- Place important clock and asynchronous control signals near ground signals and away from large switching buses.
- Avoid using I/O pins one or two pins away from PLL power supply pins for high-switching or high-drive strength pins.
- Use staggered output delays to shift the output signals through time, or use adjustable slew rate settings.

Related Information

- I/O Features and Pin Connections on page 22
- Signal & Power Integrity Design Techniques for SSN Webcast

1.6.4.2. Check on the pin connection guidelines for the ADC pins

The Quartus Prime software uses physics-based rules to define the number of I/O pins allowed in a particular bank based on the I/O's drive strength. These rules are based on noise calculation to analyze accurately the impact of I/O placement on the ADC performance. If you use the ADC block in your design, Intel recommends that you follow the guidelines.

Related Information

MAX 10 FPGA Device Family Pin Connection Guidelines
For more information about the filter type requirement at the ADC power pin, analog input pins, and VREF pin

1.7. Design Entry

In complex FPGA design development, design practices, coding styles, and IP core usage 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.

1.7.1. Use synchronous design practices

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.

1.7.2. Consider the following recommendations to avoid clock signals problems:

- Use dedicated clock pins and clock routing for best results—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.
- 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.
- In multi-clock designs, ensure that signals crossing clock domains are properly synchronized using the synchronizer, a handshake mechanism, or a FIFO.

Related Information

MAX 10 Clocking and PLL User Guide
For more information about clock networks

1.7.3. Use IP cores with the parameter editor

Instead of coding your own logic, save your design time by using Intel FPGA IP cores—a library of parameterized modules and device-specific IP cores. The IP cores are optimized for Intel FPGA device architectures and can offer more efficient logic synthesis and device implementation.

To ensure that you set all ports and parameters correctly, use the Quartus Prime parameter editor to build or change IP cores parameters.

For detailed information about a specific IP core, refer to the respective MAX 10 user guides.

1.7.4. Review the information on dynamic reconfiguration feature

The MAX 10 devices support dynamic reconfiguration—dynamically change the PMA settings or protocols of a channel affecting data transfer on adjacent channels.

1.7.5. Consider the Intel's recommended coding styles to achieve optimal synthesis results

HDL coding styles can have a significant impact on the quality of results for programmable logic designs. For example, when designing memory and digital system processing (DSP) functions, understanding the device architecture helps you to take advantage of the dedicated logic block sizes and configurations.

- You can use the HDL templates provided in the Quartus Prime software as examples for your reference. To access the templates, right click the editing area in the Quartus Prime text editor and click **Insert Template**.
- For additional tool-specific guidelines, refer to the documentation of your synthesis tool.

Related Information

Recommended HDL Coding Styles
For specific HDL coding examples and recommendations

1.7.6. Enable the chip-wide reset to clear all registers if required

MAX 10 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).

- DEV_CLRn pin is driven low—all registers are cleared or reset to 0. The affected register behave as if they are preset to a high value when synthesis performs an optimization called NOT-gate-push back due to register control signals.
- DEV_CLRn pin is driven high—all registers behave as programmed.

  To enable chip-wide reset, before compiling your design, turn on **Enable device-wide reset (DEV_CLRn)** under the **Options** list of the **General** category in the **Device and Pin Options** dialog box of the Quartus Prime software.

1.7.7. Use device architecture-specific register control signals

Each MAX 10 logic array block (LAB) contains dedicated logic for driving register control signals to its ALMs. It is important that the control signals use the dedicated control signals in the device architecture. In some cases, you may be required to limit the number of different control signals in your design.

Related Information

MAX 10 FPGA Device Architecture
For more information about LAB and ALM architecture

1.7.8. Review recommended reset architecture

- 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 deasserted synchronously.
- The source of the reset signal is connected to the asynchronous port of the registers, which can be directly connected to global routing resources.
- The synchronous deassertion allows all state machines and registers to start at the same time.
- Synchronous deassertion avoids an asynchronous reset signal from being released at, or near, the active clock edge of a flipflop that can cause the output of the flipflop to go to a metastable unknown state.

Related Information

www.sunburst-design.com/papers
For more information about good reset design

1.7.9. Review the synthesis options available in your synthesis tool

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

- By default, the Quartus Prime software Integrated Synthesis turns on the **Power-Up Don't Care** logic option that assumes your design does not depend on the power-up state of the device architecture. 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 with asynchronous reset that allows you to power up the design safely with the reset active, regardless of the power-up conditions of the device.
- Some synthesis tools can also read the default or initial values for registered signals in your source code and implement the behavior in the device. For example, the Quartus Prime software Integrated Synthesis converts HDL default and initial values for registered signals into **Power-Up Level** settings. The synthesized behavior matches the power-up conditions 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 (preset signal), synthesis tools typically use the clear signals available on the registers and perform the NOT-gate push back optimization technique. 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.

Related Information

Quartus Prime Integrated Synthesis
For more information about the Power-Up Level settings and the altera_attribute assignment that sets the power-up state

1.7.10. Consider resources available for register power-up and control signals

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.

Related Information

Recommended HDL Coding Styles
For more information about reset logic and power up conditions

1.7.11. Consider Intel's recommendations for creating design partitions

Partitioning a design for an FPGA requires planning to ensure optimal results when the partitions are integrated and ensures that each partition is well placed, relative to other partitions in the device.

Follow Intel's recommendations for creating design partitions to improve the overall quality of results. For example, registering partition I/O boundaries keeps critical timing paths inside one partition that can be optimized independently. Plan your source code so that each design block is defined in a separate file. The software can automatically detect changes to each block separately.

Use hierarchy in your design to provide more flexibility when partitioning. Keep your design logic in the leaves of the hierarchy trees; that is, the top level of the hierarchy should have very little logic, and the lower-level design blocks contain the logic.

Related Information

Best Practices for Incremental Compilation Partitions and Floorplan Assignments
For guidelines to help you create design partitions

1.7.12. Perform timing budgeting and resource balancing between partitions

If your design is created in multiple projects, it is important that the system architect provide guidance to designers of lower-level blocks to ensure that each partition uses the appropriate device resources:

- Because the designs are developed independently, each lower-level designer has no information about the overall design or how their partition connects with other partitions, which can lead to problems during system integration.
- The top-level project information, including pin locations, physical constraints, and timing requirements, should be communicated to the designers of lower-level partitions before they start their design.
- The system architect can plan design partitions at the top level and use the Quartus Prime software **Generate Bottom-Up Design Partition Scripts** option on the Project menu to automate the process of transferring top-level project information to lower-level modules.

1.7.13. Create a design floorplan for incremental compilation partitions

- A design floorplan avoids conflicts between design partitions and ensure that each partition is well-placed relative to other partitions. When you create different location assignments for each partition, no location conflicts occur.
- A design floorplan helps avoid situations in which the Fitter is directed to place or replace a portion of the design in an area of the device where most resources have already been claimed.
- Floorplan assignments are recommended for timing-critical partitions in top-down flows. You can use the Quartus Prime Chip Planner to create a design floorplan using LogicLock region assignments for each design partition.
- With a basic design framework for the top-level design, the floorplan editor enables you to view connections between regions, estimate physical timing delays on the chip, and move regions around the device floorplan.
- After you compiled the full design, you can also view logic placement and locate areas of routing congestion to improve the floorplan assignments.

Related Information

- Best Practices for Incremental Compilation Partitions and Floorplan Assignments
  For more information and guidelines in creating a design floorplan and placement assignments in the floorplan
- Analyzing and Optimizing the Design Floorplan with the Chip Planner.
  For more information about the Floorplan Editor

1.8. Design Implementation

1.8.1. Synthesis and Compilation

1.8.1.1. Specify your synthesis tool and use correct supported version

The Quartus Prime software includes integrated synthesis that fully supports Verilog HDL, VHDL, 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 that you use 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.
- Perform placement and routing in the Quartus Prime software to get accurate timing analysis and logic utilization results.
- Your synthesis tool may 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.

Related Information

- Quartus Prime Integrated Synthesis
- Synopsys Synplify Support
- Mentor Graphics Precision Synthesis Support
- Mentor Graphics LeonardoSpectrum Support
- Quartus Prime Software Release Notes
  For information about the officially supported version of each synthesis tool in a Quartus Prime software version

1.8.1.2. Review resource utilization reports after compilation

After compilation in the Quartus Prime software, review the device resource utilization information:

- Use the 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, use the information to analyze fitting problems.
- 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 usage.
- For more detailed resource information, view 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 number of fully and partially used ALMs, 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 that includes 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.

  Low logic utilization does not mean the lowest possible ALM utilization. A design that is reported to be close to 100% may still have space for extra logic. The Fitter uses ALUTs in different ALMs, even when the logic can be placed within one ALM, so that it can achieve the best timing and routability results. Logic might be spread throughout the device when achieving these results. As the device fills up, the Fitter automatically searches for logic that can be placed together in one ALM.

1.8.1.3. 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. Understand the significance of 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. You can suppress the messages if you have determined that your action is not required.

Related Information

Managing Quartus Prime Projects
For more information about messages and message suppression

1.8.1.4. Consider using incremental compilation

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.

1.8.1.5. Ensure parallel compilation is enabled

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.

1.8.1.6. Use the Compilation Time Advisor

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.

Related Information

Area and Timing Optimization

1.8.2. Timing Optimization and Analysis

Use the guidelines in the following checklist for analyzing your design timing and optimizing your timing performance.

1.8.2.1. Ensure timing constraints are complete and accurate

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.

Related Information

Timing Analysis Overview

1.8.2.2. Review the TimeQuest Timing Analyzer reports after compilation

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. To generate the required timing netlist, specify the tool in the New Project Wizard or the **EDA Tools Settings** page of the **Settings** dialog box.

1.8.2.3. Ensure that the I/O timings are not violated when data is provided to the FPGA

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.

1.8.2.4. Perform Early Timing Estimation before running a full compilation

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 in the Quartus Prime software 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.

1.8.2.5. Consider the following recommendations for timing optimization and analysis assignment:

- Turn on **Optimize multi-corner timing** on the Fitter Settings page in the Settings dialog box.
- Use `create_clock` and `create_generated_clock` to specify the frequencies and relationships for all clocks in your design.
- Use `set_input_delay` and `set_output_delay` to specify the external device or board timing parameters
- 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.
- Use `derive_clock_uncertainty` to automatically apply inter-clock, intra-clock, and I/O interface uncertainties.
- Use `check_timing` to generate a report on any problem with the design or applied constraints, including missing constraints
- Use the Quartus Prime optimization features to achieve timing closure or improve the resource utilization.
- Use the Timing and Area Optimization Advisors to suggest optimization settings.

Related Information

The Quartus Prime TimeQuest Timing Analyzer
For more guidelines about timing constraints

1.8.2.6. Perform functional simulation at the beginning of your design flow

Perform the simulation 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.

1.8.2.7. Perform timing simulation to ensure your design works in targeted device

Timing simulation uses the timing netlist generated by the TimeQuest Timing Analyzer, which includes the delay of different device blocks and placement and routing information. You can perform timing simulation for the top-level design at the end of your design flow to ensure that your design works in the targeted device.

1.8.2.8. Specify your simulation tool and use correct supported version

- Intel provides the ModelSim - Intel FPGA Edition simulator Starter Edition and offers the higher-performance ModelSim - Intel FPGA Edition that 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.
- If you use a third-party simulation tool, use the software version that is supported with your Quartus Prime software version.
- 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.
- Use only the model libraries provided with your Quartus Prime software version. Libraries may change between versions and this can cause a mismatch with your simulation netlist.
- To create a testbench in the Quartus Prime software, on the **Processing** menu, point to **Start** and click **Start Testbench Template Writer**.

Related Information

- Quartus Prime Software Release Notes
  For information about the officially supported version of each simulation tool in a Quartus Prime software version
- Simulating Intel FPGA Designs
- Mentor Graphics ModelSim and QuestaSim Support
- Synopsys VCS and VCS MX Support
- Cadence Incisive Enterprise Simulator Support
- Aldec Active-HDL and Rivera-PRO Support

1.8.3. Formal Verification

Use the following guidelines if your design requires formal verification.

1.8.3.1. Determine if you require formal verification for your design

If formal verification is required for your design, it is easier to plan for limitations and restrictions in the beginning than to make changes later in the design flow.

1.8.3.2. Check for support and design limitations for formal verification

The Quartus Prime software supports some formal verification flows. Using a formal verification flow can impact performance results because it requires that certain logic optimizations be turned off, such as register retiming, and forces hierarchy blocks to be preserved, which can restrict optimization.

Related Information

Cadence Encounter Conformal Support
For more information

1.8.3.3. Specify your formal verification tool and use correct supported version

Specify your formal verification tool in the **EDA Tools Settings** page of the **Settings** dialog box to generate the appropriate output netlist.

Related Information

Quartus Prime Software Release Notes
For information about the officially supported version of each simulation tool in a Quartus Prime software version

1.8.4. Power Analysis and Optimization

After compiling your design, analyze the power consumption and heat dissipation with the Quartus Prime PowerPlay Power Analyzer to calculate the dynamic, static, and I/O thermal power consumption and ensure the design has not violated power supply and thermal budgets.

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

1.8.4.1. Provide accurate typical signal activities to get accurate power analysis result

You need to provide accurate typical signal activities to PowerPlay Power Analyzer:

- Compile a design to derive the information about design resources, placement and routing, and I/O standards.
- Derive signal activity data (toggle rates and static probabilities) 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 Value Change Dump File (`.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.

1.8.4.2. Specify the correct operating conditions for power analysis

Specify the operating conditions, including the core voltage, device power characteristics, ambient and junction temperature, cooling solution, and the board thermal model.

In the Quartus Prime software, select the appropriate settings on the **Operating Settings and Conditions** page in the **Settings** dialog box.

1.8.4.3. Analyze power consumption and heat dissipation in the PowerPlay Power Analyzer

In the Quartus Prime software, on the Processing menu, click **PowerPlay 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 PowerPlay Power Analyzer report is a power estimate and is not a power specification. Always refer to the device datasheet for the power specification.

Related Information

PowerPlay Power Analysis
For more information about power analysis and recommendations for simulation settings for creating signal activity information

1.8.4.4. Review recommended design techniques and Quartus Prime options to optimize power consumption

The Power Optimization Advisor provides specific power optimization advice and recommendations based on the current design project settings and assignments.

Related Information

Power Optimization
For information about design techniques to optimize power consumption

1.8.4.5. Consider using a faster speed grade device

If your design includes many critical timing paths that require the high-performance mode, you might be able to reduce power consumption by using a faster speed grade device if available. With a faster device, the software might be able to set more device tiles to use the low-power mode.

1.8.4.6. 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 blocks 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.

Related Information

MAX 10 Clocking and PLL User Guide
For more information about using clock control blocks

1.8.4.7. Reduce the number of memory clocking events

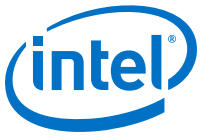
Reduce the number of memory clocking events to reduce memory power consumption. You can use clock gating or the clock enable signals in the memory ports.

1.8.4.8. Consider I/O power guidelines

- The dynamic power consumed in the I/O buffer is proportional to the total load capacitance—lower capacitance reduces power consumption.
- Dynamic power is proportional to the square of the voltage. Use lower voltage I/O standards to reduce dynamic power. Non-terminated I/O standards such as **LVTTL** and **LVCMOS** have a rail-to-rail output swing equal to the V_{CCIO} supply voltage and consume little static power.
- Dynamic power is 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. Because of this, the 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 PowerPlay Power Analyzer calculations. Ensure that you include the external devices power separately in your system power calculations.

1.8.4.9. Reduce design glitches through pipelining and retiming

A design that has many glitches consumes more power because of faster switching activity. Pipelining by inserting flip flops into long combinational paths can reduce design glitches.



However, if there are not many glitches in your design, pipelining may increase power consumption due to the addition of unnecessary registers.

1.8.4.10. Review the information on power-driven compilation and Power Optimization Advisor

The Quartus Prime software offers power-driven compilation to fully optimize device power consumption. Power-driven compilation focuses on reducing your design's total power consumption using power-driven synthesis and power-driven place-and-route.

Related Information

Power Optimization

1.8.4.11. Reduce power consumption with architectural optimization

Use specific device architecture features to reduce power consumption.

For example, use the dedicated DSP block available in the MAX 10 device in place of LEs to perform arithmetic-related functions; build large shift registers from RAM-based FIFO buffers instead of building the shift registers from the LE registers.

1.9. Document Revision History

Table 1. Document Revision History

Date	Version	Changes
May 2017	2017.05.03	• Rebranded as Intel.
December 2014	2014.12.15	• Changed the following terms: Dual image to dual configuration image Dual image configuration to dual configuration • Changed MAX 10 EMIF IP core to UNIPHY IP core.
September 2014	2014.09.22	Initial release.