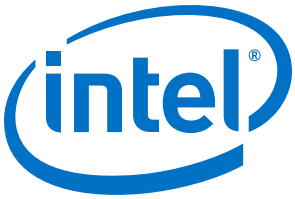


AN 886: Intel® Agilex™ Device Design Guidelines

AN-886 | 2019.09.30
Latest document on the web: PDF | HTML

Contents

1. Introduction to the Intel® Agilex™ Device Design Guidelines

This document provides a set of design guidelines, recommendations, and a list of factors to consider for designs that use Intel® Agilex™ FPGAs. It is important to follow Intel recommendations throughout the design process for high-density, high-performance Intel Agilex designs. This document also assists you with planning the FPGA and system early in the design process, which is crucial to successfully meet design requirements.

Note: This document does not include all Intel Agilex device details and features. For more information about Intel Agilex devices and features, refer to the respective Intel Agilex User Guides.

The material references the Intel Agilex device architecture as well as aspects of the Intel Quartus® Prime software and third-party tools that you might use in your design. The guidelines presented in this document can improve productivity and avoid common design pitfalls.

This document does not include all the Intel Agilex FPGA and Hard Processor System (HPS) device details, features or information on designing the hardware or software system.

For more information about the Intel Agilex FPGA, refer to the Intel Agilex Support page.

Note: Intel recommends that you use Intel Quartus Prime Pro Edition and the Intel SoC FPGA Embedded Development Suite Pro to develop Intel Agilex SoC designs. Hardware developed with Intel Quartus Prime Pro Edition only supports software developed with the Intel SoC FPGA Embedded Development Suite Pro.

Related Information

Intel Agilex FPGAs and SoCs Support
For Intel Agilex documentation and support

1.1. Design Flow

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

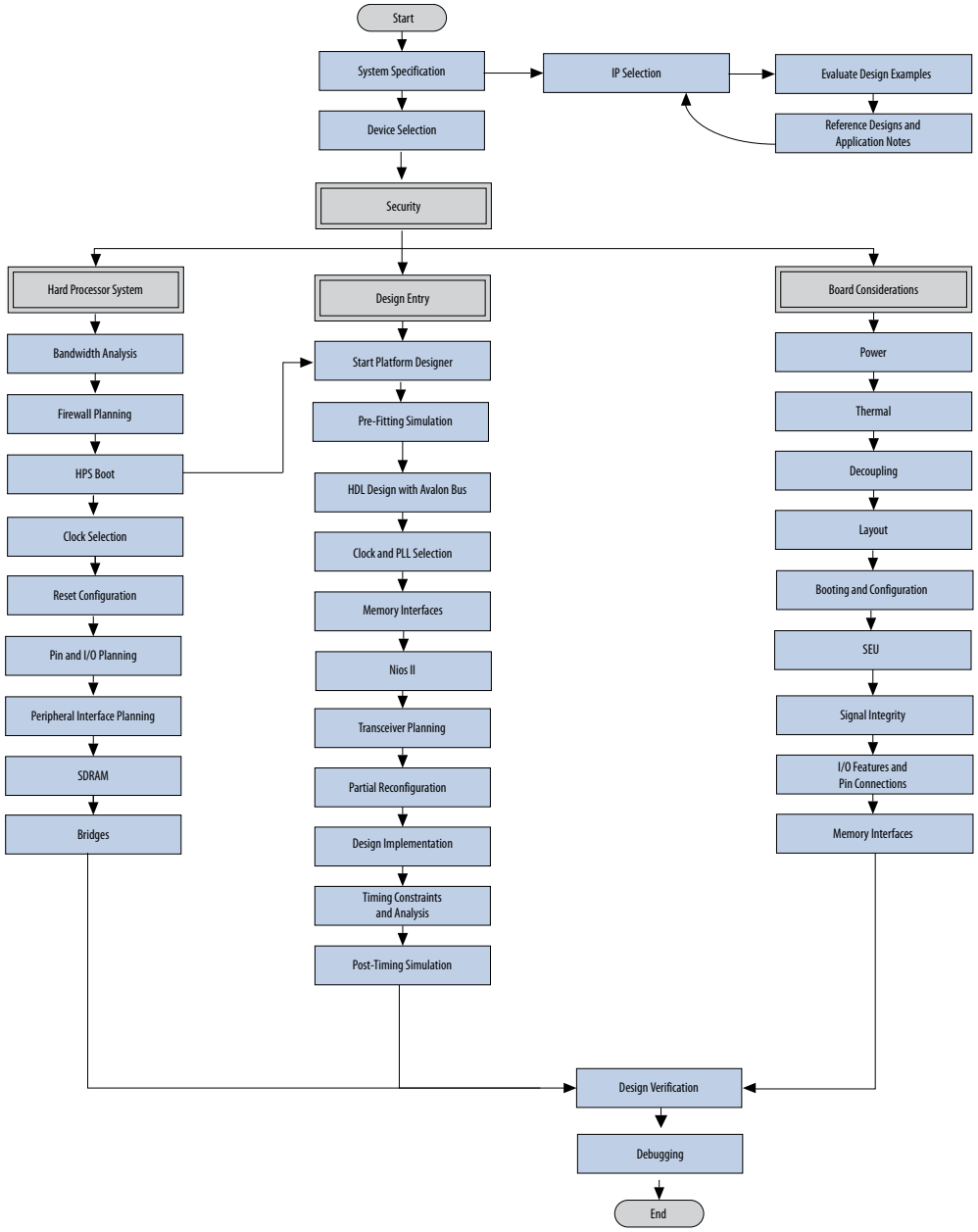
Stages of the Design Flow	Description
System Specification	Planning, design specifications, IP selection
Device Selection	Device information, determining device variant and density, package offerings, migration, speed grade
Security Considerations	Authentication, encryption, base security, firewalls and fuses

continued...

Stages of the Design Flow	Description
Hard Processor System	Bandwidth analysis, firewall planning, HPS boot methods, reset and I/O planning, peripheral, bridge and SDRAM configuration
Design Entry	Coding styles and design recommendations, Platform Designer, planning for hierarchical or team-based design
Board considerations	Early power estimation, thermal management option, board design guidelines, configuration scheme, boot mode, signal integrity, I/O and clock planning, pin connections, reset plan, memory interfaces, verification
Design verification	System console, simulation, debug timing analysis
Debugging	Debug tools, remote debugging, simulation, system console, JTAG
Embedded software design guidelines	Software requirements and architecture, tools, driver considerations, application development, test and validate

The flow diagram depicted below represents the general high level design flow when you design with an Intel Agilex FPGA device. Certain points in the design flow such as IP selection may be iterative; and others, such as security considerations may be encountered at multiple points in your design.

Figure 1. Intel Agilex Device Design Flow



1.2. Introduction to the Intel Agilex Device Design Guidelines Revision History

Table 2. Introduction to the Intel Agilex Device Design Guidelines Revision History

Document Version	Changes
2019.09.30	Initial release

2. System Specification

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

2.1. Design Specifications

Table 3. Design Specifications Checklist

Number	Done?	Checklist Item
1		Create detailed design specifications and a test plan if appropriate.
2		Plan 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
- Create a functional verification/test plan
- Consider a common design directory structure
- Consider the use of an Revision Control System (RCS) for checking in and out files so development time is easier

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 manufacture ability. 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 Information

Intel Agilex Device Family Pin Connection Guidelines

2.2. Install Intel Quartus Prime Software

Before proceeding with IP selection and simulation, install the Intel Quartus Prime Pro Edition software.

For more information, refer to the following documentation:

Table 4. Intel Quartus Prime Related Documentation

Related Documentation
Intel Quartus Prime Software Suite For information about the different Intel Quartus Prime editions
Intel Quartus Prime Design Software - Support Center For information about the Intel Quartus Prime software features
Download Center for FPGAs For information about Intel Quartus Prime software versions and device families to select
Intel FPGA Licensing Support Center For information about license types, getting a license file, setting up a license file, and resolving license-related issues
Intel FPGA Software Installation and Licensing For information about installing Intel Quartus Prime Pro Edition

2.3. IP Selection

Table 5. IP Selection Checklist

Number	Done?	Checklist Item
1		Evaluate available HPS IP.
2		Evaluate soft IP and I/O interfaces.
3		Ensure that your board design supports JTAG connections.

2.3.1. Evaluate Available HPS IP

The HPS architecture integrates a wide set of peripherals that reduce board size and increase performance within a system. Before evaluating soft IP for the FPGA core, identify which HPS peripherals can be leveraged to save FPGA I/O:

- EMACs
- USB Controllers
- I^2C Conctrollers
- UARTs
- SPI Master Controllers
- SPI Slave Controllers
- GPIO Interfaces

For more information about evaluating the available HPS IP, refer to the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

2.3.2. Select Soft IP and I/O Interfaces

Intel offers a wide variety of soft IP you can instantiate in your design. Please refer to the Intel FPGA IP Portfolio web page for more information.

A broad portfolio of soft IP also work with the Nios II processor that can be integrated into your design. For all IP that work with the Nios II processor refer to the *Embedded Peripheral IP User Guide*.

Related Information

Embedded Peripheral IP User Guide

2.3.2.1. IP Cores

Table 6. IP Cores Checklist

Number	Done?	Checklist Item
1		Configure and evaluate IP cores by using the IP Parameter Editor that is part of Intel Quartus Prime Pro Edition.

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 Intel 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.

For more information, refer to *Introduction to Intel FPGA IP Cores*.

Related Information

Introduction to Intel FPGA IP Cores

2.4. Simulation

If you decide to use RTL and gate-level design simulation, refer to the following documentation:

- *Intel Quartus Prime Pro Edition User Guide: Third Party Simulation*
- *Simulation Quick-Start for ModelSim - Intel FPGA Edition: Intel Quartus Prime Pro Edition*
- *Embedded Peripherals IP User Guide*

Related Information

- Simulation Quick-Start for ModelSim - Intel FPGA Edition: Intel Quartus Prime Pro Edition
- Intel Quartus Prime Pro Edition User Guide: Third Party Simulation
- Embedded Peripherals IP User Guide

2.5. Preparing for Design Entry

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

2.5.1. Coding Styles and Design Recommendations

Table 7. 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.

Table 8. Design Recommendations Checklist

Number	Done?	Checklist Item
1		Use synchronous design practices. Pay attention to clock and reset signals.

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.

For information about Hardware Description Language (HDL) coding style recommendations, refer to the *Intel Quartus Prime Pro Edition User Guide: Design Recommendations.*

Related Information

- Intel Quartus Prime Pro Edition User Guide: Design Recommendations
- Intel Stratix® 10 High-Performance Design Handbook

2.5.2. Platform Designer

Table 9. Platform Designer Checklist

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

Platform Designer is a system integration tool included as part of the Intel Quartus Prime software. Platform Designer 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. Platform Designer facilitates design reuse by packaging and integrating your custom components with Intel and third-party IP components. Platform Designer 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.

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

For more information about Platform Designer, refer to *Intel Quartus Prime Pro Edition User Guide Platform Designer*.

Related Information

Intel Quartus Prime Pro Edition User Guide: Platform Designer

2.6. I/O Summary

One of the most important considerations when configuring the device is to understand how the I/O is organized in the Intel Agilex SoC devices.

1. HPS EMIF I/O

There are three modular I/O sub-bank that can connect to SDRAM memory. One of the I/O banks is used to connect the address, command and ECC data signals. The other two banks are for connecting the data signals.

2. HPS Dedicated I/O

These 48 I/O are physically located inside the HPS, are dedicated for the HPS, and are used for the HPS clock and peripherals, including mass storage flash memory.

Note: HPS EMIF I/O and HPS Dedicated I/O are only located on an HPS device.

3. Secure Device Manager (SDM) Dedicated I/O

The SDM has 24 dedicated I/Os, which include JTAG, clock, reset, configuration, reference voltages, boot and configuration flash interfaces, and MSEL.

Note: SDM Dedicated I/O can be found on both FPGA and HPS devices.

4. General Purpose I/O

You can use general purpose I/O for FPGA logic, FPGA external memory interfaces and high-speed serial interfaces. It is possible to export most HPS peripheral interfaces to the FPGA fabric for custom adaptation and routing to FPGA I/O.

Note: GPIO can be found on both FPGA and HPS devices.

The table below summarizes the characteristics of each I/O type.

Table 10. Summary of I/O Types

	Dedicated HPS I/O	HPS EMIF I/O	Dedicated SDM I/O	General Purpose I/O
Number of Available I/O	48	Up to 3 I/O 48 sub-banks (using 2 I/O96 banks)	24	All other device I/O
Location	Inside the HPS Only available for devices with HPS.	Only available for devices with HPS. • Bottom sub-bank in bank 3C • Top and Bottom sub-bank in Bank 3D	Inside the SDM	I/O Columns are in the FPGA device
Voltages Supported	1.8V	1.5V True Differential Signal support of DDR4 protocols	1.8V	1.2V I/O, 1.5V I/O, and high speed serial transceivers
Purpose	HPS Clock, HPS peripherals, mass storage flash, HPS JTAG	HPS main memory	FPGA JTAG through SDM dedicated pins, clock, reset, configuration, reference voltages, boot and configuration flash interfaces	General purpose and transceiver I/O
Timing Constraints	Fixed	Provided by memory controller IP	Fixed	User defined
Recommended Peripherals	HPS peripheral I/O such as Ethernet PHY, USB PHY, mass storage flash (NAND, SD/MMC), TRACE debug.	DDR4	Boot and configuration source, FPGA JTAG through SDM dedicated pins, MSEL signals, and AVSTx8 are connected to the SDM.	Slow speed HPS peripherals (I^2C, SPI, EMAC-MII), FPGA I/O such as FPGA EMIFs, transceiver I/O, AVSTx16, AVSTx32, and other parallel and control/status I/O.

2.7. Using Intel Agilex HPS in your Device

Take note of HPS components you must consider when planning your system design.

Table 11. Considerations for using Intel Agilex SoC in your Device Checklist

Number	Done?	Checklist Item
1		HPS_OSC_CLK (**mandatory**)— there are other ways to clock the HPS, but this is the most common and easiest way.
2		HPS_COLD_nRESET (**optional**)—if you want external reset control over the HPS, this is the easiest way to achieve it.

continued...

Number	Done?	Checklist Item
3		HPS_EMIF (**mandatory**)—the HPS is designed to run software out of a large DDR style memory. Not provisioning the HPS_EMIF makes the software environment constrained and in most cases unusable.
4		HPS_UART (**mandatory**)—pin one of these out on the HPS dedicated pins so you can see early boot telemetry from your software.
5		HPS_JTAG (**mandatory**)—this is mandatory for board bring up and debugging early boot flow issues. This can be serially chained with the SDM JTAG TAP or broken out on HPS dedicated pins.
6		HPS_EMAC (**optional**)—if you can allocate one of these you can provide luxury debug and maintenance support for the software environment.
7		HPS flash memory (**optional**)—since the HPS is loaded by the SDM and the HPS can subsequently access the SDM flash, this may not be mandatory. Many software environments require some sort of persistent storage.

2.8. System Specification Revision History

Table 12. System Specification Revision History

Document Version	Changes
2019.09.30	Initial release

3. Device Selection

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

3.1. Device Variant

Table 13. Device Variant Checklist

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

The Intel Agilex device family consists of several device variants, like F-Series and I-Series optimized to meet different application requirements.

- Intel Agilex F-Series FPGAs and SoCs are optimized for a wide range of FPGA applications that require optimal balance of power and performance, with the power efficiency of Intel's industry-leading 10-nm FinFET process technology.
- Intel Agilex I-Series FPGAs and SoCs offer high-performance processor interfaces and transceiver rates for bandwidth-intensive applications.

For more information, refer to the *Intel Agilex FPGA Advanced Information Brief (Device Overview)*.

Related Information

Intel Agilex FPGA Advanced Information Brief (Device Overview)

3.2. PLLs and Clock Routing

Table 14. PLLs and Clock Routing Checklist

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

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

Related Information

- Intel Agilex FPGA Advanced Information Brief (Device Overview)
- Intel Agilex Clocking and PLL User Guide

3.3. Logic, Memory, and Multiplier Density

Table 15. Logic, Memory, and Multiplier Density Checklist

Number	Done?	Checklist Item
1		Estimate the required logic, memory, and multiplier density. For more information, refer to the *Device Variant* section.
2		Reserve device resources for future development and debugging.

Intel Agilex devices offer a range of densities that provide different amounts of device logic resources, including memory, multipliers, and adaptive logic module (ALM) logic cells. 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. Intel Agilex devices support vertical migration, which provides flexibility.

You may observe 10% – 15% increase in resource utilization when you migrate your design from existing devices to Intel Agilex devices. 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 Intel 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.

For more information about determining resource utilization for a compiled design, refer to the *Device Resource Utilization Reports* section.

Related Information

- Device Resource Utilization Reports on page 89
- Device Variant on page 15

3.4. I/O Pin Count, LVDS SERDES Channels, and Package Offering

Table 16. 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 SERDES channels are enough.
4		Evaluate fabric speed grade and the transceiver speed grade.
5		Consider I/O voltages required for chip to chip interfaces and ensure that they are compatible with supported standards.

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 unidirectional LVDS SERDES channels for differential signaling; ensure that your device density-package combination includes enough LVDS SERDES 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 Intel Quartus Prime software to determine how many I/O pins are used. Also consider reserving I/O pins for debugging.

3.5. Speed Grade

Table 17. 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.

3.6. Vertical Device Migration

Table 18. Vertical Device Migration Checklist

Number	Done?	Checklist Item
1		Consider vertical device migration availability and requirements.
2		Refer to *Intel Agilex FPGA External Memory Interface Overview* and the *External Memory Interfaces IP - Support Center* web page for information about the external memory interface (EMIF) pin pairing.

Intel Agilex 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 Intel Agilex 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 Intel Quartus Prime software at the beginning of your design cycle or as soon as the device migration selection is possible in the Intel 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 devices. 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 Intel Quartus Prime Pin Planner highlights pins that change function in the migration device when compared to the currently selected device.

Related Information

- External Memory Interface Spec Estimator
- Intel Agilex FPGA External Memory Interface Overview
- External Memory Interfaces IP - Support Center

3.7. Device Selection Revision History

Table 19. Device Selection Revision History

Document Version	Changes
2019.09.30	Initial release

4. Security Considerations

Table 20. Security Considerations Checklist

Number	Done?	Checklist Item
1		Consider whether your design requires device security features to be enabled. If so, plan to provide power to the VCCFUSEWR_SDM rail for authentication fuse management.
2		Consider whether your design requires bitstream encryption, and whether the encryption keys are stored in Battery-Backed RAM (BBRAM). If so, plan to provide power to the VCCBAT pin using a battery on the board.
3		Consider licensing terms that best suit your requirements for the available device variants .

Intel Agilex devices provide flexible and robust security features to protect sensitive data, intellectual property, and the device itself under both remote and physical attacks. Intel Agilex devices provide two main categories of security features:

- Authentication—Authentication ensures that the device firmware and optionally the configuration bitstream are from a trusted source. Authentication is fundamental to Intel Agilex security in that any other Intel Agilex security features cannot be enabled without first enabling owner authentication. Device firmware authentication is always performed. Additionally, integrity verification of device firmware and bitstream is always performed in order to prevent an Intel Agilex device from loading a bitstream with unexpected changes, such as from corruption or malicious attack.
- Encryption—Encryption protects confidential information in the owner configuration bitstream and reduces the threat of intellectual property theft.

When designing a system with an Intel Agilex device that utilizes the device security features, you must consider provisions for authentication key storage, permissions, and cancellation. You may also need to consider encryption key storage and management. The hash of the owner root public key is always stored in eFuses on an Intel Agilex device, and both Intel firmware key cancellation and owner authentication key cancellation are managed through eFuses as well. Therefore, it is important to provide appropriate power to the VCCFUSEWR_SDM pin. For more information about powering on VCCFUSEWR_SDM, refer to *Intel Agilex Pin Connection Guidelines*.

If bitstream encryption is enabled on the Intel Agilex device, you need to store the encryption key on the device. The encryption key may be stored in Battery-Backed RAM (BBRAM) or eFuses. Storing the encryption key in eFuses is permanent, while storing the encryption key in BBRAM allows for key wipe or reprovisioning. If the design requires encryption key storage in BBRAM, a non-volatile battery must be connected to the VCCBAT pin. For more information about connecting a battery to the VCCBAT pin, refer to the *Intel Agilex Pin Connection Guidelines*.

Related Information

Intel Agilex Device Family Pin Connection Guidelines

4.1. Security Considerations Revision History

Table 21. Security Considerations Revision History

Document Version	Changes
2019.09.30	Initial release

5. Design Entry

This section contains the content for both FPGA-only and SoC design entry.

5.1. Design Entry for SoC Devices

5.1.1. Firewall Planning

You can use the system interconnect firewalls to enforce security policies for slave and memory regions in the system interconnect. For more information about firewalls, refer to the *System Interconnect* section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.2. Boot And Configuration Considerations

The Intel Agilex SoC HPS does not have a boot ROM. Instead the SDM has a BootROM which loads the initial FPGA configuration bitstream. This bitstream also contains the HPS First Stage Bootloader (FSBL) binary.

5.1.2.1. Selecting HPS Boot Options

The Intel Agilex SoC device supports two boot and configuration modes. When designing your system, you must choose one of the following boot modes for your application: HPS First and FPGA First:

- **FPGA Configuration First**: The SDM configures the FPGA core and all the periphery I/O before loading the FSBL into the HPS on-chip RAM and releasing the HPS from reset. If any errors exist during initial configuration, the HPS is not released from reset.
- **HPS First**: The SDM only configures the I/O required for the HPS EMIF, and then loads the FSBL into the HPS on-chip RAM before releasing the HPS from reset. The FPGA core, as well as the other unused I/O, remain unconfigured. The HPS configures the rest of the FPGA.

Note: Faster HPS boot times are possible using HPS First boot mode.

Select a configuration and boot mode from **Assignments ➤ Device ➤ Device and Pin Options ➤ HPS/FPGA** configuration order tab in Intel Quartus Prime Pro Edition.

HPS First and FPGA First Boot Considerations

Guideline: Engineering Sample Device Restrictions

Engineering Sample (ES) devices only support the HPS First boot mode with dual flash. Any boards built with ES devices must be designed to support HPS First boot mode. It is possible to design your board for both HPS First and FPGA First boot with either dual or single flash, if you plan to use a different boot scheme in production.

Guideline: HPS First Boot Mode Utilizes Early I/O Release

Follow the guidelines in this document to properly design your board and the SoC device pin out for the HPS EMIF interface for Early I/O Release.

For more information about the supported boot modes, refer to the *Intel Agilex SoC Boot User Guide* and the "Boot and Configuration" section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.2.2. Configuration Sources

The initial FPGA configuration and the HPS FSBL are part of the initial configuration bitstream, which can be obtained from several sources:

- Avalon®-ST Data Source: An external Avalon-ST master provides the bitstream.
- JTAG Interface: An external JTAG master (usually driven by a host tool) provides the bitstream.
- SDM Flash: A flash device connected on SDM side provides the bitstream.

5.1.2.3. Remote System Update (RSU)

Intel Agilex SoC supports the RSU feature. When you use this feature, you have the option to store multiple production images alongside a failsafe factory image on the external SDM flash. Upon exiting POR, SDM attempts to load the production images in your specific sequence. If all the production images fail to load, then the failsafe factory image is loaded.

Related Information

Intel Agilex Configuration User Guide

5.1.3. HPS Clocking and Reset Design Considerations

The main clock and reset sources for the HPS are:

- HPS_OSC_CLK device I/O pin—The external clock source for the HPS PLLs, which generate clocks for the MPU Subsystem, CCU, SMMU, L3 Interconnect, HPS peripherals and HPS-to-FPGA user clocks.
- nCONFIG device I/O pin—nCONFIG is a dedicated input pin to the SDM that holds off initial configuration and initiates FPGA reconfiguration. An nCONFIG assertion cold resets the HPS.
- HPS_COLD_nRESET device I/O pin—An optional reset input that cold resets only the HPS and is configured for bidirectional operation.

GUIDELINE: You can configure the HPS_COLD_nRESET pin to be on any open SDM I/O pin.

From Intel Quartus Prime,

1. Click **Assignments ➤ Device**.
2. Click the "Device and Pin Options" button.
3. Go to the "Configuration" tab.
4. Click the "Configuration Pin Options" button.
5. Click the "USE_HPS_COLD_nRESET" check box and select available SDM_IO pin.

For more information, refer to the “Pin Features and Connection for HPS Clocks, Reset and POR.” section.

Related Information

Intel Agilex Device Family Pin Connection Guidelines

5.1.3.1. HPS Clock Planning

HPS clock planning involves choosing clock sources and defining frequencies of operation for the following HPS components:

- HPS PLLs
- MPU Subsystem
- L3 Interconnect
- HPS Peripherals
- HPS-to-FPGA user clocks

HPS clock planning depends on board-level clock planning, clock planning for the FPGA portion of the device, and HPS peripheral external interface planning. Therefore, it is important to validate your HPS clock configuration before finalizing your board design.

GUIDELINE: Verify the MPU and peripheral clocks using Platform Designer.

Use Platform Designer to initially define your HPS component configuration. Set the HPS input clocks, peripheral source clocks and frequencies. Note any Platform Designer warning or error messages and address them by modifying clock settings or verifying that a warning does not adversely affect your application.

5.1.3.2. Early Pin Planning and I/O Assignment Analysis

The HPS clock input resides in the HPS Dedicated I/O Bank shared with I/O from HPS peripherals such as Ethernet, mass storage flash, and UART console. It's location within this bank is user configurable.

GUIDELINE: Choose an I/O voltage level for the HPS Dedicated I/O.

The HPS Dedicated I/Os are LVCMOS/LVTTL supporting a 1.8V voltage level. Make sure any HPS peripheral interfaces (for example: Ethernet PHY, UART console) configured to use the HPS Dedicated I/O bank as well as board-level clock circuitry for the HPS are compatible with 1.8V LVCMOS signaling.

5.1.3.3. Pin Features and Connections for HPS Clocks, Reset and PoR

The HPS clock pin and optional reset pin have certain functional behaviors and requirements that you should consider when planning for and designing your board-level reset logic and circuitry.

GUIDELINE: Choose a pin location for the HPS clock input.

The HPS_OSC_CLK can be located anywhere within the HPS Dedicated I/O Bank. Use the HPS Platform Designer component to select the pin for HPS_OSC_CLK and verify its compatibility with other HPS peripheral I/O locations assigned to this bank.

GUIDELINE: Observe the minimum assertion time specifications of nCONFIG and HPS_COLD_nRESET.

The nCONFIG and HPS_COLD_nRESET pins must be asserted for the minimum time specified in the HPS section of the *Intel Agilex Device Family Pin Connection Guidelines*.

GUIDELINE: Do not connect HPS_COLD_nRESET to SDM QSPI reset.

HPS_COLD_nRESET is a bi-directional pin that is input to the SDM to initiate a cold reset procedure to the HPS and its peripherals. The HPS_COLD_nRESET output can be used to reset any other devices on the board that should be reset when the HPS is reset. However, the SDM handles reset for the QSPI through software. Connecting HPS_COLD_nRESET to the SDM QSPI reset can result in undefined system behavior.

Related Information

Intel Agilex Device Family Pin Connection Guidelines

5.1.3.4. Direct to Factory Pin Support for Remote System Update (RSU) Feature

Intel Agilex SoC supports the RSU feature. RSU implements device reconfiguration using dedicated RSU circuitry available in all Intel Agilex devices. When you use this feature, you have the option to store multiple production images alongside a failsafe factory image on the external SDM flash. Upon exiting POR, SDM attempts to load the production images in your specific sequence. If all the production images fail to load, then the failsafe factory image is loaded.

GUIDELINE: Use the Direct to Factory Image pin to instruct the SDM to load either Factory or Application Image when exiting POR.

The Direct to Factory Image is an optional pin that can be used with the RSU feature.[1] If this pin is asserted during POR, the SDM does not attempt to load any production image; instead, the SDM loads the factory image directly from the external SDM flash.

For more information about using the HPS together with the RSU feature, refer to the *Intel Agilex Configuration User Guide*.

Related Information

Intel Agilex Configuration User Guide

[1] Both factory and application images are stored in the SDM flash.

5.1.3.5. Internal Clocks

Once you have validated the HPS clock configuration as described in the HPS Clock Configuration Planning guidelines, you must implement your HPS clock settings under software control, which is typically done by the boot loader software. You must also follow guidelines for transferring reference clocks between the HPS and FPGA.

GUIDELINE: Avoid cascading PLLs between the HPS and FPGA.

Cascading PLLs between the FPGA and HPS has not been characterized. Unless you perform a jitter analysis, do not chain the FPGA and HPS PLLs together. Output clocks from HPS are not intended to be fed into PLLs in the FPGA.

There are specific requirements for managing HPS PLLs and clocks under software control.

For more information, refer to the "Clock Manager" section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.3.6. HPS Peripheral Reset Management

HPS peripherals and subsystems have specific reset sequencing requirements. The boot loader software provided in SoC EDS implements the reset management sequence according to the requirements in the Reset Manager section.

GUIDELINE: Use the latest boot loader software in SoC EDS to manage HPS reset.

For more information about the required software flow, refer to the "Reset Manager" section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.4. Reset Configuration

5.1.4.1. HPS Peripheral Reset Management

HPS peripherals and subsystems have specific reset sequencing requirements. The boot loader software provided in SoC EDS implements the reset management sequence according to the requirements in the Reset Manager section.

GUIDELINE: Use the latest boot loader software in SoC EDS to manage HPS reset.

For more information about the required software flow, refer to the "Reset Manager" section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.4.2. System Reset Considerations

Table 22. System Reset Checklist

Number	Done?	Checklist Item
1		Intel strongly recommends using the Reset Release IP in your design to provide a known initialized state for your logic to begin operation. The Reset Release IP is described in the *Intel Agilex Configuration User Guide*.

After any one of the four Watchdog timers expire and generates a system reset request to the SDM, the SDM then performs one of three types of system resets:

- HPS Cold reset
- HPS Warm reset
- HPS Cold and trigger remote update

Note: One of these three options can be chosen from within the Intel Quartus Prime Pro Edition tool.

In the Intel Quartus Prime Pro Edition tool, you must select the "HPS Clocks and resets" tab, then the "Resets" tab, then click on the "Enable watchdog reset" check box, and then choose one of three choices from the pull-down menu for the "How SDM handles HPS watchdog reset" label:

- **HPS Cold reset**
  - **Impact on HPS**—The SDM holds the processor in reset. The SDM loads the FSBL from the same bitstream that was loaded into the device prior to the cold reset into the HPS on-chip memory. When successfully completed, the SDM releases the HPS reset causing the processor to start executing code from the reset exception address.
  - **Impact on FPGA**—The FPGA core fabric is untouched during the reset. After exiting reset, software determines whether to reconfigure the FPGA portion.
- **HPS Warm reset**
  - **Impact on HPS**—The SDM holds the processor in reset. The FSBL remains in the on-chip RAM during a warm reset. The SDM takes the processor out of reset, and the processor runs the FSBL in on-chip RAM.
  - **Impact on FPGA**—The FPGA portion is left alone during the reset. After exiting reset, software determines whether to reconfigure the FPGA portion.
- **HPS Cold reset and trigger a remote Update**
  - **Impact on HPS**—The SDM holds the processor in reset. The SDM loads the FSBL from the next valid *.pof image or factory image into the HPS on-chip memory. The *.pof contains the data to configure the FPGA portion of the SoC and the FSBL payload. When successfully completed, the SDM releases the HPS from reset and the processor begins executing code from the reset exception address.
  - **Impact on FPGA**—The FPGA portion is first erased, then reconfigured with the next valid Core RBF or Factory Core RBF. There must always be a valid factory RBF present.

Related Information

Intel Agilex Configuration User Guide

5.1.5. HPS Pin Multiplexing Design Considerations

There is a total of 48 dedicated HPS I/O pins. The HPS component in Platform Designer offers pin multiplexing settings as well as the option to route most of the peripherals into the FPGA fabric.

GUIDELINE: Route the USB, EMAC and Flash interfaces to the HPS Dedicated I/O first, starting with USB.

Intel recommends that you start by routing high speed interfaces such as USB, Ethernet, and flash to the Dedicated I/O first.

5.1.6. HPS I/O Settings: Constraints and Drive Strengths

GUIDELINE: Ensure that you have correctly configured the I/O settings for the HPS Dedicated I/O.

The HPS pin location assignments are managed automatically when you generate the Platform Designer system containing the HPS. Likewise, timing and I/O constraints for the HPS EMIF interface are managed by the Intel Agilex External Memory Interfaces for HPS IP. You must manage the following I/O constraints for the HPS Dedicated I/O using the Intel Quartus Prime software in the same way as for FPGA I/O: drive strength, weak pull-up enables, and termination settings. Any peripherals configured to use FPGA I/O must also be fully constrained, including pin locations, using the Intel Quartus Prime software.

5.1.7. Design Guidelines for HPS Interfaces

This section outlines the design guidelines for HPS Interfaces such as EMAC, USB, SD/MMC, NAND, UART and I^2C. For more detailed information about the peripherals in the HPS, please refer to the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.7.1. Design Considerations for Selecting PHY Interfaces

The following PHY interfaces can be selected when configuring your system:

- HPS EMAC PHY Interfaces
  - Reduced Media Independent Interface (RMII)
  - Reduced Gigabit Media Independent Interface (RGMII)
- PHY Interfaces connected through FPGA I/O
  - GMII/MII
    - RMII—Using the MII-to-RMII Adapter
    - Serial Gigabit Media Independent Interface (SGMII)—Using the GMII-to-SGMII Adapter
  - Intel Management Data Input/Output (MDIO)

5.1.7.1.1. HPS EMAC PHY Interfaces

There are three EMACs based on the Synopsys* DesignWare* 3504-0 Universal 10/100/1000 Ethernet MAC IP version. When configuring an HPS component for EMAC peripherals within Platform Designer, you must select from one of the following supported PHY interfaces, located in the HPS Dedicated I/O Bank(2), for each EMAC instance:

- Reduced Media Independent Interface (RMII)
- Reduced Gigabit Media Independent Interface (RGMII)

GUIDELINE: When selecting a PHY device, consider the desired Ethernet rate, available I/O and available transceivers, PHY devices that offer the skew control feature, and device driver availability.

It is possible to adapt the MII/GMII PHY interfaces exposed to the FPGA fabric by the HPS component to other PHY interface standards such as RMII, SGMII, SMII and TBI using soft adaptation logic in the FPGA and features in the general-purpose FPGA I/O and transceiver FPGA I/O.

For more information, refer to the device drivers available for your operating system of choice or the Linux device driver provided with the Intel Agilex Transceiver-SoC Development Kit.

The EMAC provides a variety of PHY interfaces and control options through the HPS and the FPGA I/Os.

Note: You can connect PHYs to the HPS EMACs through the FPGA fabric using the GMII and MII bus interfaces for Gigabit and 10/100 Mbps access respectively. You can refer to the *Intel Stratix® 10 SoC SGMII Reference Design* on RocketBoards.org to learn how to implement this type of design. For more information about Embedded Peripheral IPs offered, please refer to the *Embedded Peripheral IP User Guide*.

Determine Ethernet Rate

For information about allowable Ethernet rates, refer to the following documents:

- *Intel Agilex Hard Processor System Technical Reference Manual*
- *Intel Agilex FPGA Data Sheet*

Related Information

- Intel Agilex Device Data Sheet
- Intel Agilex Hard Processor System Technical Reference Manual
- Embedded Peripheral IP User Guide
- Intel Stratix 10 SoC SGMII Reference Design

5.1.7.1.2. RMII and RGMII PHY Interfaces

Determine whether to use RMII or RGMII PHY Interfaces.

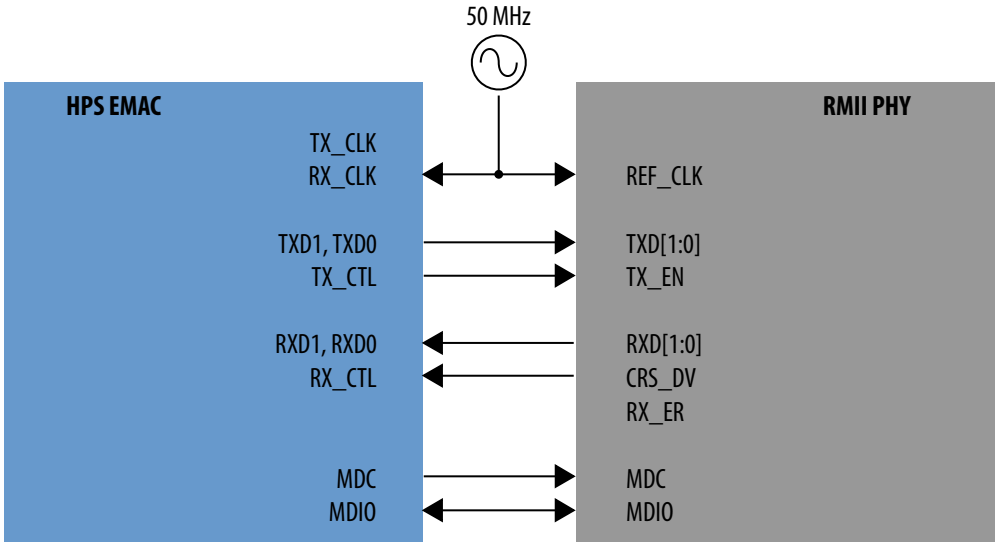
(2) The HPS Dedicated I/O Bank consists of 48 I/O with 1.8V signaling.

RMII

RMII uses a single centralized system-synchronous 50 MHz clock source (REF_CLK) for both transmit and receive paths across all ports. This simplifies system clocking and lowers pin counts in high port density systems, because your design can use a single board oscillator as opposed to per port TX_CLK/RX_CLK source synchronous clock pairs.

RMII uses two-bit wide transmit and receive data paths. All data and control signals are synchronous to the REF_CLK rising edge. The RX_ER control signal is not used. In 10Mbps mode, all data and control signals are held valid for 10 REF_CLK clock cycles.

Figure 2. RMII MAC/PHY Interface



Interface Clocking Scheme

EMACs and RMII PHYs can provide the 50 MHz REF_CLK source. Using clock resources already present such as HPS_OSC_CLK input, internal PLLs further simplifies system clocking design and eliminates the need for an additional clock source.

This section discusses system design scenarios for both HPS EMAC-sourced and PHY-sourced REF_CLK.

GUIDELINE: Consult the *Intel Agilex FPGA Data Sheet* for specifics on the choice of REF_CLK source in your application.

Note: Make sure your choice of PHY supports the REF_CLK clocking scheme in your application. **Note** any requirements and usage considerations specified in the *Intel Agilex FPGA Data Sheet*.

You can use one of the following two methods for sourcing REF_CLK:

- HPS-Sourced REF_CLK
- PHY-Sourced REF_CLK

Figure 3. HPS Sourced REF_CLK

In this scheme, connect the EMAC's HPS RMII I/O TX_CLK output to both the HPS RMII I/O RX_CLK and PHY REF_CLK inputs.

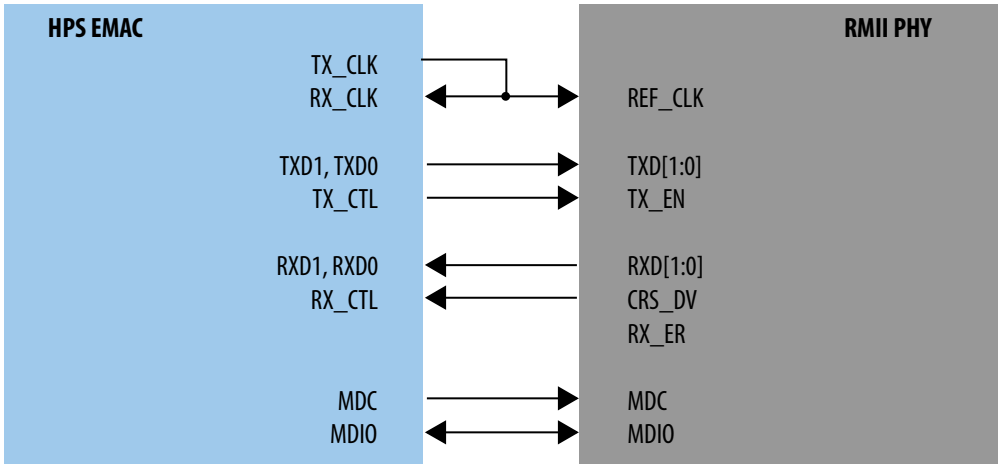
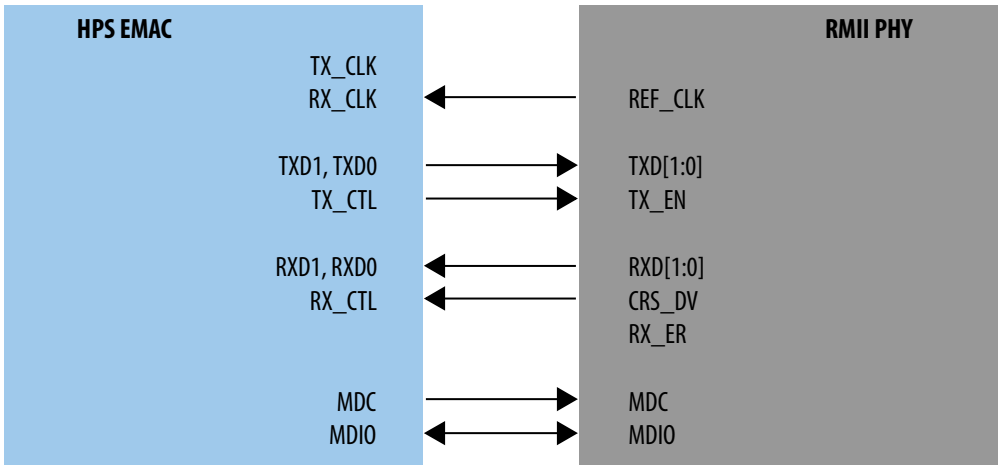


Figure 4. PHY Sourced REF_CLK

In this scheme, connect the PHY's REF_CLK output to the EMAC's HPS RMII I/O RX_CLK input. Leave the EMAC's HPS RMII I/O TX_CLK output unconnected. PHYs capable of sourcing REF_CLK are typically configured to do so through pin bootstrapping and require an external crystal or clock input to generate REF_CLK.



If RX_CLK is routed daisy-chain from source to MAC to PHY or source to PHY, you must account for the flight time difference as both REF_CLK loads observes the clock at different times.

GUIDELINE: Take into account routing delays and skews on the data and control signals to ensure meeting setup and hold as specified in the HPS SoC Device data sheet and PHY data sheet.

Signal length matching is not necessary unless you have signal lengths in excess of 24 inches, in which case you must perform some basic timing analysis with clock delays versus data delays.

The period is 20 ns with the 50 MHz REF_CLK and remains at this frequency regardless of whether the PHY is set to 10Mbps or 100Mbps mode.

All clocking in the HPS EMAC is based on the RX_CLK, so the Tco and PCB flight time of REF_CLK from either the EMAC or PHY can be ignored. Typical board traces up to 12 inches yield only 2 ns of flight time and Tsu of RXD to RX_CLK is 4 ns minimum, well under the 20 ns period.

There is a 2 ns hold requirement of RXD versus RX_CLK which is easily satisfied as well because the Tco of TXD with respect to RX_CLK for either the MAC or the PHY is typically over 2 ns. For the Intel Agilex SoC device, the Tco of TXD with respect to RX_CLK is 2 ns to 10 ns.

GUIDELINE: Ensure the REF_CLK source meets the duty cycle requirement.

There is no jitter specification for the REF_CLK, but there is a duty cycle requirement of 35 to 65 percent. This requirement is met by the Intel Agilex SoC device PLLs and clock outputs for GPIO or for the TX_CLK signal coming from the HPS IP specifically.

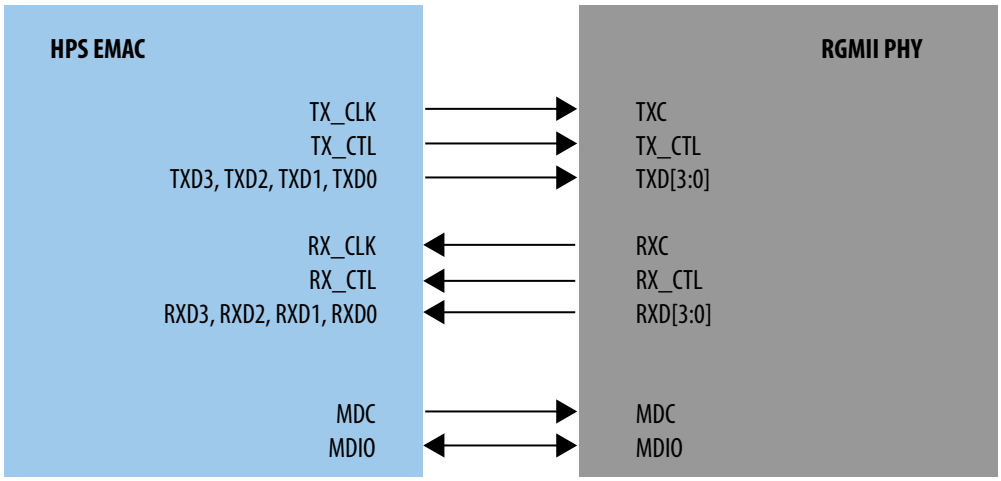
RGMII

RGMII is the most common interface because it supports 10 Mbps, 100 Mbps, and 1000 Mbps connection speeds at the PHY layer.

RGMII uses four-bit wide transmit and receive data paths, each with its own source-synchronous clock. All transmit data and control signals are source synchronous to TX_CLK, and all receive data and control signals are source synchronous to RX_CLK.

For all speed modes, TX_CLK is sourced by the MAC, and RX_CLK is sourced by the PHY. In 1000 Mbps mode, TX_CLK and RX_CLK are 125 MHz, and Dual Data Rate (DDR) signaling is used. In 10 Mbps and 100 Mbps modes, TX_CLK and RX_CLK are 2.5 MHz and 25 MHz, respectively, and rising edge Single Data Rate (SDR) signaling is used.

Figure 5. RGMII MAC/PHY Interface



I/O Pin Timing

This section addresses RGMII interface timing from the perspective of meeting requirements in the 1000 Mbps mode. The interface timing margins are most demanding in 1000 Mbps mode, thus it is the only scenario you consider here.

At 125 MHz, the period is 8 ns, but because both edges are used, the effective period is only 4 ns. The TX and RX busses are separate and source synchronous, simplifying timing. The RGMII specification calls for CLK to be delayed from DATA at the receiver in either direction by a minimum 1.0 ns and a maximum 2.6 ns.

In other words, the TX_CLK must be delayed from the MAC output to the PHY input and the RX_CLK from the PHY output to the MAC input. The signals are transmitted source synchronously within the +/- 500 ps RGMII skew specification in each direction as measured at the output pins. The minimum delay needed in each direction is 1 ns but Intel recommends to target a delay of 1.5 ns to 2.0 ns to ensure significant timing margin.

Transmit path setup/hold

Only setup and hold for TX_CLK to TX_CTL and TXD[3:0] matter for transmit. The Intel Agilex I/O can provide up to 2.25 ns additional delay on outputs in 150 ps increments. For specific values, refer to the *Intel Agilex FPGA Data Sheet*. This delay is enabled using the output delay logic option within the assignment editor in Intel Quartus Prime.

GUIDELINE: For TX_CLK from the Intel Agilex device, you must introduce 1.8 ns I/O delay to meet the 1.0 ns PHY minimum input setup/hold time in the RGMII spec.

The Intel Agilex SoC HPS dedicated I/O and FPGA I/O support adding up to 2.25[(3)] ns of output delay in 150 ps increments. For specific values, refer to the *Intel Agilex FPGA Data Sheet*. The delay added to the MAC's TX_CLK output when using HPS dedicated I/O can be configured in the HPS Platform Designer IP component.

Receive path setup/hold

Only setup and hold for RX_CLK to RX_CTL and RXD[3:0] are necessary to consider for receive timings. The Intel Agilex I/O can provide up to 2.25 ns[(3)] additional delay on inputs. For specific values, refer to the *Intel Agilex FPGA Data Sheet*. For Intel Agilex device inputs, the up to 2.25 ns[(3)] I/O delay can achieve this timing for RX_CLK without any other considerations on the PHY side or board trace delay side.

GUIDELINE: If the PHY does not support RGMII-ID, use the configurable delay elements in the Intel Agilex SoC HPS dedicated I/O or FPGA I/O to center the RX_CLK in the center of the RX_DATA/CTL data valid window.

If using HPS I/O, configure delay on the RX_CLK in the HPS Platform Designer IP component. If using FPGA I/O, add delay on the RX_CLK input with an input delay setting in the project settings file (.qsf).

GUIDELINE: Since the TSE MAC IP with 1000BASE-X PCS option no longer provides an option for the transceiver I/O, to implement an SGMII PHY interface using the FPGA transceiver I/O for an Intel Agilex HPS EMAC

(3) This value is pending characterization.

instance, you must select "NONE" for the PCS I/O option, which gives you a TBI interface. The transceiver PHY IP must be separately instanced and connected in the Intel Agilex device.

Related Information

Intel Agilex Device Data Sheet

5.1.7.1.3. PHY Interfaces Connected Through FPGA I/O

Using FPGA I/O for an HPS EMAC PHY interface can be helpful when there are not enough left to accommodate the PHY interface or when you want to adapt to a PHY interface not natively supported by the HPS EMAC.

GUIDELINE: Specify the PHY interface transmit clock frequency when configuring the HPS component in Platform Designer.

For either GMII or MII, including adapting to other PHY interfaces, specify the maximum transmit path clock frequency for the HPS EMAC PHY interface: 125 MHz for GMII, 25 MHz for MII. This configuration results in the proper clock timing constraints being applied to the PHY interface transmit clock upon Platform Designer system generation.

GMII/MII

GMII and MII are only available in the Intel Agilex device by driving the EMAC signals into the FPGA core routing logic, then ultimately to FPGA I/O pins or to internal registers in the FPGA core.

GUIDELINE: Apply timing constraints and verify timing with Timing Analyzer.

Because routing delays can vary widely in the FPGA core and I/O structures, it is important to read the timing reports, and especially for GMII, create timing constraints. GMII has a 125 MHz clock and is single data rate unlike RGMII. GMII does not have the same considerations for CLK-to-DATA skew though; its signals are automatically centered by design by being launched with the negative edge and captured with the rising edge.

GUIDELINE: Register interface I/O at the FPGA I/O boundary.

With core and I/O delays easily exceeding 8 ns, Intel recommends to register these buses in each direction in I/O Element (IOE) registers, so they remain aligned as they travel across the core FPGA logic fabric. On the transmit data and control, maintain the `clock-to-data/control` relationship by latching these signals on the falling edge of the `emac[0,1,2]_gtx_clk` output from the HPS EMAC. Latch the receive data and control at the FPGA I/O inputs on the rising edge of the RX_CLK sourced by the PHY.

GUIDELINE: Consider transmit timing in MII mode.

MII is 25 MHz when the PHY is in 100 Mbps mode and 2.5 MHz when the PHY is in 10 Mbps mode, so the shortest clock period is 40 ns. The PHY sources the clock for both transmit and receive directions. Because the transmit timing is relative to the TX_CLK clock provided by the PHY, the turnaround time may be of concern, but this is usually not an issue due to the long 40 ns clock period.

Since the reference clock is transmitted through the FPGA, then out for the data – the round-trip delay must be less than 25 ns as there is a 15 ns input setup time. Note that the transmit data and control are launched into the FPGA fabric by the HPS EMAC transmit path logic on the negative edge of the PHY-sourced TX_CLK, which removes 20 ns of the 40 ns clock-to-setup timing budget.

With the round-trip clock path delay on the data arrival timing incurring PHY-to-SoC board propagation delay plus the internal path delay from the SoC pin to and through the HPS EMAC transmit clock mux taking away from the remaining 20 ns setup timing budget, it may be necessary to retime the transmit data and control to the rising edge of the `phy_txclk_o` clock output registers in the FPGA fabric for MII mode transmit data and control.

Adapting to RGMII

The Intel Agilex SoC device does not support adapting the HPS EMAC signals to RGMII using FPGA I/O pins.

Adapting to RMII

It is possible to adapt the MII HPS EMAC PHY signals to an RMII PHY interface at the FPGA I/O pins using logic in the FPGA.

GUIDELINE: Provide a 50 MHz REF_CLK source.

An RMII PHY uses a single 50 MHz reference clock (REF_CLK) for both transmit and receive data and control. Provide the 50 MHz REF_CLK either with a board-level clock source, a generated clock from the FPGA fabric, or from a PHY capable of generating the REF_CLK.

GUIDELINE: Adapt the transmit and receive data and control paths.

The HPS EMAC PHY interface exposed in the FPGA fabric is MII, which requires separate transmit and receive clock inputs of 2.5 MHz and 25 MHz for 10 Mbps and 100 Mbps modes of operation, respectively. Both transmit and receive datapaths are 4-bits wide. The RMII PHY uses the 50 MHz REF_CLK for both its transmit and receive datapaths and at both 10 Mbps and 100 Mbps modes of operation. The RMII transmit and receive datapaths are 2-bits wide. At 10 Mbps, transmit and receive data and control are held stable for 10 clock cycles of the 50 MHz REF_CLK. You must provide adaptation logic in the FPGA fabric to adapt between the HPS EMAC MII and external RMII PHY interfaces: four bits at 25MHz and 2.5 MHz, to and from two bits at 50 MHz, and 10x oversampled in 10 Mbps mode.

GUIDELINE: Provide a glitch-free clock source on the HPS EMAC MII `tx_clk_in` clock input.

The HPS component's MII interface requires a 2.5/25 MHz transmit clock on its `emac[0,1,2]_tx_clk_in` input port. The switch between 2.5 MHz and 25 MHz must be done glitch free as required by the HPS EMAC. An FPGA PLL can be used to provide the 2.5 MHz and 25 MHz transmit clock along with an ALTCLKCTRL IP block to select between counter outputs glitch-free.

Adapting to SGMII

You can use the GMII-to-SGMII Adapter core to adapt the GMII HPS EMAC PHY signals to a Serial Gigabit Media Independent Interface (SGMII) PHY interface at the FPGA transceiver I/O pins using logic in the FPGA and the multi gigabit transceiver I/O. While it is possible to design custom logic for this adaptation, this section describes using Platform Designer adapter IP.

GUIDELINE: Use the GMII to SGMII Adapter IP available in Platform Designer.

Configure the HPS component in Platform Designer for an EMAC "To FPGA" I/O instance and choose GMII as the PHY interface type along with a management interface. Do not export the resulting HPS component GMII signals in Platform Designer. Instead, add the Intel GMII to SGMII Adapter IP to the Platform Designer subsystem and connect to the HPS component's GMII signals. The GMII to SGMII Adapter IP makes use of the Intel HPS EMAC Interface Splitter IP in Platform Designer to split out the "emac" conduit from the HPS component for use by the GMII to SGMII Adapter. The adapter IP instantiates the Intel Triple Speed Ethernet (TSE) MAC IP, configured in 1000BASE-X/SGMII PCS PHY-only mode (that is, no soft MAC component). For more information about how to use the Intel GMII to SGMII Adapter IP, refer to the *Embedded Peripherals User Guide*.

Related Information

Embedded Peripheral IP User Guide

5.1.7.1.4. Consider Device Driver Availability

Refer to the device drivers available for your operating system of choice or the Linux* device driver provided with the Intel Agilex SoC development kit.

For more information, refer to the following documentation:

Table 23. Device Driver Related Documentation

Related Documentation
Golden System Reference Design (GSRD) User Manuals
Linux Drivers web page on RocketBoards.org
Embedded Software Developer Center
Linux Developer Center
Intel SoC FPGA Embedded Development Suite (SoC EDS) User Guide

5.1.7.1.5. MDIO

The Intel Management Data Input/Output (MDIO) PHY management bus has two signals per MAC: MDC and MDIO. MDC is the clock output, which is not free running. At 2.5 MHz, it has a 400 ns minimum period. MDIO is a bidirectional data signal with a High-Z bus turnaround period.

When the MAC writes to the PHY, the data is launched on the falling edge, meaning there is 200 ns -10 ns = 190 ns for flight time, signal settling, and setup at the receiver. Because data is not switched until the following negative edge, there is also 200 ns of hold time. These requirements are very easy to meet with almost any board topology. When the MAC reads from the PHY, the PHY is responsible to output the read

data from 0 to 300 ns back to the MAC, leaving 100 ns less 10 ns setup time, or 90 ns for flight time, signal settling, and setup at the receiver. This requirement is also very easy to meet.

GUIDELINE: Board pull-ups on MDC/MDIO.

Both signals require an external pull-up resistor. Consult your PHY's datasheet for the correct pull-up resistor value. 1K Ohm is a typical resistor value.

GUIDELINE: Ensure interface timing that MDIO requires.

MDIO requires a 10 ns setup and hold time for data with respect to MDC. For specific values, refer to the *Intel Agilex FPGA Data Sheet*.

Related Information

Intel Agilex Device Data Sheet

5.1.7.1.6. Signal Integrity

GUIDELINE: Make use of the SoC device's On-Chip Termination (OCT).

Intel Agilex devices can tune their outputs to many settings, with 50 ohm output impedance often being the best value. Intel Quartus Prime automatically uses series OCT without calibration on RGMII outputs. Check the Intel Quartus Prime fitter report to verify the OCT settings on the interface's outputs.

GUIDELINE: Use appropriate board-level termination on PHY outputs.

Only a few PHYs offer I/O tuning for their outputs, so Intel recommends that you verify the signal path to the Intel Agilex device with a simulator. Place a series resistor on each signal near the PHY output pins to reduce the reflections if necessary.

GUIDELINE: Ensure reflections at PHY TX_CLK and EMAC RX_CLK inputs are minimized to prevent double-clocking.

Be cognizant if the connection is routed as a "T" as signal integrity must be maintained such that no double-edges are seen at REF_CLK loads. Ensure reflections at REF_CLK loads are minimized to prevent double-clocking.

GUIDELINE: Use a Signal Integrity (SI) simulation tool.

It is simple to run SI simulations on these unidirectional signals. These signals are almost always point-to-point, so simply determining an appropriate series resistor to place on each signal is usually sufficient. Many times, this resistor is not necessary, but the device drive strength and trace lengths as well as topology should be studied when making this determination.

5.1.7.2. USB Interface Design Guidelines

The Intel Agilex HPS can connect its embedded USB MACs directly to industry-standard USB 2.0 ULPI PHYs using the 1.8 V dedicated HPS I/O. No FPGA routing resources are used and timing is fixed, which simplifies design.

For more information about the design considerations for USB, refer to the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.7.3. SD/MMC and eMMC Card Interface Design Guidelines

The Secure Digital/Multimedia Card (SD/MMC) controller, based on the Synopsys DesignWare attached to the hard processor system (HPS) is used for mass storage. This module supports:

- SD version 3.01, in addition to 3.0
- Embedded MMC (eMMC) version 4.51 and 5.0, in addition to 4.5(4)

GUIDELINE: Ensure that voltage translation transceivers are properly implemented if using 1.8V SD card operation.

HPS I/O use a fixed voltage level of 1.8 V. Many SD cards have an option to signal at 1.8 V or 3.3 V, although the initial power-up voltage requirement is 3.3 V. In cases when you want to use a 3.3 V SD card, voltage switching is required. To have the correct voltage level to power the card, voltage translation transceivers are required.

Follow the guidelines in the Voltage Switching section of the *SD/MMC Controller* chapter in the *Intel Agilex Hard Processor System Technical Reference Manual*

Table 24. Level Shifter Requirements

HPS I/O Bank Voltage	SD Card Voltage	Level Shifter Needed
1.8 V	3.0 V	Yes
1.8 V	1.8 V	No

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.7.4. Design Guidelines for Flash Interfaces

GUIDELINE: Connecting the QSPI flash to the SoC device.

The HPS does not have a QSPI flash controller. The HPS has access to the QSPI controller in the SDM.

For an example of Flash Memory implementation, refer to the *Intel Agilex F-Series Transceiver-SoC Development Kit Schematics*.

GUIDELINE: In the Intel Quartus Prime Pro Edition GUI, select the configuration clock speed to match the capabilities of the QSPI flash device that you selected.

For more information about considerations when connecting QSPI flash to the SDM QSPI interface, refer to the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

(4) The HS400 mode is not supported.

5.1.7.4.1. NAND Flash Interface Design Guidelines

GUIDELINE: Ensure that the selected NAND flash device is an 8- or 16-bit ONFI 1.0 compliant device.

The NAND flash controller in the HPS requires:

- The external flash device is 8- or 16-bit ONFI 1.0 compliant
- Supports x16 for mass storage usage
- Single-level cell (SLC) or multi-level cell (MLC)
- Only one ce# and rb# pin pair is available for the boot source. Up to three additional pairs are available for mass storage
- Page size: 512 bytes, 2 KB, 4 KB or 8 KB
- Pages per block: 32, 64, 128, 256, 384 or 512
- Error correction code (ECC) sector size can be programmed to 512 bytes (for 4, 8 or 16 bit correction) or 1024 bytes (24-bit correction)

For more information about the NAND Flash Controller, refer to the *NAND Flash Controller* section in the *Intel Agilex Hard Processor System Technical Reference Manual*,

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.7.5. UART Interface Design Guidelines

HPS boot firmware outputs console status messages throughout the boot process to the HPS UART port. If you want to view boot firmware console output, consider the following guidelines to assign the HPS UART peripheral to device I/O that are available at HPS boot time.

GUIDELINE: For the HPS First boot and configuration scheme, assign the HPS UART peripheral to the HPS Dedicated I/O Bank.

The SDM configures and releases to user-mode (Early I/O Release flow) the HPS Dedicated I/O and HPS EMIF I/O before booting the HPS. The remaining FPGA I/O and fabric are not available until the rest of the FPGA is configured at a later point in the boot flow.

GUIDELINE: For the FPGA First boot and configuration scheme, you can assign the HPS UART to either HPS Dedicated or FPGA I/O.

The SDM configures the entire FPGA portion, including the entire I/O ring before booting the HPS.

GUIDELINE: Properly connect flow control signals when routing the UART signals through the FPGA fabric.

When routing UART signals through the FPGA, the flow control signals are available. If flow control is not being used, connect the signals in the FPGA as shown in the following table:

Table 25. UART Interface Design

Signal	Direction	Connection
CTS	input	low
DSR	input	high
DCD	input	high
RI	input	high
DTR	output	No-Connection
RTS	output	No-Connection
OUT1_N	output	No-Connection
OUT2_N	output	No-Connection

For more information, refer to the "UART Controller" section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.7.6. I²C Interface Design Guidelines

GUIDELINE: Instantiate the open-drain buffer when routing I²C signals through the FPGA fabric.

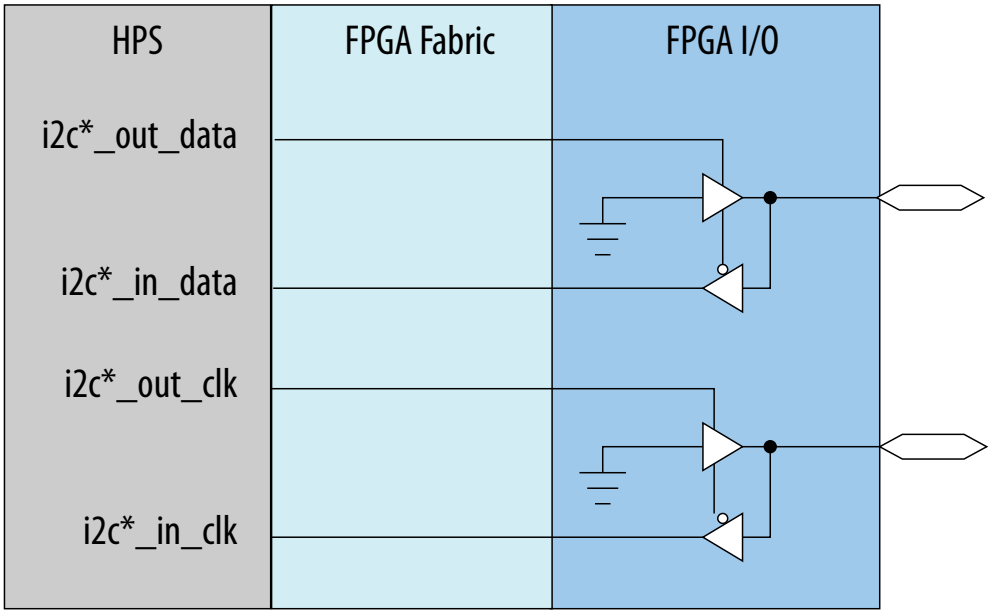
When routing I²C signals through the FPGA, note that the I²C pins from the HPS to the FPGA fabric (`i2c*_out_data`, `i2c*_out_clk`) are not open-drain and are logic level inverted. Thus, to drive a logic level zero onto the I²C bus, drive the corresponding pin high. This implementation is useful as they can be used to tie to an output enable of a tri-state buffer directly. You must use the `altiobuff` to implement the open-drain buffer.

Intel recommends that you use I/O Buffer (ALTIOBUF) IP core when you expose I²C to FPGA fabric.

GUIDELINE: Ensure that the pull-ups are added to the external SDA and SCL signals in the board design.

Because the I²C signals are open drain, pull-ups are required to make sure that the bus is pulled high when no device on the bus is pulling it low.

Figure 6. I²C Wiring to FPGA pins



GUIDELINE: Ensure that the high and low clock counts are configured correctly for the speed of the I²C interface

There is an I²C internal clock located in the:

- SDM—125 MHz
- HPS—100 MHz

The default settings for the high and low clock counts are configured for 125 MHz, so the default high and low clocks for the HPS I²C are longer than expected.

5.1.8. Interfacing between the FPGA and HPS

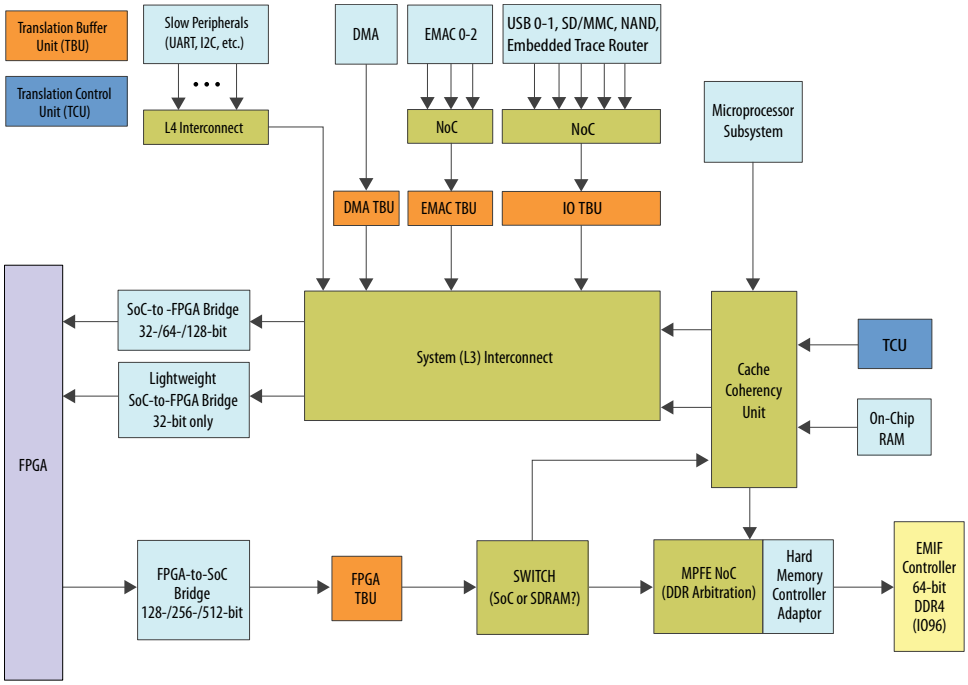
The memory-mapped connectivity between the HPS and the FPGA fabric is a crucial tool to maximize the performance of your design. Use the guidelines in this chapter for recommended topologies to optimize your system's performance.

5.1.8.1. Overview of HPS Memory-Mapped Interfaces

The HPS exposes two memory-mapped SoC-to-FPGA interfaces:

- SoC-to-FPGA bridge: 32-, 64-, or 128-bit wide Advanced Microcontroller Bus Architecture (AMBA*) Advanced eXtensible Interface (AXI*)-4
- Lightweight SoC-to-FPGA bridge: 32-bit wide AXI-4
- FPGA-to-SoC bridge: 128-, 256-, 512-bit wide ACE*-Lite

Figure 7. Intel Agilex HPS Connectivity



Timing Closure Considerations

The bridges exposed to the FPGA are synchronous; and clock crossing is performed within the interface itself. As a result, you must only ensure that both the FPGA-facing logic and your design close timing in Timing Analyzer. Interrupts are considered asynchronous by the HPS, and as a result the HPS logic resynchronizes them to the internal HPS clock domain so there is no need to close timing for them.

5.1.8.1.1. SoC-to-FPGA Bridge

GUIDELINE: Use the SoC-to-FPGA bridge to connect memory hosted by the FPGA to the HPS.

The SoC-to-FPGA bridge allows masters in the HPS such as the microprocessor unit (MPU), DMA, or peripherals with integrated masters to access memory hosted by the FPGA portion of the SoC device. This bridge supports 32-, 64-, and 128-bit data paths allowing the width to be tuned to the largest slave data width in the FPGA fabric connected to the bridge. This bridge is intended to be used by masters performing bursting transfers and should not be used for accessing peripheral registers in the FPGA fabric. Control and status register accesses should be sent to the lightweight SoC-to-FPGA bridge instead.

GUIDELINE: If memory connected to the SoC-to-FPGA bridge is used for HPS boot, ensure that the FPGA portion of the SoC device is configured first.

The SoC-to-FPGA bridge is accessed if the MPU boots from the FPGA. Before the MPU boots from the FPGA, the FPGA portion of the SoC device must be configured, and the SoC-to-FPGA bridge must be remapped into addressable space. Otherwise, access to the SoC-to-FPGA bridge during the boot process results in a bus error. To satisfy these

requirements, use the FPGA First boot and configuration scheme. The standard tool flow for boot firmware generation takes care of mapping the SoC-to-FPGA bridge into addressable memory space.

For more information about the FPGA First boot and configuration scheme and generating boot firmware for the Intel Agilex HPS, refer to the *Intel Agilex SoC Boot User Guide*.

5.1.8.1.2. Lightweight SoC-to-FPGA Bridge

GUIDELINE: Use the lightweight SoC-to-FPGA bridge to connect IP that needs to be controlled by the HPS.

The lightweight SoC-to-FPGA bridge allows masters in the HPS to access memory-mapped control slave ports in the FPGA portion of the SoC device. Typically, only the MPU inside the HPS accesses this bridge to perform control and status register accesses to peripherals in the FPGA.

GUIDELINE: Do not use the lightweight SoC-to-FPGA bridge for FPGA memory. Instead use the SoC-to-FPGA bridge for memory.

When the MPU accesses control and status registers within peripherals, these transactions are typically strongly ordered (non-posted). By dedicating the lightweight SoC-to-FPGA bridge to only register accesses, the access time is minimized because bursting traffic is routed to the SoC-to-FPGA bridge instead. The lightweight SoC-to-FPGA bridge has a fixed 32-bit width connection to the FPGA fabric because most IP cores implement 32-bit control and status registers; but Platform Designer can adapt the transactions to widths other than 32 bits within the interconnect generated in the FPGA portion.

5.1.8.1.3. FPGA-to-SoC Bridge

GUIDELINE: Use the FPGA-to-SoC bridge for cache coherent memory accesses to the CCU or non-cacheable accesses to the HPS SDRAM from masters in the FPGA.

The FPGA-to-SoC bridge provides access to the peripherals in the HPS or access to the HPS SDRAM from the FPGA. This access is available to any master implemented in the FPGA fabric. You can configure the bridge slave, which is exposed to the FPGA fabric, to support the ACE-Lite protocol, with a data width of 128, 256, and 512 bits.

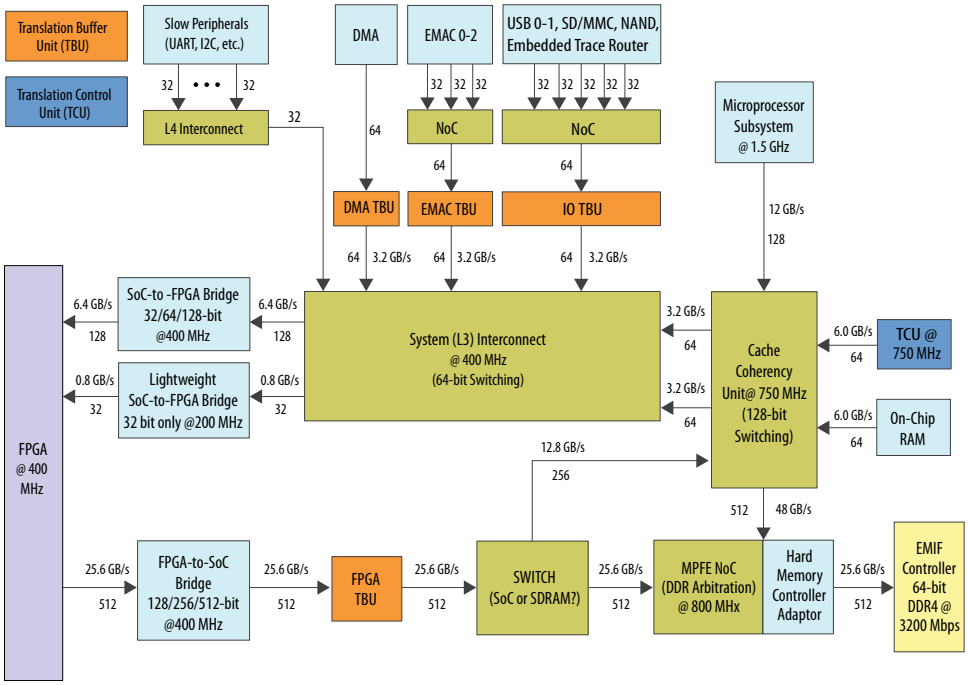
For more information about the ACE-Lite protocol extensions for cache coherent transactions, refer to the AMBA AXI and ACE Protocol Specification on the Arm* Developer website.

5.1.8.1.4. Interface Bandwidths

To identify which interface should be used to move data between the HPS and FPGA fabric, an understanding of the bandwidth of each interface is necessary. The figure below illustrates the peak throughput available between the HPS and FPGA fabric as well as the internal bandwidths within the HPS. The example shown assumes that the FPGA fabric operates at 400 MHz, the MPU operates at 1500 MHz, and the 64-bit external SDRAM operates at 3200 Mbits per second.

Figure 8. Intel Agilex HPS Memory-Mapped Bandwidth

For abbreviations, refer to the figure in *Overview of HPS Memory-Mapped Interfaces*.



Relative Latencies and Throughputs for Each HPS Interface

Interface	Transaction Use Case	Latency	Throughput
SoC-to-FPGA	MPU accessing memory in FPGA	Medium	Medium
SoC-to-FPGA	MPU accessing peripheral in FPGA	Medium	Very Low
Lightweight SoC-to-FPGA	MPU accessing register in FPGA	Low	Low
Lightweight SoC-to-FPGA	MPU accessing memory in FPGA	Low	Very Low
FPGA-to-SoC	FPGA master accessing non-cache coherent SDRAM	High	Medium
FPGA-to-SoC	FPGA master accessing SoC on-chip RAM	Low	High
FPGA-to-SoC	FPGA master accessing SoC peripheral	Low	Low
FPGA-to-SoC	FPGA master accessing coherent memory resulting in cache miss	High	Medium
FPGA-to-SoC	FPGA master accessing coherent memory resulting in cache hit	Low	Medium-High
FPGA-to-SoC	FPGA master accessing SoC directly	Medium	High-Very High

Note: For the interfaces with no configuration recommended, refer to the corresponding interface sections: "SoC-to-FPGA Bridge", "Lightweight SoC-to-FPGA Bridge", and "FPGA-to-SoC Bridge".

GUIDELINE: Avoid using the SoC-to-FPGA bridge to access peripheral registers in the FPGA from the MPU.

The SoC-to-FPGA bridge is optimized for bursting traffic and peripheral accesses are typically short word-sized accesses of only one beat. As a result if peripherals are accessed through the SoC-to-FPGA bridge, the transaction can be stalled by other bursting traffic that is already in flight.

GUIDELINE: Avoid using the lightweight SoC-to-FPGA bridge to access memory in the FPGA from the MPU.

The lightweight SoC-to-FPGA bridge is optimized for non-bursting traffic and typically memory accesses are performed as bursts (often 32 bytes due to cache operations). As a result, if memory is accessed through the lightweight SoC-to-FPGA bridge, the throughput is limited.

GUIDELINE: Use soft logic in the FPGA (for example, a DMA controller) to move shared data between the HPS and FPGA. Avoid using the MPU and the HPS DMA controller for this use case.

When moving shared data between the HPS and FPGA Intel recommends to do so from the FPGA instead of moving the data using the MPU or HPS DMA controller. If the FPGA must access cache coherent data then it must access the FPGA-to-SoC bridge with the appropriate ACE-Lite cache extensions signaling to issue a cacheable transaction. If non-cache coherent data must be moved to the FPGA or HPS, a DMA engine implemented in FPGA logic can move the data through the FPGA-to-SoC bridge, achieving the highest throughput possible. Even though the HPS includes a DMA engine internally that can move data between the HPS and FPGA, its purpose is to assist peripherals that do not master memory or provide memory to memory data movements on behalf of the MPU.

5.1.8.2. Recommended System Topologies

Selecting the right system topology can help your design achieve the highest throughput possible. For optimum performance, observe Intel's topology guidelines moving data between the HPS and FPGA. These guidelines cover both cache coherent and non-cache coherent data movements.

5.1.8.2.1. System Level Cache Coherency

Table 26. Device Variant Checklist

Number	Done?	Checklist Item
1		Consider how many and which masters to use.
2		Determine how to manage cacheable accesses.

Consider how many masters and what masters to use:

- MPU
- DMA
- Peripherals with master interfaces
- Masters in FPGA connected to HPS.

  Cache coherency is a fundamental topic to understand any time data must be shared amongst multiple masters in a system. In the context of a SoC device these masters can be the MPU, DMA, peripherals with master interfaces, and masters in the FPGA connected to the HPS. Since the MPU contains level 1 and level 2 cache controllers, it can hold more up-to-date contents than main memory in the system. The HPS supports two mechanisms to make sure masters in the system observe a coherent view of memory: ensuring main memory contains the latest value, or have masters access a directory-based CCU fabric using the ACE-Lite interface.

  The MPU can allocate buffers to be non-cacheable which ensures data is never cached by the L1 and L2 caches. The MPU can also access cacheable data and either flush it to main memory or copy it to a non-cacheable buffer before other masters attempt to access the data. Operating systems typically provide mechanisms for maintaining cache coherency both ways described above.

  Masters in the system access coherent data by either relying on the MPU to place data into main memory instead of having it cached, or by having the master in the system perform a cacheable access through the CCU. The mechanism you use depends on the size of the buffer of memory the master is accessing.

  For more information, refer to the *Interfacing to the FPGA* section.

GUIDELINE: Ensure that data accessed through the CCU fits in the 1 MB L2 cache to avoid thrashing overhead.

Since the L2 cache is 1 MB in size, if a master in the system frequently accesses buffers whose total size exceeds 1 MB, thrashing results.

Cache thrashing is a situation where the size of the data exceeds the size of the cache, causing the cache to perform frequent evictions and prefetches to main memory. Thrashing negates the performance benefits of caching the data.

In potential thrashing situation, it makes more sense to have the masters access non-cache coherent data and allow software executing on the MPU maintain the data coherency throughout the system.

GUIDELINE: For small buffers of data shared between the MPU and system masters, consider having the system master perform cacheable accesses to avoid overhead caused by cache flushing operations.

If a master in the system requires access to smaller coherent blocks of data then you should consider having the MPU access the buffer as cacheable memory and the master in the system perform cacheable accesses to the data. Cacheable accesses to the CCU through the ACE-Lite protocol supported by the FPGA-to-HPS bridge ensure that the master and MPU access the same copy of the data. By having the MPU use cacheable buffers and the system master performing cacheable accesses, software does not have to maintain system wide coherency ensuring both the MPU and system master observe the same copy of data.

Related Information

- Interfacing between the FPGA and HPS on page 40

- Interfacing between the FPGA and HPS on page 40

5.1.8.2.2. HPS Accesses to FPGA Fabric

There are two bridges available for masters in the HPS to access the FPGA fabric. Each bridge is optimized for specific traffic patterns and as a result you should determine which is applicable to your system if an HPS master needs to access the FPGA fabric.

GUIDELINE: Connect the HPS to soft logic peripherals in the FPGA through the lightweight SoC-to-FPGA bridge.

If your hardware design has peripherals that are accessible to the HPS then you should connect them to the lightweight SoC-to-FPGA bridge. Peripherals are typically accessed by the HPS MPU one register at a time using strongly ordered (non-posted) accesses. Since the accesses are strongly ordered, the transaction from the MPU does not complete until the response from the slave returns. As a result, strongly ordered accesses are latency sensitive so the lightweight SoC-to-FPGA bridge is included in the HPS to reduce the latency of strongly ordered accesses.

GUIDELINE: Connect the HPS to FPGA memory through the SoC-to-FPGA bridge.

If your hardware design has memory that is accessible to the HPS then you should connect it to the SoC-to-FPGA bridge. Unlike the lightweight SoC-to-FPGA bridge, the SoC-to-FPGA bridge is intended for bursting traffic such as DMA transfers or MPU software execution from FPGA memory.

GUIDELINE: If the HPS must access both memory and peripherals in your FPGA logic, use SoC-to-FPGA and lightweight SoC-to-FPGA bridge.

It is important to include both SoC-to-FPGA and lightweight SoC-to-FPGA bridge in your design if the FPGA logic contains a mix of memory and peripherals accessible to the HPS. Since peripheral accesses are typically latency-sensitive, using the lightweight SoC-to-FPGA bridge for those accesses prevents starvation when other bursting accesses to the FPGA fabric are made through the SoC-to-FPGA bridge. Both bridge can be accessed in parallel if there are multiple HPS masters accessing the FPGA fabric at the same time so including both bridge can also improve the performance of the system.

5.1.8.2.3. MPU Sharing Data with FPGA

You can optimize data throughput by selecting the correct method of sharing data between the HPS and the FPGA. This section assumes that the HPS SDRAM is the data source and the FPGA require access to it. There are two main ways for the FPGA to access data that originates in HPS SDRAM:

- FPGA accesses non-cached data directly through the FPGA-to-SoC bridge targeting the SDRAM.
- FPGA accesses cached data through the FPGA-to-SoC bridge targeting the CCU.

If the data in the SDRAM is the most recent copy of the data (software managed coherency) then the highest throughput method of accessing the data is to have masters in the FPGA access the data directly through the FPGA-to-SoC bridge.

If the data in the SDRAM potentially is not the most recent copy of the data and software does not flush the MPU caches to ensure system wide coherency is maintained, then the FPGA master should perform cacheable transactions to the FPGA-to-SoC bridge to ensure the most recent data is accessed.

GUIDELINE: Avoid using the HPS DMA controller to move data between the FPGA and HPS. Use a soft DMA controller in the FPGA fabric instead. Use the HPS DMA controller only for memory copies or peripheral data movements that remain inside the HPS.

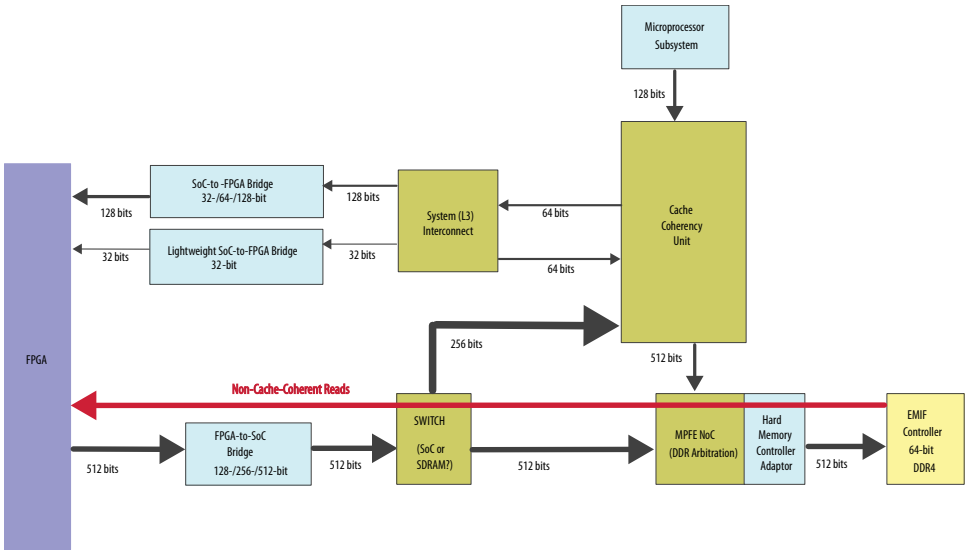
It is not recommended to use the HPS DMA to move the data to the FPGA because the DMA bandwidth into the HPS SDRAM is limited. The HPS DMA is intended to be used to move buffers on behalf of the MPU or used for transfers between peripherals and memory. As a result, any time the FPGA needs access to buffers in HPS memory, or if the HPS requires access to data stored in the FPGA, it is always recommended to have masters in the FPGA perform these transfers instead of the HPS initiating them.

5.1.8.2.4. Examples of Cacheable and Non-Cacheable Data Accesses From the FPGA

Example 1: FPGA Reading Non-Cache Coherent Data from HPS EMIF Directly

In this example the FPGA requires access to data that is stored in the HPS EMIF. For the FPGA to access the same copy of the data as the MPU has access to, the L1 data cache and L2 cache need to be flushed if they already have a copy of the data. Once the HPS EMIF contains the most up-to-date copy of the data, the optimal path for the FPGA to access this data is for FPGA masters to read the data through the FPGA-to-SoC bridge directly targeting the SDRAM.

Figure 9. FPGA Reading Non-Cache Coherent Data

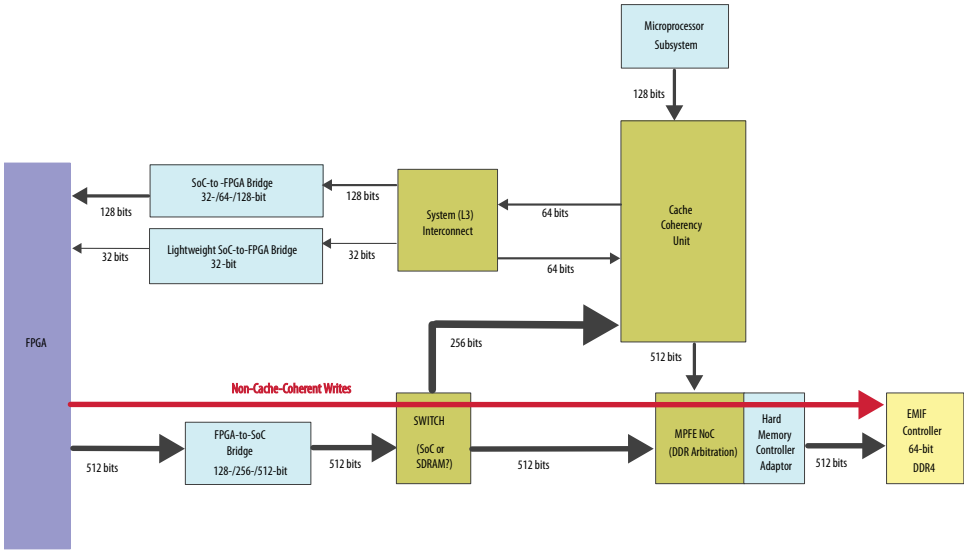


The FPGA-to-SoC bridge can be optimized to maximize the read throughput by setting the bridge width according to your system requirements. Intel recommends to use a burst capable master in the FPGA to read from the SDRAM, capable of posting burst lengths of four beats or larger.

Example 2: FPGA Writing Non-Cache Coherent Data into HPS EMIF Directly

In this example the HPS MPU requires access to data that originates from within the FPGA. For the MPU to be able to access the data coherently after it is written, software may need to flush or invalidate cache lines before the transfer starts, to ensure that the SDRAM contains the latest data after it is written. Failing to perform cache operations can cause one or more cache lines to eventually become evicted overwriting the data that was written by the FPGA master.

Figure 10. FPGA Writing Non-Cache Coherent Data



Note: Like in "Example 1: FPGA Reading Data from HPS EMIF Directly", the FPGA-to-SoC bridge can be optimized to maximize the write throughput by setting the bridge width according to your system requirements.

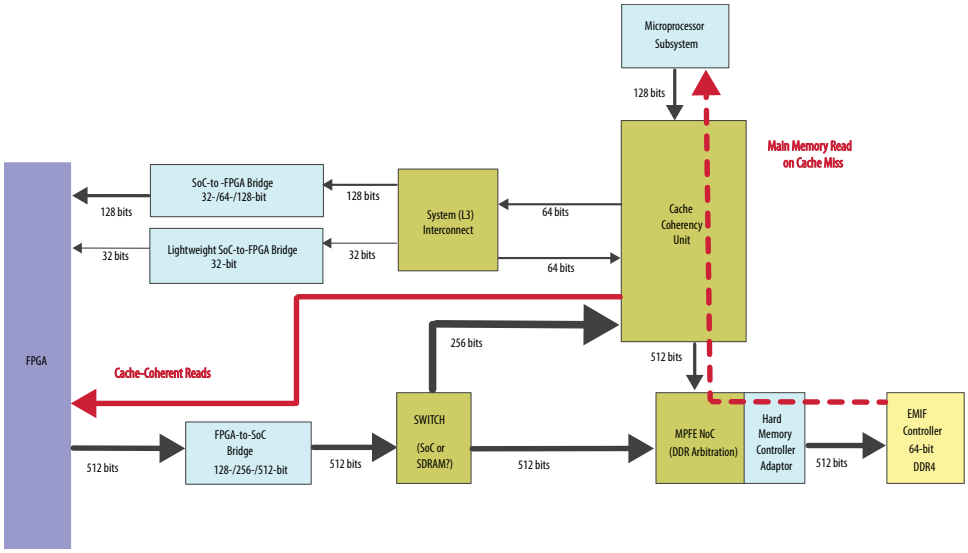
Example 3: FPGA Reading Cache Coherent Data from HPS

In this example the FPGA requires access to data originating in the HPS. The MPU in the HPS recently accessed this data so there is a chance that the data is still contained in the cache and therefore it may be optimal for the FPGA to access the cached data. To avoid the overhead of software having to flush dirty cache lines the FPGA can perform cache coherent reads through the FPGA-to-SoC bridge. It is important that the buffers being read be relatively small. Otherwise, the L2 cache might thrash reading data from SDRAM for most of the transfer. For large buffer transfers it is more appropriate to have the FPGA read data through the FPGA-to-SoC bridge directly accessing the SDRAM, as shown in "Example 1: FPGA Reading Data from HPS EMIF Directly".

GUIDELINE: Perform full accesses targeting FPGA-to-SoC bridge.

For the transaction to be cacheable, the FPGA master must read from the FPGA-to-SoC bridge and utilize the cache extension signaling of the ACE-Lite protocol. For more information about the ACE-Lite protocol signaling extensions for cache coherent accesses, refer to the *Cache Coherency Unit* section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Figure 11. FPGA Reading Cache Coherent Data



GUIDELINE: Perform cacheable accesses aligned to 64 bytes targeting the FPGA-to-SoC bridge.

The CCU of the HPS is optimized for transactions that are the same size as the cache line (64 bytes). As a result you should attempt to align the data to 64 byte boundaries and ensure after data width adaptation the burst length into the 512-bit FPGA-to-SoC bridge is maximized. For example, a 128-bit FPGA master should align the data to be 64 byte aligned and perform full 128-bit (16-byte) accesses with a burst length of 4.

GUIDELINE: Access 64 bytes per cacheable transaction.

Ensure that each burst transaction accesses 64 bytes. Each transaction must start on a 64-byte boundary.

Table 27. Burst Lengths for 64-byte Alignment

FPGA Master Width (Bits)	Access Size (Bytes)	Burst Length
32	4	16
64	8	8
128	16	4
256	32	2
512	64	1

Example 4: FPGA Writing Cache Coherent Data to HPS

In this example the HPS MPU requires access to data that originates in the FPGA. The most efficient mechanism for sharing small blocks of data with the MPU is to have logic in the FPGA perform cacheable writes to the HPS. It is important that the amount of data to be written to the HPS be in the form of relatively small blocks because large block writes cause the L2 cache to thrash, causing the cache to write to SDRAM for

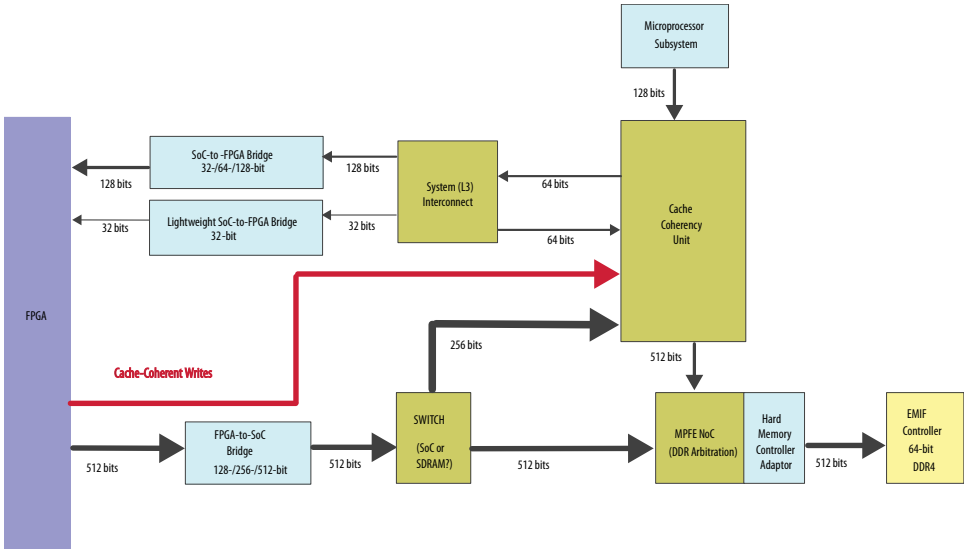
most of the transfer. For large buffer transfers it is more appropriate to have the FPGA write data to the FPGA-to-SoC bridge targeting SDRAM directly as shown in Example 2.

GUIDELINE: Perform full accesses targeting FPGA-to-SoC bridge.

For the transaction to be cacheable, the FPGA master must write to the FPGA-to-SoC bridge and utilize the cache extension signaling of the ACE-Lite protocol. For more information about the ACE-Lite protocol signaling extensions for cache coherent accesses, refer to the *Cache Coherency Unit* section in the *Intel Agilex Hard Processor System Technical Reference Manual*.

Figure 12. FPGA Writing Cache Coherent Data

For abbreviations, refer to the figure in *Overview of HPS Memory-Mapped Interfaces*.



GUIDELINE: When L2 ECC is enabled, ensure that cacheable accesses to the FPGA-to-SoC bridge are aligned to 8-byte boundaries.

If you enable error checking and correction (ECC) in the L2 cache you must also ensure each 8-byte group of data is completely written. The L2 cache performs ECC operations on 64-bit boundaries so when performing cacheable accesses you must always align the access to 8-byte boundaries and write to all eight lanes at once. Failing to follow these rules results in double bit errors, which cannot be recovered.

Regardless whether ECC is enabled or disabled, 64 byte cache transactions result in the best performance. For more information about 64 byte cache transactions, refer to **GUIDELINE: Access 64 bytes per cacheable transaction** in the "Example 3: FPGA Reading Cache Coherent Data from HPS" section.

GUIDELINE: When L2 ECC is enabled, ensure that cacheable accesses to the FPGA-to-SoC bridge have groups of eight write strobes enabled.

- For FPGA-to-SoC accesses from 32-bit FPGA masters, burst length must be 2, 4, 8, or 16 with all write byte strobes enabled.
- For FPGA-to-SoC accesses from 64-bit FPGA masters, all write byte strobes must be enabled.
- For FPGA-to-SoC accesses from 128-bit FPGA masters, the upper eight or lower eight (or both) write byte strobes must be enabled.

Related Information

Intel Agilex Hard Processor System Technical Reference Manual

5.1.8.3. Recommended Starting Point for SoC-to-FPGA Interface Designs

GUIDELINE: Intel recommends that you start with the Golden Hardware Reference Design (GHRD) as an example of interfacing the HPS to soft IP in the FPGA.

The Golden Hardware Reference Design (GHRD) has the optimum default settings and timing that you can use as a basis for your "Getting Started" system.

For more information, refer to the "Golden Hardware Reference Design (GHRD)" section.

Related Information

Golden Hardware Reference Design (GHRD)

5.1.8.4. Information on How to Configure and Use the Bridges

By default, the SSBL only brings all the bridges out of reset. It does not automatically configure or enable the bridges. You must specifically configure and enable all the bridges according to your own design. This can be accomplished by creating a "`u-boot.scr`" script file that is executed by the SSBL, where the SSBL modifies any registers necessary to configure the bridges. At this point the bridges are configured and enabled, and cannot be changed by the SSBL, even during any future FPGA configurations.

For more information, refer to the example located on the Creating the U-boot Script located on *RocketBoards.org*.

Related Information

Creating the U-boot script

5.1.9. Implementing the Intel Agilex HPS Component

The HPS component is a wrapper that interfaces logic in your design to the:

- HPS hard logic
- Simulation models
- Bus functional models (BFMs)
- Software handoff files

The HPS component instantiates the HPS hard logic in your design and enables other soft components to interface with the HPS hard logic.

For more information, refer to the *Intel Agilex Hard Processor System Component Reference Manual*.

Related Information

Intel Agilex Hard Processor System Component Reference Manual

5.2. Design Entry for FPGA-only Devices

5.2.1. Clocking and Reset Design Considerations

You must follow the configuration clocking guidelines detailed in the *Intel Agilex Configuration User Guide* to ensure proper operation. The continual increases in clock frequency, device size, and design complexity now necessitate a well-thought out reset strategy that considers the possible effects of slight differences in the release from reset. This reset strategy must hold the device in reset until all registers and core logic are in user mode. Intel strongly recommends that you use the **nINIT_DONE**. The output of the Reset Release Intel Agilex FPGA IP is one of the initial inputs to your reset circuit. For more information, refer to *AN 891: Using the Reset Release FPGA IP*.

Related Information

- AN 891: Using the Reset Release FPGA IP
- Intel Agilex Configuration User Guide

5.2.2. 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 Intel Agilex 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.

5.2.2.1. Making FPGA Pin Assignments

Table 28. Making FPGA Pin Assignments Checklist

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

With the Intel 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 Intel 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 Intel 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.

Intel Quartus Prime designers can then pass the pin location information to PCB designers. Pin assignments between the Intel 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 Intel Quartus Prime software generates the `.pin` file. You can use this file to verify that each pin is correctly connected in the board schematics.

5.2.2.2. Early Pin Planning and I/O Assignment Analysis for the FPGA Device

Table 29. 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 Intel 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 SERDES 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 Intel 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 Intel Quartus Prime Fitter for the final sign-off of the pin assignments.

5.2.2.3. I/O Features and Pin Connections

Intel Agilex 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.

5.2.2.3.1. I/O Signaling Type

Table 30. 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.

Intel Agilex 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 (for example, external memory interface data and address buses). 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.

Intel Agilex I/O pins are organized in pairs to support differential standards. Each I/O pin pair can support unidirectional differential input or output operations. Half of the true differential channels support dedicated transmitter pins and the other half support dedicated true receiver pins. 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 Intel Quartus Prime software automatically places the corresponding negative pin.

5.2.2.3.2. Selectable Standards and Flexible I/O Banks

Table 31. 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. The I/O banks are located in the top and bottom I/O bank row. Each I/O bank contains its own PLL, DPA, and SERDES circuitries.
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 and transceiver features.
7		Place I/O pins that share OCT calibration block in the same I/O tile.

Intel Agilex I/O pins are arranged in groups called modular I/O banks. Be sure to use the correct dedicated pin inputs for signals such as clocks and global control signals.

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

To accommodate voltage-referenced I/O standards, each I/O bank supports multiple VREF pins feeding a common VREF bus. Intel Agilex GPIO bank supports internal and external VREF types. Each I/O lane must share the same VREF type. 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_PIO} voltage level and a single V_{REF} voltage level at a given time. If the VREF pins are not used as voltage references, they cannot be used as generic I/O pins and should be tied to V_{CCIO_PIO} 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 V_{REF} setting. Voltage-referenced bi-directional and output signals must drive out at the I/O bank's V_{CCIO_PIO} voltage level.

Different I/O banks include different support for 1.5V True Differential Signaling, and the Intel Agilex transceiver banks include additional I/O support.

The GPIO bank supports true differential input standard at 1.2V/ 1.5V V_{CCIO_PIO}. The true differential output standard is supported at 1.5V V_{CCIO_PIO} bank. The I/O pins form pairs of unidirectional true differential channels.

5.2.2.3.3. Dual-Purpose and Special Pin Connections

Table 32. 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.

Intel Agilex 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 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.

5.2.2.3.4. Intel Agilex I/O Features

Table 33. Intel Agilex I/O Features Checklist

Number	Done?	Checklist Item
1		Check available device I/O features that can help I/O interfaces: 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.
4		Choose the appropriate mode of DPA, non-DPA, or soft-CDR for high-speed LVDS SERDES interfaces. For more information, refer to the *Intel Agilex LVDS SERDES Design Guidelines* section in the *Intel Agilex General Purpose I/O and LVDS SERDES User Guide*.

The Intel Agilex bi-directional I/O element (IOE) features support rapid system integration while simultaneously providing the high bandwidth required to maximize internal logic capabilities and system-level performance. Advanced features for device interfaces assist in high-speed data transfer into and out of the device and reduce the complexity and cost of the PCB.

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

Related Information

Intel Agilex General Purpose I/O and LVDS SERDES User Guide

5.2.2.4. Clock and PLL Selection

Table 34. 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.

Intel Agilex 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.

Intel Agilex devices contain dedicated resources for distributing signals throughout the fabric with balanced delay. These resources are typically used for clock signals. You can also use these resources for other signals with low-skew requirements. In Intel Agilex devices, these resources are implemented as a programmable clock routing, which allows for the implementation of low-skew clock networks of variable size.

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 Intel 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.

5.2.2.5. PLL Feature Guidelines

Table 35. 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 Intel Quartus Prime parameter editor to enter your settings in core, and check the results to verify whether particular features and input/output frequencies can be implemented in a particular PLL.

You can use I/O PLLs to reduce the number of oscillators required on the board, as well as to reduce the clock pins used in the FPGA by synthesizing multiple clock frequencies from a single reference clock source.

Intel Agilex device PLLs are feature rich, and support advanced capabilities such as clock feedback modes, switchover, and dynamic phase shifting.

5.2.2.5.1. Clock Feedback Mode

Table 36. Clock Feedback Mode Checklist

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

Intel Agilex PLLs support six different clock feedback modes.

5.2.2.5.2. Clock Outputs

Table 37. 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. I/O PLL can connect to a clock network or a dedicated clock pin.

5.2.2.6. Clock Control Features

Table 38. Clock Control Features Checklist

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

Intel Agilex devices uses these clock control features: clock gating and clock divider. The clock from the I/O PLL output can be gated dynamically. These clock signals along with other clock sources go to the periphery distributed clock multiplexer (DCM). In the periphery DCM, the clock signal can either pass straight through, be gated by the root clock gate, or be divided by the clock divider.

5.2.2.7. I/O Simultaneous Switching Noise

Table 39. 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.

continued...

Number	Done?	Checklist Item
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.

5.3. EMIF Considerations

This section describes the EMIF design considerations for the HPS and FPGA.

5.3.1. Memory Interfaces

Table 40. Memory Interfaces Checklist

Number	Done?	Checklist Item
1		Use the External Memory Interfaces Intel Agilex core for each memory interface, and follow connection guidelines and restrictions in *Intel Agilex FPGA External Memory Interface Overview* and the *External Memory Interfaces IP - Support Center* web page.
2		For a given bank, most memory pins are tied to a dedicated location. Refer to the *Intel Agilex Device Family Pin Connection Guidelines* and *Intel Agilex External Memory Interface Pin Information* for pin assignments.

Intel Agilex devices provide an efficient architecture to quickly and easily fit wide external memory interfaces with their small modular I/O banks. The Intel Agilex FPGA can support DDR external memory on any I/O banks on all sides of the device that do not support transceivers.

The self-calibrating External Memory Interfaces IP core is optimized to take advantage of the Intel Agilex I/O structure. The External Memory Interfaces 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 Intel FPGA IP functions, the External Memory Interfaces IP core is instantiated automatically. If you design multiple memory interfaces into the device using Intel FPGA IP core, generate a unique interface for each instance to ensure good results instead of designing it once and instantiating it multiple times.

The data strobe DQS and data DQ pin locations are fixed in Intel Agilex devices. Before you design your device pin-out, refer to the memory interface guidelines in the *Intel Agilex FPGA External Memory Interface Overview* for details and important restrictions related to the connections for these and other memory-related signals.

You can implement a protocol that is not supported by External Memory Interfaces IP core by using the PHY Lite for Parallel Interfaces Intel Agilex FPGA IP core.

Address and command pins within the address/command bank must follow a fixed pin-out scheme, as defined in the `<variation_name>_readme.txt` file generated with your IP core. The pin-out scheme varies according to the topology of the memory interface. The pin-out scheme is a hardware requirement that you must follow. Some schemes require three lanes to implement address and command pins, while others require four lanes.

Related Information

- Intel Agilex External Memory Interface Pin Information
- Intel Agilex Device Family Pin Connection Guidelines
- Intel Agilex FPGA External Memory Interface Overview
- External Memory Interfaces IP - Support Center

5.3.2. HPS EMIF Design Considerations

A critical component to the HPS is its external SDRAM memory. The following design considerations help you properly design the interface between SDRAM memory and the HPS.

When connecting external SDRAM to the HPS, refer to the following EMIF planning tools and essential documentation:

EMIF Planning Tools

Tools	Description
External Memory Interfaces IP - Support Center	The External Memory Interfaces IP - Support Center is a collection of tools and documentation resources to aid in the design of external memory interfaces for Intel FPGAs.

For more information about EMIF IP generation and Intel Quartus Prime compilation and timing closure aids, refer to the External Memory Interfaces IP - Support Center website.

Essential Documentation

Documentation	Description
Intel Agilex General Purpose I/O and LVDS SERDES User Guide	The *Intel Agilex General Purpose I/O and LVDS SERDES User Guide* describes the I/O column architecture and where the specific Hard Memory Controller block accessible to the HPS resides, For guidance on connecting the HPS-accessible hard memory controller block to the HPS, refer to *Package Selection and I/O Vertical Migration Support* of the *Intel Agilex General Purpose I/O and LVDS SERDES User Guide*. This section shows the I/O row and bank locations for all device and package combinations across all Intel Agilex family variants, including the relative location of the HPS to its accessible banks.
Intel Agilex FPGA External Memory Interface Overview	The *Intel Agilex External Memory Interfaces User Guide* includes the details required to understand what specific I/O banks are used for HPS external memory interfaces and where address/command, ECC and data signals are located. The user guide also consists of important information on restrictions on the placement of these external memory interface signals within the banks and any flexibility the designer has in varying from the default placement. While Intel recommends that you familiarize yourself

continued...

Documentation	Description
	with all the content available in this user guide, understanding the following sections is a prerequisite to properly design the Intel Agilex EMIF for the HPS IP in your application. • **Intel Agilex EMIF IP Product Architecture** chapter—This section describes in greater detail the I/O Row, HMC, I/O lanes, and the hardened feature support for DDR SDRAM memories in the I/O elements. • **Restrictions on I/O Bank Usage for Intel Agilex EMIF IP with HPS** chapter—This section provides a diagram that shows the specific I/O bank and lane locations for address/command, ECC, and data signals.

The following design guidelines supplement the information found in the above referenced documentation.

Related Information

- Intel Agilex FPGA External Memory Interface Overview
- Intel Agilex General Purpose I/O and LVDS SERDES User Guide
- HPS Memory Debug on page 65
- External Memory Interfaces IP - Support Center website
- Intel Agilex General Purpose I/O and LVDS SERDES User Guide

5.3.2.1. Considerations for Connecting HPS to SDRAM

The hard memory controller for the Intel Agilex HPS is in the FPGA I/O row along with the other hardware memory controllers. The HPS EMIF has optimized the interconnect to the HPS core.

Instantiating the Intel Agilex HPS EMIF IP

Connecting external SDRAM to the Intel Agilex HPS requires the use of an EMIF IP that is specific to the HPS. Follow the below guidelines for properly instantiating and configuring the correct EMIF IP for the HPS.

GUIDELINE: Instantiate the Intel Agilex External Memory Interfaces for HPS IP in Platform Designer.

You must use a specific EMIF IP in Platform Designer to connect the HPS to external SDRAM memory.

The EMIF module is found in the IP catalog pane by selecting: **Library ➤ Processors and Peripherals ➤ Hard Processor Components ➤ External Memory Interfaces for HPS Intel Agilex**.

GUIDELINE: Connect the `hps_emif` conduit to the HPS component

To connect the HPS to the EMIF in Platform Designer, you must connect the `hps_emif` conduit in the instantiated **emif_fm_hps_1** module to the `hps_emif` conduit in the **agilex_hps_0** module.

GUIDELINE: You must provide a free running and stable reference clock source to external memory interface before the start of device configuration.

For more information, refer to the *Intel Agilex FPGA EMIF IP Overview*.

GUIDELINE: Make sure the HPS EMIF IP block is not reset while the HPS is accessing external SDRAM or resources in the multiport front end (MPFE).

Asserting reset to the HPS EMIF IP block should coincide with the HPS reset assertion unless the application can save and recover context in co-ordination with HPS EMIF IP reset assertion. This can be achieved simply by connecting the HPS EMIF reset input to one or a combination of resets from the following sources: HPS reset outputs (for example: `h2f_reset, h2f_cold_reset`), other resets in the system that also source an HPS cold reset input (for example: `nCONFIG` and `HPS_COLD_nRESET` reset input pin.

If the HPS EMIF IP is reset without resetting the HPS as described above, the application must put the MPFEt in reset using the `brgmodrst` register, bit 6 (ddrsch) in the Reset Manager before HPS EMIF IP reset assertion and not release it until after the HPS EMIF IOPLL has locked. Failure to do so can result in locking up the processor on subsequent accesses to external SDRAM or resources in the MPFE.

GUIDELINE: Ensure that the HPS EMIF controller Data Mask (DM) pins are enabled.

When you instantiate the memory controller in Platform Designer, you must select the checkbox to enable the data mask pins. If this control is not enabled, data corruption occurs any time a master accesses data in SDRAM that is smaller than the native word size of the memory.

Note: The checkbox to enable data masking is found in the "Parameters" tab for the External Memory Interfaces for HPS Intel Agilex Intel FPGA IP within the Topology section of the memory sub-tab.

GUIDELINE: Ensure that you choose only DDR4 components or modules in configurations that are supported by the Intel Agilex EMIF for HPS IP and your specific device and package combination.

Intel's *External Memory Interface Spec Estimator* is a parametric tool that allows you to compare supported external memory interface types, configurations and maximum performance characteristics in Intel FPGA and SoC devices.

Related Information

- Intel Agilex FPGA External Memory Interface Overview
- External Memory Interface Spec Estimator

5.3.2.2. HPS EMIF I/O Locations

The Intel Agilex EMIF for HPS IP includes default pin location assignments for all the external memory interface signals in constraint files created at IP generation time and read by Intel Quartus Prime Pro Edition software during design compilation.

GUIDELINE: Intel recommends that you use these automated default pin location assignments as a starting point.

You may need to modify the default pinout to meet the restrictions shown in this section.

GUIDELINE: Verify the HPS memory controller I/O locations in the Intel Quartus Prime project pinout file in the "`output_files`" sub-folder before finalizing board layout.

By default, Intel Quartus Prime generates output reports, log files and programming files in the `output_files` subfolder of the project folder. See the `.pin` text file after compilation for the pinout for your design, including the pin locations for the HPS EMIF.

GUIDELINE: Make sure all I/O associated with the HPS memory interface are located within the active HPS EMIF I/O banks.

It is critical that you ensure all I/O necessary for a functioning HPS memory interface are located within the active banks for your HPS memory width.

For a description about the pin assignment and the restriction on I/O Bank Usage for Intel Agilex EMIF IP with HPS, refer to the *Intel Agilex FPGA EMIF IP Overview*.

Table 41. HPS EMIF I/O Locations

EMIF Width	Tile 3C								Tile 3D							
	Top				Bottom				Top				Bottom			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
16-bit	GPIO				GPIO				NC		16-bit Data		NC	Addr/ Command/RZQ/ RefClk		
16-bit + ECC	GPIO				GPIO				NC		16-bit Data		ECC	Addr/ Command/RZQ/ RefClk		
32-bit	GPIO				GPIO				32-bit Data				NC	Addr/ Command/RZQ/ RefClk		
32-bit + ECC	GPIO				GPIO				32-bit Data				ECC	Addr/ Command/RZQ/ RefClk		
64-bit	GPIO (with restrictions)				64-bit Data								NC	Addr/ Command/RZQ/ RefClk		
64-bit + ECC	GPIO (with restrictions)				64-bit Data								ECC	Addr/ Command/RZQ/ RefClk		
Note: NC = No Connect																

Pin Assignments

1. Within a single data lane (which implements a single x8 DQS group):

- DQ pins must use pins at indices 0, 1, 2, 3, 8, 9, 10, 11. You may swap the locations between the DQ bits (that is, you may swap location of DQ[0] and DQ[3]) so long as the resulting pin-out uses pins at these indices only.
- DM/DBI pin must use pin at index 6. There is no flexibility.
- DQS_P must use pin at index 4, and DQS_N must use pin at index 5. There is no flexibility.
- pin index 7 must be "no connect".

2. Assignment of data lanes must be as illustrated in the above figure. You are allowed to swap the locations of entire byte lanes (that is, you may swap locations of byte 0 and byte 1) so long as the resulting pin-out uses only the lanes permitted by your HPS EMIF configuration, as shown in the above figure.
3. I/O Tile 3D, Bottom Bank Lanes 0, 1, and 2 must only be used for Address/Command/RZQ/REFCLK, otherwise "no connect".
4. If not using ECC, I/O Tile 3D, Bottom Bank Lane 3 must be "no connect". If using ECC, the ECC DQS group must be in I/O Tile 3D, Bottom Bank Lane 3.
5. You must not change placement of the address and command pins from the default placement.
6. Place the ALERT# pin at I/O Tile 3D, Bottom Bank Lane 2, pin index 8 only, else "no connect".
7. HPS REFCLK_P must use I/O Tile 3D, Bottom Bank Lane 2, pin index 0. HPS REFCLK_N must use I/O Tile 3D, Bottom Bank Lane 2, pin index 1.
8. RZQ must use I/O Tile 3D, Bottom Bank Lane 2, pin index 2.

DQ/DQS Group Placement

Configuration	DQS Group Placement
16 bit	Must be placed in I/O lanes Top[1:0] of Bank 3D
16 bit + ECC	Must be placed in I/O lanes Top[1:0] of Bank 3D and Bottom[3] of Bank 3D
32 bit	Must be placed in I/O lanes Top[3:0] of Bank 3D
32 bit + ECC	Must be placed in I/O lanes Top[3:0] of Bank 3D and Bottom[3] of Bank 3D
64 bit	Must be placed in I/O lanes Top[3:0] of Bank 3D and Bottom[3:0] of Bank 3C
64 bit + ECC	Must be placed in I/O lanes Top[3:0] of Bank 3D and Bottom[3:0] of Bank 3C and Bottom[3] of Bank 3D

Note: In all cases, the DQ/DQS groups can be swapped around in the I/O banks shown.

Related Information

Intel Agilex FPGA External Memory Interface Overview

5.3.2.3. HPS Memory Debug

GUIDELINE: Verify the memory interface is operational using an FPGA EMIF and the external memory tool kit.

Because the HPS EMIF controller does not support the external memory interface toolkit, verify that the memory interface is operational using the non-HPS memory controller first. Create a design that instantiates the FPGA memory controller and routes it to the same I/O that the HPS EMIF uses.

For more information, refer to the following documentation:

- *Intel Agilex External Memory Interface Pin Information*
- *Intel Agilex FPGA External Memory Interface Overview*

Related Information

- Intel Agilex External Memory Interface Pin Information
- Intel Agilex FPGA External Memory Interface Overview

5.3.3. FPGA EMIF Design Considerations

Table 42. FPGA EMIF Checklist

Number	Done?	Checklist
1		Use the External Memory Interfaces Intel Agilex FPGA IP core for each memory interface, and follow connection guidelines and restrictions in the appropriate documentation.
2		For a given sub-bank, most memory pins are tied to a dedicated location. Refer to *Intel Agilex External Memory Interface Pin Information* to determine available pin usage for EMIF interfaces and the *Intel Agilex Device Family Pin Connection Guidelines* for pin assignments.
3		Generate the External Memory Interfaces Intel Calibration IP and connect it to all the EMIF interfaces located in the same I/O row.

Intel Agilex devices provide an efficient architecture to quickly and easily fit wide external memory interfaces with their small modular I/O banks. The Intel Agilex FPGA can support DDR external memory on any I/O banks located on the top or bottom I/O row. A memory interface can occupy one or more sub-banks. When multiple sub-banks are needed, the sub-banks must be consecutive.

The data strobe DQS and data DQ pin locations are fixed in Intel Agilex 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.

Address and command pins within the address/command bank must follow a fixed pin-out scheme, as defined in the `<variation_name>_readme.txt` file generated with your IP core. The pin-out scheme varies according to the topology of the memory interface. The pin-out scheme is a hardware requirement that you must follow. Some schemes require three lanes to implement address and command pins, while others require four lanes.

The self-calibrating External Memory Interfaces IP core is optimized to take advantage of the Intel Agilex I/O structure. The External Memory Interfaces IP core allows you to set external memory interface features and helps set up the physical interface (PHY)

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

In Intel Agilex devices, the calibration IP is instantiated separately from the EMIF IP core. Every EMIF IP core needs to be connected to the calibration IP. You can only have one calibration IP in an I/O row. If you have multiple EMIF IP core located in the same I/O row, connect all the interfaces in the row to the same calibration IP.
The following checklist supplements the restrictions found in the EMIF user guide.

Table 43. Restrictions for FPGA EMIF Pin

Number	Done?	Checklist
1		All the 96 pins in a given bank (2 sub-banks) share the same voltage level.
2		Unused pins in I/O lane of used data bank or address/command bank of EMIF interface are not permitted as GPIO signals.
3		Arbitrary placement of data mask pins within data lanes is not permitted. Pin index 6 must be used as data mask pin if DM/RDI/WDBI is enabled.
4		True LVDS input clock for PLL reference clock is no longer supported. Intel recommends that every external memory interface to have its own PLL reference clock source. For more information about clock and voltage, refer to the Intel Agilex Device Data Sheet.
5		Every EMIF interface must have its own RZQ pin and must be placed in Lane 2, pin index 2 in the address/command tiles

Table 44. Recommended Board Guideline for Initial Board Bring Up

Number	Done?	Checklist
1		Perform board simulation to confirm adequate margin on address/command and data path.
2		If you are using DIMM, connect every signal from the FPGA to the DIMM if the design does not use it (for example: wider address width, all CS/CKE/ODT signals)
3		Have probe points for voltage rails, address/command channel signals and one data lane.
4		Use a programmable reference clock generator for EMIF to support multiple operating frequency
5		Leave adequate clearance for socket/cooling solution, and logic analyzer interfaces on the DIMM.

The guidelines above ensure the board is designed with adequate margin and allow easy probing of critical signals and stability of voltage rails during debug. Ability to change the reference clock generator allows you to test the interface for multiple operating frequency. If the interface works at a lower speed, the interface is correctly pinned out and functional.

Related Information

- Intel Agilex External Memory Interface Pin Information
- Intel Agilex Device Data Sheet
- Intel Agilex Device Family Pin Connection Guidelines

5.4. Nios II

The Nios II processor supports all Intel FPGA and SoC families. A Nios II processor system is equivalent to a microcontroller or "computer on a chip" that includes a processor and a combination of peripherals and memory on a single chip. A Nios II processor system consists of a Nios II processor core, a set of on-chip peripherals, on-chip memory, and interfaces to off-chip memory, all implemented on a single Intel FPGA device. Like a microcontroller family, all Nios II processor systems use a consistent instruction set and programming model.

For more information, refer to the *Nios II Processor Support web page*.

Related Information

Nios II Processor Support web page

5.5. Transceiver Planning

There are four types of transceiver tiles available in Intel Agilex FPGAs:

- E-Tile: General Purpose Transceiver
- P-Tile: PCIe* Gen4 Transceiver
- F-Tile: General Purpose and PCIe Gen4 Transceiver
- R-Tile: PCIe Gen5 and Compute Express Link (CXL)

Note: **Key:** GPIO (LVDS) / E-Tile 28.9G (58G) / P-Tile Gen4 (16G_PCIe) **Example:** If an entry in the table below contains 576(288)/24(12)/16, it means that 576 GPIO of which 288 are LVDS; twenty-four 28.9 NRZ channels and twelve 58G PAM4 channels; sixteen up to 16G/lane PCIe

Table 45. Intel Agilex F-Series FPGAs with P-Tile and E-Tile Package Options and I/O Pins

Intel Agilex F-Series Device Names	R2068A [5]	R2486A [6]	R2486B [7]
AGF004	—	—	—
AGF006	—	—	—
AGF008	576(288)/24(12)/16	—	—
AGF012	576(288)/24(12)/16	768(384)/16(8)/16	768(384)/24(12)/16
AGF014	576(288)/24(12)/16	768(384)/16(8)/16	768(384)/24(12)/16
AGF022	—	—	768(384)/24(12)/16
AGF027	—	—	768(384)/24(12)/16

Note: R2486A and R2486B are not package compatible or migratable.

[5] (E-Tile + P-Tile) (52 mm x 37.5 mm, Hex 1.0 mm pitch)

[6] (E-Tile + P-Tile) (55 mm x 42.5 mm, Hex 1.0 mm pitch)

[7] (E-Tile + P-Tile) (55 mm x 42.5 mm, Hex 1.0 mm pitch)

For the R2486A package E-tile, the channel bondout uses all 16 channels that have access to the Ethernet Hard IP (EHIP)s. The 16 channels that have access to EHIPs are channels:

- 0 - 3
- 8 - 15
- 20 - 23

Table 46. Available E-Tile Transceiver Channels in Intel Agilex FPGA Devices

Intel Agilex F-Series Device Names	Number of E-Tile Transceiver Channels	Available E-Tile Transceiver Channel Locations
AGF 004	—	—
AGF 006	—	—
AGF 008	24	0 through 23
AGF 012	16 or 24	16 channels: 0,1,2,3,8,9,10,11,12,13,14,15,20,21,22,23 24 channels: 0 through 23
AGF 014	16 or 24	16 channels: 0,1,2,3,8,9,10,11,12,13,14,15,20,21,22,23 24 channels: 0 through 23
AGF 022	24	0 through 23
AGF 027	24	0 through 23

Related Information

- Intel Agilex FPGA Advanced Information Brief (Device Overview)
- E-Tile Transceiver PHY User Guide

5.6. Reconfiguration

Table 47. Reconfiguration Checklist

Number	Done?	Checklist Item
1		Consider the reconfiguration feature for your board development.

Intel Agilex devices allow you to easily modify your transceivers and FPGA-core while other portions of your design are still running by using dynamic reconfiguration and partial reconfiguration, respectively.

Intel Agilex devices allow you to dynamically reconfigure different portions of the transceivers for different protocols, data rates, and PMA settings without powering down any part of the device or interrupting adjacent transceiver channels. This feature becomes available in a future release of the Intel Quartus Prime software.

If you are interested in using partial reconfiguration, contact your local Intel representatives for support.

Related Information

- Intel Agilex Configuration User Guide
- Intel Quartus Prime Pro Edition User Guide: Partial Reconfiguration

5.7. Design Entry Revision History

Table 48. Design Entry Revision History

Document Version	Changes
2019.09.30	Initial release

6. Board and Software Considerations

6.1. 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 Intel Quartus Prime software. Early planning allows the FPGA team to provide early information to PCB board and system designers.

6.1.1. SmartVID

Table 49. SmartVID Checklist

Number	Done?	Checklist Item
1		Is voltage regulator for VCC/VCCP PMBus compliant?
2		Is the PWRMGT_SDA, PWRMGT_SCL, PWRMGT_ALERT (for Slave mode) connected with a 1.8V I/O standard?

The Intel Agilex device is using the SmartVID feature, which needs a voltage regulator that is PMBus compliant to provide power to VCC/VCCP. Besides, all the PWRMGT_SDA, PWRMGT_SCL, PWRMGT_ALERT (for Slave mode) signals must be connected with a 1.8V I/O standard.

6.1.2. Early Power Estimation

Table 50. 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 is used in the design, as well as the operating frequency, toggle rates, and environmental considerations. You can use the tool to obtain thermal design parameters, with which you can perform detailed thermal simulation and cooling solution 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 Intel Quartus Prime software to provide input to the spreadsheet.

The EPE spreadsheet includes the Import Data macro, which parses the information in the Intel 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 Intel 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 Intel 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.

Note: To obtain the EPE tool, contact your local sales representative.

Related Information

Intel Agilex Power Management User Guide

6.1.3. Thermal Management and Design

Table 51. Temperature Design Checklist

Number	Done?	Checklist Item
1		Obtain thermal design power and thermal parameters from EPE.
2		Perform thermal simulation to determine a proper cooling solution.

The Intel Agilex device is a multi-chip module, depending on package configuration and design information, power distribution on all dies can be quite different. This feature makes the Intel Agilex device thermal characteristics design dependent. EPE takes into consideration of your design input, and generates unique thermal parameters for your design in the Thermal Page. You can get power consumption for each die, thermal resistance for all dies (Ψ_{JC}), cooling solution requirement (Ψ_{CA}), and maximum allowed package case temperature (T_{case}).

The thermal analysis of the Intel Agilex device requires you use a Compact Thermal Model (contact your local Intel representatives to obtain the model) and perform simulation in a Computational Fluid Dynamics (CFD) tool. The result of the CFD analysis gives the T_{case} which should be lower than the required value in the EPE Thermal Page. With the simulated T_{case}, Ψ_{JC}, and total package power, you can obtain

the actual junction temperature T_j, which needs to stay below your requirement, for example, 95°C. You can adjust your cooling solution (heat sink design, airflow, and so on) to optimize your thermal design.

6.1.4. Temperature Sensing for Thermal Management

Table 52. Temperature Sensing Checklist

Number	Done?	Checklist Item
1		Set up the temperature sensing diode (TSD) in your design to measure the device junction temperature for thermal management.
2		Include offset values form EPE calculation to TSD reading.

Intel Agilex devices offers local and remote temperature sensing capabilities.

You can measure the junction temperature, T_J, with the following methods:

- Using local Temperature Sensor by instantiating the core.
- Using external thermal diode designed to interface with third-party sensor chip. Ensure the third-party sensor chip matches the external TSD specifications as documented in the *Intel Agilex Device Data Sheet*.

Monitoring the actual junction temperature is crucial for thermal management. Intel Agilex devices include TSD on each die with embedded analog-to-digital converter (ADC) circuitry. You can access the digital temperature readout through the Temperature Sensor IP core.

The Intel Agilex TSD can self-monitor the device junction temperature and can be used with external circuitry for activities such as controlling air flow to the FPGA. This requires including the TSD circuitry by instantiating a Temperature Sensor IP core.

The flexibility in Intel Agilex design can lead to non-uniform power distribution on transceiver dies. Therefore, the hotspot on the transceiver dies may not necessarily be at the TSD location. This results in a temperature difference between TSD readout and real junction temperature. EPE calculates this difference and reports an offset value for each TSD. You need to add the offset values to your TSD reading to get the actual junction temperature.

For more information about temperature sensor details, refer to *Intel Agilex Device Data Sheet*.

Related Information

- Intel Agilex Device Data Sheet
- Intel Agilex Power Management User Guide

6.1.5. Voltage Sensor

Table 53. Voltage Sensor Checklist

Number	Done?	Checklist Item
1		Determine if you need to use the voltage sensor.

Intel Agilex devices have an on-chip voltage sensor. The sensor provides a 7-bit digital representation of the analog signal being observed. This feature can be used for live monitoring of critical on-chip power supplies and external analog voltage.

For more information about voltage sensor details, refer to *Intel Agilex Power Management User Guide*.

Related Information

Intel Agilex Power Management User Guide

6.1.6. Device Power-Up

Table 54. Device Power-Up Checklist

Number	Done?	Checklist Item
1		Design board for power-up: All Intel Agilex GPIO pins are tri-stated until the device is configured and configuration pins drive out. The transceiver pins are at high impedance before the device periphery could get programmed. Once the periphery is programmed, the termination and V_{cm} are set immediately after transceiver calibration is complete.
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.
5		Pull nSTATUS pin high to VCCIO_SDM. Ensure no external component to drive nSTATUS low during power up.

The minimum current requirement for the power-on-reset (POR) supplies must be available during device power-up.

The Intel Agilex device has Power-On-Reset circuitry, which keeps the device in a reset state until the power supply outputs are within the recommended operating range. The device must reach the recommended operating range within the maximum power supply ramp time. If the ramp time is not met, the device I/O pins and programming registers remain tri-stated and device configuration fails. For the Intel Agilex device to exit POR, you must power the V_{CCBAT} power supply even if you do not use the volatile key.

In Intel Agilex devices, when the pin-selectable option (MSEL) is set to non FAST QSPI mode, a typical POR time setting of 4 ms or 100 ms is available.

Intel Agilex devices have power-up sequencing requirements. You should consider the power-up timing and power-down timing for each rail in order to meet the power sequencing requirements.

Intel uses GND as a reference for 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 Intel device.

6.1.7. Power Pin Connections and Power Supplies

Table 55. Power Pin Connections and Power Supplies Checklist

Number	Done?	Checklist Item
1		Connect all power pins correctly as specified in the *Intel Agilex 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 *Intel Agilex Device Family Pin Connection Guidelines*.

Intel Agilex devices require various voltage supplies depending on your design requirements.

Intel Agilex 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.

The VREFP_ADC pin is not a power supply pin. It provides the reference voltage for the ADC for the voltage sensor. For better voltage sensor performance, connect the VREFP_ADC pin to an external reference 1.25 V source. Connecting the VREFP_ADC pin to GND actives an on-chip reference source.

Related Information

Intel Agilex Device Family Pin Connection Guidelines

6.1.7.1. Decoupling Capacitors

Table 56. 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.

Intel Agilex devices include on-die and on package 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.

6.1.7.2. PLL Board Design Guidelines

Table 57. 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.

6.1.7.3. Transceiver Board Design Guidelines

Table 58. Transceiver Board Design Guidelines Checklist

Number	Done?	Checklist Item
1		Review the transceiver board design guidelines when designing your board.

6.1.8. Planning for Device Configuration

Table 59. Planning for Device Configuration Checklist

Number	Done?	Checklist Item
1		Consider whether you require multiple configuration schemes.
2		Ensure that you have OSC_CLK_1 and REFCLK external clocks for Transceivers and CLK for EMIF.
3		Follow the configuration guidelines and additional clock requirements if your design is using PCIe, transceiver channels, HPS, High Bandwidth Memory (HBM2) IP core, or SmartVID. Refer to the *Intel Agilex Configuration User Guide* and *Intel Agilex Power Management User Guide* for the guidelines.
4		Intel strongly recommends using the Intel Agilex Reset Release IP in your design to provide a known initialized state for your logic to begin operation. The Reset Release IP is available in the Intel Quartus Prime software version 19.1 and later. Refer to the *Intel Agilex Configuration User Guide* for the guidelines.

Intel Agilex devices are based on SRAM cells. You must download configuration data to the Intel Agilex 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, Intel Agilex devices offer advanced configuration features, depending on your configuration scheme. Intel Agilex devices also include optional configuration pins and a reconfiguration option that you should choose early in the design process (and set up in the Intel Quartus Prime software), so you have all the information required for your board and system design.

Related Information

- Intel Agilex Configuration User Guide

- Intel Agilex Power Management User Guide

6.1.8.1. Configuration Scheme Selection

Table 60. Configuration Scheme Selection Checklist

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

Intel Agilex devices offer several configuration schemes.

You can enable any specific configuration scheme by driving the Intel Agilex device MSEL pins to specific values on the board.

Active Serial (AS) configuration scheme uses a serial configuration device, JTAG configuration scheme uses a download cable, and Avalon Streaming (AvST) configuration scheme uses an external controller (for example, MAX (MAX II, MAX V, Intel MAX 10) devices or a microcontroller).

6.1.8.1.1. Serial Configuration Devices

Table 61. 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.

Quad SPI flash devices are used as serial configuration devices in the AS configuration scheme.

Serial configuration devices can be programmed using a Intel FPGA Download Cable II with the Intel 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 Intel Agilex FPGA as a bridge between the JTAG interface and the configuration device, allowing both devices to use the same JTAG interface.

Programming the Quad SPI flash device from JTAG using the Intel Agilex FPGA as a bridge is slower than using the standard AS interface.

6.1.8.1.2. Download Cables

Table 62. Download Cables Checklist

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

The Intel Quartus Prime programmer supports configuration of the Intel Agilex 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.

6.1.8.2. Configuration Features

Table 63. Configuration Features Checklist

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

This section describes Intel Agilex device configuration features and how they affect your design process.

Configuration Bitstream Compression

Configuration bitstream compression is always enabled in Intel Agilex device configuration. The Intel Quartus Prime software generates configuration files with compressed configuration data. This compressed file reduces the storage requirements in the configuration device or flash memory, and decreases the time required to transmit the configuration bitstream to the Intel Agilex device.

Due to compressed configuration bitstream, passive configuration schemes for example Avalon-ST ×8, ×16, and ×32 may require the external configuration host to monitor the AVST_READY signal and pause sending configuration data when the AVST_READY low signal is detected.

SEU Mitigation

Dedicated circuitry is built into Intel Agilex devices 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 an Intel Agilex device is correct and alerts the system to a configuration error.

When using the SEU mitigation features, an SDM pin is used to implement the SEU_ERROR function. This pin flags errors for your system to take appropriate actions. Prior to compiling your design, enable the SEU_ERROR function and select an unused SDM pin to implement the SEU_ERROR function in the Intel Quartus Prime software.

RSU

RSU implements device reconfiguration using dedicated RSU circuitry available in all Intel Agilex devices.

For more information, refer to *Intel Agilex Configuration User Guide*.

Related Information

Intel Agilex Configuration User Guide

6.1.8.3. Intel Quartus Prime Configuration Settings

Table 64. Intel Quartus Prime Configuration Settings Checklist

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

There are several configuration options that you can set in the Intel Quartus Prime Pro Edition 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.

6.1.8.3.1. Optional Configuration Pins

Table 65. 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:

- OSC_CLK_1—Must be connected to a 25 MHz, 100 MHz, or 125 MHz source if used.
- CONF_DONE
- INIT_DONE
- CVP_CONFDONE
- SEU_ERROR
- HPS_COLD_nRESET
- Direct to Factory Image
- nCATTRIP

Note: Intel Agilex devices use OSC_CLK_1 pin as the reference clock for transceiver calibration. You must provide a stable and free running clock input at this pin. For more guidance on configuration pins, refer to the *Intel Agilex Device Family Pin Connection Guidelines*.

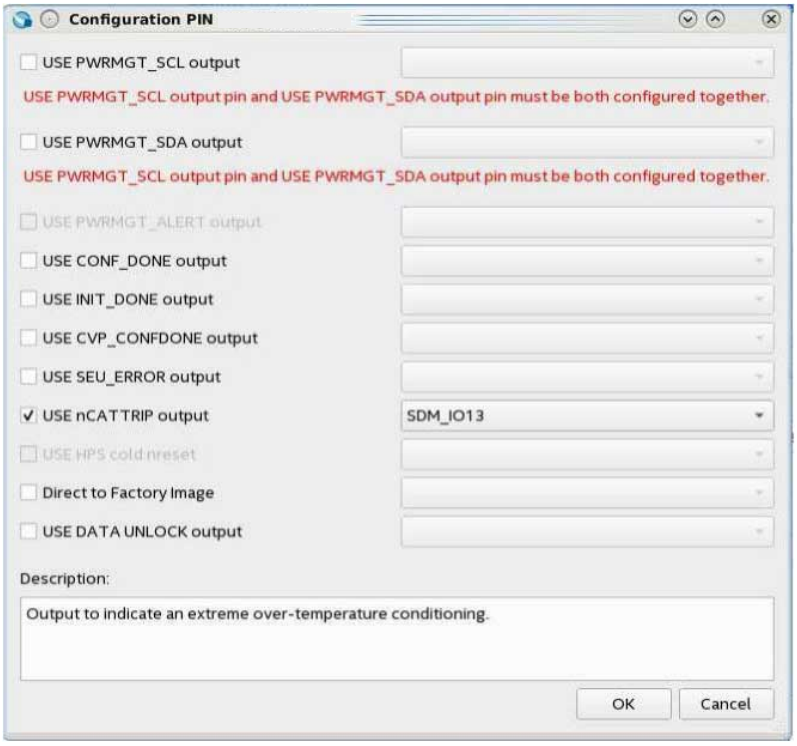
nCATTRIP

The Catastrophic Trip (nCATTRIP) is an optional signal assignable to any unused SDM_IO pin. When enabled, the nCATTRIP signal asserts when core temperature is greater than 125° C.

Attention: When nCATTRIP asserts, you must immediately power down the FPGA to avoid permanent damage to the device.

To enable the nCATTRIP output, select "USE nCATTRIP output" in the **Configuration PIN** GUI and assign the appropriate SDM I/O from the pulldown menu.

Figure 13. nCATTRIP Configuration Pin



For more information about nCATTRIP and other optional configuration pins, refer to the *Secure Device Manager (SDM) Optional Signal Pins* section in *Intel Agilex Device Family Pin Connection Guidelines* and the *Intel Agilex Power Management User Guide*.

GUIDELINE: Ensure that you follow the pull-up recommendations for the nCATTRIP signal to avoid incorrect sampling before you configure your device.

Table 66. Pull-Up Recommendations

nCATTRIP SDM_IO Assignment Options	Internal Pull-Up or Pull-Down	External Pull-Up Recommendation
1-7, 9-15	20kΩ pull-up	Not required
0, 8, 16	20kΩ pull-down	Connect 4.7kΩ to VCCIO_SDM

Related Information

- Intel Agilex Power Management User Guide
- Intel Agilex Device Family Pin Connection Guidelines

6.1.8.3.2. Dual Purpose Configuration Pins

Table 67. Dual Purpose Configuration Pins Checklist

Number	Done?	Checklist Item
1		Plan the dual purpose pins that can function as configuration pins and user I/O pins.

The below configuration pins used for the Avalon-ST ×16 and ×32 configuration schemes can optionally be used as user I/O pins after configuration has completed. Enable the pins to function as dual purpose pins in the Intel Quartus Prime software prior to compilation, if desired.

- AVST_CLK
- AVST_READY
- AVST_VALID
- AVST_DATA[15:0]
- AVST_DATA[31:16]—for Avalon-ST ×32 configuration scheme

6.1.8.4. Configuration Pin Connections

Table 68. 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 schemes.

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.

For more information, refer to *Intel Agilex Device Family Pin Connection Guidelines*.

Related Information

Intel Agilex Device Family Pin Connection Guidelines

6.1.8.4.1. Configuration Pin Voltage Level

Table 69. Configuration Pin Voltage Level Checklist

Number	Done?	Checklist Item
1		Ensure V_{CCIO_SDM} and V_{CCIO} of the configuration pins match the voltage level of the external devices used for configuration.

Configuration pins from the Intel Agilex device connect to external devices, for example the Quad SPI flash configuration device, Avalon-ST host, or SD/MMC flash memories. The voltage level of the configuration pins need to match the voltage level of the devices connected to them. The JTAG and SDM I/Os used as configuration pins are powered by the V_{CCIO_SDM} supply. For Avalon-ST ×32 and ×16 configuration schemes, the AVST_CLK, AVST_READY, AVST_VALID, and AVST_DATA pins are powered by the V_{CCIO} of the I/O bank in which the pins reside in.

6.1.8.4.2. Clock Trace Signal Integrity

Table 70. 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, AS_CLK, AVSTx8_CLK, AVST_CLK, SDMMC_CFG_CCLK, and OSC_CLK_1 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. Make sure to have clock routing as stripline. Keep the clock routing away from any high-speed signals to isolate the clock signals from other signals.

6.1.8.4.3. JTAG Pins

Table 71. 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 72. 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 73. 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 Intel Agilex device are powered up by V_{CCIO_SDM}. In a JTAG chain containing devices with different V_{CCIO} levels, ensure that the V_{IL} max, V_{IH} min, and the maximum V_I specifications of the device JTAG input pins are not violated. Level shifter might be required between devices to meet the voltage specifications of the devices input pin.

JTAG Signal Buffering

Table 74. 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.

6.1.8.4.4. MSEL Configuration Mode Pins

Table 75. MSEL Configuration Mode Pins Checklist

Number	Done?	Checklist Item
1		Connect the SDM pins with MSEL function to select the configuration scheme; do not leave them floating. Pull the pins high or low with external resistors. Do not hardwire the pins to V_{CCIO_SDM} or GND.

Select the configuration scheme by pulling the SDM pins with MSEL function high or low with external resistors. JTAG configuration is always available, regardless of the MSEL settings. The SDM pins with MSEL function are powered by the VCCIO_SDM power supply, and they have internal weak pull-up resistors.

During POR and reconfiguration, the SDM pins with MSEL function must be at LVTTL V_{IL} and V_{IH} levels to be considered as logic low and logic high, respectively. The SDM pins used for MSEL function also have other configuration functions, depending on the configuration schemes used. Do not hardwire the SDM pins with MSEL function to V_{CCIO_SDM} or GND without pull-up or pull-down resistors.

6.1.8.4.5. Other Configuration Pins

Table 76. Other Configuration Pins Checklist

Number	Done?	Checklist Item
1		Use the SDM pins which have multiple configuration functions if power management function is required.
2		When a –V device is used, you must enable the SmartVID connection between the device and the VCC voltage regulator to allow the FPGA to directly control its core voltage requirements. Refer to the *Intel Agilex Device Family Pin Connection Guidelines* and *Intel Agilex Power Management User Guide* for the pin connections and implementation.

Most of the SDM pins have multiple configuration functions, depending on the configuration schemes used. Some SDM pins also have power management functions. If power management function is required, choose the SDM pins which do not need to be used for configuration to implement the power management function.

Connect the SDM pins on your board to the external configuration host or configuration device based on the configuration scheme to be used. If more than one configuration scheme are used, ensure there is no contention between configuration host or configuration devices connected to the SDM pins.

6.2. Board Design Guidelines for Intel Agilex SoC FPGAs

6.2.1. Boundary Scan for HPS

The HPS JTAG interface does not support boundary scan tests (BST). To perform boundary scan testing on HPS I/Os, you must first chain the FPGA JTAG and HPS JTAG internally, and issue the boundary scan from the FPGA JTAG.

GUIDELINE: Chain the FPGA and HPS JTAG interfaces internally to perform boundary scan testing.

To chain the FPGA and HPS JTAG internally, go to Quartus **Device and Pins Options** and select the **Configuration** category. Under the **HPS debug access port (DAP)** settings, choose **SDM Pins** from the drop down option. If boundary scan is not being used, the FPGA JTAG and HPS JTAG interfaces can be used independently. To select HPS Dedicated I/O as the interface for HPS JTAG, select **HPS Pins** from the drop down option instead.

6.2.2. Embedded Software Debugging and Trace

This device has just one JTAG port with FPGA and HPS JTAGs that can be chained together or used independently.

GUIDELINE: Intel recommends to have an available JTAG connection to the board that could be used for development as well as to debug and diagnose field issues.

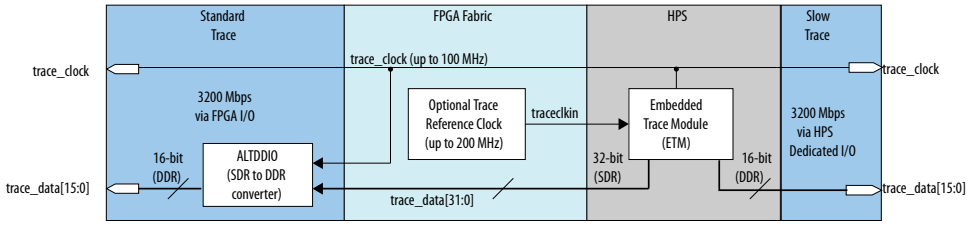
The HPS offers two trace interfaces either through HPS Dedicated I/O or FPGA I/O. The interface through HPS Dedicated I/O is a 16-bit DDR interface that you can use to trace low bandwidth traffic (such as the MPU operating at a low frequency).

To improve the trace bandwidth, you can use the standard trace interface which is a 32-bit single data rate interface to the FPGA. Since trace modules typically expect trace data to be sent at a double data rate you need to convert the single data rate trace data to double data rate.

Intel recommends that you instantiate the DDIO Megawizard IP and set it up in output only mode to perform this conversion. The lowest 16 bits of trace data must be sent off chip first so you connect those bits to the datain_l[15:0] port of the DDIO IP.

Consult your trace vendor's datasheet to determine if the trace bus requires termination. Failure to include termination when the trace vendor requires it can lead to trace data corruption or limit the maximum operating frequency of the interface.

Figure 14. Trace Diagram



The HPS Debug Access Port (DAP) can be accessed through dedicated HPS pins configured as JTAG, or it can be accessed through FPGA JTAG interface pins.

The option to access the HPS JTAG interface through the FPGA JTAG pins is available in the Intel Quartus Prime Pro Edition project.

At power up, the FPGA appears as the first device in the JTAG chain. Once the FPGA is configured with an image for which the HPS JTAG interface is made available to the FPGA JTAG pins, the HPS appears as the first interface in the JTAG chain; and the FPGA appears as the second interface. This requires different connection settings for the FPGA tools, like the Intel Quartus Prime Pro Edition Programmer when it is used at power up and after FPGA configuration.

GUIDELINE: You must have an available JTAG connection to the board that can be used for development as well as to debug and diagnose field issues.

The HPS offers two trace interfaces either through HPS Dedicated I/O or FPGA I/O. The interface through HPS Dedicated I/O is a slow trace interface that you can use to trace low bandwidth traffic (such as the MPU operating at a low frequency).

6.3. Pin Connection Considerations for Board Design

When designing the interfaces to the Intel Agilex device, various factors can affect the PCB design.

6.3.1. Board-Related Intel Quartus Prime Settings

Table 77. Board-Related Intel Quartus Prime Settings Checklist

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

The Intel 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 Intel Quartus Prime project is created, and plan for the functionality during board design.

6.3.1.1. Unused Pins

Table 78. 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 Intel Quartus Prime software-generated `.pin` file. Do not connect RESERVED pins.

You can specify the state of unused pins in the Intel 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.

6.3.2. 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.

6.3.2.1. High-Speed Board Design

Table 79. High-Speed Board Design Checklist

Number	Done?	Checklist Item
1		Refer to the Board Design Resource Center.

If your design has high-speed signals, especially with Intel Agilex GX/SX device high-speed transceivers, the board design has a major impact on the signal integrity in the system.

6.3.2.2. Voltage Reference Pins

Table 80. 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.

6.3.2.3. Simultaneous Switching Noise

Table 81. 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.

6.3.2.4. I/O Termination

Table 82. 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.

Intel Agilex on-chip series and parallel 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. Intel Agilex devices provide an optional on-chip differential resistor when using 1.5V True Differential Signaling.

6.3.3. Board-Level Simulation and Advanced I/O Timing Analysis

Table 83. 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 Intel 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 Intel 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 Intel Quartus Prime software. When **Enable Advanced I/O Timing** is turned on (**Timing Analyzer** page in the **Settings** dialog box), the 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.

6.4. Board Considerations Revision History

Table 84. Board Considerations Revision History

Document Version	Changes
2019.09.30	Initial release

7. 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 Intel Quartus Prime Fitter then performs placement and routing to implement the design elements in specific device resources. If required, you can use the Intel 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. This section provides guidelines for these stages of the compilation flow.

7.1. Selecting a Synthesis Tool

Table 85. Selecting a Synthesis Tool Checklist

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

The Intel 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 Intel 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 Intel Quartus Prime software to get accurate timing analysis and logic utilization results.

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

7.2. Device Resource Utilization Reports

Table 86. Device Resource Utilization Reports Checklist

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

After compilation in the Intel 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.

For Intel Agilex devices, low logic utilization does not have the lowest ALM utilization possible. In addition, a design that is reported as close to 100% full might 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.

More detailed resource information is available by viewing the reports under **Fitter ➤ Place** 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 Intel Quartus Prime integrated synthesis, the reports under **Analysis & Synthesis ➤ Partition <partition_name> ➤ Optimization Results** 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.

7.3. Intel Quartus Prime Messages

Table 87. Intel Quartus Prime Messages Checklist

Number	Done?	Checklist Item
1		Review all Intel Quartus Prime messages, critical warning, 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 Intel 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.

For more information about message and message suppression, refer to the *Viewing Project Messages* section in the *Intel Quartus Prime Pro Edition User Guide: Getting Started*.

Related Information

Intel Quartus Prime Pro Edition User Guide: Getting Started

7.4. Timing Constraints and Analysis

Table 88. 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 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 Intel Agilex 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 Intel 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 Intel Quartus Prime software includes the Intel Quartus Prime 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.

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 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.

For more information about timing constraint, refer to *Intel Quartus Prime Pro Edition User Guide: Timing Analyzer*.

Related Information

Intel Quartus Prime Pro Edition User Guide: Timing Analyzer

7.4.1. Recommended Timing Optimization and Analysis Assignments

Table 89. 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_clock_uncertainty` to automatically apply inter-clock, intra-clock, and I/O interface uncertainties.
5		Use `check_timing` to generate a report on any problem with the design or applied constraints, including missing constraints.
6		Use `set_false_path` or `set_clock_groups` for asynchronous paths.

These assignments and settings are important for large designs such as those in Intel Agilex 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 Timing Analyzer `.sdc` constraints file, apply the recommended constraints to your design.

7.5. Area and Timing Optimization

Table 90. Area and Timing Optimization Checklist

Number	Done?	Checklist Item
1		Run Fitter (Plan) if you want timing estimates before running a full compilation.
2		Use Intel Quartus Prime optimization features to achieve timing closure or improve the resource utilization.
3		Use the Timing Optimization Advisors to suggest optimization settings.

This section highlights some of the features offered in the Intel 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 run Fitter (Plan) to estimate your design's timing results before the software performs full placement and routing. Click **Processing ➤ Start ➤ Start Fitter (Plan)** 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 by selecting **High Performance Effort** or **Superior Performance** Optimization Mode in

the **Compiler Settings**. These optimization modes turn on the **Advanced Physical Synthesis** option under the **Advanced Fitter Settings**. 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 II (DSE II) is a utility that automates the process to find optimal project settings for resource, performance, or power optimization goals. DSE II attempts multiple seeds to identify one that meets your requirements. The **Exploration Panel ➤ Exploration mode** allows you a predefine exploration space to target design performance, area of improvements, or power reduction with multiple compilations.

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

For more information about the topics listed below, refer to the various sections listed below in the *Intel Quartus Prime Pro Edition User Guide: Design Optimization*.

Table 91.

Chapter	Optimization Areas
1	Design Space Explorer II
3	Netlist Optimizations and Physical Synthesis
4	Area Optimization
5	Timing Closure and Optimization
1 and 5	Power Optimization

Related Information

Intel Quartus Prime Pro Edition User Guide: Design Optimization

7.6. Preserving Performance and Reducing Compilation Time

Table 92. 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 Intel 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.

7.7. Designing with Intel Hyperflex™

Table 93. Designing with Intel Hyperflex™ Checklist

Number	Done?	Checklist Item
1		Use Intel Hyperflex™ feature to optimize your design and achieve enhanced performance.

Intel Hyperflex core architecture adds registers to both the interconnect routing and the inputs of all major functional blocks in the FPGA. These added registers, called Hyper-Registers, are different from conventional registers. Conventional registers are present only in the adaptive logic modules (ALMs). Hyper-Registers can help to achieve significant core performance improvement.

To achieve this enhanced performance, you must optimize your designs using the following steps:

1. Hyper-Retiming
2. Hyper-Pipelining
3. Hyper-Optimization

For more information about high performance design, refer to the Intel FPGA Technical Training website.

7.8. Simulation

Table 94. Simulation Checklist

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

The Intel 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 Intel 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.

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

For a list of simulation tools supporting the Intel Quartus Prime Pro Edition software, refer to *Intel Quartus Prime Pro Edition Version 19.2 Software and Device Support Release Notes*.

Related Information

Intel Quartus Prime Pro Edition Version 19.2 Software and Device Support Release Notes

7.9. Power Analysis

Table 95. 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 Intel 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.

7.10. Power Optimization

Intel Agilex devices utilize advanced process and circuit techniques, along with major circuit and architecture innovations, to minimize power and deliver high performance.

To reduce dynamic power consumption in Intel Agilex devices, you can use various design and software techniques to optimize your design.

Power optimization in the Intel 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.

For more information about power optimization, refer to the *Intel Agilex Power Management User Guide.*

Related Information

Intel Agilex Power Management User Guide

7.10.1. Device and Design Power Optimization Techniques

Table 96. Device and Design Power Optimization Techniques Checklist

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

7.10.1.1. Device Speed Grade

Table 97. Device Speed Grade Checklist

Number	Done?	Checklist Item
1		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.

7.10.1.2. Clock Power Management

Table 98. 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 Intel 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 Intel Quartus Prime software automatically promotes register-level clock enable signals to the LAB level.

7.10.1.3. Memory Power Reduction

Table 99. 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.

7.10.1.4. I/O Power Guidelines

Table 100. 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.

7.10.2. Intel Quartus Prime Power Optimization Techniques

Table 101. Intel Quartus Prime Power Optimization Techniques Checklist

Number	Done?	Checklist Item
1		Review recommended design techniques and Intel Quartus Prime options to optimize power consumption.

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

Optimizing your design for area also saves power because fewer logic blocks are used; therefore, there is typically less switching activity. Improving your design source code to optimize for performance can also reduce power usage. You can use the DSE and Power Optimization Advisor to provide additional suggestions to reduce power.

7.10.2.1. Power Optimization Advisor

Table 102. Power Optimization Advisor Checklist

Number	Done?	Checklist Item
1		Use the Power Optimization Advisor to suggest optimization settings.

The Intel 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.

7.11. Design Implementation, Analysis, Optimization, and Verification Revision History

Table 103. Design Implementation, Analysis, Optimization, and Verification Revision History

Document Version	Changes
2019.09.30	Initial release

8. Debugging

8.1. On-Chip Debug Overview

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, Intel 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 tools.

8.1.1. Planning Guidelines for Debugging Tools

Table 104. Planning Guidelines for Debugging Tools Checklist

Number	Done?	Checklist Item
1		Select on-chip debugging schemes 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 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. Ensure that the JTAG signals have a clean timing.
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 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, 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, Intel Quartus Prime project, and design.

For more information about debug tools, please refer to *Intel Quartus Prime Pro Edition User Guide: Debug Tools*.

Related Information

Intel Quartus Prime Pro Edition User Guide: Debug Tools

8.2. On-Chip Debugging Tools

The Intel 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. Signal Tap may affect routing of the original design.
- 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 custom 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 Intel FPGA 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** Intel FPGA 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.

- EMIF Debug Toolkit—Tcl-based graphical user interface communicating via a JTAG connection to enable external memory interface on the circuit board to retrieve calibration status and debug information. The Driver Margining feature of the tool kit allows you to measure margins on your memory interface using a driver with arbitrary traffic patterns. Tcl-based graphical user interface that provides access to memory calibration data gathered by the Nios II sequencer, via a JTAG connection. The Toolkit allows you to mask ranks for calibration, and to request recalibration of the interface. The Driver Margining feature of the toolkit allows you to measure margins on the memory interface using a driver with arbitrary traffic patterns. The EMIF Toolkit can communicate with several different memory interfaces on the same device, but only one at a time.
- Transceiver Toolkit—Uses System Console technology to help FPGA and board designers validate transceiver link signal integrity real time in a system and improve board bring-up time. Test for bit-error rate (BER) while simultaneously running multiple links at your target data rate to validate your board design with Transceiver Toolkit. Tune transceiver analog settings for optimal link performance while using different test metrics to quantify results. Simultaneously test multiple devices across one or more boards using link tests in the Transceiver Toolkit GUI.
- P-Tile Toolkit—Provides support for an Avalon-MM interface with DMA and is designed to optimize the performance of large-size data transfers. If you want to achieve maximum performance with small-size transfers, Intel recommends the use of an Avalon-ST IP core like the P-Tile Avalon-ST Hard IP for PCIe. P-Tile supports PCI Express* Gen4 in Endpoint, Root Port and TLP Bypass modes.
- SDM Debug Toolkit—The SDM Debug Toolkit provides access to the current status of the Intel Agilex device.

Related Information

Intel Quartus Prime Pro Edition User Guide: Debug Tools

8.3. Debugging Revision History

Table 105. Debugging Revision History

Document Version	Changes
2019.09.30	Initial release

9. Embedded Software Design Guidelines for Intel Agilex SoC FPGAs

9.1. Overview

This chapter covers the design considerations for assembling your software development platform for the Intel Agilex Hard Processor System.

You must follow the provided recommendations to select the components of your software platform that suit the performance, support and time-to-market requirements of your end application.

9.2. Define Software Requirements

Start by defining the software requirements, typically including the use cases that need to be supported and various quality goals such as testability and extensibility. Care must be exercised to ensure all the required features are specified. Errors at this stage can be potentially be costly to remedy later on.

9.3. Define Software Architecture

Define the software architecture, making sure that the software requirements are met by the proposed architecture. Typically the architecture focuses on the high-level view of how the software is organized. Some projects also have another, low-level implementation document, augmenting the architecture document. Errors at this stage can also be very costly to fix later on.

9.4. Selecting Software Tools

This section describes design considerations for selecting various software development tools.

Note: When using a specific Partner OS or RTOS, consult the OS vendor and the OS documentation for any specific tools that are required. Some OS vendors also provide a full set of tools that are recommended to be used with that operating system.

9.4.1. Selecting Software Build Tools

You must decide which software development tools to use, including which version:

- Compiler
- Assembler
- Linker
- Archiver

9.4.2. Selecting Software Debug Tools

You must decide which software debug tools to use and check with the tools provider to ensure support is available for Intel Agilex devices in the desired time frame.

The Arm DS-5* Intel SoC FPGA Edition includes a fully featured Eclipse-based debugging environment. There are also other debugging tool offerings from third party providers such as Lauterbach* T32.

The debug tools require a JTAG connection to the Intel SoC FPGA device. You can achieve a JTAG connection through:

- An embedded Intel FPGA Download Cable II like what is available in the Intel Agilex SoC Development Kit.
- External JTAG hardware similar to what may be required when using the Lauterbach T32 tools.

9.4.3. Selecting Software Trace Tools

Tracing can be very helpful for profiling performance bottlenecks, debugging crash scenarios and debugging complex cases. Tracing can be performed in two ways:

- **Non-real-time**: by storing trace data in system memory (for example, SDRAM) or the embedded trace buffer, then stopping the system, downloading the trace information through JTAG, and analyzing it.
- **Real-time**: by using an external adapter to capture trace data from the trace port. The target board needs to support this scenario.

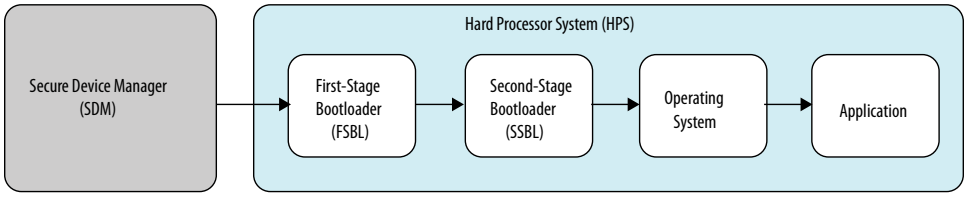
Typically, the debug tools also offer tracing of the embedded software program execution, but external hardware may be required. For example, the Arm DS-5 Intel SoC FPGA Edition provided with the SoC EDS supports both non-real-time and real-time tracing. When used for real-time tracing, an additional external trace unit called Arm DSTREAM is required.

Lauterbach T32 also requires external hardware for real-time tracing.

9.4.4. Choosing the Bootloader Software

The typical Intel Agilex SoC HPS boot flow is depicted in the figure below:

Figure 15. Typical Intel Agilex SoC Boot Flow



The bootloader software is one of the most important components of your software development platform. The bootloader initializes the system and then loads and passes control to the next boot image which is either an operating system or a bare-metal application.

The Intel Agilex SoC bootloader software is split into two different stages:

- First Stage Bootloader (FSBL) – Loaded by the SDM from the FPGA configuration bitstream into the HPS side on-chip memory:
  - Provides essential initial hardware settings to configure the HPS
  - Software features to control the flash and peripheral components of the HPS
  - Utilities to enable early debugging and troubleshooting
- Second Stage Bootloader (SSBL) – Loaded by FSBL into the DDRAM and potentially having significantly more capabilities than FSBL, such as: network access, command line interface and scripting support.

Several bootloaders are in the process of being enabled for Intel Agilex devices:

- **U-Boot Bootloader:** Inherits several features available from the open source community and is popular with Linux OS users. U-Boot bootloader is governed by GPL licensing.
- **UEFI Bootloader:** Feature rich and popular with RTOS users and is governed by an open-source BSD style license.
- **ATF (ARM Trusted Firmware) Bootloader**: Used by the UEFI and provides just the first stage bootloader. It uses a BSD-style license, and could be used to directly load a bare-metal application instead of a SSBL.

GUIDELINE: To select the right bootloader for your software development platform, use the latest version and familiarize yourself with the GPL and open-source BSD licenses and consider which licensing terms best suit your requirements.

A typical HPS system has hundreds of registers that must be set for a given configuration of the MPU subsystem, the network-on-chip interconnect component, the DDRAM memory, flash boot source and peripheral interfaces.

GUIDELINE: Given the amount of initialization settings that are required, it is not recommended to write a bootloader from scratch. The provided bootloader options contain optimum and default configuration settings for various parts of the HPS.

9.4.5. Selecting an Operating System for Your Application

9.4.5.1. Using Linux or RTOS

There are many factors that go into the selection of an operating system for SoC FPGAs including:

- Features of the operating system
- Licensing terms
- Availability of collaborative software projects and frameworks based on the operating system
- Available device drivers and reference software
- In-house legacy code and familiarity with the operating system
- Real time requirements of your system
- Functional safety and other certifications required for your application

To select an appropriate operating system for your application, familiarize yourself with the features and support services offered by the commercial and open source operating systems available for the SoC FPGA. Intel's OS partners' websites are a good source of information you can use to help make your selection. Contact the provider to ensure Intel Agilex support is available in the desired time frame.

Linux is currently being enabled for Intel Agilex devices, with a Yocto Project compatible, Ångström distribution.

Partner OS providers offer board support packages and commercial support for the SoC FPGA devices. The Linux community also offers board support packages and community support for the SoC FPGA devices.

There are several misconceptions when it comes to real time performance of operating systems versus bare-metal applications. For an Arm Cortex* A-class of processor, there are several features that real time operating systems provide that make efficient use of the processor's resources in addition to the facilities provided to manage the run-time application.

You may find that these efficiencies result in sufficient real-time performance for your application, enabling you to inherit a large body of available device drivers, middleware packages, software applications and support services. You must take this into account when selecting an operating system.

9.4.5.2. Using the Bootloader as a Bare-Metal Framework

If your application is relatively simple, and does not require complex features such as multi-core or multi-tasking, one option is to include it in the bootloader.

Including your application in the bootloader has the following advantages:

- Potentially faster boot time
- Access to features already implemented in the bootloader, such as mass storage and networking

9.4.5.3. Using Symmetrical vs. Asymmetrical Multiprocessing (SMP vs. AMP) Modes

The Quad Core Arm Cortex-A53 MPCore* in the Intel Agilex HPS can support both Symmetrical Multi Processing (SMP) and Asymmetrical Multi-processing (AMP) operating modes.

In SMP mode, a single operating system instance controls all four cores. The SMP configuration is supported by a wide variety of operating system manufacturers and is the most common and straightforward configuration mode for multiprocessing.

Linux and commercially developed operating systems offer features that take full advantage of the CPU cores resources and use them in an efficient manner resulting in optimized performance and ease of use. For instance, SMP-enabled operating systems offer the option of setting processor affinity. This means that each task or thread can be assigned to run on a specific core. This feature allows you to better control the workload distribution for each Arm Cortex-A53 core and making the system more responsive as an alternative to AMP.

GUIDELINE: Familiarize yourself with the performance and optimizations available in commercial operating systems to see if an SMP-enabled operating system or RTOS meets your performance and real-time requirements.

In the AMP configuration, up to four different operating systems could run on the four Cortex-A53 cores, which allows more valid combinations. You could also combine AMP and SMP allowing you to have two cores running an SMP and the other two cores running an AMP.

Special Considerations

- Use AMP only if you are familiar with the techniques to manage and schedule processes, handle inter-process communication, synchronize between events, and manage secure processes between the two instances of the operating systems.
- OS providers do not generally offer support for using their operating system in an AMP mode, so a special support agreement is typically needed in this case.
- If you use AMP, it is best to use the virtualization feature of the Cortex-A53 because the Cortex-A53 includes native hardware support for virtualization solving most of the AMP resource sharing issues.

9.5. Driver Considerations

- Determine which IP modules need drivers, including both hardened IPs on the HPS side and soft IPs on the FPGA side.
- Check with the OS provider which IPs are already supported. Typically most of the HPS peripherals are supported, but not all of them.
- Check whether the exact functionality needed is implemented by the available drivers. Typically the most common use cases are supported, but some special ones may not be.
- Decide whether the additional drivers or driver functionality that is required can be implemented in-house or requested from the OS provider or from a 3rd party and proceed accordingly to have them done.

9.6. Develop Application

Develop the end application, according to the defined software architecture and using the build and debug tools which were selected.

9.7. Test and Validate

Test and validate the end application, making sure that all the functional and quality requirements are met.

9.8. Embedded Software Design Guidelines Revision History

Table 106. Embedded Software Design Guidelines Revision History

Document Version	Changes
2019.09.30	Initial release