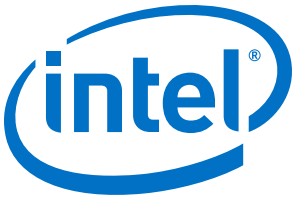


Intel® Agilex™ FPGA External Memory Interface Overview

UG-20220 | 2019.04.02
Latest document on the web: PDF | HTML

Contents

1. Intel® Agilex™ FPGA External Memory Interface Introduction

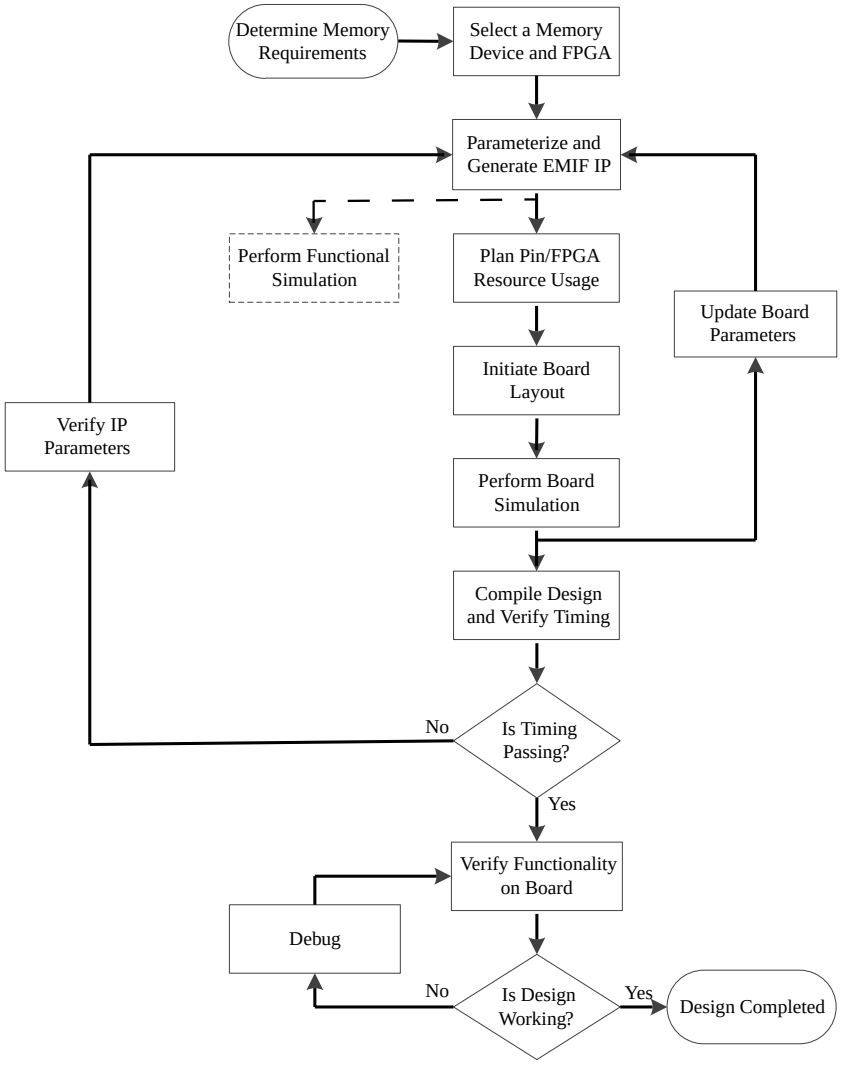
The Intel® Agilex™ external memory interface (EMIF) technology offers fast, efficient, and low-latency connectivity to high speed memory devices.

You can easily implement the external memory interface through the Intel Quartus® Prime software and associated IP. Additional toolkits help you test the external memory interface implementation.

1.1. Intel Agilex FPGA EMIF Introduction

The following figure illustrates the design flow for implementing an Intel Agilex external memory interface.

Figure 1. External Memory Interface Design Flow



1.2. Intel Agilex FPGA EMIF Design Steps

The following table elaborates on the steps of the Intel Agilex external memory interface design flow.

Table 1. External Memory Interface Design Steps

Design Step	Description
Select an FPGA	Not all devices support all memory types and configurations.

continued...

Design Step	Description
	• Intel FPGA Product Selector • External Memory Interface Device Selector • External Memory Interface Spec Estimator
Parameterize the IP	Correct IP parameterization is important for good external memory interface operation.
Generate initial IP and example design	After you have parameterized the EMIF IP, you can generate the IP, along with an optional example design.
Perform functional simulation	Simulation of the external memory interface helps to determine correct operation.
Make pin assignments	Optimal pin placement helps to ensure correct operation.
Perform board simulation	Board simulation helps determine optimal settings for signal integrity, drive strength, as well as sufficient timing margins and eye openings.
Update board parameters in the IP	Refer to board simulation results to help optimize board parameters in the IP.
Verify timing closure	Timing analysis tools help to identify and remedy timing violations.
Verify the design on hardware	
Debug issues with preceding steps	Operational problems can generally be attributed to one of the following: interface configuration, pin/resource planning, signal integrity, or timing. Debug procedures and tools are available to help diagnose hardware issues.

2. Intel Agilex FPGA EMIF IP – Product Architecture

This chapter describes the Intel Agilex FPGA EMIF IP product architecture.

2.1. Intel Agilex EMIF Architecture: Introduction

The Intel Agilex EMIF architecture contains many new hardware features designed to meet the high-speed requirements of emerging memory protocols, while consuming the smallest amount of core logic area and power.

Note: The current version of the External Memory Interfaces Intel Agilex FPGA IP supports the DDR4 memory protocol. Future versions will include support for QDR-IV and RLDRAM 3 protocols.

The following are key hardware features of the Intel Agilex EMIF architecture:

Hard Sequencer

The sequencer employs a hard Nios® II processor, and can perform memory calibration for a wide range of protocols. You can share the sequencer among multiple memory interfaces of the same or different protocols.

Note: You cannot use the hard Nios II processor for any user applications after calibration is complete.

Hard PHY

The PHY circuitry in Intel Agilex devices is hardened in the silicon, which simplifies the challenges of achieving timing closure and minimizing power consumption.

Hard Memory Controller

The hard memory controller reduces latency and minimizes core logic consumption in the external memory interface. The hard memory controller supports the DDR4 memory protocol.

PHY-Only Mode

The PHY-Only option is available if you want to implement your own controller in the FPGA fabric, rather than using the hardened controller in the I/O subsystem or the soft controllers. Control of the PHY is passed to the user after the interface calibrates successfully.

High-Speed PHY Clock Tree

Dedicated high speed PHY clock networks clock the I/O buffers in Intel Agilex EMIF IP. The PHY clock trees exhibit low jitter and low duty cycle distortion, maximizing the data valid window.

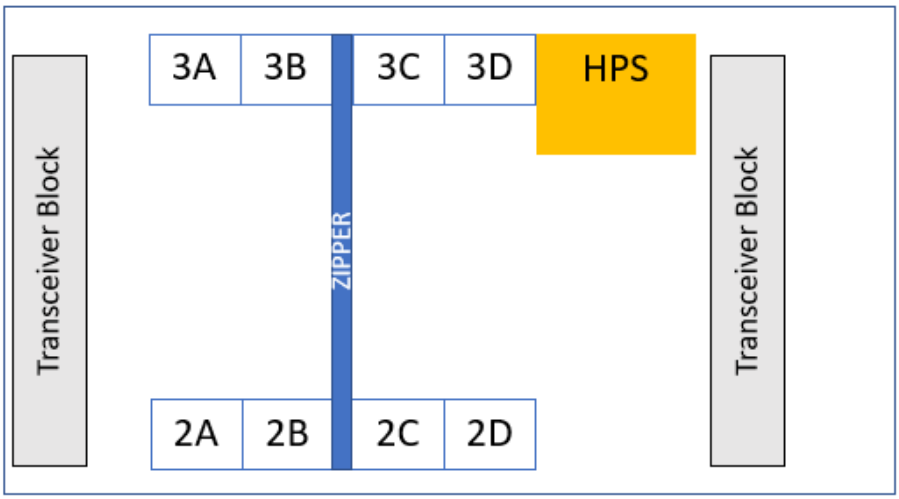
Automatic Clock Phase Alignment

Automatic clock phase alignment circuitry dynamically adjusts the clock phase of core clock networks to match the clock phase of the PHY clock networks. The clock phase alignment circuitry minimizes clock skew that can complicate timing closure in transfers between the FPGA core and the periphery.

2.1.1. Intel Agilex EMIF Architecture: I/O Subsystem

In Intel Agilex devices, the I/O subsystem consists of two rows at the edge of the core.

Figure 2. Intel Agilex I/O Subsystem



The I/O subsystem provides the following features:

- General-purpose I/O registers and I/O buffers
- On-chip termination control (OCT)
- I/O PLLs
  - I/O Bank I/O PLL for external memory interfaces and user logic
  - Fabric-feeding for non-EMIF/non-LVDS SERDES IP applications
- True Differential Signaling
- External memory interface components, as follows:
  - Hard memory controller
  - Hard PHY
  - Hard Nios processor and calibration logic
  - DLL

2.1.2. Intel Agilex EMIF Architecture: I/O SSM

Each I/O row includes one I/O subsystem manager (I/O SSM), which contains a hardened Nios II processor with dedicated memory. The I/O SSM is responsible for calibration of all the EMIFs in the I/O row. The I/O SSM is located immediately to the left of the zipper.

The I/O SSM includes dedicated memory which stores both the calibration algorithm and calibration run-time data. The hardened Nios II processor and the dedicated memory can be used only by an external memory interface, and cannot be employed for any other use. The I/O SSM can interface with soft logic, such as the debug toolkit, via an Avalon-MM bus.

The I/O SSM is clocked by the on-chip configuration network, and therefore does not consume a PLL.

Each EMIF instance must be connected to the I/O SSM through the External Memory Interfaces Calibration IP. The Calibration IP exposes a calibration bus master port, which must be connected to the slave calibration bus port on every EMIF instance.

Only one calibration IP is allowed for each I/O row. All the EMIFs in the same I/O row must be connected to the same calibration I/P. You can specify the number of EMIF interfaces to be connected to the calibration IP when parameterizing the IP. Connect the `emif_calbus` and `emif_calbus_clk` on the calibration IP to the `emif_calbus` and `emif_calbus_clk`, respectively, on the EMIF IP core.

Figure 3. Connectivity Between Calibration IP and Single EMIF Interface

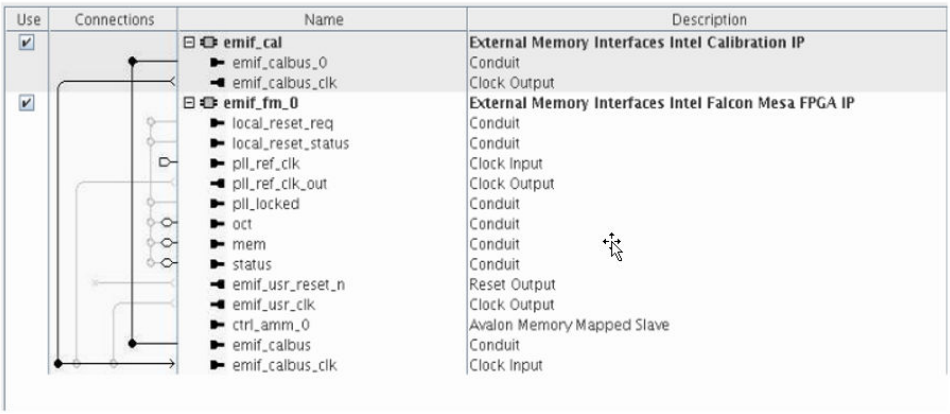
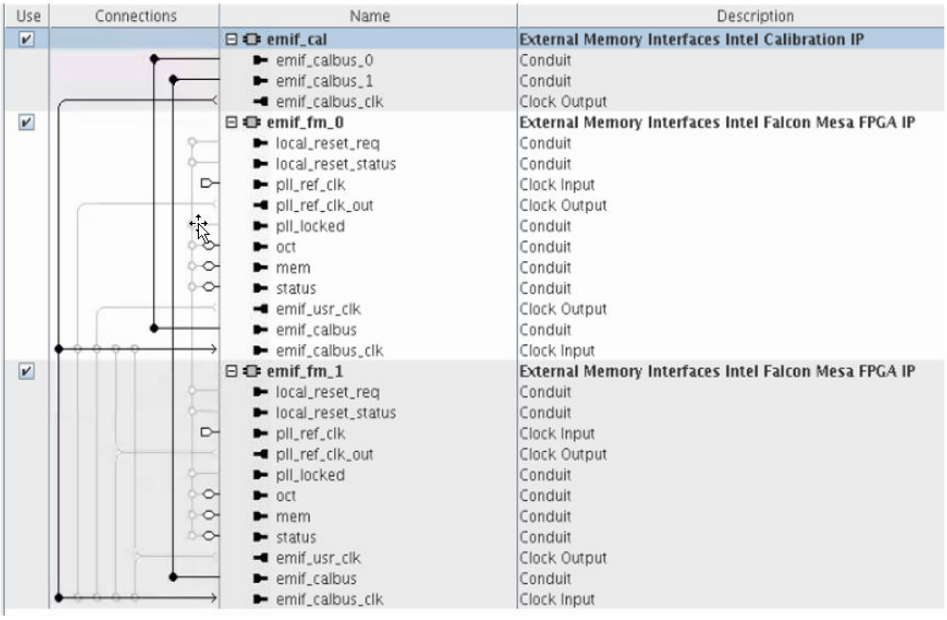


Figure 4. Connectivity Between Calibration IP and Multiple EMIF Interfaces on the Same I/O Row



2.1.3. Intel Agilex EMIF Architecture: I/O Bank

Each I/O row contains up to four I/O banks; the exact number of banks depends on device size and pin package.

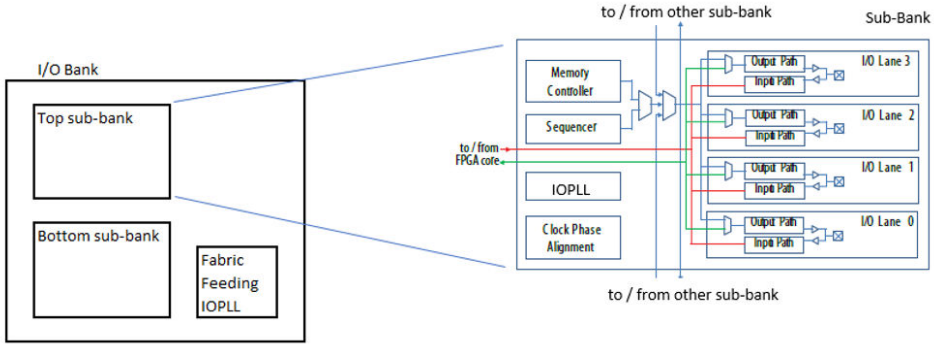
Each I/O bank consists of two sub-banks, and each sub-bank contains the following components:

- Hard memory controller
- Sequencer components
- I/O PLL and PHY clock trees

- DLL
- Input DQS clock trees
- 48 pins, organized into four I/O lanes of 12 pins each

A single I/O sub-bank contains all the hardware needed to build an external memory interface. You can make a wider interface by connecting multiple adjacent sub-banks together.

Figure 5. I/O Bank Architecture in Intel Agilex Devices



Within an I/O bank, the top sub-bank is placed near the edge of the die, and the bottom sub-bank is placed near the FPGA core.

There are interconnects between the sub-banks which chain the sub-banks into a row. The following figures show how I/O lanes in various sub-banks are chained together to form the top and bottom I/O rows in Intel Agilex AGF012 and AGF014 device variants, respectively. These figures represent the top view of the silicon die that corresponds to a reverse view of the device package.

Figure 6. Sub-Bank Ordering in Top I/O Row in Intel Agilex AGF012 and AGF014 devices

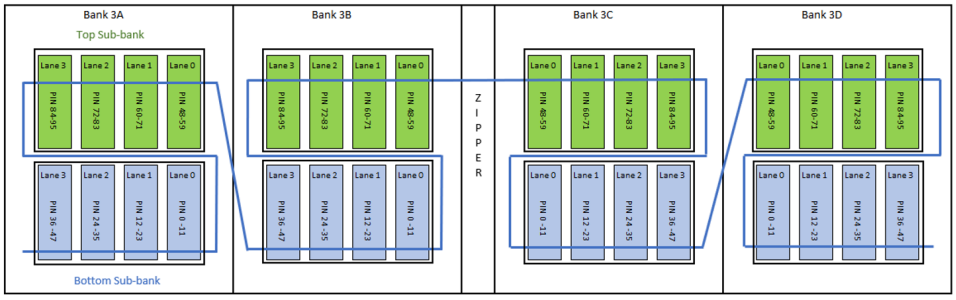
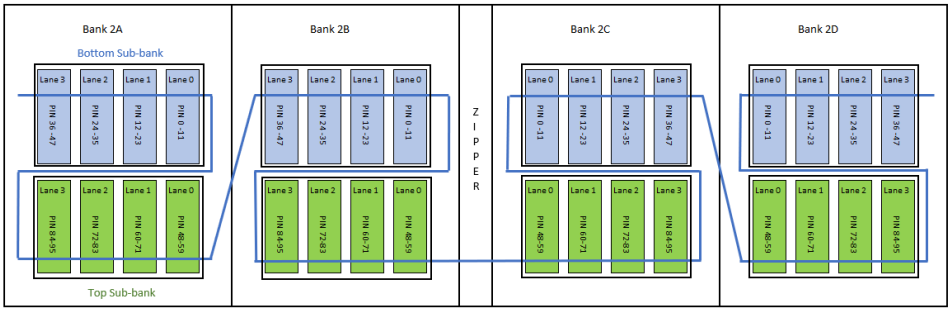


Figure 7. Sub-Bank Ordering in Bottom I/O Row in Intel Agilex AGF012 and AGF014 devices



The two sub-banks within an I/O bank are adjacent to each other, unless any of the sub-banks is not bonded out or partially bonded out. The blue line in the above figures shows the connectivity between the sub-banks.

For example, in the top row in Intel Agilex AGF012 and AGF014 devices (Figure 6):

- The top sub-bank in 3A is adjacent to the bottom sub-bank in 3A and the bottom sub-bank in 3B.
- The top sub-bank in 3B is adjacent to the bottom sub-bank in 3B and the top sub-bank in 3C.
  - The top sub-bank in 3B is adjacent to the top sub-bank in 3C even though there is a zipper block between the two sub-banks.
- The top sub-bank in 3B is not adjacent to the bottom sub-bank in 3A.

You can identify where a pin is located within an I/O bank based on its `Index within I/O Bank` value in the device pinout file.

Zipper Block

The zipper is a block that performs necessary routing adjustments where routing wires cross the zipper.

I/O Sub-Bank Usage

The pins in an I/O bank can serve as address and command pins, data pins, or clock and strobe pins for an external memory interface. You can implement a narrow interface, DDR4 x8 interface, with only a single I/O sub-bank. A wider interface of up to 72 bits can be implemented by configuring multiple adjacent banks in a multi-bank interface.

Every sub-bank includes a hard memory controller which you can configure for DDR4. In a multi-bank interface, only the controller of one sub-bank is active; controllers in the remaining sub-banks are turned off to conserve power.

To use a multi-bank Intel Agilex EMIF interface, you must observe the following rules:

- Designate one sub-bank as the address and command bank.
- The address and command sub-bank must contain all the address and command pins.
- The locations of individual address and command pins within the address and command sub-bank must adhere to the pin map defined in the pin table—regardless of whether you use the hard memory controller or not.
- If you do use the hard memory controller, the address and command sub-bank contains the active hard controller.

All the sub-banks are capable of functioning as the address and command bank. However, for minimal latency, you should select the center-most bank of the interface as the address and command bank.

2.1.4. Intel Agilex EMIF Architecture: I/O Lane

An I/O bank contains two sub-banks. Each sub-bank contains 48 I/O pins, organized into four I/O lanes of 12 pins each. You can identify where a pin is located within an I/O bank based on its `Index within I/O Bank` in the device pinout.

Table 2. Pin Index Mapping

Pin Index	Lane	Sub-bank Location
0-11	0	Bottom
12-23	1	
23-35	2	
36-47	3	
48-59	0	Top
60-71	1	
72-83	2	
84-95	3	

Each I/O lane can implement one x8/x9 read capture group (DQS group), with two pins functioning as the read capture clock/strobe pair (DQS/DQS#), and up to 10 pins functioning as data pins (DQ and DM pins). To implement a x18 group, you can use multiple lanes within the same sub-bank.

It is also possible to implement a pair of x4 groups in a lane. In this case, four pins function as clock/strobe pair, and 8 pins function as data pins. DM is not available for x4 groups. There must be an even number of x4 groups for each interface.

For x4 groups, DQS0 and DQS1 must be placed in the same I/O lane as a pair. Similarly, DQS2 and DQS3 must be paired. In general, DQS(x) and DQS(x+1) must be paired in the same I/O lane.

Table 3. Lanes Used Per Group

Group Size	Number of Lanes Used	Maximum Number of Data Pins per Group
x8 / x9	1	10
x18	2	22
pair of x4	1	4 per group, 8 per lane

Figure 8. x4 Group

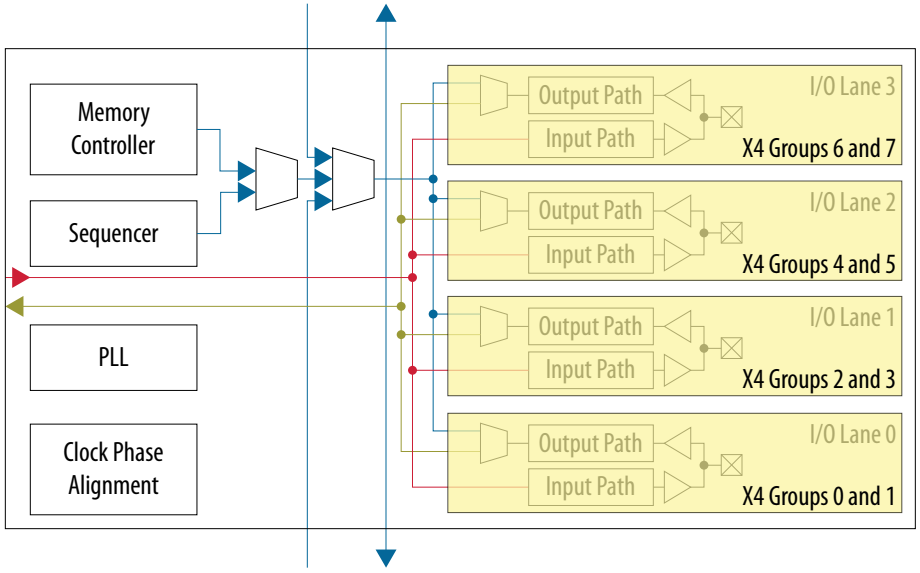


Figure 9. x8 Group

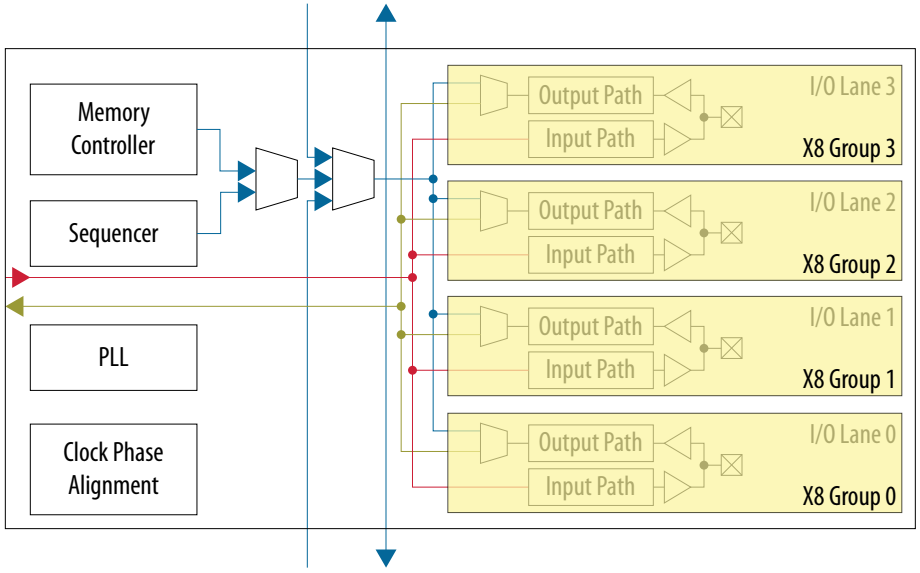
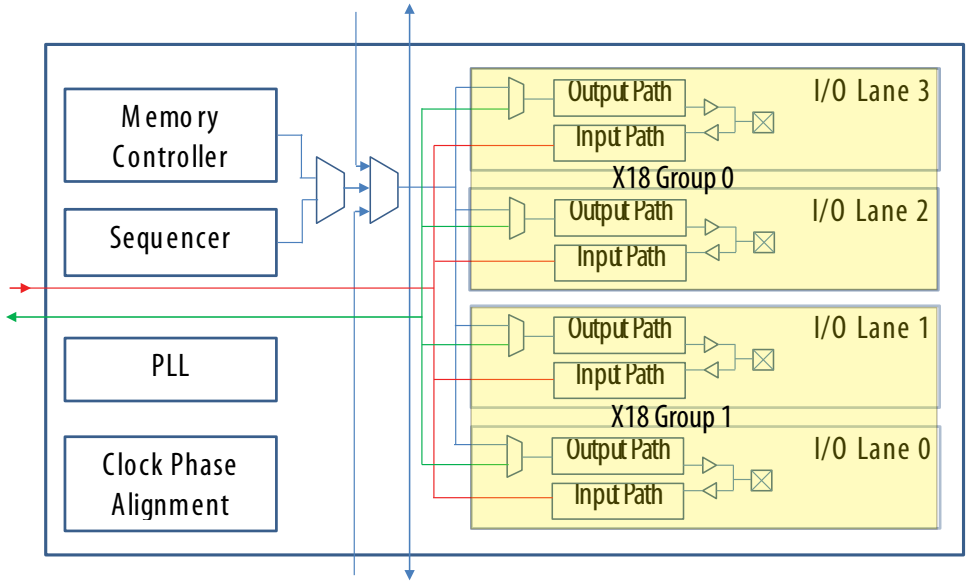


Figure 10. x18 Group



2.1.5. Intel Agilex EMIF Architecture: Input DQS Clock Tree

The input DQS clock tree is a balanced clock network that distributes the read capture clock (such as QK/QK# which are free-running read clocks) and strobe (such as DQS/DQS#) from the external memory device to the read capture registers inside the I/Os.

You can configure an input DQS clock tree in x4 mode, x8/x9 mode, or x18 mode.

Within every bank, only certain physical pins at specific locations can drive the input DQS clock trees. The pin locations that can drive the input DQS clock trees vary, depending on the size of the group.

Table 4. Pins Usable as Read Capture Clock / Strobe Pair

Group Size	Index of Lanes Spanned by Clock Tree [1]	Sub-Bank	Index of Pins Usable as Read Capture Clock / Strobe Pair	
			DQS p	DQS n
x4	0A	Bottom	4	5
x4	0B		6	7
x4	1A		16	17
x4	1B		18	19
x4	2A		28	29
x4	2B		30	31
x4	3A		40	41
x4	3B		42	43
x8 / x9	0		4	5
x8 / x9	1		16	17

continued...

Group Size	Index of Lanes Spanned by Clock Tree [1]	Sub-Bank	Index of Pins Usable as Read Capture Clock / Strobe Pair	
			DQS p	DQS n
x8 / x9	2		28	29
x8 / x9	3		40	41
x18	0, 1		4	5
x18	2, 3		28	29
x4	0A	Top	52	53
x4	0B		54	55
x4	1A		64	65
x4	1B		66	67
x4	2A		76	77
x4	2B		78	79
x4	3A		88	89
x4	3B		90	91
x8 / x9	0		52	53
x8 / x9	1		64	65
x8 / x9	2		76	77
x8 / x9	3		88	89
x18	0,1		53	53
x18	2,3		76	77
Note: [1] *A* and *B* refer to the two nibbles within the lane.				

2.1.6. Intel Agilex EMIF Architecture: PHY Clock Tree

Dedicated high-speed clock networks drive I/Os in Intel Agilex EMIF. Each PHY clock network spans only one sub-bank.

The relatively short span of the PHY clock trees results in low jitter and low duty-cycle distortion, maximizing the data valid window.

The PHY clock tree in Intel Agilex devices can run as fast as 1.6 GHz. All Intel Agilex external memory interfaces use the PHY clock trees.

2.1.7. Intel Agilex EMIF Architecture: PLL Reference Clock Networks

Each sub-bank includes an I/O bank I/O PLL that can drive the PHY clock trees of that bank, through dedicated connections. In addition to supporting EMIF-specific functions, the I/O bank I/O PLLs can also serve as general-purpose PLLs for user logic.

The PLL reference clock must be constrained to the address and command sub-bank only.

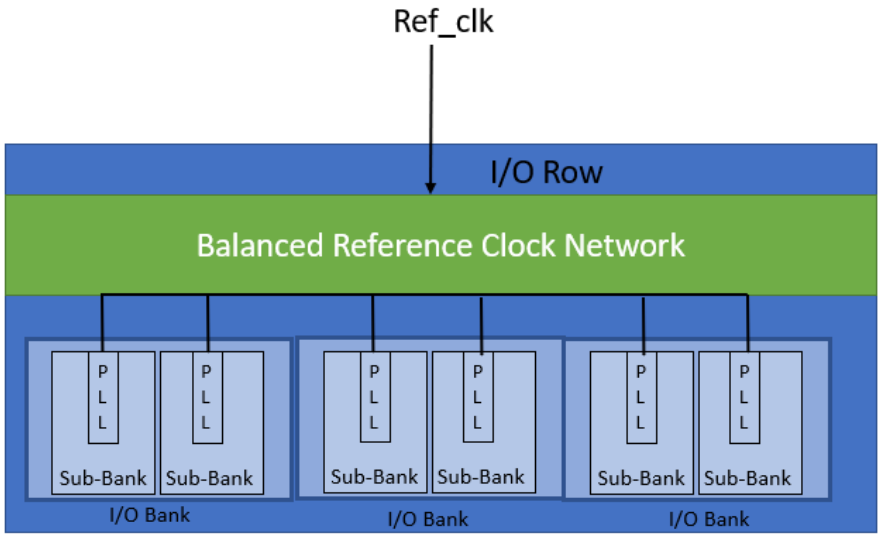
- A single-ended reference clock must be constrained to pin index 0 in lane 2. You cannot use pin index 1 in lane 2 as a general purpose I/O pin.
- Differential reference clocks must be constrained to pin indices 0 and 1 in lane 2.

Intel Agilex external memory interfaces that span multiple banks use the PLL in each bank. The Intel Agilex architecture allows for relatively short PHY clock networks, reducing jitter and duty-cycle distortion.

The following mechanisms ensure that the clock outputs of individual I/O bank I/O PLLs in a multi-bank interface remain in phase:

- A single PLL reference clock source feeds all I/O bank I/O PLLs. The reference clock signal reaches the PLLs by a balanced PLL reference clock tree. The Intel Quartus Prime software automatically configures the PLL reference clock tree so that it spans the correct number of banks.
- The EMIF IP sets the PLL configuration (counter settings, bandwidth settings, compensation and feedback mode setting) values appropriately to maintain synchronization among the clock dividers across the PLLs. This requirement restricts the legal PLL reference clock frequencies for a given memory interface frequency and clock rate. The Intel Agilex EMIF IP parameter editor automatically calculates and displays the set of legal PLL reference clock frequencies. If you plan to use an on-board oscillator, you must ensure that its frequency matches the PLL reference clock frequency that you select from the displayed list.

Figure 11. PLL Balanced Reference Clock Tree



2.1.8. Intel Agilex EMIF Architecture: Clock Phase Alignment

In Intel Agilex external memory interfaces, a global clock network clocks registers inside the FPGA core, and the PHY clock network clocks registers inside the FPGA periphery. Clock phase alignment circuitry employs negative feedback to dynamically adjust the phase of the core clock signal to match the phase of the PHY clock signal.

The clock phase alignment feature effectively eliminates the clock skew effect in all transfers between the core and the periphery, facilitating timing closure. All Intel Agilex external memory interfaces employ clock phase alignment circuitry.

Figure 12. Clock Phase Alignment Illustration

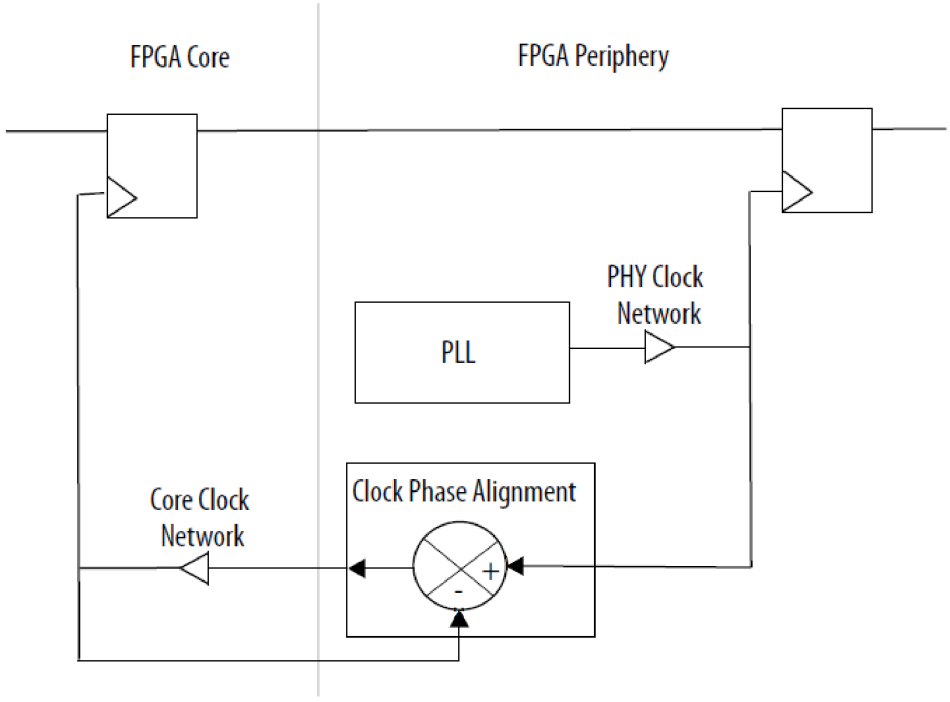
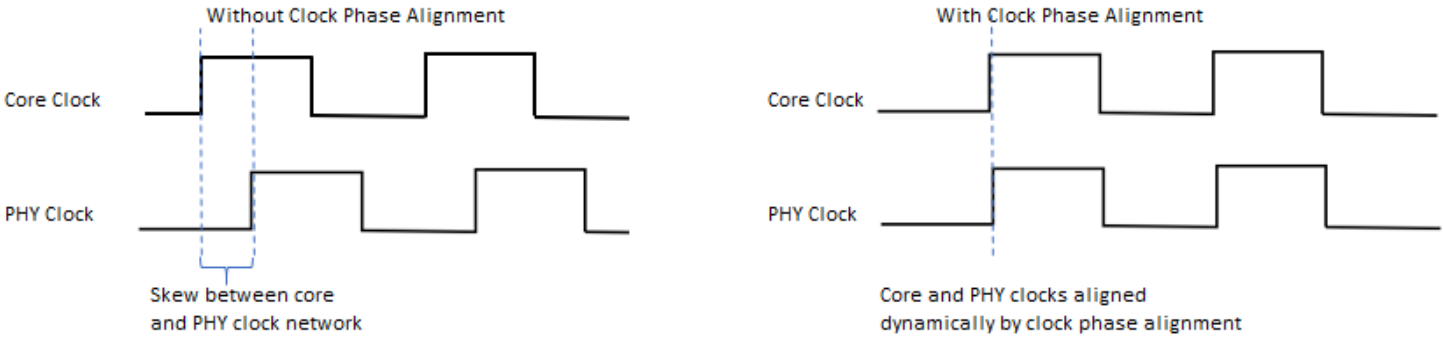


Figure 13. Effect of Clock Phase Alignment



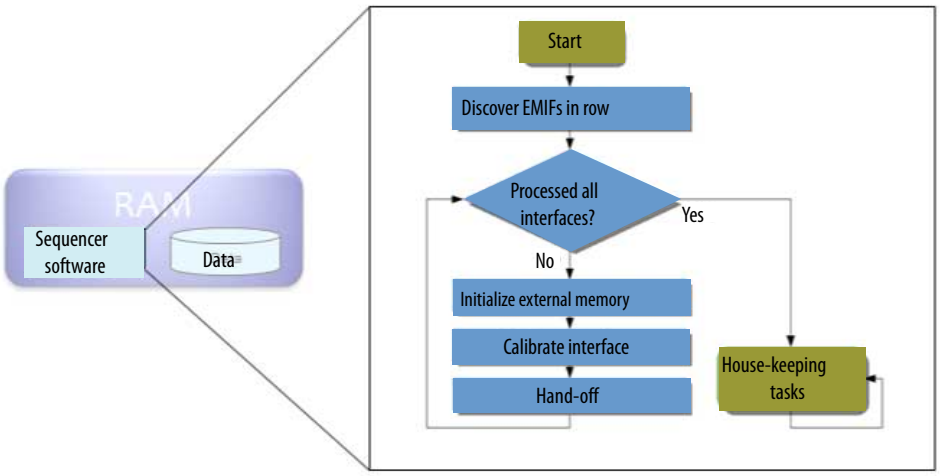
2.2. Intel Agilex EMIF Sequencer

The Intel Agilex EMIF sequencer is fully hardened in silicon, with executable code to handle protocols and topologies. Hardened RAM contains the calibration algorithm.

The Intel Agilex EMIF sequencer is responsible for the following operations:

- Initializes memory devices.
- Calibrates the external memory interface.
- Governs the hand-off of control to the memory controller.
- Handles recalibration requests and debug requests.
- Handles all supported protocols and configurations.

Figure 14. Intel Agilex EMIF Sequencer Operation



2.2.1. Intel Agilex EMIF DQS Tracking

DQS tracking tracks read capture clock/strobe timing variation over time, for improved read capture I/O timing. This feature takes sufficient samples to confirm the variation and adjust the DQS-enable position to maintain adequate operating margins.

DQS tracking is enabled for QDR-IV and RLDRAM 3 protocols; it is not available for DDR4. For QDR-IV and RLDRAM 3, DQS tracking does not need a specific command to initiate tracking, because the read capture clock/strobe is free running. Tracking happens constantly and automatically when the circuitry is enabled.

2.3. Intel Agilex EMIF Calibration

The calibration process compensates for skews and delays in the external memory interface.

The calibration process enables the system to compensate for the effects of factors such as the following:

- Timing and electrical constraints, such as setup/hold time and V_{ref} variations.
- Circuit board and package factors, such as skew, fly-by effects, and manufacturing variations.
- Environmental uncertainties, such as variations in voltage and temperature.
- The demanding effects of small margins associated with high-speed operation.

For a given external memory interface, calibration occurs on multiple pins in parallel whenever possible; however, some operations still operate on individual byte lanes sequentially. Interfaces in a row are calibrated in the order in which they are connected to the calibration IP (first the interface connected to calbus_0, then the interface connected to calbus_1, and so forth.)

Note: The calibration process is intended to maximize margins for robust EMIF operation; it cannot compensate for an inadequate PCB layout. Examples of PCB-related issues that cannot be calibrated, include the following:

- Excessive skew between signals sithin a byte lane.
- Inter-symbol interference caused by suboptimal trace topology, such as multiple vias, impedance mismatches, or discontinuities.
- Simultaneously-switching signal effects (victim/aggressor coupling caused by iinsufficient trace spacing, broadside coupling, or layer-to-layer coupling.
- Electrical noise effects such as improper plane referencing, split-plane crossing, routing signals too close to noisy sources such as switching power supplies or other high-frequency noise generators.
- Impedence mismatches, such as improper choices for FPGA/DRAM-side transmit/receive termination relative to PCB trace impedence, or excessive loading on the address/command or data buses due to multiple loads.

2.3.1. Intel Agilex Calibration Stages

At a high level, the calibration routine consists of address and command calibration, read calibration, and write calibration.

The stages of calibration vary, depending on the protocol of the external memory interface.

Table 5. Calibration Stages by Protocol

Stage	DDR4	RLDRAM 3	QDR-IV
Address and command			
Leveling	Yes	—	—
Deskew	Yes	—	Yes
Read			
DQSen	Yes	Yes	Yes
Deskew	Yes	Yes	Yes
VREF-In	Yes	—	Yes
LFIFO	Yes	Yes	Yes
Write			
Leveling	Yes	Yes	Yes
Deskew	Yes	Yes	Yes
VREF-Out	Yes	—	—

2.3.2. Intel Agilex Calibration Stages Descriptions

The various stages of calibration perform address and command calibration, read calibration, and write calibration.

Address and Command Calibration

The goal of address and command calibration is to delay address and command signals as necessary to optimize the address and command window. This stage is not available for all protocols and cannot compensate for a poorly implemented board design.

Address and command calibration consists of the following parts:

- Leveling calibration— Centers the CS# signal and the entire address and command bus, relative to the CK clock. This operation is available for DDR4 interfaces only.
- Deskew calibration— Provides per-bit deskew for the address and command bus (except CS#), relative to the CK clock. This operation is available for DDR4 and QDR-IV interfaces only.

Read Calibration

Read calibration consists of the following parts:

- DQSen calibration— Calibrates the timing of the read capture clock gating and ungating, so that the PHY can gate and ungate the read clock at precisely the correct time—if too early or too late, data corruption can occur. The algorithm for this stage varies, depending on the memory protocol.
- Deskew calibration— Performs per-bit deskew of read data relative to the read strobe or clock.
- VREF-In calibration— Calibrates the VREF level at the FPGA.
- LFIFO calibration: Normalizes differences in read delays between groups due to fly-by, skews, and other variables and uncertainties.

Write Calibration

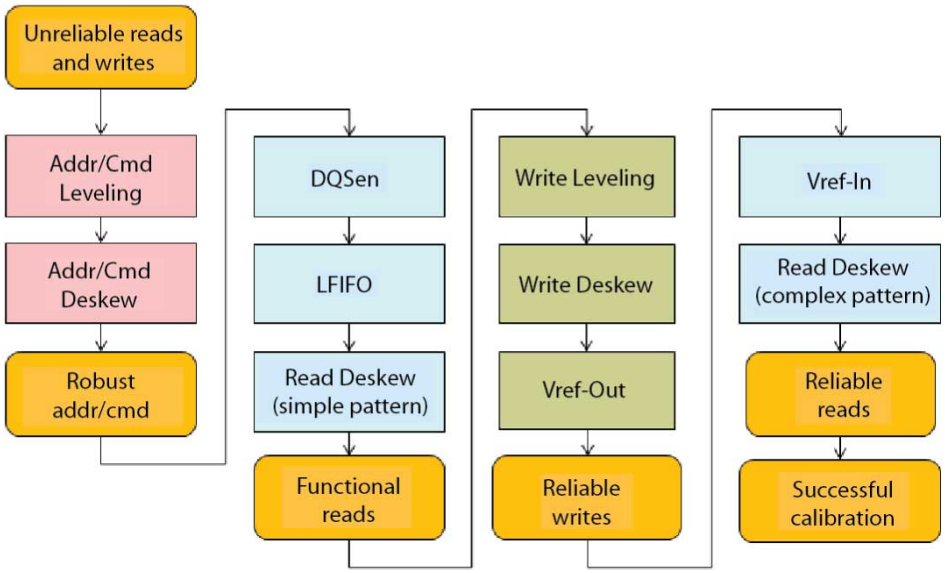
Write calibration consists of the following parts:

- Leveling calibration— Aligns the write strobe and clock to the memory clock, to compensate for skews, especially those associated with fly-by topology. The algorithm for this stage varies, depending on the memory protocol.
- Deskew calibration— Performs per-bit deskew of write data relative to the write strobe and clock.
- VREF-Out calibration— Calibrates the VREF level at the memory device.

2.3.3. Intel Agilex Calibration Flowchart

The following flowchart illustrates the Intel Agilex calibration flow.

Figure 15. Calibration Flowchart



2.3.4. Intel Agilex Calibration Algorithms

The calibration algorithms sometimes vary, depending on the targeted memory protocol.

Address and Command Calibration

Address and command calibration consists of the following parts:

- Leveling calibration— (DDR4 only) Toggles the CS# and CAS# signals to send read commands while keeping other address and command signals constant. The algorithm monitors for incoming DQS signals, and if the DQS signal toggles, it indicates that the read commands have been accepted. The algorithm then repeats using different delay values, to find the optimal window.
- Deskew calibration— (DDR4 and QDR-IV only)
  - (DDR4) Uses the DDR4 address and command parity feature. The FPGA sends the address and command parity bit, and the DDR4 memory device responds with an alert signal if the parity bit is detected. The alert signal from the memory device tells the FPGA that the parity bit was received.

    Deskew calibration requires use of the PAR/ALERT# pins, so you must not omit these pins from your design. One limitation of deskew calibration is that it cannot deskew ODT and CKE pins.
  - (QDR-IV) Uses the QDR-IV loopback mode. The FPGA sends address and command signals, and the memory device sends back the address and command signals which it captures, via the read data pins. The returned signals indicate to the FPGA what the memory device has captured. Deskew calibration can deskew all synchronous address and command signals.

    Note: For more information about loopback mode, refer to your QDR-IV memory device data sheet.

Read Calibration

- DQSen calibration— (DDR4, RLDRAM 3, and QDR-IV) DQSen calibration occurs before Read deskew, therefore only a single DQ bit is required to pass in order to achieve a successful read pass.
  - (DDR4) The DQSen calibration algorithm searches the DQS preamble using a hardware state machine. The algorithm sends many back-to-back reads with a one clock cycle gap between. The hardware state machine searches for the DQS gap while sweeping DQSen delay values. The algorithm then increments the VFIFO value, and repeats the process until a pattern is found. The process then repeats for all other read DQS groups.
  - (RLDRAM 3 and QDR-IV) The DQSen calibration algorithm does not use a hardware state machine; rather, it calibrates cycle-level delays using software and subcycle delays using DQS tracking hardware. The algorithm requires good data in memory, and therefore relies on guaranteed writes. (Writing a burst of 0s to one location, and a burst of 1s to another; back-to-back reads from these two locations are used for read calibration.)

    The algorithm enables DQS tracking to calibrate the phase component of DQS enable, and then issues a guaranteed write, followed by back-to-back reads. The algorithm sweeps DQSen values cycle by cycle until the read operation succeeds. The process then repeats for all other read groups.
- Deskew calibration— Read deskew calibration is performed before write leveling, and must be performed at least twice: once before write calibration, using simple data patterns from guaranteed writes, and again after write calibration, using complex data patterns.

  The deskew calibration algorithm performs a guaranteed write, and then sweeps `dqs_in` delay values from low to high, to find the right edge of the read window. The algorithm then sweeps `dq-in` delay values low to high, to find the left edge of the read window. Updated `dqs_in` and `dq_in` delay values are then applied to center the read window. The algorithm then repeats the process for all data pins.
- Vref-In calibration— Read `Vref-In` calibration begins by programming `Vref-In` with an arbitrary value. The algorithm then sweeps the `Vref-In` value from the starting value to both ends, and measures the read window for each value. The algorithm selects the `Vref-In` value which provides the maximum read window.
- LFIFO calibration— Read LFIFO calibration normalizes read delays between groups. The PHY must present all data to the controller as a single data bus. The LFIFO latency should be large enough for the slowest read data group, and large enough to allow proper synchronization across FIFOs.

Write Calibration

- Leveling calibration— Write leveling calibration aligns the write strobe and clock to the memory clock, to compensate for skews. In general, leveling calibration tries a variety of delay values to determine the edges of the write window, and then selects an appropriate value to center the window. The details of the algorithm vary, depending on the memory protocol.
  - (DDR4) Write leveling occurs before write deskew, therefore only one successful DQ bit is required to register a pass. Write leveling staggers the DQ bus to ensure that at least one DQ bit falls within the valid write window.
  - (RLDRAM 3) Optimizes for the CK versus DK relationship.
  - (QDR-IV) Optimizes for the CK versus DK relationship. It is covered by address and command deskew using the loopback mode.
- Deskew calibration— Performs per-bit deskew of write data relative to the write strobe and clock. Write deskew calibration does not change `dqs_out` delays; the write clock is aligned to the CK clock during write leveling.
- VREF-Out calibration— (DDR4) Calibrates the VREF level at the memory device. The VREF-Out calibration algorithm is similar to the VREF-In calibration algorithm.

2.4. Intel Agilex EMIF Controller

2.4.1. Hard Memory Controller

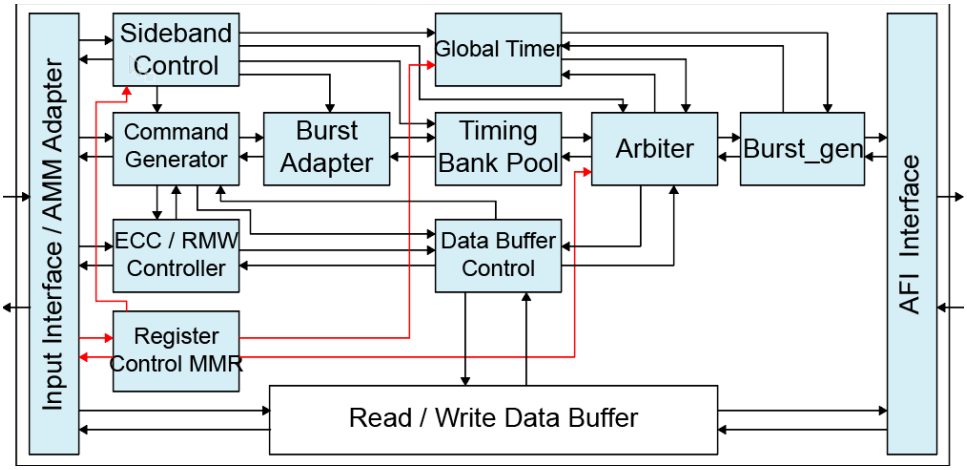
The Intel Agilex hard memory controller is designed for high speed, high performance, high flexibility, and area efficiency. The Intel Agilex hard memory controller supports the DDR4 memory standard.

The hard memory controller implements efficient pipelining techniques and advanced dynamic command and data reordering algorithms to improve bandwidth usage and reduce latency, providing a high performance solution.

The controller architecture is modular and fits in a single I/O sub-bank. The structure allows you to:

- Configure each I/O sub-bank as either:
  - A control path that drives all the address and command pins for the memory interface.
  - A data path that drives up to 32 data pins for DDR-type interfaces.
- Place your memory controller in any location.
- Pack up multiple banks together to form memory interfaces of different widths up to 72 bits.
- Bypass the hard memory controller and use your own custom IP if required.

Figure 16. Hard Memory Controller Architecture



The hard memory controller consists of the following logic blocks:

- Core and PHY interfaces
- Main control path
- Data buffer controller
- Read and write data buffers

The core interface supports the Avalon® Memory-Mapped (Avalon-MM) interface. The interface communicates to the PHY using the Altera PHY Interface (AFI). The whole control path is split into the main control path and the data buffer controller.

2.4.1.1. Hard Memory Controller Features

Table 6. Features of the Intel Agilex Hard Memory Controller

Feature	Description
Memory standards support	Supports DDR4 SDRAM.
Memory devices support	Supports the following memory devices: • Discrete • UDIMM • RDIMM • LRDIMM • SODIMM
3D Stacked Die support	Supports 2 and 4 height of 3D stacked die for DDR4 to increase memory capacity.
Memory controller bypass mode	You can use this configurable mode to bypass the hard memory controller and use your own customized controller.
Interface protocols support	• Supports Avalon-MM interface. • The PHY interface adheres to the AFI protocol.
Rate support	The legal options are:

continued...

Feature	Description
	• HMC half-rate, user logic half-rate (extremely slow interfaces only) • HMC half-rate, user-logic quarter-rate • HMC quarter-rate, user-logic quarter-rate (extremely high-speed interfaces only)
Configurable memory interface width	Supports data widths from 8 to 72 bits, in 8 bit increments
Multiple ranks support	Supports: • 4 ranks with single slot • 2 ranks with dual slots
Burst adapter	Able to accept burst lengths of 1–127 on the local interface of the controller and map the bursts to efficient memory commands. For applications that must strictly adhere to the -MM specification, the maximum burst length is 64. No burst chop support for DDR4.
Efficiency optimization features	• Open-page policy—by default, opens page on every access. However, the controller intelligently closes a row based on incoming traffic, which improves the efficiency of the controller especially for random traffic. • Pre-emptive bank management—the controller issues bank management commands early, which ensures that the required row is open when the read or write occurs. • Data reordering—the controller reorders read/write commands. • Additive latency—the controller can issue a `READ`/`WRITE` command after the `ACTIVATE` command to the memory bank prior to t_{RCD}, which increases the command efficiency.
Starvation counter	Ensures all requests are served after a predefined time out period, which ensures that low priority access are not left behind while reordering data for efficiency.
Bank interleaving	Able to issue read or write commands continuously to "random" addresses. You must correctly cycle the bank addresses.
On-die termination	The controller controls the on-die termination signal for the memory. This feature improves signal integrity and simplifies your board design.
Refresh features	• User-controlled refresh timing—optionally, you can control when refreshes occur and this allows you to prevent important read or write operations from clashing with the refresh lock-out time. • Per-rank refresh—allows refresh for each individual rank. • Controller-controlled refresh.
ECC support	• 8 bit ECC code; single error correction, double error detection (SECDED). • User ECC supporting pass through user ECC bits as part of data bits.

continued...

Feature	Description
Power saving features	• Low power modes (power down and self-refresh)—optionally, you can request the controller to put the memory into one of the two low power states. • Automatic power down—puts the memory device in power down mode when the controller is idle. You can configure the idle waiting time. • Memory clock gating.
DDR4 features	• Bank group support—supports different timing parameters for between bank groups. • Command/Address parity—command and address bus parity check. • Support Direct Dual CS Mode and Direct QuadCS Mode for DDR4 LRDIMM devices. • Support Encoded Quad CSMode for single CS assertion memory mapping for DDR4 LRDIMM devices.
User ZQ calibration	Long or short ZQ calibration request for DDR4.

2.4.1.2. Hard Memory Controller Main Control Path

The main control path performs the following functions:

- Contains the command processing pipeline.
- Monitors all the timing parameters.
- Keeps track of dependencies between memory access commands.
- Guards against memory access hazards.

Table 7. Main Control Path Components

Component	Description
Input interface	• Accepts memory access commands from the core logic at half or quarter rate. • Uses the Avalon-MM protocol. • You can connect the Avalon-MM interface to an AXI bus master in Platform Designer. To connect the Avalon-MM interface, implement the AXI bus master as a Platform Designer component and connect the AXI bus master to the Avalon-MM slave. The Platform Designer interconnect performs the bus translation between the AXI and Avalon-MM bus interfaces.
Command generator and burst adapter	• Drains your commands from the input interface and feeds them to the timing bank pool. • If read-modify-write is required, inserts the necessary read-modify-write read and write commands into the stream. • The burst adapter chops your arbitrary burst length to the number specified by the memory types.
Timing Bank Pool	• Key component in the memory controller. • Sets parallel queues to track command dependencies. • Signals the ready status of each command being tracked to the arbiter for the final dispatch. • Big scoreboard structure. The number of entries is currently sized to 16 where it monitors up to 16 commands at the same time. • Handles the memory access hazards such as Read After Write (RAW), Write After Read (WAR), and Write After Write (WAW), while part of the timing constraints are being tracked. • Assist the arbiter in reordering row commands and column commands. • When the pool is full, a flow control signal is sent back upstream to stall the traffic.

continued...

Component	Description
Arbiter	• Enforces the arbitration rules. • Performs the final arbitration to select a command from all ready commands, and issues the selected command to the memory. • Supports Quasi-1T mode for half rate mode. • For the quasi modes, a row command must be paired with a column command.
Global Timer	Tracks the global timing constraints including: • t_{FAW}—the Four Activates Window parameter that specifies the time period in which only four activate commands are allowed. • t_{RRD}—the delay between back-to-back activate commands to different banks. • Some of the bus turnaround time parameters.
MMR/IOCSR	• The host of all the configuration registers. • Uses Avalon-MM bus to talk to the core. • Core logic can read and write all the configuration bits.
Sideband	Executes the refresh and power down features.
ECC controller	Although ECC encoding and decoding is performed in soft logic(1), the ECC controller maintains the read-modify-write state machine in the hard solution.
AFI interface	The memory controller communicates with the PHY using this interface.

2.4.1.3. Data Buffer Controller

The data buffer controller performs the following operations:

- Manages the read and write access to the data buffers:
  - Provides the data storing pointers to the buffers when the write data is accepted or the read return data arrives.
  - Provides the draining pointer when the write data is dispatched to memory or the read data is read out of the buffer and sent back to users.
- Satisfies the required write latency.
- If ECC support is enabled, assists the main control path to perform read-modify-write.

Data reordering is performed with the data buffer controller and the data buffers.

2.4.2. Intel Agilex Hard Memory Controller Rate Conversion Feature

The hard memory controller's rate conversion feature allows the hard memory controller and PHY to run at half-rate, even though user logic is configured to run at quarter-rate.

To improve efficiency and help reduce overall latency, the hard memory controller and PHY run at half rate when the rate conversion feature is enabled. User logic runs at quarter-rate.

(1) ECC encoding and decoding is performed in soft logic to exempt the hard connection from routing data bits to a central ECC calculation location. Routing data to a central location removes the modular design benefits and reduces flexibility.

The rate conversion feature is enabled automatically during IP generation whenever all of the following conditions are met:

- The hard memory controller is in use.
- User logic runs at quarter-rate.
- Running the hard memory controller at half-rate does not exceed the fMax specification of the hard memory controller and hard PHY.

When the rate conversion feature is enabled, you should see the following info message displayed in the IP generation GUI:
`PHY and controller running at 2x the frequency of user logic for improved efficiency.`

2.5. User-requested Reset in Intel Agilex EMIF IP

The following table summarizes information about the user-requested reset mechanism in the Intel Agilex EMIF IP.

Table 8.

	Description
Reset-related signals	`local_reset_req` (input) `local_reset_done` (output)
When can user logic request a reset?	`local_reset_req` has effect only when `local_reset_done` is high. After device power-on, the `local_reset_done` signal transitions high upon completion of the first calibration, whether the calibration is successful or not.
Is user-requested reset a requirement?	A user-requested reset is optional. The I/O SSM automatically ensures that the memory interface begins from a known state as part of the device power-on sequence. A user-requested reset is necessary only if the user logic must explicitly reset a memory interface after the device power-on sequence.
When does a user-requested reset actually happen?	Each EMIF IP instance has its own local reset request port which it must assert in order to be recalibrated. The I/O SSM continually scans the reset requests of all the EMIF interfaces that it controls, and recalibrates them when it is able to do so. The exact timing of the recalibration cannot be predicted.
Timing requirement and triggering mechanism.	Reset request is sent by transitioning the `local_reset_req` signal from low to high, then keeping the signal at the high state for a minimum of 2 EMIF core clock cycles, then transitioning the signal from high to low. `local_reset_req` is asynchronous in that there is no setup/hold timing to meet, but it must meet the minimum pulse width requirement of 2 EMIF core clock cycles.
How long can an external memory interface be kept in reset?	It is not possible to keep an external memory interface in reset indefinitely. Asserting `local_reset_req` high continuously has no effect as a reset request is completed by a full 0->1->0 pulse.

continued...

	Description
Delaying initial calibration.	Initial calibration cannot be skipped. The `local_reset_done` signal is driven high only after initial calibration has completed.
Reset scope (within an external memory interface).	Only circuits that are required to restore EMIF to power-up state are reset. Excluded from the reset sequence are the IOSSM, the IOPLL(s), the DLL(s), and the CPA.
Reset scope (within an I/O row).	`local_reset_req` is a per-interface reset.

Method for Initiating a User-requested Reset

Step 1 - Precondition

Before asserting `local_reset_req`, user logic must ensure that the `local_reset_done` signal is high.

As part of the device power-on sequence, the `local_reset_done` signal automatically transitions to high upon the completion of the interface calibration sequence, regardless of whether calibration is successful or not.

Note: When targeting a group of interfaces that share the same core clocks, user logic must ensure that the `local_reset_done` signal of every interface is high.

Step 2 - Reset Request

After the pre-condition is satisfied, user logic can send a reset request by driving the `local_cal_req` signal from low to high and then low again (that is, by sending a pulse of 1).

- The low-to-high and high-to-low transitions can occur asychronously; that is, they need not happen in relation to any clock edges. However, the pulse must meet a minimum pulse width of at least 2 EMIF core clock cycles. For example, if the `emif_usr_clk` has a period of 4ns, then the `local_reset_req` pulse must last at least 8ns (that is, two `emif_usr_clk` periods).
- The reset request is considered complete only after the high-to-low transition. The EMIF IP does not initiate the reset sequence when the `local_reset_req` is simply held high.
- Additional pulses to `local_reset_req` are ignored until the reset sequence is completed.

Optional - Detecting local_reset_done deassertion and assertion

If you want, you can monitor the status of the `local_reset_done` signal to explicitly detect the status of the reset sequence.

- After the EMIF IP receives a reset request, it deasserts the `local_reset_done` signal. After initial power-up calibration, `local_reset_done` is de-asserted only in response to a user-requested reset. The reset sequence is imminent when `local_reset_done` has transitioned to low, although the exact timing depends on the current state of the I/O SSM. As part of the EMIF reset sequence, the core reset signal (`emif_usr_reset_n`, `afi_reset_n`) is driven low. Do not use a register reset by the core reset signal to sample `local_reset_done`.
- After the reset sequence has completed, `local_reset_done` is driven high again. `local_reset_done` being driven high indicates the completion of the reset sequence and the readiness to accept a new reset request; however, it does not imply that calibration was successful or that the hard memory controller is ready to accept requests. For these purposes, user logic must check signals such as `afi_cal_success`, `afi_cal_fail`, `local_cal_success`, `local_cal_fail`, and `amm_ready`.

2.6. Intel Agilex EMIF for Hard Processor Subsystem

The Intel Agilex EMIF IP can enable the Intel Agilex Hard Processor Subsystem (HPS) to access external DRAM memory devices.

Note: The current version of the External Memory Interfaces Intel Agilex FPGA IP does not support the Hard Processor Subsystem; HPS support will be available in a future version.

To enable connectivity between the Intel Agilex HPS and the Intel Agilex EMIF IP, you must create and configure an instance of the Intel Agilex External Memory Interface for HPS IP core, and use Platform Designer to connect it to the Intel Agilex Hard Processor Subsystem instance in your system.

Supported Modes

The Intel Agilex Hard Processor Subsystem is compatible with the following external memory configurations:

Table 9. Intel Agilex Hard Processor Subsystem Compatibility

Protocol	DDR4
Maximum memory clock frequency	1600MHz
Configuration	Hard PHY with hard memory controller
Clock rate of PHY and hard memory controller	Half-rate, Quarter-rate
Data width (without ECC)	16-bit, 32-bit, 64-bit
Data width (with ECC)	24-bit, 40-bit, 72-bit
DQ width per group	x8
Memory format	Supports up to 32GB of memory. • Discrete components with up to 2 chip selects * • Non-3DS UDIMM or RDIMM with up to 2 chip selects * • SODIMM with up to 2 ranks *

* Only one differential memory clock output is provided; therefore, you must do one of the following:

- Use single-rank discrete components, UDIMMs, or SODIMMs.
- Use dual-rank components that require only one clock input (for example, dual-die packages).
- Use RDIMMs that rely only on one clock input.
- Use the single clock output to drive both clock inputs and confirm through simulation that the memory interface margins are not adversely affected by the double loading of the clock output.

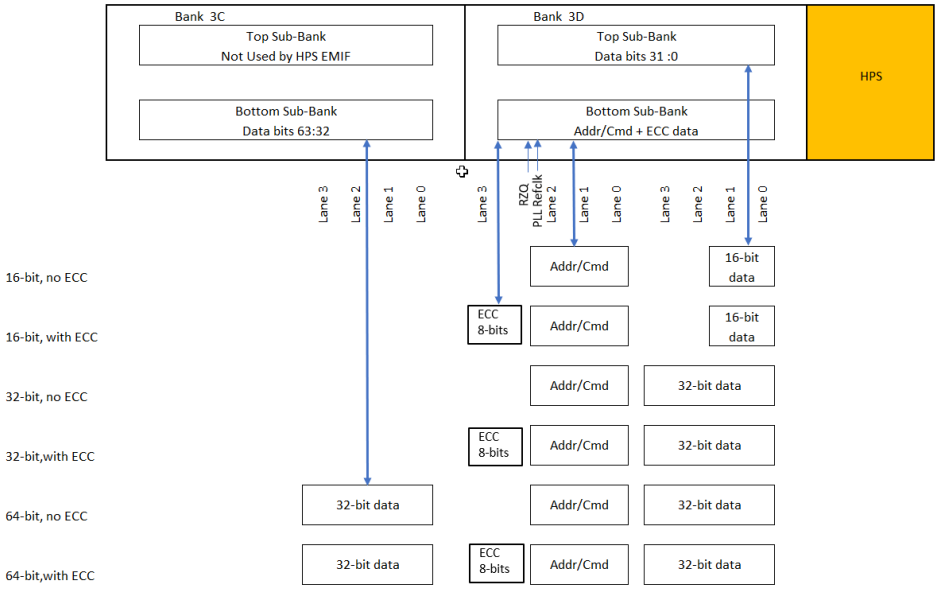
2.6.1. Restrictions on I/O Bank Usage for Intel Agilex EMIF IP with HPS

You can use only certain Intel Agilex I/O banks to implement Intel Agilex EMIF IP with the Intel Agilex Hard Processor Subsystem (HPS).

The restrictions on I/O bank usage result from the Intel Agilex HPS having hard-wired connections to the EMIF circuits in the I/O banks closest to the HPS. For any given EMIF configuration, the pin-out of the EMIF-to-HPS interface is fixed.

The following diagram illustrates the use of I/O banks and lanes for various EMIF-HPS data widths:

Figure 17. Intel Agilex HPS - EMIF I/O Bank and Lanes Usage



The HPS EMIF uses the closest located external memory interfaces I/O banks to connect to SDRAM. This arrangement of HPS EMIF address and command tile relative to the data tiles is not supported for fabric EMIF in the current version of the Intel Quartus Prime Design Suite.

The following diagram illustrates restrictions on I/O pin usage. Refer to the text following the diagram for a detailed explanation of these restrictions.

Figure 18. I/O Pin Usage Restrictions for Intel Agilex External Memory Interface with HPS (1 of 3)

Bank 3C, Bottom Sub-Bank(Sub_bank for Data bits 63:32)

	Index			x16/x24		x32/x40		x64/x72	
Lane	Within Sub-bank	Within Lane	Pin Name	ECC OFF	ECC ON	ECC OFF	ECC ON	ECC OFF	ECC ON
Lane 3	47	11	LVDSRX3C_13N						
	46	10	LVDSRX3C_13P						
	45	9	LVDSTX3C_13N						
	44	8	LVDSTX3C_13P						
	43	7	LVDSRX3C_14N						
	42	6	LVDSRX3C_14P						
	41	5	LVDSTX3C_14N						
	40	4	LVDSTX3C_14P						
	39	3	LVDSRX3C_15N						
	38	2	LVDSRX3C_15P						
	37	1	LVDSTX3C_15N						
	36	0	LVDSTX3C_15P						
Lane 2	35	11	LVDSRX3C_16N						
	34	10	LVDSRX3C_16P						
	33	9	LVDSTX3C_16N						
	32	8	LVDSTX3C_16P						
	31	7	LVDSRX3C_17N						
	30	6	LVDSRX3C_17P						
	29	5	LVDSTX3C_17N						
	28	4	LVDSTX3C_17P						
	27	3	LVDSRX3C_18N						
	26	2	LVDSRX3C_18P						
	25	1	LVDSTX3C_18N						
	24	0	LVDSTX3C_18P						
Lane 1	23	11	LVDSRX3C_19N						
	22	10	LVDSRX3C_19P						
	21	9	LVDSTX3C_19N						
	20	8	LVDSTX3C_19P						
	19	7	LVDSRX3C_20N						
	18	6	LVDSRX3C_20P						
	17	5	LVDSTX3C_20N						
	16	4	LVDSTX3C_20P						
	15	3	LVDSRX3C_21N						
	14	2	LVDSRX3C_21P						
	13	1	LVDSTX3C_21N						
	12	0	LVDSTX3C_21P						
Lane 0	11	11	LVDSRX3C_22N						
	10	10	LVDSRX3C_22P						
	9	9	LVDSTX3C_22N						
	8	8	LVDSTX3C_22P						
	7	7	LVDSRX3C_23N						
	6	6	LVDSRX3C_23P						
	5	5	LVDSTX3C_23N						
	4	4	LVDSTX3C_23P						
	3	3	LVDSRX3C_24N						
	2	2	LVDSRX3C_24P						
	1	1	LVDSTX3C_24N						
	0	0	LVDSTX3C_24P						

- Addr/Cmd pinss only (Do not use unused pins)
- ALERT_N Pin only
- Do Not Use (No Connect)
- DQSp Pin Only
- DQSn Pin Only
- DBI/DM Pin Only
- DQ Pin Only
- RZQ only
- HPS REFCLK_P only
- HPS REFCLK_N only (LVDS reference clock mode, no connect if unused)
- Free for FPGA Fabric Use

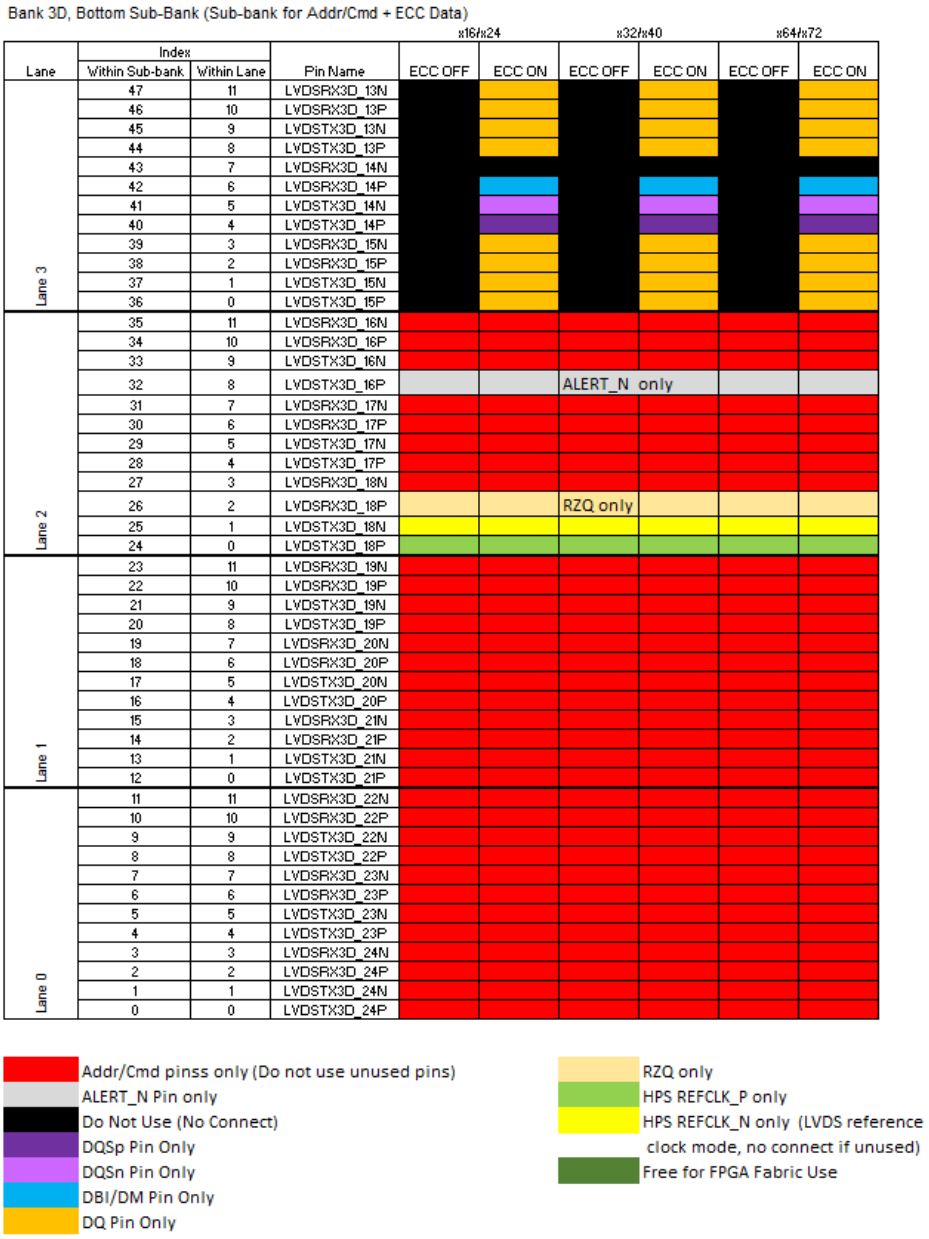
Figure 19. I/O Pin Usage Restrictions for Intel Agilex External Memory Interface with HPS (2 of 3)

Bank 3D, Top Sub-Bank (Sub-bank for Data bits 31:0)

	Index			x16/x24		x32/x40		x64/x72	
Lane	Within Sub-bank	Within Lane	Pin Name	ECC OFF	ECC ON	ECC OFF	ECC ON	ECC OFF	ECC ON
Lane 3	95	11	LVDSRX3D_1N						
	94	10	LVDSRX3D_1P						
	93	9	LVDSTX3D_1N						
	92	8	LVDSTX3D_1P						
	91	7	LVDSRX3D_2N						
	90	6	LVDSRX3D_2P						
	89	5	LVDSTX3D_2N						
	88	4	LVDSTX3D_2P						
	87	3	LVDSRX3D_3N						
	86	2	LVDSRX3D_3P						
	85	1	LVDSTX3D_3N						
	84	0	LVDSTX3D_3P						
Lane 2	83	11	LVDSRX3D_4N						
	82	10	LVDSRX3D_4P						
	81	9	LVDSTX3D_4N						
	80	8	LVDSTX3D_4P						
	79	7	LVDSRX3D_5N						
	78	6	LVDSRX3D_5P						
	77	5	LVDSTX3D_5N						
	76	4	LVDSTX3D_5P						
	75	3	LVDSRX3D_6N						
	74	2	LVDSRX3D_6P						
	73	1	LVDSTX3D_6N						
	72	0	LVDSTX3D_6P						
Lane 1	71	11	LVDSRX3D_7N						
	70	10	LVDSRX3D_7P						
	69	9	LVDSTX3D_7N						
	68	8	LVDSTX3D_7P						
	67	7	LVDSRX3D_8N						
	66	6	LVDSRX3D_8P						
	65	5	LVDSTX3D_8N						
	64	4	LVDSTX3D_8P						
	63	3	LVDSRX3D_9N						
	62	2	LVDSRX3D_9P						
	61	1	LVDSTX3D_9N						
	60	0	LVDSTX3D_9P						
Lane 0	59	11	LVDSRX3D_10N						
	58	10	LVDSRX3D_10P						
	57	9	LVDSTX3D_10N						
	56	8	LVDSTX3D_10P						
	55	7	LVDSRX3D_11N						
	54	6	LVDSRX3D_11P						
	53	5	LVDSTX3D_11N						
	52	4	LVDSTX3D_11P						
	51	3	LVDSRX3D_12N						
	50	2	LVDSRX3D_12P						
	49	1	LVDSTX3D_12N						
	48	0	LVDSTX3D_12P						

- Addr/Cmd pinss only (Do not use unused pins)
- ALERT_N Pin only
- Do Not Use (No Connect)
- DQSp Pin Only
- DQSn Pin Only
- DBI/DM Pin Only
- DQ Pin Only
- RZQ only
- HPS REFCLK_P only
- HPS REFCLK_N only (LVDS reference clock mode, no connect if unused)
- Free for FPGA Fabric Use

Figure 20. I/O Pin Usage Restrictions for Intel Agilex External Memory Interface with HPS (3 of 3)



The HPS EMIF IP must be used whenever the HPS is active. Thus, you should be aware that enabling the HPS necessarily means that an EMIF must be placed at this location in order to implement an FPGA design.

If there is an HPS EMIF in a system, the unused HPS EMIF pins can be used as FPGA general purpose I/O, with the following restrictions:

- Bank 3D, Bottom Sub-bank (Sub-bank for Address/Command + ECC Data):
  - Lane 3 is used for data bits only when ECC mode is active. Whether ECC is active or not, you must not put general purpose I/Os in this lane.
  - Lanes 2, 1, and 0 are used for SDRAM address and command. Unused pins in these lanes must not be used by the FPGA fabric.
  - ALERT_N pin must be placed at pin index 8, lane 2. There is no flexibility on this,
- Bank 3D, Top Sub-bank (Sub-bank for data bits 31:0) :
  - Lanes 3, 2, 1, and 0 are used for data bits.
  - With 32-bit data widths, unused pins in this bank must not be used by the FPGA fabric.
  - With 16-bit data widths, lanes 0 and 1 are used as data lanes. Unused pins in lane 0 and lane 1 must not be used by FPGA fabric. Unused pins in lanes 2 and 3 must not be used by the FPGA fabric, even though lanes 2 and 3 are not used by HPS EMIF.
- Bank 3C, Bottom Sub-bank (Sub-bank for Data bits 63:32)
  - With 64-bit data widths, lanes 3, 2, 1, and 0 are used for data bits [63:32]. Unused pins in these lanes must not be used by the FPGA fabric.
  - With 32-bit data widths, the entire bottom sub-bank can be used by the FPGA fabric. There are no restrictions.
- Bank 3C, Top Sub-bank
  - Not used by HPS EMIF. Unused pins in this bank can be used by FPGA fabric when the bottom sub-bank in 3C is not used for 64-bit HPS EMIF.
  - The following restrictions apply on the top sub-bank when the bottom sub-bank in 3C is used for 64-bit HPS EMIF:
    - This sub-bank can be used to form a larger non-HPS EMIF, but you cannot place an address and command bank in this sub-bank.
    - 1.5V true differential signaling is not supported.
    - I/O PLL reconfiguration is not supported.

By default, the Intel Agilex External Memory Interface for HPS IP core together with the Intel Quartus Prime Fitter automatically implements a starting point placement which you may need to modify. You must adhere to the following requirements, which are specific to HPS EMIF:

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 and DQS# must use pins at index 4 and 5, respectively. There is no flexibility.
- Pin index 7 must have no fabric usage and cannot implement general purpose I/Os.

2. In all cases the DQS groups can be swapped around the I/O banks shown. There is no requirement for the ECC DQS group to be placed in the bottom sub-bank in bank 3D.
3. In the bottom sub-bank in bank 3D (sub-bank for address and command + ECC data):
   - You must not change placement of the address and command pins from the default.
   - Place the `alert#` pin in lane 2, pin index 8.
   - Place the PLL reference clock in this sub-bank. Failure to place the PLL reference clock in this sub-bank will cause device configuration problems. The PLL reference clock must be running at the correct frequency before device configuration occurs.
   - Place the RZQ pin in this sub-bank. Failure to place the RZQ pin in this sub-bank will cause Fitter or device configuration problems.
4. To override the default generated pin assignments, comment out the relevant `HPS_LOCATION` assignments in the `.qip` file, and add your own location assignments (using `set_location_assignment`) in the `.qsf` file.

3. Document Revision History for Intel Agilex FPGA External Memory Interface Overview

Document Version	Changes
2019.04.02	• Initial release.