Arria V Device Handbook

Volume 1: Device Interfaces and Integration

Subscribe

Send Feedback

AV-5V2
2020.04.13

101 Innovation Drive
San Jose, CA 95134
www.altera.com

ALTERA® now part of Intel

Contents

Logic Array Blocks and Adaptive Logic Modules in Arria V Devices

1

2020.04.13

AV-52001

Subscribe Send Feedback

This chapter describes the features of the logic array block (LAB) in the Arria® V core fabric.

The LAB is composed of basic building blocks known as adaptive logic modules (ALMs) that you can configure to implement logic functions, arithmetic functions, and register functions.

You can use a quarter of the available LABs in the Arria V devices as a memory LAB (MLAB).

The Intel® Quartus® Prime software and other supported third-party synthesis tools, in conjunction with parameterized functions such as the library of parameterized modules (LPM), automatically choose the appropriate mode for common functions such as counters, adders, subtractors, and arithmetic functions.

This chapter contains the following sections:

- LAB
- ALM Operating Modes

Related Information

Arria V Device Handbook: Known Issues
Lists the planned updates to the Arria V Device Handbook chapters.

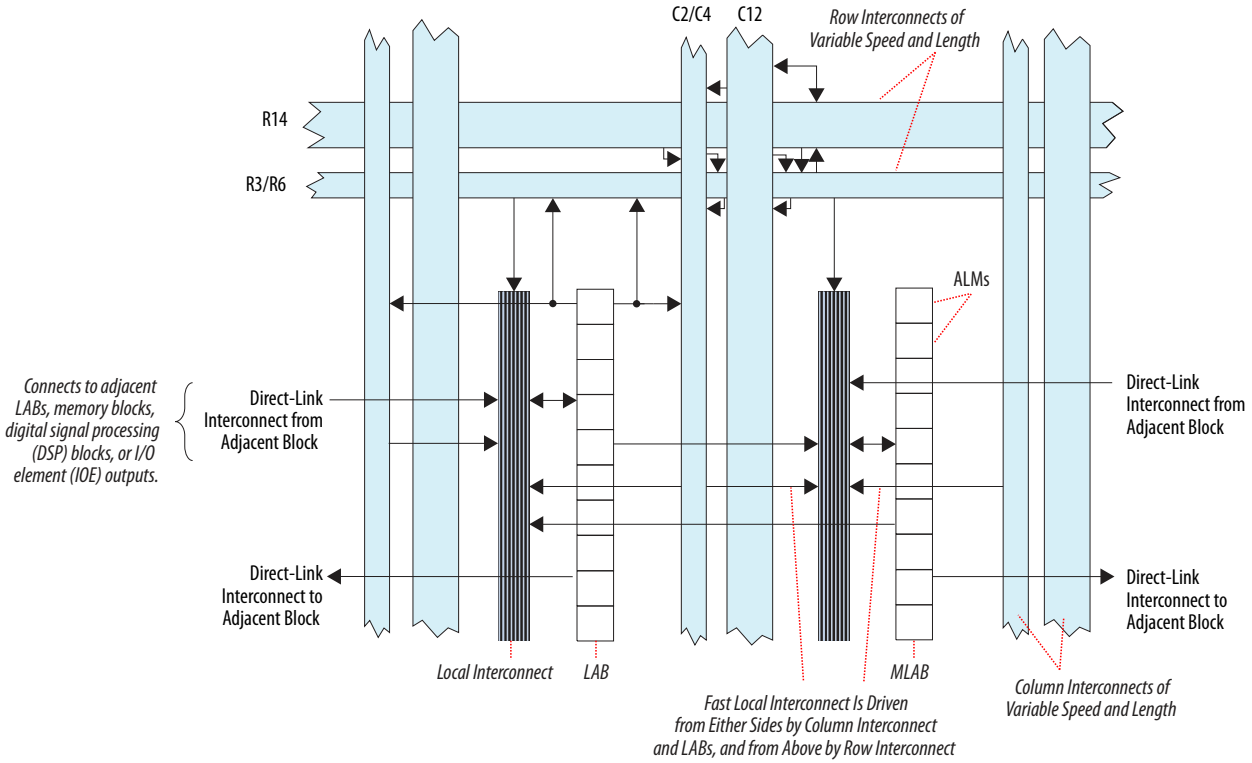
LAB

The LABs are configurable logic blocks that consist of a group of logic resources. Each LAB contains dedicated logic for driving control signals to its ALMs.

MLAB is a superset of the LAB and includes all the LAB features.

Figure 1-1: LAB Structure and Interconnects Overview in Arria V Devices

This figure shows an overview of the Arria V LAB and MLAB structure with the LAB interconnects.



MLAB

Each MLAB supports a maximum of 640 bits of simple dual-port SRAM.

You can configure each ALM in an MLAB in the following configurations:

- A 32 x 2 memory block, resulting in a configuration of 32 x 20 simple dual-port SRAM block for Arria V GX, GT, SX, and ST devices
- Either a 64 × 1 or a 32 × 2 block, resulting in a configuration of either a 64 × 10 or a 32 × 20 simple dual-port SRAM block for Arria V GZ devices

Figure 1-2: LAB and MLAB Structure for Arria V GX, GT, SX, and, ST Devices

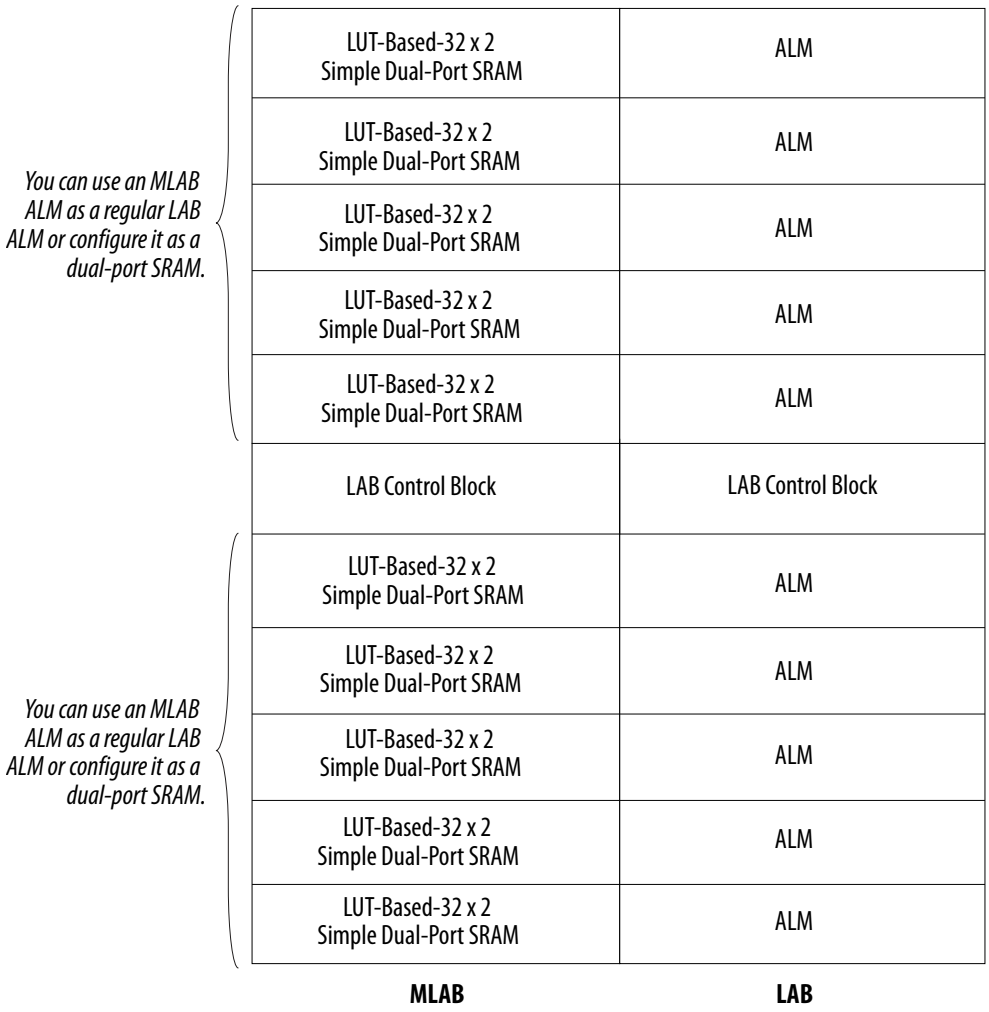
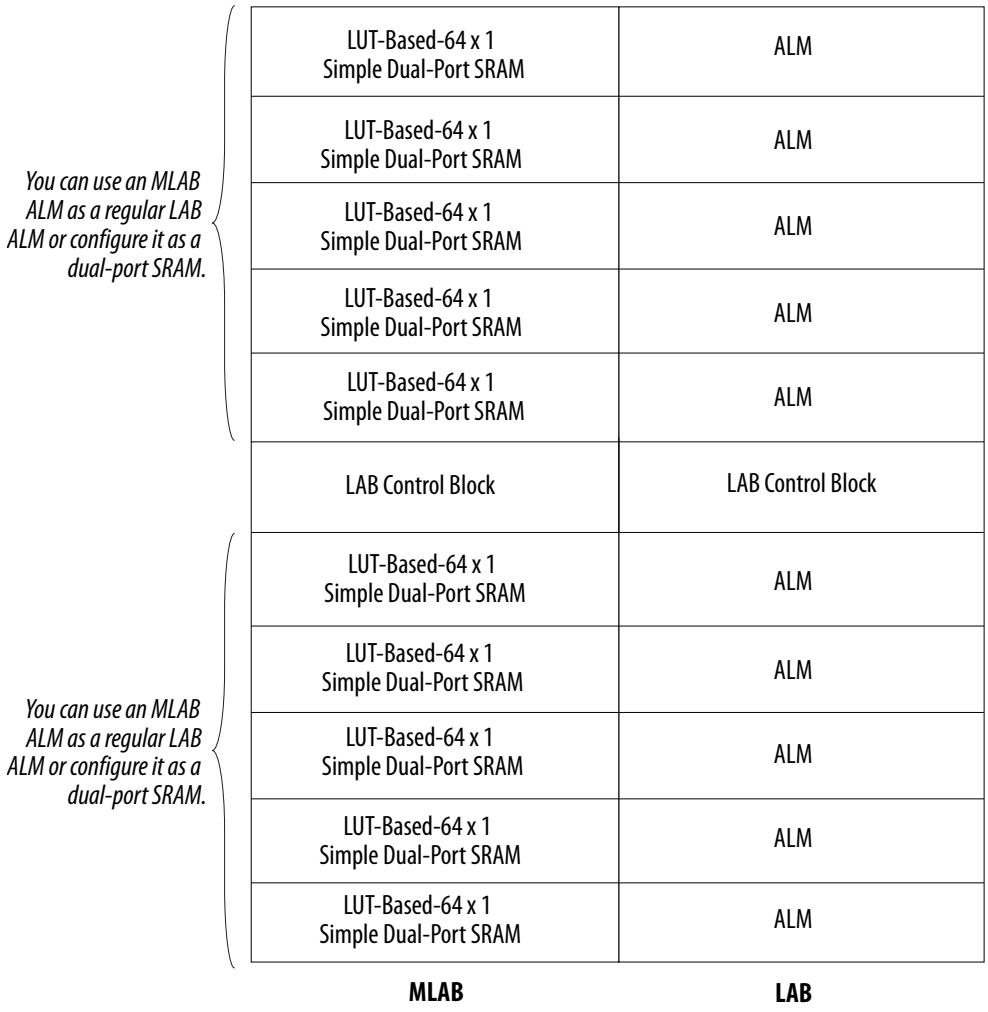


Figure 1-3: LAB and MLAB Structure for Arria V GZ Devices



Local and Direct Link Interconnects

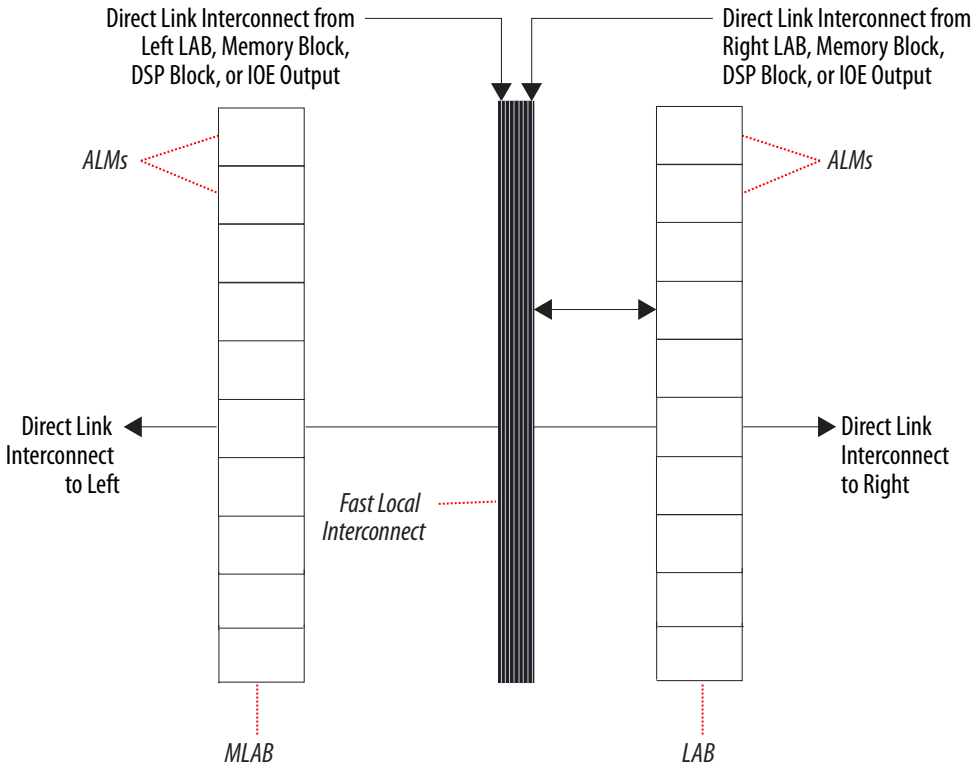
Each LAB can drive 30 ALMs through fast-local and direct-link interconnects. Ten ALMs are in any given LAB and ten ALMs are in each of the adjacent LABs.

The local interconnect can drive ALMs in the same LAB using column and row interconnects and ALM outputs in the same LAB.

Neighboring LABs, MLABs, M20K and M10K blocks, or digital signal processing (DSP) blocks from the left or right can also drive the LAB's local interconnect using the direct link connection.

The direct link connection feature minimizes the use of row and column interconnects, providing higher performance and flexibility.

Figure 1-4: LAB Fast Local and Direct Link Interconnects for Arria V Devices



LAB Control Signals

Each LAB contains dedicated logic for driving the control signals to its ALMs, and has two unique clock sources and three clock enable signals.

The LAB control block generates up to three clocks using the two clock sources and three clock enable signals. An inverted clock source is considered as an individual clock source. Each clock and the clock enable signals are linked.

De-asserting the clock enable signal turns off the corresponding LAB-wide clock.

Figure 1-5: LAB-Wide Control Signals for Arria V GX, GT, SX, and, ST Devices

This figure shows the clock sources and clock enable signals in a LAB.

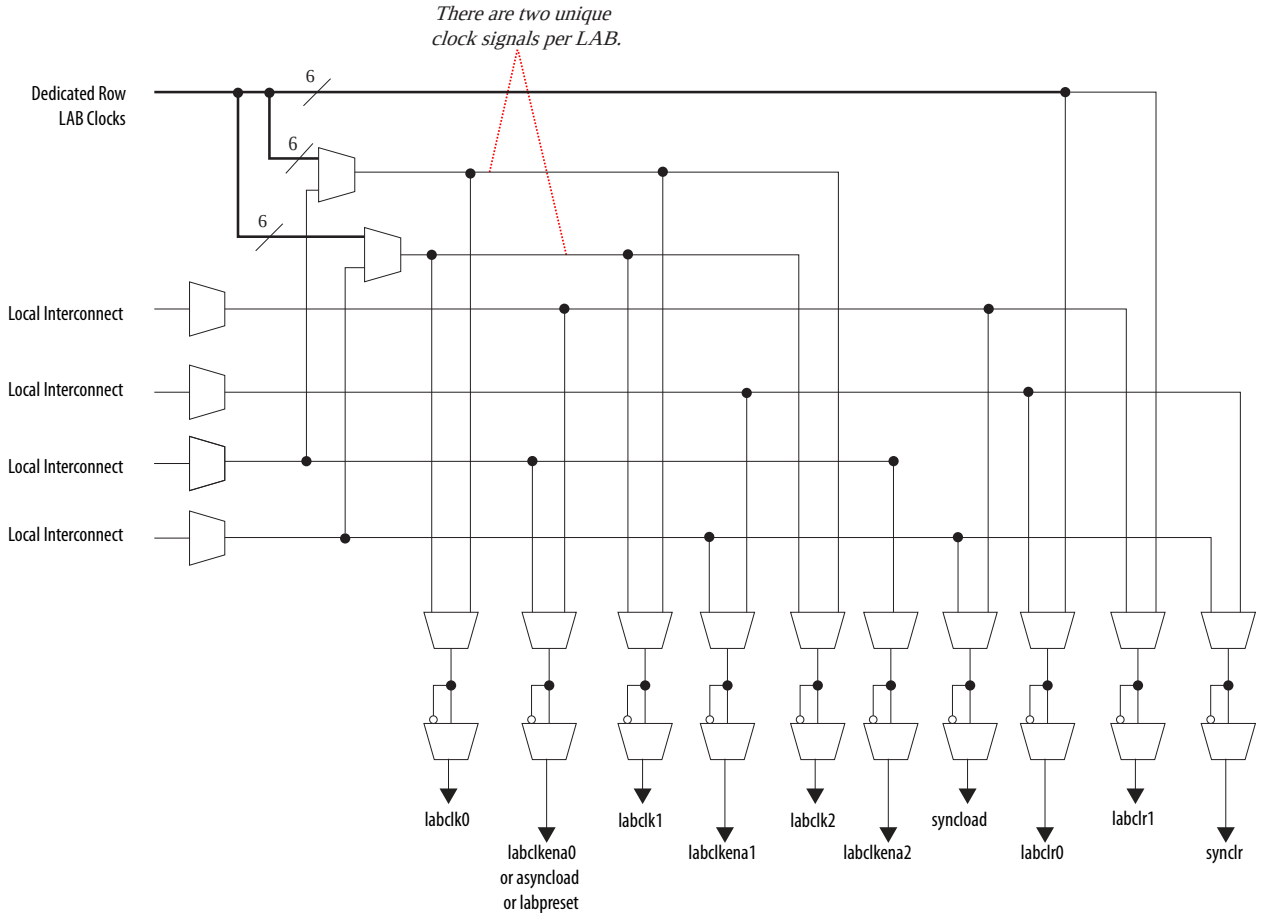
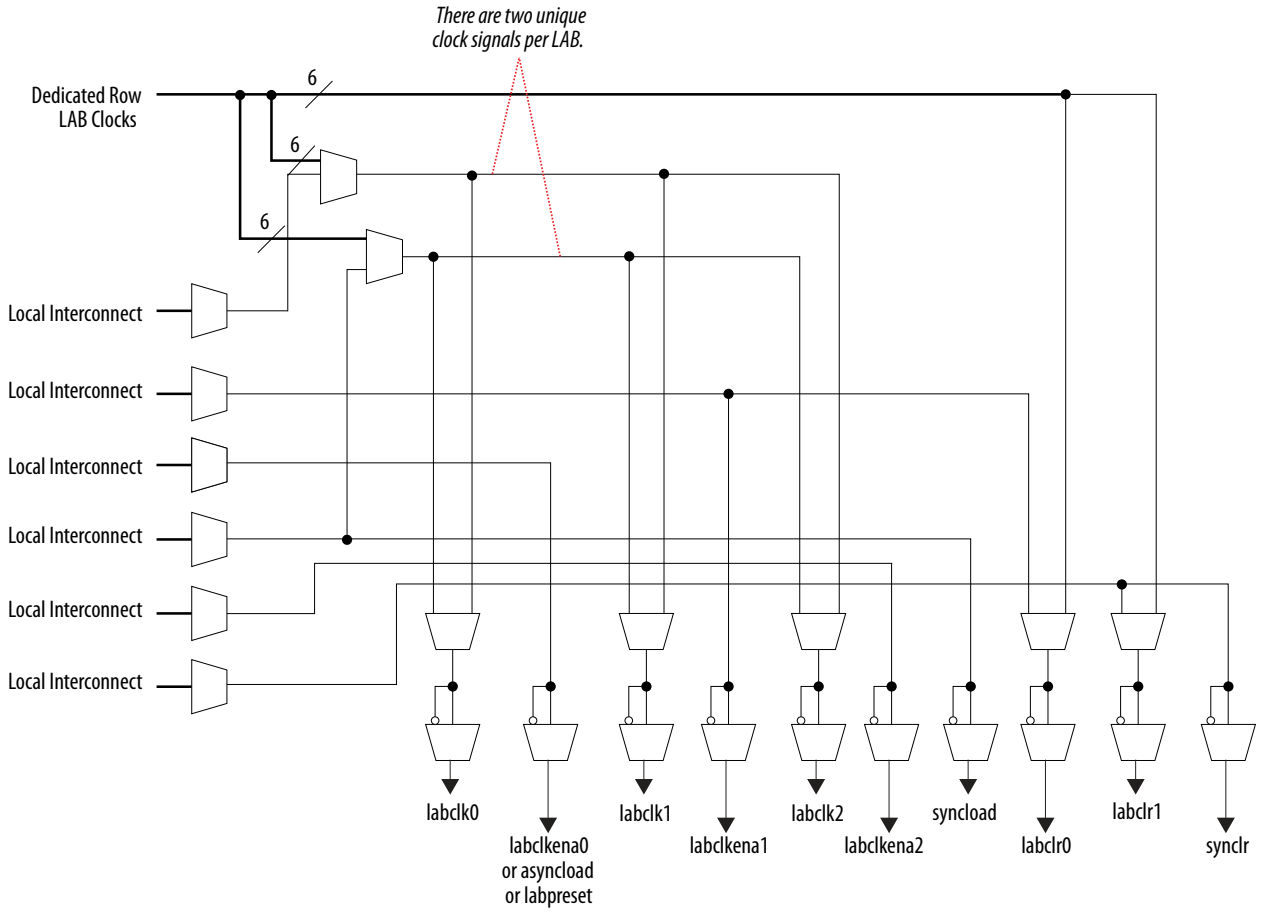


Figure 1-6: LAB-Wide Control Signals for Arria V GZ Devices

This figure shows the clock sources and clock enable signals in a LAB.



ALM Resources

One ALM contains four programmable registers. Each register has the following ports:

- Data
- Clock
- Synchronous and asynchronous clear
- Synchronous load

Global signals, general-purpose I/O (GPIO) pins, or any internal logic can drive the clock and clear control signals of an ALM register.

GPIO pins or internal logic drives the clock enable signal.

For combinational functions, the registers are bypassed and the output of the look-up table (LUT) drives directly to the outputs of an ALM.

Note: The Intel Quartus Prime software automatically configures the ALMs for optimized performance.

Figure 1-7: ALM High-Level Block Diagram for Arria V GX, GT, SX, and, ST Devices

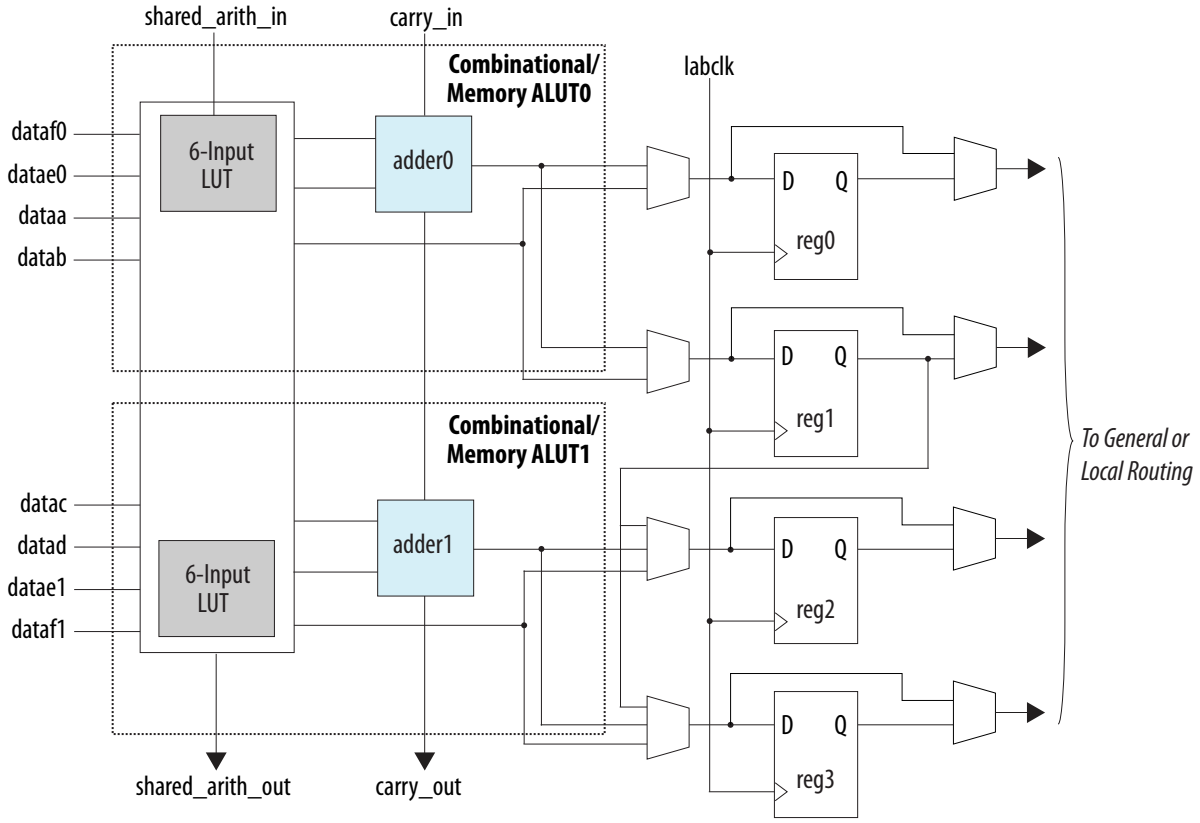
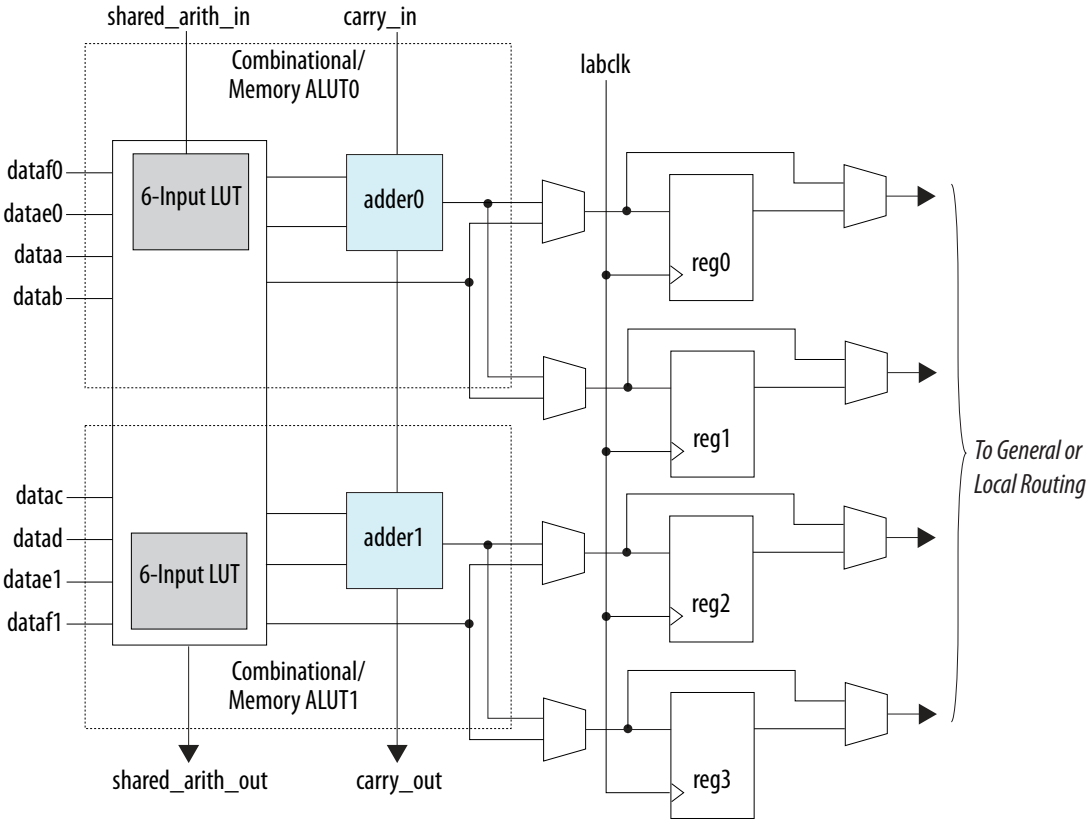


Figure 1-8: ALM High-Level Block Diagram for Arria V GZ Devices



ALM Output

The general routing outputs in each ALM drive the local, row, and column routing resources. Two ALM outputs can drive column, row, or direct link routing connections, and one of these ALM outputs can also drive local interconnect resources.

The LUT, adder, or register output can drive the ALM outputs. The LUT or adder can drive one output while the register drives another output.

Register packing improves device utilization by allowing unrelated register and combinational logic to be packed into a single ALM. Another mechanism to improve fitting is to allow the register output to feed back into the look-up table (LUT) of the same ALM so that the register is packed with its own fan-out LUT. The ALM can also drive out registered and unregistered versions of the LUT or adder output.

Figure 1-9: ALM Connection Details for Arria V GX, GT, SX, and, ST Devices

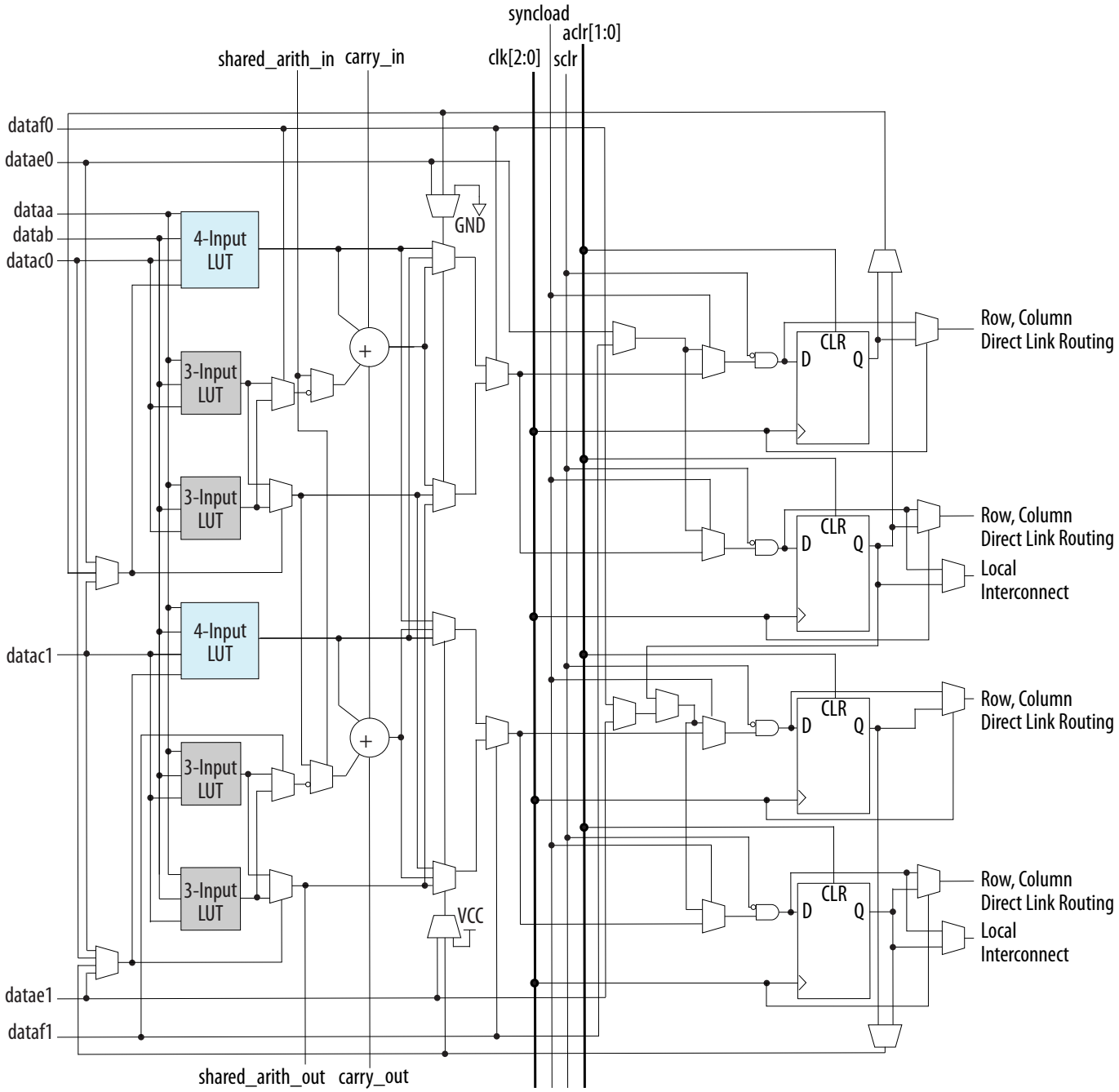
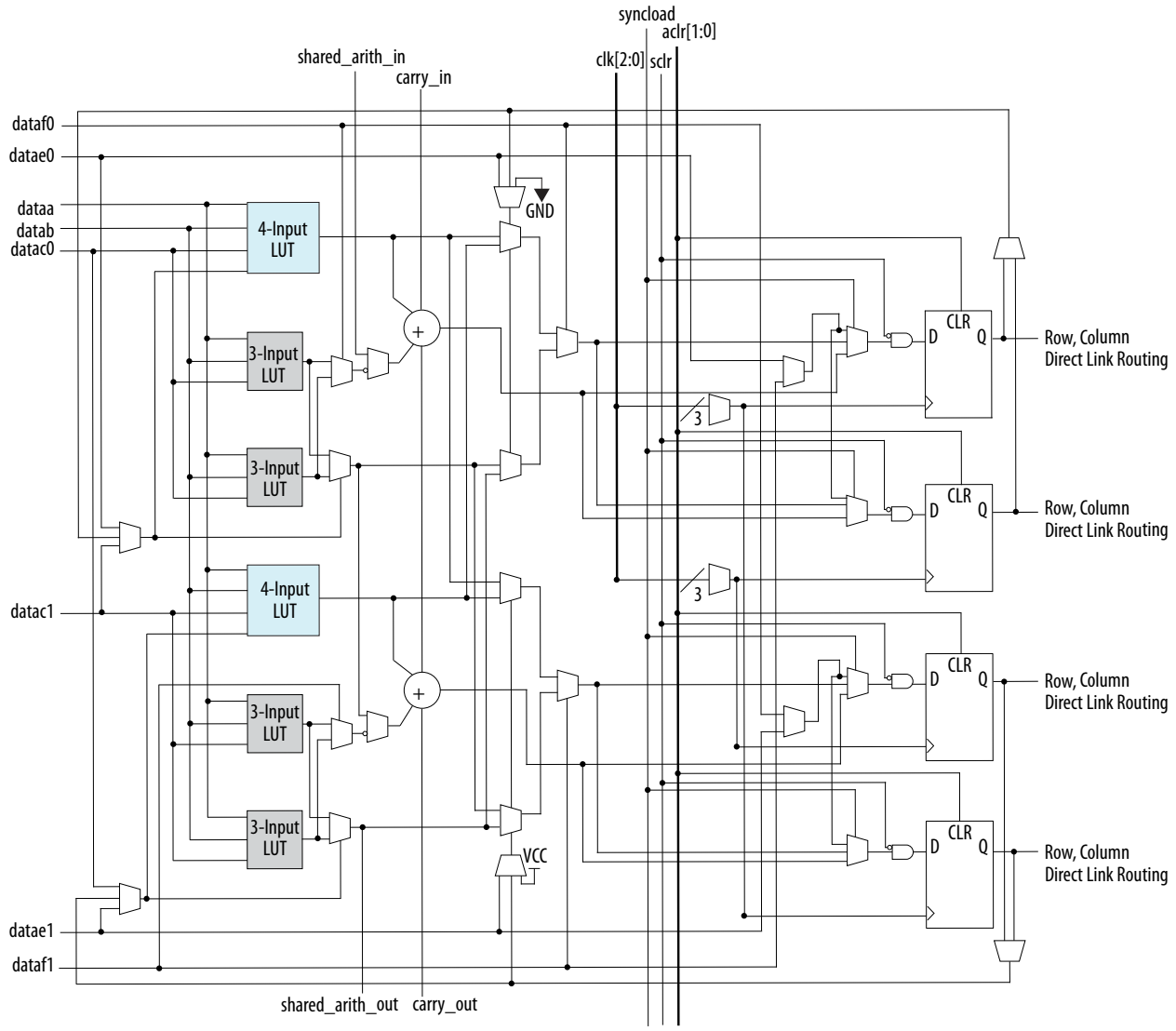


Figure 1-10: ALM Connection Details for Arria V GZ Devices



ALM Operating Modes

The Arria V ALM operates in any of the following modes:

- Normal mode
- Extended LUT mode
- Arithmetic mode
- Shared arithmetic mode

Normal Mode

Normal mode allows two functions to be implemented in one Arria V ALM, or a single function of up to six inputs.

Up to eight data inputs from the LAB local interconnect are inputs to the combinational logic.

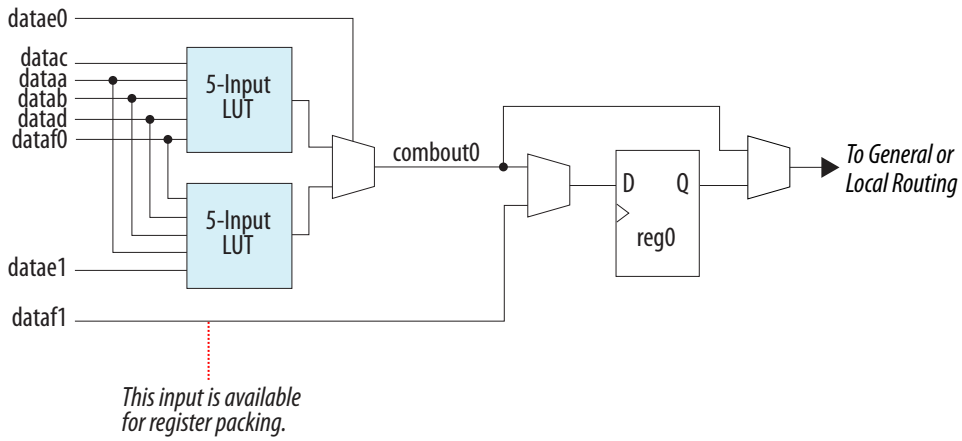
The ALM can support certain combinations of completely independent functions and various combinations of functions that have common inputs.

Extended LUT Mode

In this mode, if the 7-input function is unregistered, the unused eighth input is available for register packing.

Functions that fit into the template, as shown in the following figure, often appear in designs as "if-else" statements in Verilog HDL or VHDL code.

Figure 1-11: Template for Supported 7-Input Functions in Extended LUT Mode for Arria V Devices



Arithmetic Mode

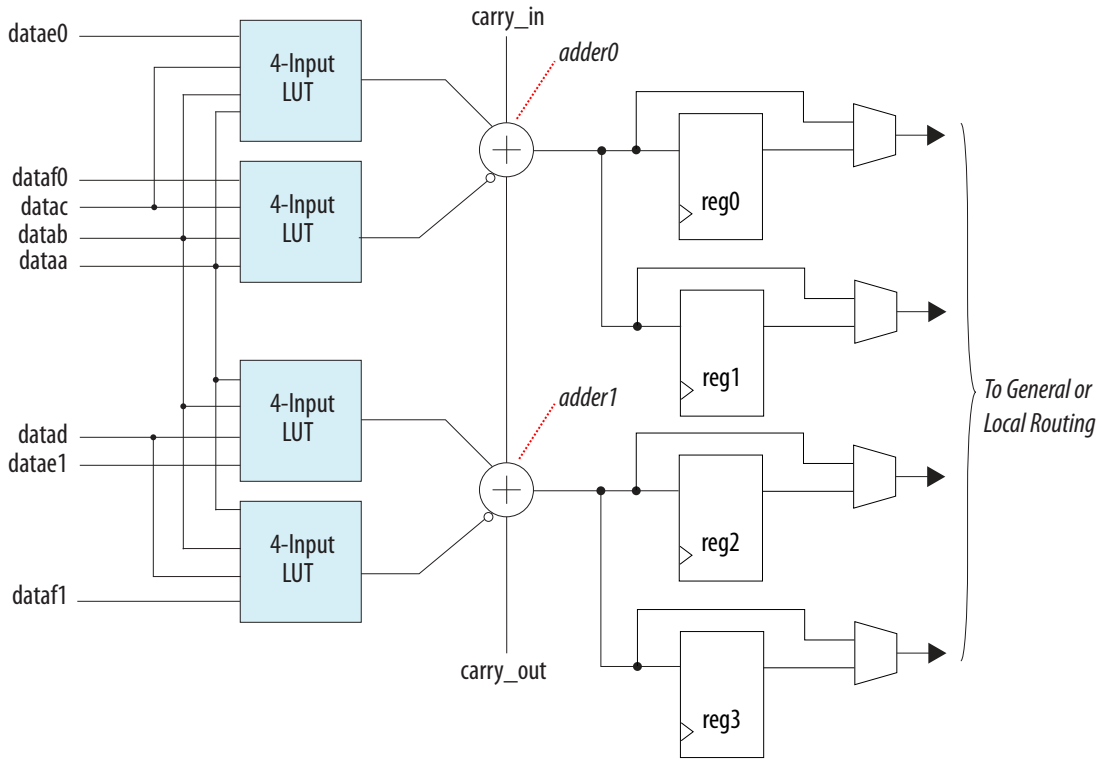
The ALM in arithmetic mode uses two sets of two 4-input LUTs along with two dedicated full adders.

The dedicated adders allow the LUTs to perform pre-adder logic; therefore, each adder can add the output of two 4-input functions.

The ALM supports simultaneous use of the adder's carry output along with combinational logic outputs. The adder output is ignored in this operation.

Using the adder with the combinational logic output provides resource savings of up to 50% for functions that can use this mode.

Figure 1-12: ALM in Arithmetic Mode for Arria V Devices



Carry Chain

The carry chain provides a fast carry function between the dedicated adders in arithmetic or shared arithmetic mode.

The two-bit carry select feature in Arria V devices halves the propagation delay of carry chains within the ALM. Carry chains can begin in either the first ALM or the fifth ALM in a LAB. The final carry-out signal is routed to an ALM, where it is fed to local, row, or column interconnects.

To avoid routing congestion in one small area of the device when a high fan-in arithmetic function is implemented, the LAB can support carry chains that only use either the top half or bottom half of the LAB before connecting to the next LAB. This leaves the other half of the ALMs in the LAB available for implementing narrower fan-in functions in normal mode. Carry chains that use the top five ALMs in the first LAB carry into the top half of the ALMs in the next LAB in the column. Carry chains that use the bottom five ALMs in the first LAB carry into the bottom half of the ALMs in the next LAB within the column. You can bypass the top-half of the LAB columns and bottom-half of the MLAB columns.

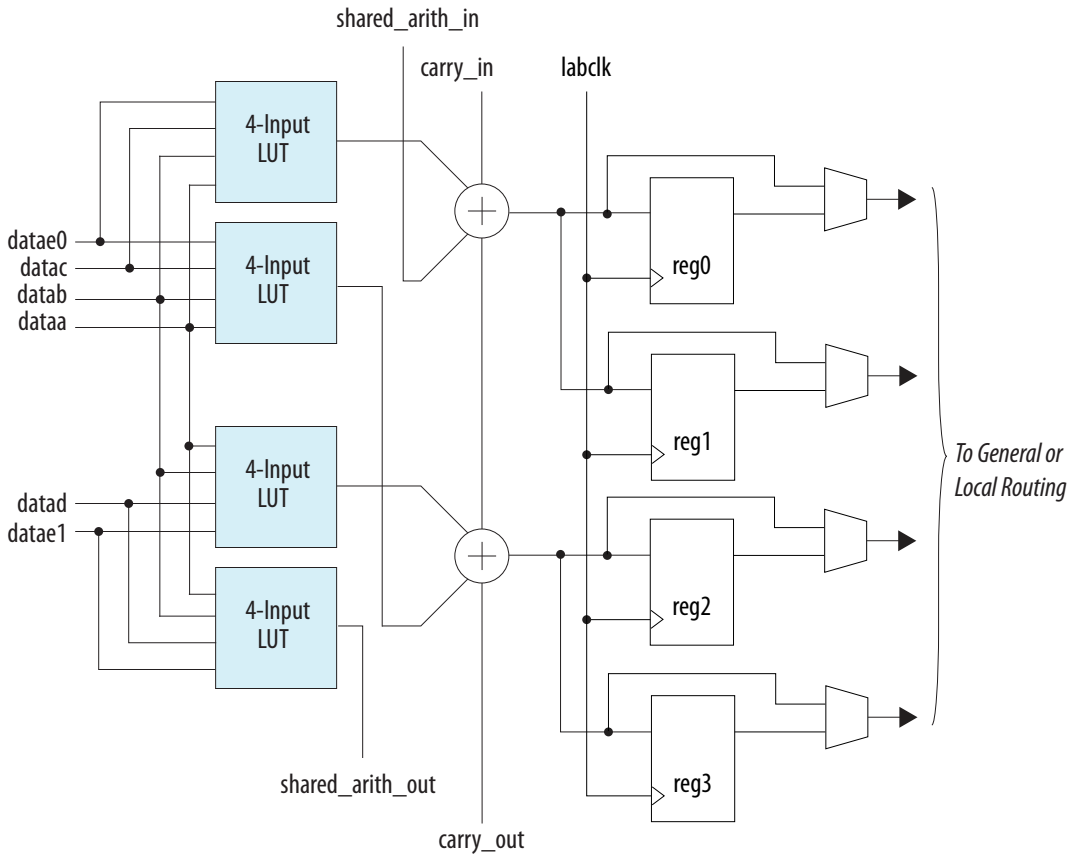
The Intel Quartus Prime Compiler creates carry chains longer than 20 ALMs (10 ALMs in arithmetic or shared arithmetic mode) by linking LABs together automatically. For enhanced fitting, a long carry chain runs vertically, allowing fast horizontal connections to the TriMatrix memory and DSP blocks. A carry chain can continue as far as a full column.

Shared Arithmetic Mode

The ALM in shared arithmetic mode can implement a 3-input add in the ALM.

This mode configures the ALM with four 4-input LUTs. Each LUT either computes the sum of three inputs or the carry of three inputs. The output of the carry computation is fed to the next adder using a dedicated connection called the shared arithmetic chain.

Figure 1-13: ALM in Shared Arithmetic Mode for Arria V Devices



Shared Arithmetic Chain

The shared arithmetic chain available in enhanced arithmetic mode allows the ALM to implement a 3-input adder. This significantly reduces the resources necessary to implement large adder trees or correlator functions.

The shared arithmetic chain can begin in either the first or sixth ALM in a LAB.

Similar to carry chains, the top and bottom half of the shared arithmetic chains in alternate LAB columns can be bypassed. This capability allows the shared arithmetic chain to cascade through half of the ALMs in an LAB while leaving the other half available for narrower fan-in functionality. In every LAB, the column is top-half bypassable; while in MLAB, columns are bottom-half bypassable.

The Intel Quartus Prime Compiler creates shared arithmetic chains longer than 20 ALMs (10 ALMs in arithmetic or shared arithmetic mode) by linking LABs together automatically. To enhance fitting, a long shared arithmetic chain runs vertically, allowing fast horizontal connections to the TriMatrix memory and DSP blocks. A shared arithmetic chain can continue as far as a full column.

Logic Array Blocks and Adaptive Logic Modules in Arria V Devices Revision History

Date	Version	Changes
December 2016	2016.12.09	Added description on clock source in the LAB Control Signals section.
December 2015	2015.12.21	Changed instances of *Quartus II* to *Quartus Prime*.
January 2014	2014.01.10	Added multiplexers for the bypass paths and register outputs in the following diagrams: • ALM High-Level Block Diagram for Arria V GX, GT, SX, and ST Devices • ALM High-Level Block Diagram for Arria V GZ Devices • Template for Supported 7-Input Functions in Extended LUT Mode for Arria V Devices • ALM in Arithmetic Mode for Arria V Devices • ALM in Shared Arithmetic Mode for Arria V Devices
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated local and direct link interconnects section to add M20K memory block. • Removed register chain outputs information in ALM output section. • Removed `reg_chain_in` and `reg_chain_out` ports in ALM high-level block diagram and ALM connection details diagram for Arria V GX, GT, SX, and ST devices.
November 2012	2012.11.19	• Added MLAB structure for Arria V GZ devices. • Added LAB-wide control signals diagram for Arria V GZ devices. • Added ALM high level block diagram for Arria V GZ devices. • Added ALM connection details diagram for Arria V GZ devices. • Reorganized content and updated template.
June 2012	2.0	Updated for the Quartus II software v12.0 release: • Restructured chapter. • Updated Figure 1–6.
November 2011	1.1	Restructured chapter.
May 2011	1.0	Initial release.

Embedded Memory Blocks in Arria V Devices

2

2020.04.13

AV-52002

Subscribe Send Feedback

The embedded memory blocks in the devices are flexible and designed to provide an optimal amount of small- and large-sized memory arrays to fit your design requirements.

Related Information

Arria V Device Handbook: Known Issues
Lists the planned updates to the *Arria V Device Handbook* chapters.

Types of Embedded Memory

The Arria V devices contain two types of memory blocks:

- 20 Kb M20K or 10 Kb M10K blocks—blocks of dedicated memory resources. The M20K and M10K blocks are ideal for larger memory arrays while still providing a large number of independent ports.
- 640 bit memory logic array blocks (MLABs)—enhanced memory blocks that are configured from dual-purpose logic array blocks (LABs). The MLABs are ideal for wide and shallow memory arrays. The MLABs are optimized for implementation of shift registers for digital signal processing (DSP) applications, wide shallow FIFO buffers, and filter delay lines. Each MLAB is made up of ten adaptive logic modules (ALMs). In the Arria V devices, you can configure these ALMs as ten 32 x 2 blocks, giving you one 32 x 20 simple dual-port SRAM block per MLAB. You can also configure these ALMs, in Arria V GZ devices, as ten 64 x 1 blocks, giving you one 64 x 10 simple dual-port SRAM block per MLAB.

Embedded Memory Capacity in Arria V Devices

Table 2-1: Embedded Memory Capacity and Distribution in Arria V Devices

Variant	Member Code	M20K		M10K		MLAB		Total RAM Bit (Kb)
		Block	RAM Bit (Kb)	Block	RAM Bit (Kb)	Block	RAM Bit (Kb)	
Arria V GX	A1	—	—	800	8,000	741	463	8,463
	A3	—	—	1,051	10,510	1538	961	11,471
	A5	—	—	1,180	11,800	1877	1,173	12,973
	A7	—	—	1,366	13,660	2317	1,448	15,108
	B1	—	—	1,510	15,100	2964	1,852	16,952
	B3	—	—	1,726	17,260	3357	2,098	19,358
	B5	—	—	2,054	20,540	4052	2,532	23,072
	B7	—	—	2,414	24,140	4650	2,906	27,046
Arria V GT	C3	—	—	1,051	10,510	1538	961	11,471
	C7	—	—	1,366	13,660	2317	1,448	15,108
	D3	—	—	1,726	17,260	3357	2,098	19,358
	D7	—	—	2,414	24,140	4650	2,906	27,046
Arria V GZ	E1	585	11,700	—	—	4,151	2,594	14,294
	E3	957	19,140	—	—	6,792	4,245	23,385
	E5	1,440	28,800	—	—	7,548	4,718	33,518
	E7	1,700	34,000	—	—	8,490	5,306	39,306
Arria V SX	B3	—	—	1,729	17,290	3223	2,014	19,304
	B5	—	—	2,282	22,820	4253	2,658	25,478
Arria V ST	D3	—	—	1,729	17,290	3223	2,014	19,304
	D5	—	—	2,282	22,820	4253	2,658	25,478

Embedded Memory Design Guidelines for Arria V Devices

There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

Guideline: Consider the Memory Block Selection

The Intel Quartus Prime software automatically partitions the user-defined memory into the memory blocks based on your design's speed and size constraints. For example, the Intel Quartus Prime software may spread out the memory across multiple available memory blocks to increase the performance of the design.

To assign the memory to a specific block size manually, use the RAM IP core in the IP Catalog.

For the memory logic array blocks (MLAB), you can implement single-port SRAM through emulation using the Intel Quartus Prime software. Emulation results in minimal additional use of logic resources.

Because of the dual-purpose architecture of the MLAB, only data input and output registers are available in the block. The MLABs gain read address registers from the ALMs. However, the write address and read data registers are internal to the MLABs.

Guideline: Implement External Conflict Resolution

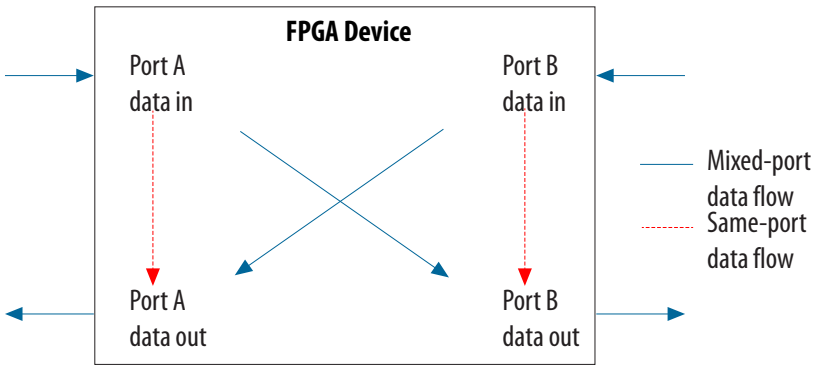
In the true dual-port RAM mode, you can perform two write operations to the same memory location. However, the memory blocks do not have internal conflict resolution circuitry. To avoid unknown data being written to the address, implement external conflict resolution logic to the memory block.

Guideline: Customize Read-During-Write Behavior

Customize the read-during-write behavior of the memory blocks to suit your design requirements.

Figure 2-1: Read-During-Write Data Flow

This figure shows the difference between the two types of read-during-write operations available—same port and mixed port.



Same-Port Read-During-Write Mode

The same-port read-during-write mode applies to a single-port RAM or the same port of a true dual-port RAM.

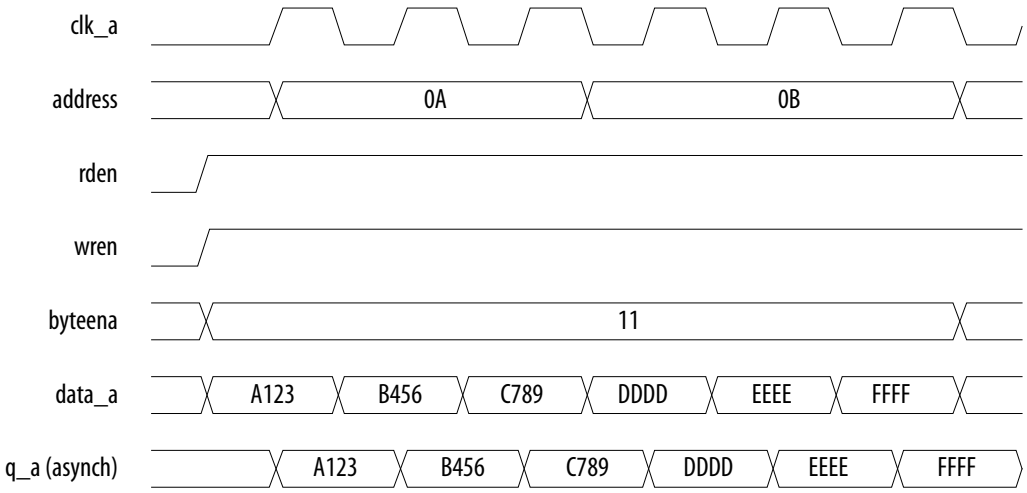
Table 2-2: Output Modes for Embedded Memory Blocks in Same-Port Read-During-Write Mode

This table lists the available output modes if you select the embedded memory blocks in the same-port read-during-write mode.

Output Mode	Memory Type	Description
"new data" (flow-through)	M20K, M10K	The new data is available on the rising edge of the same clock cycle on which the new data is written.
"don't care"	M10K, MLAB	The RAM outputs "don't care" values for a read-during-write operation.

Figure 2-2: Same-Port Read-During-Write: New Data Mode

This figure shows sample functional waveforms of same-port read-during-write behavior in the "new data" mode.



Mixed-Port Read-During-Write Mode

The mixed-port read-during-write mode applies to simple and true dual-port RAM modes where two ports perform read and write operations on the same memory address using the same clock—one port reading from the address, and the other port writing to it.

Table 2-3: Output Modes for RAM in Mixed-Port Read-During-Write Mode

Output Mode	Memory Type	Description
"new data"	MLAB	A read-during-write operation to different ports causes the MLAB registered output to reflect the "new data" on the next rising edge after the data is written to the MLAB memory. This mode is available only if the output is registered.
"old data"	M20K, M10K, MLAB	A read-during-write operation to different ports causes the RAM output to reflect the "old data" value at the particular address. For MLAB, this mode is available only if the output is registered.

Output Mode	Memory Type	Description
"don't care"	M20K, M10K, MLAB	The RAM outputs “don’t care” or “unknown” value. • For M20K or M10K memory, the Intel Quartus Prime software does not analyze the timing between write and read operations. • For MLAB, the Intel Quartus Prime software analyzes the timing between write and read operations by default. To disable this behavior, turn on the **Do not analyze the timing between write and read operation. Metastability issues are prevented by never writing and reading at the same address at the same time** option.
"constrained don't care"	MLAB	The RAM outputs “don’t care” or “unknown” value. The Intel Quartus Prime software analyzes the timing between write and read operations in the MLAB.

Figure 2-3: Mixed-Port Read-During-Write: New Data Mode

This figure shows a sample functional waveform of mixed-port read-during-write behavior for the “new data” mode.

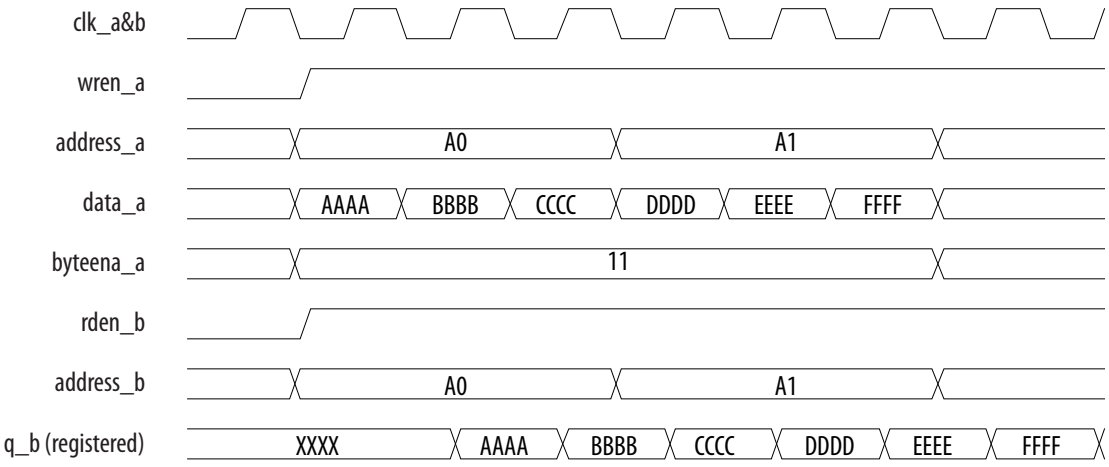


Figure 2-4: Mixed-Port Read-During-Write: Old Data Mode

This figure shows a sample functional waveform of mixed-port read-during-write behavior for the "old data" mode.

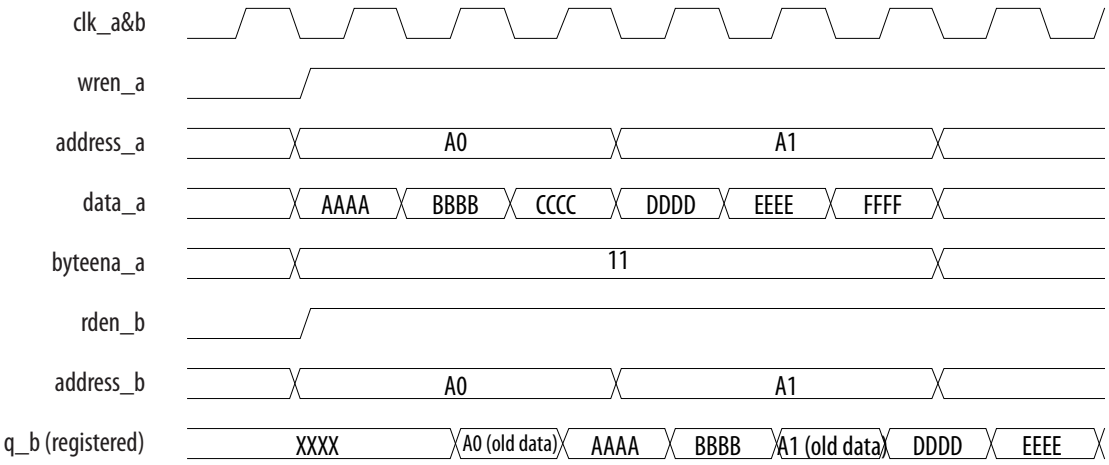
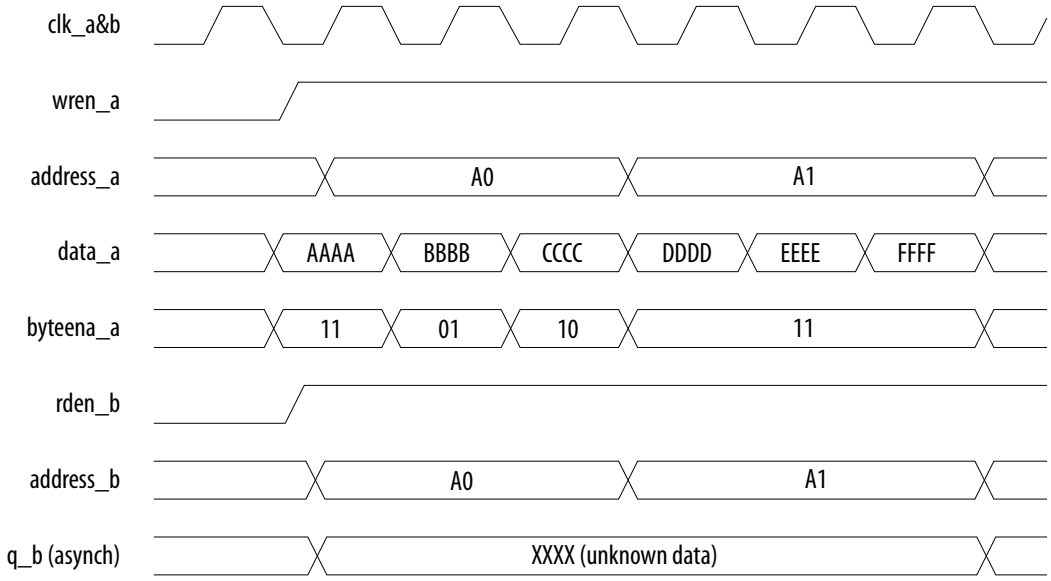


Figure 2-5: Mixed-Port Read-During-Write: Don't Care or Constrained Don't Care Mode

This figure shows a sample functional waveform of mixed-port read-during-write behavior for the "don't care" or "constrained don't care" mode.



In the dual-port RAM mode, the mixed-port read-during-write operation is supported if the input registers have the same clock. The output value during the operation is "unknown."

Related Information

Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide
Provides more information about the RAM IP core that controls the read-during-write behavior.

Guideline: Consider Power-Up State and Memory Initialization

Consider the power up state of the different types of memory blocks if you are designing logic that evaluates the initial power-up values, as listed in the following table.

Table 2-4: Initial Power-Up Values of Embedded Memory Blocks

Memory Type	Output Registers	Power Up Value
MLAB	Used	Zero (cleared)
	Bypassed	Read memory contents
M20K, M10K	Used	Zero (cleared)
	Bypassed	Zero (cleared)

By default, the Intel Quartus Prime software initializes the RAM cells in Arria V devices to zero unless you specify a **.mif**.

All memory blocks support initialization with a **.mif**. You can create **.mif** files in the Intel Quartus Prime software and specify their use with the RAM IP core when you instantiate a memory in your design. Even if a memory is pre-initialized (for example, using a **.mif**), it still powers up with its output cleared.

Related Information

- **Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide**
  Provides more information about **.mif** files.
- **Quartus II Handbook**
  Provides more information about **.mif** files.

Guideline: Control Clocking to Reduce Power Consumption

Reduce AC power consumption in your design by controlling the clocking of each memory block:

- Use the read-enable signal to ensure that read operations occur only when necessary. If your design does not require read-during-write, you can reduce your power consumption by de-asserting the read-enable signal during write operations, or during the period when no memory operations occur.
- Use the Intel Quartus Prime software to automatically place any unused memory blocks in low-power mode to reduce static power.

Embedded Memory Features

Table 2-5: Memory Features in Arria V Devices

This table summarizes the features supported by the embedded memory blocks.

Features	M20K, M10K	MLAB
Maximum operating frequency	M20K—600 MHz M10K—400 MHz	Arria V GX, GT, SX, and ST—500 MHz Arria V GZ—600 MHz
Capacity per block (including parity bits)	M20K—20,480 M10K—10,240	640
Parity bits	Supported	Supported
Byte enable	Supported	Supported
Packed mode	Supported	—
Address clock enable	Supported	Supported
Simple dual-port mixed width	Supported	—
True dual-port mixed width	Supported	—
FIFO buffer mixed width	Supported	—
Memory Initialization File (.mif)	Supported	Supported
Mixed-clock mode	Supported	Supported
Fully synchronous memory	Supported	Supported
Asynchronous memory	—	Only for flow-through read memory operations.
Power-up state	Output ports are cleared.	• Registered output ports—Cleared. • Unregistered output ports—Read memory contents.
Asynchronous clears	Output registers and output latches	Output registers and output latches
Write/read operation triggering	Rising clock edges	Rising clock edges
Same-port read-during-write	• M20K—output ports set to new data • M10K—output ports set to "new data" or "don't care" (The "don't care" mode applies only for the single-port RAM mode).	Output ports set to "don't care".

Features	M20K, M10K	MLAB
Mixed-port read-during-write	Output ports set to "old data" or "don't care".	Output ports set to "old data", "new data", "don't care", or "constrained don't care".
ECC support	Soft IP support using the Intel Quartus Prime software. Built-in support in x32-wide simple dual-port mode (M20K only).	Soft IP support using the Intel Quartus Prime software.

Related Information

Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide
Provides more information about the embedded memory features.

Embedded Memory Configurations

Table 2-6: Supported Embedded Memory Block Configurations for Arria V Devices

This table lists the maximum configurations supported for the embedded memory blocks. The information is applicable only to the single-port RAM and ROM modes.

Memory Block	Depth (bits)	Programmable Width
MLAB	32	x16, x18, or x20
	64(1)	x10
M20K	512	x40
	1K	x20
	2K	x10
	4K	x5
	8K	x2
	16K	x1
M10K	256	x40 or x32
	512	x20 or x16
	1K	x10 or x8
	2K	x5 or x4
	4K	x2
	8K	x1

(1) Available for Arria V GZ devices only.

Mixed-Width Port Configurations

The mixed-width port configuration is supported in the simple dual-port RAM and true dual-port RAM memory modes.

Note: MLABs do not support mixed-width port configurations.

Related Information

Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide
Provides more information about dual-port mixed width support.

M20K Blocks Mixed-Width Configurations

The following table lists the mixed-width configurations of the M20K blocks in the simple dual-port RAM mode.

Table 2-7: M20K Block Mixed-Width Configurations (Simple Dual-Port RAM Mode)

Read Port	Write Port									
	16K x 1	8K x 2	4K x 4	4K x 5	2K x 8	2K x 10	1K x 16	1K x 20	512 x 32	512 x 40
16K x 1	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
8K x 2	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
4K x 4	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
4K x 5	—	—	—	Yes	—	Yes	—	Yes	—	Yes
2K x 8	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
2K x 10	—	—	—	Yes	—	Yes	—	Yes	—	Yes
1K x 16	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
1K x 20	—	—	—	Yes	—	Yes	—	Yes	—	Yes
512 x 32	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
512 x 40	—	—	—	Yes	—	Yes	—	Yes	—	Yes

The following table lists the mixed-width configurations of the M20K blocks in true dual-port mode.

Table 2-8: M20K Block Mixed-Width Configurations (True Dual-Port Mode)

Port A	Port B							
	16K x 1	8K x 2	4K x 4	4K x 5	2K x 8	2K x 10	1K x 16	1K x 20
16K x 1	Yes	Yes	Yes	—	Yes	—	Yes	—
8K x 2	Yes	Yes	Yes	—	Yes	—	Yes	—
4K x 4	Yes	Yes	Yes	—	Yes	—	Yes	—
4K x 5	—	—	—	Yes	—	Yes	—	Yes
2K x 8	Yes	Yes	Yes	—	Yes	—	Yes	—
2K x 10	—	—	—	Yes	—	Yes	—	Yes
1K x 16	Yes	Yes	Yes	—	Yes	—	Yes	—
1K x 20	—	—	—	Yes	—	Yes	—	Yes

M10K Blocks Mixed-Width Configurations

Table 2-9: M10K Block Mixed-Width Configurations in Simple Dual-Port RAM Mode

Read Port	Write Port									
	8K x 1	4K x 2	2K x 4	2K x 5	1K x 8	1k x 10	512 x 16	512 x 20	256 x 32	256 x 40
8K x 1	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
4K x 2	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
2K x 4	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
2K x 5	—	—	—	Yes	—	Yes	—	Yes	—	Yes
1K x 8	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
1K x 10	—	—	—	Yes	—	Yes	—	Yes	—	Yes
512 x 16	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
512 x 20	—	—	—	Yes	—	Yes	—	Yes	—	Yes
256 x 32	Yes	Yes	Yes	—	Yes	—	Yes	—	Yes	—
256 x 40	—	—	—	Yes	—	Yes	—	Yes	—	Yes

Table 2-10: M10K Block Mixed-Width Configurations in True Dual-Port Mode

Port B	Port A							
	8K x 1	4K x 2	2K x 4	2K x 5	1K x 8	1K x 10	512 x 16	512 x 20
8K x 1	Yes	Yes	Yes	—	Yes	—	Yes	—
4K x 2	Yes	Yes	Yes	—	Yes	—	Yes	—
2K x 4	Yes	Yes	Yes	—	Yes	—	Yes	—
2K x 5	—	—	—	Yes	—	Yes	—	Yes
1K x 8	Yes	Yes	Yes	—	Yes	—	Yes	—
1K x 10	—	—	—	Yes	—	Yes	—	Yes
512 x 16	Yes	Yes	Yes	—	Yes	—	Yes	—
512 x 20	—	—	—	Yes	—	Yes	—	Yes

Embedded Memory Modes

Caution: To avoid corrupting the memory contents, do not violate the setup or hold time on any of the memory block input registers during read or write operations. This is applicable if you use the memory blocks in single-port RAM, simple dual-port RAM, true dual-port RAM, or ROM mode.

Table 2-11: Memory Modes Supported in the Embedded Memory Blocks

This table lists and describes the memory modes that are supported in the Arria V embedded memory blocks.

Memory Mode	M20K and M10K Support	MLAB Support	Description
Single-port RAM	Yes	Yes	You can perform only one read or one write operation at a time. Use the read enable port to control the RAM output ports behavior during a write operation: • To retain the previous values that are held during the most recent active read enable—create a read-enable port and perform the write operation with the read enable port deasserted. • To show the new data being written, the old data at that address, or a "Don't Care" value when read-during-write occurs at the same address location—do not create a read-enable signal, or activate the read enable during a write operation.

Memory Mode	M20K and M10K Support	MLAB Support	Description
Simple dual-port RAM	Yes	Yes	You can simultaneously perform one read and one write operations to different locations where the write operation happens on port A and the read operation happens on port B.
True dual-port RAM	Yes	—	You can perform any combination of two port operations: two reads, two writes, or one read and one write at two different clock frequencies.
Shift-register	Yes	Yes	You can use the memory blocks as a shift-register block to save logic cells and routing resources. This is useful in DSP applications that require local data storage such as finite impulse response (FIR) filters, pseudo-random number generators, multi-channel filtering, and auto- and cross- correlation functions. Traditionally, the local data storage is implemented with standard flip-flops that exhaust many logic cells for large shift registers. The input data width (w), the length of the taps (m), and the number of taps (n) determine the size of a shift register ($w \times m \times n$). You can cascade memory blocks to implement larger shift registers.
ROM	Yes	Yes	You can use the memory blocks as ROM. • Initialize the ROM contents of the memory blocks using a **.mif** or **.hex**. • The address lines of the ROM are registered on M20K or M10K blocks but can be unregistered on MLABs. • The outputs can be registered or unregistered. • The output registers can be asynchronously cleared. • The ROM read operation is identical to the read operation in the single-port RAM configuration.
FIFO	Yes	Yes	You can use the memory blocks as FIFO buffers. Use the SCFIFO and DCFIFO IP cores to implement single- and dual-clock asynchronous FIFO buffers in your design. For designs with many small and shallow FIFO buffers, the MLABs are ideal for the FIFO mode. However, the MLABs do not support mixed-width FIFO mode.

Related Information

- **Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide**
  Provides more information memory modes.

- **RAM-Based Shift Register (ALTSHIFT_TAPS) IP Core User Guide**
  Provides more information about implementing the shift register mode.
- **SCFIFO and DCFIFO IP Core User Guide**
  Provides more information about implementing FIFO buffers.

Embedded Memory Clocking Modes

This section describes the clocking modes for the Arria V memory blocks.

Caution: To avoid corrupting the memory contents, do not violate the setup or hold time on any of the memory block input registers during read or write operations.

Clocking Modes for Each Memory Mode

Table 2-12: Memory Blocks Clocking Modes Supported for Each Memory Mode

Clocking Mode	Memory Mode				
	Single-Port	Simple Dual-Port	True Dual-Port	ROM	FIFO
Single clock mode	Yes	Yes	Yes	Yes	Yes
Read/write clock mode	—	Yes	—	—	Yes
Input/output clock mode	Yes	Yes	Yes	Yes	—
Independent clock mode	—	—	Yes	Yes	—

Note: The clock enable signals are not supported for write address, byte enable, and data input registers on MLAB blocks.

Single Clock Mode

In the single clock mode, a single clock, together with a clock enable, controls all registers of the memory block.

Read/Write Clock Mode

In the read/write clock mode, a separate clock is available for each read and write port. A read clock controls the data-output, read-address, and read-enable registers. A write clock controls the data-input, write-address, write-enable, and byte enable registers.

Input/Output Clock Mode

In input/output clock mode, a separate clock is available for each input and output port. An input clock controls all registers related to the data input to the memory block including data, address, byte enables, read enables, and write enables. An output clock controls the data output registers.

Independent Clock Mode

In the independent clock mode, a separate clock is available for each port (A and B). Clock A controls all registers on the port A side; clock B controls all registers on the port B side.

Note: You can create independent clock enable for different input and output registers to control the shut down of a particular register for power saving purposes. From the parameter editor, click **More Options** (beside the clock enable option) to set the available independent clock enable that you prefer.

Asynchronous Clears in Clocking Modes

In all clocking modes, asynchronous clears are available only for output latches and output registers. For the independent clock mode, this is applicable on both ports.

Output Read Data in Simultaneous Read/Write

If you perform a simultaneous read/write to the same address location using the read/write clock mode, the output read data is unknown. If you require the output read data to be a known value, use single-clock or input/output clock mode and select the appropriate read-during-write behavior in the IP Catalog.

Note: MLAB memory blocks only support simultaneous read/write operations when operating in single clock mode.

Independent Clock Enables in Clocking Modes

Independent clock enables are supported in the following clocking modes:

- Read/write clock mode—supported for both the read and write clocks.
- Independent clock mode—supported for the registers of both ports.

To save power, you can control the shut down of a particular register using the clock enables.

Related Information

Guideline: Control Clocking to Reduce Power Consumption on page 2-7

Parity Bit in Memory Blocks

Table 2-13: Parity Bit Support for the Embedded Memory Blocks

This table describes the parity bit support for the memory blocks.

M20K, M10K	MLAB
• The parity bit is the fifth bit associated with each 4 data bits in data widths of 5, 10, 20, and 40 (bits 4, 9, 14, 19, 24, 29, 34, and 39). • In non-parity data widths, the parity bits are skipped during read or write operations. • Parity function is not performed on the parity bit.	• The parity bit is the ninth bit associated with each byte. • The ninth bit can store a parity bit or serve as an additional bit. • Parity function is not performed on the parity bit.

Byte Enable in Embedded Memory Blocks

The embedded memory blocks support byte enable controls:

- The byte enable controls mask the input data so that only specific bytes of data are written. The unwritten bytes retain the values written previously.
- The write enable (`wren`) signal, together with the byte enable (`byteena`) signal, control the write operations on the RAM blocks. By default, the `byteena` signal is high (enabled) and only the `wren` signal controls the writing.
- The byte enable registers do not have a `clear` port.
- If you are using parity bits, on the M20K and M10K blocks, the byte enable function controls 8 data bits and 2 parity bits; on the MLABs, the byte enable function controls all 10 bits in the widest mode.
- The MSB and LSB of the `byteena` signal correspond to the MSB and LSB of the data bus, respectively.
- The byte enables are active high.

Byte Enable Controls in Memory Blocks

Table 2-14: `byteena` Controls in x20 Data Width

byteena[1:0]	Data Bits Written	
11 (default)	[19:10]	[9:0]
10	[19:10]	—
01	—	[9:0]

Table 2-15: `byteena` Controls in x40 Data Width

byteena[3:0]	Data Bits Written			
1111 (default)	[39:30]	[29:20]	[19:10]	[9:0]
1000	[39:30]	—	—	—
0100	—	[29:20]	—	—
0010	—	—	[19:10]	—
0001	—	—	—	[9:0]

Note: If you use the ECC feature on the M20K blocks, you cannot use the byte enable feature.

Data Byte Output

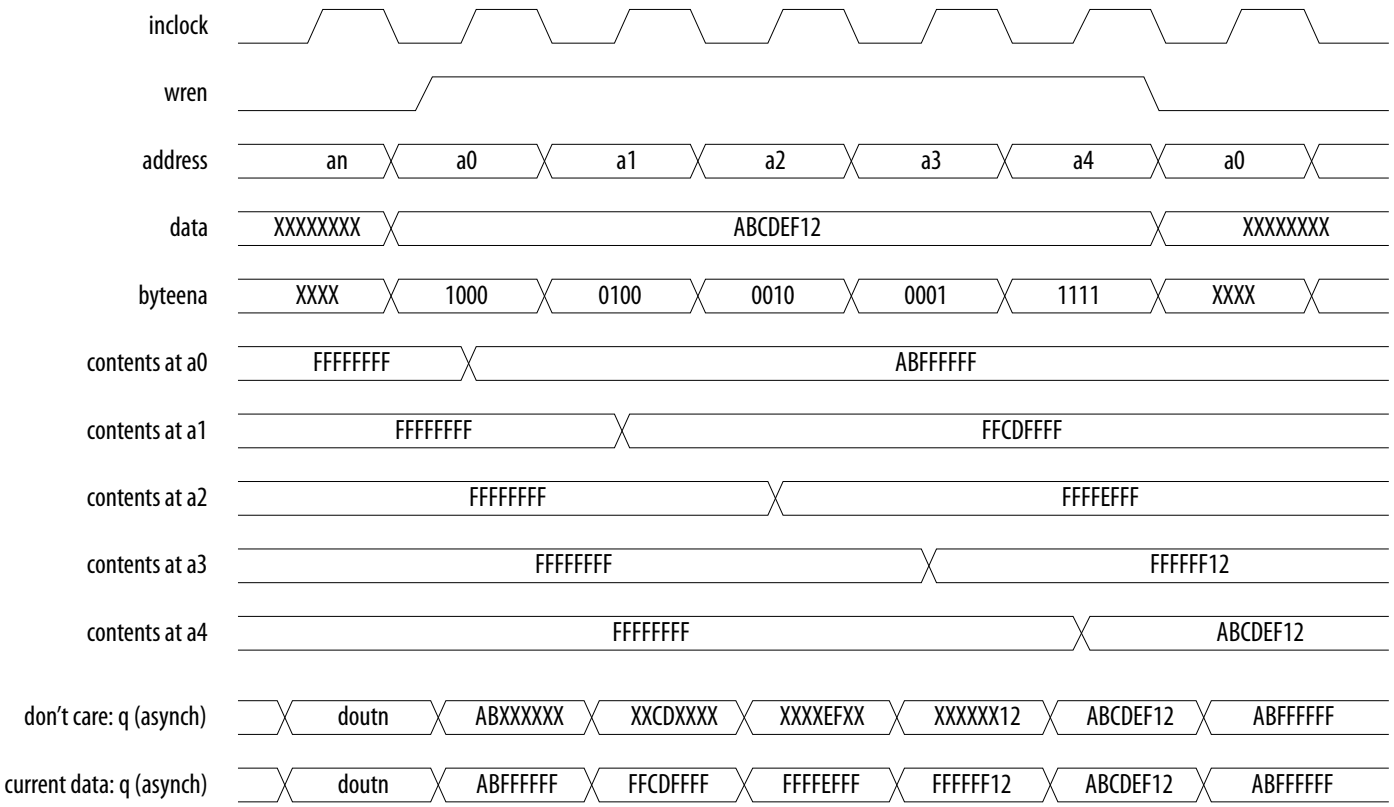
In M10K blocks, the corresponding masked data byte output appears as a “don’t care” value.

In M20K blocks or MLABs, when you de-assert a byte-enable bit during a write cycle, the corresponding data byte output appears as either a “don't care” value or the current data at that location. You can control the output value for the masked byte in the M20K blocks or MLABs by using the Intel Quartus Prime software.

RAM Blocks Operations

Figure 2-6: Byte Enable Functional Waveform

This figure shows how the `wren` and `byteena` signals control the operations of the RAM blocks. For the M10K blocks, the write-masked data byte output appears as a "don't care" value because the "current data" value is not supported.



Memory Blocks Packed Mode Support

The M20K and M10K memory blocks support packed mode.

The packed mode feature packs two independent single-port RAM blocks into one memory block. The Intel Quartus Prime software automatically implements packed mode where appropriate by placing the physical RAM block in true dual-port mode and using the MSB of the address to distinguish between the two logical RAM blocks. The size of each independent single-port RAM must not exceed half of the target block size.

Memory Blocks Address Clock Enable Support

The embedded memory blocks support address clock enable, which holds the previous address value for as long as the signal is enabled (`addressstall = 1`). When the memory blocks are configured in dual-port mode, each port has its own independent address clock enable. The default value for the address clock enable signal is low (disabled).

Figure 2-7: Address Clock Enable

This figure shows an address clock enable block diagram. The address clock enable is referred to by the port name `addressstall`.

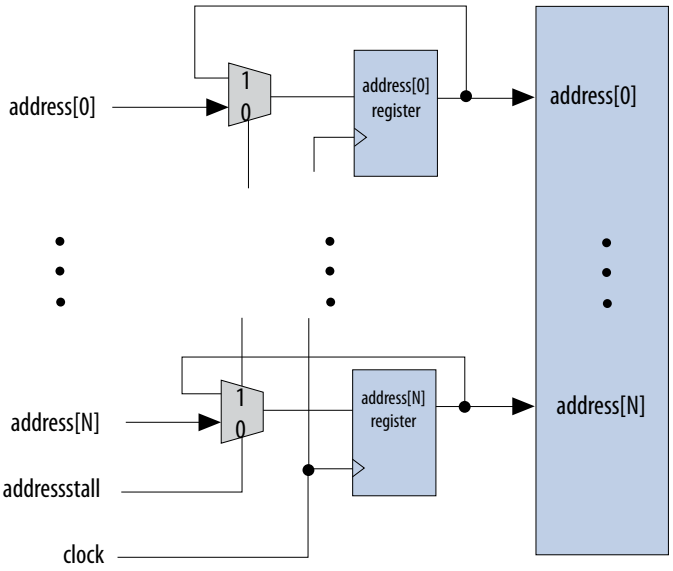


Figure 2-8: Address Clock Enable During Read Cycle Waveform

This figure shows the address clock enable waveform during the read cycle.

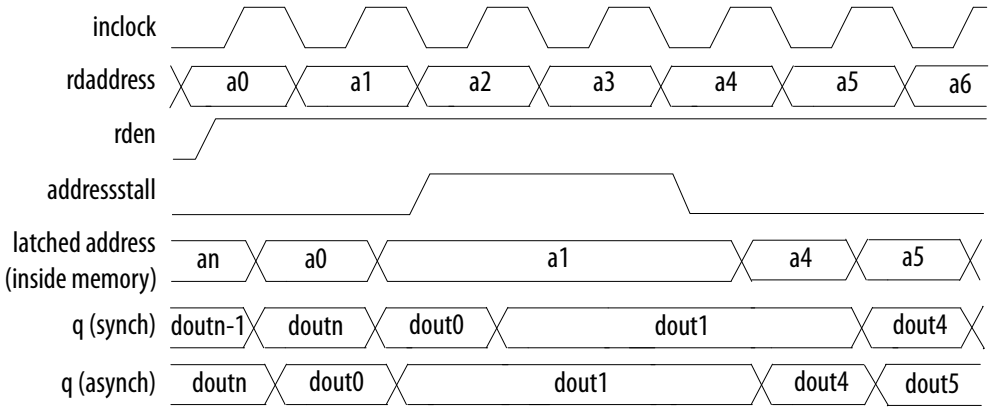
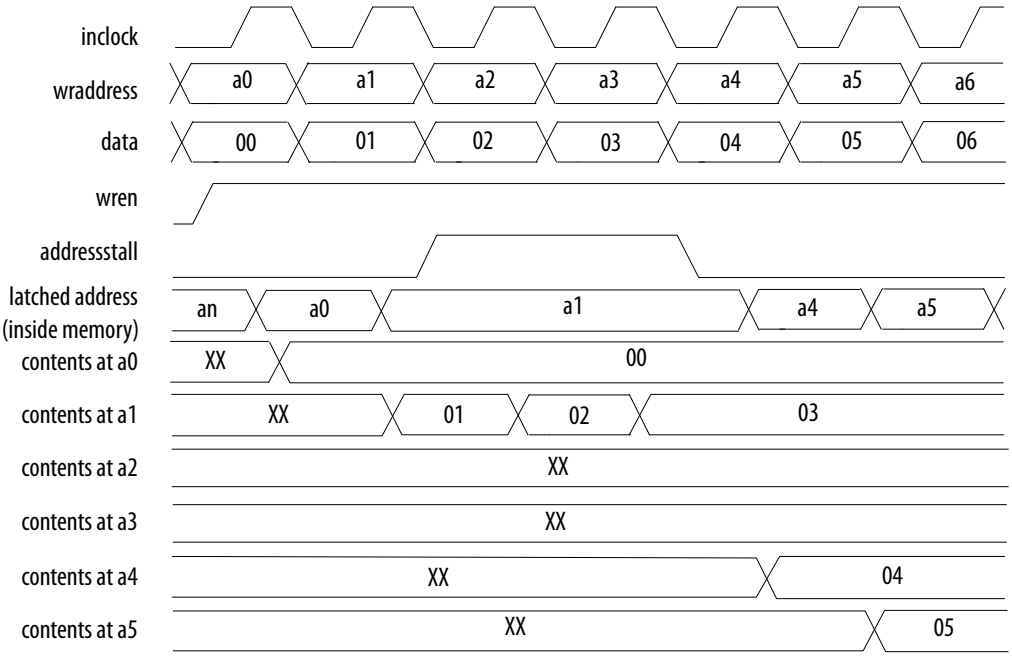


Figure 2-9: Address Clock Enable During the Write Cycle Waveform

This figure shows the address clock enable waveform during the write cycle.



Memory Blocks Error Correction Code Support

ECC allows you to detect and correct data errors at the output of the memory. ECC can perform single-error correction, double-adjacent-error correction, and triple-adjacent-error detection in a 32-bit word. However, ECC cannot detect four or more errors.

The M20K blocks have built-in support for ECC when in x32-wide simple dual-port mode:

- The M20K runs slower than non-ECC simple-dual port mode when ECC is engaged. However, you can enable optional ECC pipeline registers before the output decoder to achieve the same performance as non-ECC simple-dual port mode at the expense of one cycle of latency.
- The M20K ECC status is communicated with two ECC status flag signals—e (error) and ue (uncorrectable error). The status flags are part of the regular output from the memory block. When ECC is engaged, you cannot access two of the parity bits because the ECC status flag replaces them.

Error Correction Code Truth Table

Table 2-16: ECC Status Flags Truth Table

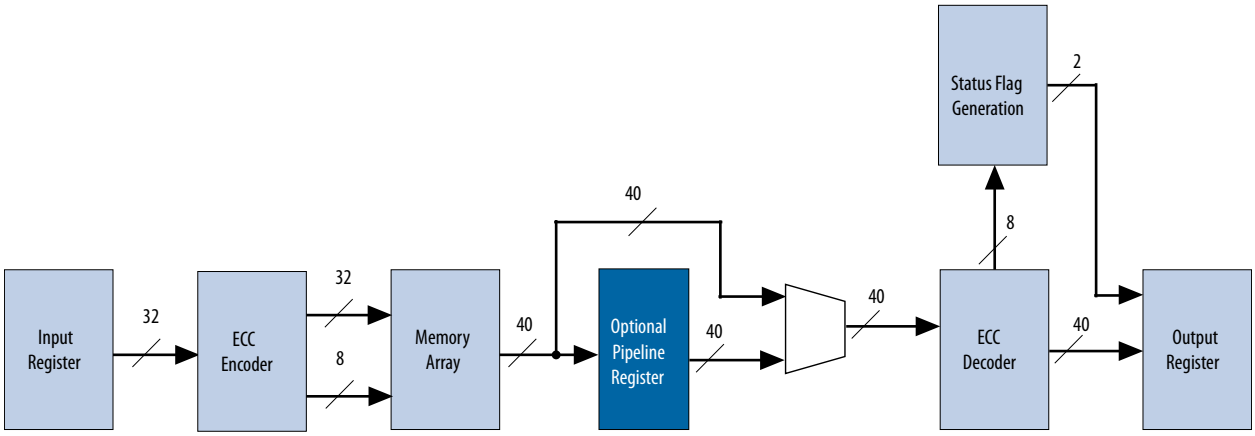
e (error) `eccstatus[1]`	ue (uncorrectable error) `eccstatus[0]`	Status
0	0	No error.
0	1	Illegal.

e (error) eccstatus[1]	ue (uncorrectable error) eccstatus[0]	Status
1	0	A correctable error occurred and the error has been corrected at the outputs; however, the memory array has not been updated.
1	1	An uncorrectable error occurred and uncorrectable data appears at the outputs.

If you engage ECC:

- You cannot use the byte enable feature.
- Read-during-write old data mode is not supported.

Figure 2-10: ECC Block Diagram for M20K Memory



Embedded Memory Blocks in Arria V Devices Revision History

Date	Version	Changes
December 2015	2015.12.21	Changed instances of *Quartus II* to *Quartus Prime*.
January 2015	2015.01.23	• Reword Total RAM bits in Memory Features in Arria V Devices table to Capacity per Block.
June 2014	2014.06.30	Added information about MLAB memory blocks support for simultaneous read/write operations. MLAB memory blocks only support simultaneous read/write operations when operating in single clock mode.

Date	Version	Changes
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Corrected the description about the "don't care" output mode for RAM in mixed-port read-during-write. • Reorganized the structure of the supported memory configurations topics (single-port and mixed-width dual-port) to improve clarity about maximum data widths supported for each configuration. • Added a description to the table listing the maximum embedded memory configurations to clarify that the information applies only to the single port or ROM mode. • Removed the topic about mixed-width configurations for MLABs and added a note to clarify that MLABs do not support mixed-width configuration.
November 2012	2012.11.19	• Reorganized content and updated template. • Added information for Arria V GZ including M20K memory, memory features, and memory capacity. • Added and updated memory capacity information from the *Arria V Device Overview* for easy reference. • Moved information about supported memory block configurations into its own table. • Added short descriptions of each clocking mode. • Added topic about the packed mode support. • Added topic about the address clock enable support. • Added topic about ECC support and the ECC truth table.
June 2012	2.0	• Restructured the chapter. • Updated the “Memory Modes”, “Clocking Modes”, and “Design Considerations” sections. • Updated Table 2–1. • Added the “Parity Bit” and “Byte Enable” sections. • Moved the memory capacity information to the *Arria V Device Overview*.
November 2011	1.1	• Updated Table 2–1. • Restructured chapter.
May 2011	1.0	Initial release.

Variable Precision DSP Blocks in Arria V Devices

3

2020.04.13

AV-52003

Subscribe Send Feedback

This chapter describes how the variable-precision digital signal processing (DSP) blocks in Arria V devices are optimized to support higher bit precision in high-performance DSP applications.

Related Information

Arria V Device Handbook: Known Issues
Lists the planned updates to the *Arria V Device Handbook* chapters.

Features

The Arria V variable precision DSP blocks offer the following features:

- High-performance, power-optimized, and fully registered multiplication operations
- 9-bit, 18-bit, 27-bit, and 36-bit(2) word lengths
- 18 x 19 and 18 x 25 complex multiplications (2)
- Built-in addition, subtraction, and 64-bit accumulation unit to combine multiplication results
- Cascading 19-bit or 27-bit to form the tap-delay line for filtering applications
- Cascading 64-bit output bus to propagate output results from one block to the next block without external logic support
- Hard pre-adder supported in 18-bit, 19-bit, and 27-bit mode for symmetric filters
- Internal coefficient register bank for filter implementation
- 18-bit and 27-bit systolic finite impulse response (FIR) filters with distributed output adder

Related Information

Arria V Device Overview
Provides more information about the number of multipliers in each Arria V device.

(2) Only applicable for certain device variant. Refer to Supported Operational Modes in Arria V Devices for details.

Supported Operational Modes in Arria V Devices

Table 3-1: Variable Precision DSP Blocks Operational Modes for Arria V GX, GT, SX, and ST Devices

Variable-Precision DSP Block Resource	Operation Mode	Supported Instance	Pre-Adder Support	Coefficient Support	Input Cascade Support[3]	Chainout Support
1 variable precision DSP block	Independent 9 x 9 multiplication	3	No	No	No	No
	Independent 18 x 18 multiplication	2	Yes	Yes	Yes	No
	Independent 18 x 19 multiplication	2	Yes	Yes	Yes	No
	Independent 18 x 25 multiplication	1	Yes	Yes	Yes	Yes
	Independent 20 x 24 multiplication	1	Yes	Yes	Yes	Yes
	Independent 27 x 27 multiplication	1	Yes	Yes	Yes	Yes
	Two 18 x 19 multiplier adder mode	1	Yes	Yes	Yes	Yes
	18 x 18 multiplier adder summed with 36-bit input	1	Yes	No	No	Yes
2 variable precision DSP blocks	Complex 18 x 19 multiplication	1	No	No	Yes	No

(3) When you enable the pre-adder feature, the input cascade support is not available.

Table 3-2: Variable Precision DSP Blocks Operational Modes for Arria V GZ Devices

Variable Precision DSP Block Resources	Operational Mode	Supported Instance	Pre-adder Support	Coefficient Support	Input Cascade Support	Chainout Support
1 variable precision DSP block	Independent 9 x 9 multiplication	3	No	No	No	No
	Independent 16 x 16 multiplication	2	Yes	Yes	Yes	No
	Independent 18 x 18 partial multiplication (32-bit)	2	Yes	Yes	Yes	No
	Independent 18 x 18 multiplication	1	Yes	Yes	Yes	No
	Independent 27 x 27 multiplication	1	Yes	Yes	Yes	Yes
	Independent 36 x 18 multiplication	1	No	Yes	No	Yes
	Two 18 x 18 multiplier adder	1	Yes	Yes	Yes	Yes
	Two 16 x 16 multiplier adder	1	Yes	Yes	Yes	Yes
	Sum of 2 square	1	Yes(4)	No	No	Yes
	18 x 18 multiplication summed with 36-bit input	1	No	No	No	Yes

(4) The pre-adder feature for this mode is automatically enabled.

Variable Precision DSP Block Resources	Operational Mode	Supported Instance	Pre-adder Support	Coefficient Support	Input Cascade Support	Chainout Support
2 variable precision DSP blocks	Independent 18 x 18 multiplication	3	No	No	No	No
	Independent 36 x 36 multiplication	1	No	No	No	No
	Complex 18 x 18 multiplication	1	Yes	Yes	Yes	Yes
	Four 18 x 18 multiplier adder	1	Yes	Yes	Yes	No
	Two 27 x 27 multiplier adder	1	Yes	Yes	Yes	No
	Two 18 x 36 multiplier adder	1	No	Yes	No	No
3 variable precision DSP blocks	Complex 18 x 25 multiplication	1	Yes(4)	No	No	No
4 variable precision DSP blocks	Complex 27 x 27 multiplication	1	Yes	Yes	Yes	No

Resources

Table 3-3: Number of Multipliers in Arria V Devices

The table lists the variable-precision DSP resources by bit precision for each Arria V device.

Variant	Member Code	Variable-precision DSP Block	Independent Input and Output Multiplications Operator				18 x 18 Multiplier Adder Mode	18 x 18 Multiplier Adder Summed with 36 bit Input
			9 x 9 Multiplier	18 x 18 Multiplier	27 x 27 Multiplier	36 x 36 Multiplier		
Arria V GX	A1	240	720	480	240	—	240	240
	A3	396	1,188	792	396	—	396	396
	A5	600	1,800	1,200	600	—	600	600
	A7	800	2,400	1,600	800	—	800	800
	B1	920	2,760	1,840	920	—	920	920
	B3	1,045	3,135	2,090	1,045	—	1,045	1,045
	B5	1,092	3,276	2,184	1,092	—	1,092	1,092
	B7	1,156	3,468	2,312	1,156	—	1,156	1,156
Arria V GT	C3	396	1,188	792	396	—	396	396
	C7	800	2,400	1,600	800	—	800	800
	D3	1,045	3,135	2,090	1,045	—	1,045	1,045
	D7	1,156	3,468	2,312	1,156	—	1,156	1,156
Arria V GZ	E1	800	2,400	1,600	800	400	800	800
	E3	1,044	3,132	2,088	1,044	522	1,044	1,044
	E5	1,092	3,276	2,184	1,092	546	1,092	1,092
	E7	1,139	3,417	2,278	1,139	569	1,139	1,139
Arria V SX	B3	809	2,427	1,618	809	—	809	809
	B5	1,090	3,270	2,180	1,090	—	1,090	1,090
Arria V ST	D3	809	2,427	1,618	809	—	809	809
	D5	1,090	3,270	2,180	1,090	—	1,090	1,090

Design Considerations

You should consider the following elements in your design:

- Operational modes
- Internal coefficient and pre-adder
- Accumulator
- Chainout adder

Operational Modes

The Intel Quartus Prime software includes IP cores that you can use to control the operation mode of the multipliers. After entering the parameter settings with the IP Catalog, the Intel Quartus Prime software automatically configures the variable precision DSP block.

Altera provides two methods for implementing various modes of the Arria V variable precision DSP block in a design—using the Intel Quartus Prime DSP IP cores and HDL inferring.

The following Intel Quartus Prime IP cores are supported for the Arria V variable precision DSP blocks implementation:

- LPM_MULT
- ALTERA_MULT_ADD
- ALTMULT_COMPLEX
- ALTMEMMULT

Related Information

- **Introduction to Altera IP Cores**
- **Integer Arithmetic IP Cores User Guide**
- **Floating-Point IP Cores User Guide**
- **Quartus II Software Help**

Internal Coefficient and Pre-Adder

To use the pre-adder feature, all input data and multipliers must have the same clock setting.

The input cascade support is not available when you enable the pre-adder feature.

Table 3-4: Internal Coefficient and Pre-Adder Features in Arria V Devices

Mode	Arria V GX, GT, SX, and ST	Arria V GZ
18-bit	The coefficient feature and pre-adder feature can be used independently.	The coefficient feature must be enabled when the pre-adder feature is enabled.
27-bit	The coefficient feature and pre-adder feature can be used independently.	The coefficient feature and pre-adder feature can be used independently. With pre-adder enabled: • If the multiplicand input comes from dynamic input due to width limitation in the input registers—the input data width is restricted to 22 bits. • If the multiplicand input comes from the internal coefficients—the data width of the multiplicand is 27 bits.

Note: When you enable the pre-adder feature, all input data must have the same clock setting.

Accumulator

The accumulator in the Arria V GX, GT, SX, and ST devices supports double accumulation by enabling the 64-bit double accumulation registers located between the output register bank and the accumulator.

The double accumulation registers are set statically in the programming file.

The accumulator in the Arria V GZ devices does not support double accumulation. The accumulator feature is not available in multi-block modes.

Chainout Adder

You can use the output chaining path to add results from other DSP blocks.

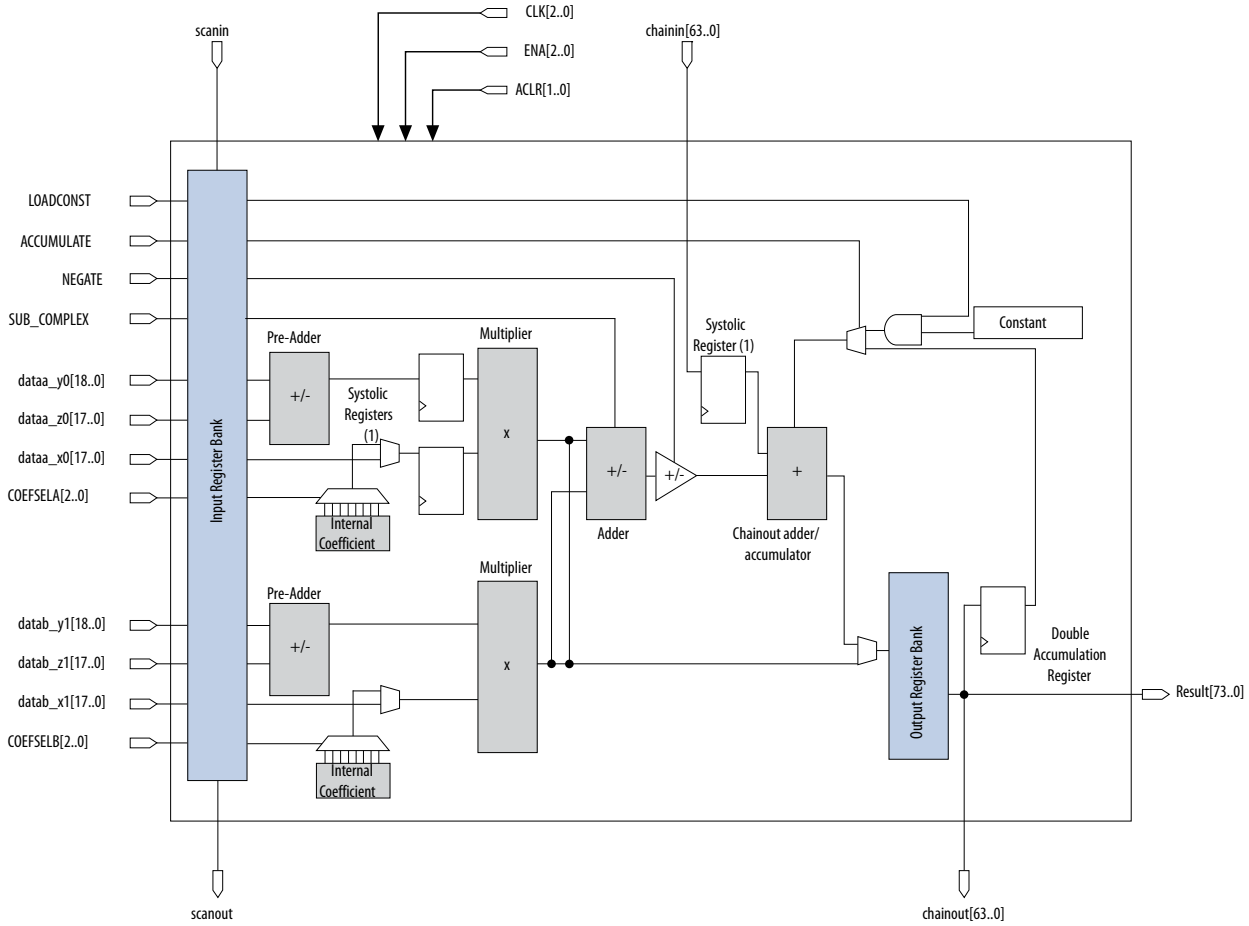
Block Architecture

The Arria V variable precision DSP block consists of the following elements:

- Input register bank
- Pre-adder
- Internal coefficient
- Multipliers
- Adder
- Accumulator and chainout adder
- Systolic registers
- Double accumulation register
- Output register bank

If the variable precision DSP block is not configured in systolic FIR mode, both systolic registers are bypassed.

Figure 3-1: Variable Precision DSP Block Architecture in 18 x 19 Mode for Arria V GX, GT, SX, and ST Devices



Note:
1. When enabled, systolic registers are clocked with the same clock source as the output register bank.

Figure 3-2: Variable Precision DSP Block Architecture in 27 x 27 Mode for Arria V GX, GT, SX, and ST Devices

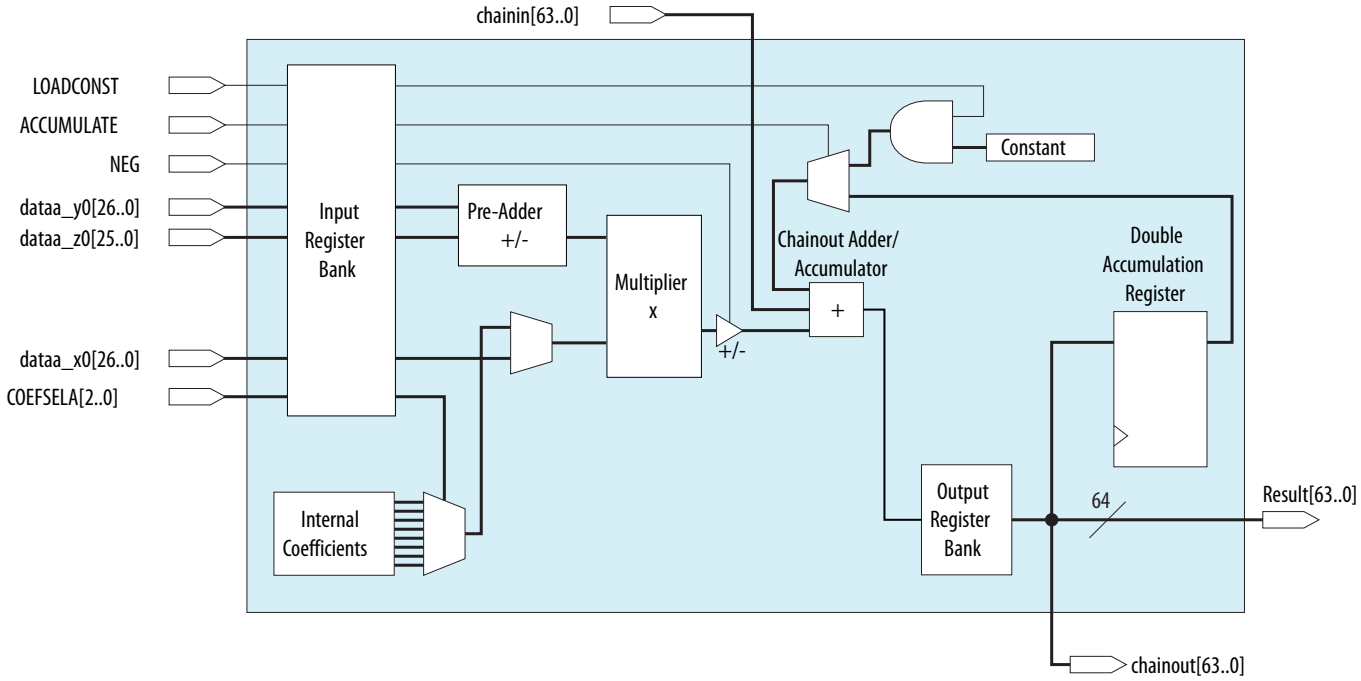


Figure 3-3: Variable Precision DSP Block Architecture in 18 x 18 Mode for Arria V GZ Devices

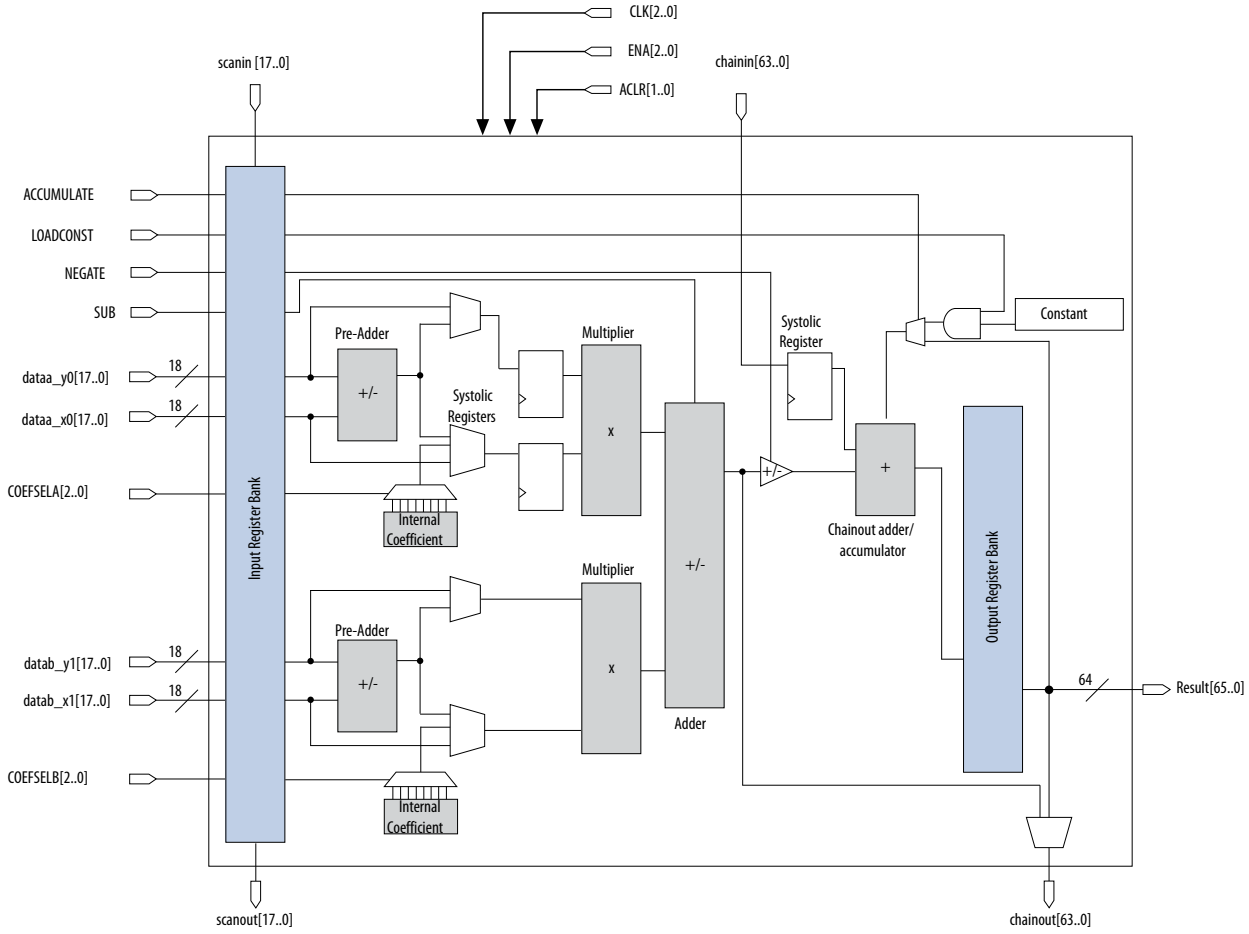
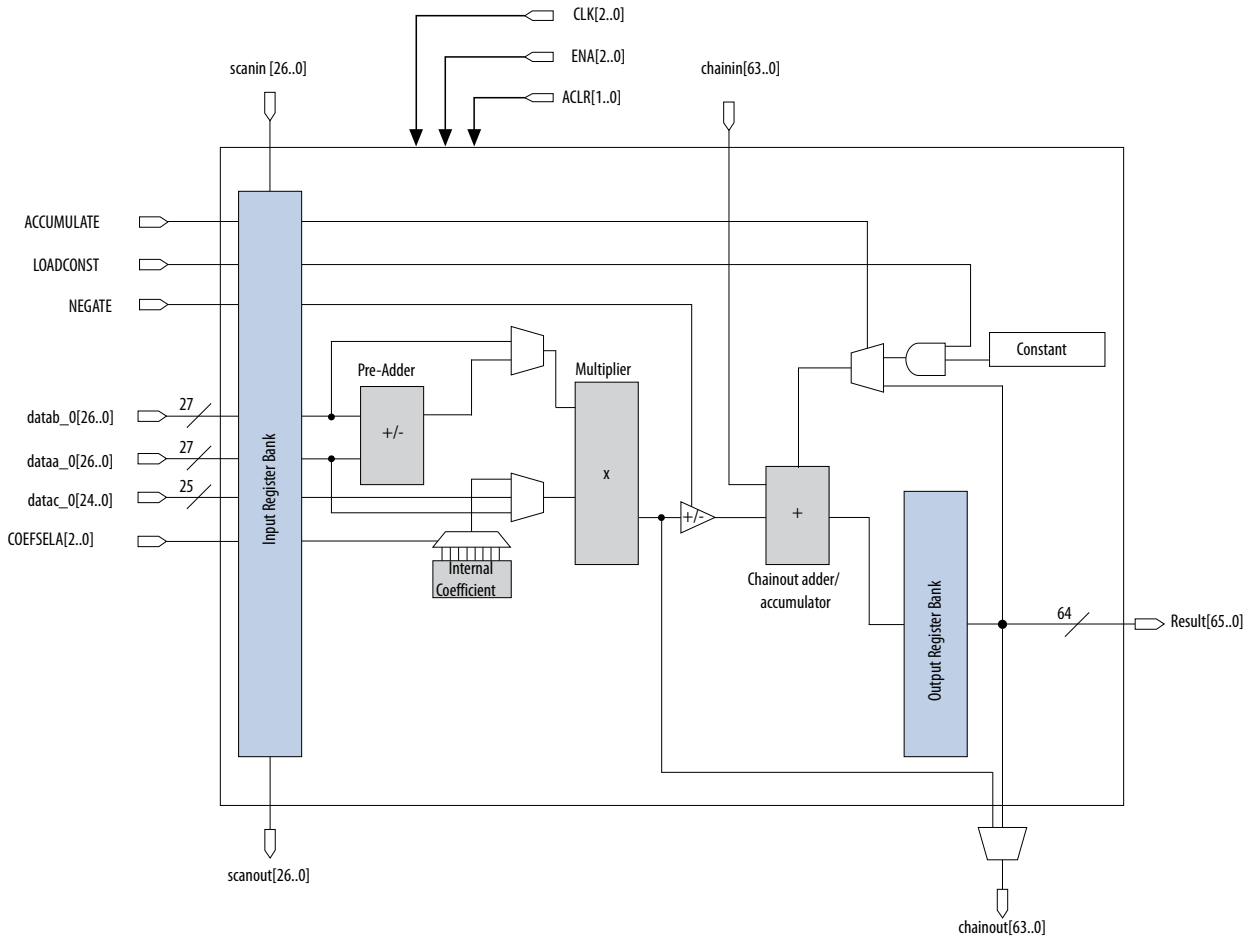


Figure 3-4: Variable Precision DSP Block Architecture in 27 x 27 Mode for Arria V GZ Devices



Input Register Bank

The input register bank consists of data, dynamic control signals, and two sets of delay registers.

All the registers in the DSP blocks are positive-edge triggered and cleared on power up. Each multiplier operand can feed an input register or a multiplier directly, bypassing the input registers.

The following variable precision DSP block signals control the input registers within the variable precision DSP block:

- `CLK[2..0]`
- `ENA[2..0]`
- `ACLR[0]`

In 18 x 18 and 18 x 19 mode, you can use the delay registers to balance the latency requirements when you use both the input cascade and chainout features.

The tap-delay line feature allows you to drive the top leg of the multiplier inputs from general routing or from the cascade chain. The following inputs can be driven from either the general routing or from the cascade chain:

- For Arria V GX, GT, SX, and ST devices:
  - `dataa_y0` and `datab_y1` in 18 x 19 mode
  - `dataa_y0` in 27 x 27 mode
- For Arria V GZ devices:
  - `dataa_y0[17..0]` and `datab_y1[17..0]` in 18 x 18 mode
  - `dataa_y0` in 27 x 27 mode

The Arria V GZ variable precision DSP block support 18-bit and 27-bit input cascading.

These figures show the input registers for Arria V devices.

Figure 3-5: Input Register of a Variable Precision DSP Block in 18 x 19 Mode for Arria V GX, GT, SX, and ST Devices

The figures show the data registers only. Registers for the control signals are not shown.

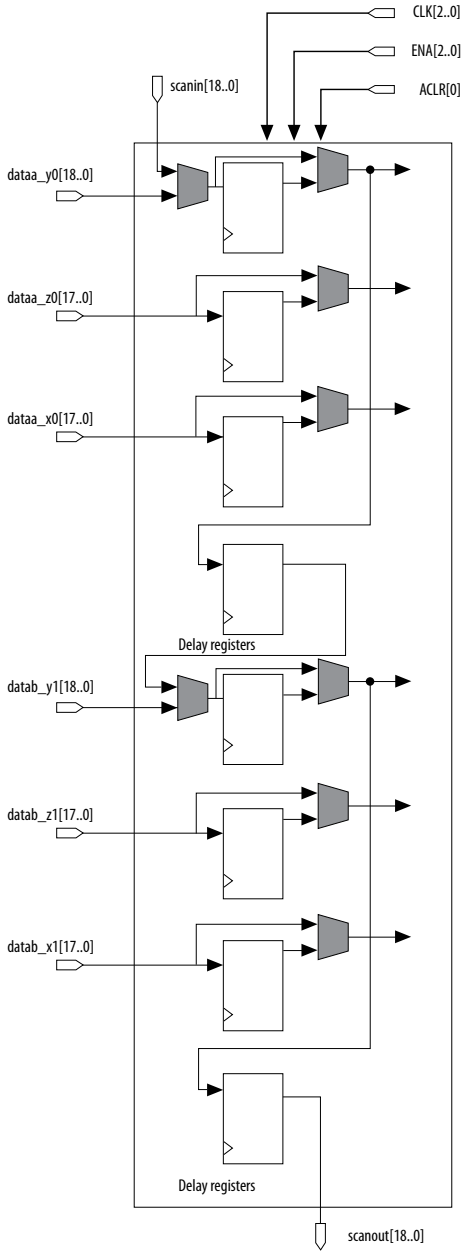


Figure 3-6: Input Register of a Variable Precision DSP Block in 27 x 27 Mode for Arria V GX, GT, SX, and ST Devices

The figures show the data registers only. Registers for the control signals are not shown.

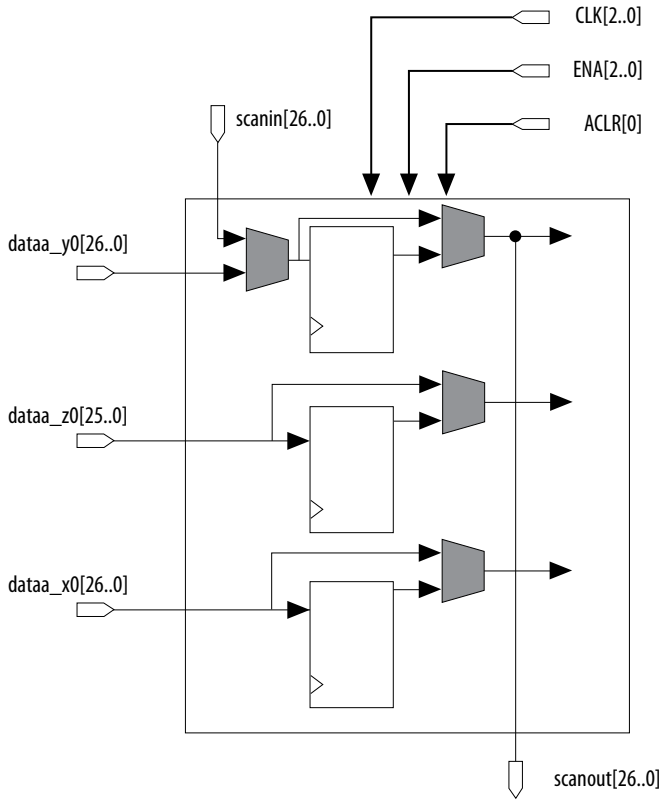


Figure 3-7: Input Register of a Variable Precision DSP Block in 18 x 18 Mode for Arria V GZ Devices

The figures show the data registers only. Registers for the control signals are not shown.

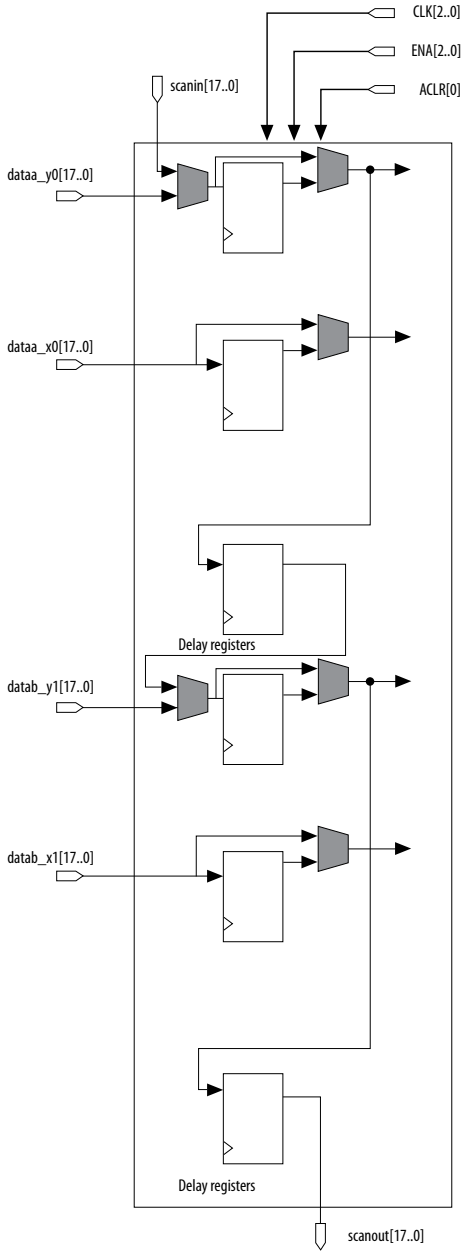
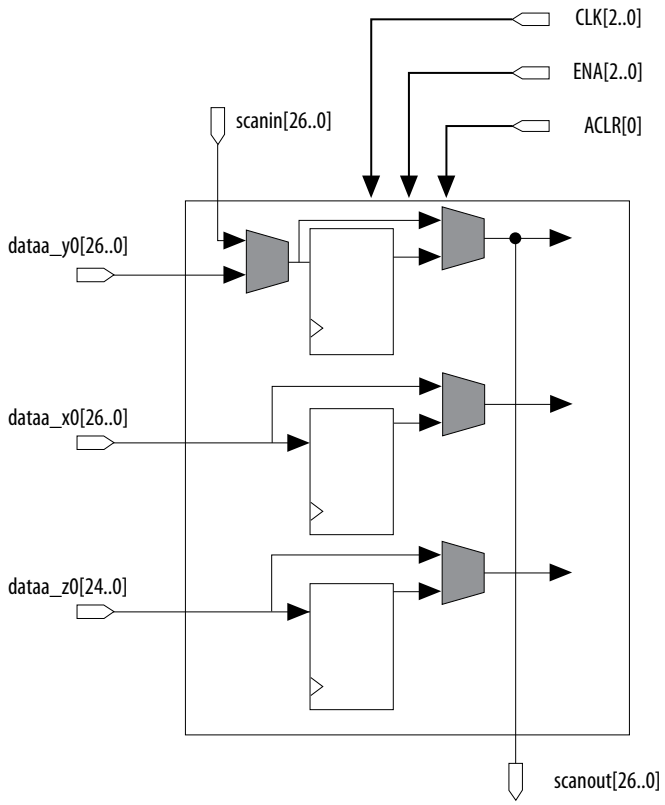


Figure 3-8: Input Register of a Variable Precision DSP Block in 27 x 27 Mode for Arria V GZ Devices

The figures show the data registers only. Registers for the control signals are not shown.



Pre-Adder

Arria V GX, GT, SX, and ST Devices

Each variable precision DSP block has two 19-bit pre-adders. You can configure these pre-adders in the following configurations:

- Two independent 19-bit pre-adders
- One 27-bit pre-adder

The pre-adder supports both addition and subtraction in the following input configurations:

- 18-bit (signed) addition or subtraction for 18 x 19 mode
- 17-bit (unsigned) addition or subtraction for 18 x 19 mode
- 26-bit addition or subtraction for 27 x 27 mode

Arria V GZ Devices

Each variable precision DSP block has two 18-bit pre-adders. You can configure these pre-adders in the following configurations:

- Two independent 18-bit adders
- One 26-bit adder

The pre-adder supports both addition and subtraction in the following input configurations:

- 17-bit addition or subtraction for 18-bit applications
- 25-bit addition or subtraction for 27-bit applications

Internal Coefficient

The Arria V variable precision DSP block has the flexibility of selecting the multiplicand from either the dynamic input or the internal coefficient.

The internal coefficient can support up to eight constant coefficients for the multiplicands in 18-bit and 27-bit modes. When you enable the internal coefficient feature, `COEFSELA/COEFSELB` are used to control the selection of the coefficient multiplexer.

Multipliers

A single variable precision DSP block can perform many multiplications in parallel, depending on the data width of the multiplier.

There are two multipliers per variable precision DSP block. You can configure these two multipliers in several operational modes.

For Arria V GX, GT, SX, and ST devices:

- One 27 x 27 multiplier
- Two 18 (signed)/(unsigned) x 19 (signed) multipliers
- Three 9 x 9 multipliers

For Arria V GZ devices:

- One 27 x 27 multiplier
- Two individual 16 x 16 multipliers
- Two individual 18 x 18 partial multipliers, with only 32-bit LSB multiplication result for each multiplication
- One individual 18 x 18 multiplier, with full 36-bit multiplication result
- One individual 27 x 27 multiplier
- One individual 36 x 18 multiplier
- There individual 9 x 9 multipliers

For Arria V GZ devices, you can use two adjacent DSP blocks to construct an individual 36-bit multiplier.

Related Information

Operational Mode Descriptions on page 3-19
Provides more information about the operational modes of the multipliers.

Adder

You can use the adder in various sizes, depending on the operational mode:

- One 64-bit adder with the 64-bit accumulator
- Two 18 x 19 modes—the adder is divided into two 37-bit adders to produce the full 37-bit result of each independent 18 x 19 multiplication
- Three 9 x 9 modes—you can use the adder as three 18-bit adders to produce three 9 x 9 multiplication results independently

Accumulator and Chainout Adder

The Arria V variable precision DSP block supports a 64-bit accumulator and a 64-bit adder.

For Arria V GX, GT, SX, and ST devices, the accumulator and chainout adder features are not supported in two independent 18 x 19 modes and three independent 9 x 9 modes.

For Arria V GZ devices, you can use the 64-bit adder as full adder.

The following signals can dynamically control the function of the accumulator:

- NEGATE
- LOADCONST
- ACCUMULATE

Table 3-5: Accumulator Functions and Dynamic Control Signals

This table lists the dynamic signals settings and description for each function. In this table, X denotes a "don't care" value.

Function	Description	NEGATE	LOADCONST	ACCUMULATE
Zeroing	Disables the accumulator.	0	0	0
Preload	Loads an initial value to the accumulator. Only one bit of the 64-bit preload value can be “1”. It can be used as rounding the DSP result to any position of the 64-bit result.	0	1	0
Accumulation	Adds the current result to the previous accumulate result.	0	X	1
Decimation	This function takes the current result, converts it into two's complement, and adds it to the previous result.	1	X	1

Systolic Registers

There are two systolic registers per variable precision DSP block. If the variable precision DSP block is not configured in systolic FIR mode, both systolic registers are bypassed.

The first set of systolic registers consists of the following registers:

- 18-bit and 19-bit registers that are used to register the 18-bit and 19-bit inputs of the upper multiplier respectively for Arria V GX, GT, SX, and ST devices
- 18-bit registers that are used to register the 18-bit inputs of the upper multiplier for Arria V GZ devices

The second set of systolic registers are used to delay the chainout output to the next variable precision DSP block.

You must clock all the systolic registers with the same clock source as the output register bank.

Double Accumulation Register

The double accumulation register is an extra register in the feedback path of the accumulator. Enabling the double accumulation register will cause an extra clock cycle delay in the feedback path of the accumulator.

This register has the same `CLK`, `ENA`, and `ACLR` settings as the output register bank.

By enabling this register, you can have two accumulator channels using the same number of variable precision DSP block.

Double accumulation register is not available in Arria V GZ devices.

Output Register Bank

The positive edge of the clock signal triggers the 64-bit bypassable output register bank and is cleared after power up.

The following variable precision DSP block signals control the output register per variable precision DSP block:

- `CLK[2..0]`
- `ENA[2..0]`
- `ACLR[1]`

Operational Mode Descriptions

This section describes how you can configure an Arria V variable precision DSP block to efficiently support the following operational modes:

- Independent Multiplier Mode
- Independent Complex Multiplier Mode
- Multiplier Adder Sum Mode
- Sum of Square Mode (Arria V GZ only)
- 18 x 18 Multiplication Summed with 36-Bit Input Mode
- Systolic FIR Mode

Independent Multiplier Mode

In independent input and output multiplier mode, the variable precision DSP blocks perform individual multiplication operations for general purpose multipliers.

Table 3-6: Variable Precision DSP Block Independent Multiplier Mode Configurations for Arria V Devices

Configuration	Multipliers per block	Device Variant Support
9 x 9	3	All
16 x 16	1	Arria V GZ

Configuration	Multipliers per block	Device Variant Support
18 x 18 (partial)	1	Arria V GZ
18 x 18	1	Arria V GZ
18 (signed) x 18 (unsigned)	2	Arria V GX, GT, SX, ST
18 (unsigned) x 18 (unsigned)	2	Arria V GX, GT, SX, ST
18 (signed) x 19 (signed)	2	Arria V GX, GT, SX, ST
18 (unsigned) x 19 (signed)	2	Arria V GX, GT, SX, ST
18 x 25	1	Arria V GX, GT, SX, ST
20 x 24	1	Arria V GX, GT, SX, ST
27 x 27	1	All
36 x 18	1	Arria V GZ

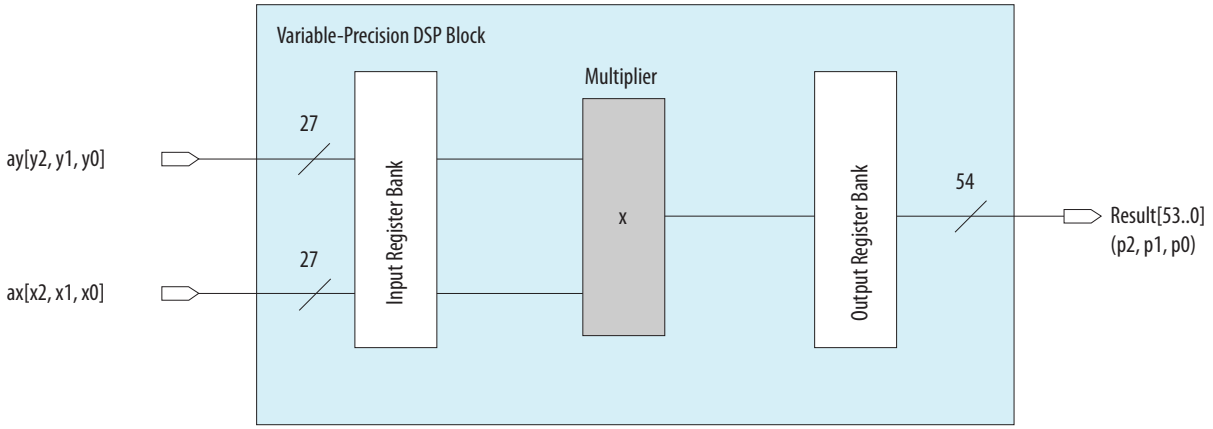
Table 3-7: Independent Multiplier Mode Configurations with Multiple Variable Precision DSP Blocks for Arria V Devices

Configuration	Number of DSP Blocks Required	Device Variant Support
3 independent 18 x 18 multipliers	2	Arria V GZ
36 x 36 multiplier	2	Arria V GZ

9 x 9 Independent Multiplier

Figure 3-9: Three 9 x 9 Independent Multiplier Mode per Variable Precision DSP Block for Arria V Devices

Three pairs of data are packed into the ax and ay ports; `result` contains three 18-bit products.



18 x 18 Independent Multiplier

Figure 3-10: One 18 x 18 Independent Multiplier Mode with One Variable Precision DSP Block for Arria V GZ Devices

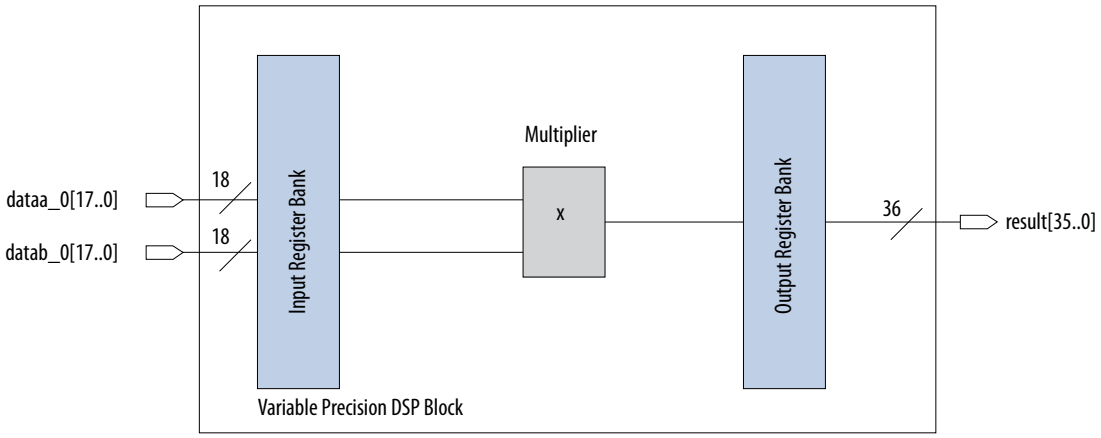
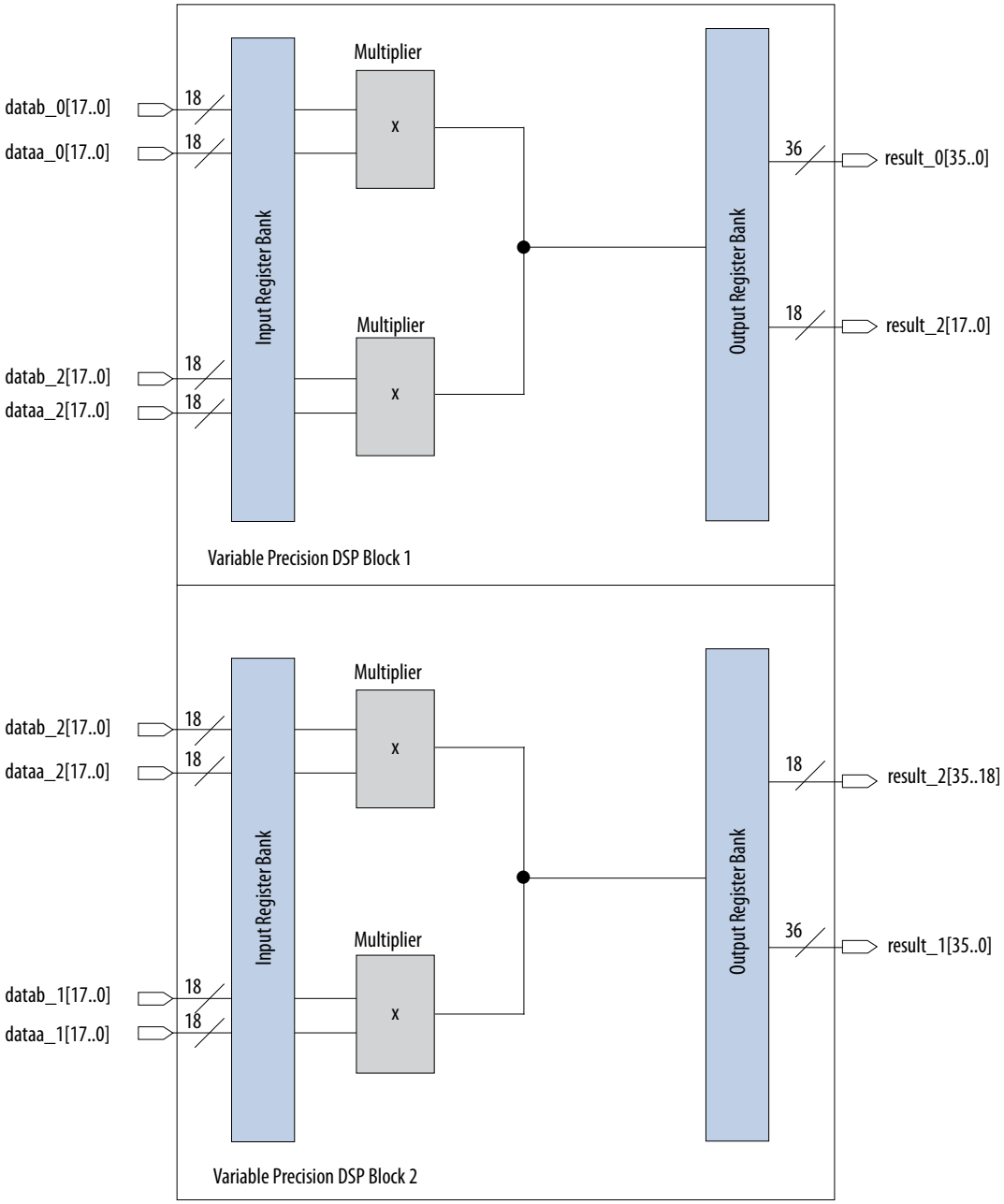


Figure 3-11: Three 18 x 18 Independent Multiplier Mode with Two Variable Precision DSP Blocks for Arria V GZ Devices

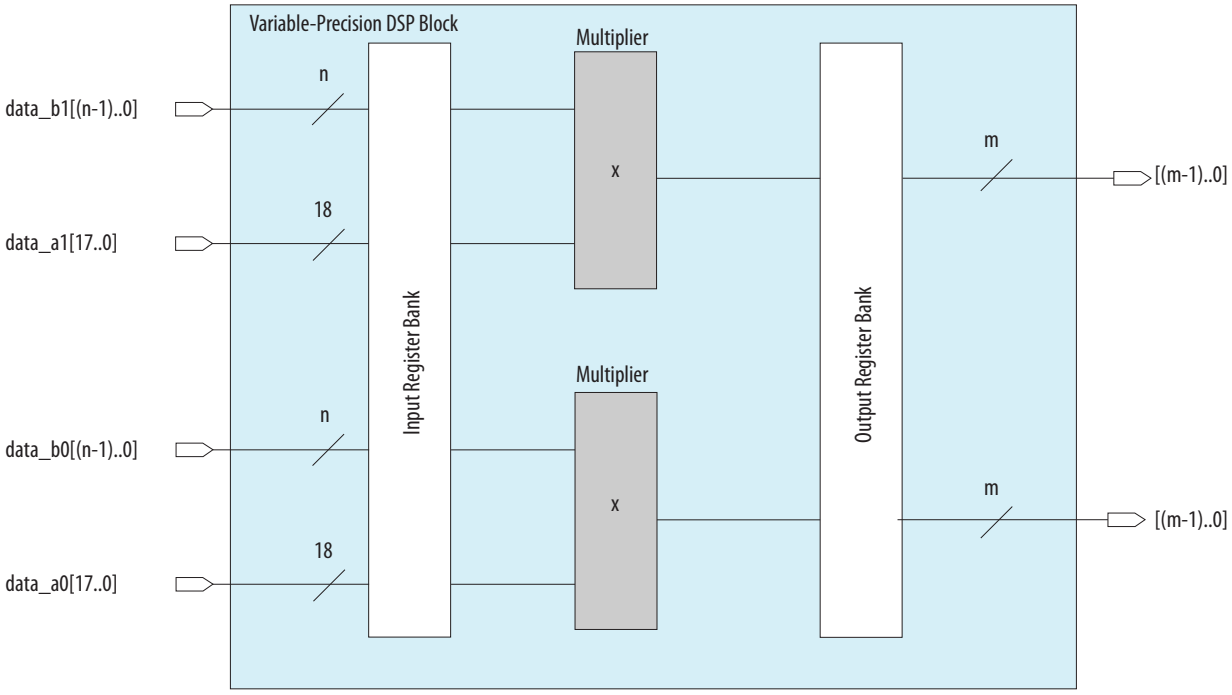


18 x 18 or 18 x 19 Independent Multiplier

Figure 3-12: Two 18 x 18 or 18 x 19 Independent Multiplier Mode per Variable Precision DSP Block for Arria V GX, GT, SX, and ST Devices

In this figure, the variables are defined as follows:

- n = 19 and m = 37 for 18 x 19 mode
- n = 18 and m = 36 for 18 x 18 mode

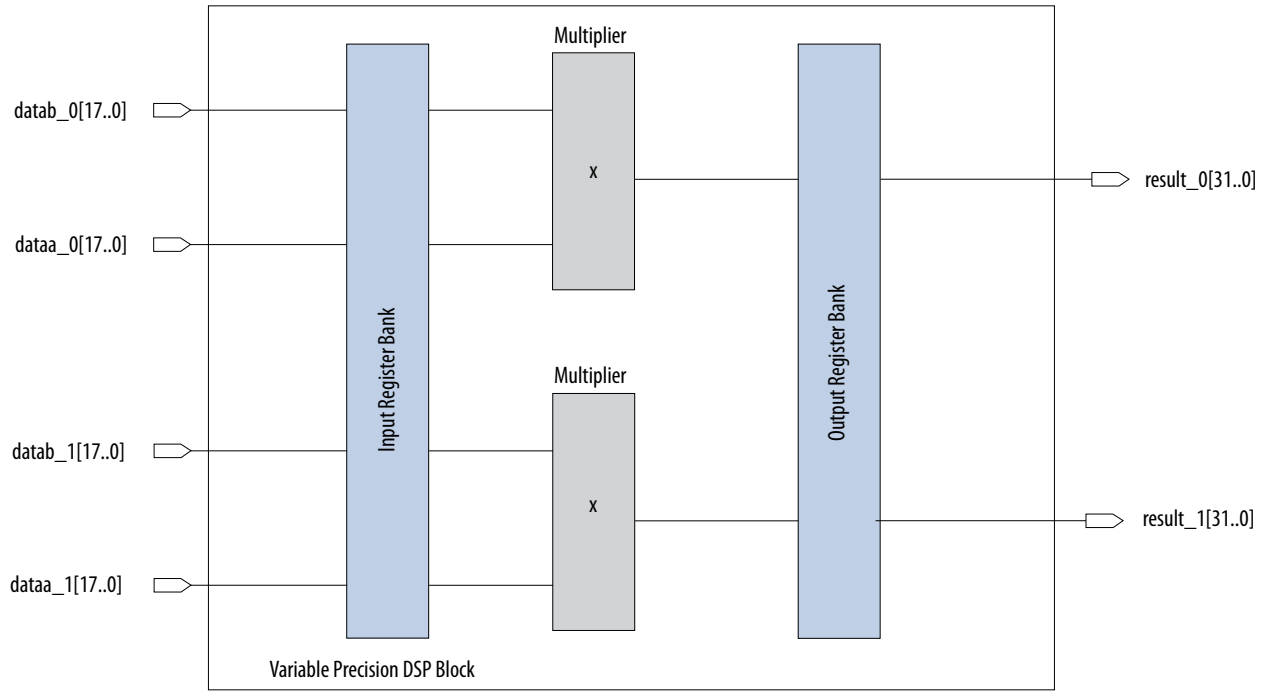


16 x 16 Independent Multiplier or 18 x 18 Independent Partial Multiplier

Figure 3-13: Two 16 x 16 Independent Multiplier Mode or Two 18 x 18 Independent Partial Multiplier Mode for Arria V GZ Devices

In this figure, the inputs for 16-bit independent multiplier mode are `data[15..0]`. The unused input bits require padding with zero.

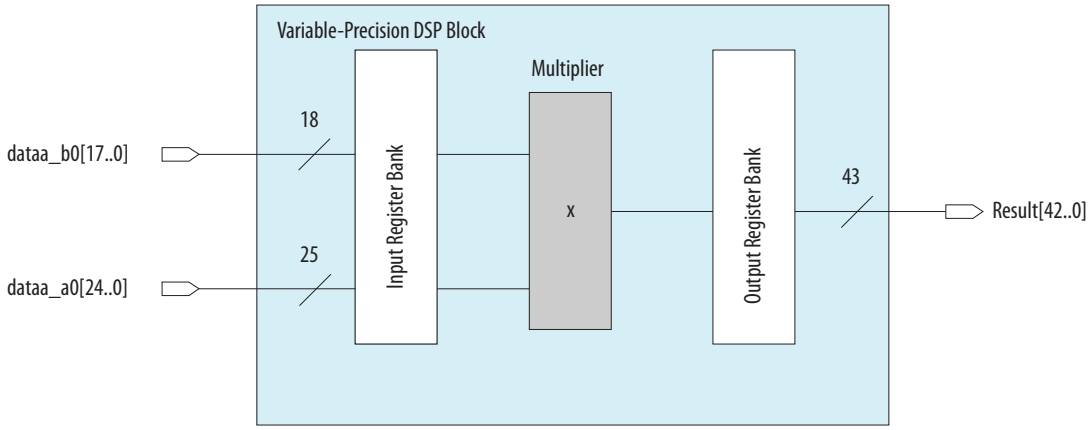
For two independent 18 x 18 partial multiplier mode, only 32-bit LSB result for each multiplication operation is routed to the output. The output has full precision if the total width of the multiplicand input is less than or equal to 32 bits for each multiplier.



18 x 25 Independent Multiplier

Figure 3-14: One 18 x 25 Independent Multiplier Mode per Variable Precision DSP Block for Arria V GX, GT, SX, and ST Devices

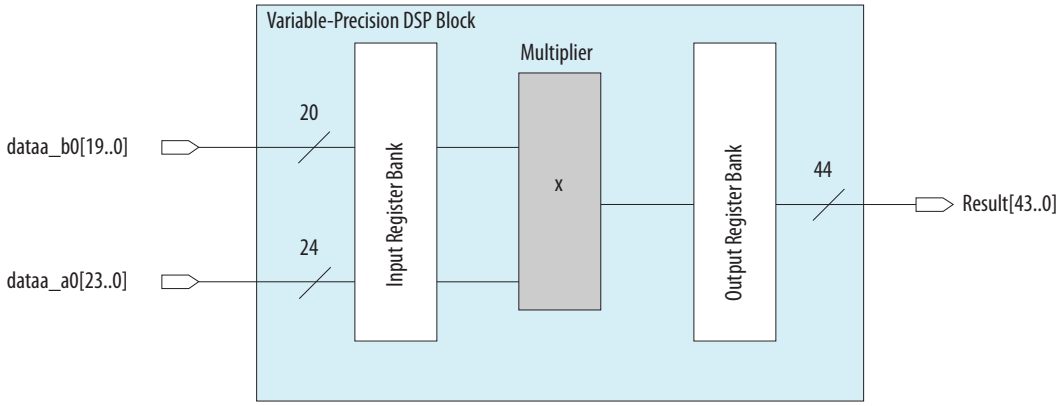
In this mode, the `result` can be up to 52 bits when combined with a chainout adder or accumulator.



20 x 24 Independent Multiplier

Figure 3-15: One 20 x 24 Independent Multiplier Mode per Variable Precision DSP Block for Arria V GX, GT, SX, and ST Devices

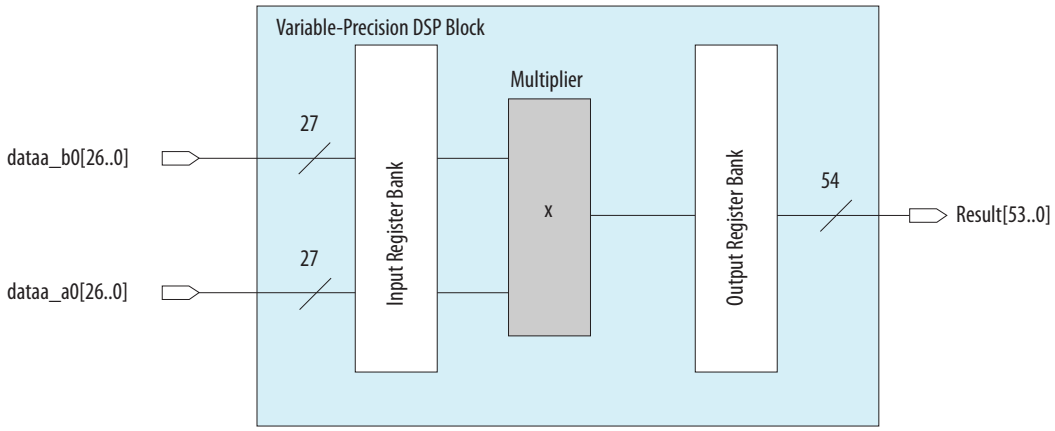
In this mode, the `result` can be up to 52 bits when combined with a chainout adder or accumulator.



27 x 27 Independent Multiplier

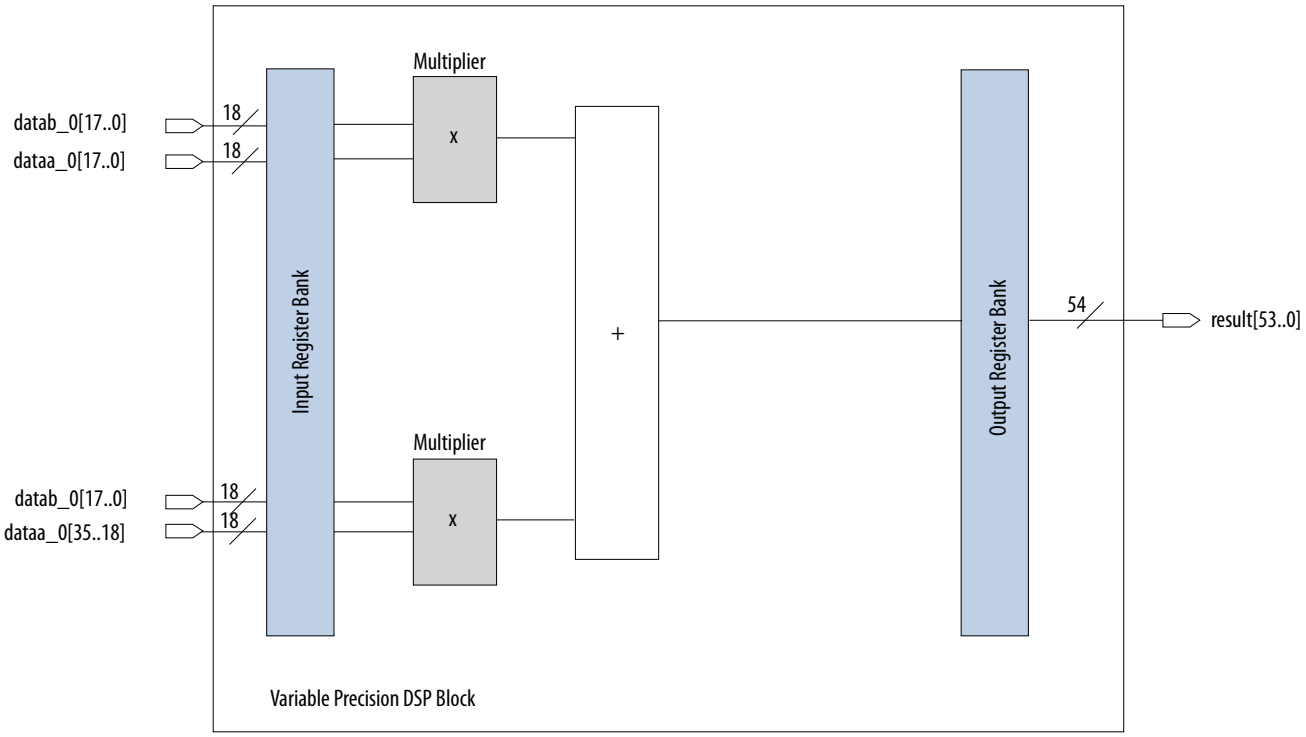
Figure 3-16: One 27 x 27 Independent Multiplier Mode per Variable Precision DSP Block for Arria V Devices

In this mode, the `result` can be up to 64 bits when combined with a chainout adder or accumulator.



36 x 18 Independent Multiplier

Figure 3-17: One 36 x 18 Independent Multiplier Mode for Arria V GZ Devices

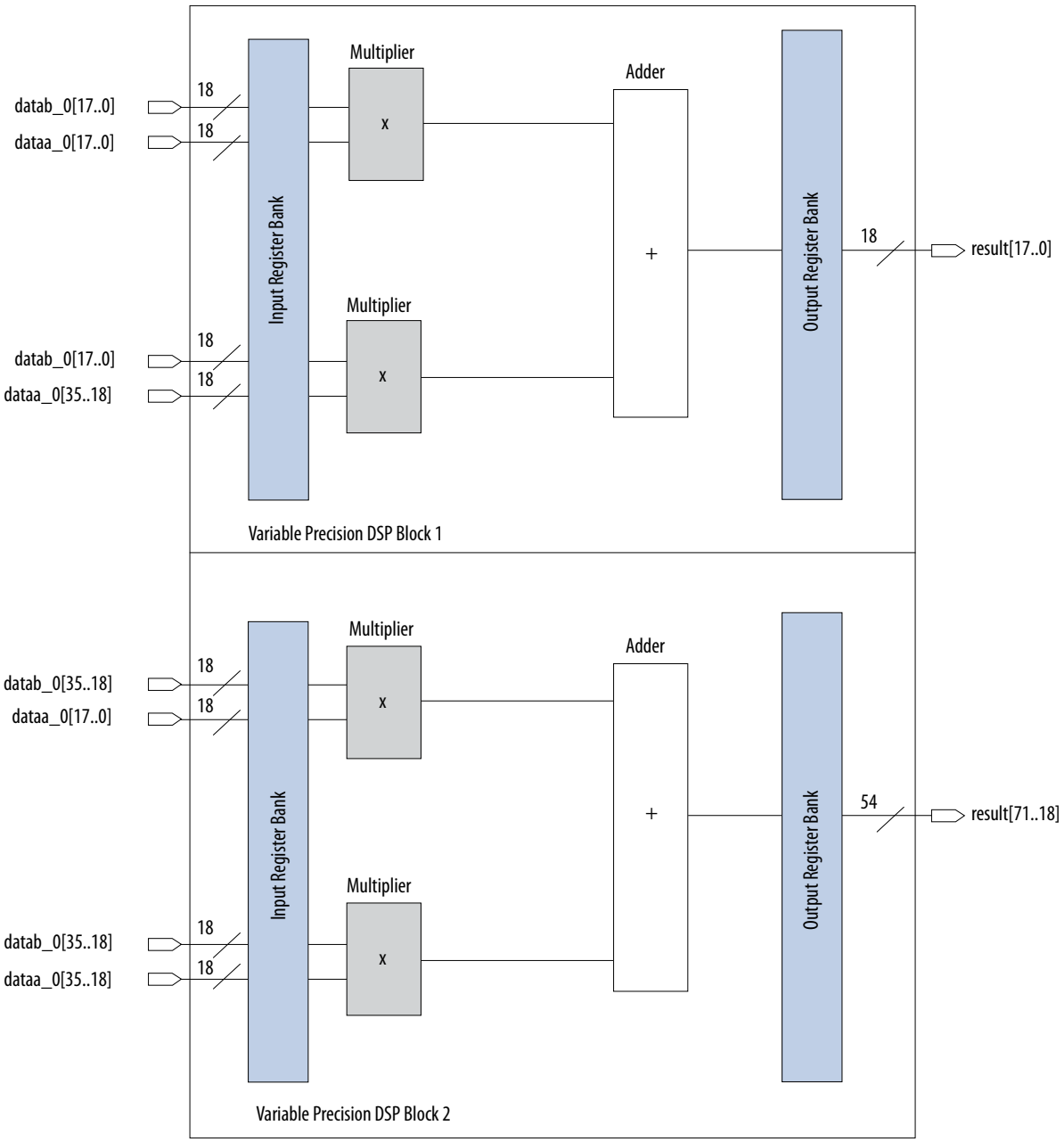


36-Bit Independent Multiplier

You can efficiently construct an individual 36-bit multiplier with two adjacent variable precision DSP blocks. The 36 x 36 multiplication consists of four 18 x 18 multipliers.

The 36-bit multiplier is useful for applications requiring more than 18-bit precision; for example, for the mantissa multiplication portion of very high precision fixed-point arithmetic applications.

Figure 3-18: 36-Bit Independent Multiplier Mode with Two Variable Precision DSP Blocks for Arria V GZ Devices



Independent Complex Multiplier Mode

The Arria V variable precision DSP block provides the means for a complex multiplication.

Figure 3-19: Sample of Complex Multiplication Equation

$$(a + jb) \times (c + jd) = [(a \times c) - (b \times d)] + j[(a \times d) + (b \times c)]$$

Table 3-8: Variable Precision DSP Block Independent Complex Multiplier Mode Configurations for Arria V Devices

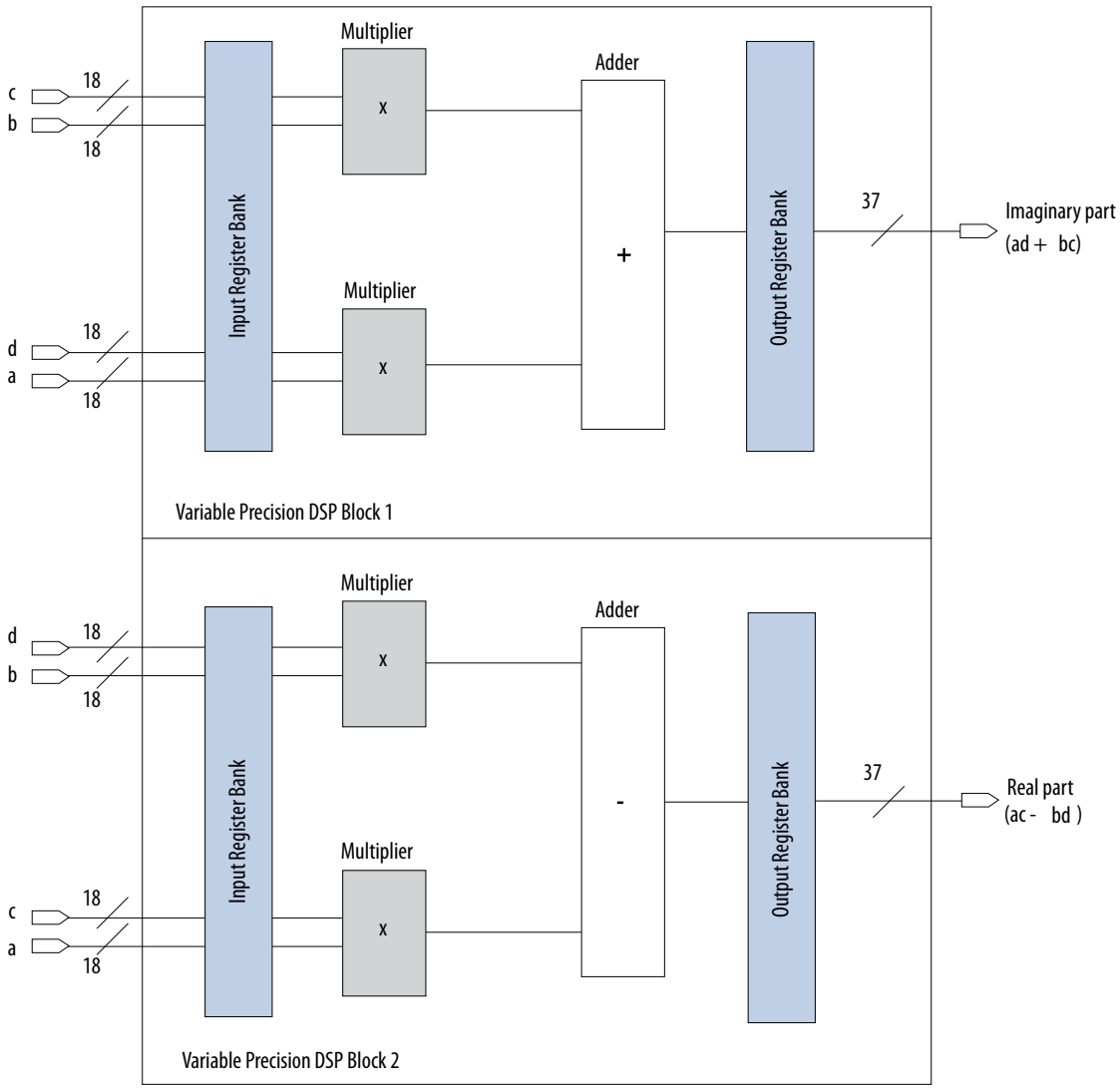
Configuration	Number of DSP Blocks Required	Device Variant Support
18 x 18	2	Arria V GZ
18 x 19	2	Arria V GX, GT, SX, ST
18 x 25	3	Arria V GZ
27 x 27	4	Arria V GZ

18 x 18 Complex Multiplier

For 18 x 18 complex multiplication mode, you require two variable precision DSP blocks to perform this multiplication.

You can implement the imaginary part [(a × d) + (b × c)] in the first variable precision DSP block, and you can implement the real part [(a × c) – (b × d)] in the second variable precision DSP block.

Figure 3-20: 18 x 18 Complex Multiplier with Two Variable Precision DSP Blocks for Arria V GZ Devices

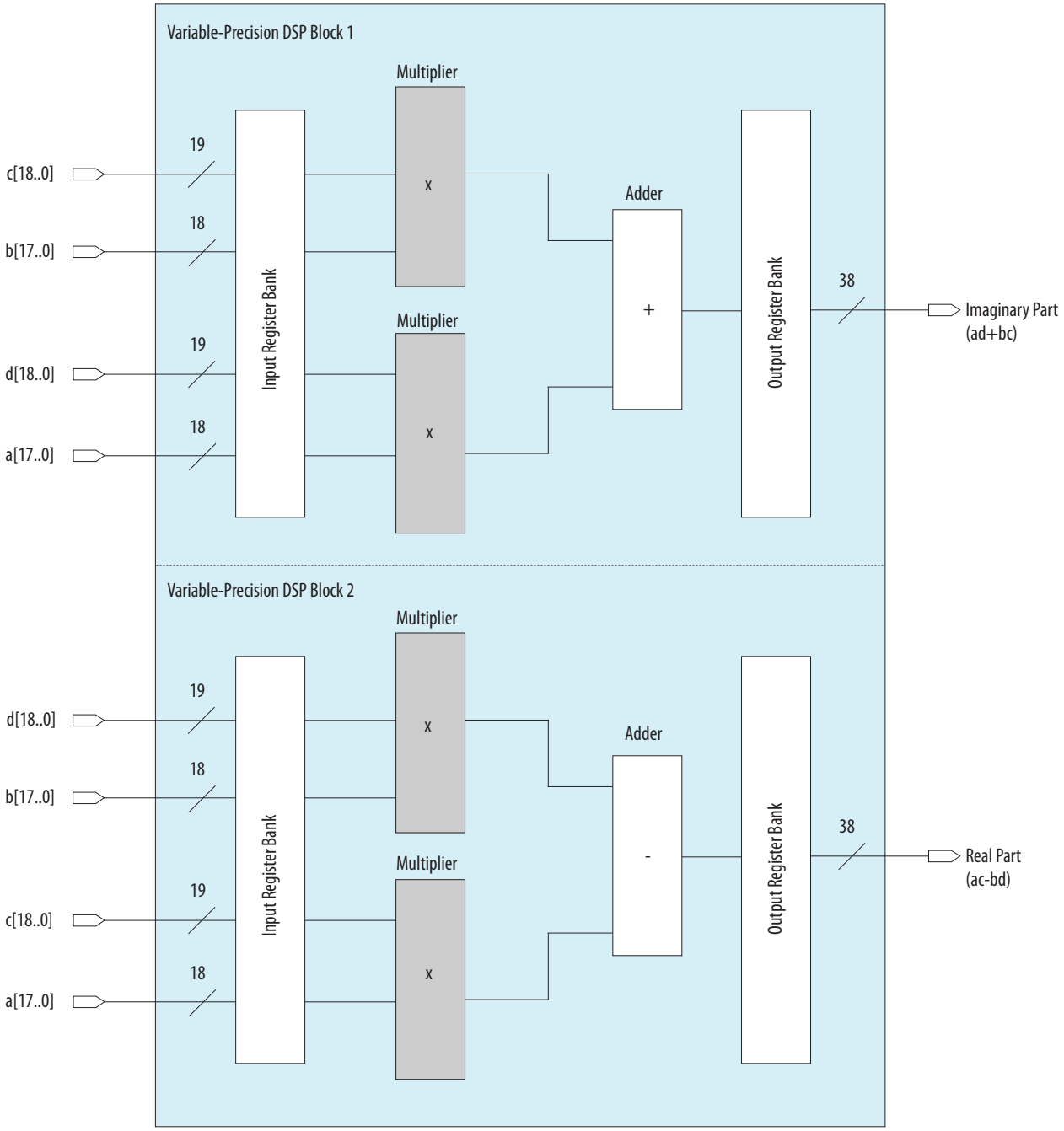


18 x 19 Complex Multiplier

For 18 x 19 complex multiplication mode, you require two variable precision DSP blocks to perform this multiplication.

The imaginary part [(a × d) + (b × c)] is implemented in the first variable precision DSP block, while the real part [(a × c) - (b × d)] is implemented in the second variable precision DSP block.

Figure 3-21: One 18 x 19 Complex Multiplier with Two Variable Precision DSP Blocks for Arria V GX, GT, SX, and ST Devices



18 x 25 Complex Multiplier

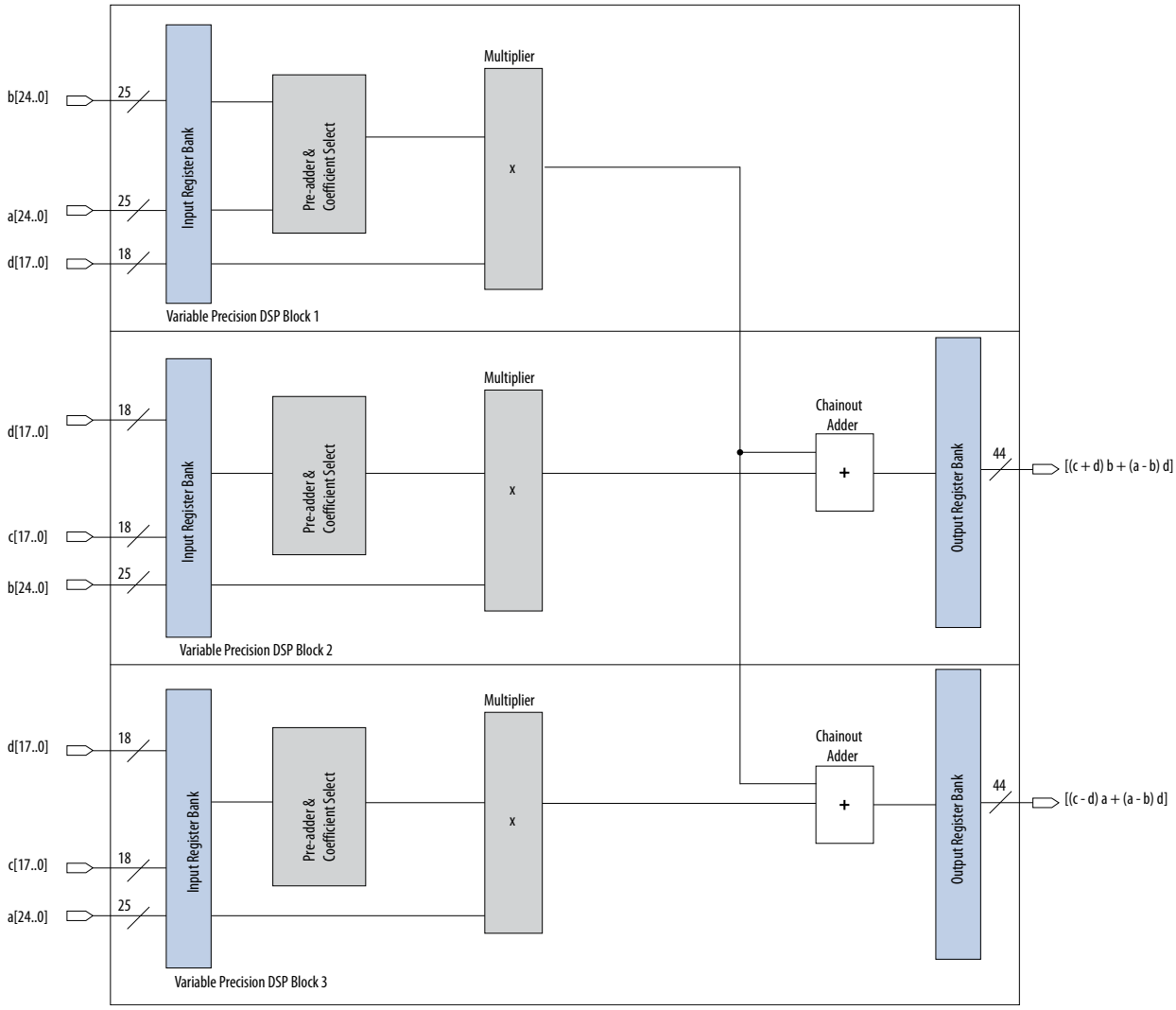
Arria V GZ devices support an individual 18 x 25 complex multiplication mode.

A 27 x 27 multiplier allows you to implement an individual 18 x 25 complex multiplication mode with three variable precision DSP blocks only. The pre-adder feature is automatically enabled for you to implement an individual 18 x 25 complex multiplication mode efficiently.

Figure 3-22: 18 x 25 Complex Multiplication Equation

$$(a + jb) \times (c + jd) = (c - d) \times a + (a - b) \times d + j[(c + d) \times b + (a - b) \times d]$$

Figure 3-23: 18 x 25 Complex Multiplier with Three Variable Precision DSP Blocks for Arria V GZ Devices



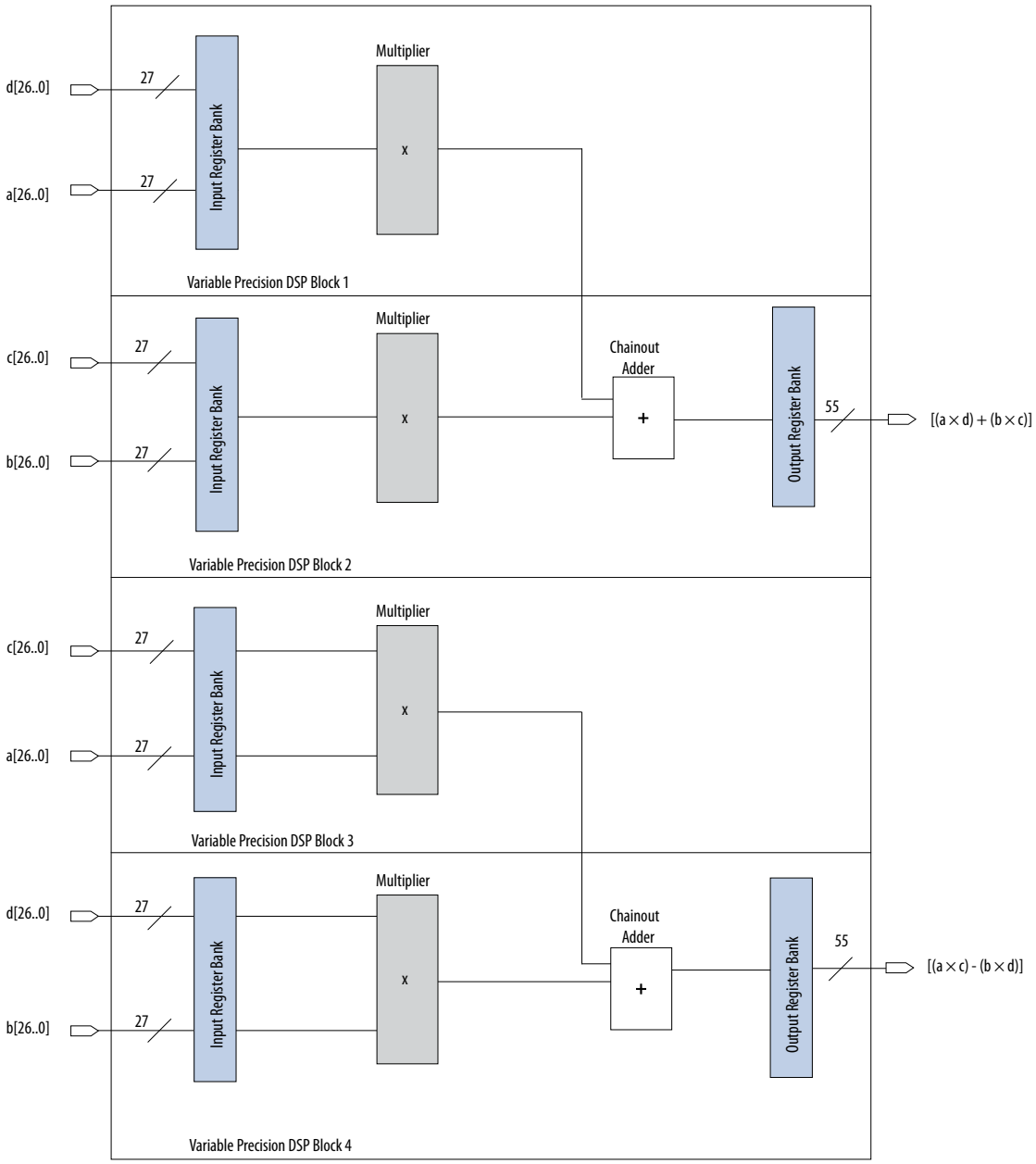
27 x 27 Complex Multiplier

Arria V GZ devices support an individual 27 x 27 complex multiplication mode. You require four variable precision DSP blocks to implement an individual 27 x 27 complex multiplication mode.

You can implement the imaginary part [(a x d) + (b x c)] in the first and second variable precision DSP blocks, and you can implement the real part [(a x c) - (b x d)] in the third and fourth variable precision DSP blocks.

You can achieve the difference of two 27 x 27 multiplications by enabling the `NEGATE` control signal in the fourth variable precision DSP block.

Figure 3-24: 27 x 27 Complex Multiplier with Four Variable Precision Blocks for Arria V GZ Devices



Multiplier Adder Sum Mode

Table 3-9: Variable Precision DSP Block Multiplier Adder Sum Mode Configurations for Arria V Devices

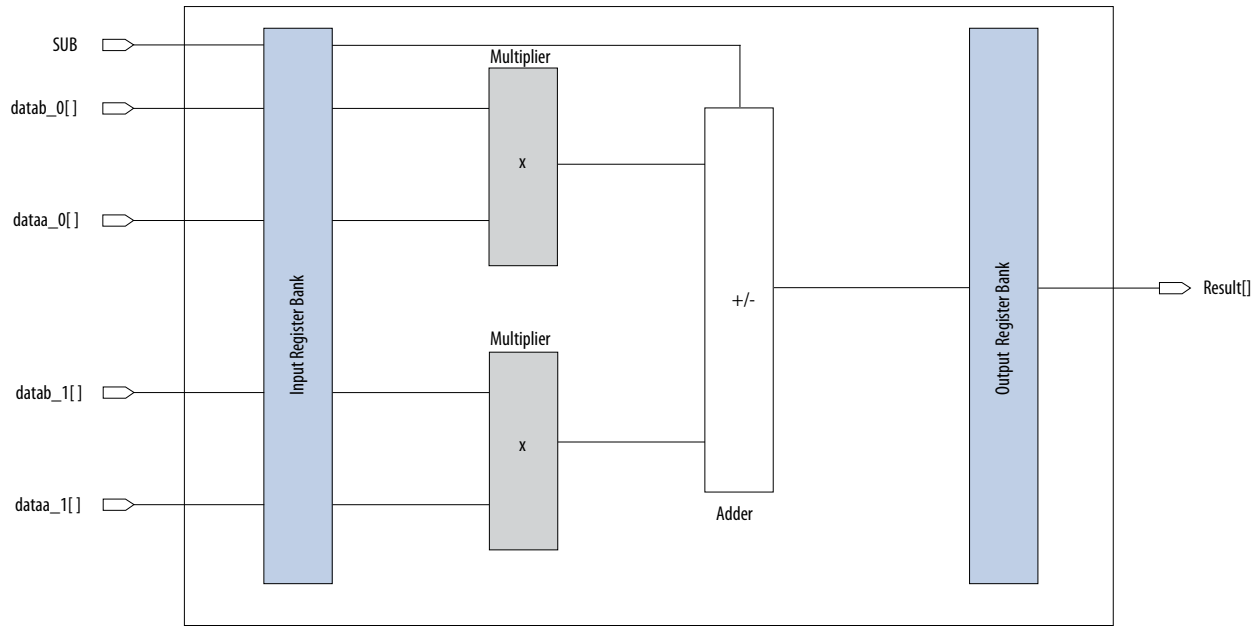
Mode	Configuration	Number of DSP Blocks Required	Device Variant Support
Two-multiplier Adder Sum	16 x 16	1	Arria V GZ
	18 x 18	1	Arria V GZ
	18 x 19	1	Arria V GX, GT, SX, ST
	27 x 27	2	Arria V GZ
	36 x 18	2	Arria V GZ
Four-multiplier Adder Sum	18 x 18	2	Arria V GZ

One Sum of Two 18 x 18 Multipliers or Two 16 x 16 Multipliers

Figure 3-25: One Sum of Two 18 x 18 Multipliers or Two 16 x 16 Multipliers with One Variable Precision DSP Block for Arria V GZ Devices

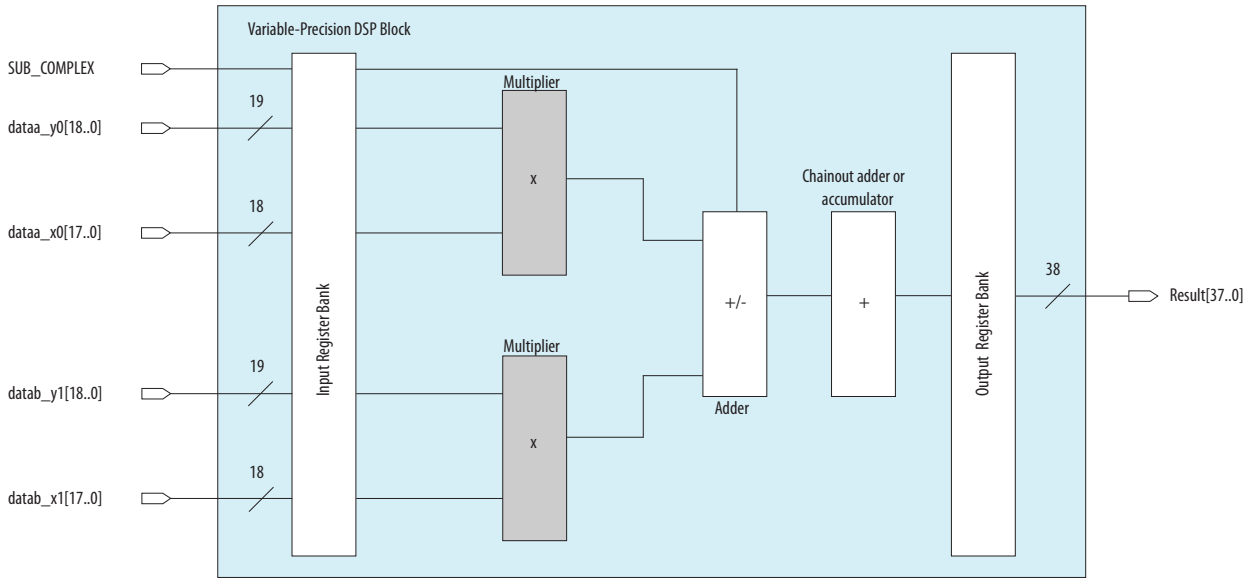
In this figure, for 18-bit multiplier adder sum mode, the input data width is 18 bits and the output data width is 37 bits.

For 16-bit multiplier adder sum mode, the input data width is 16 bits and the unused input bit requires padding with zeroes. The output data width is 33 bits.



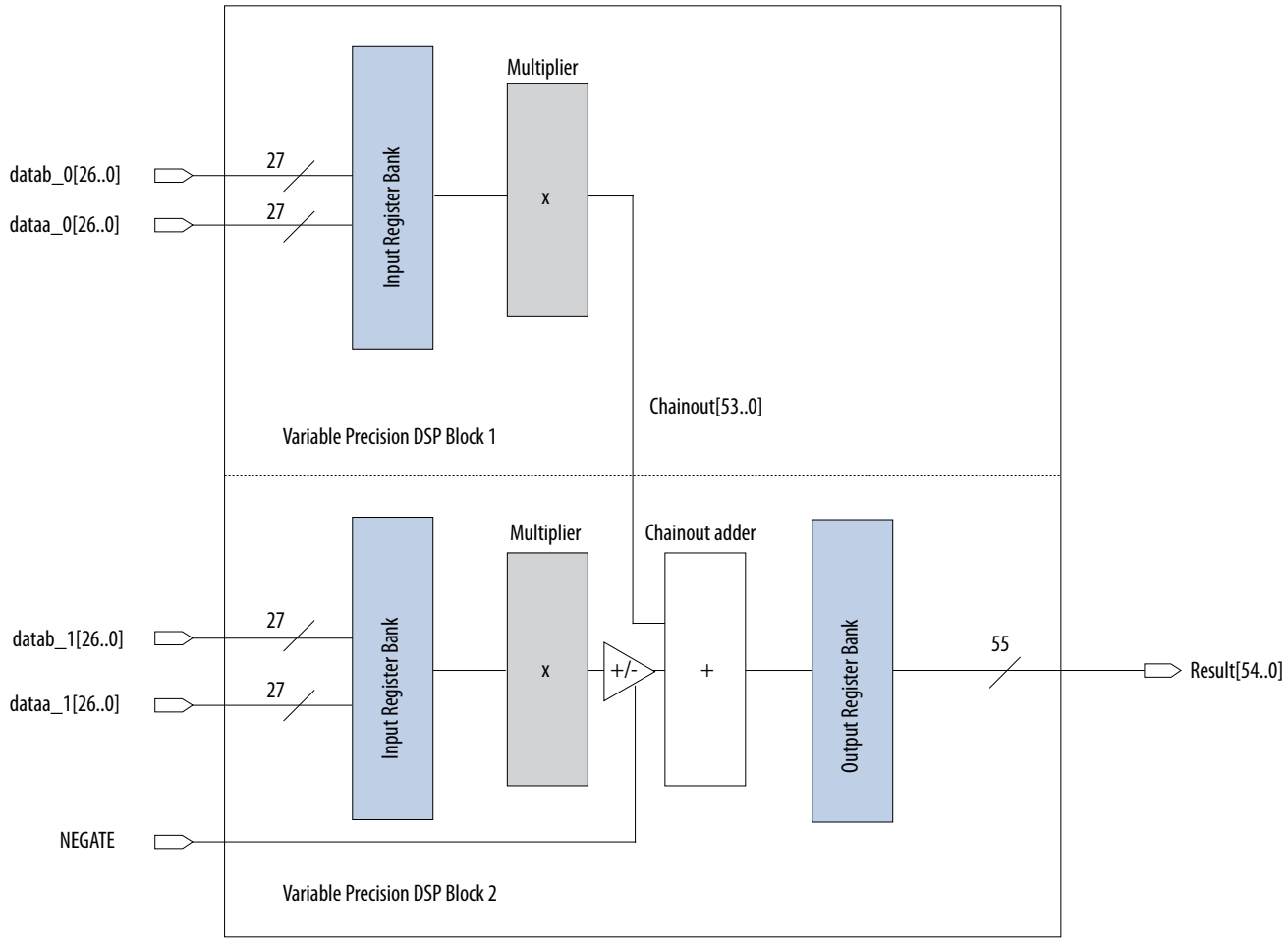
One Sum of Two 18 x 19 Multipliers

Figure 3-26: One Sum of Two 18 x 19 Multipliers with One Variable Precision DSP Block for Arria V GX, GT, SX, and ST Devices



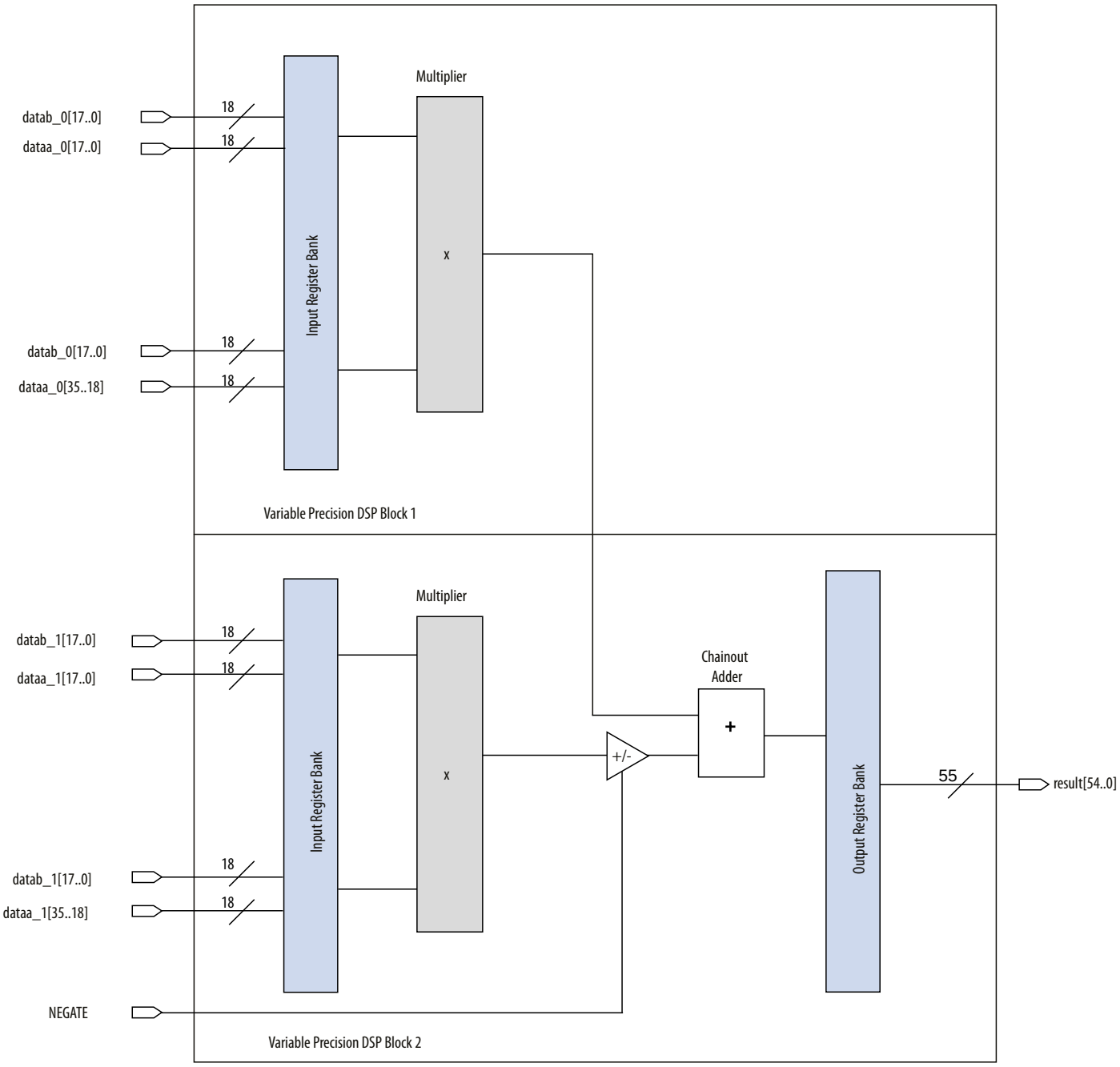
One Sum of Two 27 x 27 Multipliers

Figure 3-27: One Sum of Two 27 x 27 Multipliers with Two Variable Precision DSP Blocks for Arria V GZ Devices



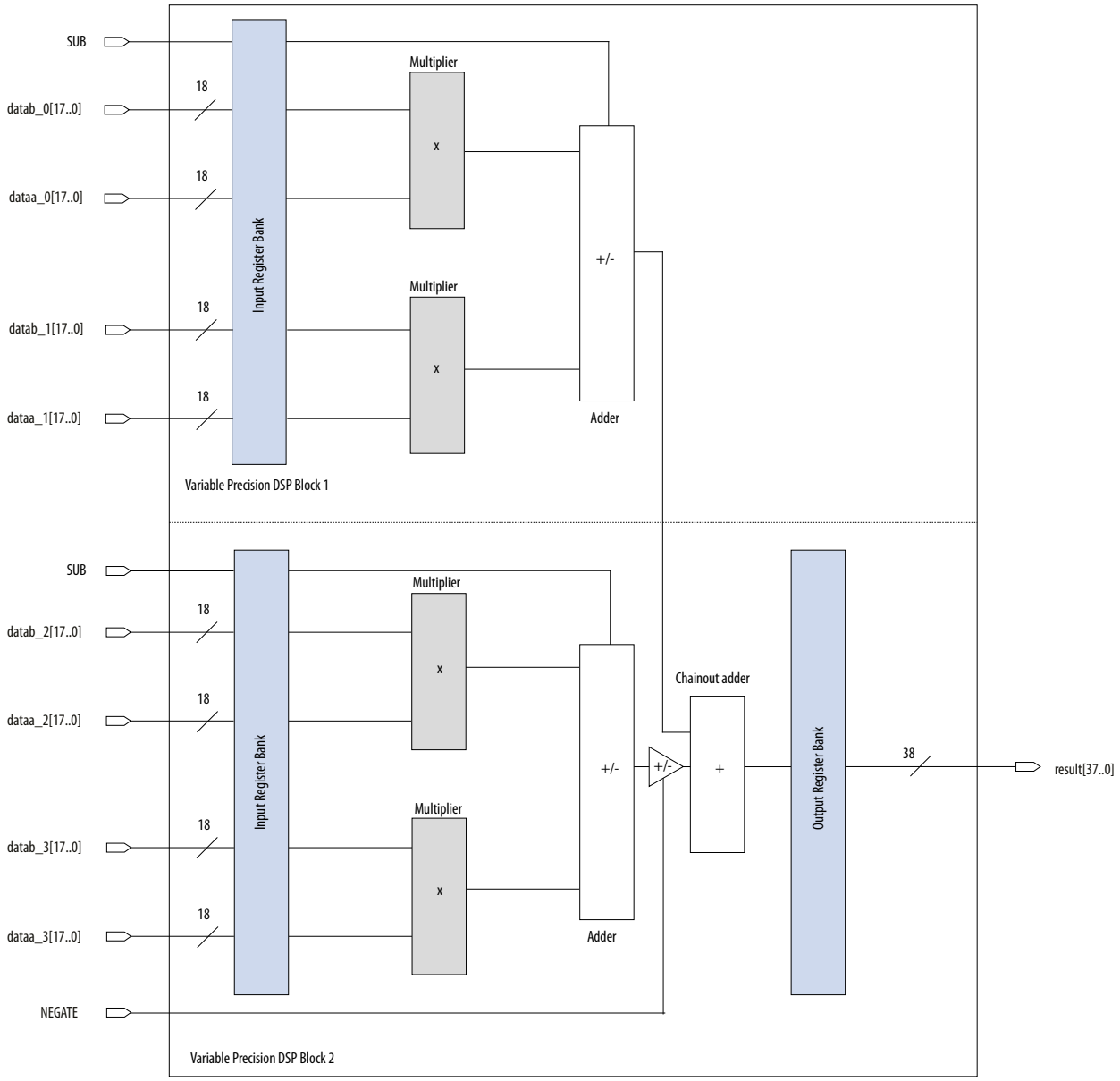
One Sum of Two 36 x 18 Multipliers

Figure 3-28: One Sum of Two 36 x 18 Multipliers with Two Variable Precision DSP Blocks for Arria V GZ Devices



One Sum of Four 18 x 18 Multipliers

Figure 3-29: One Sum of Four 18 x 18 Multipliers with Two Variable Precision DSP Blocks for Arria V GZ Devices



Sum of Square Mode

The Arria V variable precision DSP block can implement one sum of square mode.

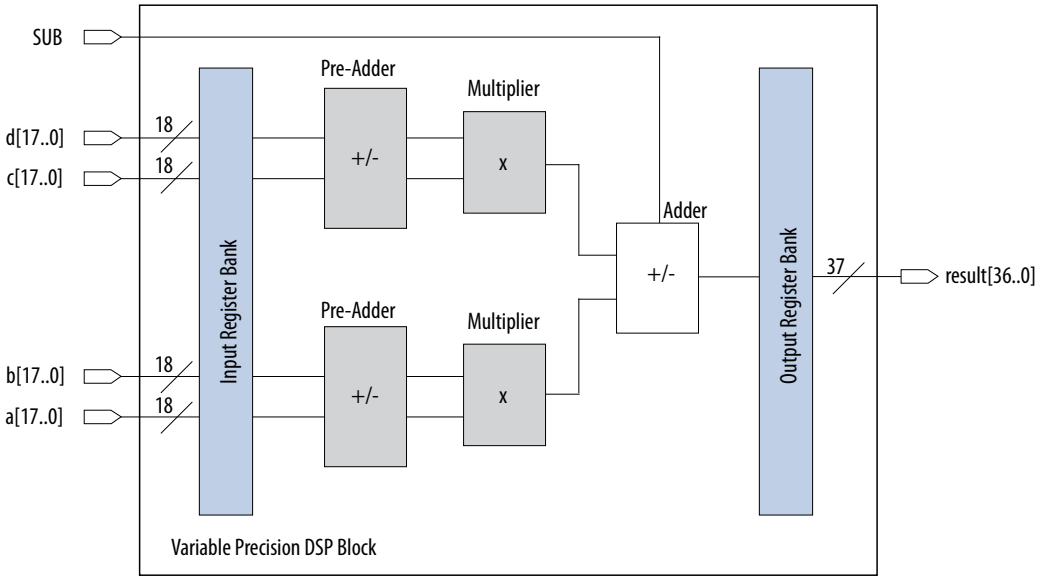
Figure 3-30: One Sum of Square Mode Equation

$$(a \pm b)^2 \times (c \pm d)^2$$

You can feed the four 18-bit inputs into the pre-adder block to convert `b` and `d` input as two's complement numbers to perform subtraction, if required.

You can feed each 18-bit pre-adder block output into both multiplicand and multiplier inputs of an 18 x 18 multiplier to generate a square result.

Figure 3-31: One Sum of Square Mode in a Variable Precision DSP Block for Arria V GZ Devices



18 x 18 Multiplication Summed with 36-Bit Input Mode

Arria V variable precision DSP blocks support one 18 x 18 multiplication summed to a 36-bit input.

Use the upper multiplier to provide the input for an 18 x 18 multiplication, while the bottom multiplier is bypassed.

The following signals are concatenated to produce a 36-bit input:

- Arria V GX, GT, SX, and ST devices: `datab_y1[17..0]` and `datab_y1[35..18]`
- Arria V GZ devices: `data1[17..0]` and `data1[35..18]`

Figure 3-32: One 18 x 18 Multiplication Summed with 36-Bit Input Mode for Arria V GX, GT, SX, and ST Devices

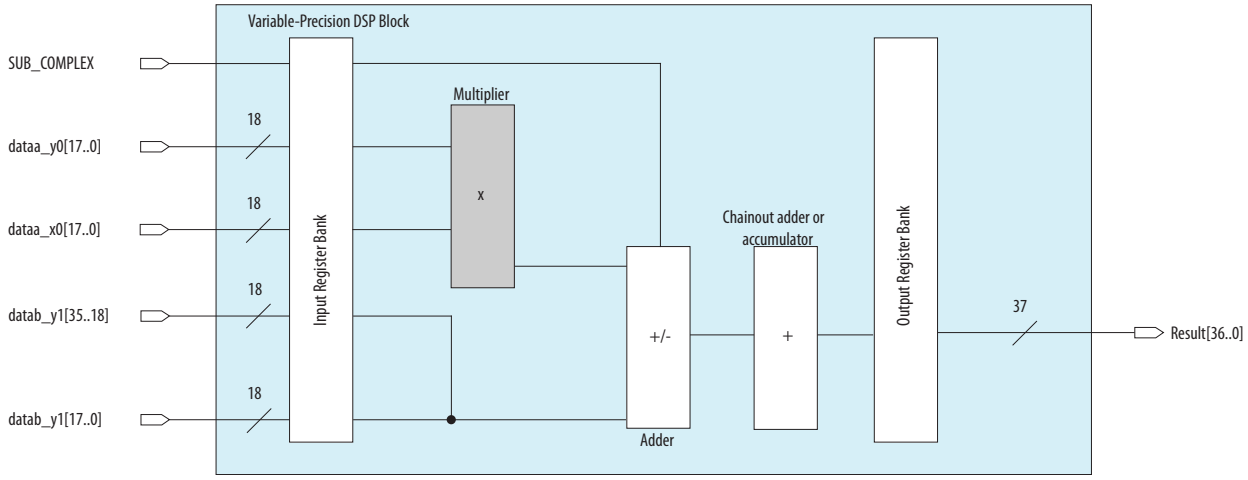
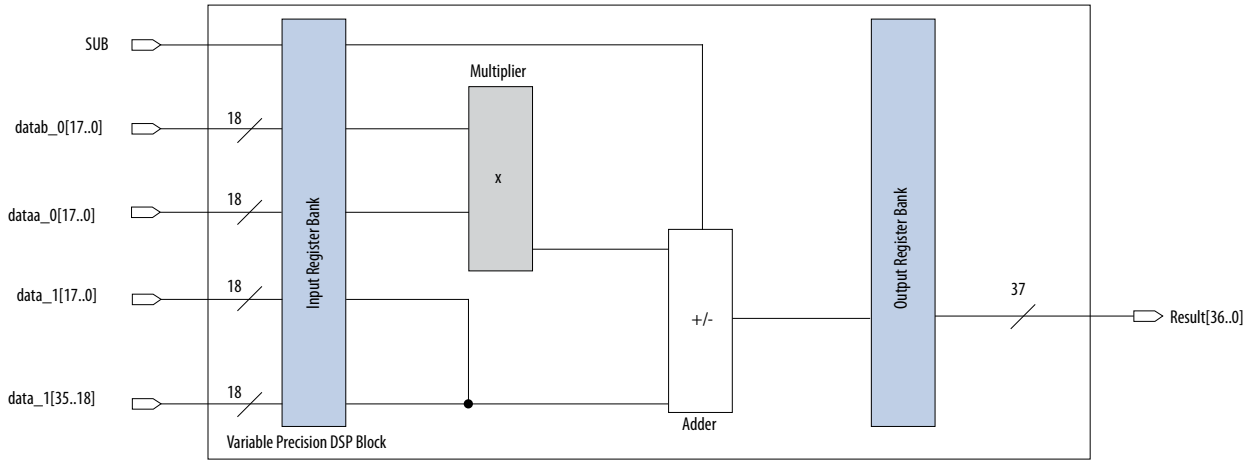


Figure 3-33: One 18 x 18 Multiplication Summed with 36-Bit Input Mode for Arria V GZ Devices



Systolic FIR Mode

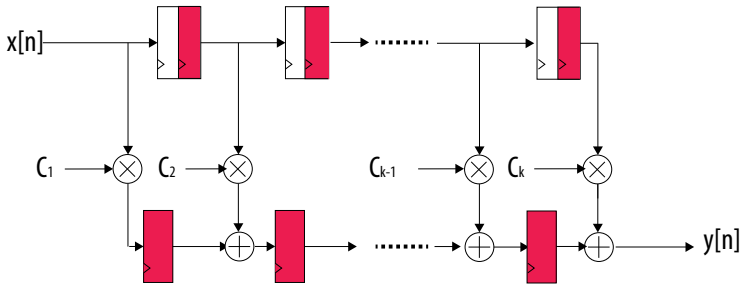
The basic structure of a FIR filter consists of a series of multiplications followed by an addition.

Figure 3-34: Basic FIR Filter Equation

$$y[n] = \sum_{i=1}^{k} c[i]x[n-i-1]$$

Depending on the number of taps and the input sizes, the delay through chaining a high number of adders can become quite large. To overcome the delay performance issue, the systolic form is used with additional delay elements placed per tap to increase the performance at the cost of increased latency.

Figure 3-35: Systolic FIR Filter Equivalent Circuit



Arria V variable precision DSP blocks support the following systolic FIR structures:

- 18-bit
- 27-bit

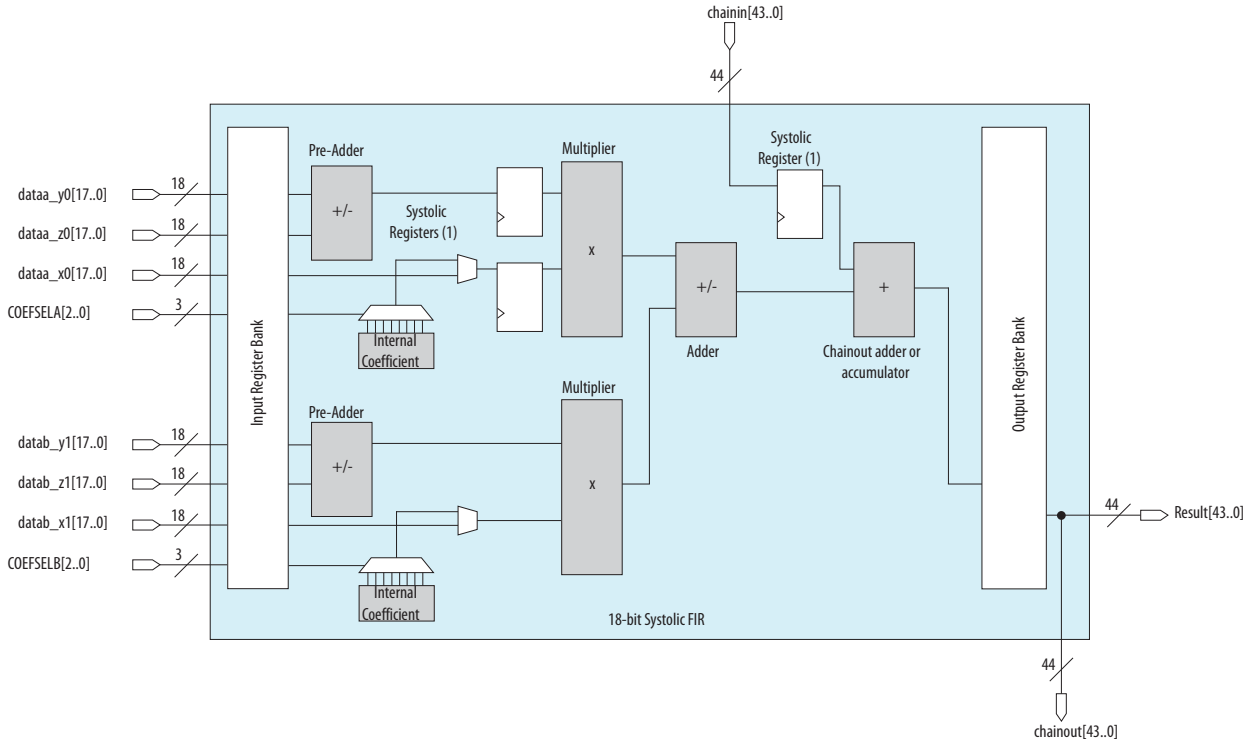
In systolic FIR mode, the input of the multiplier can come from four different sets of sources:

- Two dynamic inputs
- One dynamic input and one coefficient input
- One coefficient input and one pre-adder output
- One dynamic input and one pre-adder output (for Arria V GX, GT, SX, and ST devices only)

18-Bit Systolic FIR Mode

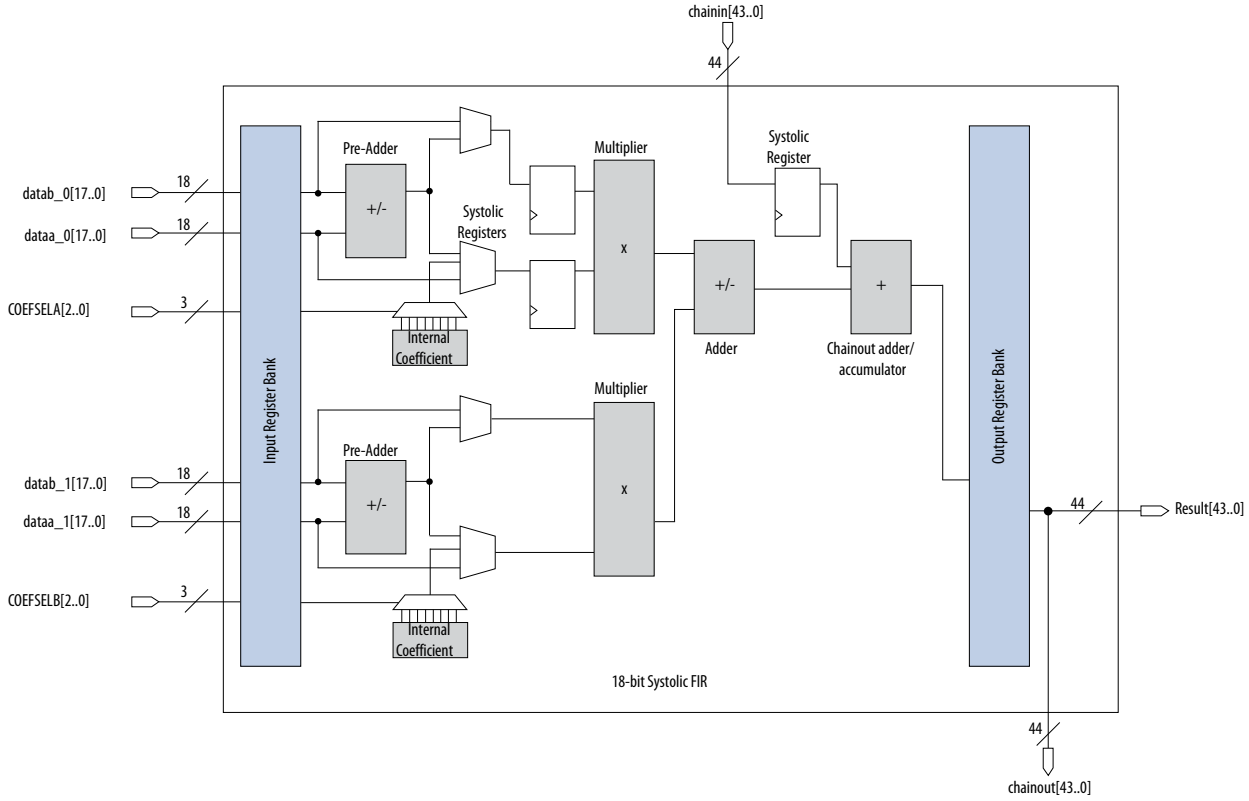
In 18-bit systolic FIR mode, the adders are configured as dual 44-bit adders, thereby giving 8 bits of overhead when using an 18-bit operation (36-bit products). This allows a total of 256 multiplier products.

Figure 3-36: 18-Bit Systolic FIR Mode for Arria V GX, GT, SX, and ST Devices



Note:
1. The systolic registers have the same clock source as the output register bank.

Figure 3-37: 18-Bit Systolic FIR Mode with Two Dynamic Inputs for Arria V GZ Devices



27-Bit Systolic FIR Mode

In 27-bit systolic FIR mode, the chainout adder or accumulator is configured for a 64-bit operation, providing 10 bits of overhead when using a 27-bit data (54-bit products). This allows a total of 1,024 multiplier products.

The 27-bit systolic FIR mode allows the implementation of one stage systolic filter per DSP block.

Figure 3-38: 27-Bit Systolic FIR Mode for Arria V GX, GT, SX, and ST Devices

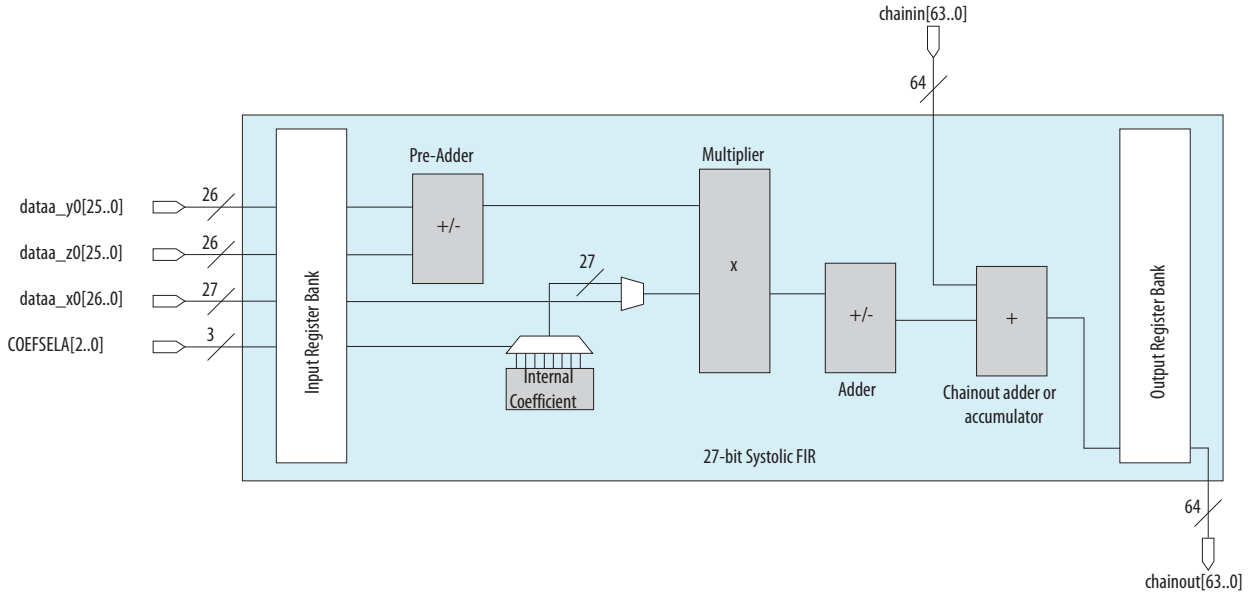
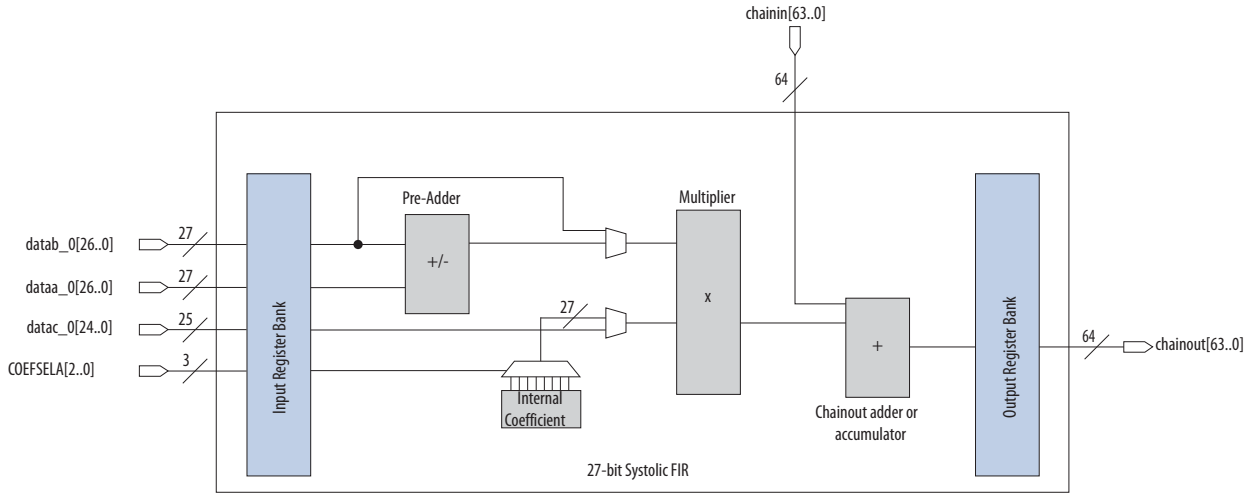


Figure 3-39: 27-Bit Systolic FIR Mode for Arria V GZ Devices



Variable Precision DSP Blocks in Arria V Devices Revision History

Date	Version	Changes
December 2015	2015.12.21	Changed instances of *Quartus II* to *Quartus Prime*.
June 2015	2015.06.12	• Updated Systolic FIR Filter Equivalent Circuit figure.

Date	Version	Changes
May 2015	2015.05.08	Added footnote in Features section to clarify certain features are only applicable to certain Arria V device variant.
June 2014	2014.06.30	Updated the supported megafunctions from ALTMULT_ADD and ALTMULT _ACCUM to ALTERA_MULT_ADD.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Moved all links to the Related Information section of respective topics for easy reference. • Updated the variable DSP blocks and multipliers counts for the Arria V SX and ST device variants. • Updated Figure 3-21, changed 37 to 38. • Updated Figure 3-22 by changing the Complex Multiplication Equation. • Update Figure 3-26, changed 37 to 38.
November 2012	2012.11.29	• Added resources for Arria V devices. • Updated design considerations for Arria V devices in operational modes. • Added DSP block architecture in 27 x 27 mode for Arria V GX, GT, SX, and ST devices. • Added DSP block architecture in 18 x 18 and 27 x 27 modes for Arria V GZ devices. • Updated DSP block architecture information on input register bank, pre-adder, multipliers, accumulator and chainout adder, and systolic registers for Arria V GZ devices. • Added 16 x 16, 18 x 18 (partial), 18 x 18, 36 x 18, and 36-bit independent multiplier modes for Arria V GZ devices. • Added 18 x 18, 18 x 25, and 27 x 27 independent complex multiplier modes for Arria V GZ devices. • Added 16 x 16, 18 x 18, 27 x 27, and 36 x 18 multiplier adder sum modes for Arria V GZ devices. • Added sum of square mode for Arria V GZ devices. • Added 18 x 18 multiplication summed with 36-bit input mode for Arria V GZ devices. • Added 18-bit and 27-bit systolic FIR modes for Arria V GZ devices. • Reorganized content and updated template.

Date	Version	Changes
June 2012	2.0	Updated for the Quartus II software v12.0 release: • Restructured chapter. • Added "Design Considerations", "Adder", and "Double Accumulation Register" sections. • Updated Figure 3–1 and Figure 3–13. • Added Table 3–3. • Updated "Systolic Registers" and "Systolic FIR Mode" sections. • Added Equation 3–2. • Added Figure 3–12.
May 2011	1.0	Initial release.

Clock Networks and PLLs in Arria V Devices

4

2020.04.13

AV-52004

Subscribe Send Feedback

This chapter describes the advanced features of hierarchical clock networks and phase-locked loops (PLLs) in Arria V devices. The Intel Quartus Prime software enables the PLLs and their features without external devices.

Related Information

Arria V Device Handbook: Known Issues
Lists the planned updates to the Arria V Device Handbook chapters.

Clock Networks

The Arria V devices contain the following clock networks that are organized into a hierarchical structure:

- Global clock (GCLK) networks
- Regional clock (RCLK) networks
- Periphery clock (PCLK) networks

Clock Resources in Arria V Devices

Table 4-1: Clock Resources in Arria V Devices

Clock Resource	Device	Number of Resources Available	Source of Clock Resource
Clock input pins	• Arria V GX A1 and A3 • Arria V GT C3	40 single-ended or 20 differential	`CLK[0..7][p,n]` and `CLK[12..23][p,n]` pins
	• Arria V SX B3 and B5 • Arria V ST D3 and D5	40 single-ended or 20 differential	`CLK[0..11][p,n]` and `CLK[16..23][p,n]` pins
	• Arria V GX A5, A7, B1, B3, B5, and B7 • Arria V GT C7, D3, and D7 • Arria V GZ E1, E3, E5, and E7	48 single-ended or 24 differential	`CLK[0..23][p,n]` pins

Clock Resource	Device	Number of Resources Available	Source of Clock Resource
GCLK and RCLK networks	• Arria V GX A1 and A3 • Arria V GT C3	76	`CLK[0..7][p,n]` and `CLK[12..23][p,n]` pins, PLL clock outputs, and logic array
	• Arria V SX B3 and B5 • Arria V ST D3 and D5	82	`CLK[0..11][p,n]` and `CLK[16..23][p,n]` pins, PLL clock outputs, and logic array
	• Arria V GX A5, A7, B1, B3, B5, and B7 • Arria V GT C7, D3, and D7	88	`CLK[0..23][p,n]` pins, PLL clock outputs, and logic array
	Arria V GZ E1, E3, E5, and E7	92	
PCLK networks	• Arria V GX A1 and A3 • Arria V GT C3	120	DPA clock outputs, PLD-transceiver interface clocks, I/O pins, and logic array
	• Arria V GX A5 and A7 • Arria V GT C7	184	
	• Arria V SX B3 and B5 • Arria V ST D3 and D5	208	
	Arria V GZ E1 and E3	210	
	• Arria V GX B1 and B3 • Arria V GT D3	224	
	• Arria V GX B5 and B7 • Arria V GT D7	248	
	Arria V GZ E5 and E7	282	

For more information about the clock input pins connections, refer to the pin connection guidelines.

Related Information

- **Arria V GT and GX Device Family Pin Connection Guidelines**
- **Arria V GZ Device Family Pin Connection Guidelines**

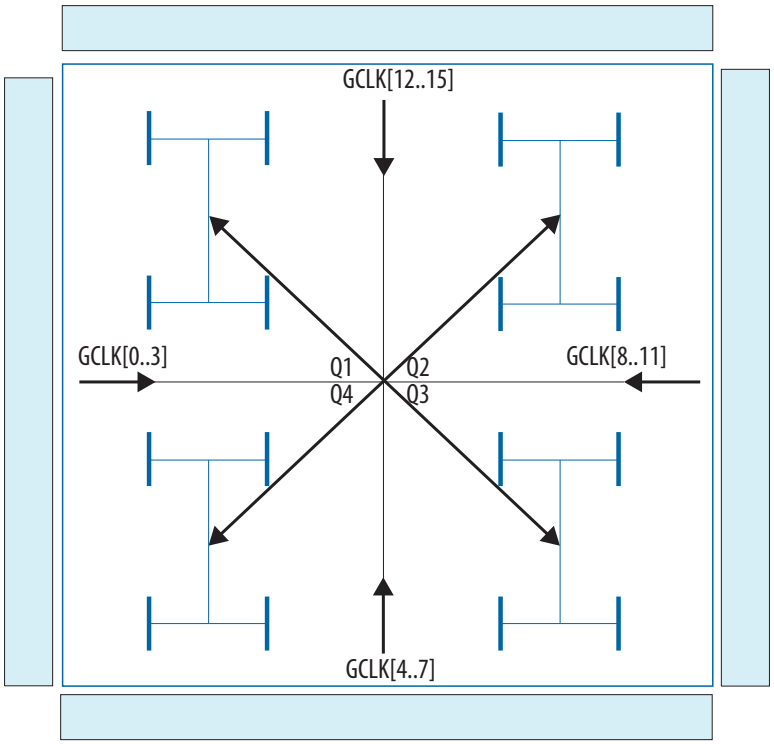
Types of Clock Networks

Global Clock Networks

Arria V devices provide GCLKs that can drive throughout the device. The GCLKs serve as low-skew clock sources for functional blocks, such as adaptive logic modules (ALMs), digital signal processing (DSP), embedded memory, and PLLs. Arria V I/O elements (IOEs) and internal logic can also drive GCLKs to create internally-generated global clocks and other high fan-out control signals, such as synchronous or asynchronous clear and clock enable signals.

Figure 4-1: GCLK Networks in Arria V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Regional Clock Networks

RCLK networks are only applicable to the quadrant they drive into. RCLK networks provide the lowest clock insertion delay and skew for logic contained within a single device quadrant. The Arria V IOEs and internal logic within a given quadrant can also drive RCLKs to create internally generated regional clocks and other high fan-out control signals.

Figure 4-2: RCLK Networks in Arria V GX, GT, SX, and ST Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

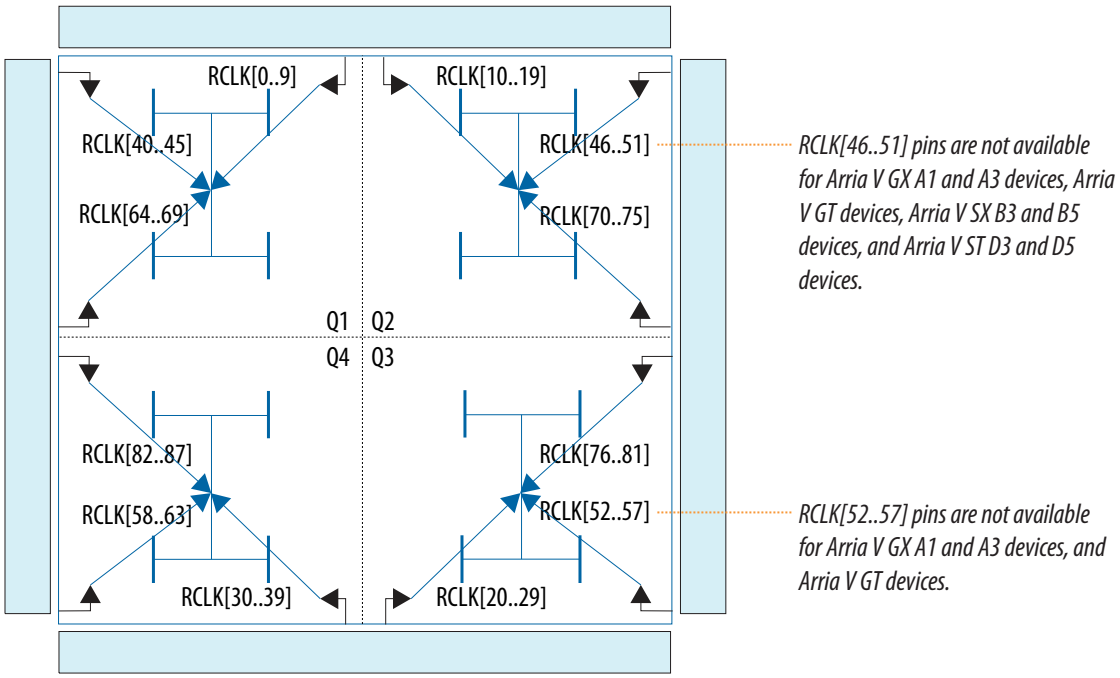
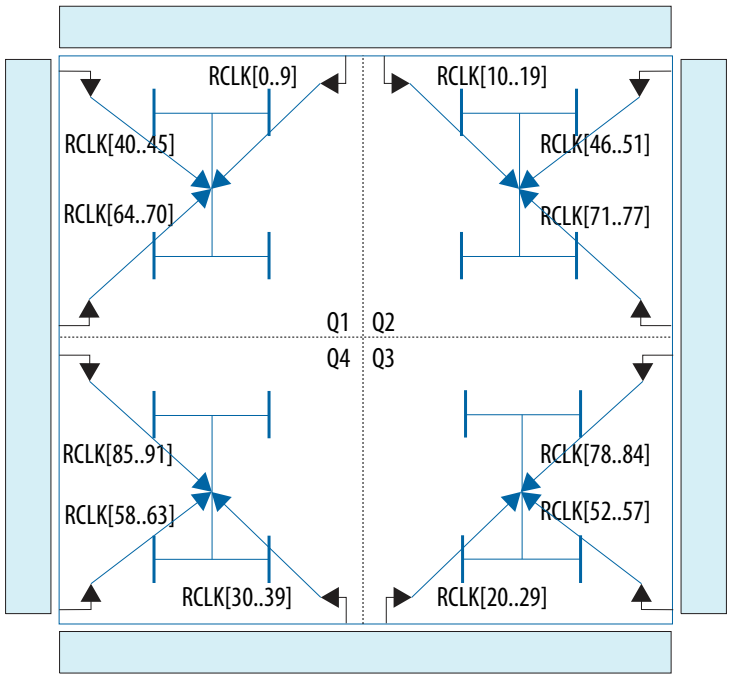


Figure 4-3: RCLK Networks in Arria V GZ Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Periphery Clock Networks

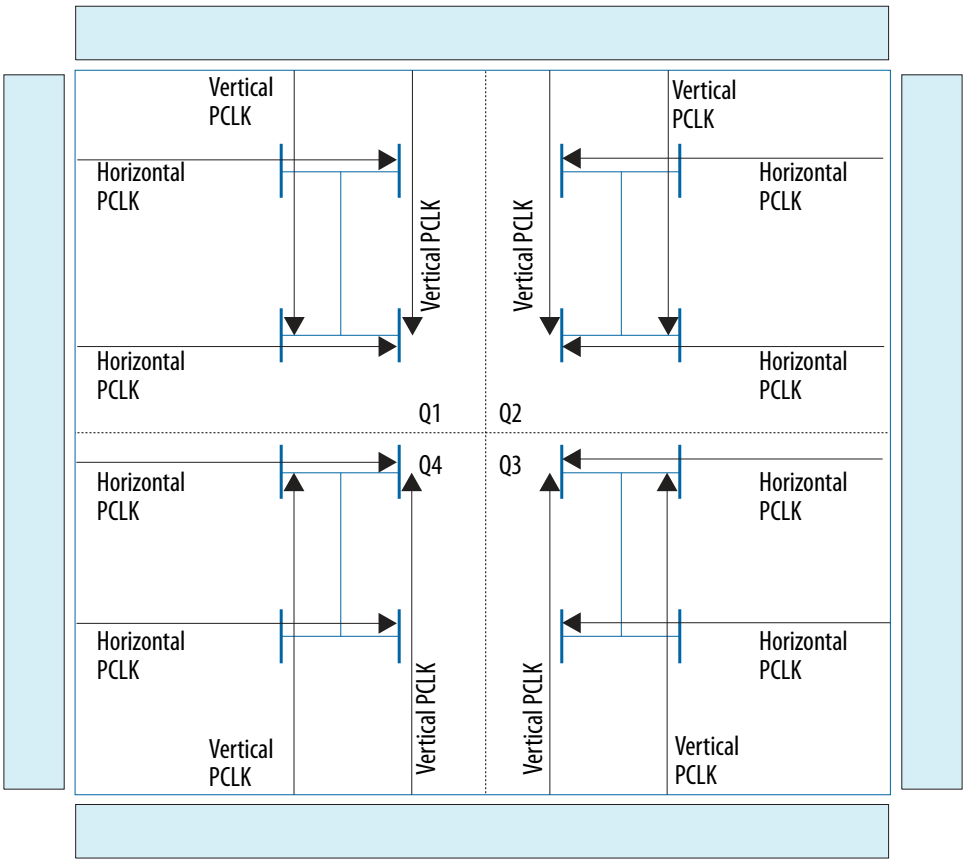
Depending on the routing direction, Arria V devices provide vertical PCLKs from the top and bottom periphery, and horizontal PCLKs from the left and right periphery.

Clock outputs from the dynamic phase aligner (DPA) block, programmable logic device (PLD)-transceiver interface clocks, I/O pins, and internal logic can drive the PCLK networks.

PCLKs have higher skew when compared with GCLK and RCLK networks. You can use PCLKs for general purpose routing to drive signals into and out of the Arria V device.

Figure 4-4: PCLK Networks in Arria V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



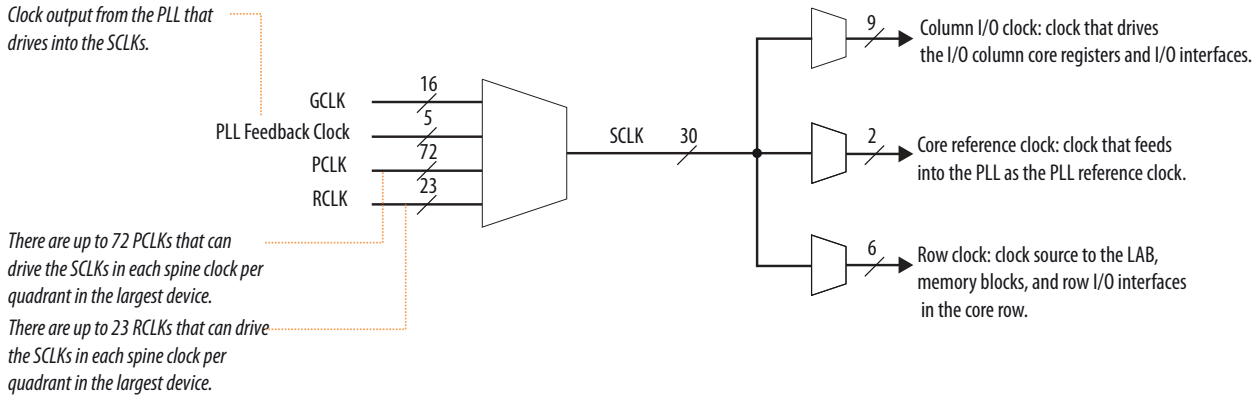
Clock Sources Per Quadrant

The Arria V devices provide 30 section clock (SCLK) networks in each spine clock per quadrant. The SCLK networks can drive six row clocks in each logic array block (LAB) row, nine column I/O clocks, and two core reference clocks. The SCLKs are the clock resources to the core functional blocks, PLLs, and I/O interfaces of the device.

A spine clock is another layer of routing between the GCLK, RCLK, and PCLK networks before each clock is connected to the clock routing for each LAB row. The settings for spine clocks are transparent. The Intel Quartus Prime software automatically routes the spine clock based on the GCLK, RCLK, and PCLK networks.

The following figure shows SCLKs driven by the GCLK, RCLK, PCLK, or the PLL feedback clock networks in each spine clock per quadrant. The GCLK, RCLK, PCLK, and PLL feedback clocks share the same routing to the SCLKs. To ensure successful design fitting in the Intel Quartus Prime software, the total number of clock resources must not exceed the SCLK limits in each region.

Figure 4-5: Hierarchical Clock Networks in Each Spine Clock Per Quadrant



Types of Clock Regions

This section describes the types of clock regions in Arria V devices.

Entire Device Clock Region

To form the entire device clock region, a source drives a signal in a GCLK network that can be routed through the entire device. The source is not necessarily a clock signal. This clock region has the maximum insertion delay when compared with other clock regions, but allows the signal to reach every destination in the device. It is a good option for routing global reset and clear signals or routing clocks throughout the device.

Regional Clock Region

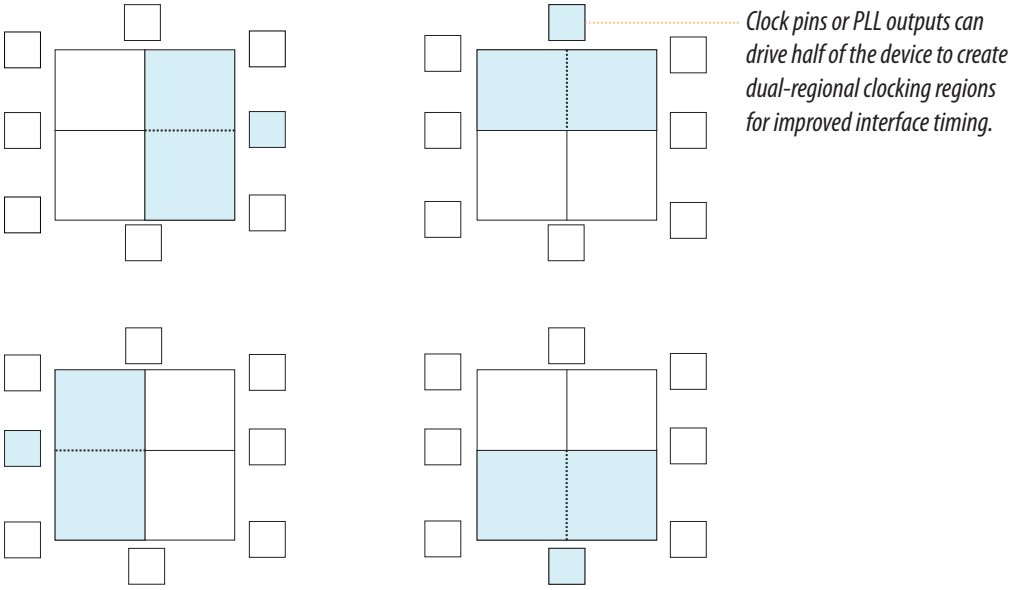
To form a regional clock region, a source drives a signal in a RCLK network that you can route throughout one quadrant of the device. This clock region provides the lowest skew in a quadrant. It is a good option if all the destinations are in a single quadrant.

Dual-Regional Clock Region

To form a dual-regional clock region, a single source (a clock pin or PLL output) generates a dual-regional clock by driving two RCLK networks (one from each quadrant). This technique allows destinations across two adjacent device quadrants to use the same low-skew clock. The routing of this signal on an entire side has approximately the same delay as a RCLK region. Internal logic can also drive a dual-regional clock network.

Figure 4-6: Dual-Regional Clock Region for Arria V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Clock Network Sources

In Arria V devices, clock input pins, PLL outputs, high-speed serial interface (HSSI) outputs, DPA outputs, and internal logic can drive the GCLK, RCLK, and PCLK networks.

Dedicated Clock Input Pins

You can use the dedicated clock input pins (`CLK[0..23][p,n]`) for high fan-out control signals, such as asynchronous clears, presets, and clock enables, for protocol signals through the GCLK or RCLK networks.

`CLK` pins can be either differential clocks or single-ended clocks. When you use the `CLK` pins as single-ended clock inputs, only the `CLK<#>p` pins have dedicated connections to the PLL. The `CLK<#>n` pins drive the PLLs over global or regional clock networks and do not have dedicated routing paths to the PLLs.

Driving a PLL over a global or regional clock can lead to higher jitter at the PLL input, and the PLL will not be able to fully compensate for the global or regional clock. Altera recommends using the `CLK<#>p` pins for optimal performance when you use single-ended clock inputs to drive the PLLs.

Internal Logic

You can drive each GCLK, RCLK, and horizontal PCLK network using LAB-routing and row clock to enable internal logic to drive a high fan-out, low-skew signal.

Note: Internally-generated GCLKs, RCLKs, or PCLKs cannot drive the Arria V PLLs. The input clock to the PLL has to come from dedicated clock input pins, PLL-fed GCLKs, or PLL-fed RCLKs.

DPA Outputs

Every DPA generates one PCLK to the core.

Related Information

High-Speed I/O Design Guidelines for Arria V Devices on page 6-7
Provides more information about DPA and HSSI outputs.

HSSI Outputs

Every three HSSI outputs generate a group of six PCLKs to the core.

Related Information

High-Speed I/O Design Guidelines for Arria V Devices on page 6-7
Provides more information about DPA and HSSI outputs.

PLL Clock Outputs

The Arria V PLL clock outputs can drive both GCLK and RCLK networks.

Clock Input Pin Connections to GCLK and RCLK Networks

Table 4-2: Dedicated Clock Input Pin Connectivity to the GCLK Networks for Arria V Devices

Clock Resources	CLK (p/n Pins)
GCLK[0,1,2,3]	CLK[0,1,2,3,20,21,22,23]
GCLK[4,5,6,7]	CLK[4,5,6,7]
GCLK[8,9,10,11]	CLK[8,9,10,11] and [5] CLK[12,13,14,15][6]
GCLK[12,13,14,15]	CLK[16,17,18,19]

Table 4-3: Dedicated Clock Input Pin Connectivity to the RCLK Networks for Arria V GX, GT, SX, and ST Devices

A given clock input pin can drive two adjacent RCLK networks to create a dual-regional clock network.

Clock Resources	CLK (p/n Pins)
RCLK[58,59,60,61,62,63,64,68,82,86]	CLK[0]
RCLK[58,59,60,61,62,63,65,69,83,87]	CLK[1]
RCLK[58,59,60,61,62,63,66,84]	CLK[2]
RCLK[58,59,60,61,62,63,67,85]	CLK[3]
RCLK[20,24,28,30,34,38]	CLK[4]
RCLK[21,25,29,31,35,39]	CLK[5]
RCLK[22,26,32,36]	CLK[6]

[5] CLK[8,9,10,11] are not available for Arria V GX A1 and A3 devices, and Arria V GT devices.
[6] CLK[12,13,14,15] are not available for Arria V Arria V SX B3 and B5 devices, and Arria V ST D3 and D5 devices.

Clock Resources	CLK (p/n Pins)
RCLK[23,27,33,37]	CLK[7]
RCLK[52,53,54,55,56,57,70,74,76,80]	CLK[8] (5)
RCLK[52,53,54,55,56,57,71,75,77,81]	CLK[9] (5)
RCLK[52,53,54,55,56,57,72,78]	CLK[10] (5)
RCLK[52,53,54,55,56,57,73,79]	CLK[11] (5)
RCLK[46,47,48,49,50,51,70,74,76,80] (7)	CLK[12] (6)
RCLK[46,47,48,49,50,51,71,75,77,81] (7)	CLK[13] (6)
RCLK[46,47,48,49,50,51,72,78] (7)	CLK[14] (6)
RCLK[46,47,48,49,50,51,73,79] (7)	CLK[15] (6)
RCLK[0,4,8,10,14,18]	CLK[16]
RCLK[1,5,9,11,15,19]	CLK[17]
RCLK[2,6,12,16]	CLK[18]
RCLK[3,7,13,17]	CLK[19]
RCLK[40,41,42,43,44,45,64,68,82,86]	CLK[20]
RCLK[40,41,42,43,44,45,65,69,83,87]	CLK[21]
RCLK[40,41,42,43,44,45,66,84]	CLK[22]
RCLK[40,41,42,43,44,45,67,85]	CLK[23]

Table 4-4: Dedicated Clock Input Pin Connectivity to the RCLK Networks for Arria V GZ Devices

A given clock input pin can drive two adjacent RCLK networks to create a dual-regional clock network.

Clock Resources	CLK (p/n Pins)
RCLK[58,59,60,61,62,63,64,68,85,89]	CLK[0]
RCLK[58,59,60,61,62,63,65,69,86,90]	CLK[1]
RCLK[58,59,60,61,62,63,66,70,87,91]	CLK[2]
RCLK[58,59,60,61,62,63,67,88]	CLK[3]
RCLK[20,24,28,30,34,38]	CLK[4]
RCLK[21,25,29,31,35,39]	CLK[5]
RCLK[22,26,32,36]	CLK[6]
RCLK[23,27,33,37]	CLK[7]
RCLK[52,53,54,55,56,57,71,75,78,82]	CLK[8]
RCLK[52,53,54,55,56,57,72,76,79,83]	CLK[9]
RCLK[52,53,54,55,56,57,73,77,80,84]	CLK[10]

(7) RCLK[46..51] are not available for Arria V GX A1 and A3 devices, and Arria V GT devices.

Clock Resources	CLK (p/n Pins)
RCLK[52,53,54,55,56,57,74,81]	CLK[11]
RCLK[46,47,48,49,50,51,71,75,78,82]	CLK[12]
RCLK[46,47,48,49,50,51,72,76,79,83]	CLK[13]
RCLK[46,47,48,49,50,51,73,77,80,84]	CLK[14]
RCLK[46,47,48,49,50,51,74,81]	CLK[15]
RCLK[0,4,8,10,14,18]	CLK[16]
RCLK[1,5,9,11,15,19]	CLK[17]
RCLK[2,6,12,16]	CLK[18]
RCLK[3,7,13,17]	CLK[19]
RCLK[40,41,42,43,44,45,64,68,85,89]	CLK[20]
RCLK[40,41,42,43,44,45,65,69,86,90]	CLK[21]
RCLK[40,41,42,43,44,45,66,70,87,91]	CLK[22]
RCLK[40,41,42,43,44,45,67,88]	CLK[23]

Clock Output Connections

For Arria V PLL connectivity to GCLK and RCLK networks, refer to the PLL connectivity to GCLK and RCLK networks spreadsheet.

Related Information

PLL Connectivity to GCLK and RCLK Networks for Arria V Devices

Clock Control Block

Every GCLK, RCLK, and PCLK network has its own clock control block. The control block provides the following features:

- Clock source selection (dynamic selection available only for GCLKs)
- Global clock multiplexing
- Clock power down (static or dynamic clock enable or disable available only for GCLKs and RCLKs)

Pin Mapping in Arria V Devices

Table 4-5: Mapping Between the Input Clock Pins, PLL Counter Outputs, and Clock Control Block Inputs

Clock	Fed by
inclk[0] and inclk[1]	Any of the four dedicated clock pins on the same side of the Arria V device.
inclk[2]	PLL counters C0 and C2 from the two center PLLs on the same side of the Arria V devices.
inclk[3]	PLL counters C1 and C3 from the two center PLLs on the same side of the Arria V devices.

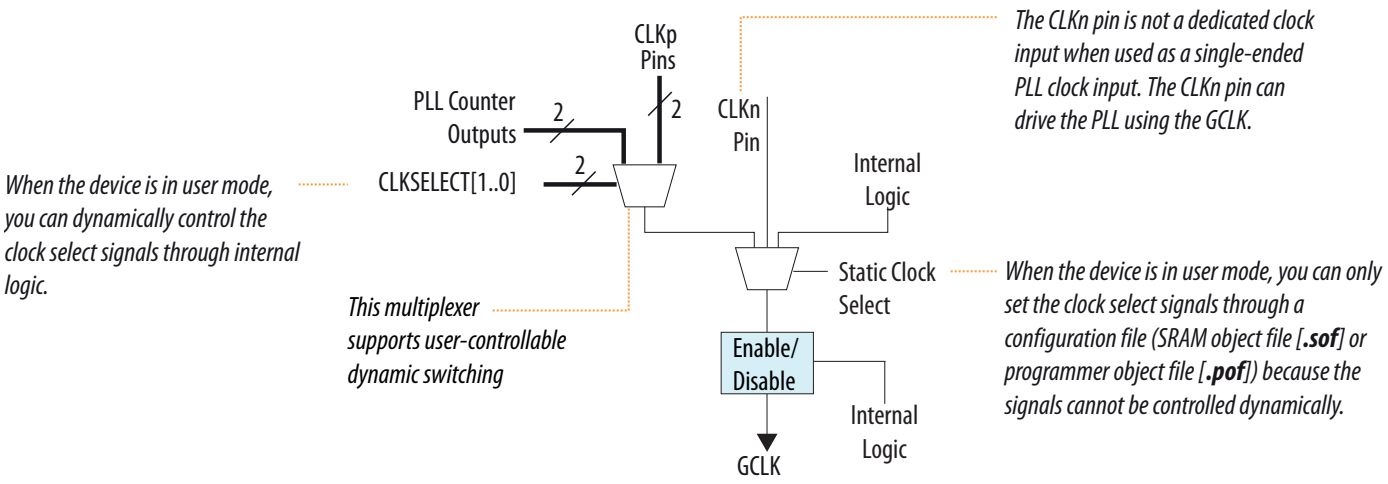
Note: You cannot use corner PLLs for dynamic clock control selection.

GCLK Control Block

You can select the clock source for the GCLK select block either statically or dynamically using internal logic to drive the multiplexer-select inputs.

When selecting the clock source dynamically, you can select either PLL outputs (such as `C0` or `C1`), or a combination of clock pins or PLL outputs.

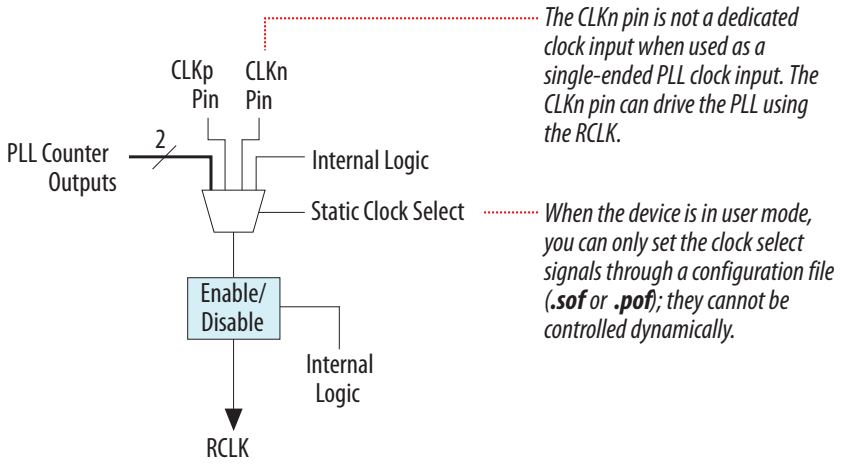
Figure 4-7: GCLK Control Block for Arria V Devices



RCLK Control Block

You can only control the clock source selection for the RCLK select block statically using configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Intel Quartus Prime software.

Figure 4-8: RCLK Control Block for Arria V Devices



You can set the input clock sources and the `clkena` signals for the GCLK and RCLK network multiplexers through the Intel Quartus Prime software using the ALTCLKCTRL IP core.

Note: When selecting the clock source dynamically using the ALTCLKCTRL IP core, choose the inputs using the `CLKSELECT[0..1]` signal. The inputs from the clock pins feed the `inclk[0..1]` ports of the multiplexer, and the PLL outputs feed the `inclk[2..3]` ports.

Related Information

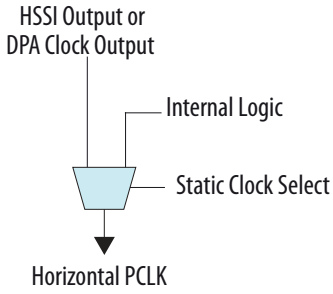
Clock Control Block (ALTCLKCTRL) IP Core User Guide
Provides more information about ALTCLKCTRL IP core.

PCLK Control Block

To drive the HSSI horizontal PCLK control block, select the HSSI output or internal logic .

To drive the DPA vertical PCLK, select the DPA clock output or internal logic . You can only use the DPA clock output to generate the vertical PCLK to the core.

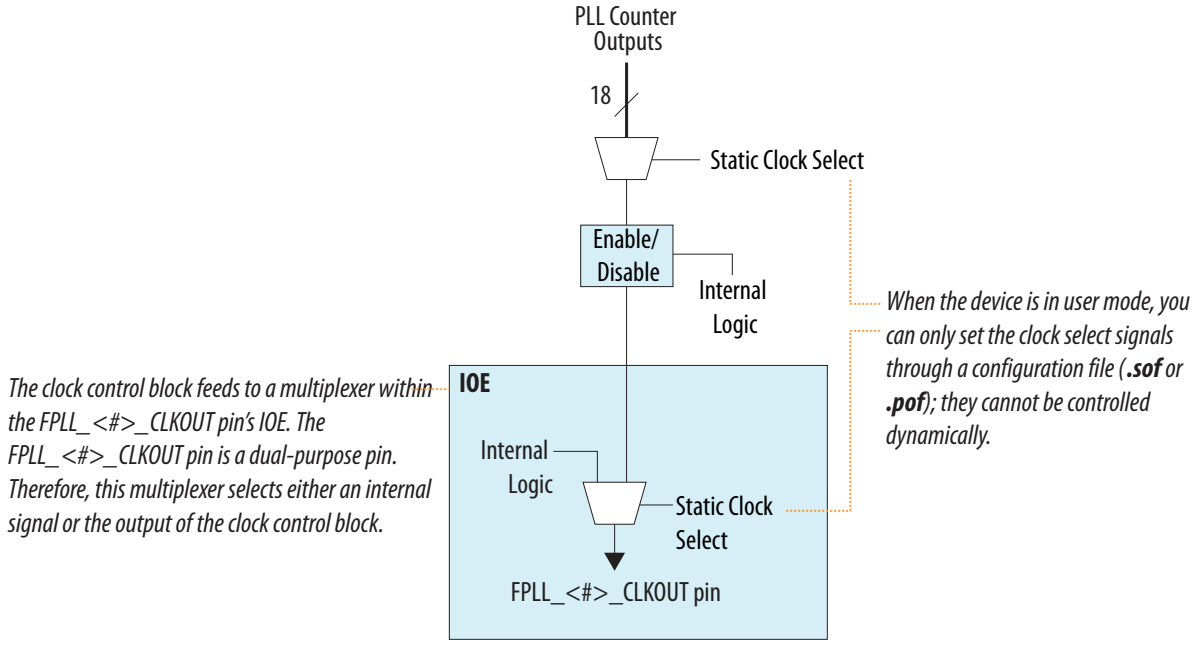
Figure 4-9: Horizontal PCLK Control Block for Arria V Devices



External PLL Clock Output Control Block

You can enable or disable the dedicated external clock output pins using the ALTCLKCTRL IP core.

Figure 4-10: External PLL Output Clock Control Block for Arria V Devices



Related Information

Clock Control Block (ALTCLKCTRL) IP Core User Guide
Provides more information about ALTCLKCTRL IP core.

Clock Power Down

You can power down the GCLK and RCLK clock networks using both static and dynamic approaches.

When a clock network is powered down, all the logic fed by the clock network is in off-state, reducing the overall power consumption of the device. The unused GCLK, RCLK, and PCLK networks are automatically powered down through configuration bit settings in the configuration file (**.sof** or **.pof**) generated by the Intel Quartus Prime software.

The dynamic clock enable or disable feature allows the internal logic to control power-up or power-down synchronously on the GCLK and RCLK networks, including dual-regional clock regions. This feature is independent of the PLL and is applied directly on the clock network.

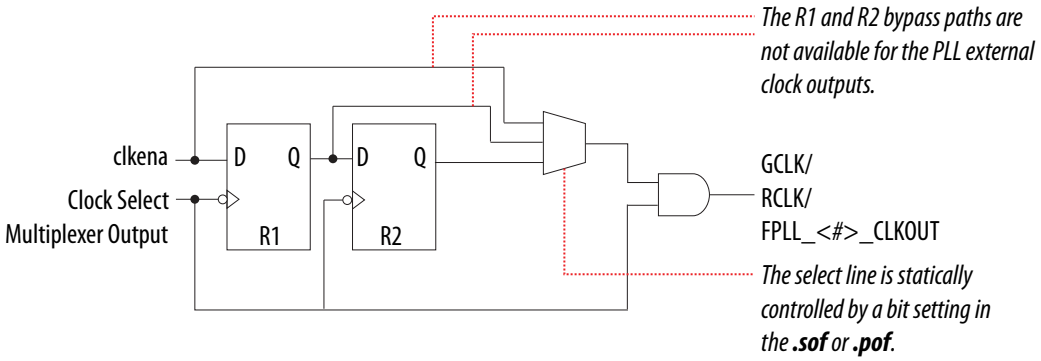
Note: You cannot dynamically enable or disable GCLK or RCLK networks that drive PLLs.

Clock Enable Signals

You cannot use the clock enable and disable circuit of the clock control block if the GCLK or RCLK output drives the input of a PLL.

Figure 4-11: `clkena` Implementation with Clock Enable and Disable Circuit

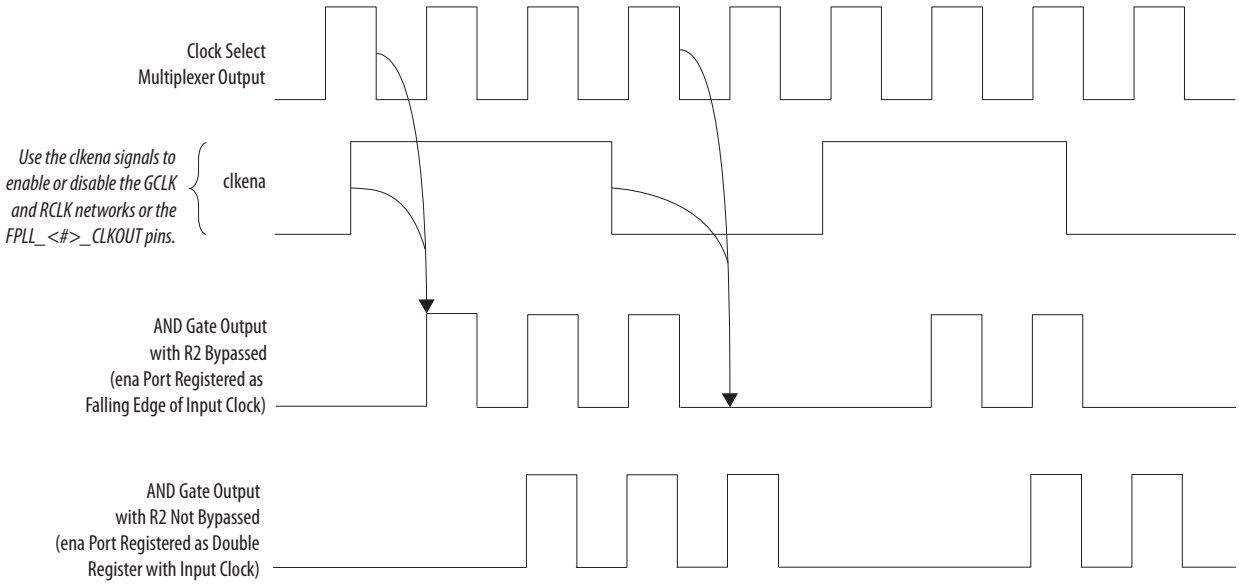
This figure shows the implementation of the clock enable and disable circuit of the clock control block.



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

Figure 4-12: Example of `clkena` Signals

This figure shows a waveform example for a clock output enable. The `clkena` signal is synchronous to the falling edge of the clock output.



Arria V devices have an additional metastability register that aids in asynchronous enable and disable of the GCLK and RCLK networks. You can optionally bypass this register in the Intel Quartus Prime software.

The PLL can remain locked, independent of the `clkena` signals, because the loop-related counters are not affected. This feature is useful for applications that require a low-power or sleep mode. The `clkena` signal

can also disable clock outputs if the system is not tolerant of frequency overshoot during resynchronization.

Arria V PLLs

PLLs provide robust clock management and synthesis for device clock management, external system clock management, and high-speed I/O interfaces.

The Arria V device family contains fractional PLLs that can function as fractional PLLs or integer PLLs. The output counters in Arria V devices are dedicated to each fractional PLL that support integer or fractional frequency synthesis.

Two adjacent PLLs share 18 `C` output counters. Any number of `C` counters can be assigned to each PLL, as long as the total number used by the two PLLs is 18 or less.

The Arria V devices offer up to 16 fractional PLLs in the larger densities. All Arria V fractional PLLs have the same core analog structure and features support.

Table 4-6: PLL Features in Arria V Devices

Feature	Support
Integer PLL	Yes
Fractional PLL	Yes
`C` output counters	18
`M`, `N`, `C` counter sizes	1 to 512
Dedicated external clock outputs	4 single-ended or 2 single-ended and 1 differential
Dedicated clock input pins	4 single-ended or 4 differential
External feedback input pin	Single-ended or differential
Spread-spectrum input clock tracking	Yes (8)
Source synchronous compensation	Yes
Direct compensation	Yes
Normal compensation	Yes
Zero-delay buffer compensation	Yes
External feedback compensation	Yes
LVDS compensation	Yes
Voltage-controlled oscillator (VCO) output drives the DPA clock	Yes
Phase shift resolution	78.125 ps (9)

(8) Provided input clock jitter is within input jitter tolerance specifications. The modulation frequency of the input clock is below the PLL bandwidth which is specified in the Fitter report.

(9) The smallest phase shift is determined by the VCO period divided by eight. For degree increments, the Arria V device can shift all output frequencies in increments of at least 45°. Smaller degree increments are possible depending on the frequency and divide parameters.

Feature	Support
Programmable duty cycle	Yes
Power down mode	Yes

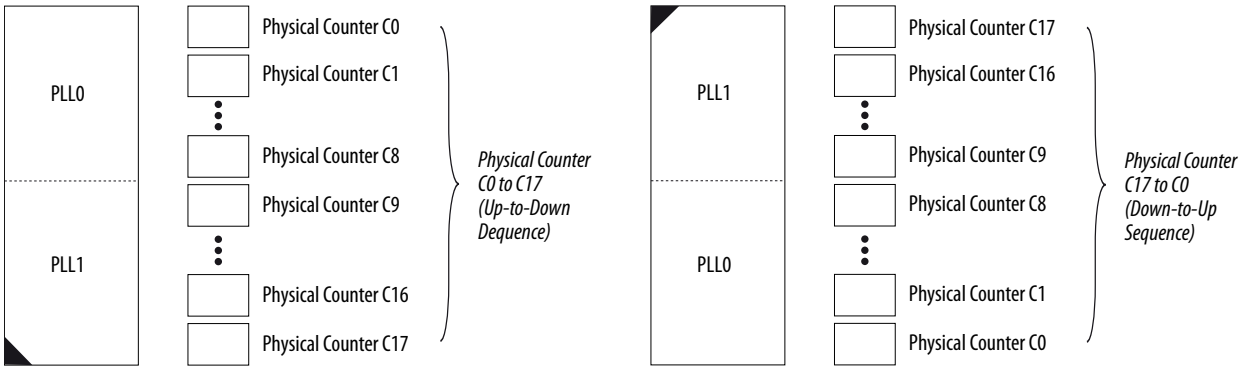
PLL Physical Counters in Arria V Devices

The physical counters for the fractional PLLs are arranged in the following sequences:

- Up-to-down
- Down-to-up

Figure 4-13: PLL Physical Counters Orientation for Arria V Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



PLL Locations in Arria V Devices

Arria V devices provide PLLs for the transceiver channels. These PLLs are located in a strip, where the strip refers to an area in the FPGA.

The total number of PLLs in the Arria V devices includes the PLLs in the PLL strip. However, the transceivers can only use the PLLs located in the strip.

The following figures show the physical locations of the fractional PLLs. Every index represents one fractional PLL in the device. The physical locations of the fractional PLLs correspond to the coordinates in the Quartus II Chip Planner.

Figure 4-14: PLL Locations for Arria V GX A1 and A3 Devices, and Arria V GT C3 Device

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

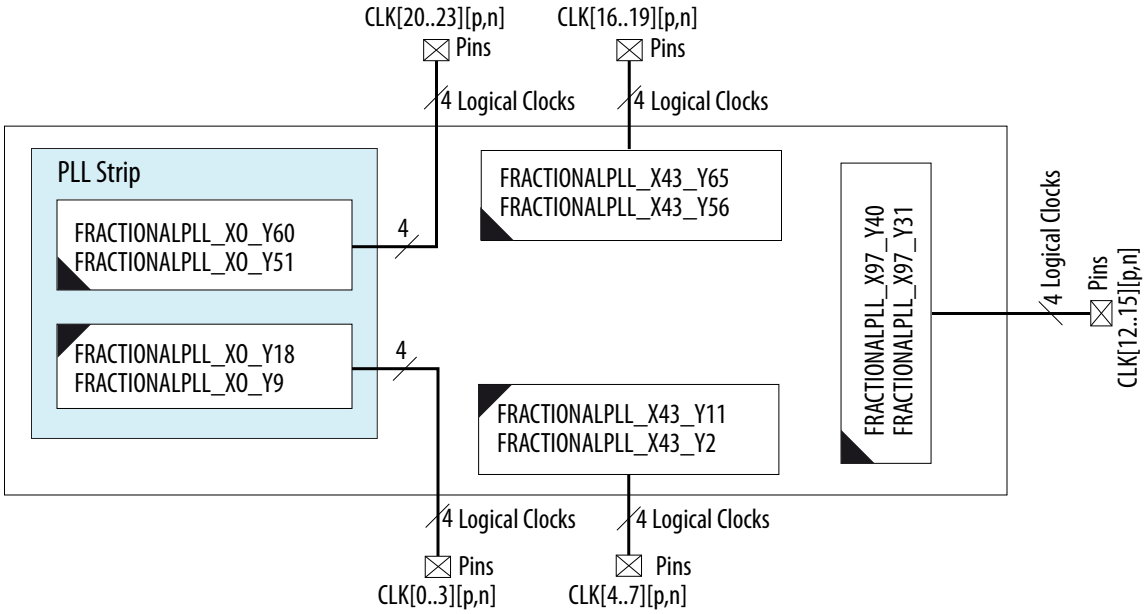


Figure 4-15: PLL Locations for Arria V GX A5 and A7 Devices, and Arria V GT C7 Device

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

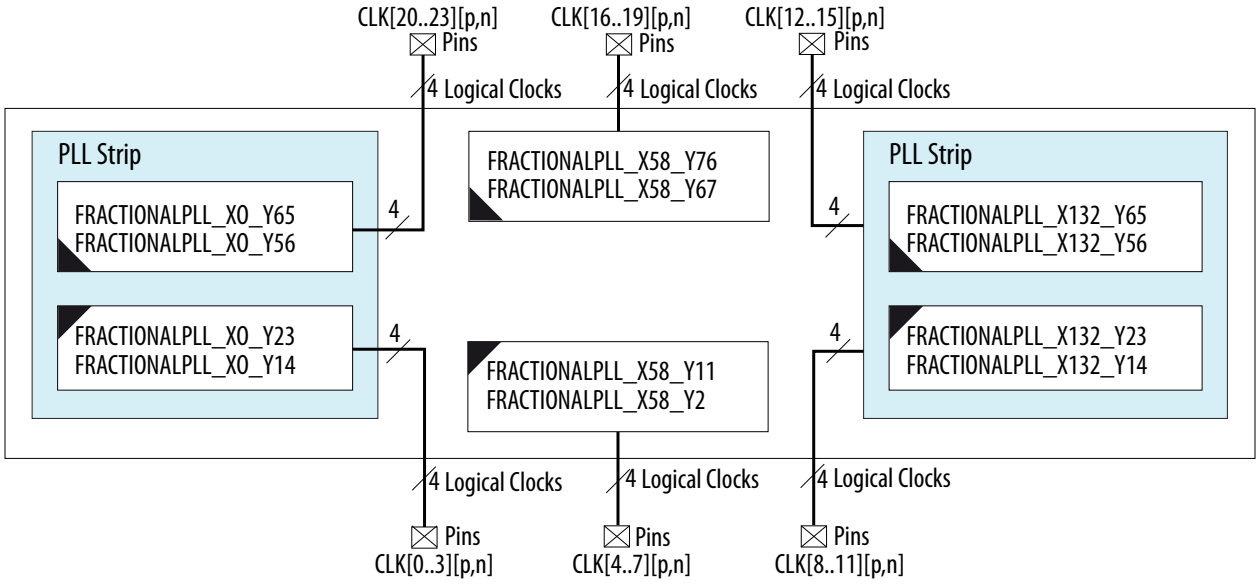


Figure 4-16: PLL Locations for Arria V GX B1 and B3 Devices, and Arria V GT D3 Device

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

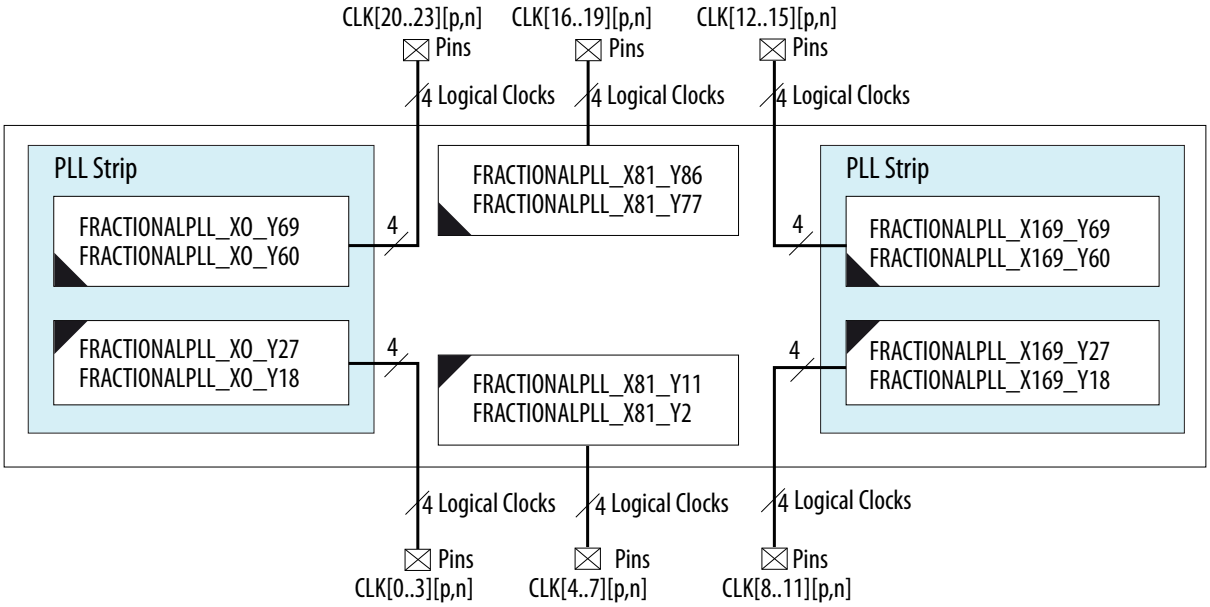


Figure 4-17: PLL Locations for Arria V GX B5 and B7 Devices, and Arria V GT D7 Device

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

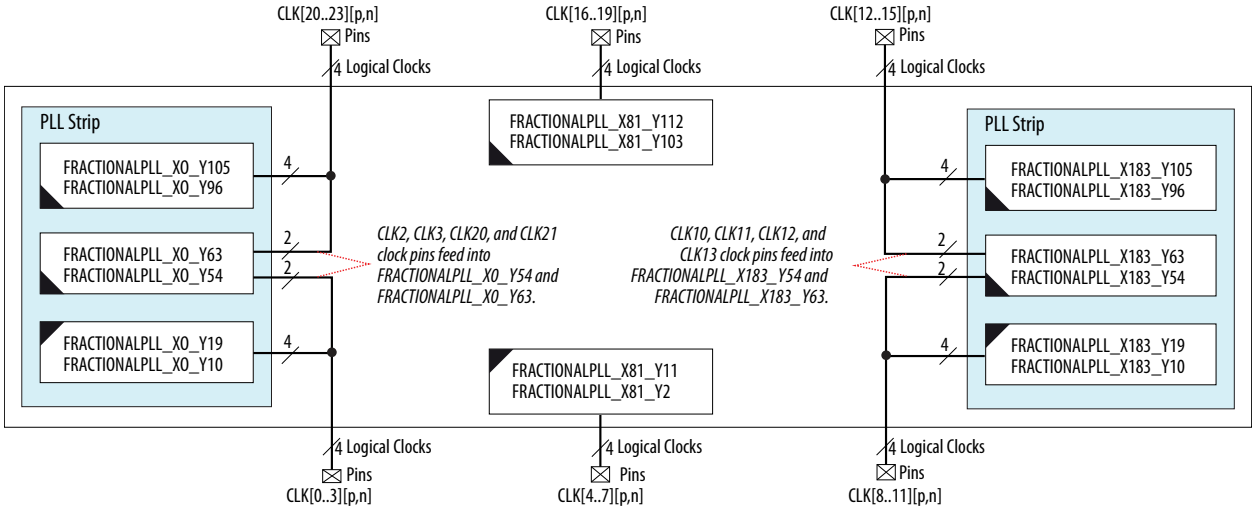


Figure 4-18: PLL Locations for Arria V GZ E1 and E3 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

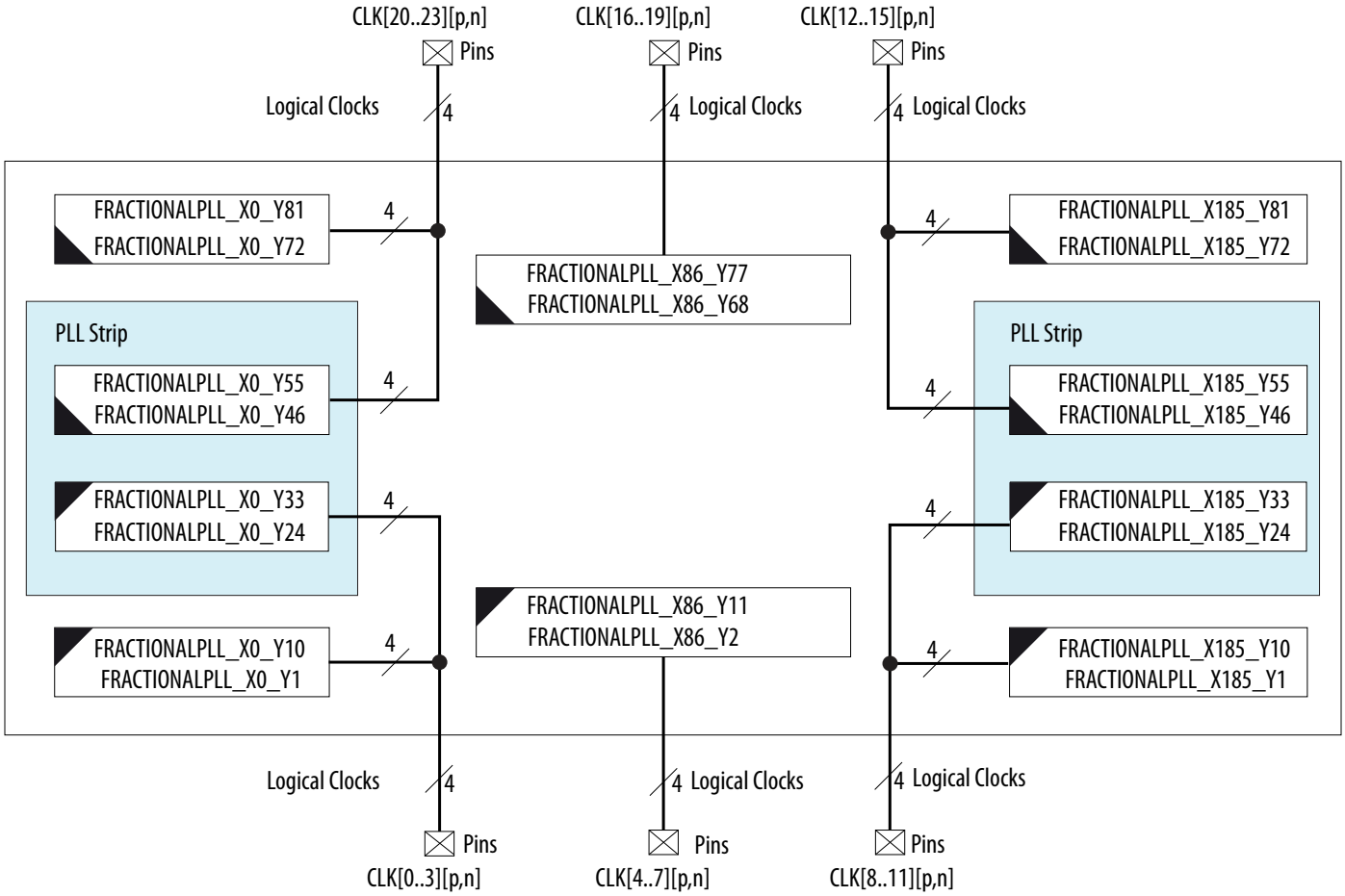


Figure 4-19: PLL Locations for Arria V GZ E5 and E7 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

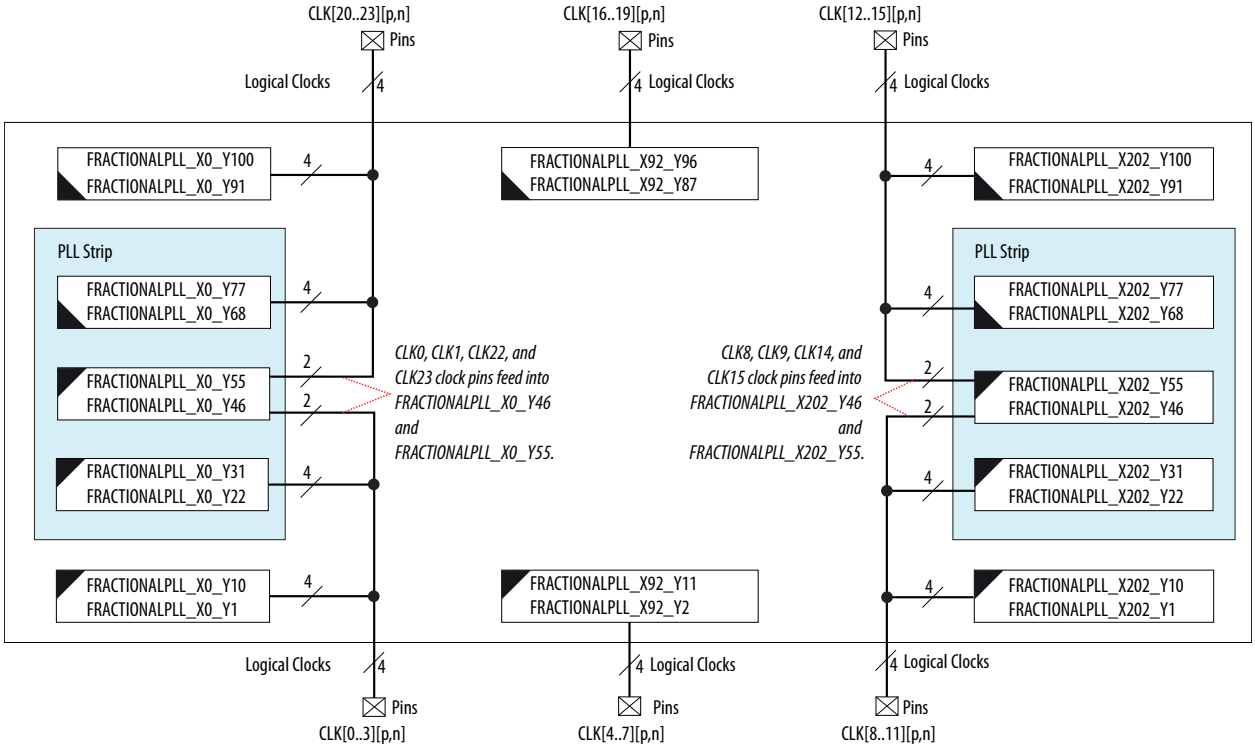
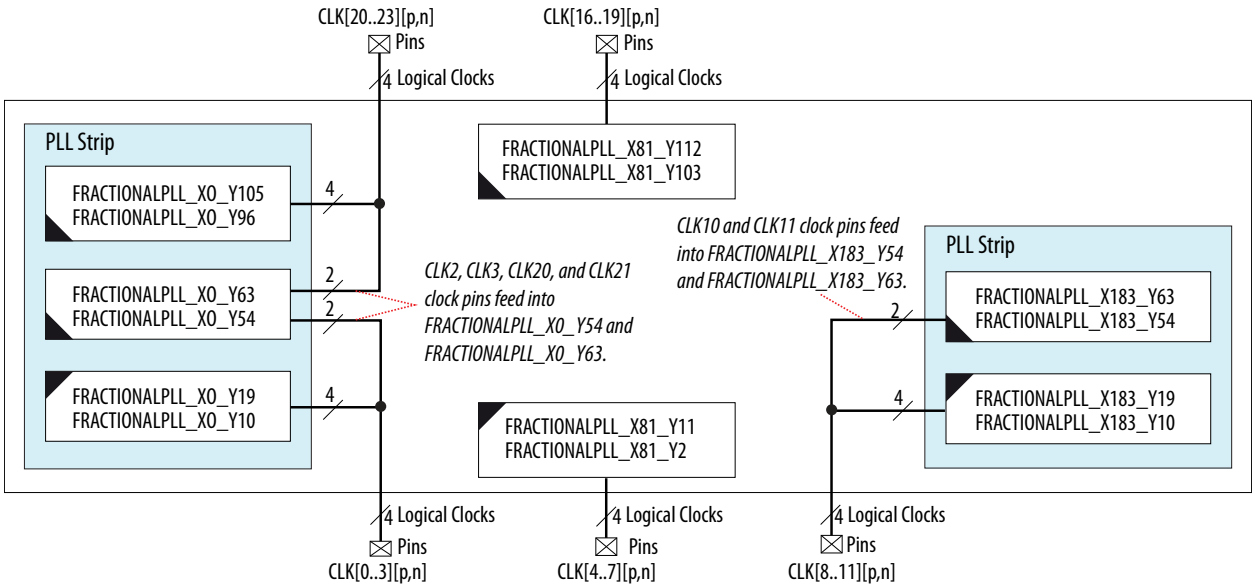


Figure 4-20: PLL Locations for Arria V SX B3 and B5 Devices, and Arria V ST D3 and D5 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Related Information

PLL Migration Guidelines on page 4-22
Provides more information about PLL migration between Arria V GX A5, A7, B1, B3, B5, and B7 devices, and Arria V GT C7, D3, and D7 devices.

PLL Migration Guidelines

If you plan to migrate your design between Arria V GX A5, A7, B1, B3, B5, and B7 devices, and Arria V GT C7, D3, and D7 devices, and your design requires a PLL to drive the HSSI and clock network (GCLK or RCLK), use the PLLs on the left and right side of the device.

Table 4-7: Location of PLLs for PLL Migration

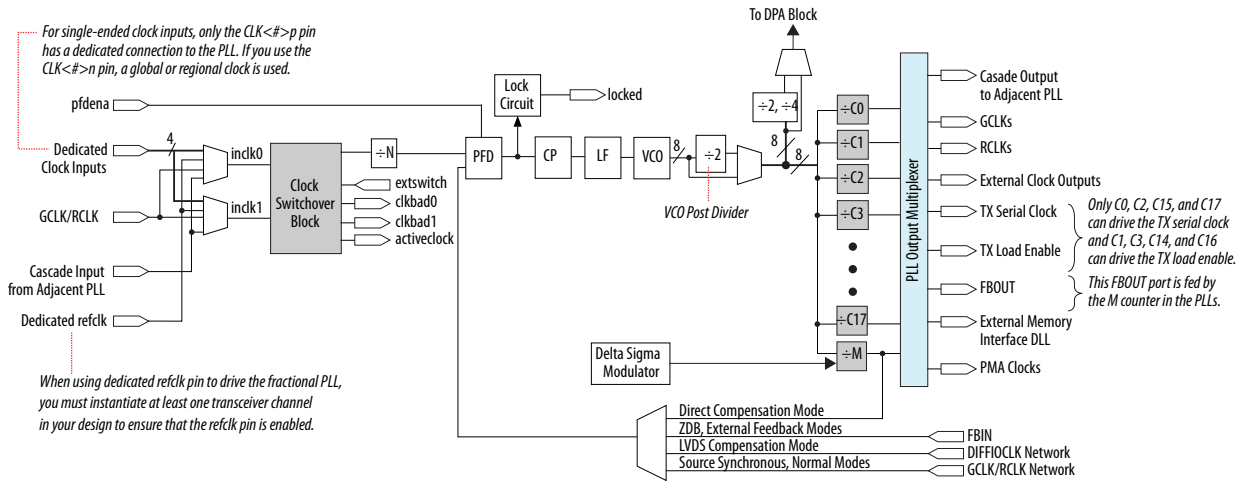
Variant	Member Code	PLL Location	
		Left Side	Right Side
Arria V GX	A5, A7	FRACTIONALPLL_X0_Y14, FRACTIONALPLL_X0_Y23	FRACTIONALPLL_X132_Y14, FRACTIONALPLL_X132_Y23
	B1, B3	FRACTIONALPLL_X0_Y18, FRACTIONALPLL_X0_Y27	FRACTIONALPLL_X169_Y18, FRACTIONALPLL_X169_Y27
	B5, B7	FRACTIONALPLL_X0_Y10, FRACTIONALPLL_X0_Y19	FRACTIONALPLL_X183_Y10, FRACTIONALPLL_X183_Y19
Arria V GT	C7	FRACTIONALPLL_X0_Y14, FRACTIONALPLL_X0_Y23	FRACTIONALPLL_X132_Y14, FRACTIONALPLL_X132_Y23
	D3	FRACTIONALPLL_X0_Y18, FRACTIONALPLL_X0_Y27	FRACTIONALPLL_X169_Y18, FRACTIONALPLL_X169_Y27
	D7	FRACTIONALPLL_X0_Y10, FRACTIONALPLL_X0_Y19	FRACTIONALPLL_X183_Y10, FRACTIONALPLL_X183_Y19

Related Information

PLL Locations in Arria V Devices on page 4-17
Provides more information about CLKIN pin connectivity to the PLLs.

Fractional PLL Architecture

Figure 4-21: Fractional PLL High-Level Block Diagram for Arria V Devices



Fractional PLL Usage

You can configure the fractional PLL to function either in the integer or in the enhanced fractional mode. One fractional PLL can use up to 18 output counters and all external clock outputs. Two adjacent fractional PLLs share the 18 output counters.

Fractional PLLs can be used as follows:

- Reduce the number of required oscillators on the board
- Reduce the clock pins used in the FPGA by synthesizing multiple clock frequencies from a single reference clock source
- Compensate clock network delay
- Zero delay buffering
- Transmit clocking for transceivers

PLL Cascading

Arria V devices support two types of PLL cascading.

PLL-to-PLL Cascading

This cascading mode synthesizes a more precise output frequency than a single PLL in integer mode. Cascading two PLLs in integer mode expands the effective range of the pre-scale counter, `N` and the multiply counter, `M`.

Arria V devices use two types of input clock sources.

- The `adjpllin` input clock source is used for inter-cascading between fracturable fractional PLLs.
- The `cclk` input clock source is used for intra-cascading within fracturable fractional PLLs.

Altera recommends using a low bandwidth setting for the source (upstream) PLL and a high bandwidth setting for destination (downstream) PLL.

Counter-Output-to-Counter-Output Cascading

This cascading mode synthesizes a lower frequency output than a single post-scale counter, `C`. Cascading two `C` counters expands the effective range of `C` counters.

PLL External Clock I/O Pins

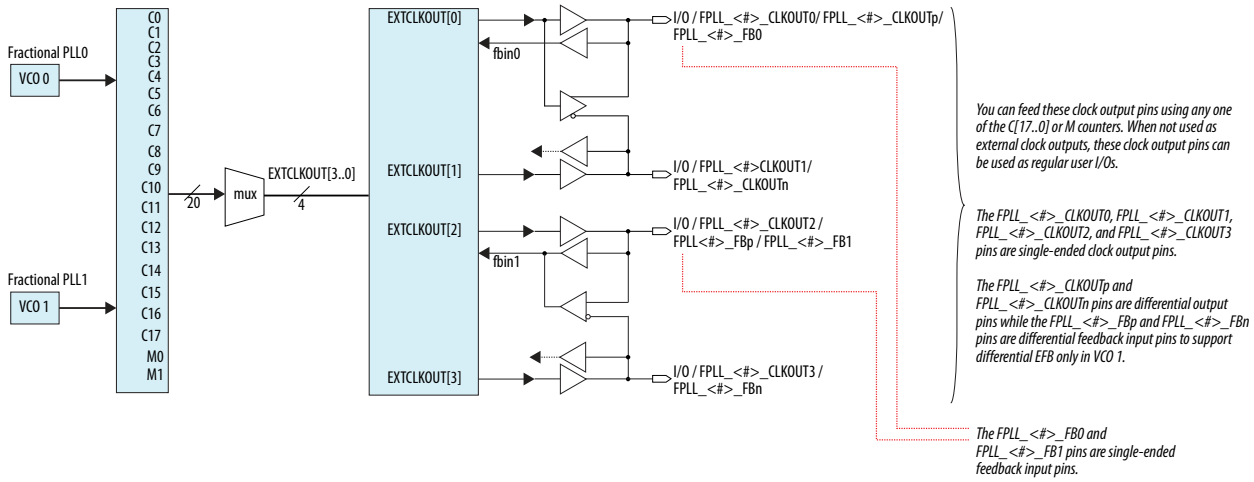
Two adjacent fractional PLLs share four dual-purpose clock I/O pins, organized as one of the following combinations:

- Four single-ended clock outputs
- Two single-ended outputs and one differential clock output
- Four single-ended clock outputs and two single-ended feedback inputs in the I/O driver feedback for zero delay buffer (ZDB) mode support
- Two single-ended clock outputs and two single-ended feedback inputs for single-ended external feedback (EFB) mode support
- One differential clock output and one differential feedback input for differential EFB support (only one of the two adjacent fractional PLLs can support differential EFB at one time while the other fractional PLL can be used for general-purpose clocking)

Note: The middle fractional PLLs on the left and right sides of Arria V GX B5 and B7 devices, and Arria V GT D7 device do not support external clock outputs.

The following figure shows that any of the output counters (`C[0..17]`) or the `M` counter on the PLLs can feed the dedicated external clock outputs. Therefore, one counter or frequency can drive all output pins available from a given PLL.

Figure 4-22: Dual-Purpose Clock I/O Pins Associated with PLL for Arria V Devices



Each pin of a single-ended output pair can be either in-phase or 180° out-of-phase. To implement the 180° out-of-phase pin in a pin pair, the Intel Quartus Prime software places a NOT gate in the design into the IOE.

The clock output pin pairs support the following I/O standards:

- Same I/O standard for the pin pairs
- LVDS
- Differential high-speed transceiver logic (HSTL)
- Differential SSTL

Arria V PLLs can drive out to any regular I/O pin through the GCLK or RCLK network. You can also use the external clock output pins as user I/O pins if you do not require external PLL clocking.

Related Information

- **I/O Standards Support in Arria V Devices** on page 5-5
  Provides more information about I/O standards supported by the PLL clock input and output pins.
- **Zero-Delay Buffer Mode** on page 4-28
- **External Feedback Mode** on page 4-30

PLL Control Signals

You can use the `areset` signal to control PLL operation and resynchronization, and use the `locked` signal to observe the status of the PLL.

areset

The `areset` signal is the reset or resynchronization input for each PLL. The device input pins or internal logic can drive these input signals.

When `areset` is driven high, the PLL counters reset, clearing the PLL output and placing the PLL out-of-lock. The VCO is then set back to its nominal setting. When `areset` is driven low again, the PLL resynchronizes to its input as it re-locks.

You must assert the `areset` signal every time the PLL loses lock to guarantee the correct phase relationship between the PLL input and output clocks. You can set up the PLL to automatically reset (self-reset) after a loss-of-lock condition using the Intel Quartus Prime IP Catalog.

You must include the `areset` signal if either of the following conditions is true:

- PLL reconfiguration or clock switchover is enabled in the design
- Phase relationships between the PLL input and output clocks must be maintained after a loss-of-lock condition

Note: If the input clock to the PLL is not toggling or is unstable after power up, assert the `areset` signal after the input clock is stable and within specifications.

locked

The `locked` signal output of the PLL indicates the following conditions:

- The PLL has locked onto the reference clock.
- The PLL clock outputs are operating at the desired phase and frequency set in the IP Catalog.

The lock detection circuit provides a signal to the core logic. The signal indicates when the feedback clock has locked onto the reference clock both in phase and frequency.

Clock Feedback Modes

This section describes the following clock feedback modes:

- Source synchronous
- LVDS compensation
- Direct
- Normal compensation
- ZDB
- EFB

Each mode allows clock multiplication and division, phase shifting, and programmable duty cycle.

The input and output delays are fully compensated by a PLL only when using the dedicated clock input pins associated with a given PLL as the clock source.

The input and output delays may not be fully compensated in the Intel Quartus Prime software for the following conditions:

- When a GCLK or RCLK network drives the PLL
- When the PLL is driven by a dedicated clock pin that is not associated with the PLL

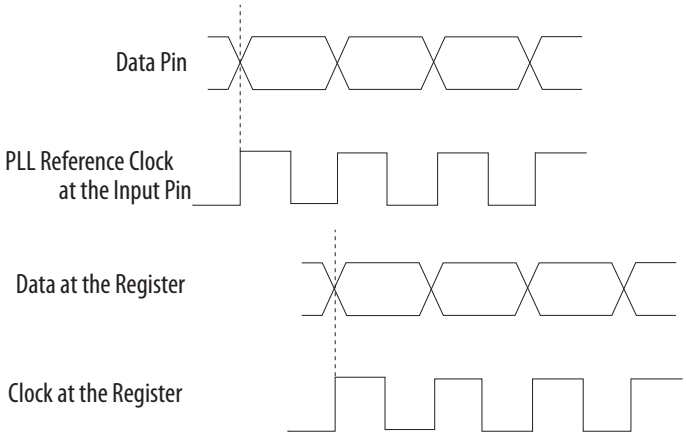
For example, when you configure a PLL in ZDB mode, the PLL input is driven by an associated dedicated clock input pin. In this configuration, a fully compensated clock path results in zero delay between the clock input and one of the clock outputs from the PLL. However, if the PLL input is fed by a non-dedicated input (using the GCLK network), the output clock may not be perfectly aligned with the input clock.

Source Synchronous Mode

If the data and clock arrive at the same time on the input pins, the same phase relationship is maintained at the clock and data ports of any IOE input register. Data and clock signals at the IOE experience similar buffer delays as long as you use the same I/O standard.

Altera recommends source synchronous mode for source synchronous data transfers.

Figure 4-23: Example of Phase Relationship Between Clock and Data in Source Synchronous Mode



The source synchronous mode compensates for the delay of the clock network used and any difference in the delay between the following two paths:

- Data pin to the IOE register input
- Clock input pin to the PLL phase frequency detector (PFD) input

The Arria V PLL can compensate multiple pad-to-input-register paths, such as a data bus when it is set to use source synchronous compensation mode.

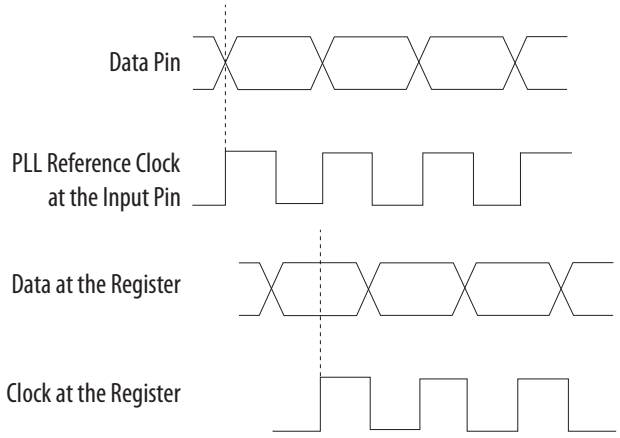
LVDS Compensation Mode

The purpose of LVDS compensation mode is to maintain the same data and clock timing relationship seen at the pins of the internal serializer/deserializer (SERDES) capture register, except that the clock is inverted (180° phase shift). Thus, LVDS compensation mode ideally compensates for the delay of the LVDS clock network, including the difference in delay between the following two paths:

- Data pin-to-SERDES capture register
- Clock input pin-to-SERDES capture register

The output counter must provide the 180° phase shift.

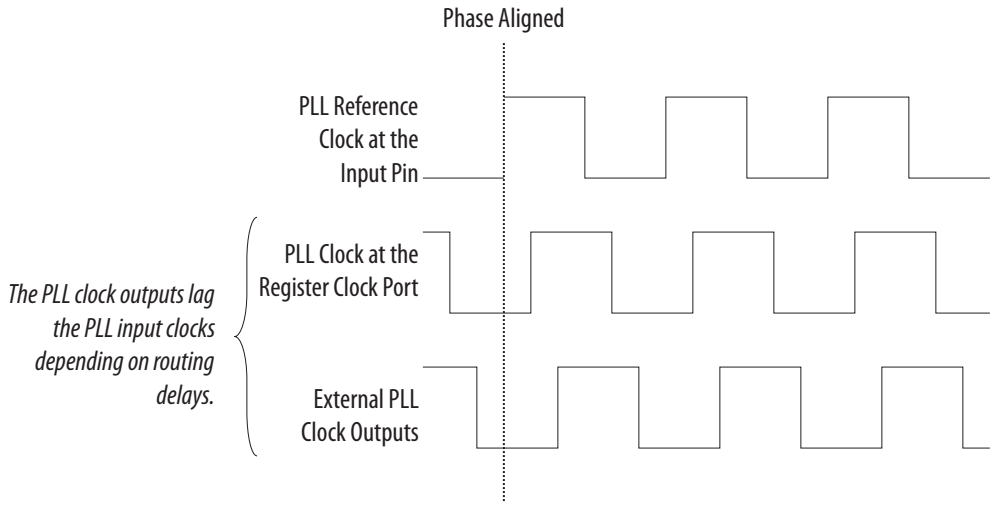
Figure 4-24: Example of Phase Relationship Between the Clock and Data in LVDS Compensation Mode



Direct Mode

In direct mode, the PLL does not compensate for any clock networks. This mode provides better jitter performance because the clock feedback into the PFD passes through less circuitry. Both the PLL internal- and external-clock outputs are phase-shifted with respect to the PLL clock input.

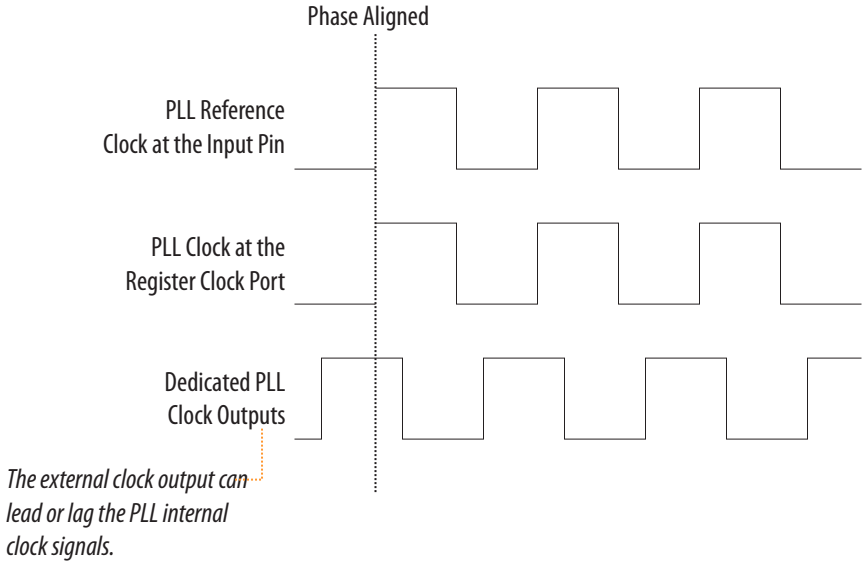
Figure 4-25: Example of Phase Relationship Between the PLL Clocks in Direct Mode



Normal Compensation Mode

An internal clock in normal compensation mode is phase-aligned to the input clock pin. The external clock output pin has a phase delay relative to the clock input pin if connected in this mode. The Intel Quartus Prime Timing Analyzer reports any phase difference between the two. In normal compensation mode, the delay introduced by the GCLK or RCLK network is fully compensated.

Figure 4-26: Example of Phase Relationship Between the PLL Clocks in Normal Compensation Mode



Zero-Delay Buffer Mode

In ZDB mode, the external clock output pin is phase-aligned with the clock input pin for zero delay through the device. This mode is supported on all Arria V PLLs.

When using this mode, you must use the same I/O standard on the input clocks and clock outputs to guarantee clock alignment at the input and output pins. You cannot use differential I/O standards on the PLL clock input or output pins.

To ensure phase alignment between the `clk` pin and the external clock output (`CLKOUT`) pin in ZDB mode, instantiate a bidirectional I/O pin in the design. The bidirectional I/O pin serves as the feedback path connecting the `fbout` and `fbin` ports of the PLL. The bidirectional I/O pin must always be assigned a single-ended I/O standard. The PLL uses this bidirectional I/O pin to mimic and compensate for the output delay from the clock output port of the PLL to the external clock output pin.

Note: To avoid signal reflection when using ZDB mode, do not place board traces on the bidirectional I/O pin.

Figure 4-27: ZDB Mode in Arria V PLLs

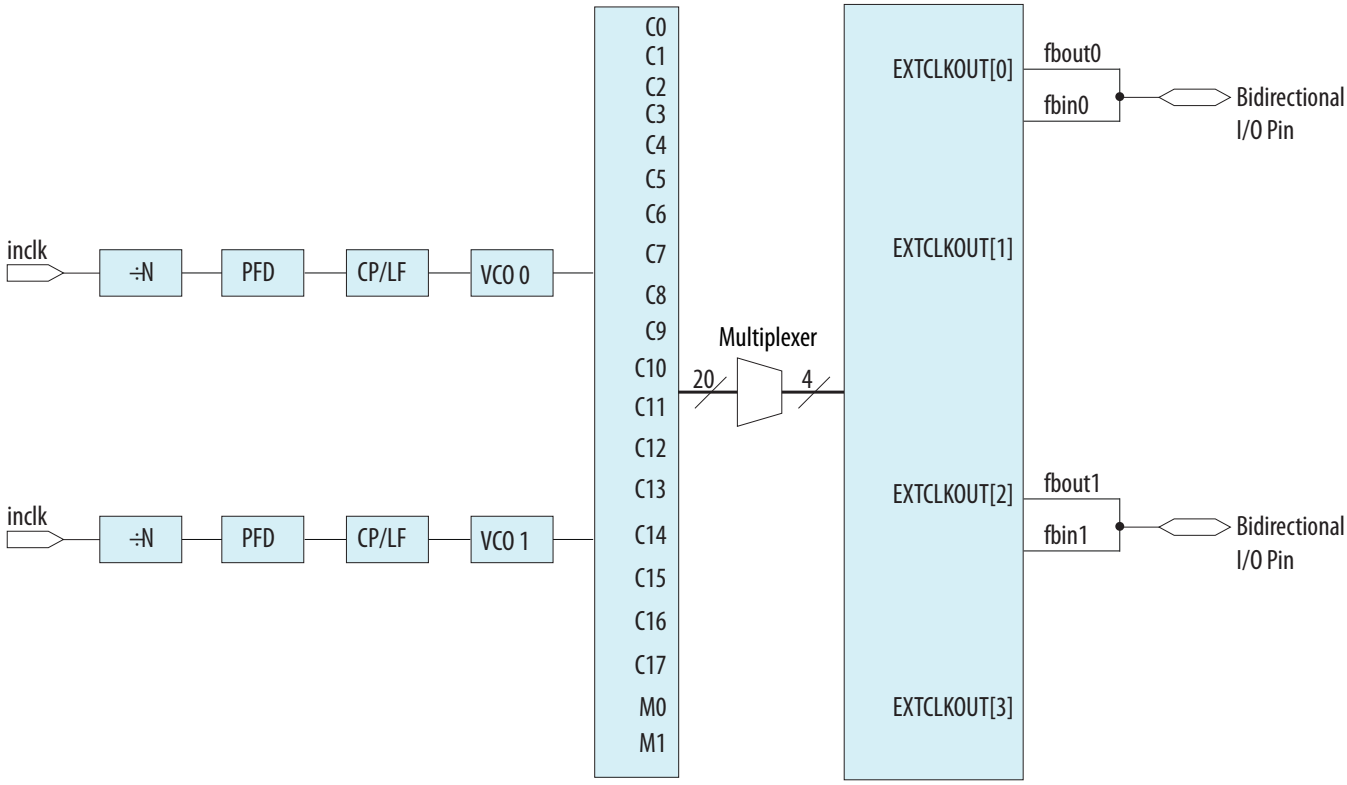
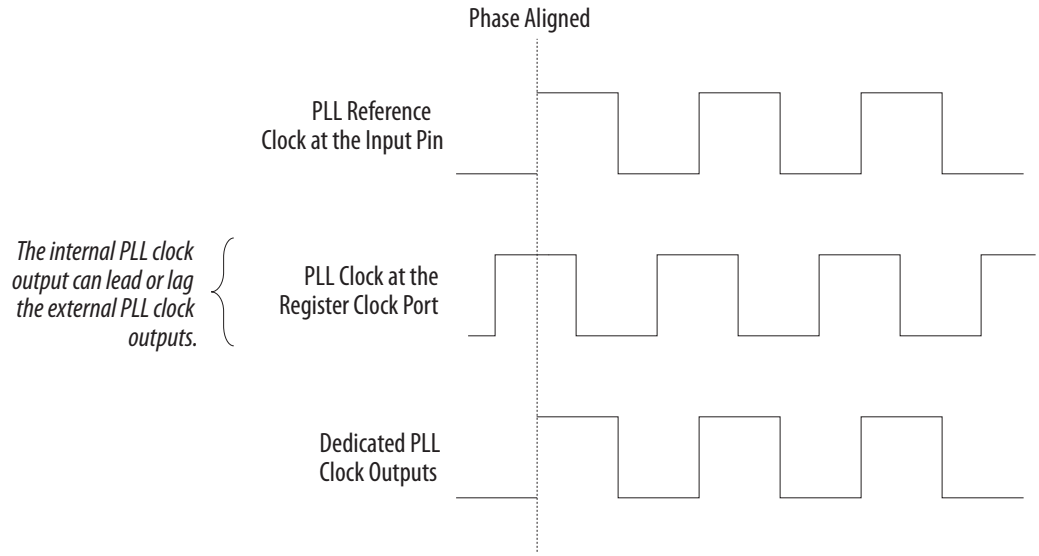


Figure 4-28: Example of Phase Relationship Between the PLL Clocks in ZDB Mode



Related Information

PLL External Clock I/O Pins on page 4-24
Provides more information about PLL clock outputs.

External Feedback Mode

In EFB mode, the output of the `M` counter (`fbout`) feeds back to the PLL `fbin` input (using a trace on the board) and becomes part of the feedback loop.

One of the dual-purpose external clock outputs becomes the `fbin` input pin in this mode. The external feedback input pin, `fbin` is phase-aligned with the clock input pin. Aligning these clocks allows you to remove clock delay and skew between devices.

When using EFB mode, you must use the same I/O standard on the input clock, feedback input, and clock outputs.

Figure 4-29: EFB Mode in Arria V Devices

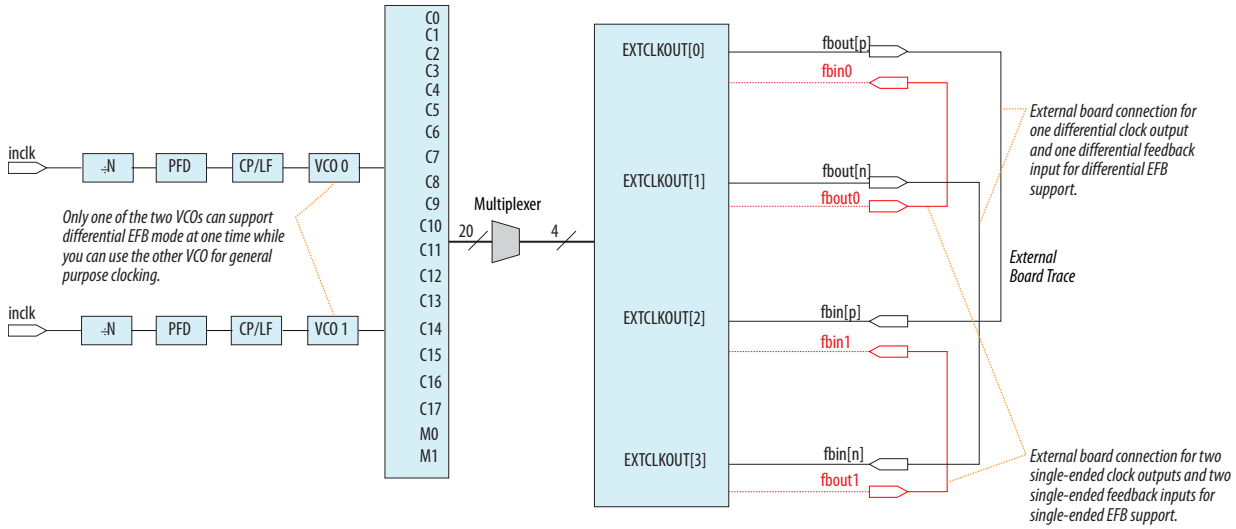
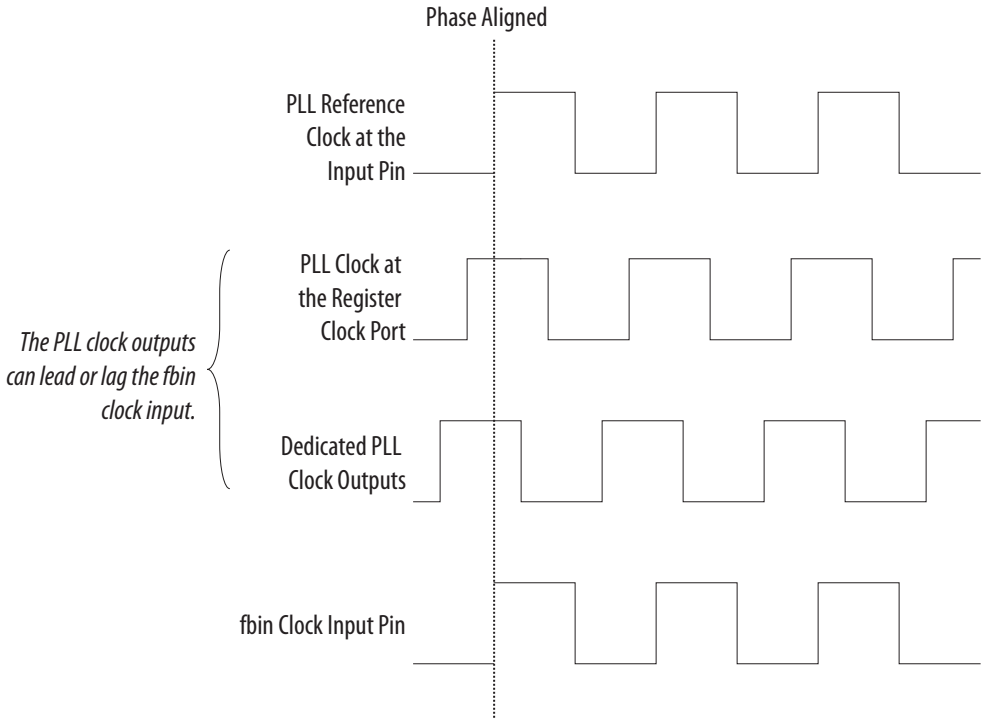


Figure 4-30: Example of Phase Relationship Between the PLL Clocks in EFB Mode



Related Information

PLL External Clock I/O Pins on page 4-24
Provides more information about PLL clock outputs.

Clock Multiplication and Division

Each Arria V PLL provides clock synthesis for PLL output ports using the M/(N × C) scaling factors. The input clock is divided by a pre-scale factor, N, and is then multiplied by the M feedback factor. The control loop drives the VCO to match $f_{in} \times (M/N)$.

The Intel Quartus Prime software automatically chooses the appropriate scaling factors according to the input frequency, multiplication, and division values entered into the ALTERA_PLL IP core.

VCO Post Divider

A VCO post divider is inserted after the VCO. When you enable the VCO post divider, the VCO post divider divides the VCO frequency by two. When the VCO post divider is bypassed, the VCO frequency goes to the output port without being divided by two.

Post-Scale Counter, C

Each output port has a unique post-scale counter, C, that divides down the output from the VCO post divider. For multiple PLL outputs with different frequencies, the VCO is set to the least common multiple of the output frequencies that meets its frequency specifications. For example, if the output frequencies required from one PLL are 33 and 66 MHz, the Intel Quartus Prime software sets the VCO to 660 MHz (the least common multiple of 33 and 66 MHz within the VCO range). Then the post-scale counters, C, scale down the VCO frequency for each output port.

Pre-Scale Counter, N and Multiply Counter, M

Each PLL has one pre-scale counter, N, and one multiply counter, M, with a range of 1 to 512 for both M and N. The N counter does not use duty-cycle control because the only purpose of this counter is to calculate frequency division. The post-scale counters have a 50% duty cycle setting. The high- and low-count values for each counter range from 1 to 256. The sum of the high- and low-count values chosen for a design selects the divide value for a given counter.

Delta-Sigma Modulator

The delta-sigma modulator (DSM) is used together with the M multiply counter to enable the PLL to operate in fractional mode. The DSM dynamically changes the M counter divide value on a cycle to cycle basis. The different M counter values allow the "average" M counter value to be a non-integer.

Fractional Mode

In fractional mode, the M counter divide value equals to the sum of the "clock high" count, "clock low" count, and the fractional value. The fractional value is equal to K/2^*X*, where K is an integer between 0 and (2^*X* – 1), and *X* = 8, 16, 24, or 32.

Integer Mode

For PLL operating in integer mode, M is an integer value and DSM is disabled.

Related Information

Altera Phase-Locked Loop (Altera PLL) IP Core User Guide
Provides more information about PLL software support in the Quartus Prime software.

Programmable Phase Shift

The programmable phase shift feature allows the PLLs to generate output clocks with a fixed phase offset.

The VCO frequency of the PLL determines the precision of the phase shift. The minimum phase shift increment is 1/8 of the VCO period. For example, if a PLL operates with a VCO frequency of 1000 MHz, phase shift steps of 125 ps are possible.

The Intel Quartus Prime software automatically adjusts the VCO frequency according to the user-specified phase shift values entered into the IP core.

Programmable Duty Cycle

The programmable duty cycle allows PLLs to generate clock outputs with a variable duty cycle. This feature is supported on the PLL post-scale counters.

The duty-cycle setting is achieved by a low and high time-count setting for the post-scale counters. To determine the duty cycle choices, the Intel Quartus Prime software uses the frequency input and the required multiply or divide rate.

The post-scale counter value determines the precision of the duty cycle. The precision is defined as 50% divided by the post-scale counter value. For example, if the `C0` counter is 10, steps of 5% are possible for duty-cycle choices from 5% to 90%. If the PLL is in external feedback mode, set the duty cycle for the counter driving the `fbin` pin to 50%.

Combining the programmable duty cycle with programmable phase shift allows the generation of precise non-overlapping clocks.

Clock Switchover

The clock switchover feature allows the PLL to switch between two reference input clocks. Use this feature for clock redundancy or for a dual-clock domain application where a system turns on the redundant clock if the previous clock stops running. The design can perform clock switchover automatically when the clock is no longer toggling or based on a user control signal, `extswitch`.

The following clock switchover modes are supported in Arria V PLLs:

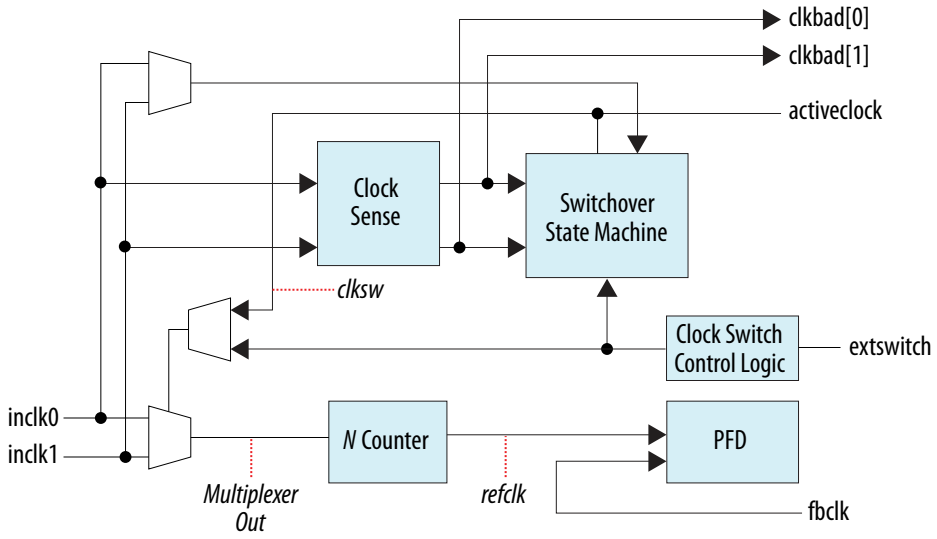
- Automatic switchover—The clock sense circuit monitors the current reference clock. If the current reference clock stops toggling, the reference clock automatically switches to `inclk0` or `inclk1` clock.
- Manual clock switchover—Clock switchover is controlled using the `extswitch` signal. When the `extswitch` signal goes from logic low to logic high, and stays high for at least three clock cycles, the reference clock to the PLL is switched from `inclk0` to `inclk1`, or vice-versa.
- Automatic switchover with manual override—This mode combines automatic switchover and manual clock switchover. When the `extswitch` signal goes high, it overrides the automatic clock switchover function.

Automatic Switchover

Arria V PLLs support a fully configurable clock switchover capability.

Figure 4-31: Automatic Clock Switchover Circuit Block Diagram

This figure shows a block diagram of the automatic switchover circuit built into the PLL.



When the current reference clock is not present, the clock sense block automatically switches to the backup clock for PLL reference. You can select a clock source as the backup clock by connecting it to the `inclk1` port of the PLL in your design.

The clock switchover circuit sends out three status signals—`clkbad[0]`, `clkbad[1]`, and `activeclock`—from the PLL to implement a custom switchover circuit in the logic array.

In automatic switchover mode, the `clkbad[0]` and `clkbad[1]` signals indicate the status of the two clock inputs. When they are asserted, the clock sense block detects that the corresponding clock input has stopped toggling. These two signals are not valid if the frequency difference between `inclk0` and `inclk1` is greater than 20%.

The `activeclock` signal indicates which of the two clock inputs (`inclk0` or `inclk1`) is being selected as the reference clock to the PLL. When the frequency difference between the two clock inputs is more than 20%, the `activeclock` signal is the only valid status signal.

Note: Glitches in the input clock may cause the frequency difference between the input clocks to be more than 20%.

Use the switchover circuitry to automatically switch between `inclk0` and `inclk1` when the current reference clock to the PLL stops toggling. You can switch back and forth between `inclk0` and `inclk1` any number of times when one of the two clocks fails and the other clock is available.

For example, in applications that require a redundant clock with the same frequency as the reference clock, the switchover state machine generates a signal (`clksw`) that controls the multiplexer select input. In this case, `inclk1` becomes the reference clock for the PLL.

When using automatic clock switchover mode, the following requirements must be satisfied:

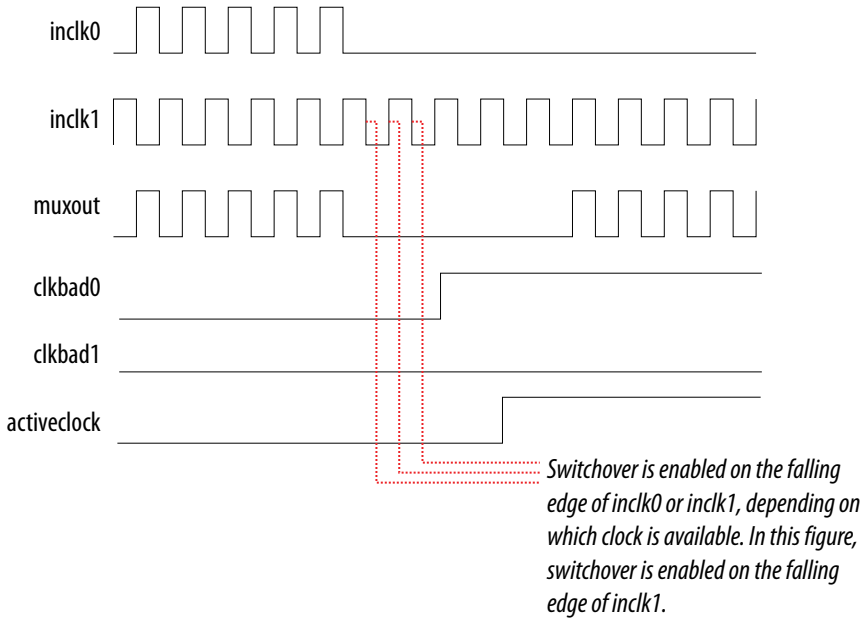
- Both clock inputs must be running when the FPGA is configured.
- The period of the two clock inputs can differ by no more than 20%.

If the current clock input stops toggling while the other clock is also not toggling, switchover is not initiated and the `clkbad[0..1]` signals are not valid. If both clock inputs are not the same frequency, but their period difference is within 20%, the clock sense block detects when a clock stops toggling. However, the PLL may lose lock after the switchover is completed and needs time to relock.

Note: Altera recommends resetting the PLL using the `areset` signal to maintain the phase relationships between the PLL input and output clocks when using clock switchover.

Figure 4-32: Automatic Switchover After Loss of Clock Detection

This figure shows an example waveform of the switchover feature in automatic switchover mode. In this example, the `inclk0` signal is stuck low. After the `inclk0` signal is stuck at low for approximately two clock cycles, the clock sense circuitry drives the `clkbad[0]` signal high. Since the reference clock signal is not toggling, the switchover state machine controls the multiplexer through the `extswitch` signal to switch to the backup clock, `inclk1`.



Automatic Switchover with Manual Override

In automatic switchover with manual override mode, you can use the `extswitch` signal for user- or system-controlled switch conditions. You can use this mode for same-frequency switchover, or to switch between inputs of different frequencies.

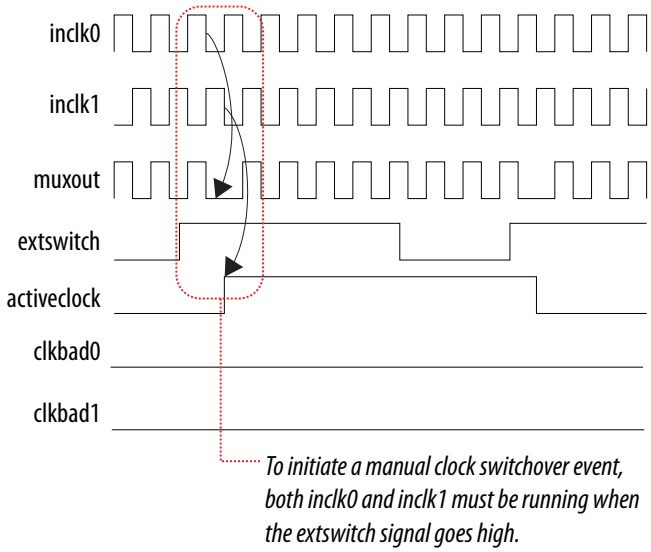
For example, if `inclk0` is 66 MHz and `inclk1` is 200 MHz, you must control switchover using the `extswitch` signal. The automatic clock-sense circuitry cannot monitor clock input (`inclk0` and `inclk1`) frequencies with a frequency difference of more than 100% (2×).

This feature is useful when the clock sources originate from multiple cards on the backplane, requiring a system-controlled switchover between the frequencies of operation.

You must choose the backup clock frequency and set the `M`, `N`, `C`, and `K` counters so that the VCO operates within the recommended operating frequency range. The ALTERA_PLL IP Catalog notifies you if a given combination of `inclk0` and `inclk1` frequencies cannot meet this requirement.

Figure 4-33: Clock Switchover Using the `extswitch` (Manual) Control

This figure shows a clock switchover waveform controlled by the `extswitch` signal. In this case, both clock sources are functional and `inclk0` is selected as the reference clock; the `extswitch` signal goes high, which starts the switchover sequence. On the falling edge of `inclk0`, the counter's reference clock, `muxout`, is gated off to prevent clock glitching. On the falling edge of `inclk1`, the reference clock multiplexer switches from `inclk0` to `inclk1` as the PLL reference. The `activeclock` signal changes to indicate the clock which is currently feeding the PLL.



In automatic override with manual switchover mode, the `activeclock` signal mirrors the `extswitch` signal. Since both clocks are still functional during the manual switch, neither `clkbad` signal goes high. Because the switchover circuit is positive-edge sensitive, the falling edge of the `extswitch` signal does not cause the circuit to switch back from `inclk1` to `inclk0`. When the `extswitch` signal goes high again, the process repeats.

The `extswitch` signal and automatic switch work only if the clock being switched to is available. If the clock is not available, the state machine waits until the clock is available.

Related Information

Altera Phase-Locked Loop (Altera PLL) IP Core User Guide
Provides more information about PLL software support in the Quartus Prime software.

Manual Clock Switchover

In manual clock switchover mode, the `extswitch` signal controls whether `inclk0` or `inclk1` is selected as the input clock to the PLL. By default, `inclk0` is selected.

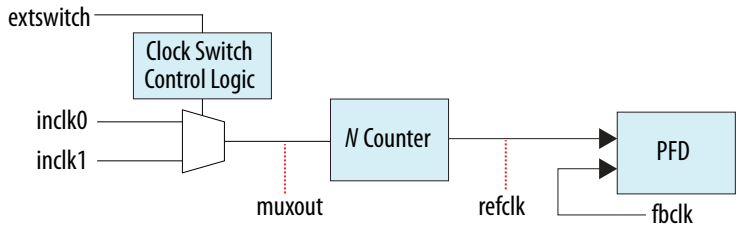
A clock switchover event is initiated when the `extswitch` signal transitions from logic low to logic high, and being held high for at least three `inclk` cycles.

You must bring the `extswitch` signal back low again for PLL to re-gain lock. If you do not require another switchover event, you can leave the `extswitch` signal in a logic low state.

Pulsing the `extswitch` signal high for at least three `inclk` cycles performs another switchover event.

If `inclk0` and `inclk1` are different frequencies and are always running, the `extswitch` signal minimum high time must be greater than or equal to three of the slower frequency `inclk0` and `inclk1` cycles.

Figure 4-34: Manual Clock Switchover Circuitry in Arria V PLLs



You can delay the clock switchover action by specifying the switchover delay in the ALTERA_PLL IP core. When you specify the switchover delay, the `extswitch` signal must be held high for at least three `inclk` cycles plus the number of the delay cycles that has been specified to initiate a clock switchover.

Related Information

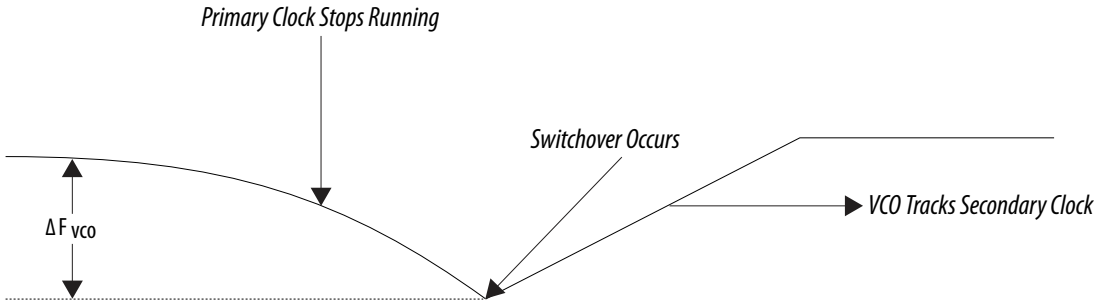
Altera Phase-Locked Loop (Altera PLL) IP Core User Guide
Provides more information about PLL software support in the Quartus Prime software.

Guidelines

When implementing clock switchover in Arria V PLLs, use the following guidelines:

- Automatic clock switchover requires that the `inclk0` and `inclk1` frequencies be within 20% of each other. Failing to meet this requirement causes the `clkbad[0]` and `clkbad[1]` signals to not function properly.
- When using manual clock switchover, the difference between `inclk0` and `inclk1` can be more than 100% (2×). However, differences in frequency, phase, or both, of the two clock sources will likely cause the PLL to lose lock. Resetting the PLL ensures that you maintain the correct phase relationships between the input and output clocks.
- Both `inclk0` and `inclk1` must be running when the `extswitch` signal goes high to initiate the manual clock switchover event. Failing to meet this requirement causes the clock switchover to not function properly.
- Applications that require a clock switchover feature and a small frequency drift must use a low-bandwidth PLL. When referencing input clock changes, the low-bandwidth PLL reacts more slowly than a high-bandwidth PLL. When switchover happens, a low-bandwidth PLL propagates the stopping of the clock to the output more slowly than a high-bandwidth PLL. However, be aware that the low-bandwidth PLL also increases lock time.
- After a switchover occurs, there may be a finite resynchronization period for the PLL to lock onto a new clock. The time it takes for the PLL to relock depends on the PLL configuration.
- The phase relationship between the input clock to the PLL and the output clock from the PLL is important in your design. Assert `areset` for at least 10 ns after performing a clock switchover. Wait for the locked signal to go high and be stable before re-enabling the output clocks from the PLL.
- The VCO frequency gradually decreases when the current clock is lost and then increases as the VCO locks on to the backup clock, as shown in the following figure.

Figure 4-35: VCO Switchover Operating Frequency



PLL Reconfiguration and Dynamic Phase Shift

For more information about PLL reconfiguration and dynamic phase shifting, refer to AN661.

Related Information

AN 661: Implementing Fractional PLL Reconfiguration with Altera PLL and Altera PLL Reconfig IP Cores

Clock Networks and PLLs in Arria V Devices Revision History

Document Version	Changes
2019.04.26	• Corrected the signal name from `clkswitch` to `extswitch`. • Updated the description for the automatic switchover with manual override mode in the *Clock Switchover* section. • Updated the description about the `extswitch` signal in the *Manual Clock Switchover* section.

Date	Version	Changes
December 2016	2016.12.09	Added a note to dedicated refclk pin in Fractional PLL High-Level Block Diagram.
December 2015	2015.12.21	Changed instances of *Quartus II* to *Quartus Prime*.

Date	Version	Changes
January 2015	2015.01.23	• Added a note pointing to FRACTIONALPLL_X183_Y63 and FRACTIONALPLL_X183_Y54 in PLL locations diagram for Arria V SX B3 and B5 Devices, and Arria V ST D3 and D5 Devices. Note: CLK10 and CLK11 clock pins feed into FRACTIONALPLL_X183_Y63 and FRACTIONALPLL_X183_Y54. • PLL coordinates for Arria V GT C3 and C7 devices are finalized. Removed the notes that state the PLL coordinates will be finalized in a future release of the Quartus II software from the PLL locations diagrams.
January 2014	2014.01.10	• Removed Preliminary tags for clock resources, clock input pin connections to GCLK and RCLK networks, and PLL features tables. • Updated clock resources table. • Added availability for RCLK[46..51] and RCLK[52..57] pins in RCLK networks diagram. • Added notes to dedicated clock input pin connectivity to GCLK and RCLK tables. • Added label for PLL strip in PLL locations diagrams. • Added descriptions for PLLs located in a strip. • Added PLL locations diagram for Arria V SX B3 and B5 devices, and Arria V ST D3 and D5 devices. • Added information on PLL migration guidelines. • Updated VCO post-scale counter, K, to VCO post divider. • Added information on PLL cascading. • Added information on programmable phase shift. • Updated automatic clock switchover mode requirement.

Date	Version	Changes
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated RCLK and PCLK clock sources per device quadrant. • Added link to Arria V GZ Device Family Pin Connection Guidelines. • Updated RCLK and PCLK clock sources in hierarchical clock networks in each spine clock per quadrant diagram. • Added PCLK networks in clock network sources section. • Updated dedicated clock input pins in clock network sources section. • Updated information on clock power down. • Added information on `C` output counters for PLLs. • Added power down mode in PLL features table. • Added PLL physical counters information and diagram. • Marked PLL physical counters orientation in PLL locations diagrams. • Updated the fractional PLL architecture diagram to add dedicated `refclk` input port and connections. • Removed information on `pfdena` PLL control signal. • Updated the scaling factors for PLL output ports. • Updated the fractional value for PLL in fractional mode. • Moved all links to the Related Information section of respective topics for easy reference. • Reorganized content.
November 2012	2012.11.19	• Added note to indicate that the figures shown are the top view of the silicon die. • Updated clock resources for Arria V GZ devices. • Added RCLK networks diagram for Arria V GZ devices. • Restructured tables for clock input pin connectivity to the GCLK and RCLK networks. • Added table for clock input pin connectivity to the RCLK networks for Arria V GZ devices. • Updated PCLK control block information. • Added PLL locations diagrams for Arria V GZ E1, E3, E5, and E7 devices. • Removed information on PLL Compensation assignment in the Quartus II software. • Updated the fractional value for PLL in fractional mode. • Reorganized content and updated template.

Date	Version	Changes
June 2012	2.0	• Restructured chapter. • Updated Figure 4–4, Figure 4–6, Figure 4–7, Figure 4–11, Figure 4–12, Figure 4–13, Figure 4–14, Figure 4–15, Figure 4–16, Figure 4–18, and Figure 4–19. • Updated Table 4–1, Table 4–2, Table 4–3, Table 4–4, and Table 4–5. • Added "Clock Regions", "Clock Network Sources", "Clock Output Connections", "Clock Enable Signals", "PLL Control Signals", "Clock Multiplication and Division", "Programmable Duty Cycle", "Clock Switchover", and "PLL Reconfiguration and Dynamic Phase Shift".
November 2011	1.1	Restructured chapter.
May 2011	1.0	Initial release.

I/O Features in Arria V Devices

5

2020.04.13

AV-52005

Subscribe Send Feedback

This chapter provides details about the features of the Arria V I/O elements (IOEs) and how the IOEs work in compliance with current and emerging I/O standards and requirements.

The Arria V I/Os support the following features:

- Single-ended, non-voltage-referenced, and voltage-referenced I/O standards
- Low-voltage differential signaling (LVDS), RSDS, mini-LVDS, HSTL, HSUL, and SSTL I/O standards
- Serializer/deserializer (SERDES)
- Programmable output current strength
- Programmable slew rate
- Programmable bus-hold
- Programmable pull-up resistor
- Programmable pre-emphasis
- Programmable I/O delay
- Programmable voltage output differential (V_{OD})
- Open-drain output
- On-chip series termination (R_S OCT) with and without calibration
- On-chip parallel termination (R_T OCT)
- On-chip differential termination (R_D OCT)

Note: The information in this chapter is applicable to all Arria V variants, unless noted otherwise.

Related Information

Arria V Device Handbook: Known Issues
Lists the planned updates to the *Arria V Device Handbook* chapters.

I/O Resources Per Package for Arria V Devices

The following package plan tables for the different Arria V variants list the maximum I/O resources available for each package.

Table 5-1: Package Plan for Arria V GX Devices

Member Code	F672		F896		F1152		F1517	
	GPIO	XCVR	GPIO	XCVR	GPIO	XCVR	GPIO	XCVR
A1	336	9	416	9	—	—	—	—
A3	336	9	416	9	—	—	—	—
A5	336	9	384	18	544	24	—	—
A7	336	9	384	18	544	24	—	—
B1	—	—	384	18	544	24	704	24
B3	—	—	384	18	544	24	704	24
B5	—	—	—	—	544	24	704	36
B7	—	—	—	—	544	24	704	36

Table 5-2: Package Plan for Arria V GT Devices

Member Code	F672			F896			F1152			F1517		
	GPIO	XCVR		GPIO	XCVR		GPIO	XCVR		GPIO	XCVR	
		6-Gbps	10-Gbps		6-Gbps	10-Gbps		6-Gbps	10-Gbps		6-Gbps	10-Gbps
C3	336	3 (9)	4	416	3 (9)	4	—	—	—	—	—	—
C7	—	—	—	384	6 (18)	8	544	6 (24)	12	—	—	—
D3	—	—	—	384	6 (18)	8	544	6 (24)	12	704	6 (24)	12
D7	—	—	—	—	—	—	544	6 (24)	12	704	6 (36)	20

Table 5-3: Package Plan for Arria V GZ Devices

Member Code	H780		F1152		F1517	
	GPIO	XCVR	GPIO	XCVR	GPIO	XCVR
E1	342	12	414	24	—	—
E3	342	12	414	24	—	—
E5	—	—	534	24	674	36
E7	—	—	534	24	674	36

Table 5-4: Package Plan for Arria V SX Devices

The HPS I/O counts are the number of I/Os in the HPS and does not correlate with the number of HPS-specific I/O pins in the FPGA. Each HPS-specific pin in the FPGA may be mapped to several HPS I/Os.

Member Code	F896			F1152			F1517		
	FPGA GPIO	HPS I/O	XCVR	FPGA GPIO	HPS I/O	XCVR	FPGA GPIO	HPS I/O	XCVR
B3	250	208	12	385	208	18	540	208	30
B5	250	208	12	385	208	18	540	208	30

Table 5-5: Package Plan for Arria V ST Devices

The HPS I/O counts are the number of I/Os in the HPS and does not correlate with the number of HPS-specific I/O pins in the FPGA. Each HPS-specific pin in the FPGA may be mapped to several HPS I/Os.

Member Code	F896				F1152				F1517			
	FPGA GPIO	HPS I/O	XCVR		FPGA GPIO	HPS I/O	XCVR		FPGA GPIO	HPS I/O	XCVR	
			6 Gbps	10 Gbps			6 Gbps	10 Gbps			6 Gbps	10 Gbps
D3	250	208	12	6	385	208	18	8	540	208	30	16
D5	250	208	12	6	385	208	18	8	540	208	30	16

For more information about each device variant, refer to the device overview.

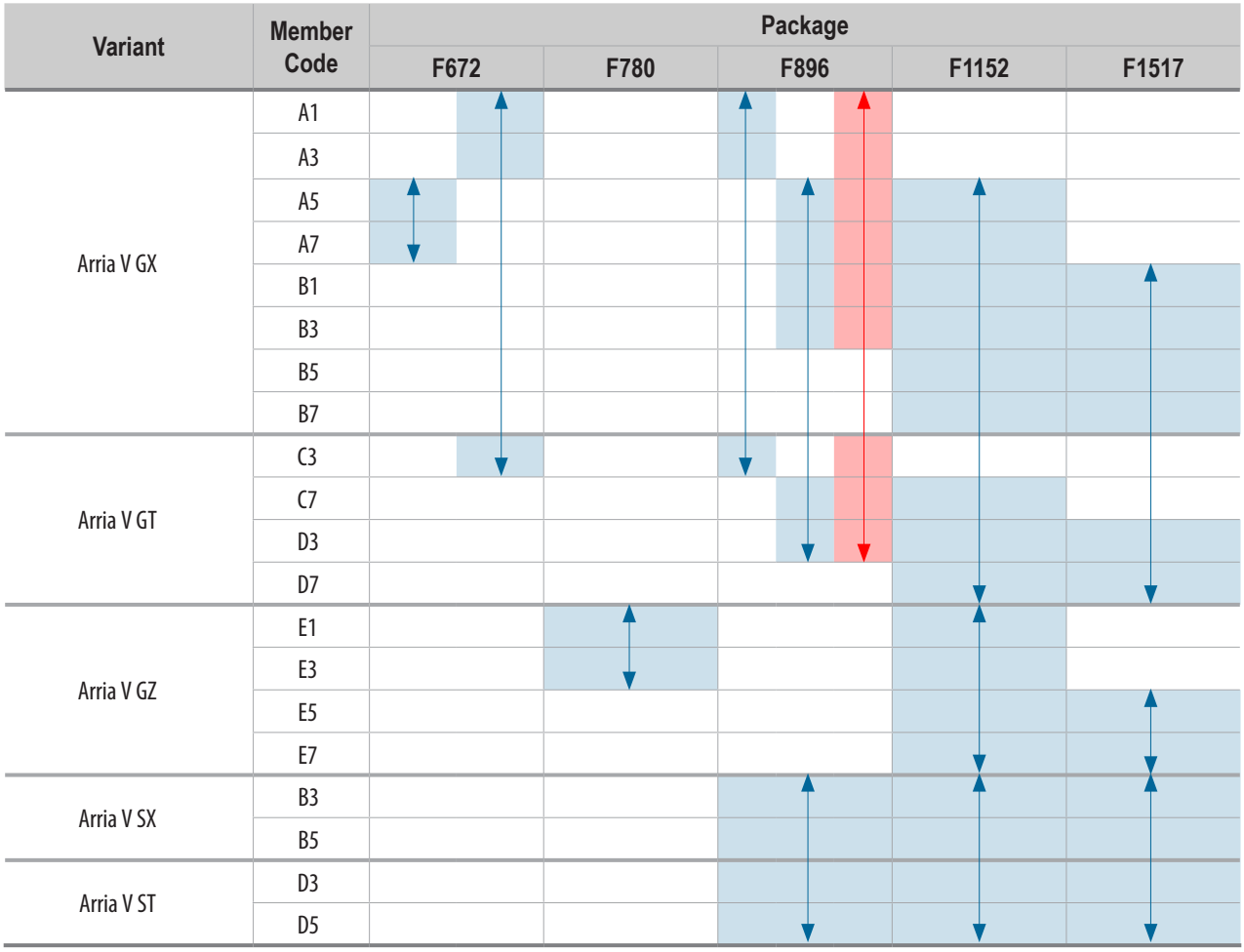
Related Information

- **True LVDS Buffers in Arria V Devices** on page 6-5
  Lists the number of LVDS channels in each device package.
- **Arria V Device Overview**

I/O Vertical Migration for Arria V Devices

Figure 5-1: Vertical Migration Capability Across Arria V Device Packages and Densities

The arrows indicate the vertical migration paths. Some packages have several migration paths. The devices included in each vertical migration path are shaded. You can also migrate your design across device densities in the same package option if the devices have the same dedicated pins, configuration pins, and power pins.



You can achieve the vertical migration shaded in red if you use only up to 320 GPIOs, up to nine 6 Gbps transceiver channels, and up to four 10 Gbps transceiver (for Arria V GT devices). This migration path is not shown in the Intel Quartus Prime software Pin Migration View.

Note: To verify the pin migration compatibility, use the Pin Migration View window in the Intel Quartus Prime software Pin Planner.

Note: Except for Arria V GX A5 and A7, and Arria V GT C7 devices, all other Arria V GX and GT devices require a specific power-up sequence. If you plan to migrate your design from Arria V GX A5 and A7, and Arria V GT C7 devices to other Arria V devices, your design must adhere to the same required power-up sequence.

Related Information

- **Arria V GX, GT, SX, and ST Power-Up Sequence** on page 11-7
- **Verifying Pin Migration Compatibility** on page 5-5
- **I/O Management chapter, Quartus II Handbook**
  Provides more information about vertical I/O migrations.
- **What is the difference between pin-to-pin compatibility and drop-in compatibility?**

Verifying Pin Migration Compatibility

You can use the **Pin Migration View** window in the Intel Quartus Prime software Pin Planner to assist you in verifying whether your pin assignments migrate to a different device successfully. You can vertically migrate to a device with a different density while using the same device package, or migrate between packages with different densities and ball counts.

1. Open **Assignments** > **Pin Planner** and create pin assignments.
2. If necessary, perform one of the following options to populate the Pin Planner with the node names in the design:
   - Analysis & Elaboration
   - Analysis & Synthesis
   - Fully compile the design
3. Then, on the menu, click **View** > **Pin Migration View**.
4. To select or change migration devices:
   a. Click **Device** to open the **Device** dialog box.
   b. Under **Migration compatibility** click **Migration Devices**.
5. To show more information about the pins:
   a. Right-click anywhere in the **Pin Migration View** window and select **Show Columns**.
   b. Then, click the pin feature you want to display.
6. If you want to view only the pins, in at least one migration device, that have a different feature than the corresponding pin in the migration result, turn on **Show migration differences**.
7. Click **Pin Finder** to open the **Pin Finder** dialog box to find and highlight pins with specific functionality.
   If you want to view only the pins highlighted by the most recent query in the **Pin Finder** dialog box, turn on **Show only highlighted pins**.
8. To export the pin migration information to a Comma-Separated Value file (**.csv**), click **Export**.

Related Information

- **I/O Vertical Migration for Arria V Devices** on page 5-4
- **I/O Management chapter, Quartus II Handbook**
  Provides more information about vertical I/O migrations.

I/O Standards Support in Arria V Devices

This section lists the I/O standards supported in the FPGA I/Os and HPS I/Os of Arria V devices, the typical power supply values for each I/O standard, and the MultiVolt I/O interface feature.

I/O Standards Support for FPGA I/O in Arria V Devices

Table 5-6: Supported I/O Standards in FPGA I/O for Arria V Devices

I/O Standard	Device Variant Support	Standard Support
3.3 V LVTTL/3.3 V LVCMOS	All	JESD8-B
3.0 V LVTTL/3.0 V LVCMOS	GX, GT, SX, and ST	JESD8-B
3.0 V PCI	GX, GT, SX, and ST	PCI Rev. 2.2
3.0 V PCI-X (10)	GX, GT, SX, and ST	PCI-X Rev. 1.0
2.5 V LVCMOS	All	JESD8-5
1.8 V LVCMOS	All	JESD8-7
1.5 V LVCMOS	All	JESD8-11
1.2 V LVCMOS	All	JESD8-12
SSTL-2 Class I	All	JESD8-9B
SSTL-2 Class II	All	JESD8-9B
SSTL-18 Class I	All	JESD8-15
SSTL-18 Class II	All	JESD8-15
SSTL-15 Class I	All	—
SSTL-15 Class II	All	—
1.8 V HSTL Class I	All	JESD8-6
1.8 V HSTL Class II	All	JESD8-6
1.5 V HSTL Class I	All	JESD8-6
1.5 V HSTL Class II	All	JESD8-6
1.2 V HSTL Class I	All	JESD8-16A
1.2 V HSTL Class II	All	JESD8-16A
Differential SSTL-2 Class I	All	JESD8-9B
Differential SSTL-2 Class II	All	JESD8-9B
Differential SSTL-18 Class I	All	JESD8-15
Differential SSTL-18 Class II	All	JESD8-15
Differential SSTL-15 Class I	All	—
Differential SSTL-15 Class II	All	—
Differential 1.8 V HSTL Class I	All	JESD8-6
Differential 1.8 V HSTL Class II	All	JESD8-6
Differential 1.5 V HSTL Class I	All	JESD8-6

(10) PCI-X does not meet the PCI-X I–V curve requirement at the linear region.

I/O Standard	Device Variant Support	Standard Support
Differential 1.5 V HSTL Class II	All	JESD8-6
Differential 1.2 V HSTL Class I	All	JESD8-16A
Differential 1.2 V HSTL Class II	All	JESD8-16A
LVDS	All	ANSI/TIA/EIA-644
RSDS(11)	All	—
Mini-LVDS(12)	All	—
LVPECL	All	—
SSTL-15	All	JESD79-3D
SSTL-135	All	—
SSTL-125	All	—
SSTL-12	GZ only	—
HSUL-12	All	—
Differential SSTL-15	All	JESD79-3D
Differential SSTL-135	All	—
Differential SSTL-125	All	—
Differential SSTL-12	GZ only	—
Differential HSUL-12	All	—

I/O Standards Support for HPS I/O in Arria V Devices

Table 5-7: Supported I/O Standards in HPS I/O for Arria V SX and ST Devices

I/O Standard	Standard Support	HPS Column I/O	HPS Row I/O
3.3 V LVTTL/3.3 V LVCMOS	JESD8-B	Yes	—
3.0 V LVTTL/3.0 V LVCMOS	JESD8-B	Yes	—
2.5 V LVCMOS	JESD8-5	Yes	—
1.8 V LVCMOS	JESD8-7	Yes	Yes
1.5 V LVCMOS	JESD8-11	Yes	—
SSTL-18 Class I	JESD8-15	—	Yes
SSTL-18 Class II	JESD8-15	—	Yes
SSTL-15 Class I	—	—	Yes

(11) The Arria V devices support true RSDS output standard with data rates of up to 360 Mbps (for Arria V GX/GT/SX/ST) and 230 Mbps (for Arria V GZ) using true LVDS output buffer types on all I/O banks.
(12) The Arria V devices support true mini-LVDS output standard with data rates of up to 400 Mbps (for Arria V GX/GT/SX/ST) and 340 Mbps (for Arria V GZ) using true LVDS output buffer types on all I/O banks.

I/O Standard	Standard Support	HPS Column I/O	HPS Row I/O
SSTL-15 Class II	—	—	Yes
1.5 V HSTL Class I	JESD8-6	Yes	—
1.5 V HSTL Class II	JESD8-6	Yes	—
SSTL-135	—	—	Yes
HSUL-12	—	—	Yes

I/O Standards Voltage Levels in Arria V Devices

Table 5-8: Arria V I/O Standards Voltage Levels

This table lists the typical power supplies for each supported I/O standards in Arria V devices.

I/O Standard	Device Variant Support	V_{CCIO} (V)		V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) [13] (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
		Input[14]	Output			
3.3 V LVTTL/3.3 V LVCMOS	GX, GT, SX, and ST	3.3/3.0/2.5	3.3	3.3	—	—
	GZ	3.0/2.5	3.0	3.0	—	—
3.0 V LVTTL/3.0 V LVCMOS	GX, GT, SX, and ST	3.3/3.0/2.5	3.0	3.0	—	—
3.0 V PCI		3.0	3.0	3.0	—	—
3.0 V PCI-X		3.0	3.0	3.0	—	—
2.5 V LVCMOS	All	3.3/3.0/2.5	2.5	2.5	—	—
1.8 V LVCMOS	All	1.8/1.5	1.8	2.5	—	—
1.5 V LVCMOS	All	1.8/1.5	1.5	2.5	—	—
1.2 V LVCMOS	All	1.2	1.2	2.5	—	—
SSTL-2 Class I	All	V_{CCPD}	2.5	2.5	1.25	1.25
SSTL-2 Class II	All	V_{CCPD}	2.5	2.5	1.25	1.25
SSTL-18 Class I	All	V_{CCPD}	1.8	2.5	0.9	0.9
SSTL-18 Class II	All	V_{CCPD}	1.8	2.5	0.9	0.9
SSTL-15 Class I	All	V_{CCPD}	1.5	2.5	0.75	0.75
SSTL-15 Class II	All	V_{CCPD}	1.5	2.5	0.75	0.75
1.8 V HSTL Class I	All	V_{CCPD}	1.8	2.5	0.9	0.9

(13) You cannot assign SSTL, HSTL, and HSUL outputs on VREF pins, even if there are no SSTL, HSTL, and HSUL inputs in the bank.

(14) Input buffers for the SSTL, HSTL, Differential SSTL, Differential HSTL, LVDS, RSDS, Mini-LVDS, LVPECL, HSUL, and Differential HSUL are powered by V_{CCPD}

I/O Standard	Device Variant Support	V_{CCIO} (V)		V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) [13] (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
		Input[14]	Output			
1.8 V HSTL Class II	All	V_{CCPD}	1.8	2.5	0.9	0.9
1.5 V HSTL Class I	All	V_{CCPD}	1.5	2.5	0.75	0.75
1.5 V HSTL Class II	All	V_{CCPD}	1.5	2.5	0.75	0.75
1.2 V HSTL Class I	All	V_{CCPD}	1.2	2.5	0.6	0.6
1.2 V HSTL Class II	All	V_{CCPD}	1.2	2.5	0.6	0.6
Differential SSTL-2 Class I	All	V_{CCPD}	2.5	2.5	—	1.25
Differential SSTL-2 Class II	All	V_{CCPD}	2.5	2.5	—	1.25
Differential SSTL-18 Class I	All	V_{CCPD}	1.8	2.5	—	0.9
Differential SSTL-18 Class II	All	V_{CCPD}	1.8	2.5	—	0.9
Differential SSTL-15 Class I	All	V_{CCPD}	1.5	2.5	—	0.75
Differential SSTL-15 Class II	All	V_{CCPD}	1.5	2.5	—	0.75
Differential 1.8 V HSTL Class I	All	V_{CCPD}	1.8	2.5	—	0.9
Differential 1.8 V HSTL Class II	All	V_{CCPD}	1.8	2.5	—	0.9
Differential 1.5 V HSTL Class I	All	V_{CCPD}	1.5	2.5	—	0.75
Differential 1.5 V HSTL Class II	All	V_{CCPD}	1.5	2.5	—	0.75
Differential 1.2 V HSTL Class I	All	V_{CCPD}	1.2	2.5	—	0.6
Differential 1.2 V HSTL Class II	All	V_{CCPD}	1.2	2.5	—	0.6
LVDS	All	V_{CCPD}	2.5	2.5	—	—
RSDS	All	V_{CCPD}	2.5	2.5	—	—
Mini-LVDS	All	V_{CCPD}	2.5	2.5	—	—

(13) You cannot assign SSTL, HSTL, and HSUL outputs on VREF pins, even if there are no SSTL, HSTL, and HSUL inputs in the bank.

(14) Input buffers for the SSTL, HSTL, Differential SSTL, Differential HSTL, LVDS, RSDS, Mini-LVDS, LVPECL, HSUL, and Differential HSUL are powered by V_{CCPD}

I/O Standard	Device Variant Support	V_{CCIO} (V)		V_{CCPD} (V) (Pre-Driver Voltage)	V_{REF} (V) [13] (Input Ref Voltage)	V_{TT} (V) (Board Termination Voltage)
		Input[14]	Output			
LVPECL (Differential clock input only)	All	V_{CCPD}	—	2.5	—	—
SSTL-15	All	V_{CCPD}	1.5	2.5	0.75	Typically does not require board termination
SSTL-135	All	V_{CCPD}	1.35	2.5	0.675	
SSTL-125	All	V_{CCPD}	1.25	2.5	0.625	
SSTL-12	GZ only	V_{CCPD}	1.2	2.5	0.6	
HSUL-12	All	V_{CCPD}	1.2	2.5	0.6	
Differential SSTL-15	All	V_{CCPD}	1.5	2.5	—	Typically does not require board termination
Differential SSTL-135	All	V_{CCPD}	1.35	2.5	—	
Differential SSTL-125	All	V_{CCPD}	1.25	2.5	—	
Differential SSTL-12	GZ only	V_{CCPD}	1.2	2.5	—	
Differential HSUL-12	All	V_{CCPD}	1.2	2.5	—	

Related Information

Guideline: Observe Device Absolute Maximum Rating for 3.3 V Interfacing on page 5-13
Provides more information about the 3.3 V LVTTL/LVCMOS I/O standard supported in Arria V GZ devices.

MultiVolt I/O Interface in Arria V Devices

The MultiVolt I/O interface feature allows Arria V devices in all packages to interface with systems of different supply voltages.

Table 5-9: MultiVolt I/O Support in Arria V Devices

V_{CCIO} (V)	Device Variant Support	V_{CCPD} (V) [14]	Input Signal (V)	Output Signal (V)
1.2	All	2.5	1.2	1.2
1.25	All	2.5	1.25	1.25
1.35	All	2.5	1.35	1.35
1.5	All	2.5	1.5, 1.8	1.5
1.8	All	2.5	1.5, 1.8	1.8
2.5	All	2.5	2.5, 3.0, 3.3	2.5

[13] You cannot assign SSTL, HSTL, and HSUL outputs on VREF pins, even if there are no SSTL, HSTL, and HSUL inputs in the bank.
[14] Input buffers for the SSTL, HSTL, Differential SSTL, Differential HSTL, LVDS, RSDS, Mini-LVDS, LVPECL, HSUL, and Differential HSUL are powered by V_{CCPD}

V_{CCIO} (V)	Device Variant Support	V_{CCPD} (V) (14)	Input Signal (V)	Output Signal (V)
3.0	GX, GT, SX, and ST	3.0	2.5, 3.0, 3.3	3.0
	GZ	3.0	2.5, 3.0, 3.3	3.0, 3.3
3.3	GX, GT, SX, and ST	3.3	2.5, 3.0, 3.3	3.3

The pin current may be slightly higher than the default value. Verify that the V_{OL} maximum and V_{OH} minimum voltages of the driving device do not violate the applicable V_{IL} maximum and V_{IH} minimum voltage specifications of the Arria V device.

The V_{CCPD} power pins must be connected to a 2.5 V, 3.0 V, or 3.3 V power supply. Using these power pins to supply the pre-driver power to the output buffers increases the performance of the output pins.

Note: If the input signal is 3.0 V or 3.3 V, Altera recommends that you use a clamping diode on the I/O pins. Use the on-chip clamping diode for the Arria V GX, GT, SX, and ST devices, and an external clamping diode for the Arria V GZ devices.

I/O Design Guidelines for Arria V Devices

There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

Mixing Voltage-Referenced and Non-Voltage-Referenced I/O Standards

Each I/O bank can simultaneously support multiple I/O standards. The following sections provide guidelines for mixing non-voltage-referenced and voltage-referenced I/O standards in the devices.

Non-Voltage-Referenced I/O Standards

Each Arria V I/O bank has its own V_{CCIO} pins and supports only one V_{CCIO} of 1.2, 1.25, 1.35, 1.5, 1.8, 2.5, 3.0, or 3.3 V(15). An I/O bank can simultaneously support any number of input signals with different I/O standard assignments if the I/O standards support the V_{CCIO} level of the I/O bank.

For output signals, a single I/O bank supports non-voltage-referenced output signals that drive at the same voltage as V_{CCIO}. Because an I/O bank can only have one V_{CCIO} value, it can only drive out the value for non-voltage-referenced signals.

For example, an I/O bank with a 2.5 V V_{CCIO} setting can support 2.5 V, 3.0 V and 3.3 V inputs but supports only 2.5 V output.

(15) Arria V GZ devices do not support 3.3 V

Voltage-Referenced I/O Standards

To accommodate voltage-referenced I/O standards:

- Each Arria V GX, GT, SX, or ST I/O bank contains a dedicated VREF pin.
- Each Arria V GZ I/O bank supports multiple dedicated VREF pins feeding a common V_{REF} bus.
- Each bank can have only a single V_{CCIO} voltage level and a single voltage reference (V_{REF}) level.

An I/O bank featuring single-ended or differential standards can support different voltage-referenced standards if the V_{CCIO} and V_{REF} are the same levels.

For performance reasons, voltage-referenced input standards use their own V_{CCPD} level as the power source. This feature allows you to place voltage-referenced input signals in an I/O bank with a V_{CCIO} of 2.5 V or below. For example, you can place HSTL-15 input pins in an I/O bank with 2.5 V V_{CCIO}. However, the voltage-referenced input with R_T OCT enabled requires the V_{CCIO} of the I/O bank to match the voltage of the input standard. R_T OCT cannot be supported for the HSTL-15 I/O standard when V_{CCIO} is 2.5 V.

Voltage-referenced bidirectional and output signals must be the same as the V_{CCIO} voltage of the I/O bank. For example, you can place only SSTL-2 output pins in an I/O bank with a 2.5 V V_{CCIO}.

Mixing Voltage-Referenced and Non-Voltage Referenced I/O Standards

An I/O bank can support voltage-referenced and non-voltage-referenced pins by applying each of the rule sets individually.

Examples:

- An I/O bank can support SSTL-18 inputs and outputs, and 1.8 V inputs and outputs with a 1.8 V V_{CCIO} and a 0.9 V V_{REF}.
- An I/O bank can support 1.5 V standards, 1.8 V inputs (but not outputs), and 1.5 V HSTL I/O standards with a 1.5 V V_{CCIO} and 0.75 V V_{REF}.

Guideline: Use the Same V_{CCPD} for All I/O Banks in a Group

One V_{CCPD} is shared in a group of I/O banks. If one I/O bank in a group uses 3.0 V V_{CCPD}, other I/O banks in the same group must also use 3.0 V V_{CCPD}.

The I/O banks with the same bank number form a group. For example, I/O banks 8A, 8B, 8C, and 8D form a group and share the same V_{CCPD}. This sharing is applicable to all I/O banks, with the following exceptions:

- Arria V GX and GT devices—No V_{CCPD} sharing in bank 4A and 7A. Each of these I/O banks has their own individual V_{CCPD}.
- Arria V SX and ST devices—No V_{CCPD} sharing in bank 4A. In these devices, banks 6A, 6B, and 7A through 7E are HPS I/O banks.
- Arria V GZ devices—No V_{CCPD} sharing across banks 3A, 3B, 3C, and 3D. Banks 3A and 3B form a group with one V_{CCPD} while bank 3C (if available) and 3D form another group with its own V_{CCPD}.

For the Arria V GZ devices, if you are using an output or bidirectional pin with the 3.3 V LVTTL or 3.3 V LVCMOS I/O standard, you must adhere to this restriction manually with location assignments.

For more information about the I/O banks available in each device package, refer to the related information.

Related Information

- **Modular I/O Banks for Arria V GX Devices** on page 5-18
- **Modular I/O Banks for Arria V GT Devices** on page 5-20
- **Modular I/O Banks for Arria V GZ Devices** on page 5-21
- **Modular I/O Banks for Arria V SX Devices** on page 5-22
- **Modular I/O Banks for Arria V ST Devices** on page 5-23

Guideline: Ensure Compatible V_{CCIO} and V_{CCPD} Voltage in the Same Bank

When planning I/O bank usage for Arria V GX, GT, SX, and ST devices, you must ensure the V_{CCIO} voltage is compatible with the V_{CCPD} voltage of the same bank. Some banks may share the same V_{CCPD} power pin. This limits the possible V_{CCIO} voltages that can be used on banks that share VCCPD power pins.

Examples:

- VCCPD4BCD is connected to 2.5 V—VCCIO pins for banks 4B, 4C, and 4D can be connected 1.2 V, 1.25 V, 1.35 V, 1.5 V, 1.8 V, or 2.5 V.
- VCCPD4BCD is connected to 3.0 V—VCCIO pins for banks 4B, 4C, and 4D must be connected to 3.0 V.

Guideline: VREF Pin Restrictions

For the Arria V GX, GT, SX, and ST devices, consider the following VREF pins guidelines:

- You cannot assign shared VREF pins as LVDS or external memory interface pins.
- SSTL, HSTL, and HSUL I/O standards do not support shared VREF pins. For example, if a particular B1p or B1n pin is a shared VREF pin, the corresponding B1p/B1n pin pair do not have LVDS transmitter support.
- Shared VREF pins will have reduced performance when used as normal I/Os.
- You must perform signal integrity analysis using your board design when using a shared VREF pin to determine the F_{MAX} for your system.

For more information about pin capacitance of the VREF pins, refer to the device datasheet.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**

Guideline: Observe Device Absolute Maximum Rating for 3.3 V Interfacing

To ensure device reliability and proper operation when you use the device for 3.3 V I/O interfacing, do not violate the absolute maximum ratings of the device. For more information about absolute maximum rating and maximum allowed overshoot during transitions, refer to the device datasheet.

Tip: Perform IBIS or SPICE simulations to make sure the overshoot and undershoot voltages are within the specifications.

Transmitter Application

If you use the Arria V device as a transmitter, use slow slew rate and series termination to limit the overshoot and undershoot at the I/O pins. Transmission line effects that cause large voltage deviations at the receiver are associated with an impedance mismatch between the driver and the transmission lines. By

matching the impedance of the driver to the characteristic impedance of the transmission line, you can significantly reduce overshoot voltage. You can use a series termination resistor placed physically close to the driver to match the total driver impedance to the transmission line impedance.

Receiver Application

If you use the Arria V device as a receiver, to limit the overshoot and undershoot voltage at the I/O pins:

- Arria V GX, GT, SX, or ST—use the on-chip clamping diode.
- Arria V GZ device—use an off-chip clamping diode.

The 3.3 V I/O standard is supported using the bank supply voltage (V_{CCIO}) at 3.0 V and a V_{CCPD} voltage of 3.0 V. In this method, the clamping diode can sufficiently clamp overshoot voltage to within the DC and AC input voltage specifications. The clamped voltage is expressed as the sum of the V_{CCIO} and the diode forward voltage.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**

Guideline: Use PLL Integer Mode for LVDS Applications

For LVDS applications, you must use the phase-locked loops (PLLs) in integer PLL mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Guideline: Pin Placement for General Purpose High-Speed Signals

For general purpose high-speed signals faster than 200 MHz, follow these guidelines to ensure I/O timing closure.

- Avoid using HMC DQ pins as the input pin.
- Avoid using HMC DQ and command pins as the output pin.

I/O signals that use the hard memory controller pins are routed through the `HMCPHY_RE` routing elements. These routing elements have a higher routing delay compared to other I/O pins. To identify the hard memory controller pins for your Arria V device and package, refer to the relevant pin-out files.

Related Information

Arria V Device Pin-Out Files
Provides the pin-out files for each Arria V device package.

I/O Banks Locations in Arria V Devices

The number of Arria V I/O banks in a particular device depends on the device density.

Figure 5-2: I/0 Banks for Arria V GX A1 and A3 Devices, and Arria V GT C3 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

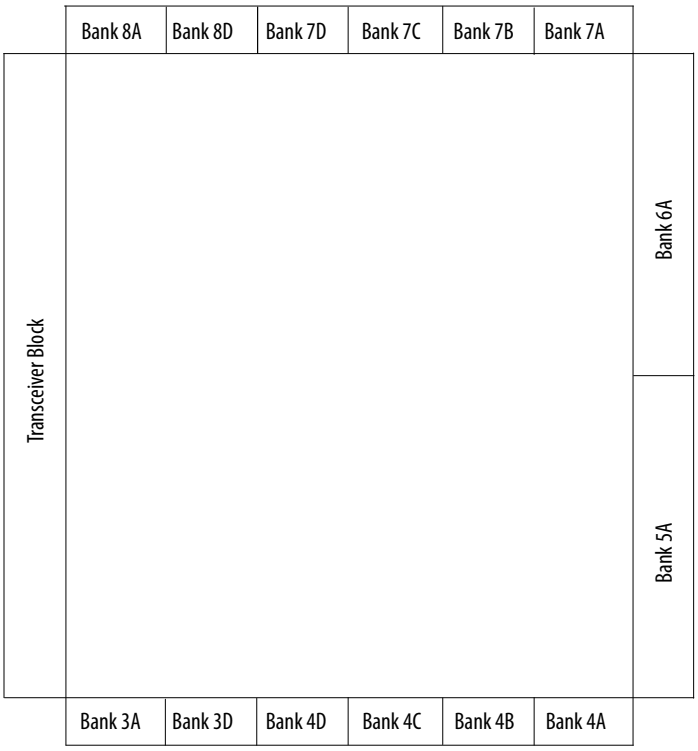


Figure 5-3: I/0 Banks for Arria V GX A5, A7, B1, B3, B5, and B7 Devices, Arria V GT C7, D3, and D7 Devices, and Arria V GZ Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.

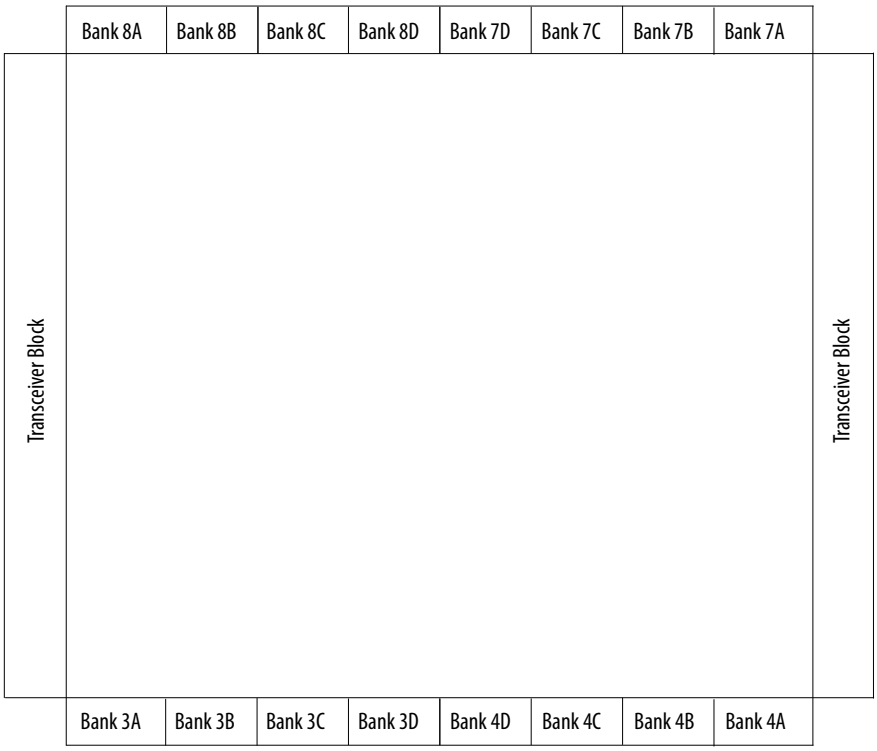
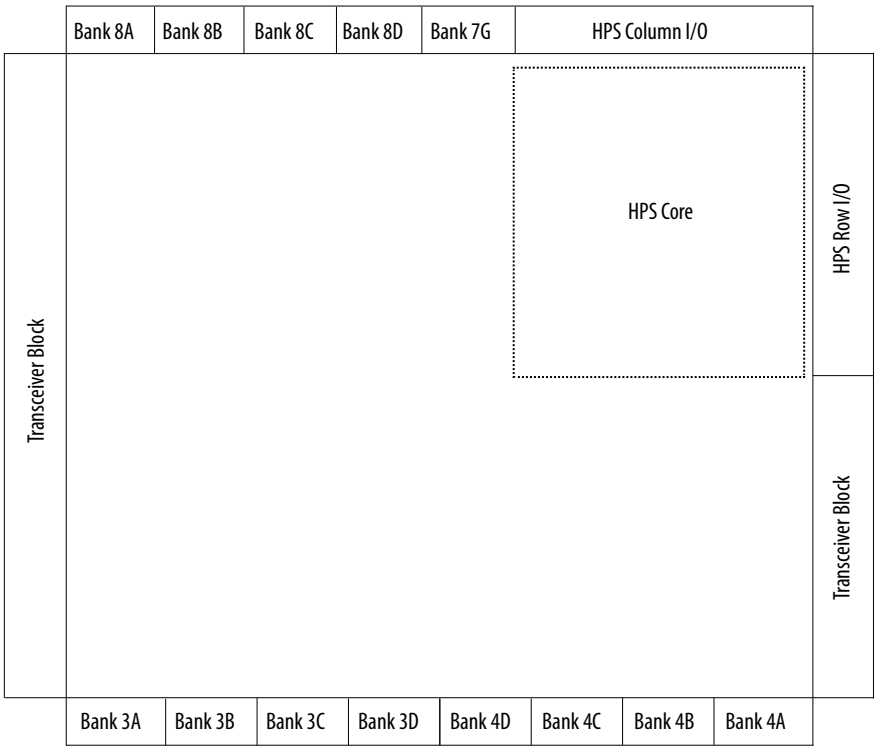


Figure 5-4: I/0 Banks for Arria V SX B3 and B5 Devices, and Arria V ST D3 and D5 Devices

This figure represents the top view of the silicon die that corresponds to a reverse view of the device package.



Related Information

- **Modular I/O Banks for Arria V GX Devices** on page 5-18
- **Modular I/O Banks for Arria V GT Devices** on page 5-20
- **Modular I/O Banks for Arria V GZ Devices** on page 5-21
- **Modular I/O Banks for Arria V SX Devices** on page 5-22
- **Modular I/O Banks for Arria V ST Devices** on page 5-23

I/O Banks Groups in Arria V Devices

The I/O pins in Arria V devices are arranged in groups called modular I/O banks:

- Modular I/O banks have independent power supplies that allow each bank to support different I/O standards.
- Each modular I/O bank can support multiple I/O standards that use the same V_{CCIO} and V_{CCPD} voltages.

Modular I/O Banks for Arria V GX Devices

Table 5-10: Modular I/O Banks for Arria V GX A1, A3, A5, and A7 Devices

Member Code		A1		A3		A5			A7		
Package		F672	F896	F672	F896	F672	F896	F1152	F672	F896	F1152
Bank	3A	24	32	24	32	24	32	48	24	32	48
	3B	—	—	—	—	—	—	32	—	—	32
	3C	—	—	—	—	—	—	32	—	—	32
	3D	32	32	32	32	20	32	32	20	32	32
	4A	16	16	16	16	28	32	32	28	32	32
	4B	—	16	—	16	32	32	32	32	32	32
	4C	32	32	32	32	32	32	32	32	32	32
	4D	32	32	32	32	32	32	32	32	32	32
	5A	32	48	32	48	—	—	—	—	—	—
	6A	32	48	32	48	—	—	—	—	—	—
	7A	16	16	16	16	28	32	32	28	32	32
	7B	—	16	—	16	32	32	32	32	32	32
	7C	32	32	32	32	32	32	32	32	32	32
	7D	32	32	32	32	32	32	32	32	32	32
	8A	24	32	24	32	24	32	48	24	32	48
	8B	—	—	—	—	—	—	32	—	—	32
	8C	—	—	—	—	—	—	32	—	—	32
	8D	32	32	32	32	20	32	32	20	32	32
Total		336	416	336	416	336	384	544	336	384	544

Table 5-11: Modular I/O Banks for Arria V GX B1, B3, B5, and B7 Devices

Member Code		B1			B3			B5		B7	
Package		F896	F1152	F1517	F896	F1152	F1517	F1152	F1517	F1152	F1517
Bank	3A	32	48	48	32	48	48	48	48	48	48
	3B	—	32	32	—	32	32	32	32	32	32
	3C	—	32	48	—	32	48	32	48	32	48
	3D	32	32	48	32	32	48	32	48	32	48
	4A	32	32	48	32	32	48	32	48	32	48
	4B	32	32	48	32	32	48	32	48	32	48
	4C	32	32	32	32	32	32	32	32	32	32
	4D	32	32	48	32	32	48	32	48	32	48
	7A	32	32	48	32	32	48	32	48	32	48
	7B	32	32	48	32	32	48	32	48	32	48
	7C	32	32	32	32	32	32	32	32	32	32
	7D	32	32	48	32	32	48	32	48	32	48
	8A	32	48	48	32	48	48	48	48	48	48
	8B	—	32	32	—	32	32	32	32	32	32
	8C	—	32	48	—	32	48	32	48	32	48
	8D	32	32	48	32	32	48	32	48	32	48
Total		384	544	704	384	544	704	544	704	544	704

Related Information

- **I/O Banks Locations in Arria V Devices** on page 5-14
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-12
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Arria V GT Devices

Table 5-12: Modular I/O Banks for Arria V GT Devices

Member Code		C3		C7		D3			D7	
Package		F672	F896	F896	F1152	F896	F1152	F1517	F1152	F1517
Bank	3A	24	32	32	48	32	48	48	48	48
	3B	—	—	—	32	—	32	32	32	32
	3C	—	—	—	32	—	32	48	32	48
	3D	32	32	32	32	32	32	48	32	48
	4A	16	16	32	32	32	32	48	32	48
	4B	—	16	32	32	32	32	48	32	48
	4C	32	32	32	32	32	32	32	32	32
	4D	32	32	32	32	32	32	48	32	48
	5A	32	48	—	—	—	—	—	—	—
	6A	32	48	—	—	—	—	—	—	—
	7A	16	16	32	32	32	32	48	32	48
	7B	—	16	32	32	32	32	48	32	48
	7C	32	32	32	32	32	32	32	32	32
	7D	32	32	32	32	32	32	48	32	48
	8A	24	32	32	48	32	48	48	48	48
	8B	—	—	—	32	—	32	32	32	32
	8C	—	—	—	32	—	32	48	32	48
	8D	32	32	32	32	32	32	48	32	48
Total		336	416	384	544	384	544	704	544	704

Related Information

- **I/O Banks Locations in Arria V Devices** on page 5-14
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-12
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Arria V GZ Devices

Table 5-13: Modular I/O Banks for Arria V GZ Devices

Member Code		E1		E3		E5		E7	
Package		F780	F1152	F780	F1152	F1152	F1517	F1152	F1517
Bank	3A	36	36	36	36	36	36	36	36
	3B	48	48	48	48	48	48	48	48
	3C	—	—	—	—	48	48	48	48
	3D	24	24	24	24	24	48	24	48
	4A	24	24	24	24	24	24	24	24
	4B	—	48	—	48	48	48	48	48
	4C	—	—	—	—	48	48	48	48
	4D	24	24	24	24	24	48	24	48
	7A	24	24	24	24	24	24	24	24
	7B	—	24	—	24	48	48	48	48
	7C	48	48	48	48	48	48	48	48
	7D	36	36	36	36	36	48	36	48
	8A	24	24	24	24	24	36	24	36
	8B	—	—	—	—	—	48	—	48
	8C	48	48	48	48	48	48	48	48
	8D	24	24	24	24	24	48	24	48
Total		360	432	360	432	552	696	552	696

Related Information

- **I/O Banks Locations in Arria V Devices** on page 5-14
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-12
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Arria V SX Devices

Table 5-14: Modular I/O Banks for Arria V SX Devices

Member Code		B3			B5		
Package		F896	F1152	F1517	F896	F1152	F1517
FPGA I/O Bank	3A	44	44	48	44	44	48
	3B	28	28	32	28	28	32
	3C	—	38	48	—	38	48
	3D	13	13	48	13	13	48
	4A	42	42	48	42	42	48
	4B	—	38	48	—	38	48
	4C	—	26	32	—	26	32
	4D	—	32	48	—	32	48
HPS Row I/O Bank	6A	56	56	56	56	56	56
	6B	44	44	44	44	44	44
HPS Column I/O Bank	7A	32	32	32	32	32	32
	7B	22	22	22	22	22	22
	7C	12	12	12	12	12	12
	7D	20	20	20	20	20	20
	7E	8	8	8	8	8	8
FPGA I/O Bank	7G	—	—	12	—	—	12
	8A	44	44	48	44	44	48
	8B	28	28	32	28	28	32
	8C	38	38	48	38	38	48
	8D	13	14	48	13	14	48
Total		444	579	734	444	579	734

Related Information

- **I/O Banks Locations in Arria V Devices** on page 5-14
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-12
  Provides guidelines about V_{CCPD} and I/O banks groups.

Modular I/O Banks for Arria V ST Devices

Table 5-15: Modular I/O Banks for Arria V ST Devices

Member Code		D3			D5		
Package		F896	F1152	F1517	F896	F1152	F1517
FPGA I/O Bank	3A	44	44	48	44	44	48
	3B	28	28	32	28	28	32
	3C	—	38	48	—	38	48
	3D	13	13	48	13	13	48
	4A	42	42	48	42	42	48
	4B	—	38	48	—	38	48
	4C	—	26	32	—	26	32
	4D	—	32	48	—	32	48
HPS Row I/O Bank	6A	56	56	56	56	56	56
	6B	44	44	44	44	44	44
HPS Column I/O Bank	7A	32	32	32	32	32	32
	7B	22	22	22	22	22	22
	7C	12	12	12	12	12	12
	7D	20	20	20	20	20	20
	7E	8	8	8	8	8	8
FPGA I/O Bank	7G	—	—	12	—	—	12
	8A	44	44	48	44	44	48
	8B	28	28	32	28	28	32
	8C	38	38	48	38	38	48
	8D	13	14	48	13	14	48
Total		444	579	734	444	579	734

Related Information

- **I/O Banks Locations in Arria V Devices** on page 5-14
- **Guideline: Use the Same VCCPD for All I/O Banks in a Group** on page 5-12
  Provides guidelines about V_{CCPD} and I/O banks groups.

I/O Element Structure in Arria V Devices

The I/O elements (IOEs) in Arria V devices contain a bidirectional I/O buffer and I/O registers to support a complete embedded bidirectional single data rate (SDR) or double data rate (DDR) transfer.

The IOEs are located in I/O blocks around the periphery of the Arria V device.

The Arria V SX and ST devices also have I/O elements for the HPS.

I/O Buffer and Registers in Arria V Devices

I/O registers are composed of the input path for handling data from the pin to the core, the output path for handling data from the core to the pin, and the output enable (OE) path for handling the OE signal to the output buffer. These registers allow faster source-synchronous register-to-register transfers and resynchronization.

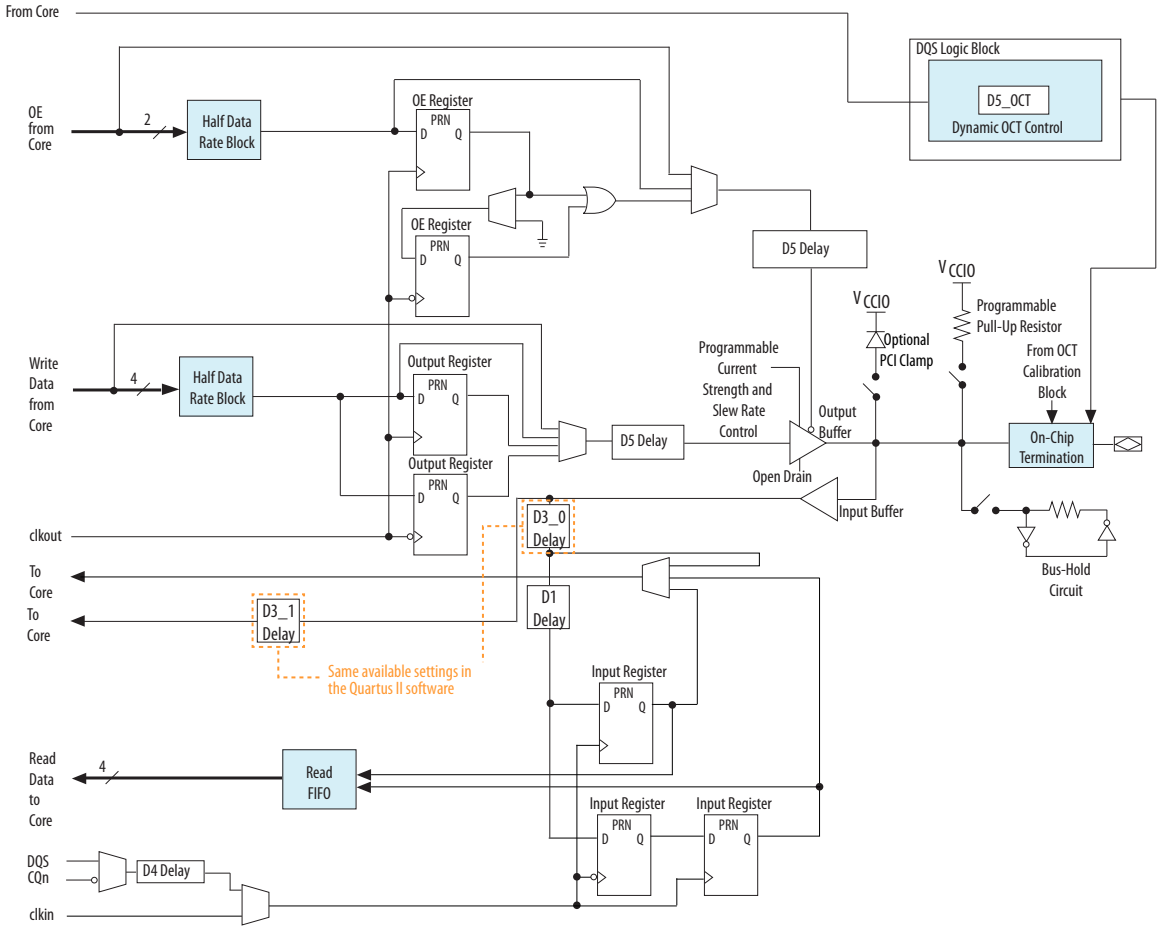
Table 5-16: Input and Output Paths in Arria V Devices

This table summarizes the input and output path in the Arria V devices.

Input Path	Output Path
Consists of: • DDR input registers • Alignment and synchronization registers • Half data rate blocks	Consists of: • Output or OE registers • Alignment registers • Half data rate blocks
You can bypass each block in the input path. The input path uses the deskew delay to adjust the input register clock delay across process, voltage, and temperature (PVT) variations.	You can bypass each block of the output and OE paths.

Figure 5-5: IOE Structure for Arria V Devices

This figure shows the Arria V FPGA IOE structure. In the figure, one dynamic on-chip termination (OCT) control is available for each DQ/DQS group.



Programmable IOE Features in Arria V Devices

Table 5-17: Summary of Supported Arria V Programmable IOE Features and Settings

Feature	Setting	Assignment Name	Supported I/O Standards	Supported in HPS I/O (SoC Devices Only)
Slew Rate Control[16]	• 0 (Slow) • 1 (Fast). Default is 1.	`Slew Rate`	• 3.0/3.3V LVTTL • 1.2/1.5/1.8/2.5/3.0/3.3 LVCMOS • SSTL-2/SSTL-18/SSTL-15 • 1.8/1.5/1.2V HSTL • 3.0V PCI • 3.0V PCI-X • Differential SSTL-2/ Differential SSTL-18/ Differential SSTL-15 • Differential 1.2/1.5/1.8V HSTL	Yes
Programmable Output Buffer Delay	• 0 ps (Default) • 50 ps • 100 ps • 150 ps	`Output Buffer Delay`	• 3.0/3.3-V LVTTL • 1.2/1.5/1.8/2.5/3.0/3.3 LVCMOS • SSTL-2, SSTL-18, SSTL-15, SSTL-135, SSTL-125 • 1.8/1.5/1.2 V HSTL • HSUL-12 • 3.0V PCI • 3.0V PCI-X • Differential SSTL-2/Diff-SSTL-18/Differential SSTL-15/Differential SSTL-135/ Differential SSTL-125 • Differential 1.2/1.5/1.8V HSTL • Differential 1.2-V HSUL	—

(16) Disabled if you use the R_S OCT feature.

Feature	Setting	Assignment Name	Supported I/O Standards	Supported in HPS I/O (SoC Devices Only)
Open-Drain Output[17]	• On • Off (Default)	—	• 3.0/3.3V LVTTL • 1.2/1.5/1.8/2.5/3.0/3.3 LVCMOS • SSTL-2/SSTL-18/SSTL-15/ SSTL-135/SSTL-125 • 1.8/1.5/1.2V HSTL • HSUL-12 • 3.0V PCI • 3.0V PCI-X	Yes
Bus-Hold[18]	• On • Off (Default)	`Enable Bus-Hold Circuitry`	• 3.0/3.3V LVTTL • 1.2/1.5/1.8/2.5/3.0/3.3 LVCMOS • SSTL-2/SSTL-18/SSTL-15/ SSTL-135/SSTL-125 • 1.8/1.5/1.2V HSTL • HSUL-12 • 3.0V PCI • 3.0V PCI-X	Yes
Weak Pull-up Resistor[19]	• On • Off (Default)	`Weak Pull-Up Resistor`	• 3.0/3.3V LVTTL • 1.2/1.5/1.8/2.5/3.0/3.3 LVCMOS • SSTL-2/SSTL-18/SSTL-15/ SSTL-135/SSTL-125 • 1.8/1.5/1.2V HSTL • HSUL-12 • 3.0V PCI • 3.0V PCI-X	Yes
Pre-Emphasis	• 0 (disabled) • 1 (enabled). Default is 1.	`Programmable Pre-emphasis`	• LVDS • RSDS • Mini-LVDS	—

(17) Open drain feature can be enabled using the OPNDRN primitive.
(18) Disabled if you use the weak pull-up resistor feature.
(19) Disabled if you use the bus-hold feature.

Feature	Setting	Assignment Name	Supported I/O Standards	Supported in HPS I/O (SoC Devices Only)
Differential Output Voltage	Arria V GX, GT, SX, and ST: • 0 (low) • 1 (medium) • 2 (high). Default is 1. Arria V GZ: • 0 (low) • 1 (medium low) • 2 (medium high) • 3 (high). Default is 1.	`Programmable Differential Output Voltage (VOD)`	• LVDS • RSDS • Mini-LVDS	—
On-Chip Clamp Diode [20] (GX, GT, SX, and ST only)	• On • Off (Default)	`Clamping Diode`	• 3.0/3.3V LVTTL • 3.0/3.3 LVCMOS • 3.0V PCI • 3.0V PCI-X	Yes

Note: The on-chip clamp diode is available on all general purpose I/O (GPIO) pins in the Arria V GX, GT, SX, and ST device variants.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**
- **Programmable Current Strength** on page 5-28
- **Programmable Output Slew Rate Control** on page 5-29
- **Programmable IOE Delay** on page 5-30
- **Programmable Output Buffer Delay** on page 5-30
- **Programmable Pre-Emphasis** on page 5-31
- **Programmable Differential Output Voltage** on page 5-31

Programmable Current Strength

You can use the programmable current strength to mitigate the effects of high signal attenuation that is caused by a long transmission line or a legacy backplane.

[20] PCI clamp diode is enabled by default for 3.0 V PCI and 3.0 V PCI-X standards.

Table 5-18: Programmable Current Strength Settings for Arria V Devices

The output buffer for each Arria V device I/O pin has a programmable current strength control for the I/O standards listed in this table.

I/O Standard	I_{OH} / I_{OL} Current Strength Setting (mA) (Default setting in bold)		Supported in HPS (SoC Devices Only)
	Arria V GX, GT, SX, and ST	Arria V GZ	
3.3-V LVTTL	**8** , 4	16, **12** , 8, 4	Yes
3.3-V LVCMOS	**2**	16, **12** , 8, 4	Yes
3.0-V LVTTL	16, **12** , 8, 4	—	Yes
3.0-V LVCMOS	16, **12** , 8, 4	—	Yes
2.5-V LVCMOS	16, **12** , 8, 4	16, **12** , 8, 4	Yes
1.8-V LVCMOS	**12** , 10, 8, 6, 4, 2	**12** , 10, 8, 6, 4, 2	Yes
1.5-V LVCMOS	**12** , 10, 8, 6, 4, 2	**12** , 10, 8, 6, 4, 2	Yes
1.2-V LVCMOS	**8** , 6, 4, 2	**8** , 6, 4, 2	—
SSTL-2 Class I	12, 10, **8**	12, 10, **8**	—
SSTL-2 Class II	**16**	**16**	—
SSTL-18 Class I	12, 10, **8** , 6, 4	12, 10, **8** , 6, 4	Yes
SSTL-18 Class II	**16**	**16** , 8	Yes
SSTL-15 Class I	12, 10, **8** , 6, 4	12, 10, **8** , 6, 4	Yes
SSTL-15 Class II	**16**	**16** , 8	Yes
1.8-V HSTL Class I	12, 10, **8** , 6, 4	12, 10, **8** , 6, 4	—
1.8-V HSTL Class II	**16**	**16**	—
1.5-V HSTL Class I	12, 10, **8** , 6, 4	12, 10, **8** , 6, 4	Yes
1.5-V HSTL Class II	**16**	**16**	Yes
1.2-V HSTL Class I	12, 10, **8** , 6, 4	12, 10, **8** , 6, 4	—
1.2-V HSTL Class II	**16**	**16**	—

For the Arria V GZ devices, the 3.3 V LVTTL and 3.3 V LVCMOS I/O standards are supported using V_{CCIO} and V_{CCPD} at 3.0 V .

Note: Intel recommends that you perform IBIS or SPICE simulations to determine the best current strength setting for your specific application.

Related Information

Programmable IOE Features in Arria V Devices on page 5-26

Programmable Output Slew Rate Control

Programmable output slew rate is available for single-ended I/O standards and emulated LVDS output standards.

The programmable output slew rate control in the output buffer of each regular- and dual-function I/O pin allows you to configure the following:

- Fast slew rate—provides high-speed transitions for high-performance systems.
- Slow slew rate—reduces system noise and crosstalk but adds a nominal delay to the rising and falling edges.

You can specify the slew rate on a pin-by-pin basis because each I/O pin contains a slew rate control.

Note: Altera recommends that you perform IBIS or SPICE simulations to determine the best slew rate setting for your specific application.

Related Information

Programmable IOE Features in Arria V Devices on page 5-26

Programmable IOE Delay

You can activate the programmable IOE delays to ensure zero hold times, minimize setup times, or increase clock-to-output times. This feature helps read and write timing margins because it minimizes the uncertainties between signals in the bus.

Each pin can have a different input delay from pin-to-input register or a delay from output register-to-output pin values to ensure that the signals within a bus have the same delay going into or out of the device.

For more information about the programmable IOE delay specifications, refer to the device datasheet.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**
- **Programmable IOE Features in Arria V Devices** on page 5-26

Programmable Output Buffer Delay

The delay chains are built inside the single-ended output buffer. There are four levels of output buffer delay settings. By default, there is no delay.

The delay chains can independently control the rising and falling edge delays of the output buffer, allowing you to:

- Adjust the output-buffer duty cycle
- Compensate channel-to-channel skew
- Reduce simultaneous switching output (SSO) noise by deliberately introducing channel-to-channel skew
- Improve high-speed memory-interface timing margins

For more information about the programmable output buffer delay specifications, refer to the device datasheet.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**

- **Programmable IOE Features in Arria V Devices** on page 5-26

Programmable Pre-Emphasis

The V_{OD} setting and the output impedance of the driver set the output current limit of a high-speed transmission signal. At a high frequency, the slew rate may not be fast enough to reach the full V_{OD} level before the next edge, producing pattern-dependent jitter. With pre-emphasis, the output current is boosted momentarily during switching to increase the output slew rate.

Pre-emphasis increases the amplitude of the high-frequency component of the output signal, and thus helps to compensate for the frequency-dependent attenuation along the transmission line. The overshoot introduced by the extra current happens only during a change of state switching to increase the output slew rate and does not ring, unlike the overshoot caused by signal reflection. The amount of pre-emphasis required depends on the attenuation of the high-frequency component along the transmission line.

Figure 5-6: Programmable Pre-Emphasis

This figure shows the LVDS output with pre-emphasis.

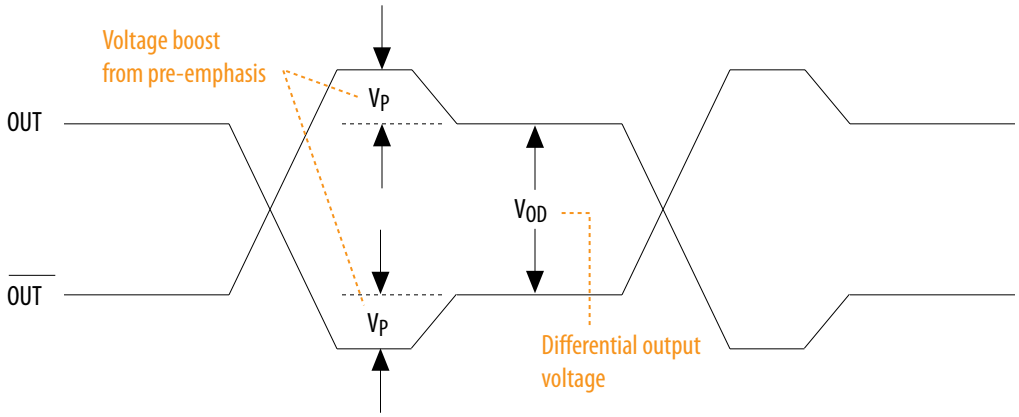


Table 5-19: Intel Quartus Prime Software Assignment Editor—Programmable Pre-Emphasis

This table lists the assignment name for programmable pre-emphasis and its possible values in the Intel Quartus Prime software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Pre-emphasis
Allowed values	0 (disabled), 1 (enabled). Default is 1.

Related Information

Programmable IOE Features in Arria V Devices on page 5-26

Programmable Differential Output Voltage

The programmable V_{OD} settings allow you to adjust the output eye opening to optimize the trace length and power consumption. A higher V_{OD} swing improves voltage margins at the receiver end, and a smaller V_{OD} swing reduces power consumption. You can statically adjust the V_{OD} of the differential signal by changing the V_{OD} settings in the Intel Quartus Prime software Assignment Editor.

Figure 5-7: Differential V_{OD}

This figure shows the V_{OD} of the differential LVDS output.

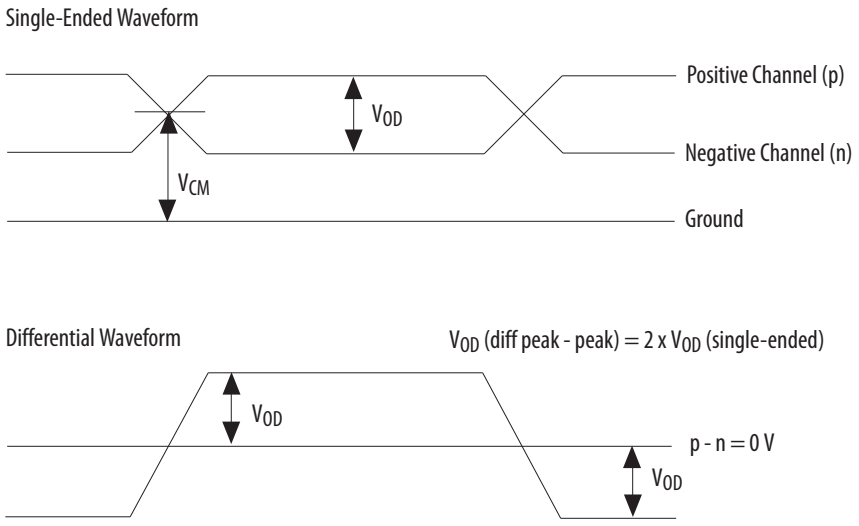


Table 5-20: Intel Quartus Prime Software Assignment Editor—Programmable V_{OD}

This table lists the assignment name for programmable V_{OD} and its possible values in the Intel Quartus Prime software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Differential Output Voltage (V_{OD})
Allowed values	• Arria V GX, GT, SX, and ST—0 (low), 1 (medium), 2 (high). Default is 1. • Arria V GZ—0 (low), **1** (medium low), 2 (medium high), 3 (high). Default is 1.

Related Information

Programmable IOE Features in Arria V Devices on page 5-26

Open-Drain Output

The optional open-drain output for each I/O pin is equivalent to an open collector output. If it is configured as an open drain, the logic value of the output is either high-Z or logic low.

You can attach several open-drain output to a wire. This connection type is like a logical OR function and is commonly called an active-low wired-OR circuit. If at least one of the outputs is in logic 0 state (active), the circuit sinks the current and brings the line to low voltage.

You can use open-drain output if you are connecting multiple devices to a bus. For example, you can use the open-drain output for system-level control signals that can be asserted by any device or as an interrupt.

You can enable the open-drain output assignment using one these methods:

- Design the tristate buffer using OPNDRN primitive.
- Turn on the **Auto Open-Drain Pins** option in the Intel Quartus Prime software.

Although you can design open-drain output without enabling the option assignment, you will not be using the open-drain output feature of the I/O buffer. The open-drain output feature in the I/O buffer provides you the best propagation delay from OE to output.

Pull-up Resistor

Each I/O pin provides an optional programmable pull-up resistor during user mode. The pull-up resistor, typically 25 kΩ, weakly holds the I/O to the V_{CCIO} level. If you enable this option, you cannot use the bus-hold feature.

The Arria V device supports programmable weak pull-up resistors only on user I/O pins.

For dedicated configuration pins, dedicated clock pins, or JTAG pins with internal pull-up resistors, these resistor values are not programmable. You can find more information related to the internal pull-up values for dedicated configuration pins, dedicated clock pins, or JTAG pins in the Arria V Pin Connection Guidelines.

Bus-Hold Circuitry

Each I/O pin provides an optional bus-hold feature that is active only after configuration. When the device enters user mode, the bus-hold circuit captures the value that is present on the pin by the end of the configuration.

The bus-hold circuitry uses a resistor with a nominal resistance (R_{BH}), approximately 7 kΩ, to weakly pull the signal level to the last-driven state of the pin. The bus-hold circuitry holds this pin state until the next input signal is present. Because of this, you do not require an external pull-up or pull-down resistor to hold a signal level when the bus is tri-stated.

For each I/O pin, you can individually specify that the bus-hold circuitry pulls non-driven pins away from the input threshold voltage—where noise can cause unintended high-frequency switching. To prevent over-driving signals, the bus-hold circuitry drives the voltage level of the I/O pin lower than the V_{CCIO} level.

If you enable the bus-hold feature, you cannot use the programmable pull-up option. To configure the I/O pin for differential signals, disable the bus-hold feature.

On-Chip I/O Termination in Arria V Devices

Dynamic R_S and R_T OCT provides I/O impedance matching and termination capabilities. OCT maintains signal quality, saves board space, and reduces external component costs.

The Arria V devices support OCT in all FPGA I/O banks. For the HPS I/Os, the column I/Os do not support OCT with calibration.

Table 5-21: OCT Schemes Supported in Arria V Devices

Direction	OCT Schemes	Supported in HPS Row I/Os
Output	R_S OCT with calibration	Yes
	R_S OCT without calibration	Yes
Input	R_T OCT with calibration	Yes
	R_D OCT (LVDS I/O standard only)	—
Bidirectional	Dynamic R_S OCT and R_T OCT	Yes

R_S OCT without Calibration in Arria V Devices

The Arria V devices support R_S OCT for single-ended and voltage-referenced I/O standards. R_S OCT without calibration is supported on output only.

Table 5-22: Selectable I/O Standards for R_S OCT Without Calibration

This table lists the output termination settings for uncalibrated OCT on different I/O standards.

I/O Standard	Device Variant Support	Uncalibrated OCT (Output)
		R_S (Ω)
3.3 V LVTTL/3.3 V LVCMOS	GZ only	25/50
3.0 V LVTTL/3.0 V LVCMOS	GX, GT, SX, and ST	25/50
2.5 V LVCMOS	All	25/50
1.8 V LVCMOS	All	25/50
1.5 V LVCMOS	All	25/50
1.2 V LVCMOS	All	25/50
SSTL-2 Class I	All	50
SSTL-2 Class II	All	25
SSTL-18 Class I	All	50
SSTL-18 Class II	All	25
SSTL-15 Class I	All	50
SSTL-15 Class II	All	25
1.8 V HSTL Class I	All	50
1.8 V HSTL Class II	All	25
1.5 V HSTL Class I	All	50
1.5 V HSTL Class II	All	25
1.2 V HSTL Class I	All	50

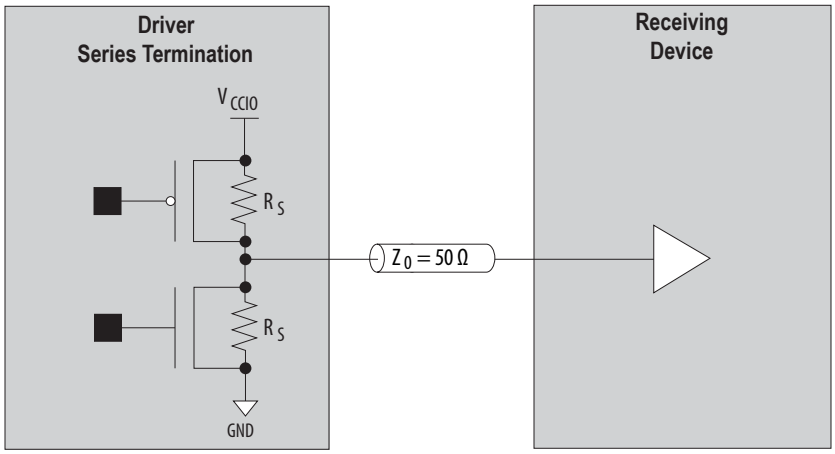
I/O Standard	Device Variant Support	Uncalibrated OCT (Output)
		R_S (Ω)
1.2 V HSTL Class II	All	25
Differential SSTL-2 Class I	All	50
Differential SSTL-2 Class II	All	25
Differential SSTL-18 Class I	All	50
Differential SSTL-18 Class II	All	25
Differential SSTL-15 Class I	All	50
Differential SSTL-15 Class II	All	25
Differential 1.8 V HSTL Class I	All	50
Differential 1.8 V HSTL Class II	All	25
Differential 1.5 V HSTL Class I	All	50
Differential 1.5 V HSTL Class II	All	25
Differential 1.2 V HSTL Class I	All	50
Differential 1.2 V HSTL Class II	All	25
SSTL-15	GZ only	25, 34, 40, 50
SSTL-135	GZ only	34, 40
SSTL-125	GZ only	34, 40
SSTL-12	GZ only	40, 60, 240
HSUL-12	GZ only	34.3, 40, 48, 60, 80

Driver-impedance matching provides the I/O driver with controlled output impedance that closely matches the impedance of the transmission line. As a result, you can significantly reduce signal reflections on PCB traces.

If you select matching impedance, current strength is no longer selectable.

Figure 5-8: R_S OCT Without Calibration

This figure shows the R_S as the intrinsic impedance of the output transistors.



R_S OCT with Calibration in Arria V Devices

The Arria V devices support R_S OCT with calibration in all banks.

Table 5-23: Selectable I/O Standards for R_S OCT With Calibration

This table lists the output termination settings for calibrated OCT on different I/O standards.

I/O Standard	Device Variant Support	Calibrated OCT (Output)	
		R_S (Ω)	RZQ (Ω)
3.3 V LVTTL/3.3 V LVCMOS	GZ only	25/50	100
3.0 V LVTTL/3.0 V LVCMOS	GX, GT, SX, and ST	25/50	100
2.5 V LVCMOS	All	25/50	100
1.8 V LVCMOS	All	25/50	100
1.5 V LVCMOS	All	25/50	100
1.2 V LVCMOS	All	25/50	100
SSTL-2 Class I	All	50	100
SSTL-2 Class II	All	25	100
SSTL-18 Class I	All	50	100
SSTL-18 Class II	All	25	100
SSTL-15 Class I	All	50	100
SSTL-15 Class II	All	25	100
1.8 V HSTL Class I	All	50	100
1.8 V HSTL Class II	All	25	100
1.5 V HSTL Class I	All	50	100

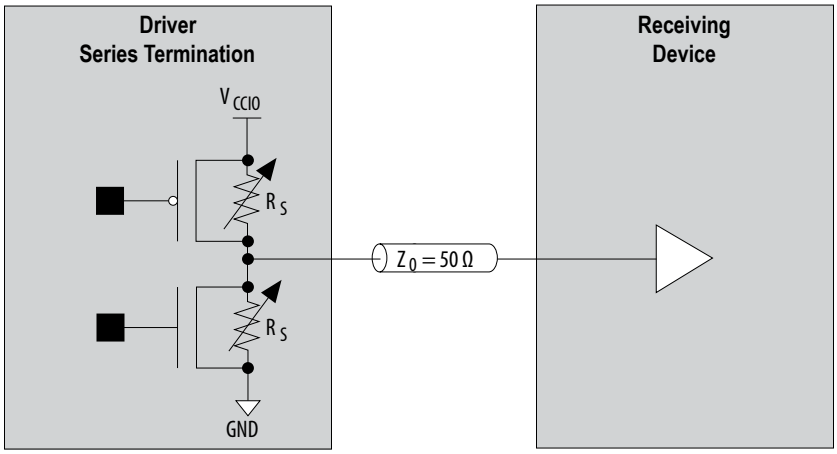
I/O Standard	Device Variant Support	Calibrated OCT (Output)	
		R_S (Ω)	RZQ (Ω)
1.5 V HSTL Class II	All	25	100
1.2 V HSTL Class I	All	50	100
1.2 V HSTL Class II	All	25	100
Differential SSTL-2 Class I	All	50	100
Differential SSTL-2 Class II	All	25	100
Differential SSTL-18 Class I	All	50	100
Differential SSTL-18 Class II	All	25	100
Differential SSTL-15 Class I	All	50	100
Differential SSTL-15 Class II	All	25	100
Differential 1.8 V HSTL Class I	All	50	100
Differential 1.8 V HSTL Class II	All	25	100
Differential 1.5 V HSTL Class I	All	50	100
Differential 1.5 V HSTL Class II	All	25	100
Differential 1.2 V HSTL Class I	All	50	100
Differential 1.2 V HSTL Class II	All	25	100
SSTL-15	All	25, 50	100
	All	34, 40	240
SSTL-135	All	34, 40	240
SSTL-125	All	34, 40	240
SSTL-12	GZ only	40, 60, 240	240
HSUL-12	All	34, 40, 48, 60, 80	240
Differential SSTL-15	All	25, 50	100
	All	34, 40	240
Differential SSTL-135	All	34, 40	240
Differential SSTL-125	All	34, 40	240
Differential SSTL-12	GZ only	40, 60, 240	240
Differential HSUL-12	All	34, 40, 48, 60, 80	240

The R_S OCT calibration circuit compares the total impedance of the I/O buffer to the external reference resistor connected to the RZQ pin and dynamically enables or disables the transistors until they match.

Calibration occurs at the end of device configuration. When the calibration circuit finds the correct impedance, the circuit powers down and stops changing the characteristics of the drivers.

Figure 5-9: R_S OCT with Calibration

This figure shows the R_S as the intrinsic impedance of the output transistors.



R_T OCT with Calibration in Arria V Devices

The Arria V devices support R_T OCT with calibration in all banks. R_T OCT with calibration is available only for configuration of input and bidirectional pins. Output pin configurations do not support R_T OCT with calibration. If you use R_T OCT, the V_{CCIO} of the bank must match the I/O standard of the pin where you enable the R_T OCT.

Table 5-24: Selectable I/O Standards for R_T OCT With Calibration

This table lists the input termination settings for calibrated OCT on different I/O standards.

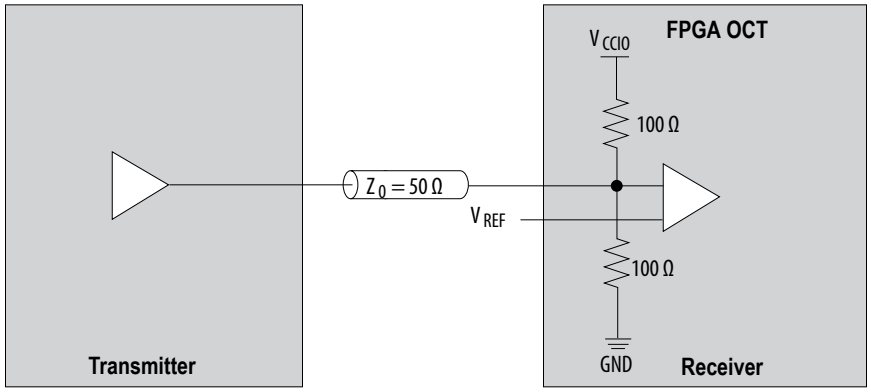
I/O Standard	Device Variant Support	Calibrated OCT (Input)	
		R_T (Ω)	RZQ (Ω)
SSTL-2 Class I	All	50	100
SSTL-2 Class II	All	50	100
SSTL-18 Class I	All	50	100
SSTL-18 Class II	All	50	100
SSTL-15 Class I	All	50	100
SSTL-15 Class II	All	50	100
1.8 V HSTL Class I	All	50	100
1.8 V HSTL Class II	All	50	100
1.5 V HSTL Class I	All	50	100
1.5 V HSTL Class II	All	50	100
1.2 V HSTL Class I	All	50	100
1.2 V HSTL Class II	All	50	100
Differential SSTL-2 Class I	All	50	100

I/O Standard	Device Variant Support	Calibrated OCT (Input)	
		R_T (Ω)	RZQ (Ω)
Differential SSTL-2 Class II	All	50	100
Differential SSTL-18 Class I	All	50	100
Differential SSTL-18 Class II	All	50	100
Differential SSTL-15 Class I	All	50	100
Differential SSTL-15 Class II	All	50	100
Differential 1.8 V HSTL Class I	All	50	100
Differential 1.8 V HSTL Class II	All	50	100
Differential 1.5 V HSTL Class I	All	50	100
Differential 1.5 V HSTL Class II	All	50	100
Differential 1.2 V HSTL Class I	All	50	100
Differential 1.2 V HSTL Class II	All	50	100
SSTL-15	All	20, 30, 40, 60,120	240
SSTL-135	All	20, 30, 40, 60, 120	240
SSTL-125	All	20, 30, 40, 60, 120	240
SSTL-12	GZ only	60, 120	240
HSUL-12	GZ only	34, 40, 48, 60, 80	240
Differential SSTL-15	All	20, 30, 40, 60,120	240
Differential SSTL-135	All	20, 30, 40, 60, 120	240
Differential SSTL-125	All	20, 30, 40, 60, 120	240
Differential SSTL-12	GZ only	60, 120	240
Differential HSUL-12	GZ only	34, 40, 48, 60, 80	240

The R_T OCT calibration circuit compares the total impedance of the I/O buffer to the external resistor connected to the RZQ pin. The circuit dynamically enables or disables the transistors until the total impedance of the I/O buffer matches the external resistor.

Calibration occurs at the end of the device configuration. When the calibration circuit finds the correct impedance, the circuit powers down and stops changing the characteristics of the drivers.

Figure 5-10: R_T OCT with Calibration



Dynamic OCT in Arria V Devices

Dynamic OCT is useful for terminating a high-performance bidirectional path by optimizing the signal integrity depending on the direction of the data. Dynamic OCT also helps save power because device termination is internal—termination switches on only during input operation and thus draw less static power.

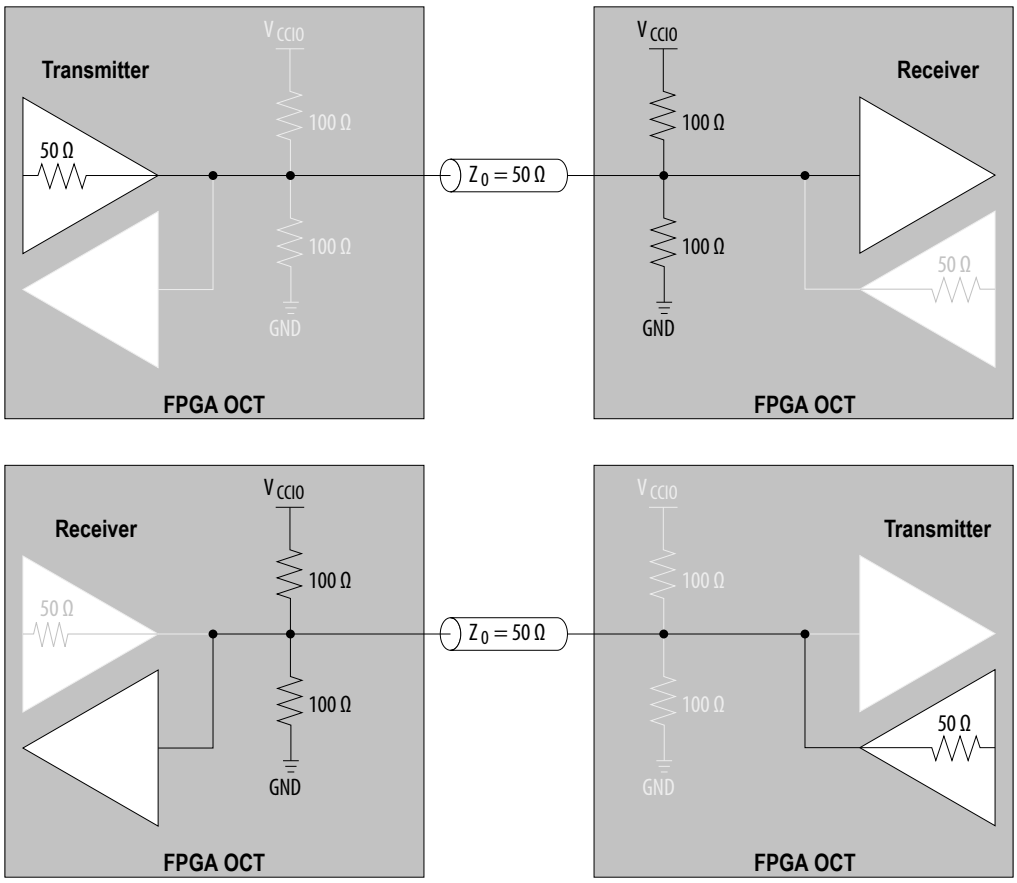
Note: If you use the SSTL-15, SSTL-135, and SSTL-125 I/O standards with external memory interfaces, Intel recommends that you use OCT with these I/O standards to save board space and cost. OCT reduces the number of external termination resistors used.

Table 5-25: Dynamic OCT Based on Bidirectional I/O

Dynamic R_T OCT or R_S OCT is enabled or disabled based on whether the bidirectional I/O acts as a receiver or driver.

Dynamic OCT	Bidirectional I/O	State
Dynamic R_T OCT	Acts as a receiver	Enabled
	Acts as a driver	Disabled
Dynamic R_S OCT	Acts as a receiver	Disabled
	Acts as a driver	Enabled

Figure 5-11: Dynamic R_T OCT in Arria V Devices



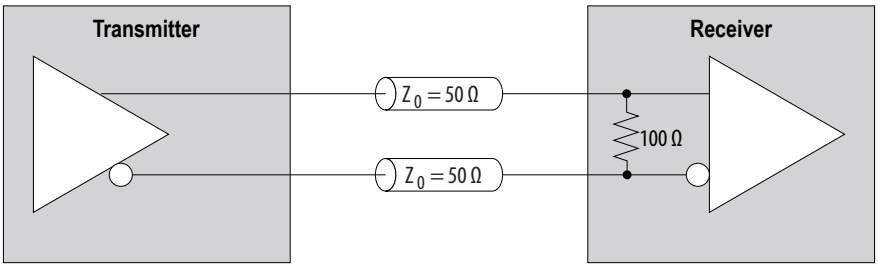
LVDS Input R_D OCT in Arria V Devices

The Arria V devices support R_D OCT in all I/O banks.

You can only use R_D OCT if you set the V_{CCPD} to 2.5 V.

Figure 5-12: Differential Input OCT

The Arria V devices support OCT for differential LVDS input buffers with a nominal resistance value of 100 Ω, as shown in this figure.



OCT Calibration Block in Arria V Devices

You can calibrate the OCT using any of the available four OCT calibration blocks for each device. Each calibration block contains one RZQ pin.

You can use R_S and R_T OCT in the same I/O bank for different I/O standards if the I/O standards use the same V_{CCIO} supply voltage. You cannot configure the R_S OCT and the programmable current strength for the same I/O buffer.

The OCT calibration process uses the RZQ pin that is available in every calibration block in a given I/O bank for series- and parallel-calibrated termination:

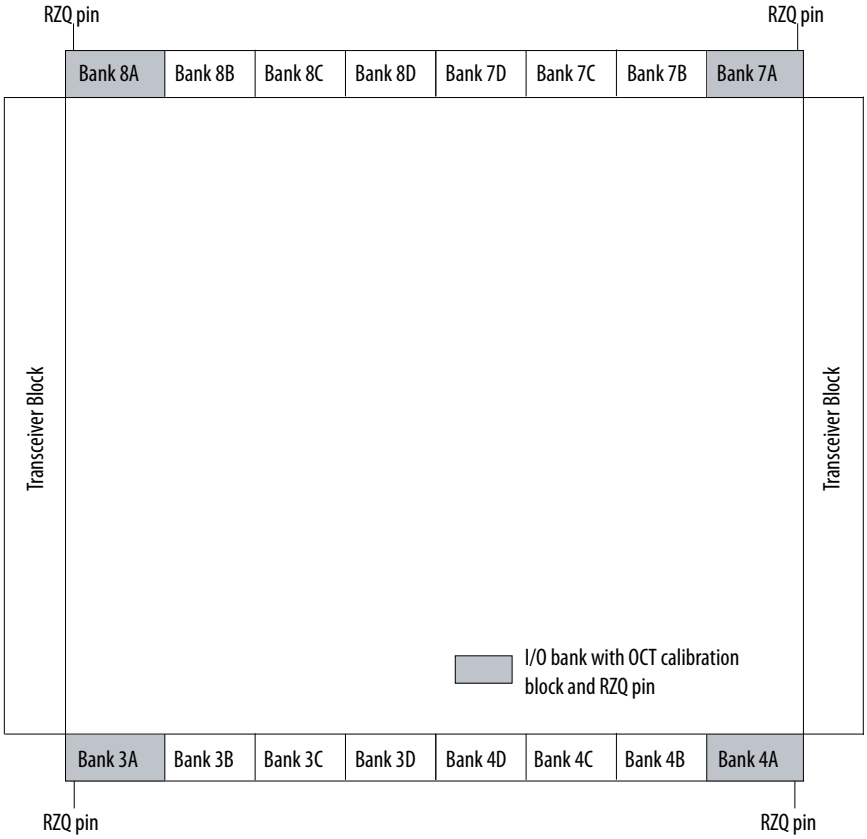
- Connect the RZQ pin to GND through an external 100 Ω or 240 Ω resistor (depending on the R_S or R_T OCT value).
- The RZQ pin shares the same V_{CCIO} supply voltage with the I/O bank where the pin is located.

Arria V devices support calibrated R_S and calibrated R_T OCT on all I/O pins except for dedicated configuration pins.

Calibration Block Locations in Arria V Devices

Figure 5-13: OCT Calibration Block and RZQ Pin Location

This figure shows the location of I/O banks with OCT calibration blocks and RZQ pins in the Arria V device. This figure represents the top view of the silicon die that corresponds to a reverse view of the device package and illustrates the highest density device in the device family.



Sharing an OCT Calibration Block on Multiple I/O Banks

An OCT calibration block has the same V_{CCIO} as the I/O bank that contains the block. All I/O banks with the same V_{CCIO} can share one OCT calibration block, even if that particular I/O bank has an OCT calibration block.

I/O banks that do not have calibration blocks share the calibration blocks in the I/O banks that have calibration blocks.

All I/O banks support OCT calibration with different V_{CCIO} voltage standards, up to the number of available OCT calibration blocks.

You can configure the I/O banks to receive calibration codes from any OCT calibration block with the same V_{CCIO}. If a group of I/O banks has the same V_{CCIO} voltage, you can use one OCT calibration block to calibrate the group of I/O banks placed around the periphery.

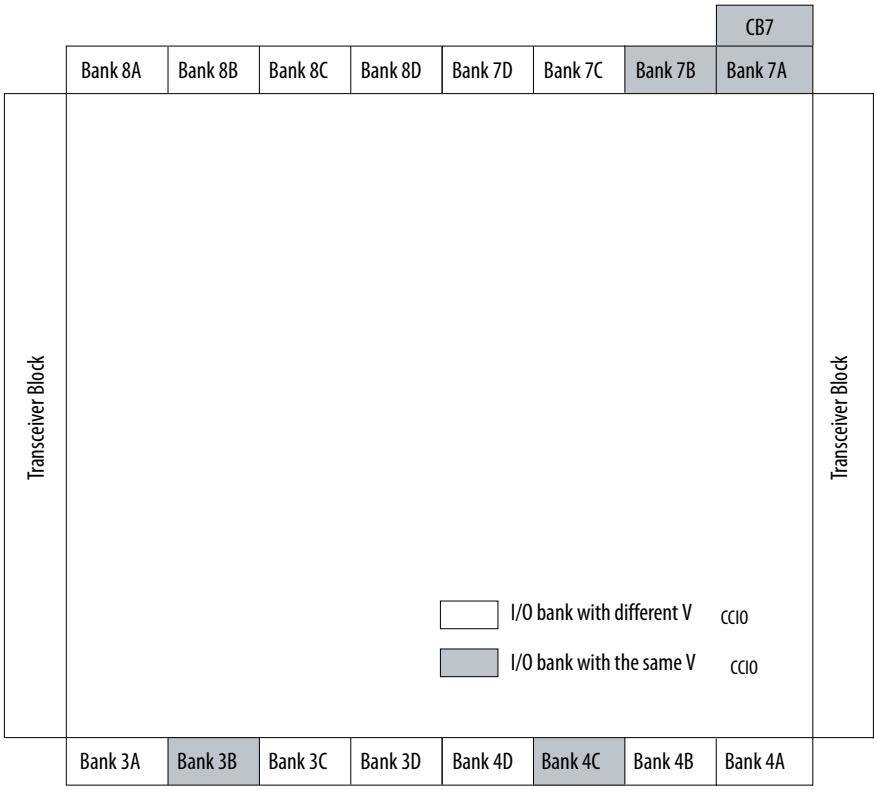
Related Information

- **OCT Calibration Block Sharing Example** on page 5-44
- **ALTOCT IP Core User Guide**
  Provides more information about the OCT calibration block.

OCT Calibration Block Sharing Example

Figure 5-14: Example of Calibrating Multiple I/O Banks with One Shared OCT Calibration Block

As an example, this figure shows a group of I/O banks that has the same V_{CCIO} voltage. The figure does not show transceiver calibration blocks. This figure represents the top view of the silicon die that corresponds to a reverse view of the device package and illustrates the highest density device in the device family.



Because banks 3B, 4C, and 7B have the same V_{CCIO} as bank 7A, you can calibrate all four I/O banks (3B, 4C, 7A, and 7B) with the OCT calibration block (CB7) located in bank 7A.

To enable this calibration, serially shift out the R_S OCT calibration codes from the OCT calibration block in bank 7A to the I/O banks around the periphery.

Related Information

- **Sharing an OCT Calibration Block on Multiple I/O Banks** on page 5-43
- **ALTOCT IP Core User Guide**
  Provides more information about the OCT calibration block.

External I/O Termination for Arria V Devices

Table 5-26: External Termination Schemes for Different I/O Standards

I/O Standard	External Termination Scheme
3.3 V LVTTL/3.3 V LVCMOS	No external termination required
3.0 V LVVTL/3.0 V LVCMOS	
3.0 V PCI	
3.0 V PCI-X	
2.5 V LVCMOS	
1.8 V LVCMOS	
1.5 V LVCMOS	
1.2 V LVCMOS	
SSTL-2 Class I	Single-Ended SSTL I/O Standard Termination
SSTL-2 Class II	
SSTL-18 Class I	
SSTL-18 Class II	
SSTL-15 Class I	
SSTL-15 Class II	
1.8 V HSTL Class I	Single-Ended HSTL I/O Standard Termination
1.8 V HSTL Class II	
1.5 V HSTL Class I	
1.5 V HSTL Class II	
1.2 V HSTL Class I	
1.2 V HSTL Class II	
Differential SSTL-2 Class I	Differential SSTL I/O Standard Termination
Differential SSTL-2 Class II	
Differential SSTL-18 Class I	
Differential SSTL-18 Class II	
Differential SSTL-15 Class I	
Differential SSTL-15 Class II	

I/O Standard	External Termination Scheme
Differential 1.8 V HSTL Class I	Differential HSTL I/O Standard Termination
Differential 1.8 V HSTL Class II	
Differential 1.5 V HSTL Class I	
Differential 1.5 V HSTL Class II	
Differential 1.2 V HSTL Class I	
Differential 1.2 V HSTL Class II	
LVDS	LVDS I/O Standard Termination
RSDS	RSDS/mini-LVDS I/O Standard Termination
Mini-LVDS	
LVPECL	Differential LVPECL I/O Standard Termination
SSTL-15 (21)	No external termination required
SSTL-135 (21)	
SSTL-125 (21)	
SSTL-12	
HSUL-12	
Differential SSTL-15 (21)	
Differential SSTL-135 (21)	
Differential SSTL-125 (21)	
Differential SSTL-12	
Differential HSUL-12	

Single-ended I/O Termination

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

The supported I/O standards such as SSTL-12, SSTL-125, SSTL-135, and SSTL-15 typically do not require external board termination.

Intel recommends that you use OCT with these I/O standards to save board space and cost. OCT reduces the number of external termination resistors used.

Note: You cannot use R_S and R_T OCT simultaneously. For more information, refer to the related information.

(21) Intel recommends that you use OCT with these I/O standards to save board space and cost. OCT reduces the number of external termination resistors used.

Figure 5-15: SSTL I/O Standard Termination

This figure shows the details of SSTL I/O termination on Arria V devices.

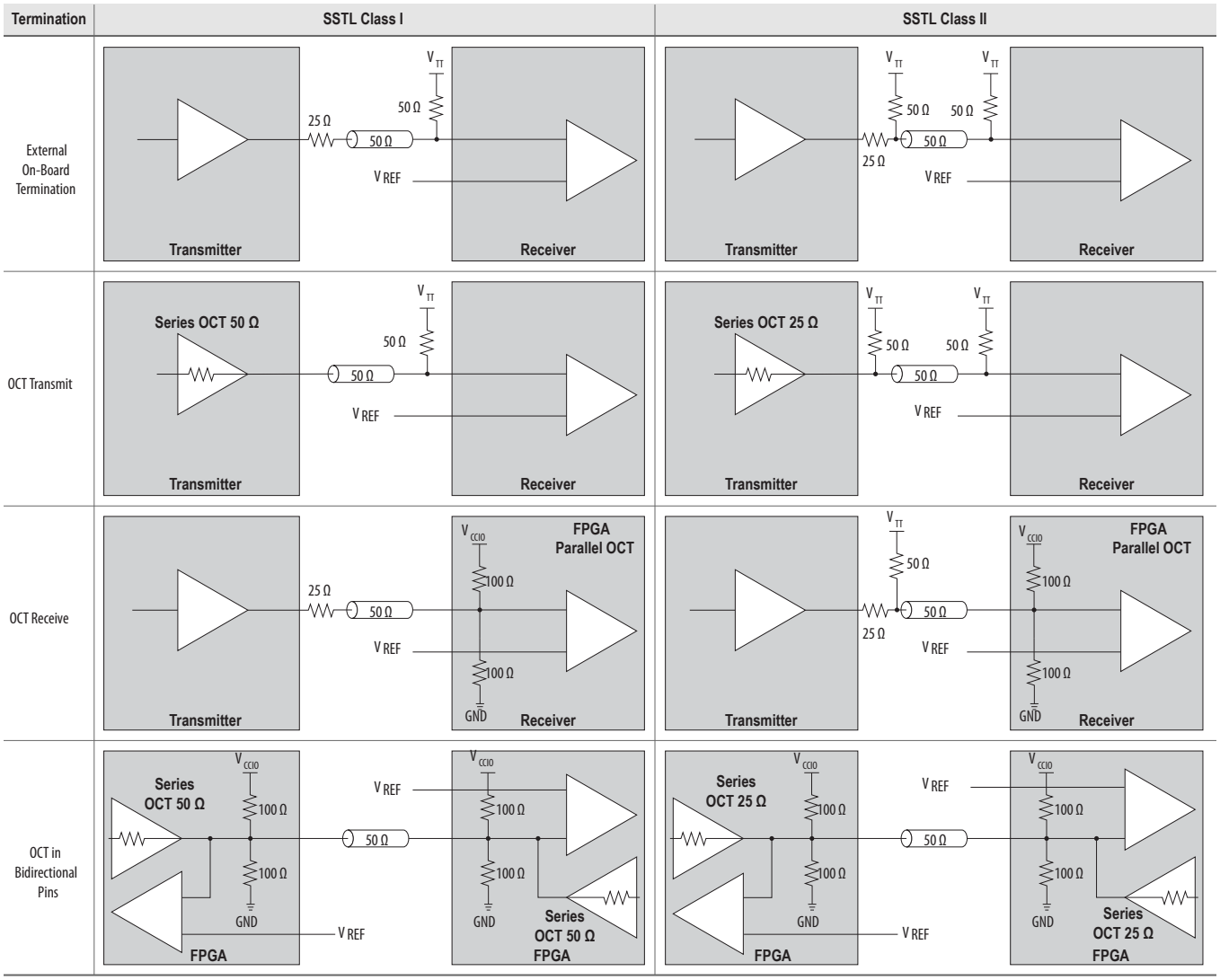
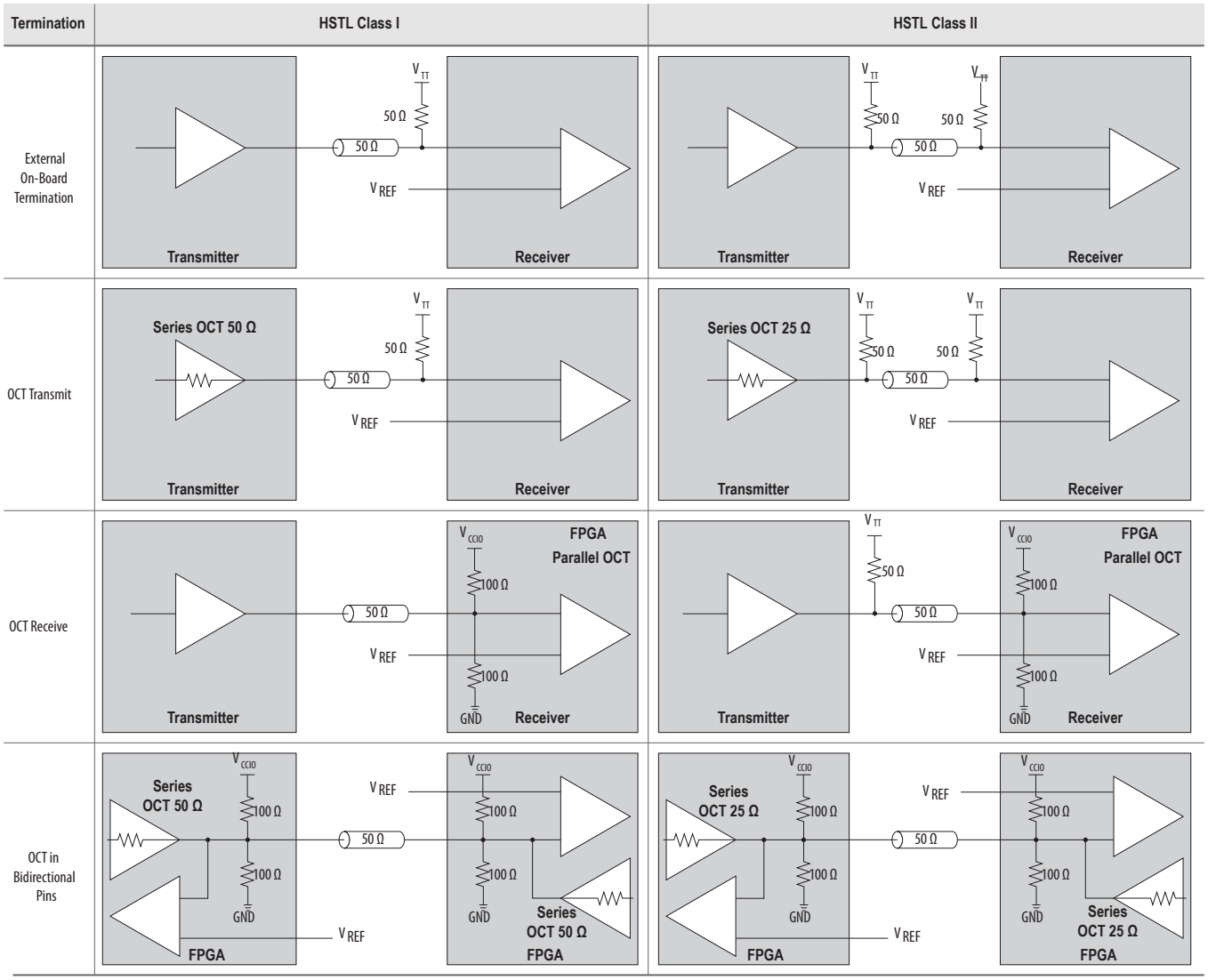


Figure 5-16: HSTL I/O Standard Termination

This figure shows the details of HSTL I/O termination on the Arria V devices.



Related Information

Dynamic OCT in Arria V Devices on page 5-40

Differential I/O Termination

The I/O pins are organized in pairs to support differential I/O standards. Each I/O pin pair can support differential input and output buffers.

The supported I/O standards such as Differential SSTL-12, Differential SSTL-15, Differential SSTL-125, and Differential SSTL-135 typically do not require external board termination.

Intel recommends that you use OCT with these I/O standards to save board space and cost. OCT reduces the number of external termination resistors used.

Differential HSTL, SSTL, and HSUL Termination

Differential HSTL, SSTL, and HSUL inputs use LVDS differential input buffers. However, R_D support is only available if the I/O standard is LVDS.

Differential HSTL, SSTL, and HSUL outputs are not true differential outputs. These I/O standards use two single-ended outputs with the second output programmed as inverted.

Figure 5-17: Differential SSTL I/O Standard Termination

This figure shows the details of Differential SSTL I/O termination on Arria V devices.

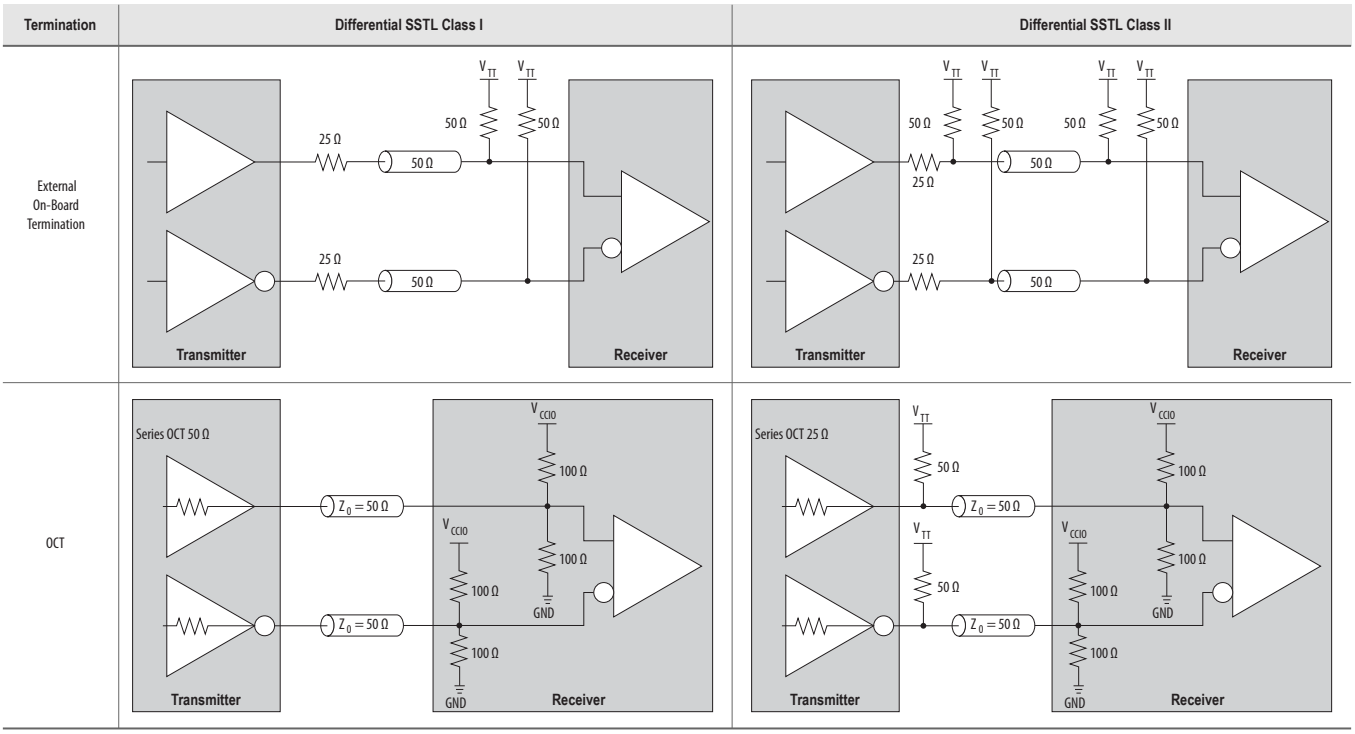
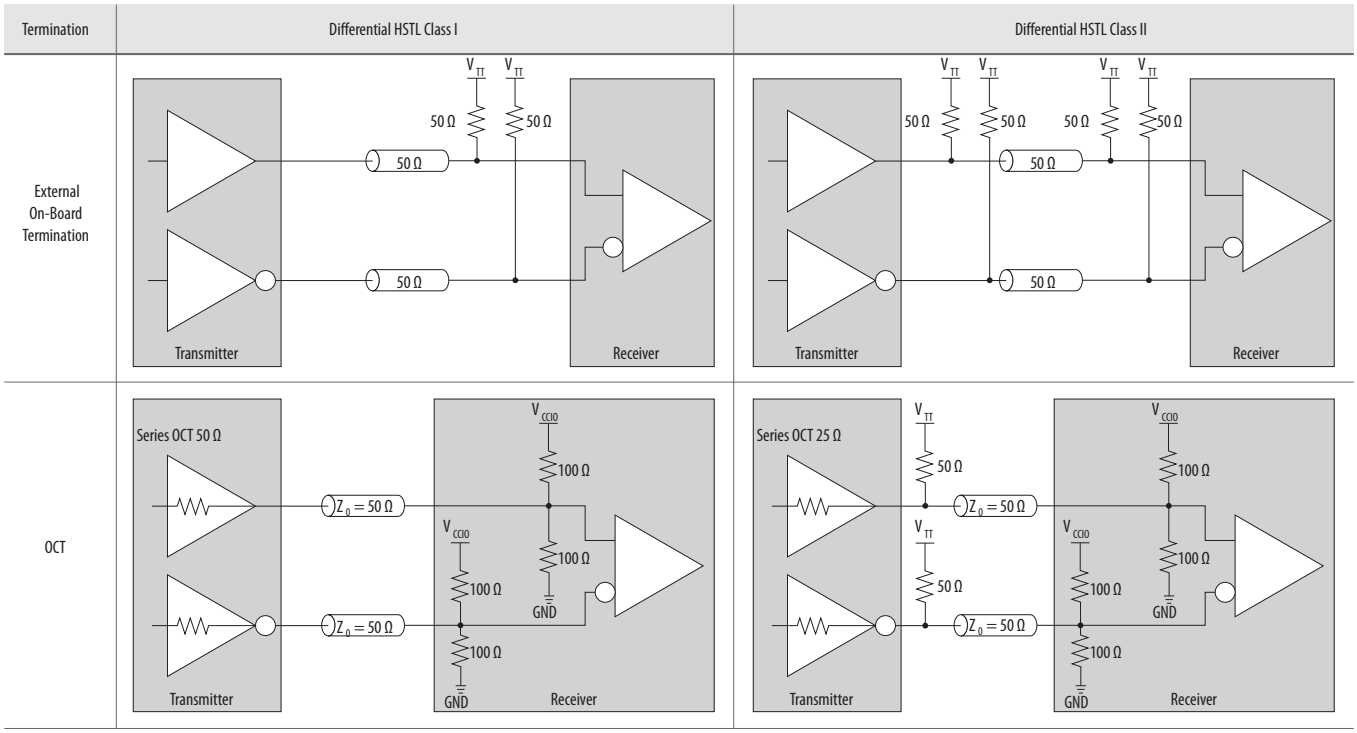


Figure 5-18: Differential HSTL I/O Standard Termination

This figure shows the details of Differential HSTL I/O standard termination on Arria V devices.

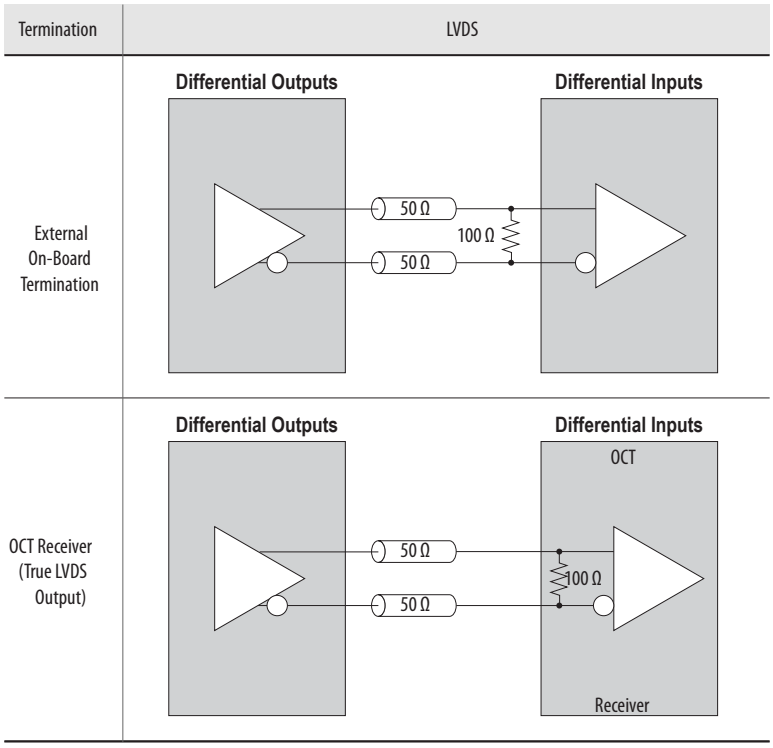


LVDS, RSDS, and Mini-LVDS Termination

All I/O banks have dedicated circuitry to support the true LVDS, RSDS, and mini-LVDS I/O standards by using true LVDS output buffers without resistor networks.

Figure 5-19: LVDS I/O Standard Termination

This figure shows the LVDS I/O standard termination. The on-chip differential resistor is available in all I/O banks.



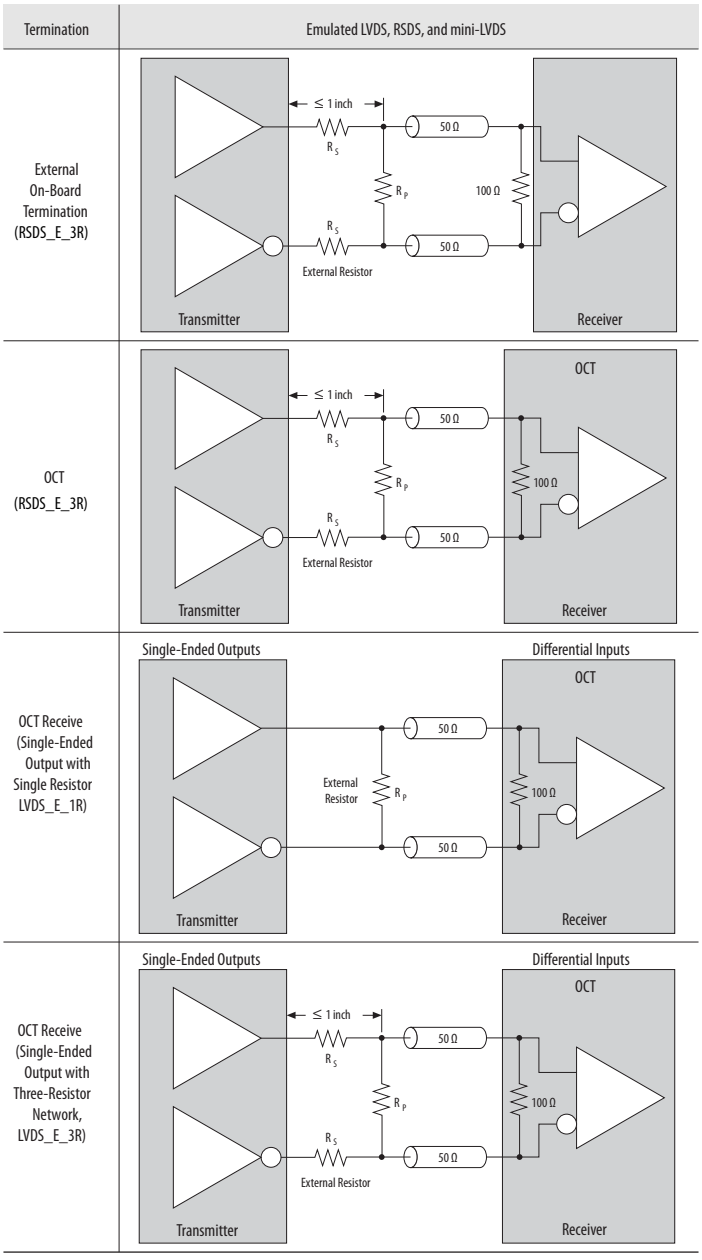
Emulated LVDS, RSDS, and Mini-LVDS Termination

The I/O banks also support emulated LVDS, RSDS, and mini-LVDS I/O standards.

Emulated LVDS, RSDS and mini-LVDS output buffers use two single-ended output buffers with an external single-resistor or three-resistor network, and can be tri-stated.

Figure 5-20: Emulated LVDS, RSDS, or Mini-LVDS I/O Standard Termination

The output buffers, as shown in this figure, are available in all I/O banks. R_S and R_P values are pending characterization.



To meet the RSDS or mini-LVDS specifications, you require a resistor network to attenuate the output-voltage swing.

You can modify the three-resistor network values to reduce power or improve the noise margin. Choose resistor values that satisfy the following equation.

Figure 5-21: Resistor Network Calculation

$$\frac{R_S \times \frac{R_P}{2}}{R_S + \frac{R_P}{2}} = 50\ \Omega$$

Note: Altera recommends that you perform additional simulations with IBIS or SPICE models to validate that the custom resistor values meet the RSDS or mini-LVDS I/O standard requirements.

For information about the data rates supported for external single resistor or three-resistor network, refer to the device datasheet.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**

LVPECL Termination

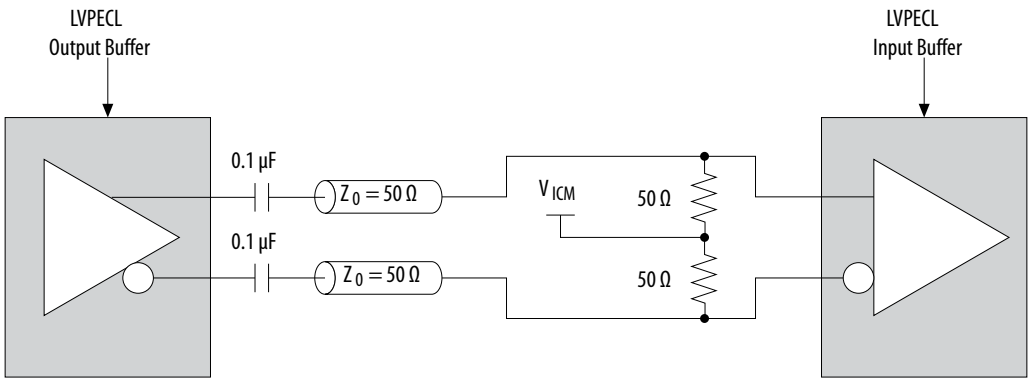
The Arria V devices support the LVPECL I/O standard on input clock pins only:

- LVPECL input operation is supported using LVDS input buffers.
- LVPECL output operation is not supported.

Use AC coupling if the LVPECL common-mode voltage of the output buffer does not match the LVPECL input common-mode voltage.

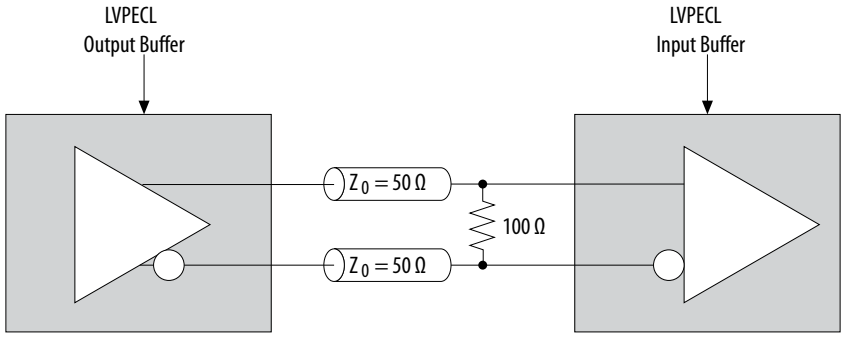
Note: Altera recommends that you use IBIS models to verify your LVPECL AC/DC-coupled termination.

Figure 5-22: LVPECL AC-Coupled Termination



Support for DC-coupled LVPECL is available if the LVPECL output common mode voltage is within the Arria V LVPECL input buffer specification.

Figure 5-23: LVPECL DC-Coupled Termination



For information about the V_{ICM} specification, refer to the device datasheet.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**

I/O Features in Arria V Devices Revision History

Document Version	Changes
2018.08.09	Updated the note in *Dynamic OCT in Arria V Devices* topic.

Date	Version	Changes
June 2016	2016.06.10	• Clarified the example quoted in **Non-Voltage-Referenced I/O Standards** can support 2.5 V, 3.0 V and 3.3 V inputs.
December 2015	2015.12.21	• Added assignment name and supported I/O standards in Summary of Supported Programmable IOE Features and Settings Table. • Changed instances of *Quartus II* to *Quartus Prime*.

Date	Version	Changes
January 215	2015.01.23	• Corrected truncated sentence in the note about the recommendation to use dynamic OCT for several I/O standards with DDR3 external memory interface. • Added pin placement guidelines for general purpose high-speed signals faster than 200 MHz. • Added 3.3V to Arria V I/O Standards Voltage Levels table for 3.3V LVTTL/3.3V LVCMOS, 3.0V LVTTL/3.0V LVCMOS and 2.5V LVCMOS I/O Standard. • Clarified that dedicated configuration pins, clock pins and JTAG pins do not support programmable pull-up resistor but these pins have fixed value of internal pull-up resistors. • Moved the Open-Drain Output, Bus-Hold Circuitry and Pull-up Resistor sections to Programmable IOE Features in Arria V Devices. • Update Open-Drain Output section with steps to enable open-drain output in Assignment Editor.
June 2014	2014.06.30	• Added footnote to clarify that some of the voltage levels listed in the MultiVolt I/O support table are for showing that multiple single-ended I/O standards are not compatible with certain V_{CCIO} voltages. • Updated the I/O banks locations figures to match the available modular I/O banks for Arria V GX, GT, SX, and ST devices. • Added information to clarify that programmable output slew rate is available for single-ended and emulated LVDS I/O standards. • Finalized calibrated R_S and R_T OCT values and updated the R_T OCT values for HSUL-12 and Differential HSUL-12 I/O standards. • Updated the V_{CCPD} guideline to clarify that bank 7A is not available as user I/O bank. In Arria V SX and ST devices, banks 6A, 6B, and 7A through 7E are allocated for the HPS.

Date	Version	Changes
January 2014	2014.01.10	• Updated statements in several topics to clarify that each modular I/O bank can support multiple I/O standards that use the same voltages. • Updated the guideline topic about using the same V_{CCPD} for I/O banks in the same V_{CCPD} group to improve clarity. • Added the optional PCI clamp diode to the figure showing the IOE structure. • Changed all "SoC FPGA" to "SoC". • Removed SSTL-125 from the list of supported I/O standards for the HPS I/O. • Removed all "preliminary" marks. • Removed the statement specifying that value "0" of the programmable V_{OD} is only available for RSDS and mini-LVDS I/O standards only. The value is now available for the LVDS I/O standards. • Clarified that you can only use R_D OCT if V_{CCPD} is 2.5 V. • Added link to Knowledge Base article that clarifies about vertical migration (drop-in compatibility). • Corrected the modular I/O banks tables for Arria V SX and ST devices. Bank 7G, available in the F1517 package, is an FPGA I/O bank instead of an HPS column I/O bank. The number of I/O pins remain the same.
August 2013	2013.08.19	Added 3.3 V input signal for 2.5 V V_{CCIO} in the table listing the MultiVolt I/O support.
June 2013	2013.06.21	• Removed 3.3 V input signal for 2.5 V V_{CCIO} in the table listing the MultiVolt I/O support. • Updated the topic about LVDS input R_D OCT to remove the requirement for setting the V_{CCIO} to 2.5 V. R_D OCT now requires only that the V_{CCPD} is 2.5 V. • Updated the topic about LVPECL termination to improve clarity.
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Added note about the power-up sequence requirement if you plan to migrate your design to devices that require the specific power-up sequence. • Updated the R_T OCT input termination settings for the 1.5 V SSTL I/O standards. • Updated the maximum speed of RSDS and mini-LVDS to 360 Mbps and 400 Mbps, respectively, in the notes for the supported FPGA I/O standards table.

Date	Version	Changes
December 2012	2012.12.04	• Added LVVTL and LVCMOS voltage levels for the Arria V GZ variant, and corrected the LVVTL and LVCMOS voltage levels for the Arria V GX, GT, SX, and ST devices. • Updated the SSTL and HSTL I/O termination figures to add VREF inputs for OCT in bidirectional pins.
November 2012	2012.11.19	• Reorganized content and updated template. • Added the I/O resources per package and I/O vertical migration sections for easy reference. • Added the steps to verify pin migration compatibility using the Quartus II software. • Updated the I/O standards support table with Arria V GZ and HPS I/O information. • Updated the guideline about mixing voltage-referenced and non-voltage-referenced I/O standards to include Arria V GZ information. • Updated the guideline about observing device absolute maximum rating for 3.3 V interfacing, specifically the off-chip clamping diode usage for Arria V GZ. • Updated the VREF pin restrictions guideline to specify that it applies only to Arria V GX, GT, SX, and ST, but not Arria V GZ. • Added the I/O bank locations for Arria V GZ devices. • Rearranged the I/O banks groups tables for easier reference. • Added modular I/O banks for Arria V GZ devices. • Restructured the information in the topic about I/O buffers and registers to improve clarity and for faster reference. • Added Arria V GZ and HPS information to the topic on programmable IOE features. • Rearranged the tables about on-chip I/O termination for clarity and topic-based reference. • Added Arria V GZ OCT information to all on-chip I/O termination tables. • Added all I/O standards and external termination schemes supported by all Arria V devices to the external I/O termination table.
June 2012	2.0	Updated for the Quartus II software v12.0 release: • Restructured chapter. • Added “Design Considerations”, “VCCIO Restriction”, “LVDS Channels”, “Modular I/O Banks”, and “OCT Calibration Block” sections. • Added Figure 5–1, Figure 5–2, and Figure 5–3 • Updated Table 5–1, Table 5–6, and Table 5–8. • Updated Figure 5–19 with emulated LVDS with external single resistor.

Date	Version	Changes
February 2012	1.2	Updated Table 5–3.
November 2011	1.1	• Restructured chapter. • Updated Table 5–3.
May 2011	1.0	Initial release.

High-Speed Differential I/O Interfaces and DPA in Arria V Devices

6

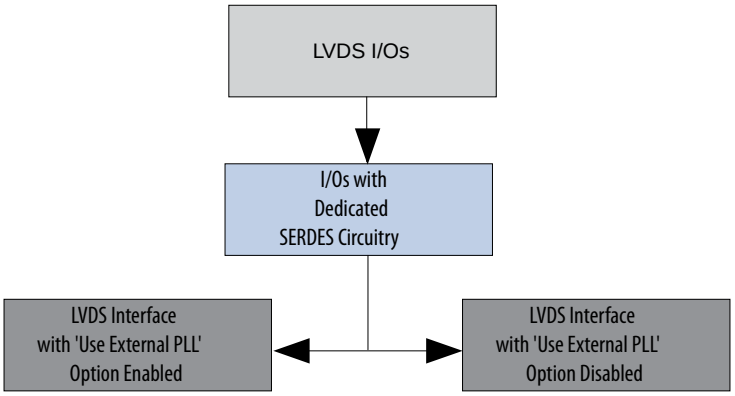
2020.04.13

AV-52006 Subscribe Send Feedback

The high-speed differential I/O interfaces and dynamic phase alignment (DPA) features in Arria V devices provide advantages over single-ended I/Os and contribute to the achievable overall system bandwidth. Arria V devices support low-voltage differential signaling (LVDS), mini-LVDS, and reduced swing differential signaling (RSDS) differential I/O standards.

The following figure shows the I/O bank support for high-speed differential I/O in the Arria V devices.

Figure 6-1: I/O Bank Support for High-Speed Differential I/O



Related Information

- **I/O Standards Support for FPGA I/O in Arria V Devices** on page 5-6
  Provides information about the supported differential I/O standards.
- **Arria V Device Handbook: Known Issues**
  Lists the planned updates to the *Arria V Device Handbook* chapters.

Dedicated High-Speed Circuitries in Arria V Devices

The following dedicated circuitries are available in the Arria V device family to support high-speed differential I/O:

- Differential I/O buffer
- Transmitter serializer
- Receiver deserializer
- Data realignment (Bit-slip)
- DPA
- Synchronizer (FIFO buffer)
- Phase-locked loops (PLLs)

SERDES and DPA Bank Locations in Arria V Devices

The dedicated serializer/deserializer (SERDES) and DPA circuitry that supports high-speed differential I/Os is located in the top and bottom banks of the Arria V devices.

The following figures show the high-level location of SERDES/DPA in the Arria V devices.

Figure 6-2: High-Speed Differential I/Os with DPA Locations in Arria V GX A1 and A3 Devices, and Arria V GT C3 Device.

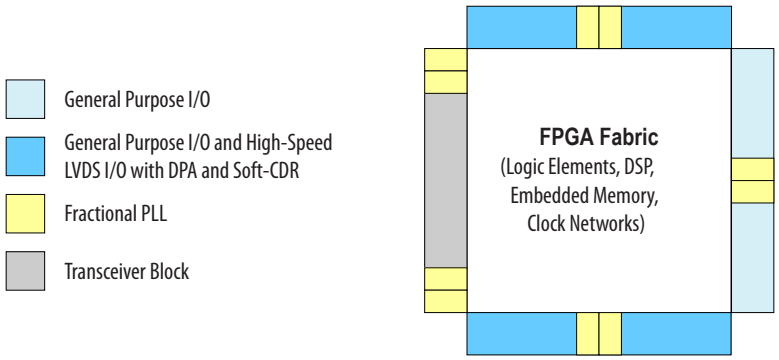


Figure 6-3: High-Speed Differential I/Os with DPA Locations in Arria V GX A5, A7, B1, and B3 Devices, and Arria V GT C7, D3, and D7 Devices

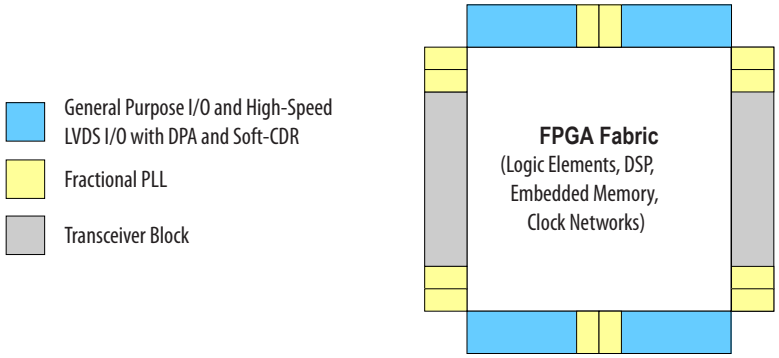


Figure 6-4: High-Speed Differential I/Os with DPA Locations in Arria V GX B5 and B7 Devices

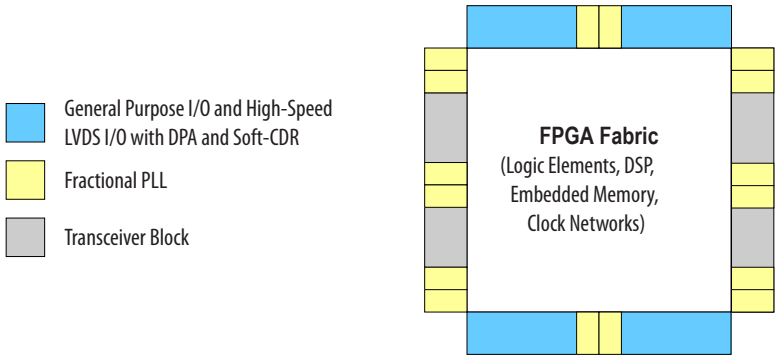
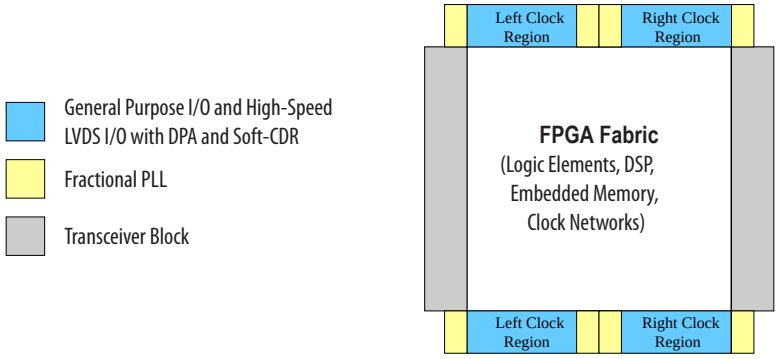


Figure 6-5: High-Speed Differential I/Os with DPA Locations in Arria V GZ Devices



Related Information

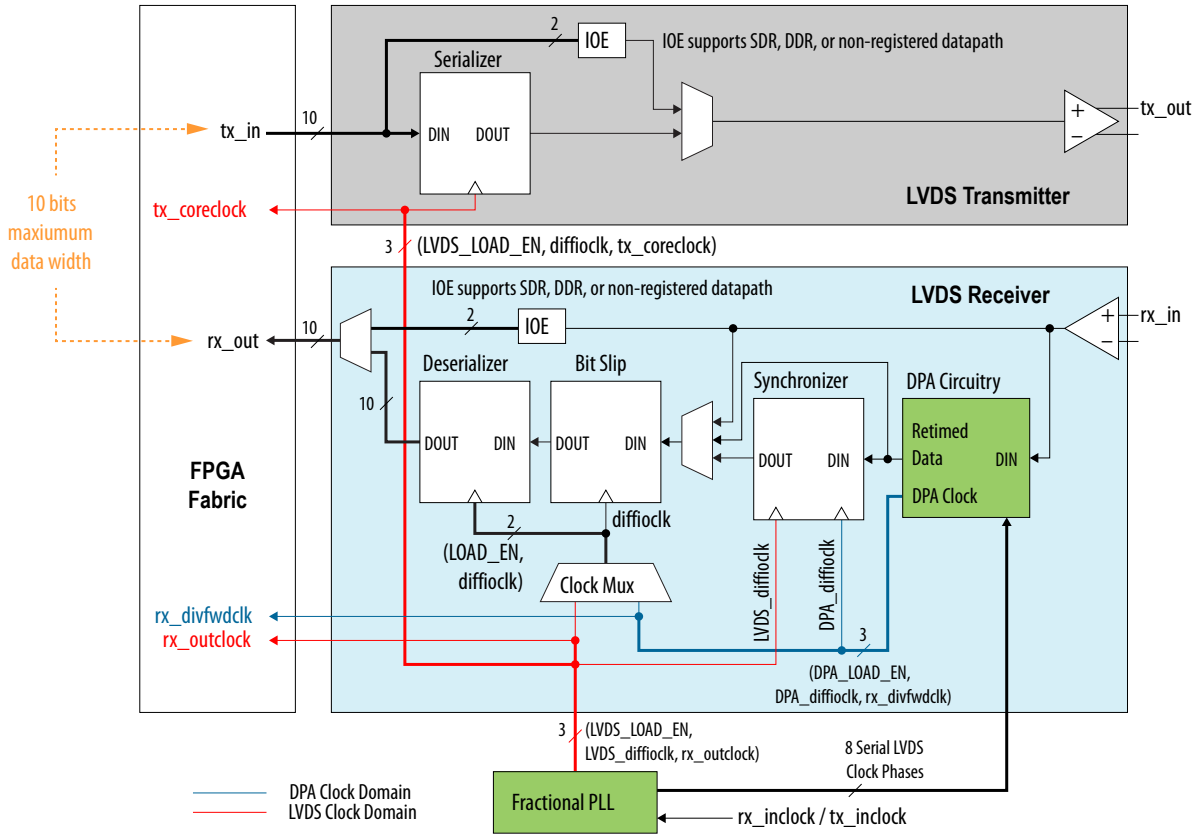
PLLs and Clocking for Arria V Devices on page 6-7

LVDS SERDES Circuitry

The Arria V devices have built-in serializer/deserializer (SERDES) circuitry that supports high-speed LVDS interfaces. You can configure the SERDES circuitry to support source-synchronous communication protocols such as RapidIO®, XSBI, serial peripheral interface (SPI), and asynchronous protocols such as Gigabit Ethernet (GbE).

The following figure shows a transmitter and receiver block diagram for the LVDS SERDES circuitry with the interface signals of the transmitter and receiver data paths.

Figure 6-6: LVDS SERDES



The preceding figure shows a shared PLL between the transmitter and receiver. If the transmitter and receiver do not share the same PLL, you require two fractional PLLs. In single data rate (SDR) and double data rate (DDR) modes, the data width is 1 and 2 bits, respectively.

The ALTLVDS transmitter and receiver requires various clock and load enable signals from a fractional PLL. The Intel Quartus Prime software configures the PLL settings automatically. The software is also responsible for generating the various clock and load enable signals based on the input reference clock and selected data rate.

Note: For the maximum data rate supported by the Arria V devices, refer to the device overview.

Related Information

- Arria V Device Overview

- **LVDS SERDES Transmitter/Receiver IP Cores User Guide**
  Provides a list of the LVDS transmitter and receiver ports and settings using ALTLVDS.
- **Guideline: Use PLLs in Integer PLL Mode for LVDS** on page 6-8

True LVDS Buffers in Arria V Devices

The following tables list the number of true LVDS buffers supported in Arria V devices with these conditions:

- The LVDS channel count does not include dedicated clock pins.
- Dedicated SERDES and DPA is available for top and bottom banks only.
- Each I/O sub-bank can support up to two independent ALTLVDS interfaces. For example, you can place two ALTLVDS interfaces in bank 8A driven by two different PLLs, provided that the LVDS channels are not interleaved.

Table 6-1: LVDS Channels Supported in Arria V GX Devices

Member Code	Package	Side	TX	RX
A1 and A3	672-pin FineLine BGA, Flip Chip	Top	28	34
		Bottom	29	34
	896-pin FineLine BGA, Flip Chip	Top	33	40
		Bottom	34	40
A5 and A7	672-pin FineLine BGA, Flip Chip	Top	34	44
		Bottom	34	44
	896-pin FineLine BGA, Flip Chip	Top	42	48
		Bottom	42	48
	1152-pin FineLine BGA, Flip Chip	Top	60	68
		Bottom	60	68
B1 and B3	896-pin FineLine BGA, Flip Chip	Top	42	48
		Bottom	42	48
	1152-pin FineLine BGA, Flip Chip	Top	60	68
		Bottom	60	68
	1517-pin FineLine BGA, Flip Chip	Top	80	88
		Bottom	80	88
B5 and B7	1152-pin FineLine BGA, Flip Chip	Top	60	68
		Bottom	60	68
	1517-pin FineLine BGA, Flip Chip	Top	80	88
		Bottom	80	88

Table 6-2: LVDS Channels Supported in Arria V GT Devices

Member Code	Package	Side	TX	RX
C3	672-pin FineLine BGA, Flip Chip	Top	26	34
		Bottom	26	34
	896-pin FineLine BGA, Flip Chip	Top	34	40
		Bottom	34	40
C7	896-pin FineLine BGA, Flip Chip	Top	42	48
		Bottom	42	48
	1152-pin FineLine BGA, Flip Chip	Top	60	68
		Bottom	60	68
D3	896-pin FineLine BGA, Flip Chip	Top	42	48
		Bottom	42	48
	1152-pin FineLine BGA, Flip Chip	Top	60	68
		Bottom	60	68
	1517-pin FineLine BGA, Flip Chip	Top	80	88
		Bottom	80	88
D7	1152-pin FineLine BGA, Flip Chip	Top	60	68
		Bottom	60	68
	1517-pin FineLine BGA, Flip Chip	Top	80	88
		Bottom	80	88

Table 6-3: LVDS Channels Supported in Arria V GZ Devices

Member Code	Package	Side	TX	RX
E1 and E3	780-pin FineLine BGA, Flip Chip	Top	42	51
		Bottom	39	39
	1152-pin FineLine BGA, Flip Chip	Top	48	57
		Bottom	51	51
E5 and E7	1152-pin FineLine BGA, Flip Chip	Top	54	63
		Bottom	75	75
	1517-pin FineLine BGA, Flip Chip	Top	79	81
		Bottom	87	87

Table 6-4: LVDS Channels Supported in Arria V SX Devices

Member Code	Package	Side	TX	RX
B3 and B5	896-pin FineLine BGA, Flip Chip	Top	14	37
		Bottom	20	37
	1152-pin FineLine BGA, Flip Chip	Top	14	37
		Bottom	30	77
	1517-pin FineLine BGA, Flip Chip	Top	40	48
		Bottom	80	88

Table 6-5: LVDS Channels Supported in Arria V ST Devices

Member Code	Package	Side	TX	RX
D3 and D5	896-pin FineLine BGA, Flip Chip	Top	14	37
		Bottom	20	37
	1152-pin FineLine BGA, Flip Chip	Top	14	37
		Bottom	30	77
	1517-pin FineLine BGA, Flip Chip	Top	40	48
		Bottom	80	88

Emulated LVDS Buffers in Arria V Devices

The Arria V device family supports emulated LVDS:

- You can use unutilized true LVDS input channels as emulated LVDS output buffers (eTX) for serialization factors of 1 and 2.
- The emulated differential output buffers support tri-state capability.

High-Speed I/O Design Guidelines for Arria V Devices

There are several considerations that require your attention to ensure the success of your designs. Unless noted otherwise, these design guidelines apply to all variants of this device family.

PLLs and Clocking for Arria V Devices

To generate the parallel clocks (`rx_outclock` and `tx_outclock`) and high-speed clocks (`diffioclk`), the Arria V devices provide fractional PLLs in the high-speed differential I/O receiver and transmitter channels.

Related Information

- **Guideline: Use PLLs in Integer PLL Mode for LVDS** on page 6-8
- **SERDES and DPA Bank Locations in Arria V Devices** on page 6-2
  Provides information about the PLL locations available for each Arria V device.
- **Guideline: Use High-Speed Clock from PLL to Clock LVDS SERDES Only** on page 6-8

Guideline: Use PLLs in Integer PLL Mode for LVDS

To drive the LVDS channels, you must use the PLLs in integer PLL mode. The center or corner PLLs can drive the LVDS receiver and transmitter channels.

However, in Arria V GZ devices, the clock tree network cannot cross over to different I/O regions. For example, the top left corner PLL cannot cross over to drive the LVDS receiver and transmitter channels on the top right I/O bank.

Related Information

Pin Placement Guidelines for DPA and Non-DPA Differential Channels on page 6-13
Provides more information about the fractional PLL clocking restrictions.

Guideline: Use High-Speed Clock from PLL to Clock LVDS SERDES Only

The high-speed clock generated from the PLL is intended to clock the LVDS SERDES circuitry only. Do not use the high-speed clock to drive other logic because the allowed frequency to drive the core logic is restricted by the PLL F_{OUT} specification.

For more information about the F_{OUT} specification, refer to the device datasheet.

Note: Spread-spectrum input clock is not supported in LVDS.

LVDS Interface with External PLL Mode

The IP Catalog provides an option for implementing the LVDS interface with the **Use External PLL** option. With this option enabled you can control the PLL settings, such as dynamically reconfiguring the PLL to support different data rates, dynamic phase shift, and other settings. You must also instantiate the an Altera_PLL IP core to generate the various clock and load enable signals.

If you enable the **Use External PLL** option with the ALTLVDS transmitter and receiver, the following signals are required from the Altera_PLL IP core:

- Serial clock input to the SERDES of the ALTLVDS transmitter and receiver
- Load enable to the SERDES of the ALTLVDS transmitter and receiver
- Parallel clock used to clock the transmitter FPGA fabric logic and parallel clock used for the receiver
- Asynchronous PLL reset port of the ALTLVDS receiver

Altera_PLL Signal Interface with ALTLVDS IP Core

Table 6-6: Signal Interface Between Altera_PLL and ALTLVDS IP Cores

This table lists the signal interface between the output ports of the Altera_PLL IP core and the input ports of the ALTLVDS transmitter and receiver. As an example, the table lists the serial clock output, load enable output, and parallel clock output generated on ports outclk0, outclk1, and outclk2, along with the locked signal of the Altera_PLL instance. You can choose any of the PLL output clock ports to generate the interface clocks.

From the Altera_PLL IP Core	To the ALTLVDS Transmitter	To the ALTLVDS Receiver
Serial clock output (outclk0) The serial clock output (outclk0) can only drive `tx_inclock` on the ALTLVDS transmitter, and `rx_inclock` and `rx_dpaclock` on the ALTLVDS receiver. This clock cannot drive the core logic.	`tx_inclock` (serial clock input to the transmitter)	`rx_inclock` (serial clock input) `rx_dpaclock`
Load enable output (outclk1)	`tx_enable` (load enable to the transmitter)	`rx_enable` (load enable for the deserializer)
Parallel clock output (outclk2)	Parallel clock used inside the transmitter core logic in the FPGA fabric	`rx_syncclock` (parallel clock input) and parallel clock used inside the receiver core logic in the FPGA fabric
~(locked)	—	`pll_areset` (asynchronous PLL reset port) The `pll_areset` signal is automatically enabled for the LVDS receiver in external PLL mode. This signal does not exist for LVDS transmitter instantiation when the external PLL option is enabled.

Note: With soft SERDES, a different clocking requirement is needed.

Related Information

LVDS SERDES Transmitter/Receiver IP Cores User Guide
More information about the different clocking requirement for soft SERDES.

Altera_PLL Parameter Values for External PLL Mode

The following examples show the clocking requirements to generate output clocks for ALTLVDS_TX and ALTLVDS_RX using the Altera_PLL IP core. The examples set the phase shift with the assumption that the clock and data are edge aligned at the pins of the device.

Note: For other clock and data phase relationships, Altera recommends that you first instantiate your ALTLVDS_RX and ALTLVDS_TX interface without using the external PLL mode option. Compile the IP cores in the Intel Quartus Prime software and take note of the frequency, phase shift, and duty cycle settings for each clock output. Enter these settings in the Altera_PLL IP core parameter editor and then connect the appropriate output to the ALTLVDS_RX and ALTLVDS_TX IP cores.

Table 6-7: Example: Generating Output Clocks Using an Altera_PLL IP Core (No DPA and Soft-CDR Mode)

This table lists the parameter values that you can set in the Altera_PLL parameter editor to generate three output clocks using an Altera_PLL IP core if you are not using DPA and soft-CDR mode.

Parameter	outclk0 (Connects to the tx_inclock port of ALTLVDS_TX and the rx_inclock port of ALTLVDS_RX)	outclk1 (Connects to the tx_enable port of ALTLVDS_TX and the rx_enable port of ALTLVDS_RX)	outclk2 (Used as the core clock for the parallel data registers for both transmitter and receiver, and connects to the rx_syncclock port of ALTLVDS_RX)
Frequency	data rate	data rate/serialization factor	data rate/serialization factor
Phase shift	–180°	[(deserialization factor – 2)/ deserialization factor] x 360°	–180/serialization factor (outclk0 phase shift divided by the serialization factor)
Duty cycle	50%	100/serialization factor	50%

Figure 6-7: Phase Relationship for External PLL Interface Signals

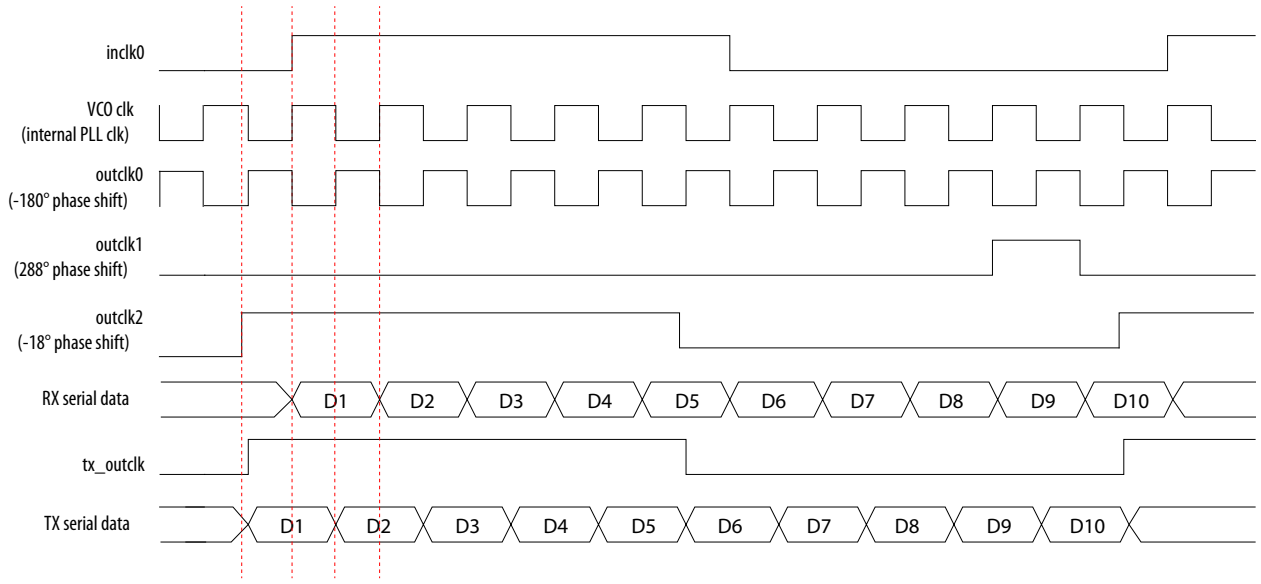


Table 6-8: Example: Generating Output Clocks Using an Altera_PLL IP Core (With DPA and Soft-CDR Mode)

This table lists the parameter values that you can set in the Altera_PLL parameter editor to generate four output clocks using an Altera_PLL IP core if you are using DPA and soft-CDR mode. The `locked` output port of Altera_PLL must be inverted and connected to the `pll_areset` port of the ALTLVDS_RX IP core if you are using DPA and soft-CDR mode.

Parameter	outclk0 (Connects to the tx_inclock port of ALTLVDS_TX and the rx_inclock port of ALTLVDS_RX)	outclk1 (Connects to the tx_enable port of ALTLVDS_TX and the rx_enable port of ALTLVDS_RX)	outclk2 (Used as the core clock for the parallel data registers for both transmitter and receiver, and connects to the rx_syncclock port of ALTLVDS_RX)	outclk3 (Connects to the rx_dpaclock port of ALTLVDS_RX)
Frequency	data rate	data rate/serialization factor	data rate/serialization factor	data rate
Phase shift	–180°	[(deserialization factor - 2)/deserialization factor] x 360°	–180/serialization factor (outclk0 phase shift divided by the serialization factor)	–180°
Duty cycle	50%	100/serialization factor	50%	50%

Connection between Altera_PLL and ALTLVDS

Figure 6-8: LVDS Interface with the Altera_PLL IP Core (Without DPA and Soft-CDR Mode)

This figure shows the connections between the Altera_PLL and ALTLVDS IP core if you are not using DPA and soft-CDR mode.

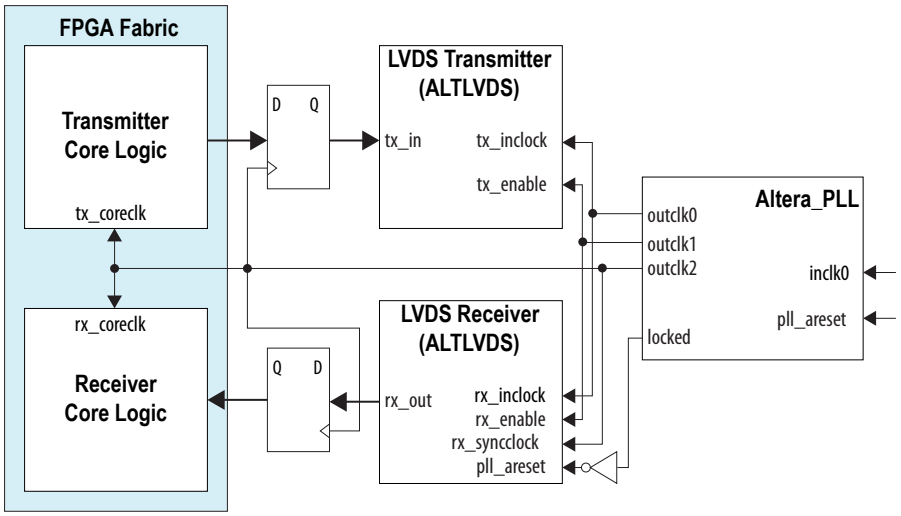


Figure 6-9: LVDS Interface with the Altera_PLL IP Core (With DPA)

This figure shows the connections between the Altera_PLL and ALTLVDS IP core if you are using DPA. The `locked` output port must be inverted and connected to the `pll_areset` port.

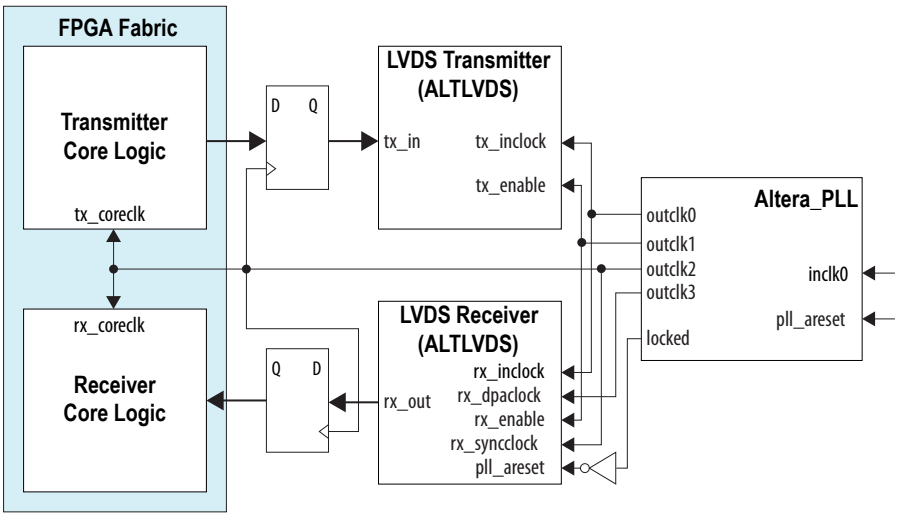
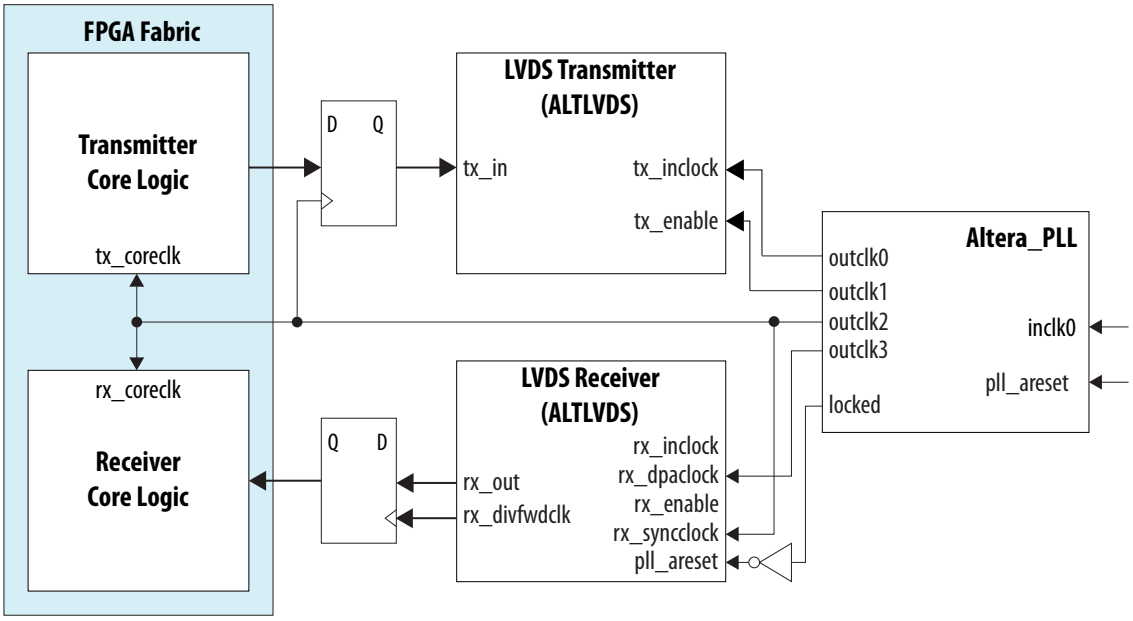


Figure 6-10: LVDS Interface with the Altera_PLL IP Core (With Soft-CDR Mode)

This figure shows the connections between the Altera_PLL and ALTLVDS IP core if you are using soft-CDR mode. The `locked` output port must be inverted and connected to the `pll_areset` port.



When generating the Altera_PLL IP core, the **Left/Right PLL** option is configured to set up the PLL in LVDS mode. Instantiation of `pll_areset` is optional.

The `rx_enable` and `rx_inclock` input ports are not used and can be left unconnected.

Pin Placement Guidelines for DPA and Non-DPA Differential Channels

DPA usage adds some constraints on the placement of high-speed differential channels. If DPA-enabled or DPA-disabled differential channels[22] in the differential banks are used, you must adhere to the differential pin placement guidelines to ensure the proper high-speed operation. The Intel Quartus Prime compiler automatically checks the design and issues an error message if the guidelines are not followed.

Note: The figures in this section show guidelines for using corner and center PLLs but do not necessarily represent the exact locations of the high-speed LVDS I/O banks.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Guideline: Using DPA-Enabled Differential Channels

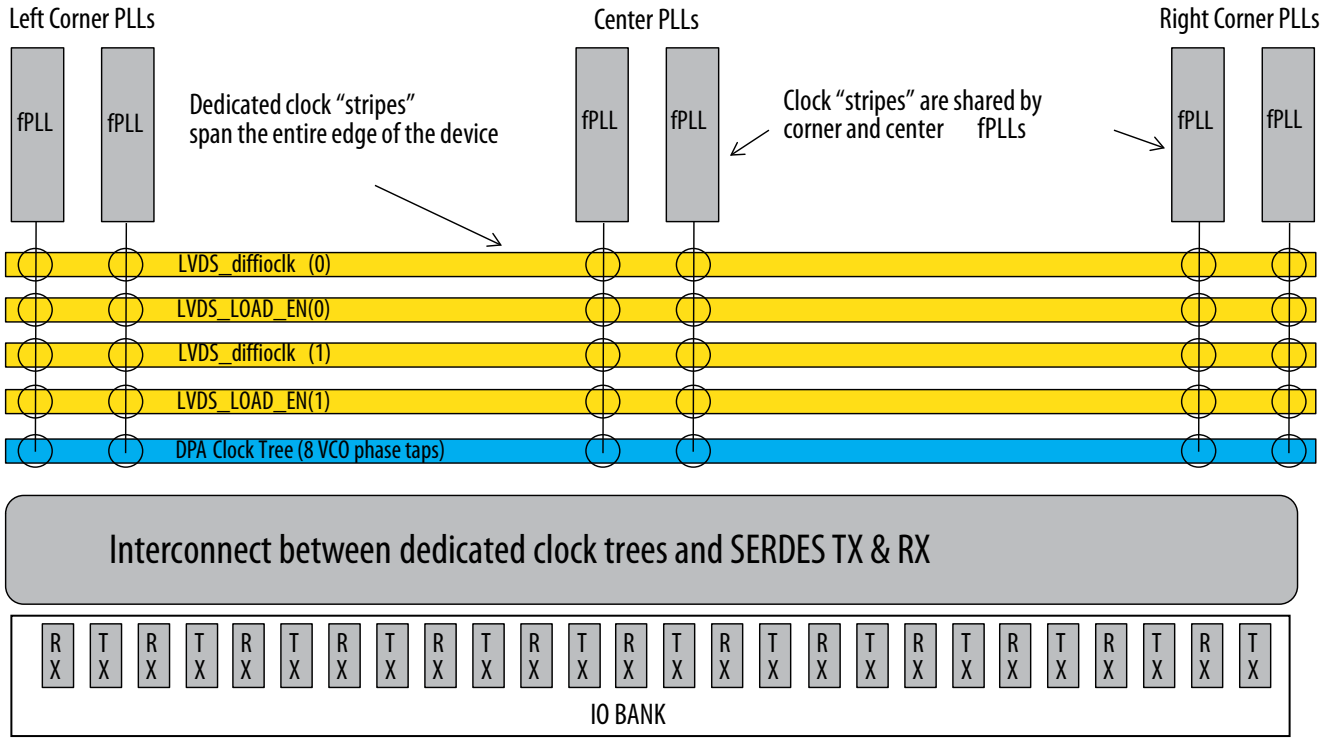
Each differential receiver in an I/O block has a dedicated DPA circuit to align the phase of the clock to the data phase of its associated channel. If you enable a DPA channel in a bank, you can use both single-ended I/Os and differential I/O standards in the bank.

You can place double data rate I/O (DDIO) output pins within I/O modules that have the same pad group number as a SERDES differential channel. However, you cannot place SDR I/O output pins within I/O modules that have the same pad group number as a receiver SERDES differential channel. You must implement the input register within the FPGA fabric logic.

The following figure illustrates the clock network for DPA and SERDES resources in Arria V devices.

[22] DPA-enabled differential channels refer to DPA mode or soft-CDR mode while DPA disabled channels refer to non-DPA mode.

Figure 6-11: LVDS and DPA Clock Network

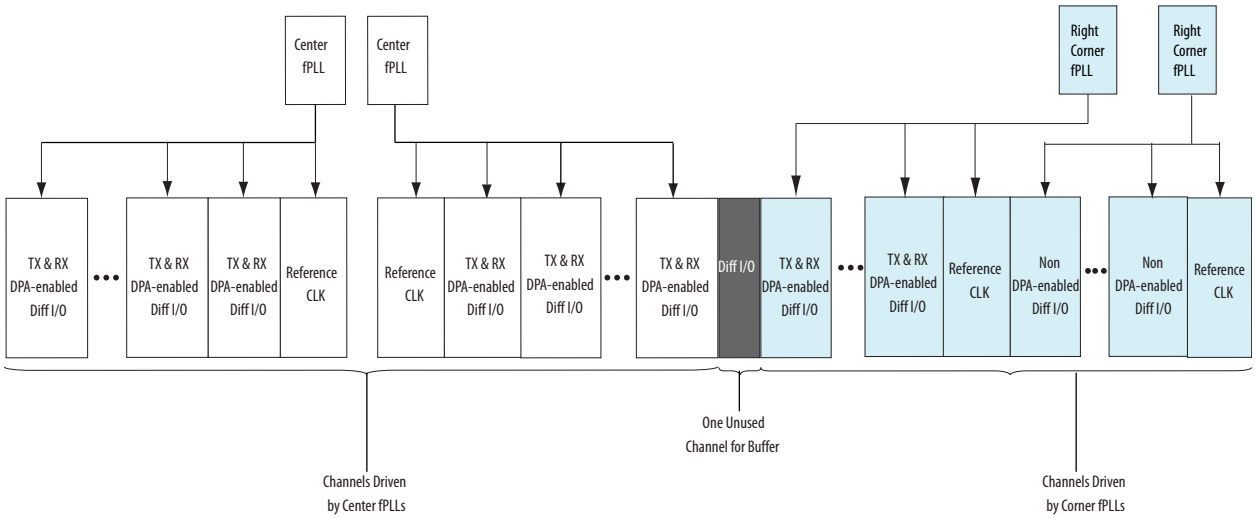


If you use DPA-enabled channels in differential banks, adhere to the following guidelines.

Using Center and Corner PLLs

If two PLLs drive the DPA-enabled channels in a bank—the corner and center PLL drive one group each—there must be at least one row (one differential channel) of separation between the two groups of DPA-enabled channels, as shown in the following figure.

Figure 6-12: Center and Corner PLLs Driving DPA-enabled Differential I/Os in the Same Bank



This separation prevents noise mixing because the two groups can operate at independent frequencies. No separation is necessary if a single PLL is driving both the DPA-enabled channels and DPA-disabled channels.

Using Both Center PLLs

You can use center PLLs to drive DPA-enabled channels simultaneously, if they drive these channels in their adjacent banks only, as shown in the previous figure. If one of the center PLLs drives the DPA-enabled channels in the left and right I/O banks in Arria V GX, GT, SX, or ST devices, you cannot use the other center PLL for DPA-enabled channels. If the center left PLL drives the DPA-enabled channels in the right I/O bank, the right center PLL cannot drive the DPA-enabled channels in the left I/O bank, and vice versa. The center PLLs cannot drive cross-banks simultaneously in Arria V GZ devices. Refer to the following figures.

Figure 6-13: Center PLLs Driving DPA-enabled Differential I/Os in Arria V GX, GT, SX, and ST Devices

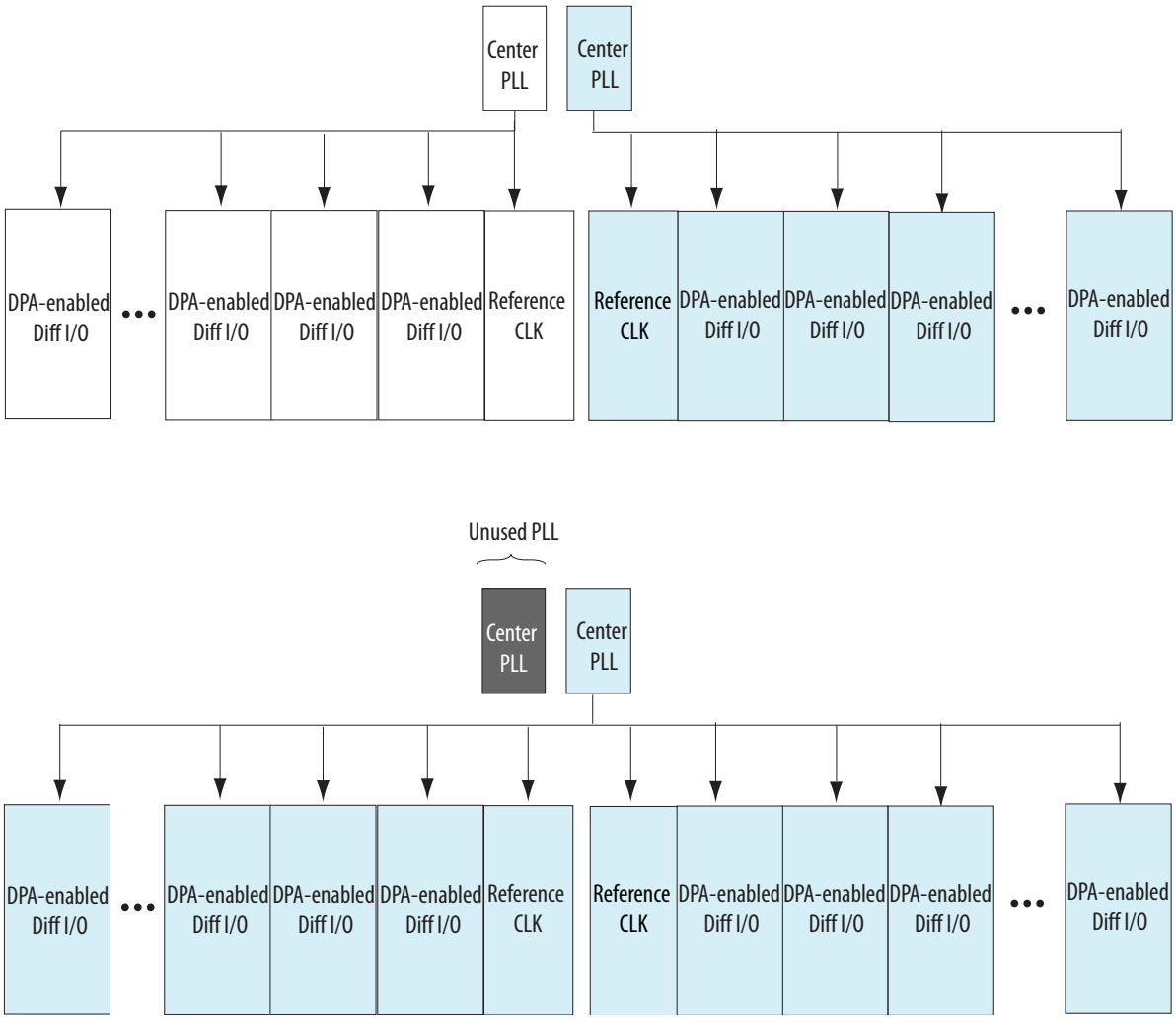


Figure 6-14: Center PLLs Driving DPA-enabled Differential I/Os in Arria V GZ Devices

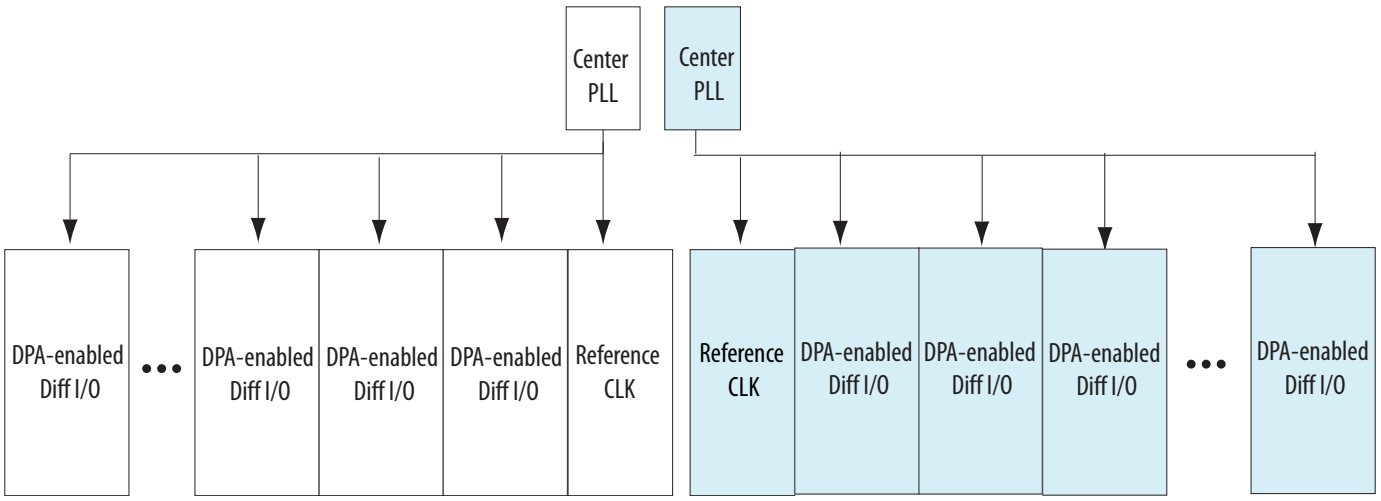
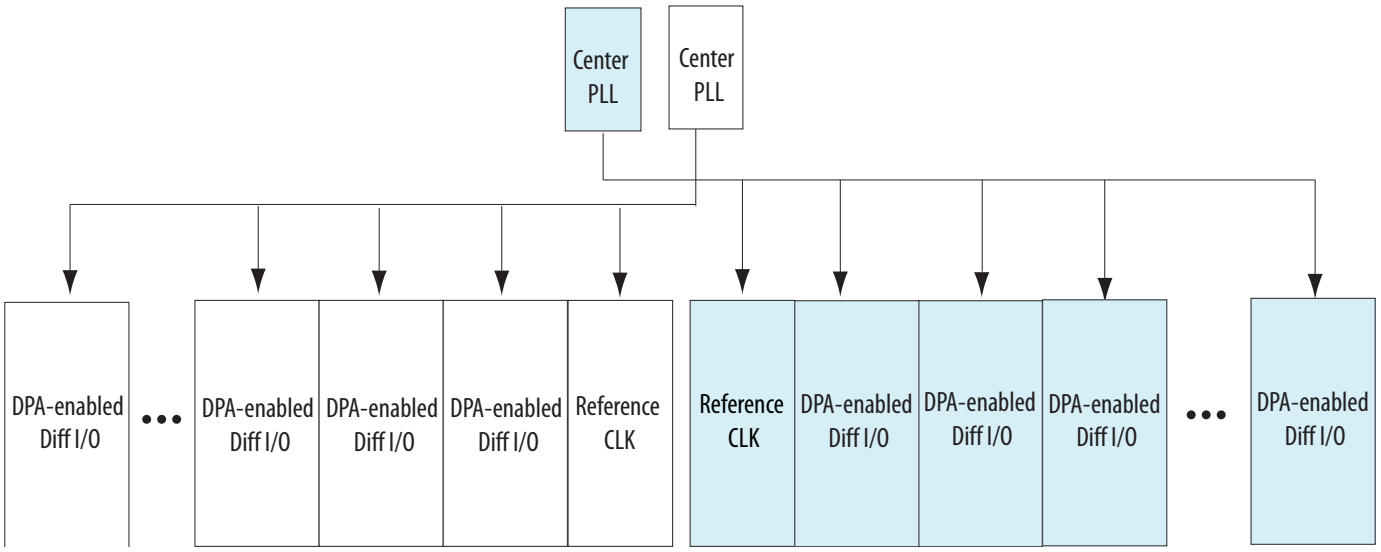


Figure 6-15: Invalid Placement of DPA-enabled Differential I/Os Driven by Both Center PLLs

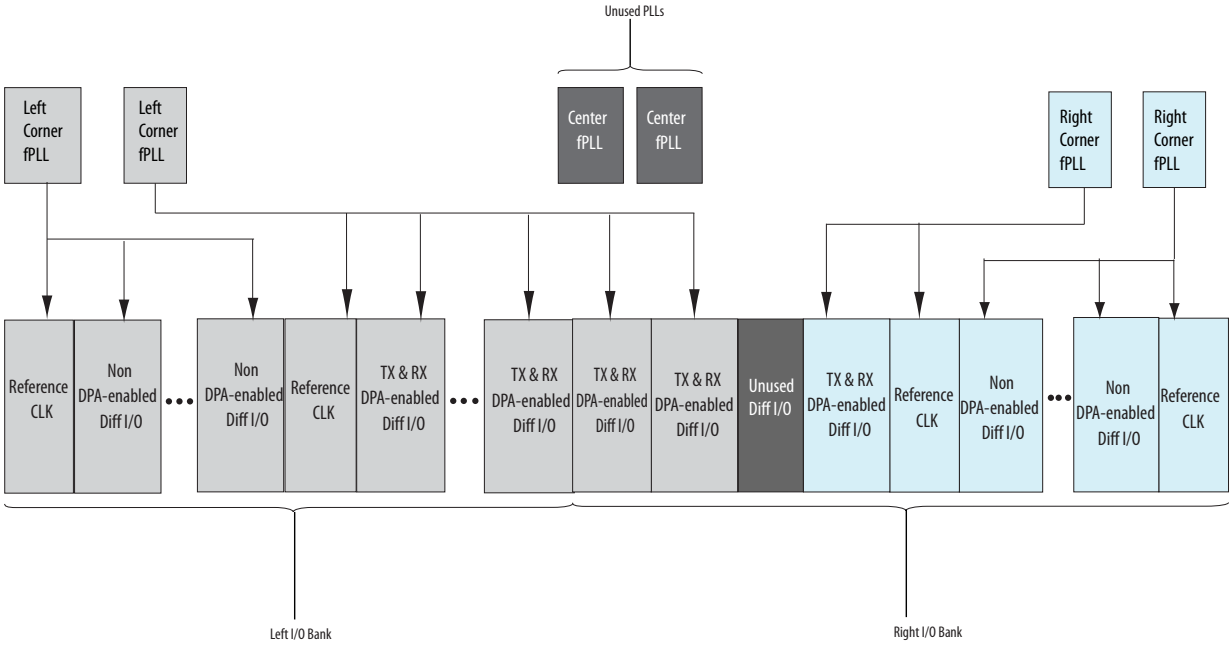


Using Both Corner PLLs

You can use the left and right corner PLLs to drive DPA-enabled channels simultaneously, if they drive the channels in their adjacent banks only. There must be at least one row of separation between the two groups of DPA-enabled channels.

There are two PLL in each corner of the device. However, only one corner PLL can be use to drive DPA-enabled channels in a quadrant.

Figure 6-16: Invalid Usage of Corner PLLs Driving DPA-enabled Differential I/Os



DPA Restrictions

Because there is only a single DPA clock bus, a PLL drives a continuous series of DPA channels.

To prevent noise mixing, use one row of separation between two groups of DPA channels.

Guideline: Using DPA-Disabled Differential Channels

If you use DPA-disabled channels, adhere to the following guidelines.

DPA-Disabled Channel Driving Distance

Each PLL can drive all the DPA-disabled channels located in the entire bank.

Using Corner and Center PLLs

You can use a corner PLL to drive all transmitter channels and a center PLL to drive all DPA-disabled receiver channels in the same I/O bank. You can drive a transmitter channel and a receiver channel in the same LAB row by two different PLLs. A corner PLL and a center PLL can drive duplex channels in the same I/O bank if the channels that are driven by each PLL are not interleaved. You do not require separation between the group of channels that are driven by the corner and center, left and right PLLs. Refer to the following figures.

Figure 6-17: Corner and Center PLLs Driving DPA-Disabled Differential I/Os in the Same Bank

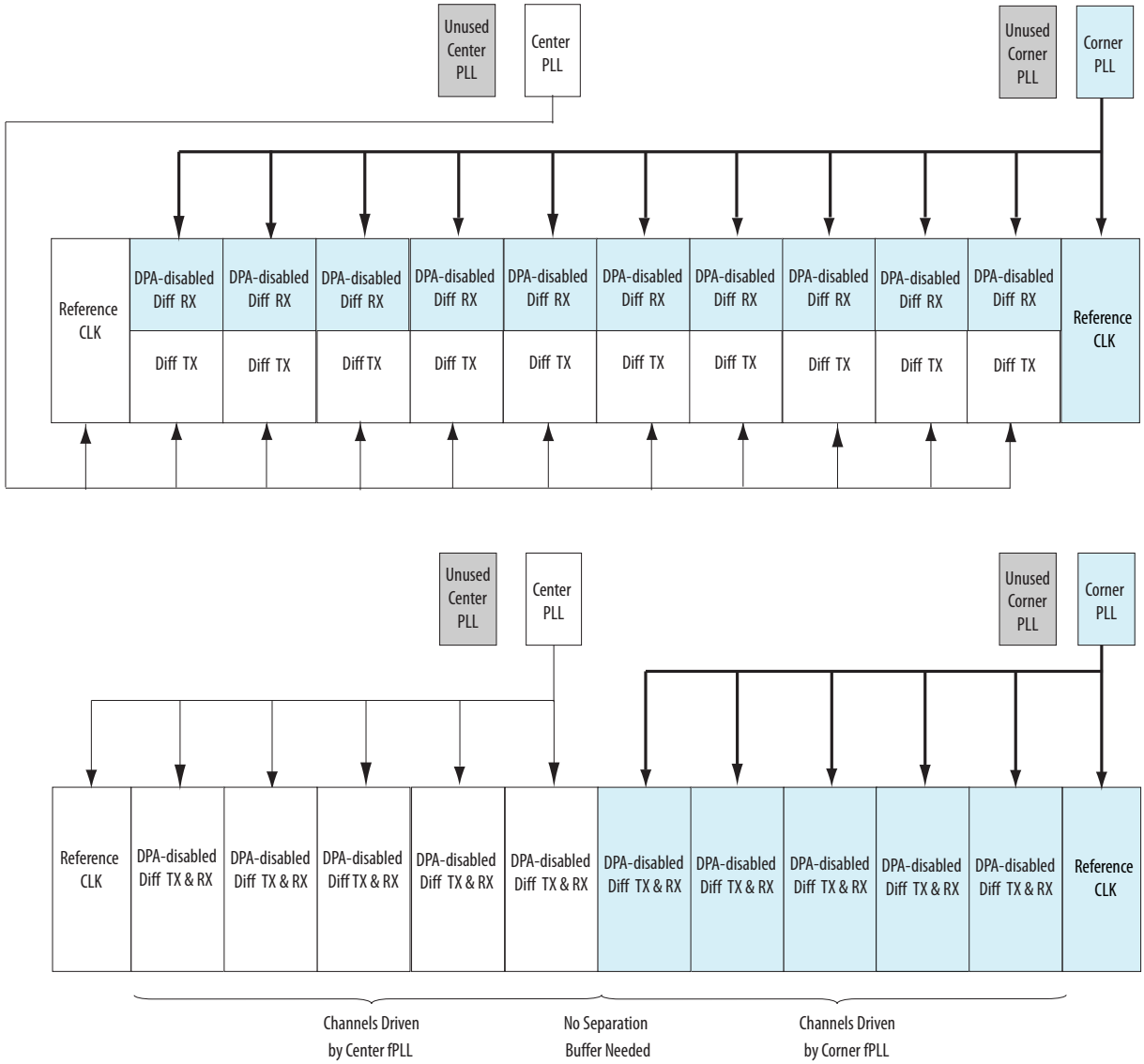
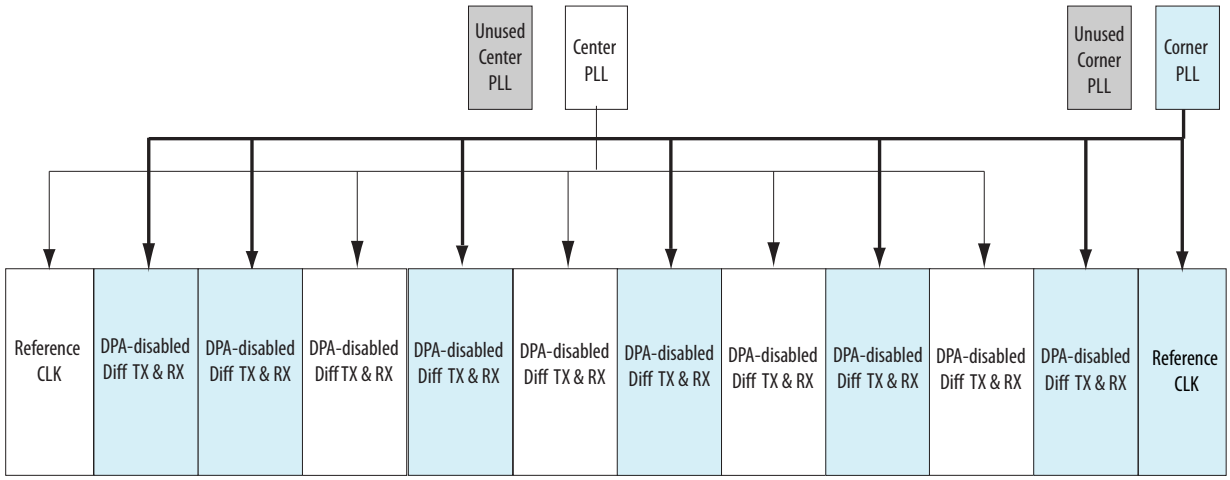


Figure 6-18: Invalid Placement of DPA-disabled Differential I/Os Due to Interleaving of Channels Driven by the Corner and Center PLLs

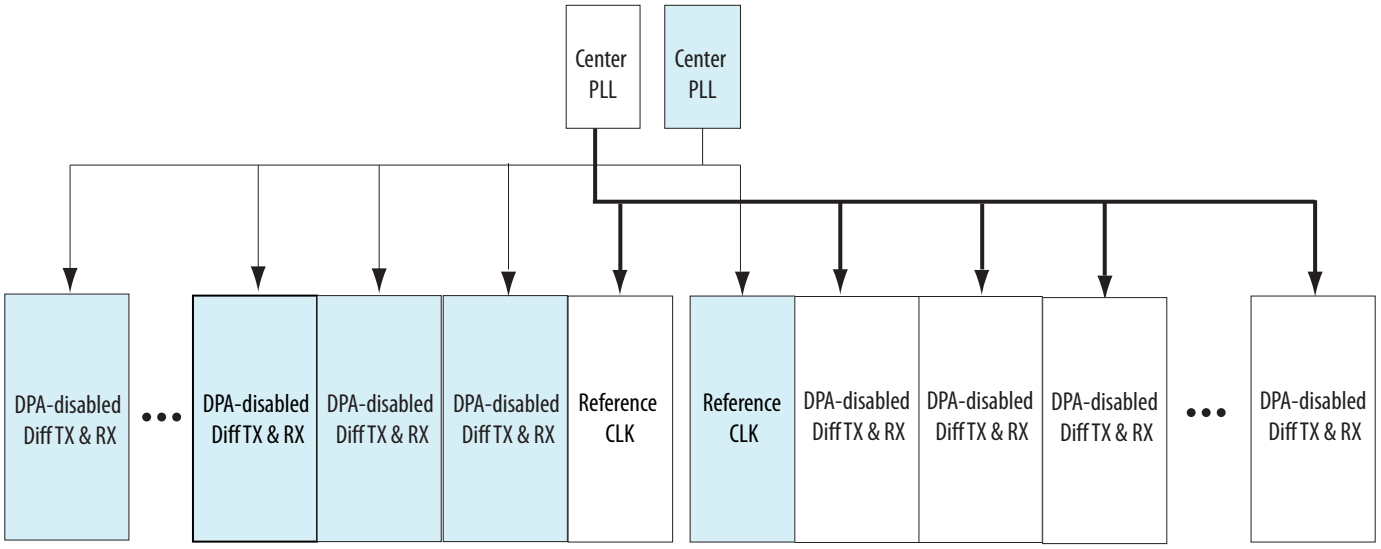


Using Both Center PLLs

You can use both center PLLs simultaneously to drive DPA-disabled channels on left and right I/O banks. Unlike DPA-enabled channels, the center PLLs can drive DPA-disabled channels cross-banks in the Arria V GX, GT, SX, and ST devices. For example, the left center PLL can drive the right I/O bank at the same time the right center PLL is driving the left I/O bank, and vice versa, as shown in the following figure.

Note: In the Arria V GZ devices, the center PLLs cannot drive DPA-disabled channels cross-banks.

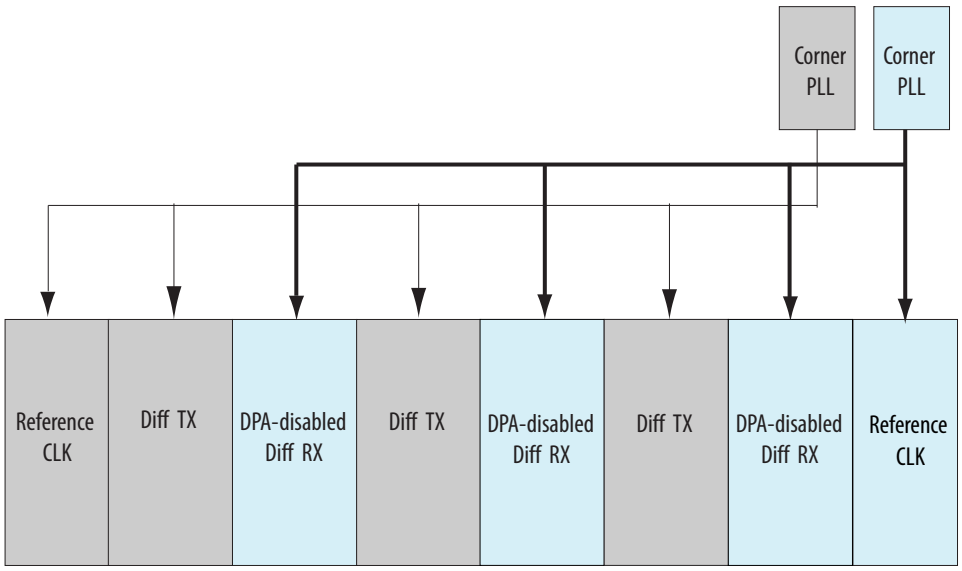
Figure 6-19: Both Center PLLs Driving Cross-Bank DPA-Disabled Channels Simultaneously in Arria V GX, GT, SX, and ST Devices



Using Both Corner PLLs

You can use both corner PLLs to drive DPA-disabled channels simultaneously. You can use a corner PLL to drive all the transmitter channels and the other corner PLL to drive all the DPA-disabled receiver channels in the same I/O bank. Both corner PLLs can drive duplex channels in the same I/O bank if the channels that are driven by each PLL are not interleaved. You do not require separation between the groups of channels that are driven by both corner PLLs.

Figure 6-20: Right Corner PLLs Driving LVDS Differential I/Os in the Same Bank



Differential Transmitter in Arria V Devices

The Arria V transmitter contains dedicated circuitry to support high-speed differential signaling. The differential transmitter buffers support the following features:

- LVDS signaling that can drive out LVDS, mini-LVDS, and RSDS signals
- Programmable V_{OD} and programmable pre-emphasis

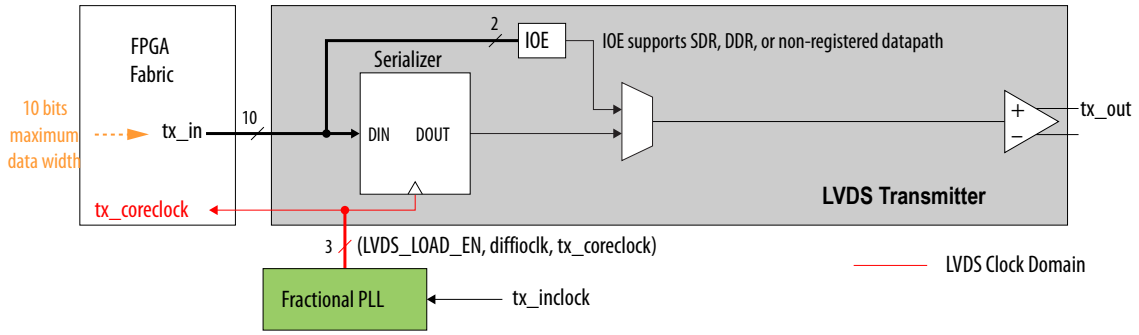
Transmitter Blocks

The dedicated circuitry consists of a true differential buffer, a serializer, and fractional PLLs that you can share between the transmitter and receiver. The serializer takes up to 10 bits wide parallel data from the FPGA fabric, clocks it into the load registers, and serializes it using shift registers that are clocked by the fractional PLL before sending the data to the differential buffer. The MSB of the parallel data is transmitted first.

Note: To drive the LVDS channels, you must use the PLLs in integer PLL mode.

The following figure shows a block diagram of the transmitter. In SDR and DDR modes, the data width is 1 and 2 bits, respectively.

Figure 6-21: LVDS Transmitter



Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Transmitter Clocking

The fractional PLL generates the load enable (`LVDS_LOAD_EN`) signal and the `diffioclk` signal (the clock running at serial data rate) that clocks the load and shift registers. You can statically set the serialization factor to x3, x4, x5, x6, x7, x8, x9, or x10 using the Intel Quartus Prime software. The load enable signal is derived from the serialization factor setting.

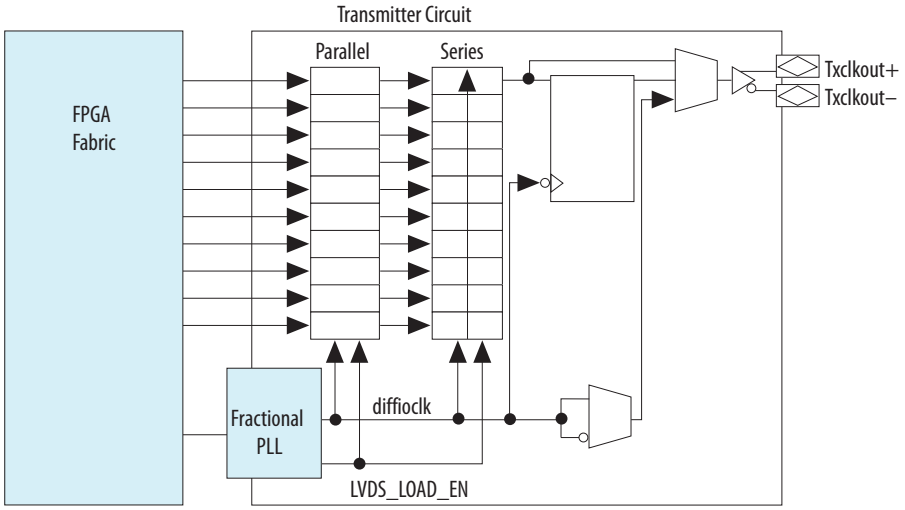
You can configure any Arria V transmitter data channel to generate a source-synchronous transmitter clock output. This flexibility allows the placement of the output clock near the data outputs to simplify board layout and reduce clock-to-data skew.

Different applications often require specific clock-to-data alignments or specific data-rate-to-clock-rate factors. You can specify these settings statically in the Intel Quartus Prime IP Catalog:

- The transmitter can output a clock signal at the same rate as the data—with a maximum output clock frequency that each speed grade of the device supports.
- You can divide the output clock by a factor of 1, 2, 4, 6, 8, or 10, depending on the serialization factor.
- You can set the phase of the clock in relation to the data using internal PLL option of the ALTLVDS IP core. The fractional PLLs provide additional support for other phase shifts in 45° increments.

The following figure shows the transmitter in clock output mode. In clock output mode, you can use an LVDS channel as a clock output channel.

Figure 6-22: Transmitter in Clock Output Mode



Related Information

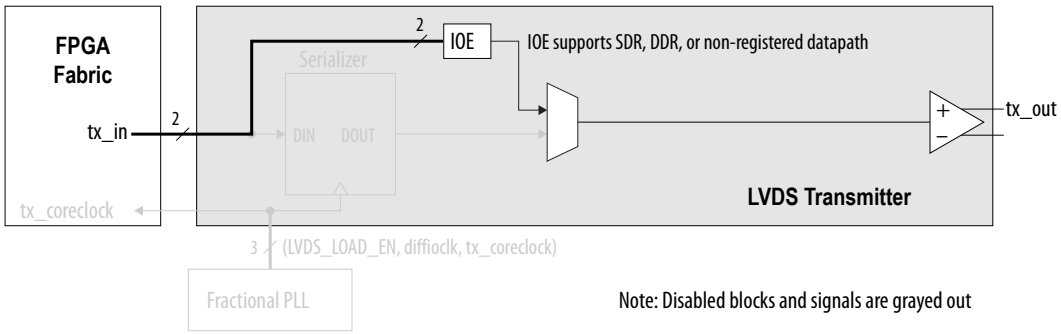
Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Serializer Bypass for DDR and SDR Operations

You can bypass the serializer to support DDR (x2) and SDR (x1) operations to achieve a serialization factor of 2 and 1, respectively. The I/O element (IOE) contains two data output registers that can each operate in either DDR or SDR mode.

Figure 6-23: Serializer Bypass

This figure shows the serializer bypass path. In DDR mode, `tx_inclock` clocks the IOE register. In SDR mode, data is passed directly through the IOE. In SDR and DDR modes, the data width to the IOE is 1 and 2 bits, respectively.



Note: Disabled blocks and signals are grayed out

Programmable Differential Output Voltage

The programmable V_{OD} settings allow you to adjust the output eye opening to optimize the trace length and power consumption. A higher V_{OD} swing improves voltage margins at the receiver end, and a smaller

V_{OD} swing reduces power consumption. You can statically adjust the V_{OD} of the differential signal by changing the V_{OD} settings in the Intel Quartus Prime software Assignment Editor.

Figure 6-24: Differential V_{OD}

This figure shows the V_{OD} of the differential LVDS output.

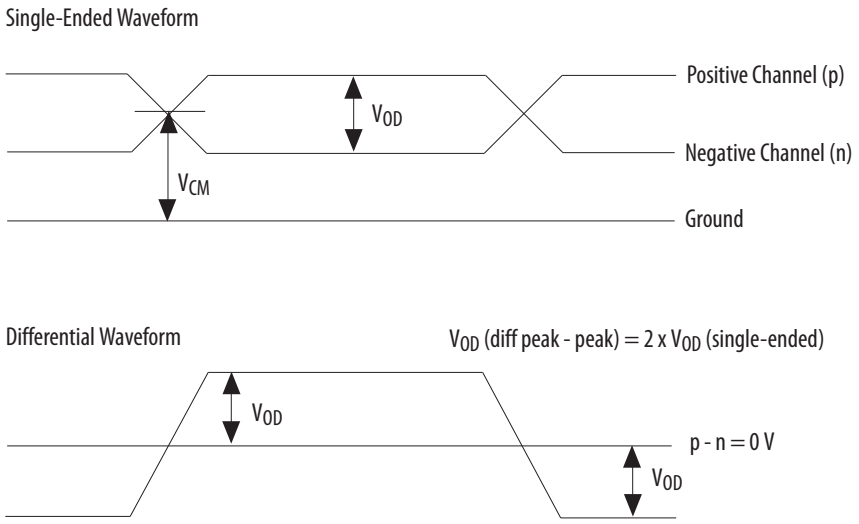


Table 6-9: Intel Quartus Prime Software Assignment Editor—Programmable V_{OD}

This table lists the assignment name for programmable V_{OD} and its possible values in the Intel Quartus Prime software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Differential Output Voltage (V_{OD})
Allowed values	• Arria V GX, GT, SX, and ST—0 (low), 1 (medium), 2 (high). Default is 1. • Arria V GZ—0 (low), **1** (medium low), 2 (medium high), 3 (high). Default is 1.

Related Information

Programmable IOE Features in Arria V Devices on page 5-26

Programmable Pre-Emphasis

The V_{OD} setting and the output impedance of the driver set the output current limit of a high-speed transmission signal. At a high frequency, the slew rate may not be fast enough to reach the full V_{OD} level before the next edge, producing pattern-dependent jitter. With pre-emphasis, the output current is boosted momentarily during switching to increase the output slew rate.

Pre-emphasis increases the amplitude of the high-frequency component of the output signal, and thus helps to compensate for the frequency-dependent attenuation along the transmission line. The overshoot introduced by the extra current happens only during a change of state switching to increase the output

slew rate and does not ring, unlike the overshoot caused by signal reflection. The amount of pre-emphasis required depends on the attenuation of the high-frequency component along the transmission line.

Figure 6-25: Programmable Pre-Emphasis

This figure shows the LVDS output with pre-emphasis.

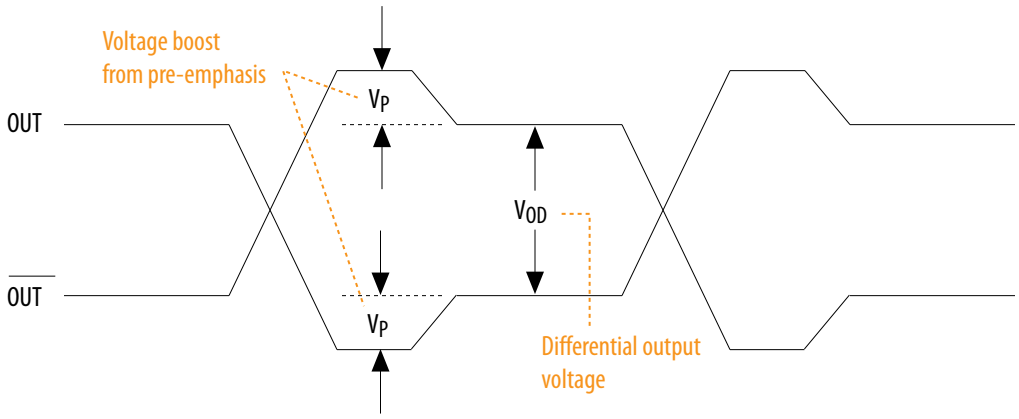


Table 6-10: Intel Quartus Prime Software Assignment Editor—Programmable Pre-Emphasis

This table lists the assignment name for programmable pre-emphasis and its possible values in the Intel Quartus Prime software Assignment Editor.

Field	Assignment
To	`tx_out`
Assignment name	Programmable Pre-emphasis
Allowed values	0 (disabled), 1 (enabled). Default is 1.

Related Information

Programmable IOE Features in Arria V Devices on page 5-26

Differential Receiver in Arria V Devices

The receiver has a differential buffer and fractional PLLs that you can share among the transmitter and receiver, a DPA block, a synchronizer, a data realignment block, and a deserializer. The differential buffer can receive LVDS, mini-LVDS, and RSDS signal levels. You can statically set the I/O standard of the receiver pins to LVDS, mini-LVDS, or RSDS in the Intel Quartus Prime software Assignment Editor.

Note: To drive the LVDS channels, you must use the PLLs in integer PLL mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

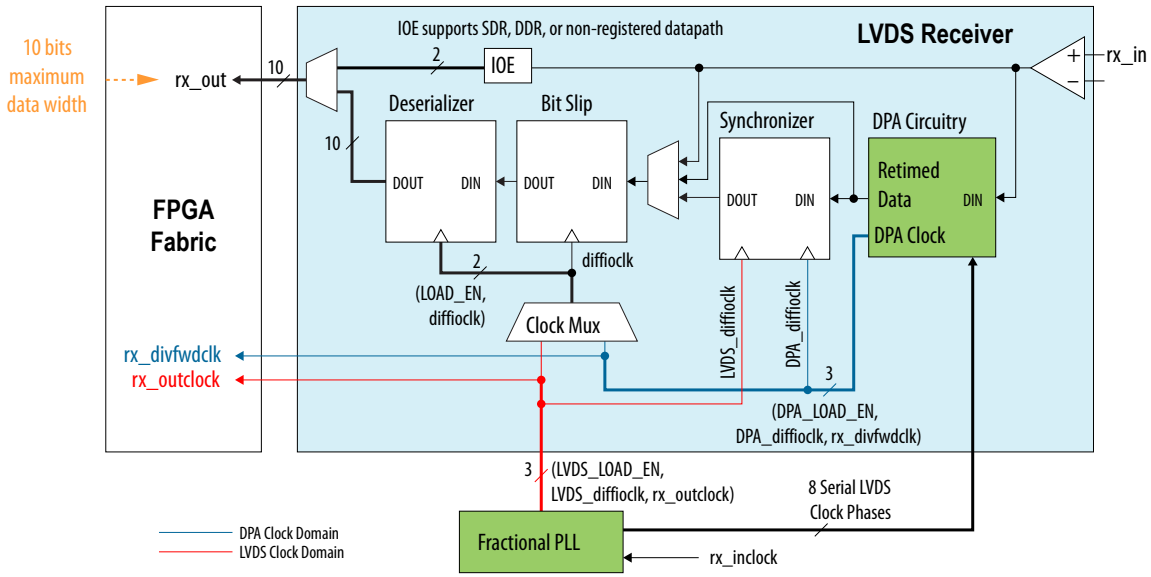
Receiver Blocks in Arria V Devices

The Arria V differential receiver has the following hardware blocks:

- DPA block
- Synchronizer
- Data realignment block (bit slip)
- Deserializer

The following figure shows the hardware blocks of the receiver. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively. The deserializer includes shift registers and parallel load registers, and sends a maximum of 10 bits to the internal logic.

Figure 6-26: Receiver Block Diagram

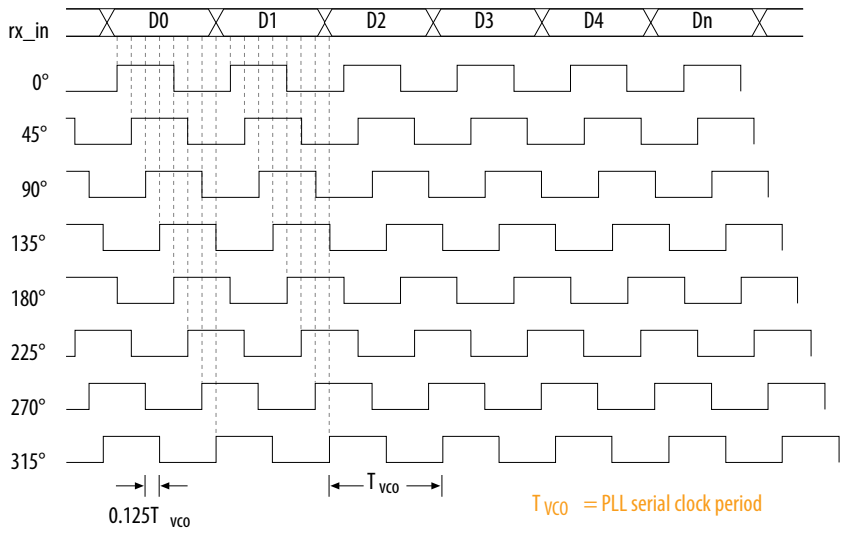


DPA Block

The DPA block takes in high-speed serial data from the differential input buffer and selects one of the eight phases that the fractional PLLs generate to sample the data. The DPA chooses a phase closest to the phase of the serial data. The maximum phase offset between the received data and the selected phase is 1/8 UI, which is the maximum quantization error of the DPA. The eight phases of the clock are equally divided, offering a 45° resolution.

The following figure shows the possible phase relationships between the DPA clocks and the incoming serial data.

Figure 6-27: DPA Clock Phase to Serial Data Timing Relationship



The DPA block continuously monitors the phase of the incoming serial data and selects a new clock phase if it is required. You can prevent the DPA from selecting a new clock phase by asserting the optional `RX_DPLL_HOLD` port, which is available for each channel.

DPA circuitry does not require a fixed training pattern to lock to the optimum phase out of the eight phases. After reset or power up, the DPA circuitry requires transitions on the received data to lock to the optimum phase. An optional output port, `RX_DPA_LOCKED`, is available to indicate an initial DPA lock condition to the optimum phase after power up or reset. This signal is not deasserted if the DPA selects a new phase out of the eight clock phases to sample the received data. Do not use the `rx_dpa_locked` signal to determine a DPA loss-of-lock condition. Use data checkers such as a cyclic redundancy check (CRC) or diagonal interleaved parity (DIP-4) to validate the data.

An independent reset port, `RX_RESET`, is available to reset the DPA circuitry. You must retrain the DPA circuitry after reset.

Note: The DPA block is bypassed in non-DPA mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Synchronizer

The synchronizer is a 1 bit wide and 6 bit deep FIFO buffer that compensates for the phase difference between `DPA_diffioclk`—the optimal clock that the DPA block selects—and the `LVDS_diffioclk` that the fractional PLLs produce. The synchronizer can only compensate for phase differences, not frequency differences, between the data and the receiver's input reference clock.

An optional port, `RX_FIFO_RESET`, is available to the internal logic to reset the synchronizer. The synchronizer is automatically reset when the DPA first locks to the incoming data. Altera recommends using `RX_FIFO_RESET` to reset the synchronizer when the data checker indicates that the received data is corrupted.

Note: The synchronizer circuit is bypassed in non-DPA and soft-CDR mode.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Data Realignment Block (Bit Slip)

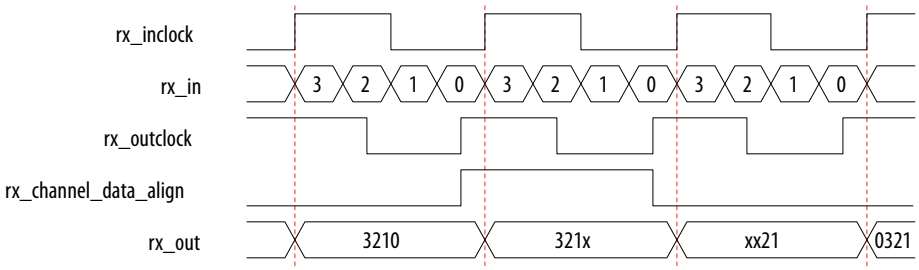
Skew in the transmitted data along with skew added by the link causes channel-to-channel skew on the received serial data streams. If you enable the DPA, the received data is captured with different clock phases on each channel. This difference may cause misalignment of the received data from channel to channel. To compensate for this channel-to-channel skew and establish the correct received word boundary at each channel, each receiver channel has a dedicated data realignment circuit that realigns the data by inserting bit latencies into the serial stream.

An optional `RX_CHANNEL_DATA_ALIGN` port controls the bit insertion of each receiver independently controlled from the internal logic. The data slips one bit on the rising edge of `RX_CHANNEL_DATA_ALIGN`. The requirements for the `RX_CHANNEL_DATA_ALIGN` signal include the following items:

- The minimum pulse width is one period of the parallel clock in the logic array.
- The minimum low time between pulses is one period of the parallel clock.
- The signal is an edge-triggered signal.
- The valid data is available two parallel clock cycles after the rising edge of `RX_CHANNEL_DATA_ALIGN`.

Figure 6-28: Data Realignment Timing

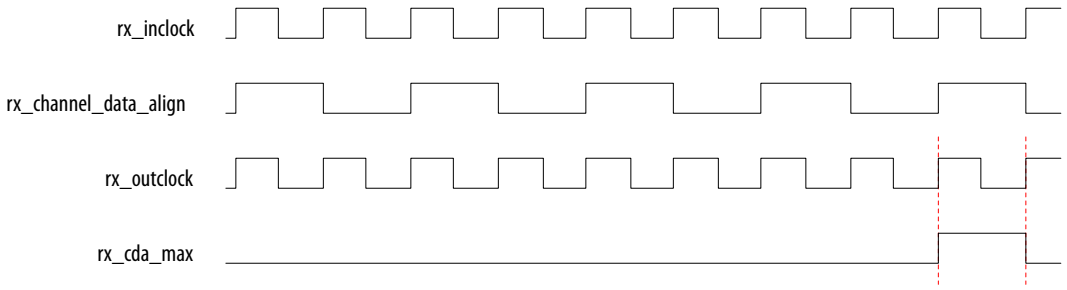
This figure shows receiver output (`RX_OUT`) after one bit slip pulse with the deserialization factor set to **4**.



The data realignment circuit can have up to 11 bit-times of insertion before a rollover occurs. The programmable bit rollover point can be from 1 to 11 bit-times, independent of the deserialization factor. Set the programmable bit rollover point equal to, or greater than, the deserialization factor—allowing enough depth in the word alignment circuit to slip through a full word. You can set the value of the bit rollover point using the IP Catalog. An optional status port, `RX_CDA_MAX`, is available to the FPGA fabric from each channel to indicate the reaching of the preset rollover point.

Figure 6-29: Receiver Data Realignment Rollover

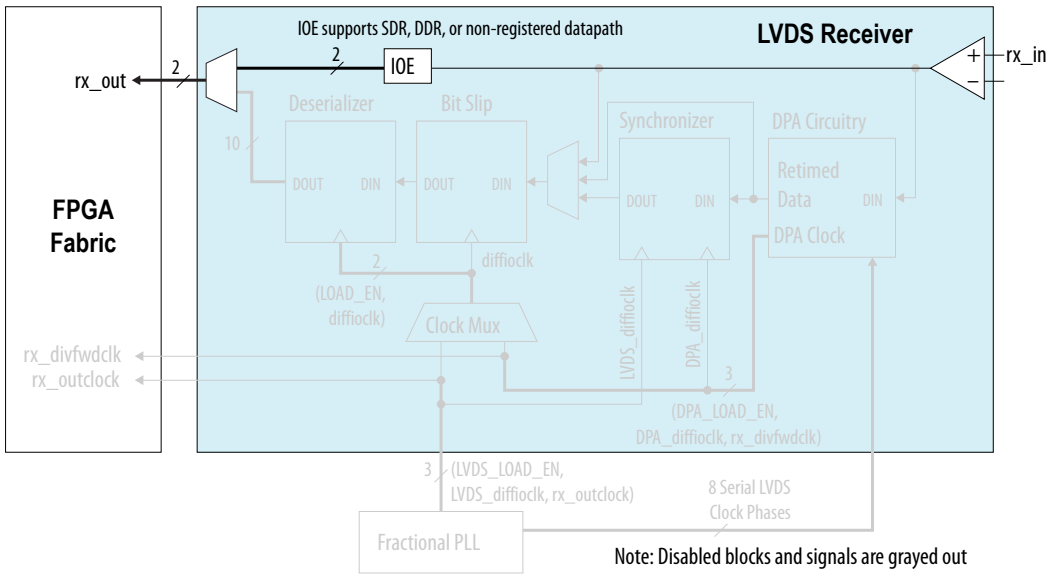
This figure shows a preset value of four bit-times before rollover occurs. The `rx_cda_max` signal pulses for one `rx_outclock` cycle to indicate that rollover has occurred.



Deserializer

You can statically set the deserialization factor to x3, x4, x5, x6, x7, x8, x9, or x10 by using the Intel Quartus Prime software. You can bypass the deserializer in the Intel Quartus Prime IP Catalog to support DDR (x2) or SDR (x1) operations, as shown in the following figure.

Figure 6-30: Deserializer Bypass



The IOE contains two data input registers that can operate in DDR or SDR mode. In DDR mode, `rx_inclock` clocks the IOE register. In SDR mode, data is directly passed through the IOE. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.

You cannot use the DPA and data realignment circuit when you bypass the deserializer.

Receiver Modes in Arria V Devices

The Arria V devices support the following receiver modes:

- Non-DPA mode
- DPA mode
- Soft-CDR mode

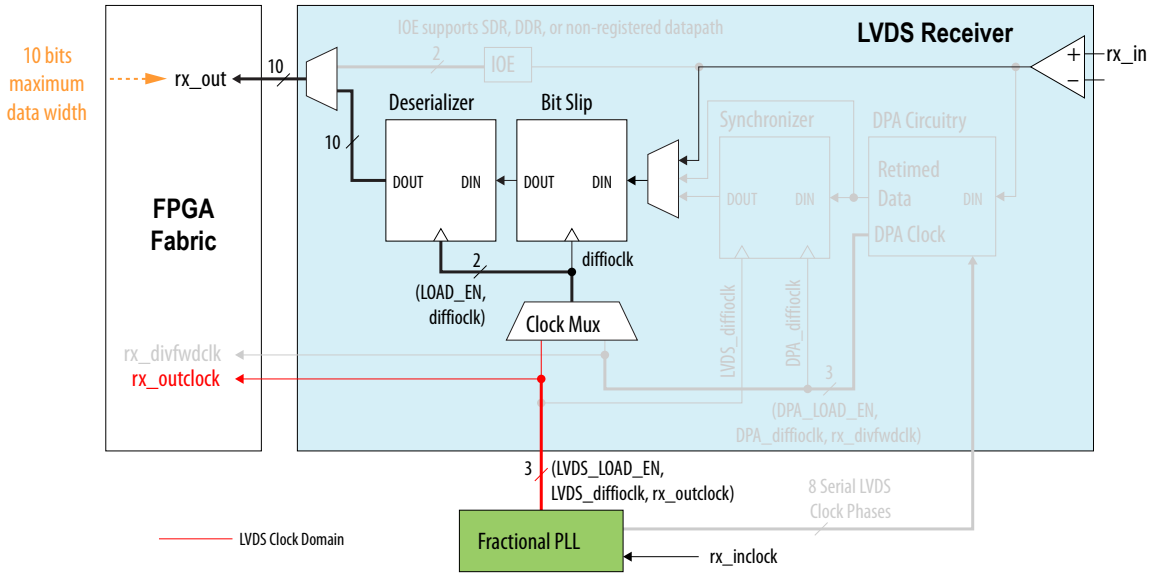
Non-DPA Mode

The non-DPA mode disables the DPA and synchronizer blocks. Input serial data is registered at the rising edge of the serial `LVDS_diffioclk` clock that is produced by the left and right PLLs.

You can select the rising edge option with the Intel Quartus Prime IP Catalog. The `LVDS_diffioclk` clock that is generated by the left and right PLLs clocks the data realignment and deserializer blocks.

The following figure shows the non-DPA datapath block diagram. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.

Figure 6-31: Receiver Data Path in Non-DPA Mode



Note: All disabled blocks and signals are grayed out

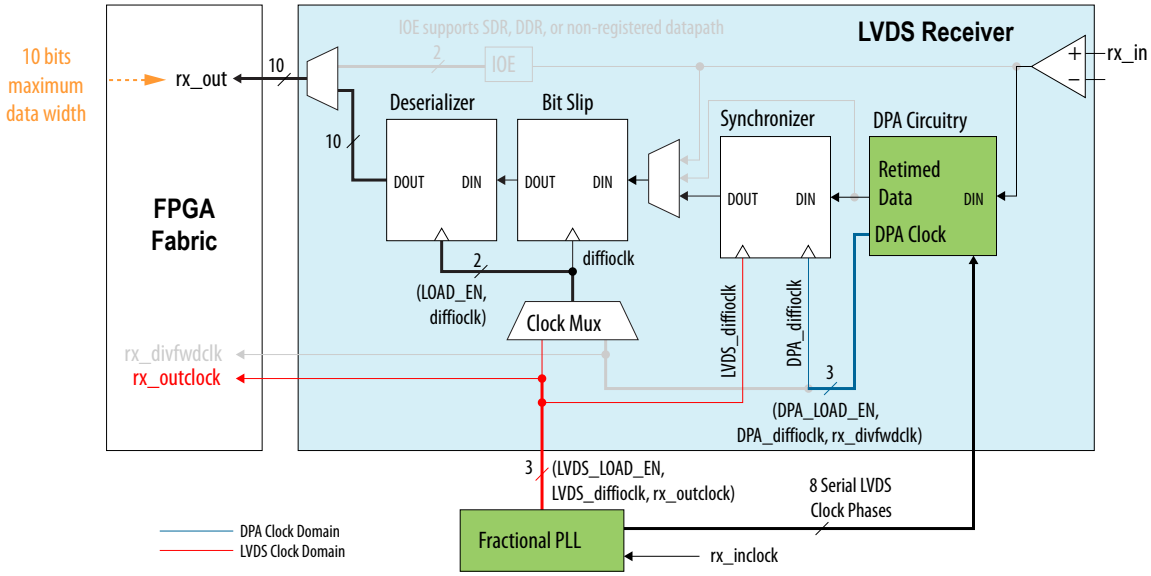
DPA Mode

The DPA block chooses the best possible clock (`DPA_diffioclk`) from the eight fast clocks that the fractional PLL sent. This serial `DPA_diffioclk` clock is used for writing the serial data into the synchronizer. A serial `LVDS_diffioclk` clock is used for reading the serial data from the synchronizer. The same `LVDS_diffioclk` clock is used in data realignment and deserializer blocks.

The following figure shows the DPA mode datapath. In the figure, all the receiver hardware blocks are active.

Figure 6-32: Receiver Datapath in DPA Mode

In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.



Related Information

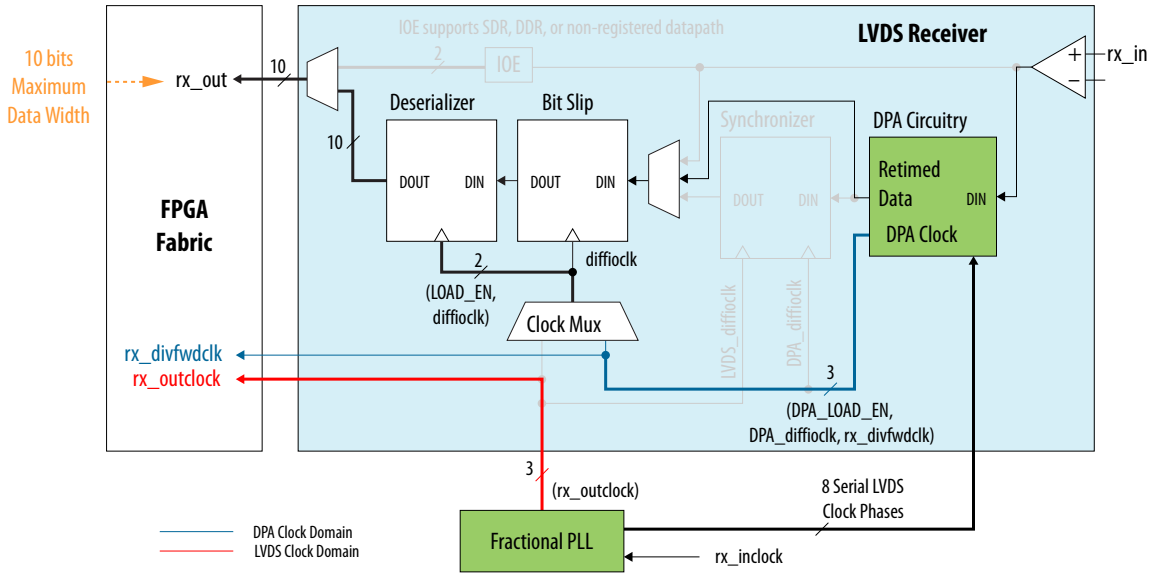
- **Guideline: Use PLLs in Integer PLL Mode for LVDS** on page 6-8
- **Receiver Blocks in Arria V Devices** on page 6-26

Soft-CDR Mode

The Arria V LVDS channel offers the soft-CDR mode to support the GbE and SGMII protocols. A receiver PLL uses the local clock source for reference.

The following figure shows the soft-CDR mode datapath. In SDR and DDR modes, the data width from the IOE is 1 and 2 bits, respectively.

Figure 6-33: Receiver Datapath in Soft-CDR Mode



Note: All disabled blocks and signals are grayed out

In soft-CDR mode, the synchronizer block is inactive. The DPA circuitry selects an optimal DPA clock phase to sample the data. Use the selected DPA clock for bit-slip operation and deserialization. The DPA block also forwards the selected DPA clock, divided by the deserialization factor called `rx_divfwdclk`, to the FPGA fabric, along with the deserialized data. This clock signal is put on the periphery clock (PCLK) network.

If you use the soft-CDR mode, do not assert the `rx_reset` port after the DPA has trained. The DPA continuously chooses new phase taps from the PLL to track parts per million (PPM) differences between the reference clock and incoming data.

You can use every LVDS channel in soft-CDR mode and drive the FPGA fabric using the PCLK network in the Arria V device family. The `rx_dpa_locked` signal is not valid in soft-CDR mode because the DPA continuously changes its phase to track PPM differences between the upstream transmitter and the local receiver input reference clocks. The parallel clock, `rx_outclock`, generated by the left and right PLLs, is also forwarded to the FPGA fabric.

Related Information

Periphery Clock Networks on page 4-5
Provides more information about PCLK networks.

Receiver Clocking for Arria V Devices

The fractional PLL receives the external clock input and generates different phases of the same clock. The DPA block automatically chooses one of the clocks from the fractional PLL and aligns the incoming data on each channel.

The synchronizer circuit is a 1 bit wide by 6 bit deep FIFO buffer that compensates for any phase difference between the DPA clock and the data realignment block. If necessary, the user-controlled data realignment circuitry inserts a single bit of latency in the serial bit stream to align to the word boundary.

The physical medium connecting the transmitter and receiver LVDS channels may introduce skew between the serial data and the source-synchronous clock. The instantaneous skew between each LVDS channel and the clock also varies with the jitter on the data and clock signals as seen by the receiver. The three different modes—non-DPA, DPA, and soft-CDR—provide different options to overcome skew between the source synchronous clock (non-DPA, DPA) /reference clock (soft-CDR) and the serial data.

Non-DPA mode allows you to statically select the optimal phase between the source synchronous clock and the received serial data to compensate skew. In DPA mode, the DPA circuitry automatically chooses the best phase to compensate for the skew between the source synchronous clock and the received serial data. Soft-CDR mode provides opportunities for synchronous and asynchronous applications for chip-to-chip and short reach board-to-board applications for SGMII protocols.

Note: Only the non-DPA mode requires manual skew adjustment.

Related Information

Guideline: Use PLLs in Integer PLL Mode for LVDS on page 6-8

Differential I/O Termination for Arria V Devices

The Arria V devices provide a 100 Ω, on-chip differential termination option on each differential receiver channel for LVDS standards. On-chip termination saves board space by eliminating the need to add external resistors on the board. You can enable on-chip termination in the Intel Quartus Prime software Assignment Editor.

All I/O pins and dedicated clock input pins support on-chip differential termination, R_D OCT.

Figure 6-34: On-Chip Differential I/O Termination

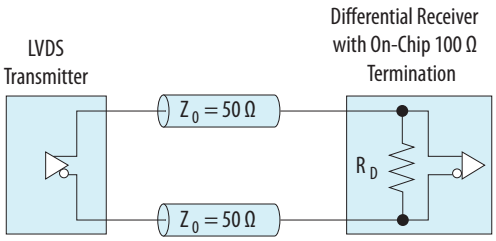


Table 6-11: Intel Quartus Prime Software Assignment Editor—On-Chip Differential Termination

This table lists the assignment name for on-chip differential termination in the Intel Quartus Prime software Assignment Editor.

Field	Assignment
To	rx_in
Assignment name	Input Termination
Value	Differential

Source-Synchronous Timing Budget

The topics in this section describe the timing budget, waveforms, and specifications for source-synchronous signaling in the Arria V device family.

The LVDS I/O standard enables high-speed transmission of data, resulting in better overall system performance. To take advantage of fast system performance, you must analyze the timing for these high-speed signals. Timing analysis for the differential block is different from traditional synchronous timing analysis techniques.

The basis of the source synchronous timing analysis is the skew between the data and the clock signals instead of the clock-to-output setup times. High-speed differential data transmission requires the use of timing parameters provided by IC vendors and is strongly influenced by board skew, cable skew, and clock jitter.

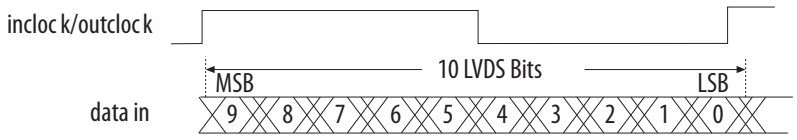
This section defines the source-synchronous differential data orientation timing parameters, the timing budget definitions for the Arria V device family, and how to use these timing parameters to determine the maximum performance of a design.

Differential Data Orientation

There is a set relationship between an external clock and the incoming data. For operations at 1 Gbps and a serialization factor of 10, the external clock is multiplied by 10. You can set phase-alignment in the PLL to coincide with the sampling window of each data bit. The data is sampled on the falling edge of the multiplied clock.

Figure 6-35: Bit Orientation in the Intel Quartus Prime Software

This figure shows the data bit orientation of the x10 mode.



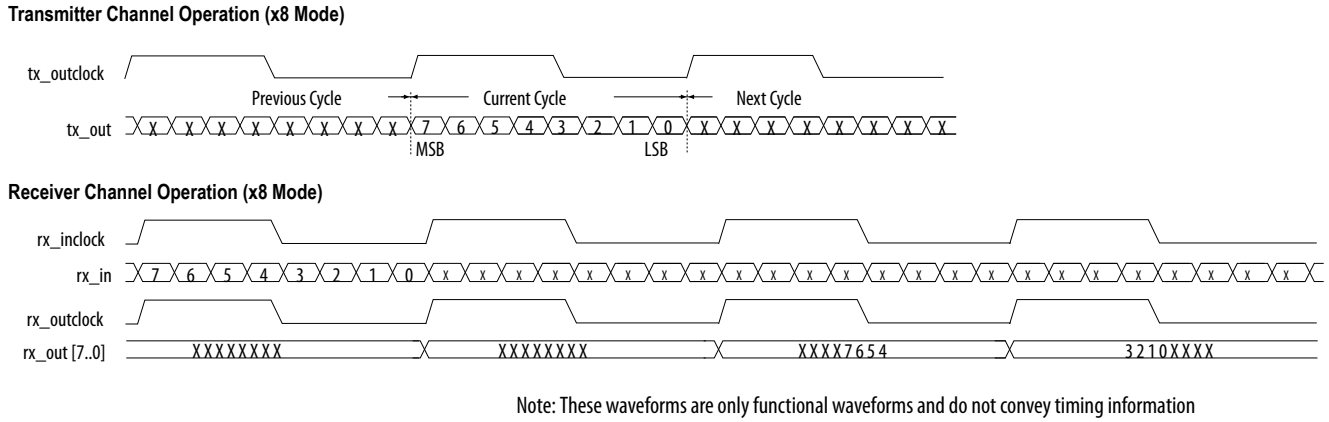
Differential I/O Bit Position

Data synchronization is necessary for successful data transmission at high frequencies.

The following figure shows the data bit orientation for a channel operation and is based on the following conditions:

- The serialization factor is equal to the clock multiplication factor.
- The phase alignment uses edge alignment.
- The operation is implemented in hard SERDES.

Figure 6-36: Bit-Order and Word Boundary for One Differential Channel



Note: These waveforms are only functional waveforms and do not convey timing information

For other serialization factors, use the Intel Quartus Prime software tools to find the bit position within the word.

Differential Bit Naming Conventions

The following table lists the conventions for differential bit naming for 18 differential channels. The MSB and LSB positions increase with the number of channels used in a system.

Table 6-12: Differential Bit Naming

This table lists the conventions for differential bit naming for 18 differential channels. The MSB and LSB positions increase with the number of channels used in a system.

Receiver Channel Data Number	Internal 8-Bit Parallel Data	
	MSB Position	LSB Position
1	7	0
2	15	8
3	23	16
4	31	24
5	39	32
6	47	40
7	55	48
8	63	56
9	71	64
10	79	72
11	87	80
12	95	88
13	103	96

Receiver Channel Data Number	Internal 8-Bit Parallel Data	
	MSB Position	LSB Position
14	111	104
15	119	112
16	127	120
17	135	128
18	143	136

Transmitter Channel-to-Channel Skew

The receiver skew margin calculation uses the transmitter channel-to-channel skew (TCCS)—an important parameter based on the Arria V transmitter in a source-synchronous differential interface:

- TCCS is the difference between the fastest and slowest data output transitions, including the T_{CO} variation and clock skew.
- For LVDS transmitters, the Timing Analyzer provides the TCCS value in the TCCS report (`report_TCCS`) in the Intel Quartus Prime compilation report, which shows TCCS values for serial output ports.
- You can also get the TCCS value from the device datasheet.

Note: For the Arria V GZ devices, perform PCB trace compensation to adjust the trace length of each LVDS channel to improve channel-to-channel skew when interfacing with non-DPA receivers at data rate above 840 Mbps.

The Intel Quartus Prime software Fitter Report panel reports the amount of delay you must add to each trace for the Arria V device. You can use the recommended trace delay numbers published under the LVDS Transmitter/Receiver Package Skew Compensation panel and manually compensate the skew on the PCB board trace to reduce channel-to-channel skew, thus meeting the timing budget between LVDS channels.

Related Information

- **Arria V GX, GT, XS, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**
- **LVDS SERDES Transmitter/Receiver IP Cores User Guide**
  More information about the LVDS Transmitter/Receiver Package Skew Compensation report panel.

Receiver Skew Margin for Non-DPA Mode

Different modes of LVDS receivers use different specifications, which can help in deciding the ability to sample the received serial data correctly:

- In DPA mode, use DPA jitter tolerance instead of the receiver skew margin (RSKM).
- In non-DPA mode, use RSKM, TCCS, and sampling window (SW) specifications for high-speed source-synchronous differential signals in the receiver data path.

The following equation expresses the relationship between RSKM, TCCS, and SW.

Figure 6-37: RSKM Equation

$$RSKM = \frac{TUI - SW - TCCS}{2}$$

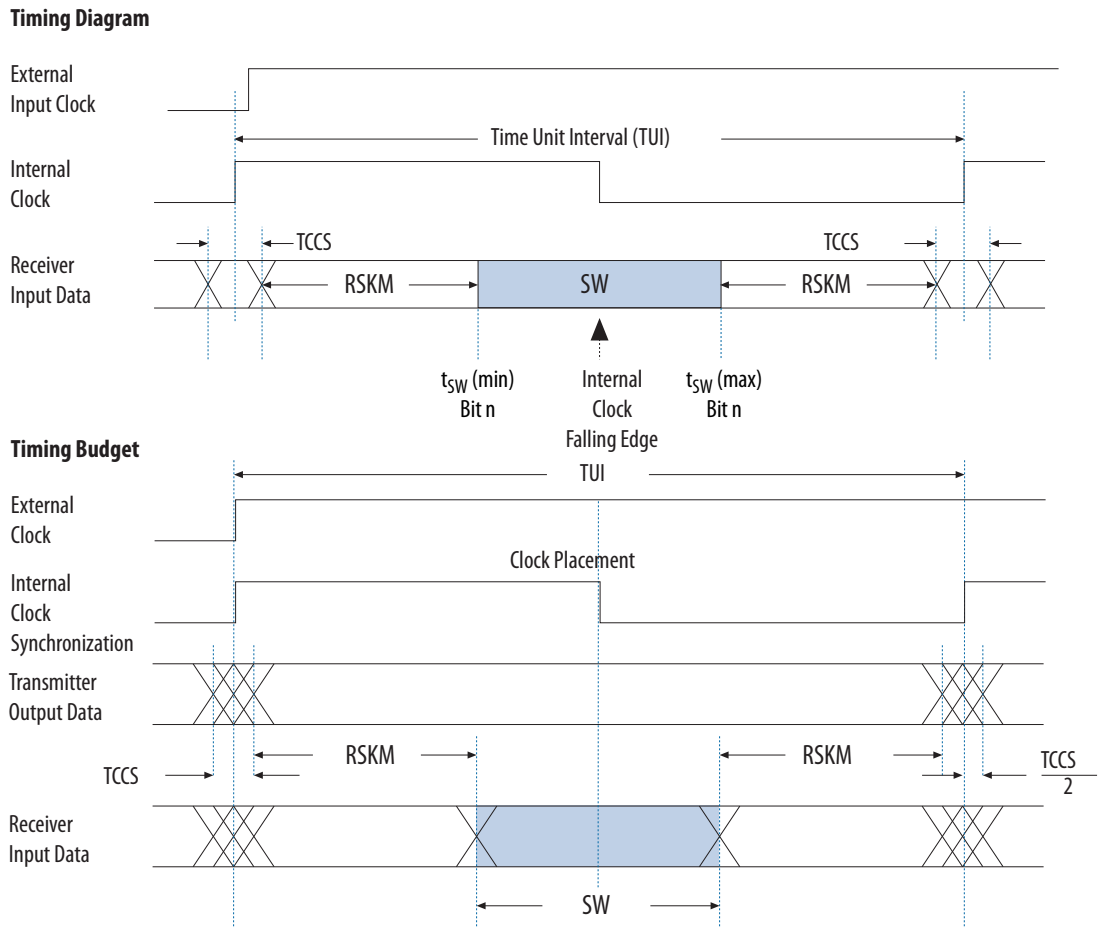
Conventions used for the equation:

- RSKM—the timing margin between the receiver's clock input and the data input sampling window.
- Time unit interval (TUI)—time period of the serial data.
- SW—the period of time that the input data must be stable to ensure that data is successfully sampled by the LVDS receiver. The SW is a device property and varies with device speed grade.
- TCCS—the timing difference between the fastest and the slowest output edges, including t_{CO} variation and clock skew, across channels driven by the same PLL. The clock is included in the TCCS measurement.

You must calculate the RSKM value to decide whether the LVDS receiver can sample the data properly or not, given the data rate and device. A positive RSKM value indicates that the LVDS receiver can sample the data properly, whereas a negative RSKM indicates that it cannot sample the data properly.

The following figure shows the relationship between the RSKM, TCCS, and the SW of the receiver.

Figure 6-38: Differential High-Speed Timing Diagram and Timing Budget for Non-DPA Mode



For LVDS receivers, the Intel Quartus Prime software provides an RSKM report showing the SW, TUI, and RSKM values for non-DPA LVDS mode:

- You can generate the RSKM report by executing the `report_RSKM` command in the Timing Analyzer. You can find the RSKM report in the Intel Quartus Prime compilation report in the Timing Analyzer section.
- To obtain the RSKM value, assign the input delay to the LVDS receiver through the constraints menu of the Timing Analyzer. The input delay is determined according to the data arrival time at the LVDS receiver port, with respect to the reference clock.
- If you set the input delay in the settings parameters for the **Set Input Delay** option, set the clock name to the clock that reference the source synchronous clock that feeds the LVDS receiver.
- If you do not set any input delay in the Timing Analyzer, the receiver channel-to-channel skew defaults to zero.
- You can also directly set the input delay in a Synopsys Design Constraint file (**.sdc**) using the `set_input_delay` command.

Related Information

- **LVDS SERDES Transmitter/Receiver IP Cores User Guide**
  More information about the RSKM equation and calculation.
- **The Intel Quartus Prime Timing Analyzer chapter, Intel Quartus Prime Standard Edition Handbook Volume 3 Verification**
  Provides more information about **.sdc** commands and the Timing Analyzer.

Assigning Input Delay to LVDS Receiver Using Timing Analyzer

To obtain the RSKM value, assign an appropriate input delay to the LVDS receiver from the Timing Analyzer constraints menu.

1. On the menu in the Timing Analyzer, select **Constraints** > **Set Input Delay**.
2. In the **Set Input Delay** window, select the desired clock using the pull-down menu. The clock name must reference the source synchronous clock that feeds the LVDS receiver.
3. Click the **Browse** button (next to the **Targets** field).
4. In the **Name Finder** window, click **List** to view a list of all available ports. Select the LVDS receiver serial input ports according to the input delay you set, and click **OK**.
5. In the **Set Input Delay** window, set the appropriate values in the **Input delay** options and **Delay value** fields.
6. Click **Run** to incorporate these values in the Timing Analyzer.
7. Repeat from **step 1** to assign the appropriate delay for all the LVDS receiver input ports. If you have already assigned Input Delay and you need to add more delay to that input port, turn on the **Add Delay** option.

High-Speed Differential I/O Interfaces and DPA in Arria V Devices Revision History

Date	Version	Changes
December 2017	2017.12.15	• Added a note to *Guideline: Use High-Speed Clock from PLL to Clock LVDS SERDES Only* topic to clarify that spread-spectrum input clock is not supported in LVDS. • Updated for latest Intel branding standards.
December 2015	2015.12.21	Changed instances of *Quartus II* to *Quartus Prime*.

Date	Version	Changes
May 2015	2015.05.08	• Changed figure title "Corner PLLs Driving DPA-enabled Differential I/Os" to "Invalid Usage of Corner PLLs Driving DPA-enabled Differential I/Os". • Added LVDS and DPA Clock Network figure in Guideline: Using DPA-Enabled Differential Channels. • Updated all figures in Guideline: Using DPA-Enabled Differential Channels. • Updated guidelines for using both corner PLLs in Arria V Devices. • Updated figures in Guideline: Using DPA-Disabled LVDS Differential Channels. • Updated the statement about emulated LVDS buffers to specify that you can use true LVDS input buffer as emulated output buffers for serialization factor of 1 and 2.
January 2015	2015.01.19	• Removed statement on explanation related to rx_synclock for figure "LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode)". • Updated figure LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode) and figure Receiver Datapath in Soft-CDR Mode. • Added a note to leave rx_enable and rx_inclock to be unconnected for figure LVDS Interface with the Altera_PLL Megafunction (With Soft-CDR Mode). • Updated timing diagram for Phase Relationship for External PLL Interface Signals to reflect the correct phase shift and frequency for outclk2.
January 2014	2014.01.10	• Updated the statement about setting the phase of the clock in relation to data in the topic about transmitter clocking. • Added a figure that shows the phase relationship for the external PLL interface signals. • Clarified that "one row of separation" between two groups of DPA-enabled channels means a separation of one differential channel. • Clarified that "internal PLL option" refers to the option in the ALTLVDS megafunction. • Updated the topic about emulated LVDS buffers to clarify that you can use unutilized true LVDS input channels (instead "buffers") as emulated LVDS output buffers.
August 2013	2013.08.19	Updated the number of LVDS channels of the Arria V GZ E5 and E7 devices (1517-pin package) from 80 to 79 (top banks TX) and 82 to 81 (top banks RX).
June 2013	2013.06.21	Updated the figure about data realignment timing to correct the data pattern after a bit slip.

Date	Version	Changes
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Clarified that the clock tree network cannot cross over to different I/O regions only applies to Arria V GZ. • Added a figure to show the center PLLs driving the DPA-enabled differential I/Os in Arria V GZ devices. • Changed the color of the transceiver blocks in the high-speed differential I/O location diagrams for clarity. • Reorganized contents under the differential receiver topic. • Added a topic about emulated LVDS buffers. • Edited the topic about true LVDS buffers. • Corrected references to upper and lower I/O banks to left and right I/O banks, respectively. • Updated the data realignment timing figure to improve clarity. • Updated the receiver data realignment rollover figure to improve clarity.
November 2012	2012.11.19	• Reorganized content and updated template. • Added Arria V GZ information. • Added Altera_PLL settings for external PLL usage in DPA and non-DPA modes. • Updated clocking examples. Altera_PLL now supports entering negative phase shift. • Rearranged the LVDS channel counts table into several tables according to device variant for ease of reference. • Updated the Arria V GX A1 and A3 LVDS channel counts, and added the channel counts for Arria V GZ. • Removed references to ALTPLL and added information about Altera_PLL. Altera_PLL now replaces ALTPLL for Arria V devices. • Added design guidelines for using LVDS interface with the external PLL mode. These include information on the signal interfaces, the parameter values, and the connection between Altera_PLL and ALTLVDS in external PLL mode. • Updated the programmable V_{OD} allowed values for Arria V GX, GT, SX, and ST, and added the values for Arria V GZ. • Moved the PLL and clocking section into design guideline topics. • Added steps to assign input delay to LVDS receiver using the TimeQuest Timing Analyzer.

Date	Version	Changes
June 2012	2.0	• Restructured the chapter. • Updated Table 6–1. • Updated Figure 6–1 and Figure 6–2. • Added Figure 6–3. • Added "Design Considerations" section. • Updated the "Differential Pin Placement" section.
November 2011	1.1	• Updated Table 6–1. • Restructured chapter.
May 2011	1.0	Initial release.

External Memory Interfaces in Arria V Devices

7

2020.04.13

AV-52007 Subscribe Send Feedback

The Arria V devices provide an efficient architecture that allows you to fit wide external memory interfaces to support a high level of system bandwidth within the small modular I/O bank structure. The I/Os are designed to provide high-performance support for existing and emerging external memory standards.

Table 7-1: Supported External Memory Standards in Arria V Devices

Memory Standard	Hard Memory Controller	Soft Memory Controller	
	Arria V GX, GT, SX, and ST	Arria V GX, GT, SX, and ST	Arria V GZ
DDR3 SDRAM	Full rate	Half rate and quarter rate	Half rate and quarter rate
DDR2 SDRAM	Full rate	Half rate	Full rate and half rate
LPDDR2 SDRAM	—	Half rate	—
RLDRAM 3	—	—	Half rate and quarter rate
RLDRAM II	—	Half rate	Full rate and half rate
QDR II+ SRAM	—	Half rate	Full rate and half rate
QDR II SRAM	—	Half rate	Full rate and half rate

Related Information

- **Arria V Device Handbook: Known Issues**
  Lists the planned updates to the *Arria V Device Handbook* chapters.
- **External Memory Interface Spec Estimator**
  For the latest information and to estimate the external memory system performance specification, use Altera's External Memory Interface Spec Estimator tool.
- **External Memory Interfaces Handbook Volume 1, 2, and 3.**
  Provides more information about the memory types supported, board design guidelines, timing analysis, simulation, and debugging information.

External Memory Performance

Table 7-2: External Memory Interface Performance in Arria V Devices

Interface	Voltage (V)	Hard Controller (MHz)	Soft Controller (MHz)	
		Arria V GX, GT, SX, and ST	Arria V GX, GT, SX, and ST	Arria V GZ
DDR3 SDRAM	1.5	533	667	800
	1.35	533	600	800
DDR2 SDRAM	1.8	400	400	400
LPDDR2 SDRAM	1.2	—	400	—
RLDRAM 3	1.2	—	—	667
RLDRAM II	1.8	—	400	533
	1.5	—	400	533
QDR II+ SRAM	1.8	—	400	500
	1.5	—	400	500
QDR II SRAM	1.8	—	400	333
	1.5	—	400	333
DDR II+ SRAM[23]	1.8	—	400	—
	1.5	—	400	—

Related Information

External Memory Interface Spec Estimator

For the latest information and to estimate the external memory system performance specification, use Altera's External Memory Interface Spec Estimator tool.

HPS External Memory Performance

Table 7-3: HPS External Memory Interface Performance

The hard processor system (HPS) is available in Arria V SoC devices only.

Interface	Voltage (V)	HPS Hard Controller (MHz)
DDR3 SDRAM	1.5	533
	1.35	533
LPDDR2 SDRAM	1.2	333

[23] Not available as Intel IP.

Related Information

External Memory Interface Spec Estimator
For the latest information and to estimate the external memory system performance specification, use Altera's External Memory Interface Spec Estimator tool.

Memory Interface Pin Support in Arria V Devices

In the Arria V devices, the memory interface circuitry is available in every I/O bank that does not support transceivers. The devices offer differential input buffers for differential read-data strobe and clock operations.

The memory clock pins are generated with double data rate input/output (DDRIO) registers.

Related Information

Planning Pin and FPGA Resources chapter, External Memory Interface Handbook
Provides more information about which pins to use for memory clock pins and pin location requirements.

Guideline: Using DQ/DQS Pins

The following list provides guidelines on using the DQ/DQS pins:

- The devices support DQ and DQS signals with DQ bus modes of x4/x8/x9, x16/x18, or x32/x36.
- You can use the DQSn or CQn pins that are not used for clocking as DQ pins.
- If you do not use the DQ/DQS pins for memory interfacing, you can use these pins as user I/Os. However, unused HPS DQ/DQS pins on the Arria V SX and ST devices cannot be used as user I/Os.
- Some pins have multiple functions such as RZQ or DQ. If you need extra RZQ pins, you can use some of the DQ pins as RZQ pins instead.

Note: For the x8, x16/x18, or x32/x36 DQ/DQS groups whose members are used as RZQ pins, Altera recommends that you assign the DQ and DQS pins manually. Otherwise, the Intel Quartus Prime software might not be able to place the DQ and DQS pins, resulting in a "no-fit" error.

Reading the Pin Table

For the maximum number of DQ pins and the exact number per group for a particular Arria V device, refer to the relevant device pin table.

In the pin tables, the DQS and DQSn pins denote the differential data strobe/clock pin pairs, while the CQ and CQn pins denote the complementary echo clock signals. The pin table lists the parity, DM, BWSn, NWSn, ECC, and QVLD pins as DQ pins.

Related Information

- **Planning Pin and FPGA Resources chapter, External Memory Interface Handbook**
  Provides more information about read clock pins usage for QDR II and QDR II+ SRAM, and RLDRAM II interfaces in Arria V GX, GT, SX, and ST devices
- **Arria V Device Pin-Out Files**
  Download the relevant pin tables from this web page.

DQ/DQS Bus Mode Pins for Arria V Devices

The following tables list the pin support per DQ/DQS bus mode, including the DQS/CQ/CQn/QK# and DQSn pins. The maximum number of data pins per group listed in the tables may vary according to the following conditions:

- Single-ended DQS signaling—the maximum number of DQ pins includes parity, data mask, and QVLD pins connected to the DQS bus network.
- Differential or complementary DQS signaling—the maximum number of data pins per group decreases by one. This number may vary per DQ/DQS group in a particular device. Check the pin table for the exact number per group.
- DDR3 and DDR2 interfaces—the maximum number of pins is further reduced for an interface larger than x8 because you require one DQS pin for each x8/x9 group to form the x16/x18 and x32/x36 groups.

Table 7-4: DQ/DQS Bus Mode Pins for Arria V GX, GT, SX, and ST Devices

Mode	DQSn Support	CQn Support	Parity or Data Mask (Optional)	QVLD [24] (Optional)	Data Pins per Group		Notes
					Typical	Maximum	
x4/x8/x9	Yes	Yes	Yes	Yes	4, 8, or 9	11	The x4 mode uses x8/x9 groups.
x16/x18	Yes	Yes	Yes	Yes	16 or 18	23	Two x8 DQ/DQS groups are stitched to create a x16/x18 group, so there are 24 pins in this group.
x32/x36	Yes	Yes	Yes	Yes	32 or 36	47	Four x8 DQ/DQS groups are stitched to create a x32/x36 group, so there are 48 pins in this group.

Table 7-5: DQ/DQS Bus Mode Pins for Arria V GZ Devices

Mode	DQSn Support	CQn Support	Parity or Data Mask (Optional)	QVLD [24] (Optional)	Data Pins per Group		Notes
					Typical	Maximum	
x4	Yes	—	—	—	4	5	If you do not use differential DQS and the group does not have additional signals, the data mask (DM) pin is supported.
x8/x9	Yes	Yes	Yes	Yes	8 or 9	11	Two x4 DQ/DQS groups are stitched to create a x8/x9 group, so there are a total of 12 pins in this group.

[24] The QVLD pin is not used in the UniPHY IP core.

Mode	DQSn Support	CQn Support	Parity or Data Mask (Optional)	QVLD [24] (Optional)	Data Pins per Group		Notes
					Typical	Maximum	
x16/x18	Yes	Yes	Yes	Yes	16 or 18	23	Four x4 DQ/DQS groups are stitched to create a x16/x18 group; so there are a total of 24 pins in this group.
x32/x36	Yes	Yes	Yes	Yes	32 or 36	47	Eight x4 DQ/DQS groups are stitched to create a x32/x36 group, so there are a total of 48 pins in this group.

DQ/DQS Groups in Arria V GX

Table 7-6: Number of DQ/DQS Groups Per Side in Arria V GX Devices

This table lists the DQ/DQS groups for the soft memory controller. For the hard memory controller, you can get the DQ/DQS groups from the pin table of the specific device.

Member Code	Package	Side	x8/x9	x16/x18	x32/x36
A1	672-pin FineLine BGA, Flip Chip	Top	8	3	—
		Bottom	8	3	—
		Right	4	2	—
	896-pin FineLine BGA, Flip Chip	Top	10	3	—
		Bottom	10	3	—
		Right	6	2	—
A3	672-pin FineLine BGA, Flip Chip	Top	8	3	—
		Bottom	8	3	—
		Right	4	2	—
	896-pin FineLine BGA, Flip Chip	Top	10	3	—
		Bottom	10	3	—
		Right	6	2	—
A5	672-pin FineLine BGA, Flip Chip	Top	8	3	1
		Bottom	8	3	1
	896-pin FineLine BGA, Flip Chip	Top	12	5	1
		Bottom	12	5	1
	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2

Member Code	Package	Side	x8/x9	x16/x18	x32/x36
A7	672-pin FineLine BGA, Flip Chip	Top	8	3	1
		Bottom	8	3	1
	896-pin FineLine BGA, Flip Chip	Top	12	5	1
		Bottom	12	5	1
	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
B1	896-pin FineLine BGA, Flip Chip	Top	12	5	1
		Bottom	12	5	1
	1517-pin FineLine BGA, Flip Chip	Top	22	10	4
		Bottom	22	10	4
	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
B3	896-pin FineLine BGA, Flip Chip	Top	12	5	1
		Bottom	12	5	1
	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
	1517-pin FineLine BGA, Flip Chip	Top	22	10	4
		Bottom	22	10	4
B5	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
	1517-pin FineLine BGA, Flip Chip	Top	22	10	4
		Bottom	22	10	4
B7	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
	1517-pin FineLine BGA, Flip Chip	Top	22	10	4
		Bottom	22	10	4

Related Information

Arria V Device Pin-Out Files
Download the relevant pin tables from this web page.

DQ/DQS Groups in Arria V GT

Table 7-7: Number of DQ/DQS Groups Per Side in Arria V GT Devices

This table lists the DQ/DQS groups for the soft memory controller. For the hard memory controller, you can get the DQ/DQS groups from the pin table of the specific device.

Member Code	Package	Side	x8/x9	x16/x18	x32/x36
C3	672-pin FineLine BGA, Flip Chip	Top	8	3	—
		Bottom	8	3	—
		Right	4	2	—
	896-pin FineLine BGA, Flip Chip	Top	10	3	—
		Bottom	10	3	—
		Right	6	2	—
C7	896-pin FineLine BGA, Flip Chip	Top	12	5	1
		Bottom	12	5	1
	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
D3	896-pin FineLine BGA, Flip Chip	Top	12	5	1
		Bottom	12	5	1
	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
	1517-pin FineLine BGA, Flip Chip	Top	22	10	4
		Bottom	22	10	4
D7	1152-pin FineLine BGA, Flip Chip	Top	17	8	2
		Bottom	17	8	2
	1517-pin FineLine BGA, Flip Chip	Top	22	10	4
		Bottom	22	10	4

Related Information

Arria V Device Pin-Out Files

Download the relevant pin tables from this web page.

DQ/DQS Groups in Arria V GZ

Table 7-8: Number of DQ/DQS Groups Per Side in Arria V GZ Devices

Member Code	Package	Side	x4	x8/x9	x16/x18	x32/x36
E1	780-pin FineLine BGA, Flip Chip	Top	28	13	6	2
		Bottom	26	13	6	1
	1152-pin FineLine BGA, Flip Chip	Top	32	15	7	2
		Bottom	34	17	8	2
E3	780-pin FineLine BGA, Flip Chip	Top	28	13	6	2
		Bottom	26	13	6	1
	1152-pin FineLine BGA, Flip Chip	Top	32	15	7	2
		Bottom	34	17	8	2
E5	1152-pin FineLine BGA, Flip Chip	Top	36	17	8	3
		Bottom	50	25	12	4
	1517-pin FineLine BGA, Flip Chip	Top	52	26	12	6
		Bottom	58	29	14	6
E7	1152-pin FineLine BGA, Flip Chip	Top	36	17	8	3
		Bottom	50	25	12	4
	1517-pin FineLine BGA, Flip Chip	Top	52	26	12	6
		Bottom	58	29	14	6

DQ/DQS Groups in Arria V SX

Table 7-9: Number of DQ/DQS Groups Per Side in Arria V SX Devices

This table lists the DQ/DQS groups for the soft memory controller. For the hard memory controller, you can get the DQ/DQS groups from the pin table of the specific device.

Member Code	Package	Side	x8/x9	x16/x18	x32/x36
B3	896-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	6	2	—
	1152-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	16	7	2
	1517-pin FineLine BGA, Flip Chip	Top	11	5	2
		Bottom	22	10	4

Member Code	Package	Side	x8/x9	x16/x18	x32/x36
B5	896-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	6	2	—
	1152-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	16	7	2
	1517-pin FineLine BGA, Flip Chip	Top	11	5	2
		Bottom	22	10	4

Related Information

Arria V Device Pin-Out Files
Download the relevant pin tables from this web page.

DQ/DQS Groups in Arria V ST

Table 7-10: Number of DQ/DQS Groups Per Side in Arria V ST Devices

This table lists the DQ/DQS groups for the soft memory controller. For the hard memory controller, you can get the DQ/DQS groups from the pin table of the specific device.

Member Code	Package	Side	x8/x9	x16/x18	x32/x36
D3	896-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	6	2	—
	1152-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	16	7	2
	1517-pin FineLine BGA, Flip Chip	Top	11	5	2
		Bottom	22	10	4
D5	896-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	6	2	—
	1152-pin FineLine BGA, Flip Chip	Top	7	3	1
		Bottom	16	7	2
	1517-pin FineLine BGA, Flip Chip	Top	11	5	2
		Bottom	22	10	4

Related Information

Arria V Device Pin-Out Files
Download the relevant pin tables from this web page.

External Memory Interface Features in Arria V Devices

The Arria V I/O elements (IOE) provide built-in functionality required for a rapid and robust implementation of external memory interfacing.

The following device features are available for external memory interfaces:

- DQS phase-shift circuitry
- PHY Clock (PHYCLK) networks
- DQS logic block
- Dynamic on-chip termination (OCT) control
- IOE registers
- Delay chains
- Hard memory controllers (Arria V GX, GT, SX, and ST only)
- Read- and write-leveling support (Arria V GZ only)

UniPHY IP

The high-performance memory interface solution includes the self-calibrating UniPHY IP that is optimized to take advantage of the Arria V I/O structure and the Intel Quartus Prime software Timing Analyzer. The UniPHY IP helps set up the physical interface (PHY) best suited for your system. This provides the total solution for the highest reliable frequency of operation across process, voltage, and temperature (PVT) variations.

The UniPHY IP instantiates a PLL to generate related clocks for the memory interface. The UniPHY IP can also dynamically choose the number of delay chains that are required for the system. The amount of delay is equal to the sum of the intrinsic delay of the delay element and the product of the number of delay steps and the value of the delay steps.

The UniPHY IP and the Altera memory controller IP core can run at half the I/O interface frequency of the memory devices, allowing better timing management in high-speed memory interfaces. The Arria V devices contain built-in circuitry in the IOE to convert data from full rate (the I/O frequency) to half rate (the controller frequency) and vice versa.

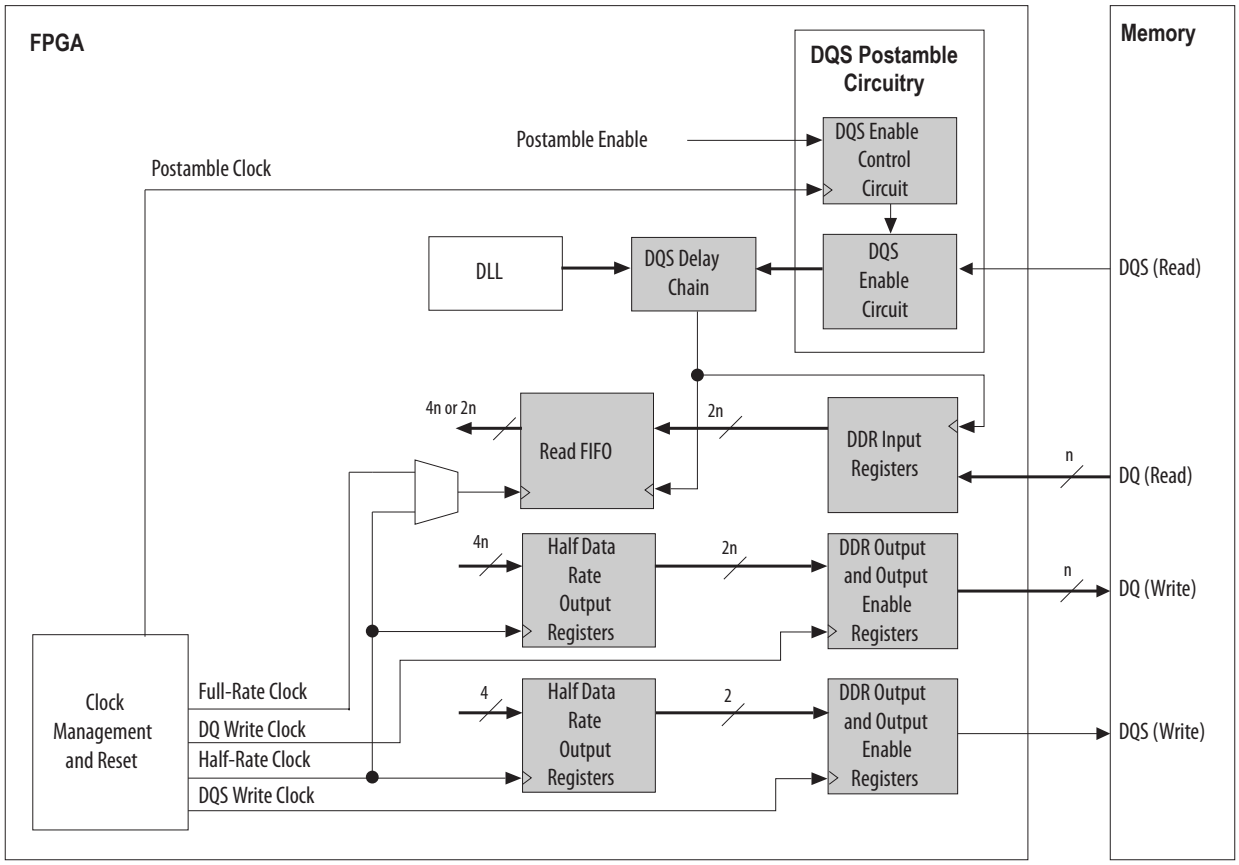
Related Information

Functional Description - UniPHY, External Memory Interface Handbook Volume 3
Provides more information about UniPHY IP.

External Memory Interface Datapath

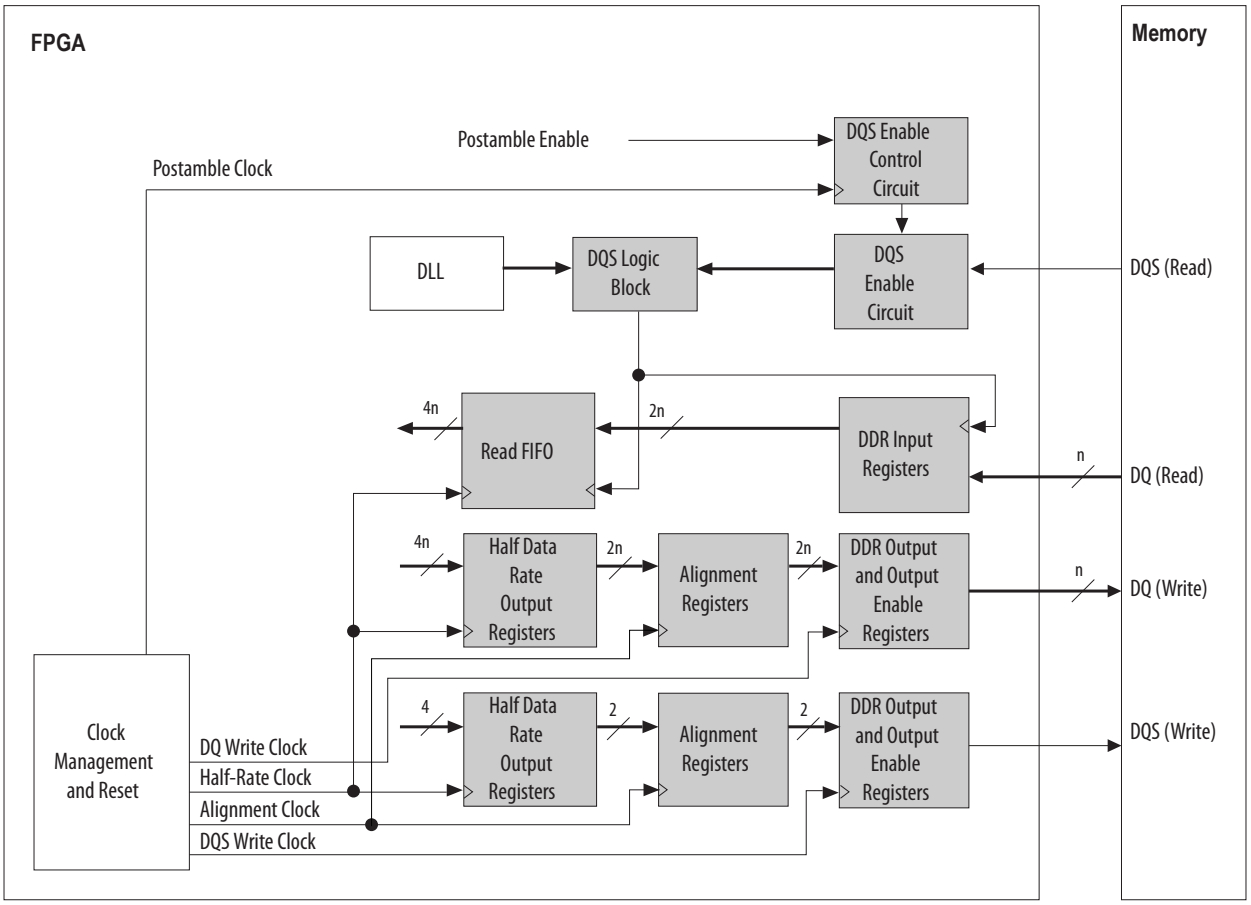
The following figures show overviews of the memory interface datapath that uses the Arria V I/O elements. In the figures, the DQ/DQS read and write signals may be bidirectional or unidirectional, depending on the memory standard. If the signal is bidirectional, it is active during read and write operations.

Figure 7-1: External Memory Interface Datapath Overview for Arria V GX, GT, SX, and ST Devices



Note: There are slight block differences for different memory interface standards. The shaded blocks are part of the I/O elements.

Figure 7-2: External Memory Interface Datapath Overview for Arria V GZ Devices



Note: There are slight block differences for different memory interface standards. The shaded blocks are part of the I/O elements.

DQS Phase-Shift Circuitry

The Arria V DLL provides phase shift to the DQS/CQ/CQn/QK# pins on read transactions if the DQS/CQ/CQn/QK# pins are acting as input clocks or strobes to the FPGA.

The following figures show how the DLLs are connected to the DQS/CQ/CQn/QK# pins in the various Arria V variants.

Figure 7-3: DQS/CQ/CQn/QK# Pins and DLLs in Arria V GX (A1 and A3) Devices

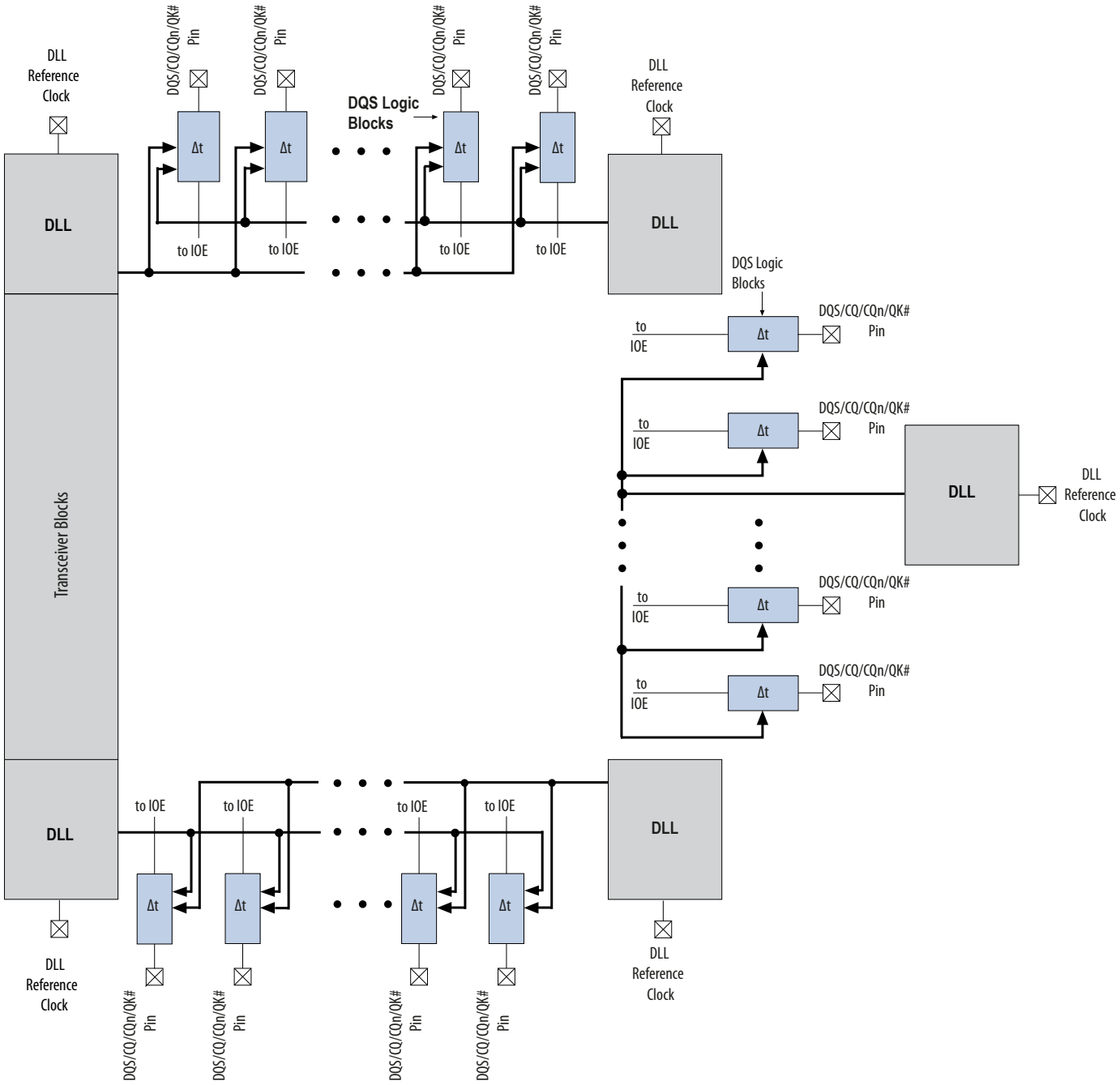


Figure 7-4: DQS/CQ/CQn/QK# Pins and DLLs in Arria V GX (A5, A7, B1, B3, B5, and B7), GT, and GZ Devices

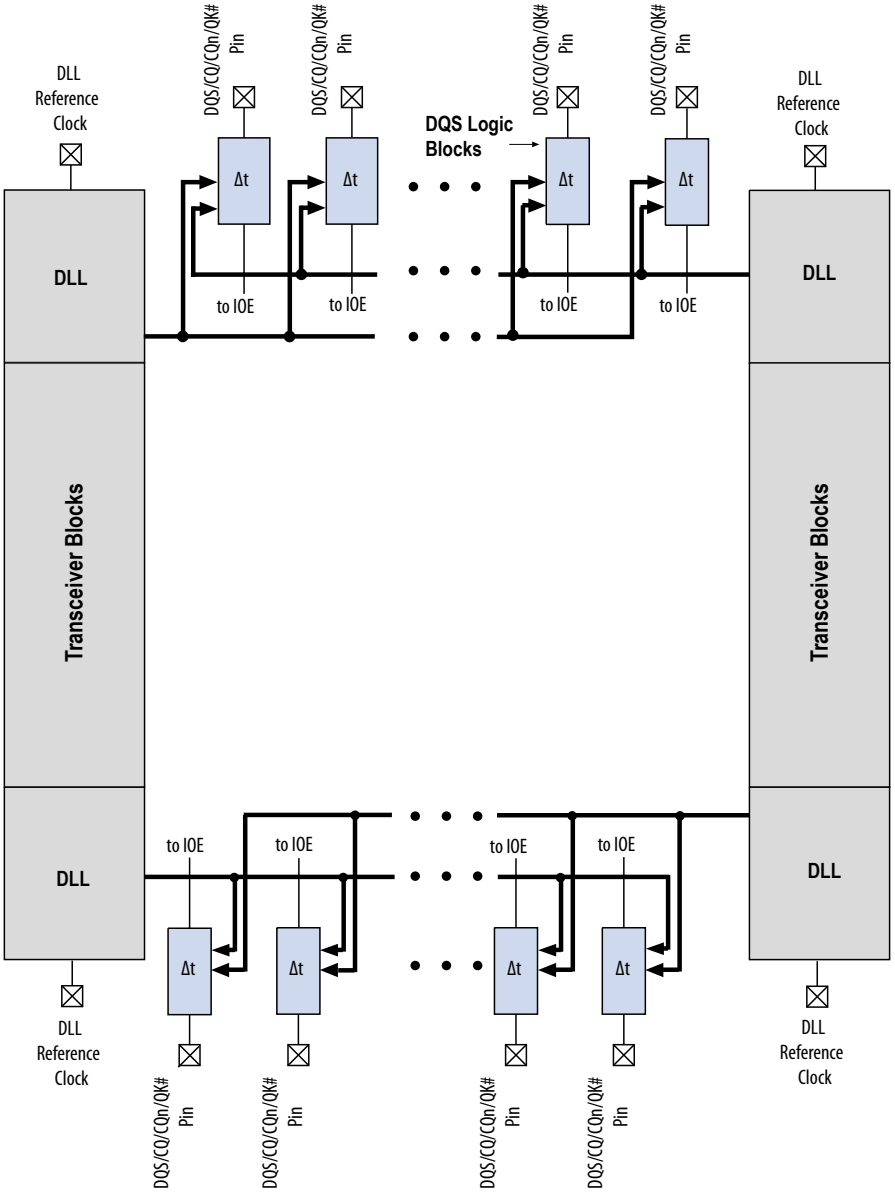
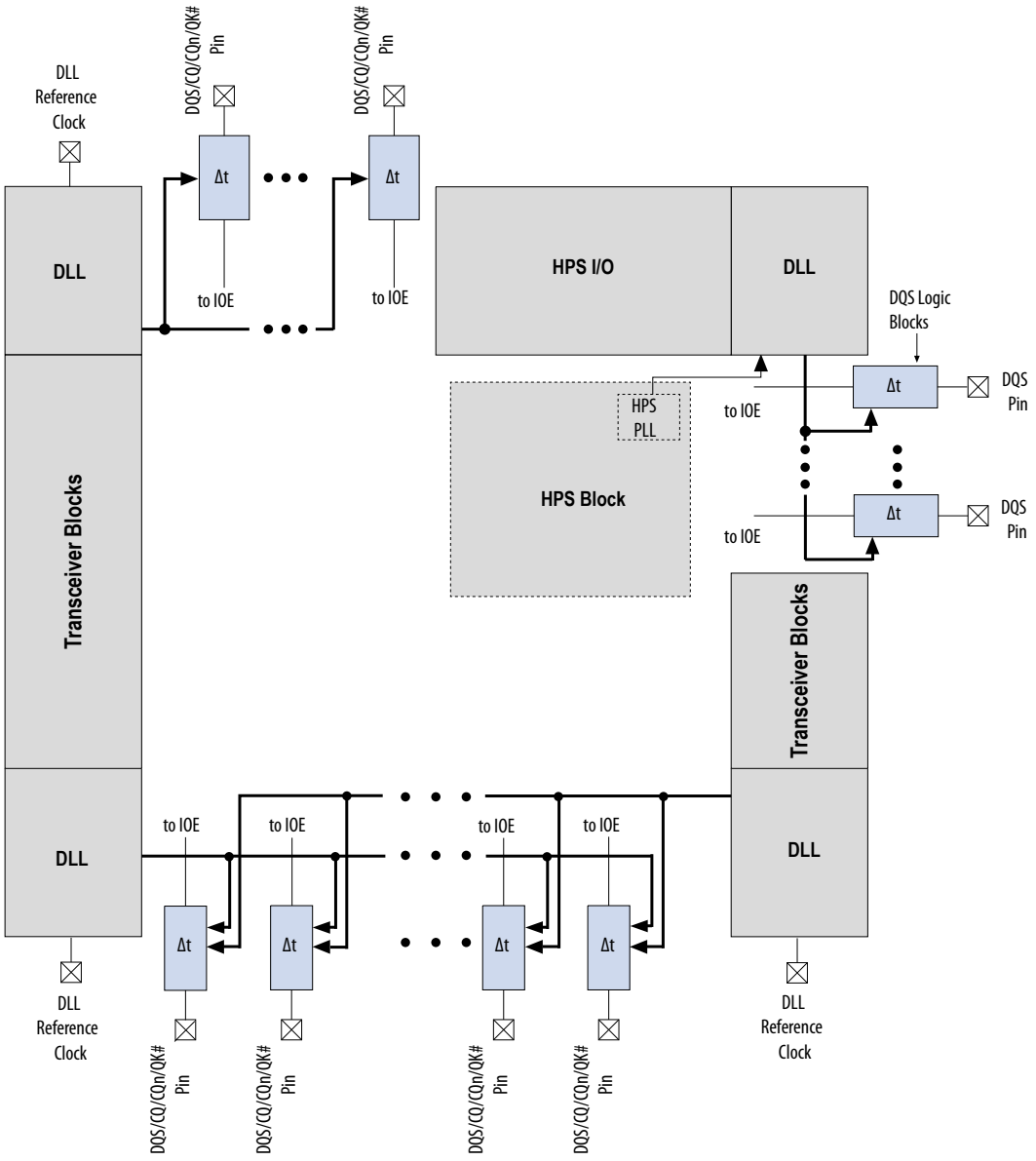


Figure 7-5: DQS/CQ/CQn/QK# Pins and DLLs in Arria V SX and ST Devices



Delay-Locked Loop

The delay-locked loop (DLL) uses a frequency reference to dynamically generate control signals for the delay chains in each of the DQS/CQ/CQn/QK# pins, allowing the delay to compensate for process, voltage, and temperature (PVT) variations. The DQS delay settings are gray-coded to reduce jitter if the DLL updates the settings.

There are a maximum of five DLLs in Arria V devices. You can clock each DLL using different frequencies.

Some of the DLLs can access the two adjacent sides from its location in the device. You can have two different interfaces with the same frequency on the two sides adjacent to a DLL, where the DLL controls the DQS delay settings for both interfaces.

I/O banks between two DLLs have the flexibility to create multiple frequencies and multiple-type interfaces. These banks can use settings from either or both adjacent DLLs. For example, DQS1R can get its phase-shift settings from DLL_TR, while DQS2R can get its phase-shift settings from DLL_BR.

The reference clock for each DLL may come from the PLL output clocks or clock input pins.

Note: If you have a dedicated PLL that only generates the DLL input reference clock, set the PLL mode to **No Compensation** to achieve better performance (or the Intel Quartus Prime software automatically changes it). Because the PLL does not use any other outputs, it does not have to compensate for any clock paths.

DLL Reference Clock Input for Arria V Devices

Table 7-11: DLL Reference Clock Input from PLL Counter Outputs for Arria V GX A1 and A3, and Arria V GT C3 Devices—Preliminary

DLL	PLL				
	1L	0L	RC	TC	BC
DLL_T0	plldout[1:0]	—	—	plldout[1:0]	—
DLL_T1	—	—	—	plldout[1:0]	—
DLL_B0	—	plldout[1:0]	—	—	plldout[1:0]
DLL_B1	—	—	—	—	plldout[1:0]
DLL_R0	—	—	plldout[1:0]	—	—

Table 7-12: DLL Reference Clock Input from PLL Counter Outputs for Arria V GX A5, A7, B1, and B3, and Arria V GT C7 and D3 Devices—Preliminary

DLL	PLL					
	TL	TR	BR	BL	TC	BC
DLL_T0	plldout[1:0]	—	—	—	plldout[1:0]	—
DLL_T1	—	plldout[1:0]	—	—	plldout[1:0]	—
DLL_B0	—	—	—	plldout[1:0]	—	plldout[1:0]

DLL	PLL					
	TL	TR	BR	BL	TC	BC
DLL_B1	—	—	plldout[1:0]	—	—	plldout[1:0]

Table 7-13: DLL Reference Clock Input from PLL Counter Outputs for Arria V GX B5 and B7, and Arria V GT D7 Devices—Preliminary

DLL	PLL					
	2L	2R	0R	0L	TC	BC
DLL_T0	plldout[1:0]	—	—	—	plldout[1:0]	—
DLL_T1	—	plldout[1:0]	—	—	plldout[1:0]	—
DLL_B0	—	—	—	plldout[1:0]	—	plldout[1:0]
DLL_B1	—	—	plldout[1:0]	—	—	plldout[1:0]

Table 7-14: DLL Reference Clock Input for Arria V GZ E1 and E3 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X84_Y77 CEN_X84_Y68	COR_X0_Y81 COR_X0_Y72	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X84_Y77 CEN_X84_Y68	COR_X185_Y81 COR_X185_Y72	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_BR	CEN_X84_Y11 CEN_X84_Y2	COR_X185_Y10 COR_X185_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X84_Y11 CEN_X84_Y2	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-15: DLL Reference Clock Input for Arria V GZ E5 and E7 Devices

DLL	PLL		CLKIN		
	Center	Corner	Left	Center	Right
DLL_TL	CEN_X92_Y96 CEN_X92_Y87	COR_X0_Y100 COR_X0_Y91	CLK20P CLK21P CLK22P CLK23P	CLK16P CLK17P CLK18P CLK19P	—
DLL_TR	CEN_X92_Y96 CEN_X92_Y87	COR_X202_Y100 COR_X202_Y91	—	CLK16P CLK17P CLK18P CLK19P	CLK12P CLK13P CLK14P CLK15P
DLL_BR	CEN_X92_Y11 CEN_X92_Y2	COR_X202_Y10 COR_X202_Y1	—	CLK4P CLK5P CLK6P CLK7P	CLK8P CLK9P CLK10P CLK11P
DLL_BL	CEN_X92_Y11 CEN_X92_Y1	COR_X0_Y10 COR_X0_Y1	CLK0P CLK1P CLK2P CLK3P	CLK4P CLK5P CLK6P CLK7P	—

Table 7-16: DLL Reference Clock Input from PLL Counter Outputs for Arria V SX B3 and B5, and Arria V ST D3 and D5 Devices—Preliminary

DLL	PLL				
	2L	0R	0L	TC	BC
DLL_T0	plldout[1:0]	—	—	plldout[1:0]	—
DLL_B0	—	—	plldout[1:0]	—	plldout[1:0]
DLL_B1	—	plldout[1:0]	—	—	plldout[1:0]

DLL Phase-Shift

The DLL can shift the incoming DQS signals by 0° or 90° by using two delay cells in the DQS logic block. The shifted DQS signal is then used as the clock for the DQ IOE input registers.

All DQS/CQ/CQn/QK# pins referenced to the same DLL, can have their input signal phase shifted by a different degree amount but all must be referenced at one particular frequency. However, not all phase-shift combinations are supported.

The 7-bit DQS delay settings from the DLL vary with PVT to implement the phase-shift delay. For example, with a 0° shift, the DQS/CQ/CQn/QK# signal bypasses both the DLL and DQS logic blocks. The Intel Quartus Prime software automatically sets the DQ input delay chains, so that the skew between the DQ and DQS/CQ/CQn/QK# pins at the DQ IOE registers is negligible if a 0° shift is implemented. You can feed the DQS delay settings to the DQS logic block and logic array.

The shifted DQS/CQ/CQn/QK# signal goes to the DQS bus to clock the IOE input registers of the DQ pins. The signal can also go into the logic array for resynchronization if you are not using IOE read FIFO for resynchronization.

For Arria V SoC devices, you can feed the hard processor system (HPS) DQS delay settings to the HPS DQS logic block only.

Figure 7-6: Simplified Diagram of the DQS Phase-Shift Circuitry (Arria V GZ, GT, SX, and ST)

This figure shows a simple block diagram of the DLL in Arria V GZ devices. All features of the DQS phase-shift circuitry are accessible from the UniPHY IP core in the Intel Quartus Prime software.

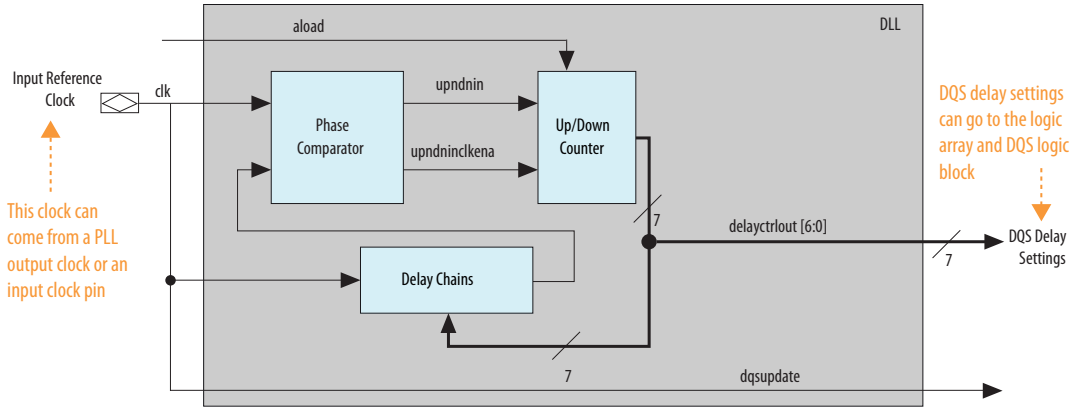
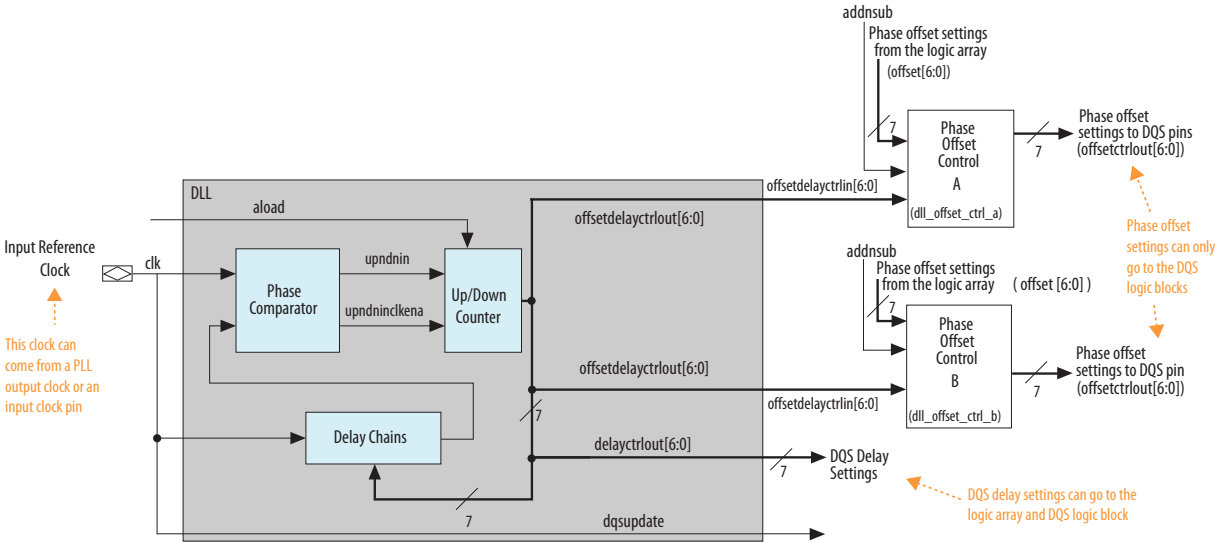


Figure 7-7: Simplified Diagram of the DQS Phase-Shift Circuitry (Arria V GZ)

This figure shows a simple block diagram of the DLL in Arria V GZ devices. All features of the DQS phase-shift circuitry are accessible from the UniPHY IP core in the Intel Quartus Prime software.



The input reference clock goes into the DLL to a chain of up to eight delay elements. The phase comparator compares the signal coming out of the end of the delay chain block to the input reference clock. The phase comparator then issues the upndn signal to the Gray-code counter. This signal increments or decrements a 7-bit delay setting (DQS delay settings) that increases or decreases the delay through the delay element chain to bring the input reference clock and the signals coming out of the delay element chain in phase.

The DLL can be reset from either the logic array or a user I/O pin. Each time the DLL is reset, you must wait for 2,560 clock cycles for the DLL to lock before you can capture the data properly. The DLL phase comparator requires 2,560 clock cycles to lock and calculate the correct input clock period.

For the frequency range of each DLL frequency mode, refer to the device datasheet.

Related Information

- **Arria V GX, GT, SX, and ST Device Datasheet**
- **Arria V GZ Device Datasheet**

PHY Clock (PHYCLK) Networks

The PHYCLK network is a dedicated high-speed, low-skew balanced clock tree designed for a high-performance external memory interface.

The top and bottom sides of the Arria V devices have up to four PHYCLK networks. There are up to two PHYCLK networks on the left and right side I/O banks. Each PHYCLK network spans across one I/O bank and is driven by one of the PLLs located adjacent to the I/O bank.

The following figures show the PHYCLK networks available in the Arria V devices.

Figure 7-8: PHYCLK Networks in Arria V GX A1 and A3 Devices

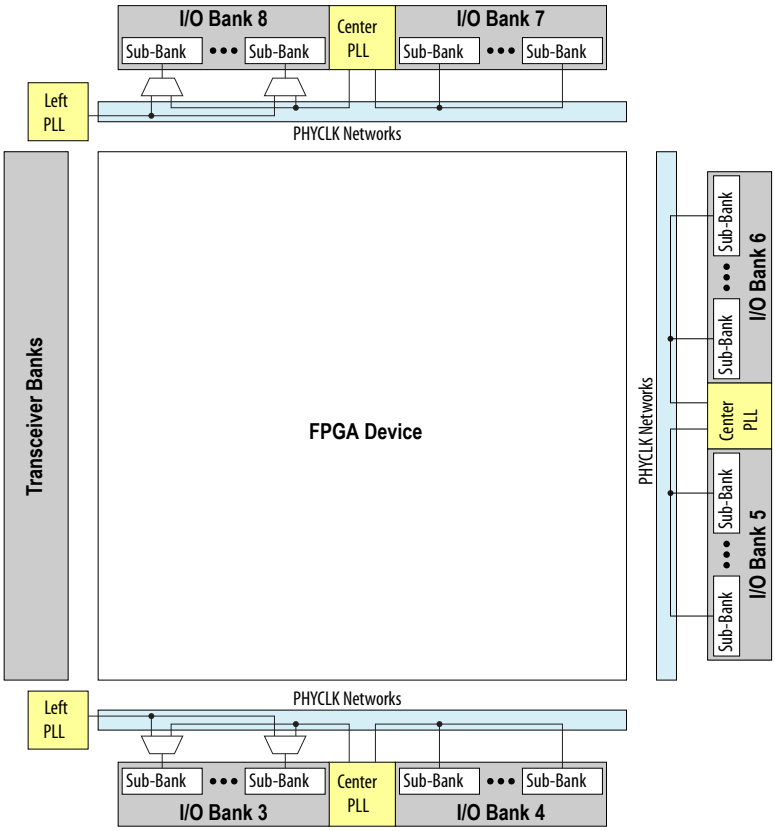


Figure 7-9: PHYCLK Networks in Arria V GX A5, A7, B1, B3, B5, and B7 Devices, and Arria V GZ E1, E3, E5, and E7 Devices

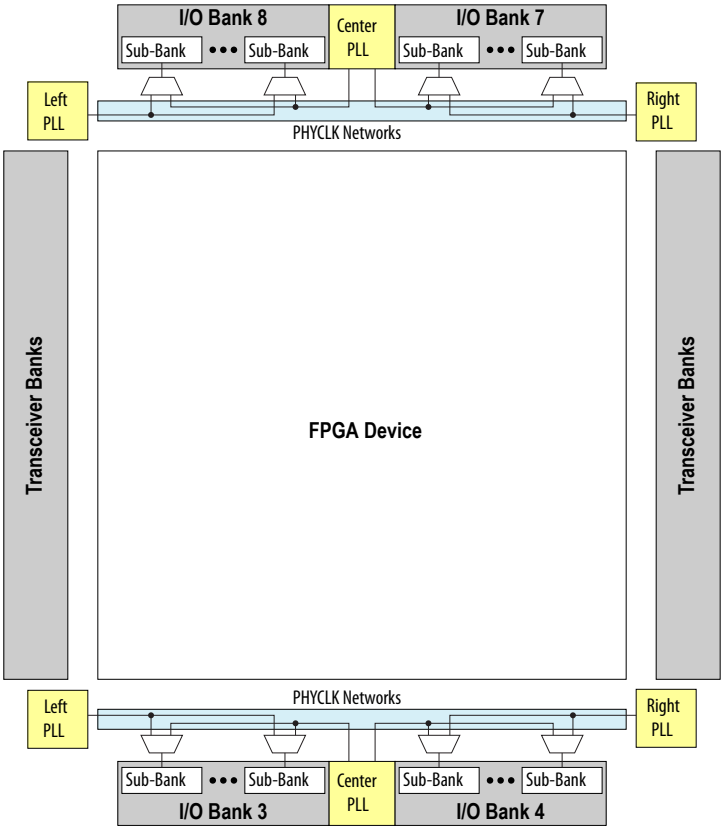
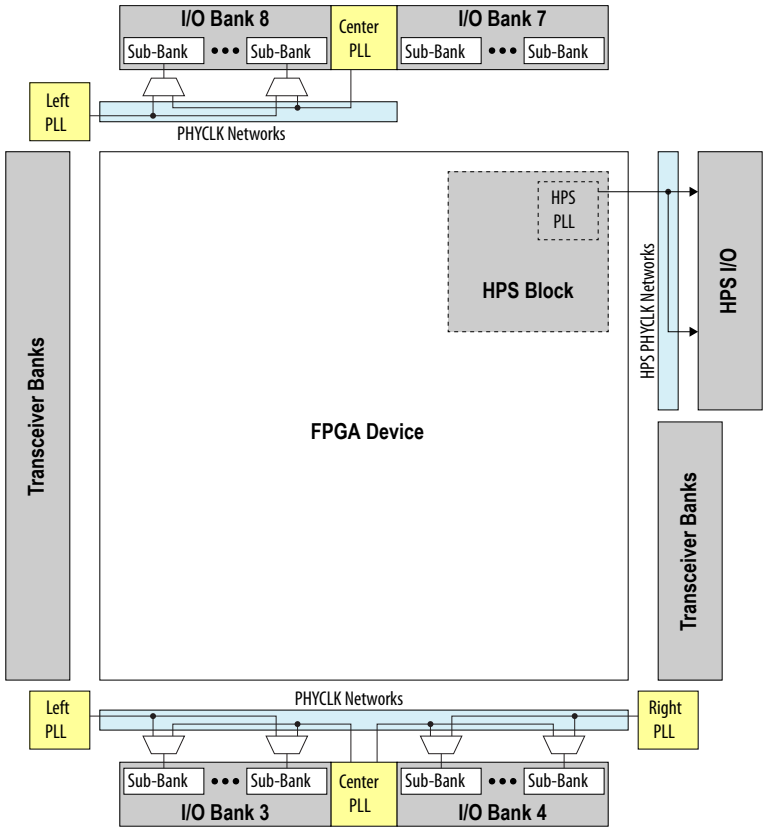


Figure 7-10: Arria V PHYCLK Networks in Arria V SX B3 and B5 Devices, and Arria V ST D3 and D5 Devices



The PHYCLK network can be used to drive I/O sub-banks in each I/O bank. Each I/O sub-bank can be driven by only one PHYCLK network—all I/O pins in an I/O sub-bank are driven by the same PHYCLK network.

DQS Logic Block

Each DQS/CQ/CQn/QK# pin is connected to a separate DQS logic block, which consists of the update enable circuitry, DQS delay chains, and DQS postamble circuitry.

The following figure shows the DQS logic block.

Figure 7-11: DQS Logic Block in Arria V GX, GT, SX, and ST Devices

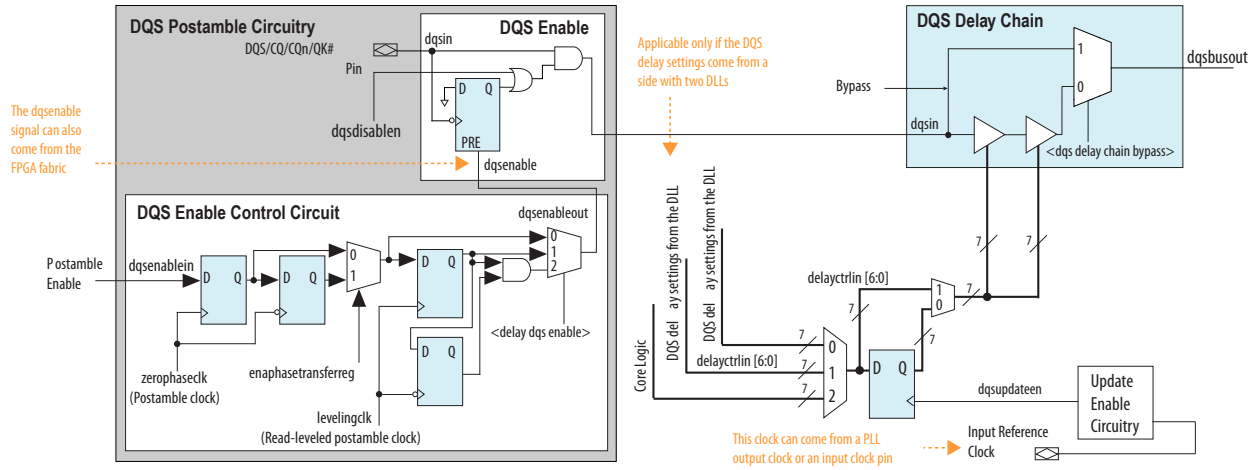
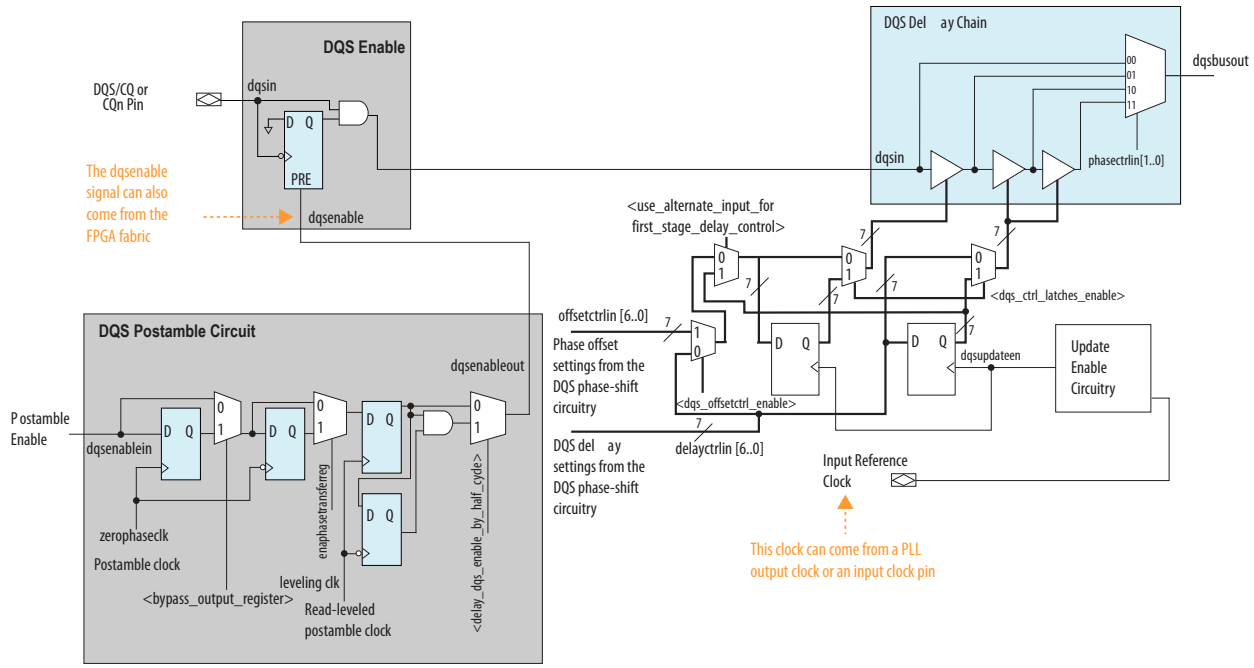


Figure 7-12: DQS Logic Block in Arria V GZ Devices



Update Enable Circuitry

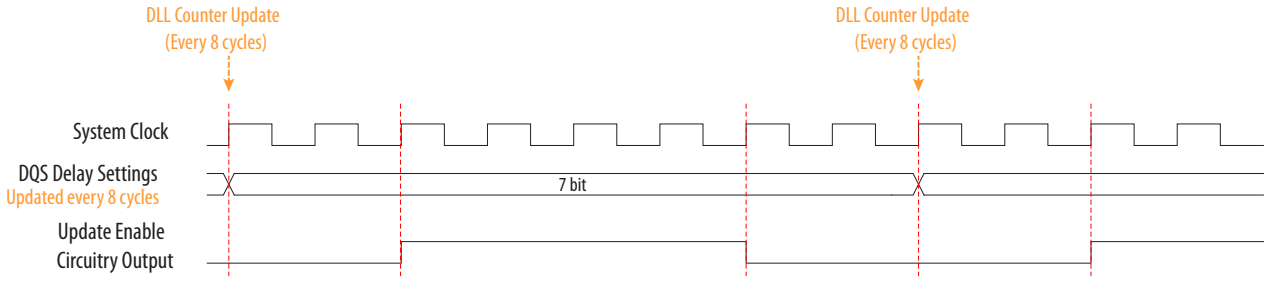
The update enable circuitry enables the registers to allow enough time for the DQS delay settings to travel from the DQS phase-shift circuitry or core logic to all the DQS logic blocks before the next change.

Both the DQS delay settings and the phase-offset settings pass through a register before going into the DQS delay chains. The registers are controlled by the update enable circuitry to allow enough time for any changes in the DQS delay setting bits to arrive at all the delay elements, which allows them to be adjusted at the same time.

The circuitry uses the input reference clock or a user clock from the core to generate the update enable output. The UniPHY intellectual property (IP) uses this circuit by default.

Figure 7-13: DQS Update Enable Waveform

This figure shows an example waveform of the update enable circuitry output.



DQS Delay Chain

DQS delay chains consist of a set of variable delay elements to allow the input DQS/CQ/CQn/QK# signals to be shifted by the amount specified by the DQS phase-shift circuitry or the logic array.

There are two delay elements in the DQS delay chain that have the same characteristics:

- Delay elements in the DQS logic block
- Delay elements in the DLL

The DQS/CQ/CQn/QK# pin is shifted by the DQS delay settings.

The number of delay chains required is transparent because the UniPHY IP automatically sets it when you choose the operating frequency.

In Arria V GX, GT, and GZ devices, if you do not use the DLL to control the DQS delay chains, you can input your own gray-coded 7 bit settings using the `delayctrlin[6..0]` signals available in the UniPHY IP.

In the Arria V SX and ST devices, the DQS delay chain is controlled by the DQS phase-shift circuitry only.

Related Information

- **ALTDQ_DQS2 IP Core User Guide**
  Provides more information about programming the delay chains.
- **Delay Chains** on page 7-31

DQS Postamble Circuitry

There are preamble and postamble specifications for both read and write operations in DDR3 and DDR2 SDRAM. The DQS postamble circuitry ensures that data is not lost if there is noise on the DQS line during the end of a read operation that occurs while DQS is in a postamble state.

The Arria V devices contain dedicated postamble registers that you can control to ground the shifted DQS signal that is used to clock the DQ input registers at the end of a read operation. This function ensures that any glitches on the DQS input signal during the end of a read operation and occurring while DQS is in a postamble state do not affect the DQ IOE registers.

- For preamble state, the DQS is low, just after a high-impedance state.
- For postamble state, the DQS is low, just before it returns to a high-impedance state.

For external memory interfaces that use a bidirectional read strobe (DDR3 and DDR2 SDRAM), the DQS signal is low before going to or coming from a high-impedance state.

Half Data Rate Block

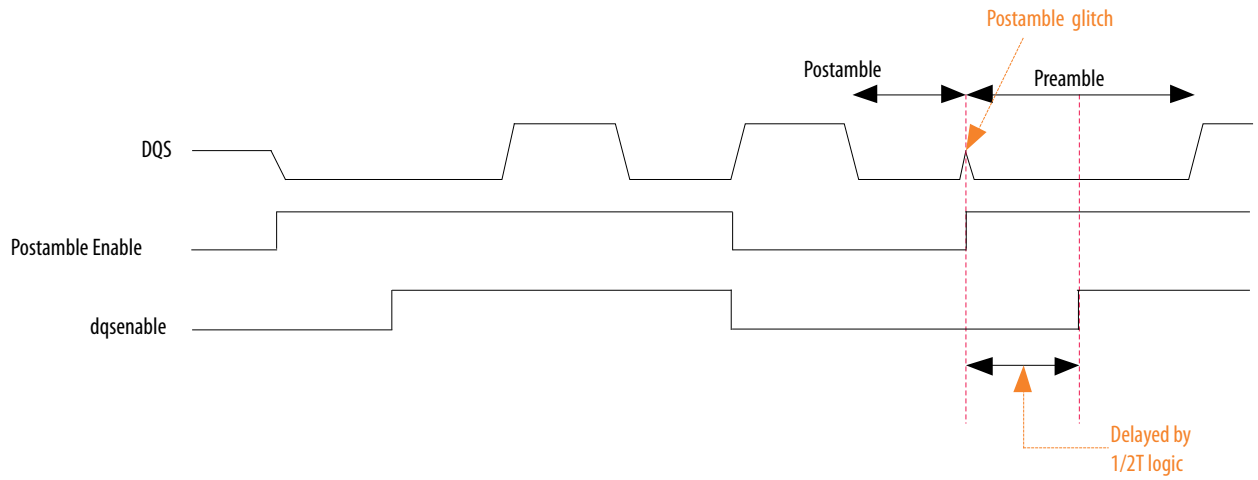
The Arria V devices contain a half data rate (HDR) block in the postamble enable circuitry.

The HDR block is clocked by the half-rate resynchronization clock, which is the output of the I/O clock divider circuit. There is an AND gate after the postamble register outputs to avoid postamble glitches from a previous read burst on a non-consecutive read burst. This scheme allows half-a-clock cycle latency for `dqsenable` assertion and zero latency for `dqsenable` deassertion.

Using the HDR block as the first stage capture register in the postamble enable circuitry block is optional. Altera recommends using these registers if the controller is running at half the frequency of the I/Os.

Figure 7-14: Avoiding Glitch on a Non-Consecutive Read Burst Waveform

This figure shows how to avoid postamble glitches using the HDR block.

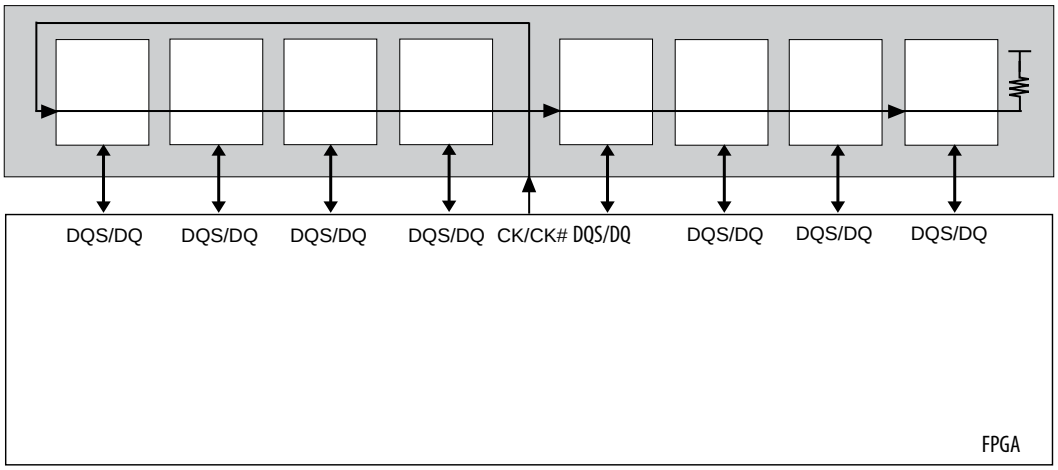


Leveling Circuitry for Arria V GZ Devices

DDR3 SDRAM unbuffered modules use a fly-by clock distribution topology for better signal integrity. This means that the CK/CK# signals arrive at each DDR3 SDRAM device in the module at different times. The difference in arrival time between the first DDR3 SDRAM device and the last device on the module can be as long as 1.6 ns.

The following figure shows the clock topology in DDR3 SDRAM unbuffered modules.

Figure 7-15: DDR3 SDRAM Unbuffered Module Clock Topology



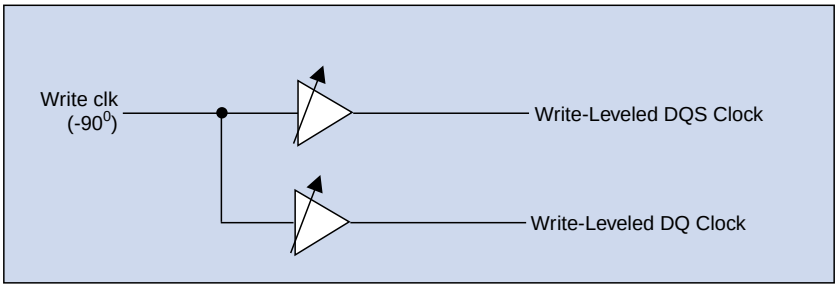
Because the data and read strobe signals are still point-to-point, take special care to ensure that the timing relationship between the CK/CK# and DQS signals (t_{DQSS}, t_{DSS}, and t_{DSH}) during a write is met at every device on the modules. In a similar way, read data coming back into the FPGA from the memory is also staggered.

The Arria V GZ devices have leveling circuitry to address these two situations. There is one leveling circuit per I/O sub-bank (for example, I/O sub-bank 1A, 1B, and 1C each has one leveling circuitry). These delay chains are PVT-compensated by the same DQS delay settings as the DLL and DQS delay chains.

The DLL uses eight delay chain taps, such that each delay chain tap generates a 45° delay. The generated clock phases are distributed to every DQS logic block that is available in the I/O sub-bank. The delay chain taps then feed a multiplexer controlled by the UniPHY IP core to select which clock phases are to be used for that x4 or x 8 DQS group. Each group can use a different tap output from the read-leveling and write-leveling delay chains to compensate for the different CK/CK# delay going into each device on the module.

Figure 7-16: Write-Leveling Delay Chains and Multiplexers

There is one leveling delay chain per I/O sub-bank (for example, I/O sub-banks 1A, 1B, and 1C). You can only have one memory interface in each I/O sub-bank when you use the leveling delay chain.



The –90° write clock of the UniPHY IP feeds the write-leveling circuitry to produce the clock to generate the DQS and DQ signals. During initialization, the UniPHY IP picks the correct write-leveled clock for the

DQS and DQ clocks for each DQ/DQS group after sweeping all the available clocks in the write calibration process. The DQ clock output is –90° phase-shifted compared to the DQS clock output.

The UniPHY IP dynamically calibrates the alignment for read and write leveling during the initialization process.

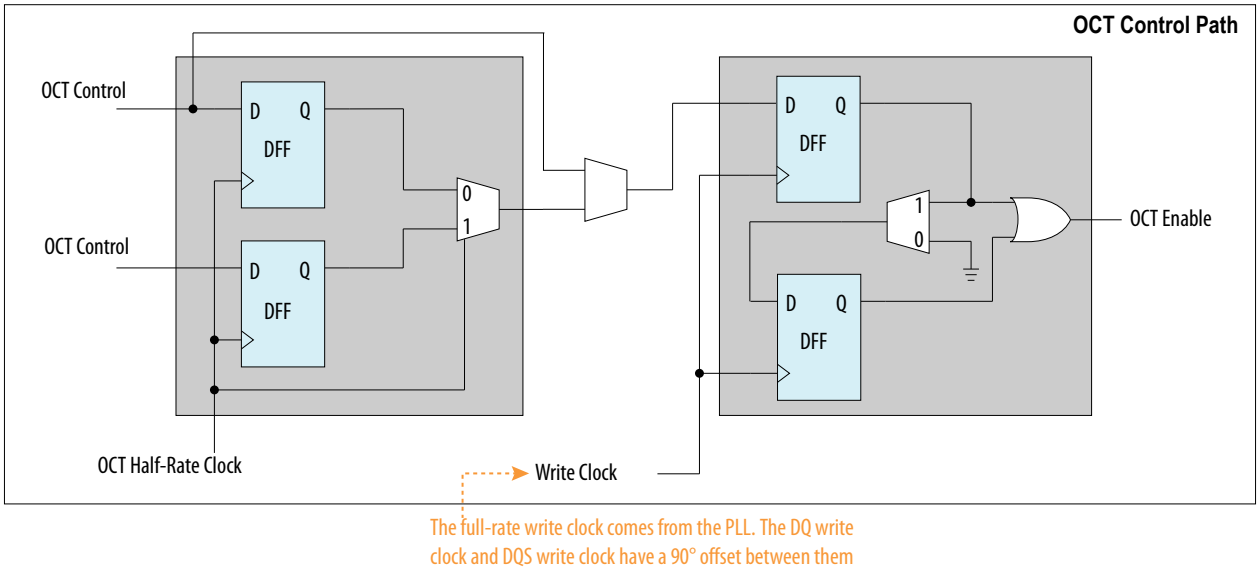
Related Information

- **Functional Description - UniPHY, External Memory Interface Handbook Volume 3**
  Provides more information about UniPHY IP.
- **DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines chapter, External Memory Interface Handbook Volume 2**
  Provides layout guidelines for DDR3 SDRAM interface.

Dynamic OCT Control

The dynamic OCT control block includes all the registers that are required to dynamically turn the on-chip parallel termination (R_T OCT) on during a read and turn R_T OCT off during a write.

Figure 7-17: Dynamic OCT Control Block for Arria V Devices



Related Information

Dynamic OCT in Arria V Devices on page 5-40
Provides more information about dynamic OCT control.

IOE Registers

The IOE registers are expanded to allow source-synchronous systems to have faster register-to-FIFO transfers and resynchronization. All top, bottom, and right IOEs have the same capability.

Input Registers

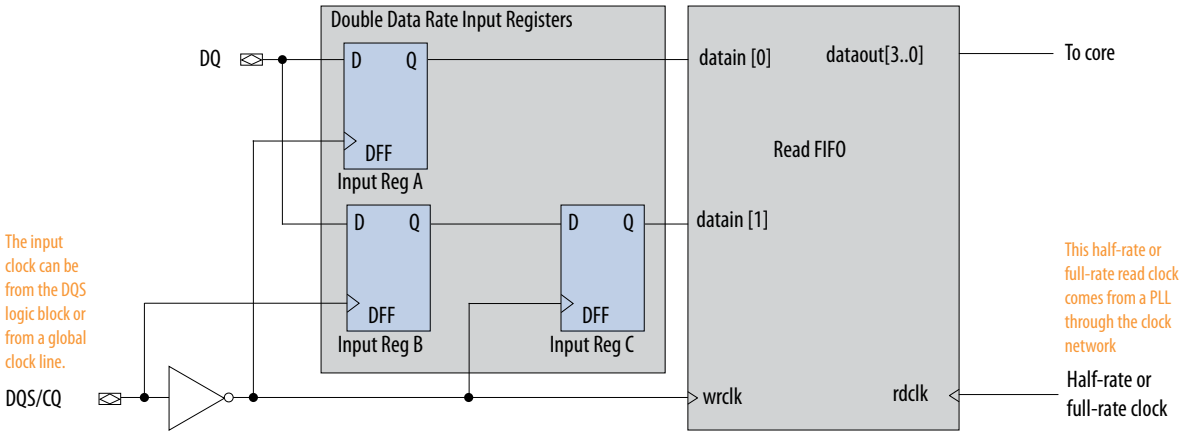
The input path consists of the DDR input registers and the read FIFO block. You can bypass each block of the input path.

There are three registers in the DDR input registers block. Registers A and B capture data on the positive and negative edges of the clock while register C aligns the captured data. Register C uses the same clock as Register A.

The read FIFO block resynchronizes the data to the system clock domain and lowers the data rate to half rate.

The following figure shows the registers available in the Arria V input path. For DDR3 and DDR2 SDRAM interfaces, the DQS and DQSn signals must be inverted. If you use Altera's memory interface IPs, the DQS and DQSn signals are automatically inverted.

Figure 7-18: IOE Input Registers for Arria V Devices



Output Registers

The Arria V output and output-enable path is divided into the HDR block, and output and output-enable registers. The device can bypass each block of the output and output-enable path.

The output path is designed to route combinatorial or registered single data rate (SDR) outputs and full-rate or half-rate DDR outputs from the FPGA core. Half-rate data is converted to full-rate with the HDR block, clocked by the half-rate clock from the PLL.

The output-enable path has a structure similar to the output path—ensuring that the output-enable path goes through the same delay and latency as the output path.

Figure 7-19: IOE Output and Output-Enable Path Registers for Arria V GX, GT, SX, and ST Devices

The following figure shows the registers available in the Arria V GX, GT, SX, and ST output and output-enable paths.

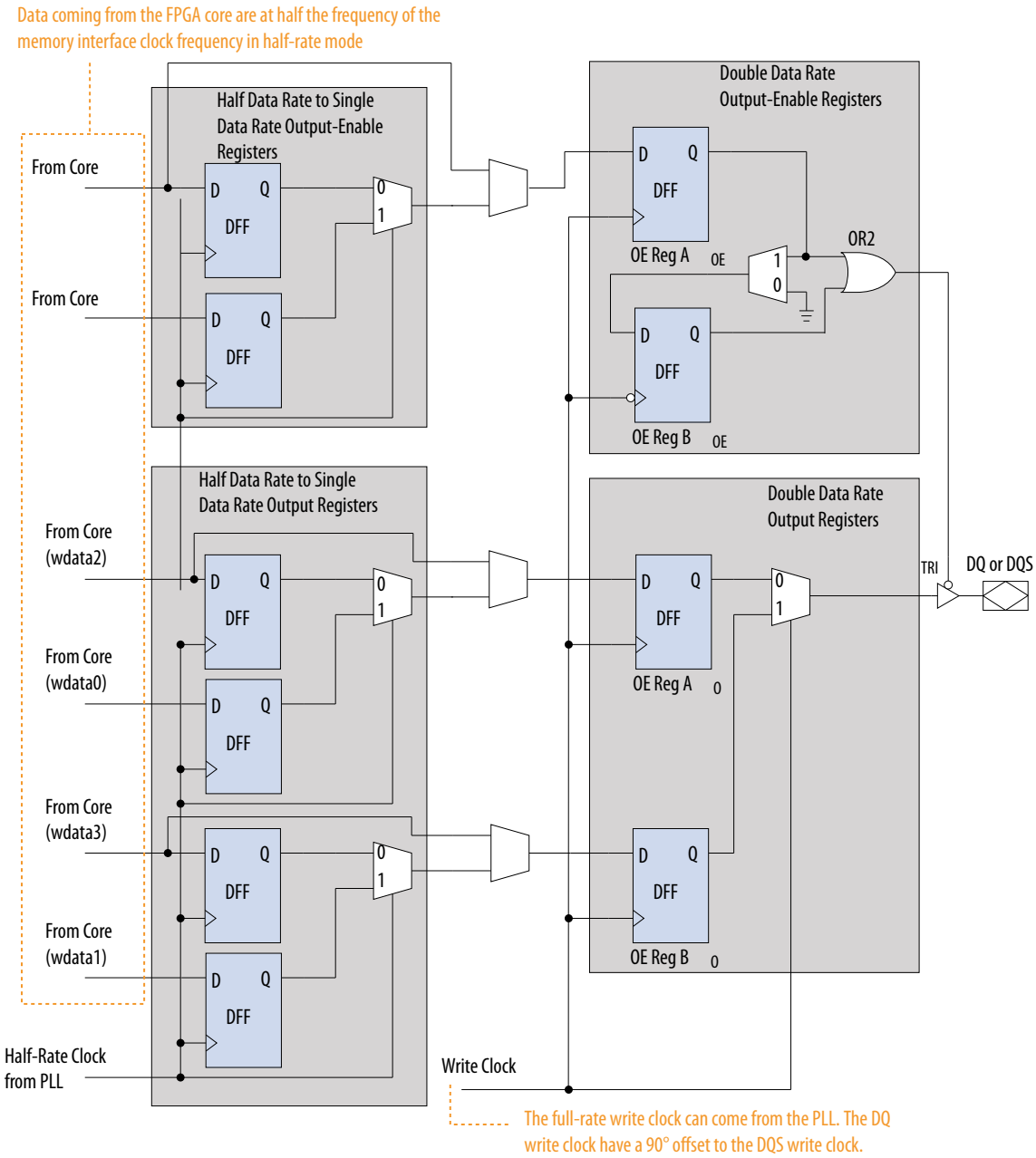
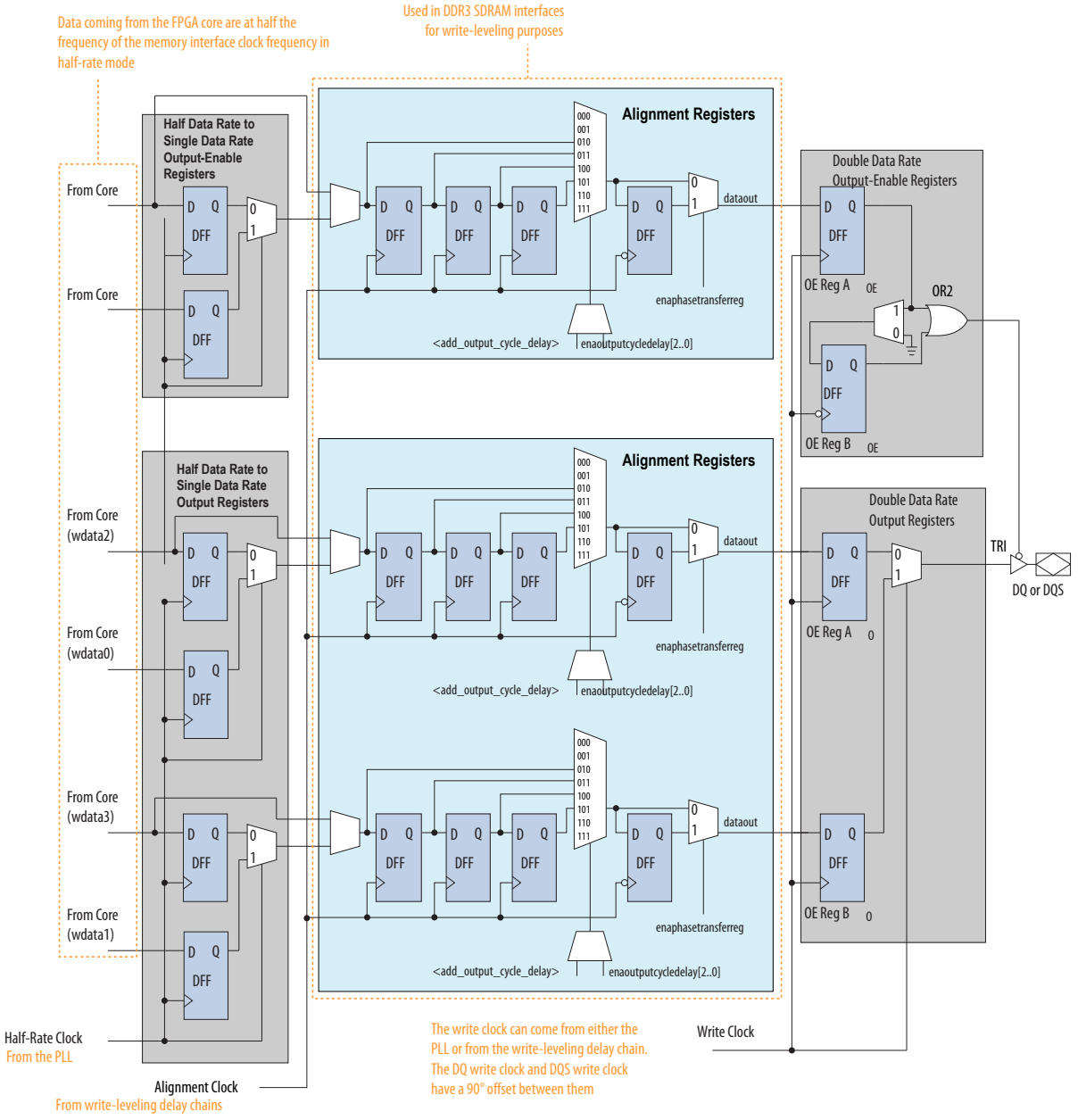


Figure 7-20: IOE Output and Output-Enable Path Registers for Arria V GZ Devices

The following figure shows the registers available in the Arria V GZ output and output-enable paths. You can bypass each register block of the output and output-enable paths.



Delay Chains

The Arria V devices contain run-time adjustable delay chains in the I/O blocks and the DQS logic blocks. You can control the delay chain setting through the I/O or the DQS configuration block output.

Every I/O block contains a delay chain between the following elements:

- The output registers and output buffer
- The input buffer and input register
- The output enable and output buffer
- The R_T OCT enable-control register and output buffer

Figure 7-21: Delay Chains in an I/O Block in the Arria V GX, GT, SX, and ST Devices

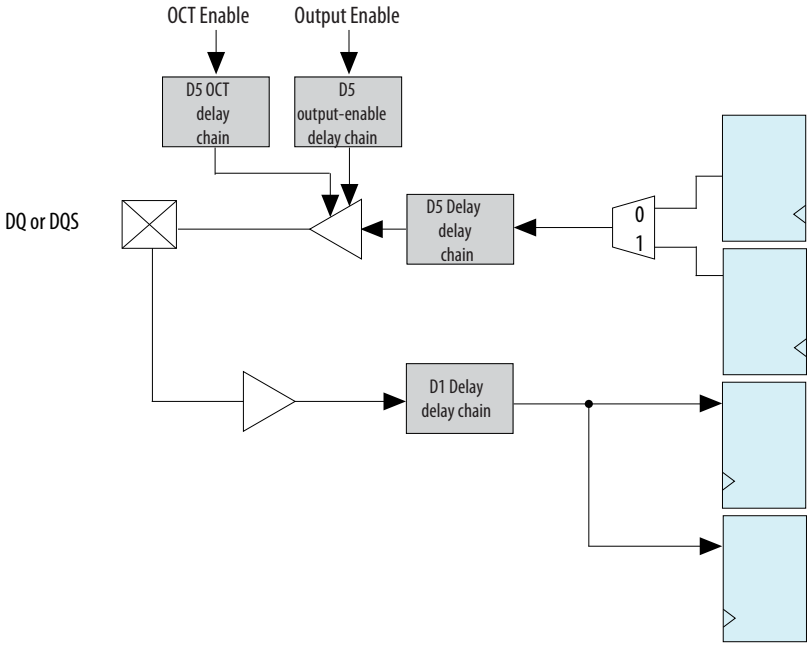
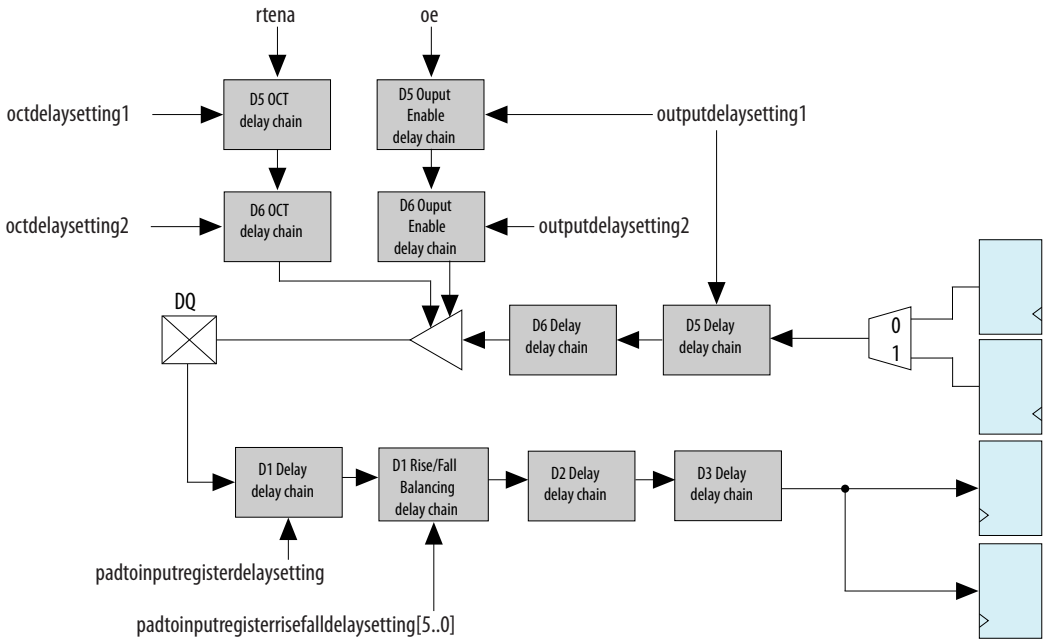
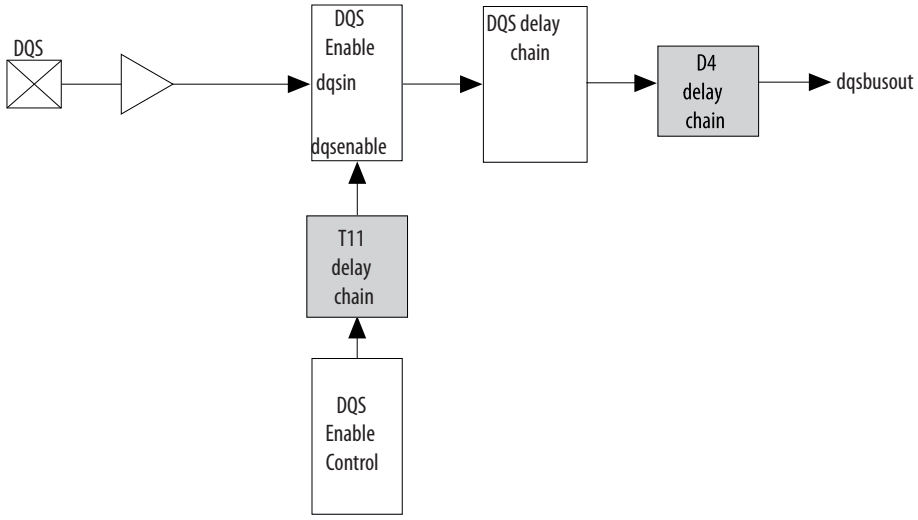


Figure 7-22: Delay Chains in an I/O Block in Arria V GZ Devices



Each DQS logic block contains a delay chain after the `dqsbusout` output and another delay chain before the `dqsenable` input.

Figure 7-23: Delay Chains in the DQS Input Path



Related Information

- **ALTDQ_DQS2 IP Core User Guide**
  Provides more information about programming the delay chains.
- **DQS Delay Chain** on page 7-25

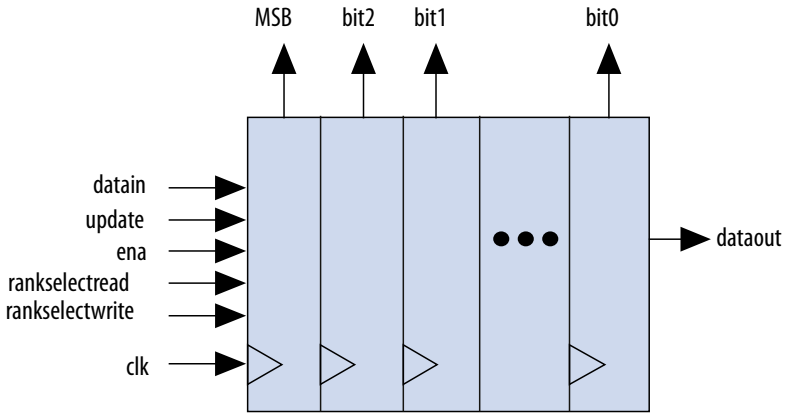
I/O and DQS Configuration Blocks

The I/O and DQS configuration blocks are shift registers that you can use to dynamically change the settings of various device configuration bits.

- The shift registers power-up low.
- Every I/O pin contains one I/O configuration register.
- Every DQS pin contains one DQS configuration block in addition to the I/O configuration register.

Figure 7-24: Configuration Block (I/O and DQS)

This figure shows the I/O configuration block and the DQS configuration block circuitry.



Related Information

ALTDQ_DQS2 IP Core User Guide
Provides more information about programming the delay chains.

Hard Memory Controller

The Arria V GX, GT, SX, and ST devices feature dedicated hard memory controllers. You can use the hard memory controllers for DDR2 and DDR3 SDRAM interfaces. Compared to the memory controllers implemented using core logic, the hard memory controllers allow support for higher memory interface frequencies with shorter latency cycles.

The hard memory controllers use dedicated I/O pins as data, address, command, control, clock, and ground pins for the SDRAM interface. If you do not use the hard memory controllers, you can use these dedicated pins as regular I/O pins.

Note: There is no hard memory controller in the Arria V GZ devices.

Related Information

- **Functional Description - HPC II Controller Chapter, External**
  The hard memory controller is functionally similar to the High-Performance Controller II (HPC II).
- **Functional Description - HPS Memory Controller, External Memory Interface Handbook Volume 3**
  Provides detailed information about application of the hard memory interface.

Features of the Hard Memory Controller

Table 7-17: Features of the Arria V Hard Memory Controller

Feature	Description
Memory Interface Data Width	• 8, 16, and 32 bit data • 16 bit data + 8 bit ECC • 32 bit data + 8bit ECC
Memory Density	The controller supports up to four gigabits density parts and two chip selects.
Memory Burst Length	• DDR3—Burst length of 8 and burst chop of 4 • DDR2—Burst lengths of 4 and 8
Command and Data Reordering	The controller increases efficiency through the support for out-of-order execution of DRAM commands—with address collision detection-and in-order return of results.
Starvation Control	A starvation counter ensures that all requests are served after a predefined time-out period. This function ensures that data with low priority access are not left behind when reordering data for efficiency.
User-Configurable Priority Support	When the controller detects a high priority request, it allows the request to bypass the current queuing request. This request is processed immediately and thus reduces latency.
Avalon®-MM Data Slave Local Interface	By default, the controller supports the Avalon Memory-Mapped protocol.
Bank Management	By default, the controller provides closed-page bank management on every access. The controller intelligently keeps a row open based on incoming traffic. This feature improves the efficiency of the controller especially for random traffic.
Streaming Reads and Writes	The controller can issue reads or writes continuously to sequential addresses every clock cycle if the bank is open. This function allows for very high efficiencies with large amounts of data.
Bank Interleaving	The controller can issue reads or writes continuously to 'random' addresses.

Feature	Description
Predictive Bank Management	The controller can issue bank management commands early so that the correct row is open when the read or write occurs. This increases efficiency.
Multiport Interface	The interface allows you to connect up to six data masters to access the memory controller through the local interface. You can update the multiport scheduling configuration without interrupting traffic on a port.
Built-in Burst Adaptor	The controller can accept bursts of arbitrary sizes on its local interface and map these bursts to efficient memory commands.
Run-time Configuration of the Controller	This feature provides support for updates to the timing parameters without requiring reconfiguration of the FPGA, apart from the standard compile-time setting of the timing parameters.
On-Die Termination	The controller controls the on-die termination (ODT) in the memory, which improves signal integrity and simplifies your board design.
User-Controlled Refresh Timing	You can optionally control when refreshes occur—allowing the refreshes to avoid clashing of important reads or writes with the refresh lock-out time.
Low Power Modes	You can optionally request the controller to put the memory into the self-refresh or deep power-down modes.
Partial Array Self-Refresh	You can select the region of memory to refresh during self-refresh through the mode register to save power.
ECC	Standard Hamming single error correction, double error detection (SECDED) error correction code (ECC) support: • 32 bit data + 8 bit ECC • 16 bit data + 8 bit ECC
Additive Latency	With additive latency, the controller can issue a READ/WRITE command after the ACTIVATE command to the bank prior to t_{RCD} to increase the command efficiency. **Caution:** Efficiency degradation may occur when using the additive latency feature with the hard memory controller for DDR3 SDRAM interfaces at 533 MHz.
Write Acknowledgment	The controller supports write acknowledgment on the local interface.

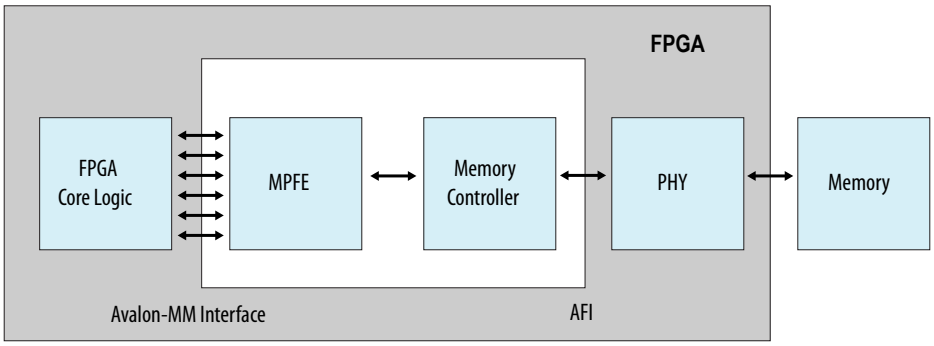
Feature	Description
User Control of Memory Controller Initialization	The controller supports initialization of the memory controller under the control of user logic—for example, through the software control in the user system if a processor is present.
Controller Bonding Support	You can bond two controllers to achieve wider data width for higher bandwidth applications.

Multi-Port Front End

The multi-port front end (MPFE) and its associated fabric interface provide up to six command ports, four read-data ports and four write-data ports, through which user logic can access the hard memory controller.

Figure 7-25: Simplified Diagram of the Arria V Hard Memory Interface

This figure shows a simplified diagram of the Arria V hard memory interface with the MPFE.



Bonding Support

Note: Bonding is supported only for hard memory controllers configured with one port. Do not use the bonding configuration when there is more than one port in each hard memory controller.

You can bond one port of any data width (64, 128, or 256 bits) from two hard memory controllers to support wider data widths.

If you bond two hard memory controllers, the data going out of the controllers to the user logic is synchronized. However, the data going out of the controllers to the memory is not synchronized.

The bonding controllers are not synchronized and remain independent with two separate address buses and two independent command buses. These buses are calibrated separately.

If you require ECC support for a bonded interface, you must implement the ECC logic external to the hard memory controllers.

Note: Only one bonding feature is available per package through the core fabric. A memory interface that uses the bonding feature has higher average latency.

Figure 7-26: Hard Memory Controllers Bonding Support in Arria V GX A1 and A3 Devices

This figure shows the bonding of two opposite hard memory controllers through the core fabric.

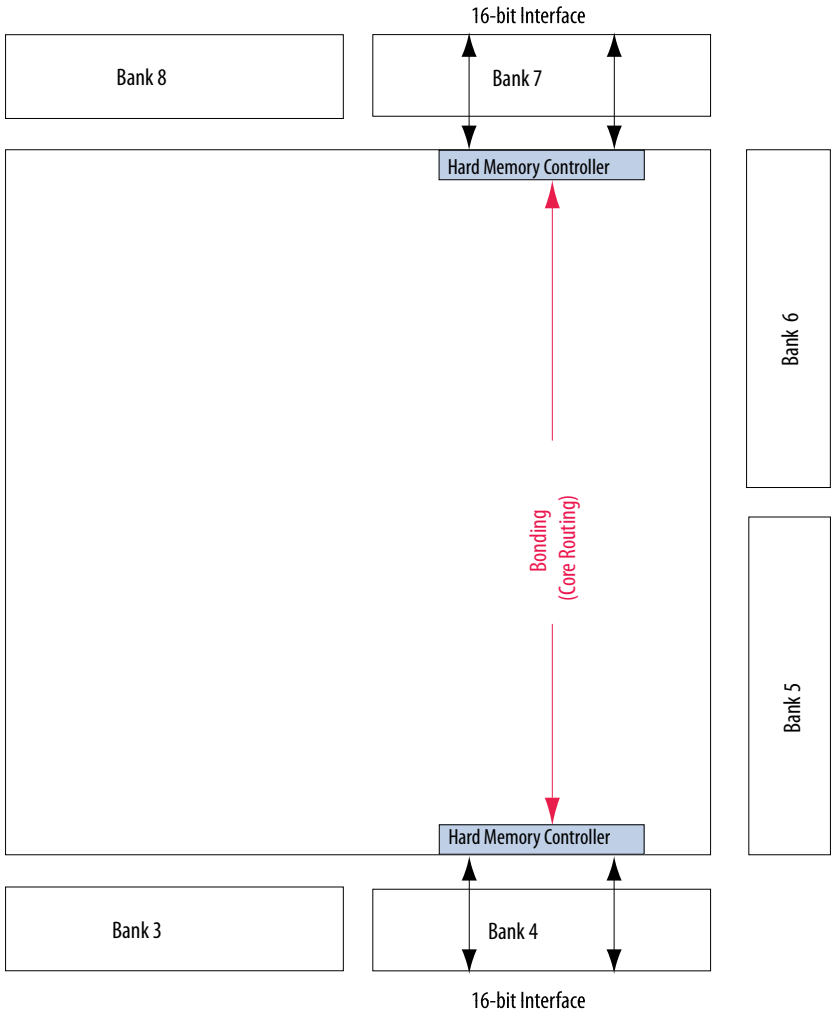


Figure 7-27: Hard Memory Controllers Bonding Support in Arria V GX A5, A7, B1, B3, B5, and B7 Devices, and Arria V GT D3 and D7 Devices

This figure shows the bonding of opposite and same side hard memory controllers through the core fabric.

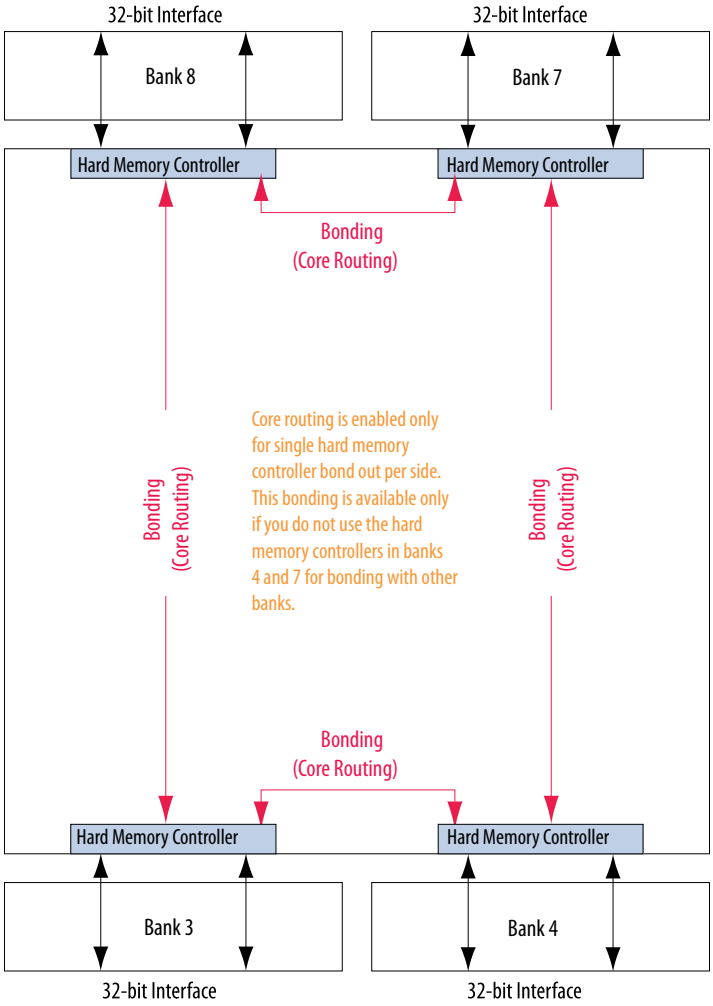
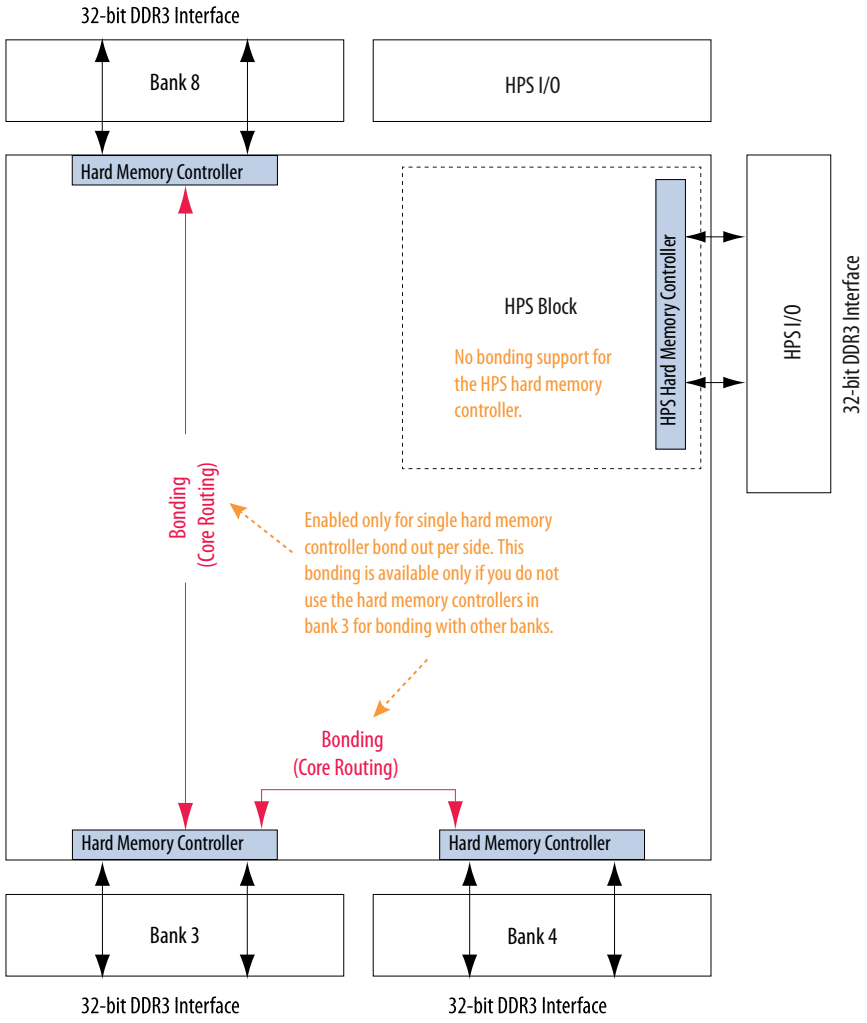


Figure 7-28: Hard Memory Controllers in Arria V SX B3 and B5 Devices, and Arria V ST D3 and D5 Devices

This figure shows the bonding of opposite and same side hard memory controllers through the core fabric.



Related Information

- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about the dedicated pins.
- **Bonding Does Not Work for Multiple MPFE Ports in Hard Memory Controller KDB**

Hard Memory Controller Width for Arria V GX

Table 7-18: Hard Memory Controller Width Per Side in Arria V GX A1, A3, A5, and A7 Devices

Package	Member Code							
	A1		A3		A5		A7	
	Top	Bottom	Top	Bottom	Top	Bottom	Top	Bottom
F672	16	16	16	16	32	32	32	32
F896	24	24	24	24	32	32	32	32
F1152	—	—	—	—	32 + 32	32 + 32	32 + 32	32 + 32

Table 7-19: Hard Memory Controller Width Per Side in Arria V GX B1, B3, B5, and B7 Devices

Package	Member Code							
	B1		B3		B5		B7	
	Top	Bottom	Top	Bottom	Top	Bottom	Top	Bottom
F896	32	32	32	32	—	—	—	—
F1152	32 + 32	32 + 32	32 + 32	32 + 32	32 + 32	32 + 32	32 + 32	32 + 32
F1517	40 + 40	40 + 40	40 + 40	40 + 40	40 + 40	40 + 40	40 + 40	40 + 40

Related Information

Arria V Device Overview
Provides more information about which device packages and feature options contain hard memory controllers.

Hard Memory Controller Width for Arria V GT

Table 7-20: Hard Memory Controller Width Per Side in Arria V GT Devices

Package	Member Code							
	C3		C7		D3		D7	
	Top	Bottom	Top	Bottom	Top	Bottom	Top	Bottom
F672	16	16	16	16	—	—	—	—
F896	24	24	24	24	24	24	—	—
F1152	—	—	32 + 32	32 + 32	32 + 32	32 + 32	32 + 32	32 + 32
F1157	—	—	—	—	40 + 40	40 + 40	40 + 40	40 + 40

Related Information

Arria V Device Overview
Provides more information about which device packages and feature options contain hard memory controllers.

Hard Memory Controller Width for Arria V SX

Table 7-21: FPGA Hard Memory Controller Width Per Side in Arria V SX Devices

Package	Member Code			
	B3		B5	
	Top	Bottom	Top	Bottom
F896	24	24	—	—
F1152	32 + 32	32 + 32	32 + 32	32 + 32
F1517	40 + 40	40 + 40	40 + 40	40 + 40

Related Information

Arria V Device Overview
Provides more information about which device packages and feature options contain hard memory controllers.

Hard Memory Controller Width for Arria V ST

Table 7-22: FPGA Hard Memory Controller Width Per Side in Arria V ST Devices

Package	Member Code	
	D3	
	Top	Bottom
F896	24	24
F1152	32 + 32	32 + 32
F1517	40 + 40	40 + 40

Related Information

Arria V Device Overview
Provides more information about which device packages and feature options contain hard memory controllers.

External Memory Interfaces in Arria V Devices Revision History

Date	Version	Changes
December 2015	2015.12.21	Changed instances of *Quartus II* to *Quartus Prime*.
January 2015	2015.01.23	• Updated hard memory controller widths for all devices. • Removed "Preliminary" notes.
June 2014	2014.06.30	• Added links to the Arria V Device Overview for more information about which device feature option supports the hard memory controllers. • Updated the hard memory controller widths for all devices where the widths are 64, 72, and 80 bits. The widths are now updated to "32 + 32", "40 + 32", and "40 + 40", respectively. The update is to clarify the maximum interface width per hard memory controller in the devices.
January 2014	2014.01.10	• Reduced the soft memory controller performance for DDR3 1.35 V in Arria V GX, GT, SX, and ST devices from 667 MHz to 600 MHz. • Removed support for DDR2 in the HPS hard memory controller. • Updated the figure that shows the delay chains in the Arria V GZ I/O block. • Added related information link to **ALTDQ_DQS2 Megafunction User Guide** for more information about using the delay chains. • Changed all "SoC FPGA" to "SoC". • Updated the figure that shows the DQS/CQ/CQn/QK# Pins and DLLs in Arria V GX A1 and A3 to add the DLL reference clock to the left side DLL. • Updated the topic about delay-locked loop (DLL) to specify that there is a maximum of five DLLs (instead of four). • Updated the topic about the PHYCLK networks to add information about using the PHYCLK network to drive the I/O sub-banks in each I/O bank. • Added links to Altera's **External Memory Spec Estimator** tool to the topics listing the external memory interface performance. • Updated topic about hard memory controller bonding support to specify that bonding is supported only for hard memory controllers configured with one port.

Date	Version	Changes
May 2013	2013.05.06	• Moved all links to the Related Information section of respective topics for easy reference. • Added link to the known document issues in the Knowledge Base. • Updated the topic about Arria V GZ leveling circuitry. • Removed the Arria V GZ phase offset control topic. • Added the I/O and DQS configuration blocks topic. • Updated the DQ/DQS groups for Arria V GX. • Added the DQ/DQS groups for Arria V GT C3 and C7. • Added the DLL reference clock input tables for all Arria V devices. • Added the FPGA hard memory controller widths for Arria V GX, GT, SX, and ST. • Added the HPS hard memory controller widths for Arria V SX and ST.
November 2012	2012.11.19	• Reorganized content and updated template. • Added information for Arria V GZ, including a topic on the leveling circuitry. • Added a list of supported external memory interface standards using the hard memory controller and soft memory controller. • Added performance information for external memory interfaces and the HPS external memory interfaces. • Separated the DQ/DQS groups tables into separate topics for each device variant for easy reference. • Moved the PHYCLK networks pin placement guideline to the **Planning Pin and FPGA Resources** chapter of the *External Memory Interface Handbook*. • Moved information from the "Design Considerations" section into relevant topics. • Removed the "DDR2 SDRAM Interface" and "DDR3 SDRAM DIMM" sections. Refer to the relevant sections in the **External Memory Interface Handbook** for the information. • Updated the diagram for DQS/CQ/CQn/QK# pins and DLLs in Arria V GX A1 and A3 devices to add DLLs on the right, top left, and bottom left, and update the DLL connections to the pins. • Updated the term "Multiport logic" to "multi-port front end" (MPFE).
June 2012	2.0	Updated for the Quartus II software v12.0 release: • Restructured chapter. • Updated “Design Considerations”, “DQS Postamble Circuitry”, and “IOE Registers”sections. • Added SoC devices information. • Added Figure 7–4, Figure 7–8, and Figure 7–20.

Date	Version	Changes
November 2011	1.1	• Updated Table 7–2. • Added "PHY Clock (PHYCLK) Networks" and "UniPHY IP" sections. • Restructured chapter.
May 2011	1.0	Initial release.

Configuration, Design Security, and Remote System Upgrades in Arria V Devices

8

2020.04.13

AV-52008

This chapter describes the configuration schemes, design security, and remote system upgrade that are supported by the Arria V devices.

Related Information

- **Arria V Device Handbook: Known Issues**
  Lists the planned updates to the *Arria V Device Handbook* chapters.
- **Arria V Device Overview**
  Provides more information about configuration features supported for each configuration scheme.
- **Arria V Device Datasheet**
  Provides more information about the estimated uncompressed **.rbf** file sizes, FPP `DCLK-to-DATA[]` ratio, and timing parameters.
- **Configuration via Protocol (CvP) Implementation in Altera FPGAs User Guide**
  Provides more information about the CvP configuration scheme.
- **Hard Processor System Technical Reference Manual**
  Provides more information about configuration via HPS configuration scheme.
- **Design Planning for Partial Reconfiguration**
  Provides more information about partial reconfiguration.

Enhanced Configuration and Configuration via Protocol

Table 8-1: Configuration Schemes and Features of Arria V Devices

Arria V devices support 1.8 V, 2.5 V, 3.0 V, and 3.3 V[25] programming voltages and several configuration schemes.

[25] Arria V GZ does not support 3.3 V.

Mode	Data Width	Max Clock Rate (MHz)	Max Data Rate (Mbps)	Decompression	Design Security	Partial Reconfiguration (26)	Remote System Update
AS through the EPCS and EPCQ serial configuration device	1 bit, 4 bits	100	—	Yes	Yes	—	Yes
PS through CPLD or external microcontroller	1 bit	125	125	Yes	Yes	—	—
FPP	8 bits	125	—	Yes	Yes	—	Parallel flash loader
	16 bits	125	—	Yes	Yes	Yes(27)	
	32 bits(28)	100	—	Yes	Yes	—	
CvP (PCIe)	x1, x2, x4, and x8 lanes	—	—	Yes	Yes	Yes	—
JTAG	1 bit	33	33	—	—	—	—
Configuration via HPS	16 bits	125	—	Yes	Yes	Yes (27)	Parallel flash loader
	32 bits	100	—	Yes	Yes	—	

Instead of using an external flash or ROM, you can configure the Arria V devices through PCIe using CvP. The CvP mode offers the fastest configuration rate and flexibility with the easy-to-use PCIe hard IP block interface. The Arria V CvP implementation conforms to the PCIe 100 ms power-up-to-active time requirement.

Note: Although Arria V GZ devices support PCIe Gen3, you can use only PCIe Gen1 and PCIe Gen2 for CvP configuration scheme.

Related Information

Configuration via Protocol (CvP) Implementation in Altera FPGAs User Guide
Provides more information about the CvP configuration scheme.

MSEL Pin Settings

To select a configuration scheme, hardwire the `MSEL` pins to V_{CCPGM} or GND without pull-up or pull-down resistors.

(26) Partial reconfiguration is an advanced feature of the device family. If you are interested in using partial reconfiguration, contact Intel for support.
(27) Supported at a maximum clock rate of 62.5 MHz.
(28) Arria V GZ only

Note: Altera recommends connecting the MSEL pins directly to V_{CCPGM} or GND. Driving the MSEL pins from a microprocessor or another controlling device may not guarantee the V_{IL} or V_{IH} of the MSEL pins. The V_{IL} or V_{IH} of the MSEL pins must be maintained throughout configuration stages.

Table 8-2: MSEL Pin Settings for Each Configuration Scheme of Arria V Devices

Configuration Scheme	Compression Feature	Design Security Feature	V_{CCPGM} (V) (29)	Power-On Reset (POR) Delay	Valid MSEL[4..0]	Device Variant Support
FPP x8	Disabled	Disabled	1.8/2.5/3.0/3.3	Fast	10100	All
				Standard	11000	All
	Disabled	Enabled	1.8/2.5/3.0/3.3	Fast	10101	All
				Standard	11001	All
	Enabled	Enabled/Disabled	1.8/2.5/3.0/3.3	Fast	10110	All
				Standard	11010	All
FPP x16 (30)	Disabled	Disabled	1.8/2.5/3.0/3.3	Fast	00000	All
				Standard	00100	All
	Disabled	Enabled	1.8/2.5/3.0/3.3	Fast	00001	All
				Standard	00101	All
	Enabled	Enabled/Disabled	1.8/2.5/3.0/3.3	Fast	00010	All
				Standard	00110	All
FPP x32 (30)	Disabled	Disabled	1.8/2.5/3.0	Fast	01000	Arria V GZ
				Standard	01100	Arria V GZ
	Disabled	Enabled	1.8/2.5/3.0	Fast	01001	Arria V GZ
				Standard	01101	Arria V GZ
	Enabled	Enabled/Disabled	1.8/2.5/3.0	Fast	01010	Arria V GZ
				Standard	01110	Arria V GZ
PS	Enabled/Disabled	Enabled/Disabled	1.8/2.5/3.0/3.3	Fast	10000	All
				Standard	10001	All
AS (x1 and x4)	Enabled/Disabled	Enabled/Disabled	3.0/3.3	Fast	10010	All
				Standard	10011	All
JTAG-based configuration	Disabled	Disabled	—	—	Use any valid MSEL pin settings above	All

(29) The Arria V GZ device does not support 3.3 V.
(30) For configuration with HPS in SoC FPGA devices, refer to the FPGA Manager for the related MSEL pin settings.

Note: You must also select the configuration scheme in the **Configuration** page of the **Device and Pin Options** dialog box in the Intel Quartus Prime software. Based on your selection, the option bit in the programming file is set accordingly.

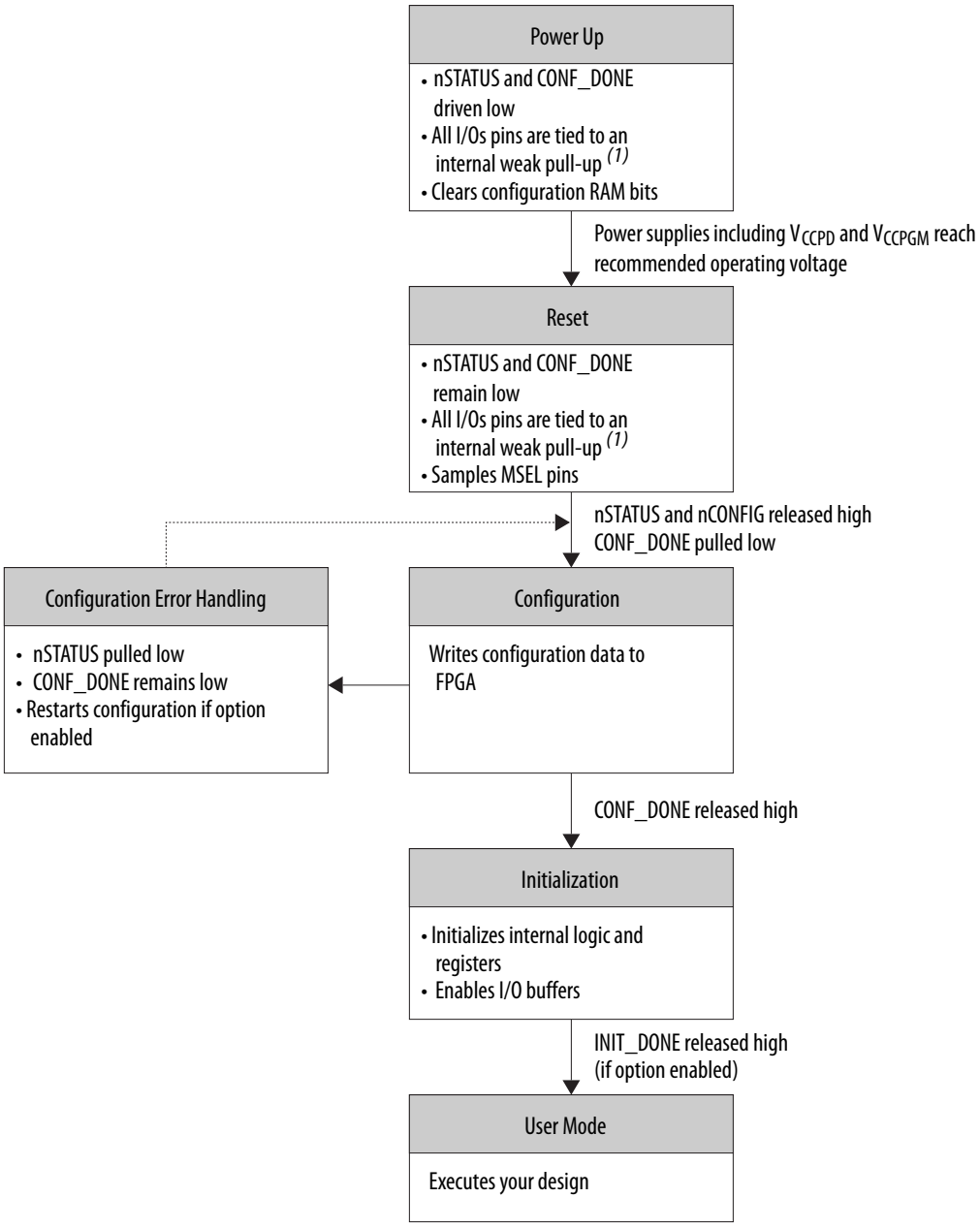
Related Information

- **FPGA Manager**
  Provides more information about the MSEL pin settings for configuration with hard processor system (HPS) in system on a chip (SoC) FPGA devices.
- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about JTAG pins voltage-level connection.
- **Arria V GZ Device Family Pin Connection Guidelines**
  Provides more information about JTAG pins voltage-level connection.

Configuration Sequence

Describes the configuration sequence and each configuration stage.

Figure 8-1: Configuration Sequence for Arria V Devices



Note:
(1) The weak-pull up is enabled after the device has exited POR.

You can initiate reconfiguration by pulling the nCONFIG pin low to at least the minimum t_{CFG} low-pulse width except for configuration using the partial reconfiguration operation. When this pin is pulled low, the nSTATUS and CONF_DONE pins are pulled low and all I/O pins are tied to an internal weak pull-up.

Power Up

Power up all the power supplies that are monitored by the POR circuitry. All power supplies, including V_{CCPGM} and V_{CCPD}, must ramp up from 0 V to the recommended operating voltage level within the ramp-up time specification. Otherwise, hold the `nCONFIG` pin low until all the power supplies reach the recommended voltage level.

V_{CCPGM} Pin

The configuration input buffers do not have to share power lines with the regular I/O buffers in Arria V devices.

The operating voltage for the configuration input pin is independent of the I/O banks power supply, V_{CCIO}, during configuration. Therefore, Arria V devices do not require configuration voltage constraints on V_{CCIO}.

V_{CCPD} Pin

Use the V_{CCPD} pin, a dedicated programming power supply, to power the I/O pre-drivers and JTAG I/O pins (`TCK`, `TMS`, `TDI`, and `TDO`).

The supported configuration voltages are 2.5, 3.0, and 3.3 V for all Arria V devices except for Arria V GZ devices. The supported configuration voltages for Arria V GZ devices are 2.5 and 3.0 V.

If V_{CCIO} of the bank is set to 2.5 V or lower, V_{CCPD} must be powered up at 2.5 V. If V_{CCIO} is set greater than 2.5 V, V_{CCPD} must be greater than V_{CCIO}. For example, when V_{CCIO} is set to 3.0 V, V_{CCPD} must be set at 3.0 V or above. When V_{CCIO} is set to 3.3 V, V_{CCPD} must be set at 3.3 V.

Related Information

- **Arria V Device Datasheet**
  Provides more information about the ramp-up time specifications.
- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about configuration pin connections.
- **Arria V GZ Device Family Pin Connection Guidelines**
  Provides more information about configuration pin connections.
- **Device Configuration Pins** on page 8-12
  Provides more information about configuration pins.
- **I/O Standards Voltage Levels in Arria V Devices** on page 5-8
  Provides more information about typical power supplies for each supported I/O standards in Arria V devices.

Reset

POR delay is the time frame between the time when all the power supplies monitored by the POR circuitry reach the recommended operating voltage and when `nSTATUS` is released high and the Arria V device is ready to begin configuration.

Set the POR delay using the `MSEL` pins.

The user I/O pins are tied to an internal weak pull-up until the device is configured.

Related Information

- **MSEL Pin Settings** on page 8-2

- **Arria V Device Datasheet**
  Provides more information about the POR delay specification.

Configuration

For more information about the DATA[] pins for each configuration scheme, refer to the appropriate configuration scheme.

Configuration Error Handling

To restart configuration automatically, turn on the **Auto-restart configuration after error** option in the **General** page of the **Device and Pin Options** dialog box in the Intel Quartus Prime software.

If you do not turn on this option, you can monitor the nSTATUS pin to detect errors. To restart configuration, pull the nCONFIG pin low for at least the duration of t_{CFG}.

Related Information

Arria V Device Datasheet
Provides more information about t_{STATUS} and t_{CFG} timing parameters.

Initialization

The initialization clock source is from the internal oscillator, CLKUSR pin, or DCLK pin. By default, the internal oscillator is the clock source for initialization. If you use the internal oscillator, the Arria V device will be provided with enough clock cycles for proper initialization.

Note: If you use the optional CLKUSR pin as the initialization clock source and the nCONFIG pin is pulled low to restart configuration during device initialization, ensure that the CLKUSR or DCLK pin continues toggling until the nSTATUS pin goes low and then goes high again.

The CLKUSR pin provides you with the flexibility to synchronize initialization of multiple devices or to delay initialization. Supplying a clock on the CLKUSR pin during initialization does not affect configuration. After the CONF_DONE pin goes high, the CLKUSR or DCLK pin is enabled after the time specified by t_{CD2CU}. When this time period elapses, Arria V devices require a minimum number of clock cycles as specified by T_{init} to initialize properly and enter user mode as specified by the t_{CD2UMC} parameter.

Related Information

Arria V Device Datasheet
Provides more information about t_{CD2CU}, t_{init}, and t_{CD2UMC} timing parameters, and initialization clock source.

User Mode

You can enable the optional INIT_DONE pin to monitor the initialization stage. After the INIT_DONE pin is pulled high, initialization completes and your design starts executing. The user I/O pins will then function as specified by your design.

During device initialization stage, the FPGA registers, core logic, and I/O are not released from reset at the same time. The increase in clock frequency, device size, and design complexity require a reset strategy that

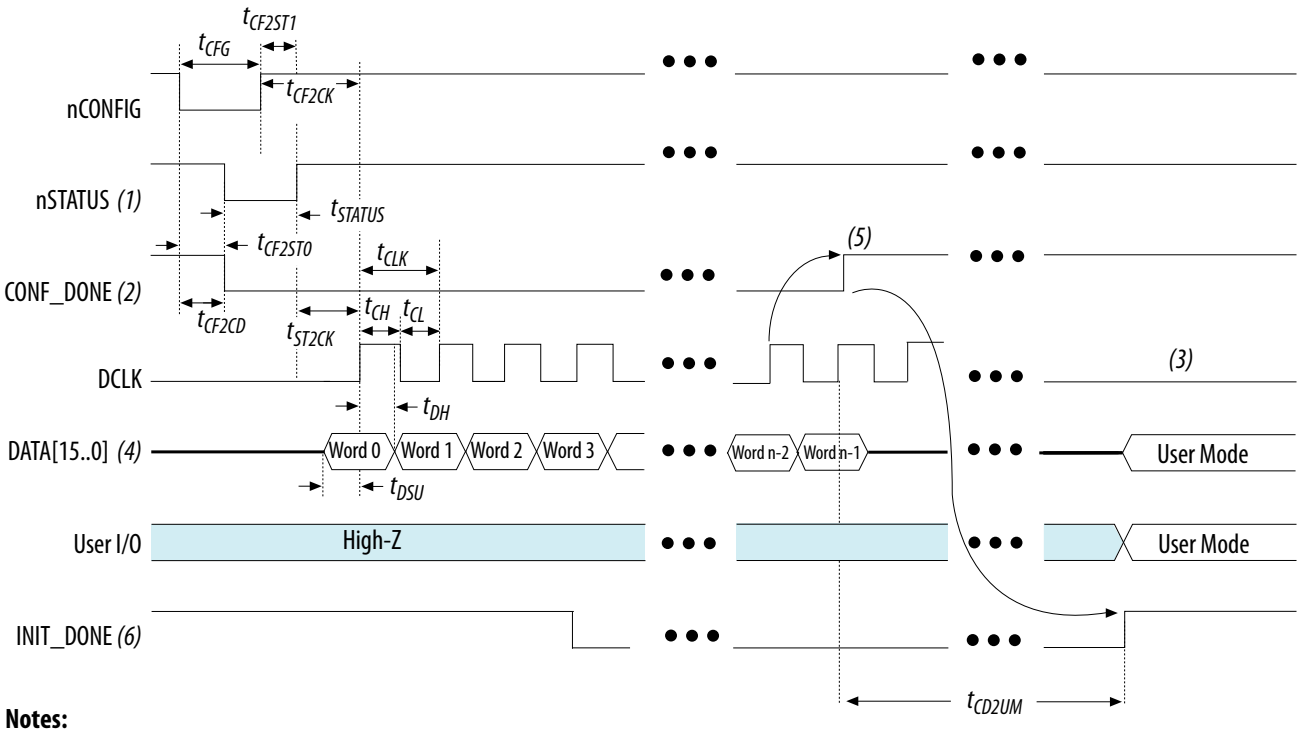
considers the differences in the release from reset. Intel recommends that you use the following implementations to reset your design properly and until the device has fully entered user mode:

- Hold the entire design in reset for a period of time by following the `CONF_DONE` high to user mode (t_{CD2UM}) or `CONF_DONE` high to user mode with `CLKUSR` option turned on (t_{CD2UMC}) specifications as defined in the *Arria V Device Datasheet* before starting any operation after the device enters into user mode. For example, the t_{CD2UM} range for Arria V device is between 175 us to 437 us.
- If you have an external device that reacts based on an Intel FPGA output pin, perform the following steps to avoid false reaction:
  - Ensure that the external device ignores the state of the FPGA output pin until the external `INIT_DONE` pin goes high. Refer to the t_{CD2UM} or t_{CD2UMC} specifications in the *Arria V Device Datasheet* for more information.
  - Keep the input state to the external device constant by using the external logic until the external `INIT_DONE` pin goes high.

Configuration Timing Waveforms

FPP Configuration Timing

Figure 8-2: FPP Configuration Timing Waveform when DCLK-to-DATA[] Ratio is 1



Notes:

(1) After power up, the FPGA holds nSTATUS low for the time of the POR delay.

(2) After power up, before and during configuration, CONF_DONE is low.

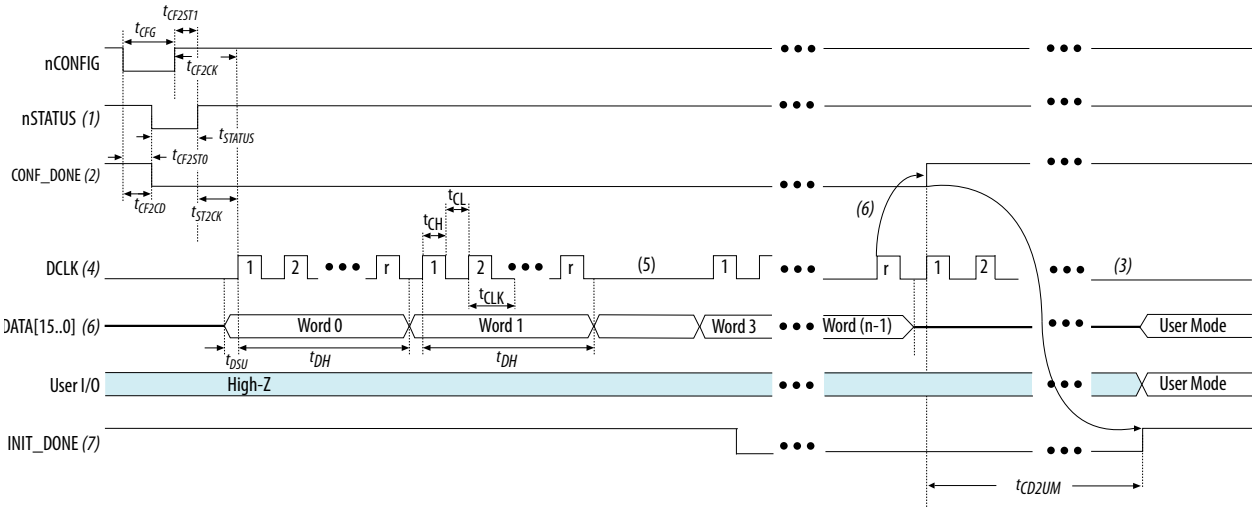
(3) Do not leave DCLK floating after configuration. DCLK is ignored after configuration is complete. It can toggle high or low if required.

(4) For FPP x16, use DATA[15..0]. For FPP x8, use DATA[7..0]. DATA[15..5] are available as a user I/O pin after configuration. The state of this pin depends on the dual-purpose pin settings.

(5) To ensure a successful configuration, send the entire configuration data to the FPGA. CONF_DONE is released high when the FPGA receives all the configuration data successfully. After CONF_DONE goes high, send two additional falling edges on DCLK to begin initialization and enter user mode.

(6) After the option bit to enable the INIT_DONE pin is configured into the device, the INIT_DONE goes low.

Figure 8-3: FPP Configuration Timing Waveform when DCLK-to-DATA[] Ratio is >1



Notes:

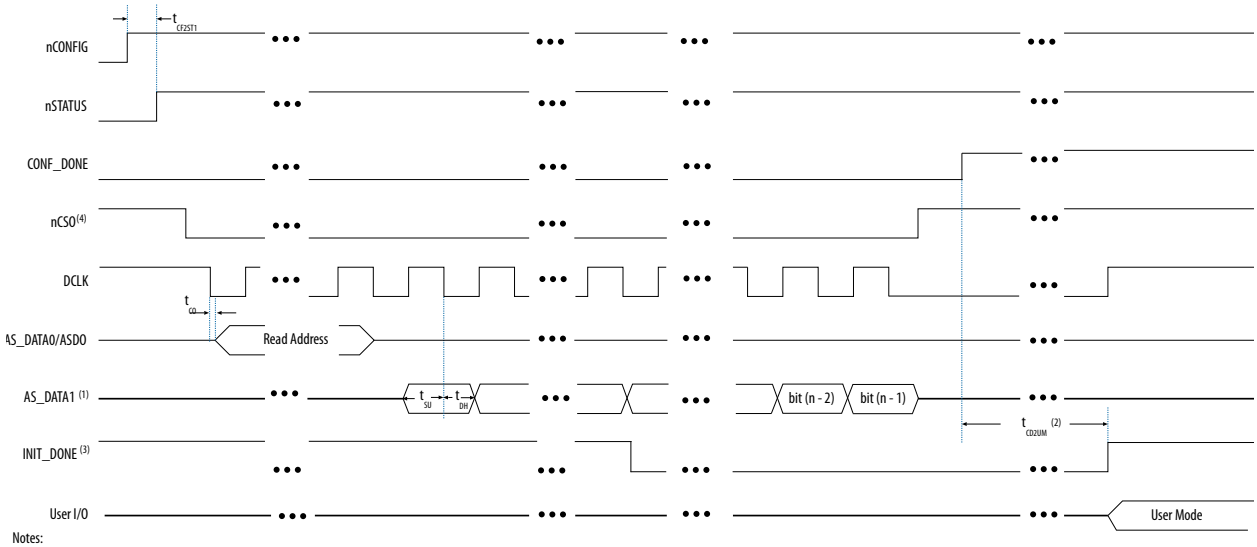
(1) After power up, the FPGA holds nSTATUS low for the time as specified by the POR delay.
(2) After power up, before and during configuration, CONF_DONE is low.
(3) Do not leave DCLK floating after configuration. DCLK is ignored after configuration is complete. It can toggle high or low if required.
(4) "r" denotes the DCLK-to-DATA[] ratio. For the DCLK-to-DATA[] ratio based on the decompression and the design security feature enable settings, refer to the DCLK-to-DATA[] Ratio table.
(5) If needed, pause DCLK by holding it low. When DCLK restarts, the external host must provide data on the DATA[15..0] pins prior to sending the first DCLK rising edge.
(6) To ensure a successful configuration, send the entire configuration data to the FPGA. CONF_DONE is released high after the FPGA device receives all the configuration data successfully. After CONF_DONE goes high, send two additional falling edges on DCLK to begin initialization and enter user mode.
(7) After the option bit to enable the INIT_DONE pin is configured into the device, the INIT_DONE goes low.

Related Information

- FPP Configuration Timing when DCLK-to-DATA[] = 1
- FPP Configuration Timing when DCLK-to-DATA[] >1

AS Configuration Timing

Figure 8-4: AS Configuration Timing Waveform



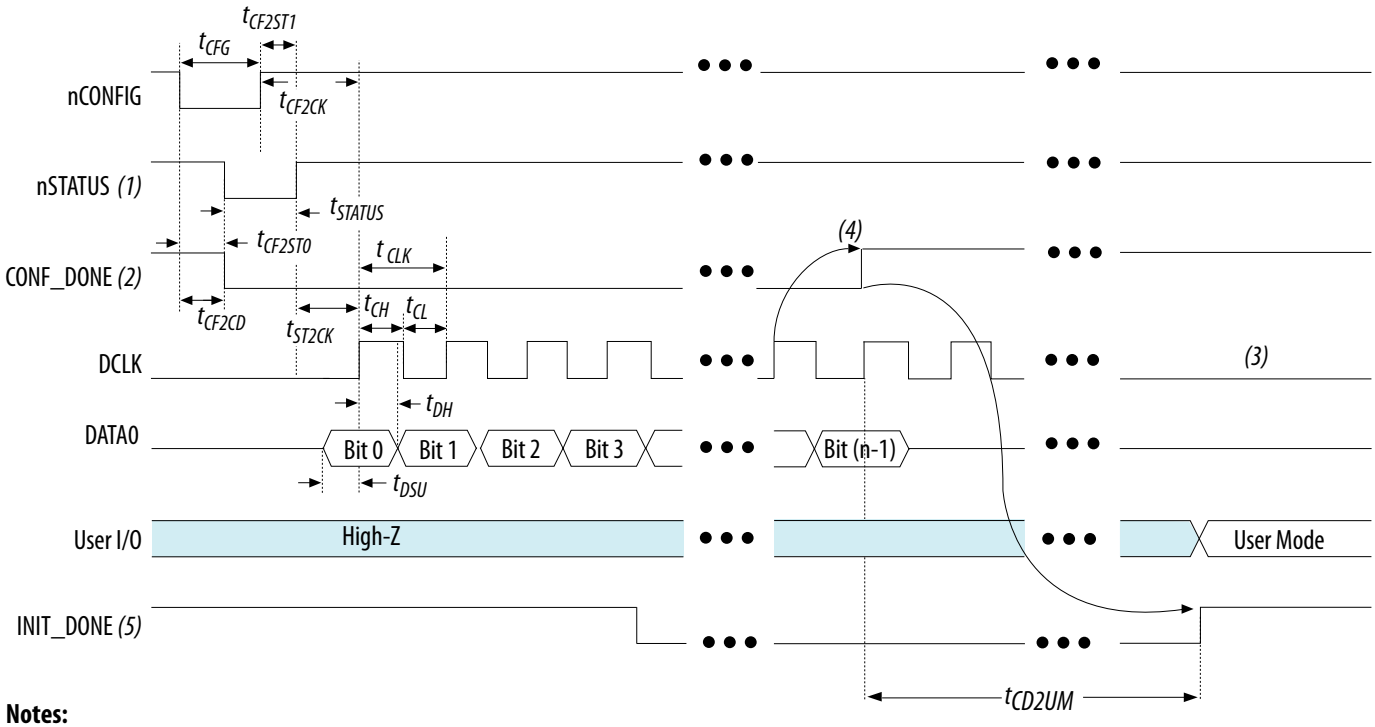
Notes:
(1) If you are using AS x4 mode, this signal represents the AS_DATA[3..0] and EPCQ sends in 4-bits of data for each DCLK cycle.
(2) The initialization clock can be from the internal oscillator or CLKUSR pin.
(3) After the option bit to enable the INIT_DONE pin is configured into the device, the INIT_DONE goes low.
(4) The time between nCSO falling edge to the first toggling of DCLK is more than 15 ns.

Related Information

AS Configuration Timing

PS Configuration Timing

Figure 8-5: PS Configuration Timing Waveform



Notes:

(1) After power up, the FPGA holds nSTATUS low for the time of the POR delay.
(2) After power up, before and during configuration, CONF_DONE is low.
(3) Do not leave DCLK floating after configuration. DCLK is ignored after configuration is complete. It can toggle high or low if required.
(4) To ensure a successful configuration, send the entire configuration data to the FPGA. CONF_DONE is released high after the FPGA receives all the configuration data successfully. After CONF_DONE goes high, send two additional falling edges on DCLK to begin initialization and enter user mode.
(5) After the option bit to enable the INIT_DONE pin is configured into the device, the INIT_DONE goes low.

Related Information

PS Configuration Timing

Device Configuration Pins

Configuration Pins Summary

The following table lists the Arria V configuration pins and their power supply.

Note:
1. The TDI, TMS, TCK, and TDO pins are powered by V_{CCPD} of the bank in which the pin resides.
2. The CLKUSR, DEV_OE, DEV_CLRn, DATA[15..5], and DATA[31..16] pins are powered by V_{CCPGM} during configuration and by V_{CCIO} of the bank in which the pin resides if you use it as a user I/O pin.
3. The DCLK, AS_DATA0/ASDO, AS_DATA1, AS_DATA2, AS_DATA3, and nCSO pins have 25 kOhm pull-up resistors when the MSEL pins are set to AS configuration scheme.

Table 8-3: Configuration Pin Summary for Arria V Devices

Configuration Pin	Configuration Scheme	Input/Output	User Mode	Powered By
TDI	JTAG	Input	—	V_{CCPD}
TMS	JTAG	Input	—	V_{CCPD}
TCK	JTAG	Input	—	V_{CCPD}
TDO	JTAG	Output	—	V_{CCPD}
CLKUSR	All schemes	Input	I/O	V_{CCPGM}/V_{CCIO} (31)
CRC_ERROR	Optional, all schemes	Output	I/O	Pull-up
CONF_DONE	All schemes	Bidirectional	—	V_{CCPGM}/Pull-up
DCLK	FPP and PS	Input	—	V_{CCPGM}
	AS	Output	—	V_{CCPGM}
DEV_OE	Optional, all schemes	Input	I/O	V_{CCPGM}/V_{CCIO} (31)
DEV_CLRn	Optional, all schemes		I/O	V_{CCPGM}/V_{CCIO} (31)
INIT_DONE	Optional, all schemes	Output	I/O	Pull-up
MSEL[4..0]	All schemes	Input	—	V_{CCPGM}
nSTATUS	All schemes	Bidirectional	—	V_{CCPGM}/Pull-up
nCE	All schemes	Input	—	V_{CCPGM}
nCEO	All schemes	Output	I/O	Pull-up
nCONFIG	All schemes	Input	—	V_{CCPGM}
nIO_PULLUP(32)	All schemes	Input	—	V_{CCPGM}

(31) This pin is powered by V_{CCPGM} during configuration and powered by V_{CCIO} of the bank in which the pin resides when you use this pin as a user I/O pin.
(32) These pins are applicable for Arria V GZ devices only.

Configuration Pin	Configuration Scheme	Input/Output	User Mode	Powered By
DATA[15..5]	FPP x8 and x16	Input	I/O	V_{CCPGM}/V_{CCIO} (31)
DATA[31..16] (32)	FPP x32	Input	I/O	V_{CCPGM}/V_{CCIO} (31)
DATA[4..0] (32)	FPP x8, x16, and x32	Input	I/O	V_{CCPGM}/V_{CCIO} (31)
nCSO/DATA4 (33)	AS	Output	—	V_{CCPGM}
	FPP	Input	—	V_{CCPGM}
AS_DATA[3..1]/ DATA[3..1] (33)	AS	Bidirectional	—	V_{CCPGM}
	FPP	Input	—	V_{CCPGM}
AS_DATA0/DATA0/ASDO (33)	AS	Bidirectional	—	V_{CCPGM}
	FPP and PS	Input	—	V_{CCPGM}
AS_DATA0/ASDO(32)	AS	Bidirectional	—	V_{CCPGM}
AS_DATA[3..1](32)	AS	Bidirectional	—	V_{CCPGM}
PR_REQUEST	Partial Reconfiguration	Input	I/O	V_{CCPGM}/V_{CCIO} (31)
PR_READY	Partial Reconfiguration	Output	I/O	V_{CCPGM}/V_{CCIO} (31)
PR_ERROR	Partial Reconfiguration	Output	I/O	V_{CCPGM}/V_{CCIO} (31)
PR_DONE	Partial Reconfiguration	Output	I/O	V_{CCPGM}/V_{CCIO} (31)
CvP_CONFDONE	CvP (PCIe)	Output	I/O	V_{CCPGM}/V_{CCIO} (31)

Related Information

- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about each configuration pin.
- **Arria V GZ Device Family Pin Connection Guidelines**
  Provides more information about each configuration pin.

(33) These pins are applicable for all Arria V devices except for Arria V GZ devices.

Configuration Pin Options in the Intel Quartus Prime Software

The following table lists the dual-purpose configuration pins available in the **Device and Pin Options** dialog box in the Intel Quartus Prime software.

Table 8-4: Configuration Pin Options

Configuration Pin	Category Page	Option
CLKUSR	General	Enable user-supplied start-up clock (CLKUSR)
DEV_CLRn	General	Enable device-wide reset (DEV_CLRn)
DEV_OE	General	Enable device-wide output enable (DEV_OE)
INIT_DONE	General	Enable INIT_DONE output
nCEO	General	Enable nCEO pin
CRC_ERROR	Error Detection CRC	Enable Error Detection CRC_ERROR pin
		Enable open drain on CRC_ERROR pin
		Enable internal scrubbing
PR_REQUEST	General	Enable PR pin
PR_READY		
PR_ERROR		
PR_DONE		

Related Information

Reviewing Printed Circuit Board Schematics with the Quartus II Software
Provides more information about the device and pin options dialog box setting.

Fast Passive Parallel Configuration

The FPP configuration scheme uses an external host, such as a microprocessor, MAX® II device, or MAX V device. This scheme is the fastest method to configure Arria V devices. The FPP configuration scheme supports 8- and 16-bits data width.

You can use an external host to control the transfer of configuration data from an external storage such as flash memory to the FPGA. The design that controls the configuration process resides in the external host. You can store the configuration data in Raw Binary File (**.rbf**), Hexadecimal (Intel-Format) File (**.hex**), or Tabular Text File (**.ttf**) formats.

You can use the PFL IP core with a MAX II or MAX V device to read configuration data from the flash memory device and configure the Arria V device.

Note: Two DCLK falling edges are required after the CONF_DONE pin goes high to begin the initialization of the device for both uncompressed and compressed configuration data in an FPP configuration.

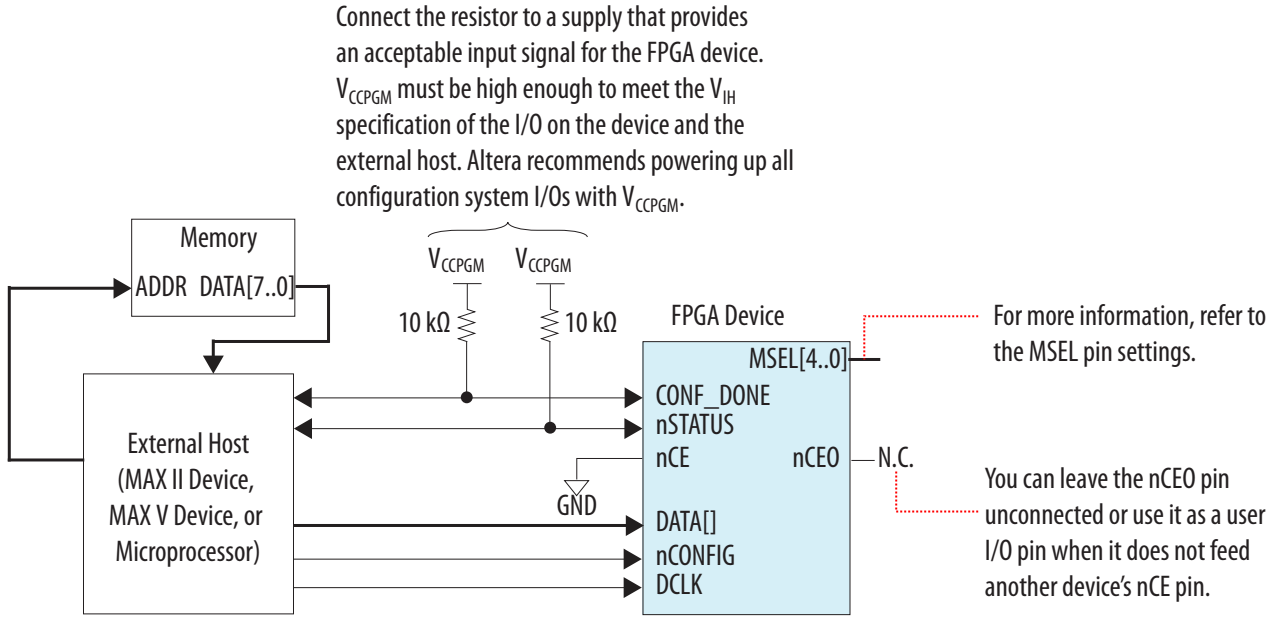
Related Information

- **Parallel Flash Loader IP Core User Guide**
- **Arria V Device Datasheet**
  Provides more information about the FPP configuration timing.

Fast Passive Parallel Single-Device Configuration

To configure an Arria V device, connect the device to an external host as shown in the following figure.

Figure 8-6: Single Device FPP Configuration Using an External Host



Fast Passive Parallel Multi-Device Configuration

You can configure multiple Arria V devices that are connected in a chain.

Pin Connections and Guidelines

Observe the following pin connections and guidelines for this configuration setup:

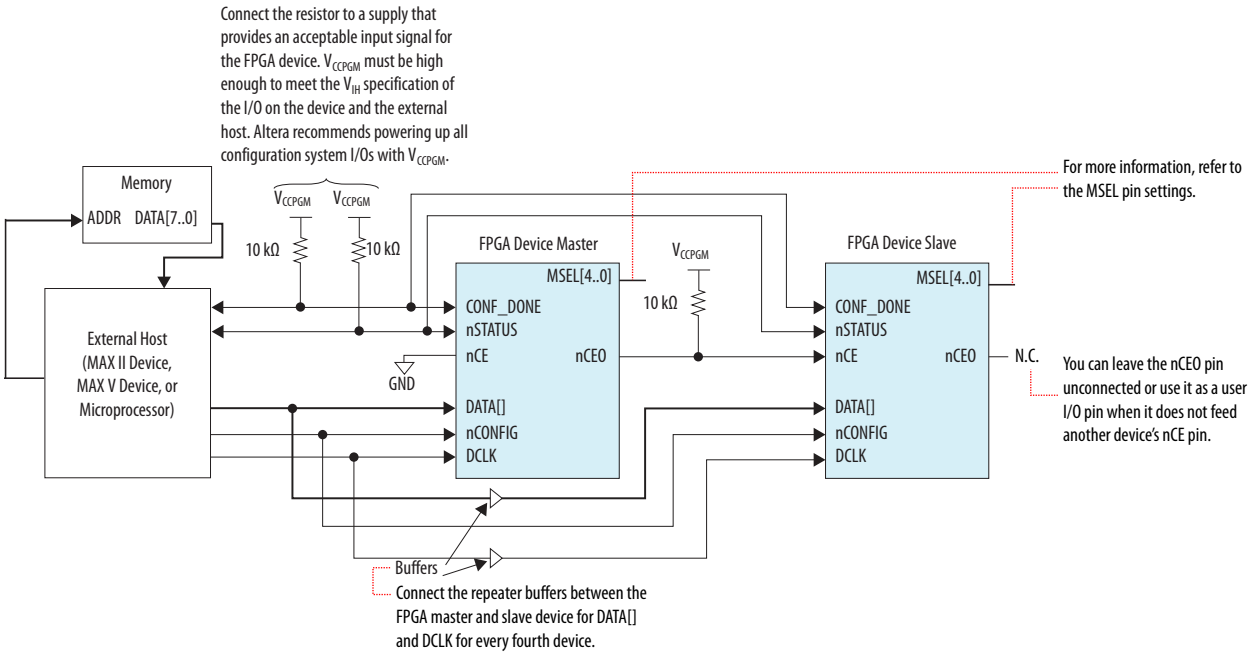
- Tie the following pins of all devices in the chain together:
  - nCONFIG
  - nSTATUS
  - DCLK
  - DATA[]
  - CONF_DONE

  By tying the CONF_DONE and nSTATUS pins together, the devices initialize and enter user mode at the same time. If any device in the chain detects an error, configuration stops for the entire chain and you must reconfigure all the devices. For example, if the first device in the chain flags an error on the nSTATUS pin, it resets the chain by pulling its nSTATUS pin low.
- Ensure that DCLK and DATA[] are buffered for every fourth device to prevent signal integrity and clock skew problems.
- All devices in the chain must use the same data width.
- If you are configuring the devices in the chain using the same configuration data, the devices must be of the same package and density.

Using Multiple Configuration Data

To configure multiple Arria V devices in a chain using multiple configuration data, connect the devices to an external host as shown in the following figure.

Figure 8-7: Multiple Device FPP Configuration Using an External Host When Both Devices Receive a Different Set of Configuration Data

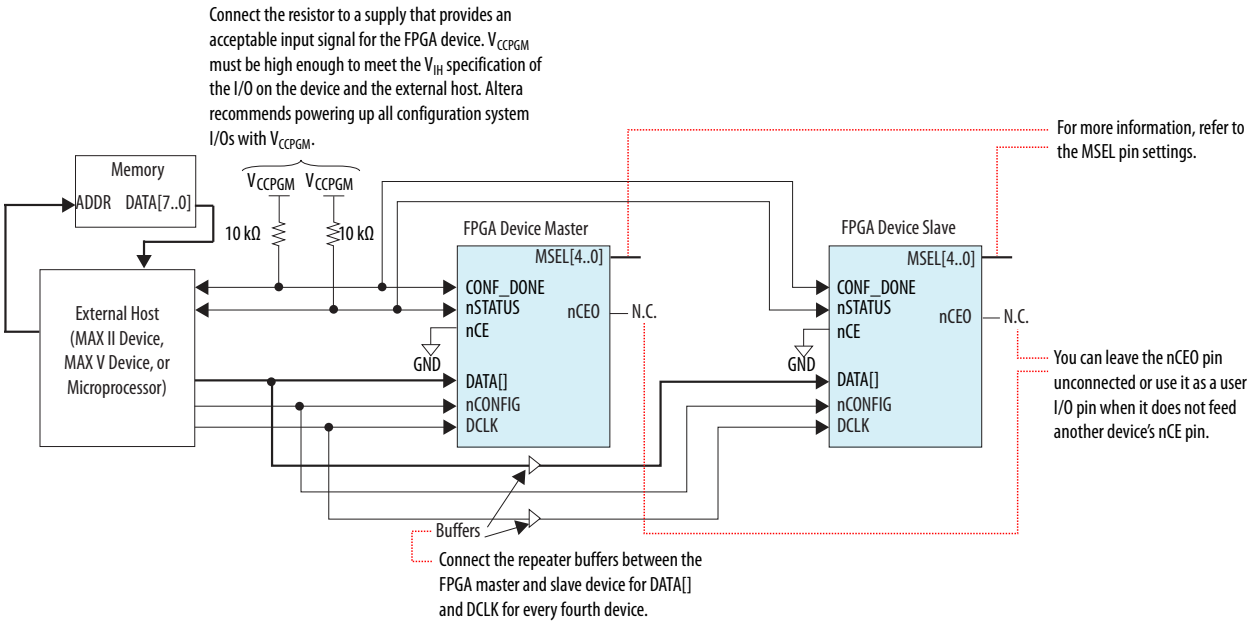


When a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device in the chain. Configuration automatically begins for the second device in one clock cycle.

Using One Configuration Data

To configure multiple Arria V devices in a chain using one configuration data, connect the devices to an external host as shown in the following figure.

Figure 8-8: Multiple Device FPP Configuration Using an External Host When Both Devices Receive the Same Data



The nCE pins of the device in the chain are connected to GND, allowing configuration for these devices to begin and end at the same time.

Transmitting Configuration Data

This section describes how to transmit configuration data when you are using **.rbf** file for FPP x8, x16, and x32 configuration schemes. The configuration data in the **.rbf** file is little endian.

For example, if the **.rbf** file contains the byte sequence 02 1B EE 01, refer to the following tables for details on how this data is transmitted in the FPP x8, x16, and x32 configuration schemes.

Table 8-5: Transmitting Configuration Data for FPP x8 Configuration Scheme

In FPP x8 configuration scheme, the LSB of a byte is BIT0, and the MSB is BIT7.

BYTE0 = 02	BYTE1 = 1B	BYTE2 = EE	BYTE3 = 01
D[7..0]	D[7..0]	D[7..0]	D[7..0]
0000 0010	0001 1011	1110 1110	0000 0001

Table 8-6: Transmitting Configuration Data for FPP x16 Configuration Scheme

In FPP x16 configuration scheme, the first byte in the file is the LSB of the configuration word, and the second byte in the file is the MSB of the configuration word.

WORD0 = 1B02		WORD1 = 01EE	
LSB: BYTE0 = 02	MSB: BYTE1 = 1B	LSB: BYTE2 = EE	MSB: BYTE3 = 01
D[7..0]	D[15..8]	D[7..0]	D[15..8]
0000 0010	0001 1011	1110 1110	0000 0001

Table 8-7: Transmitting Configuration Data for FPP x32 Configuration Scheme

In FPP x32 configuration scheme, the first byte in the file is the LSB of the configuration double word, and the fourth byte is the MSB.

Double Word = 01EE1B02			
LSB: BYTE0 = 02	BYTE1 = 1B	BYTE2 = EE	MSB: BYTE3 = 01
D[7..0]	D[15..8]	D[23..16]	D[31..24]
0000 0010	0001 1011	1110 1110	0000 0001

Ensure that you do not swap the upper bits or bytes with the lower bits or bytes when performing the FPP configuration. Sending incorrect configuration data during the configuration process may cause unexpected behavior on the CONF_DONE signal.

Active Serial Configuration

The AS configuration scheme supports AS x1 (1-bit data width) and AS x4 (4-bit data width) modes. The AS x4 mode provides four times faster configuration time than the AS x1 mode. In the AS configuration scheme, the Arria V device controls the configuration interface.

In the AS configuration process, after power-up, the Arria V device drives the DCLK pin with the default 12.5 MHz internal oscillator to read the configuration bitstream from the serial flash. The device determines the configuration options such as clock source, DCLK frequency, ASx1, or ASx4 by reading the option bits, located from 0x80 to 0x127, at the start of the programming file stored in the serial flash.

After the option bit is decoded by the Arria V configuration control block, the AS configuration continues with the design option by reading the rest of the programming file from the serial flash until CONF_DONE pin asserts high and eventually enters user mode. When any interruption (such as data corruption) occurs in the middle of the AS Configuration, the nSTATUS pin will assert low to indicate configuration error, and deassert high to restart the configuration process. If there is no image in the serial flash or if the image is corrupted, you will observe the nSTATUS will pulse low repeatedly as the control block tries to configure itself forever, until the configuration is successful.

Related Information

Arria V Device Datasheet
Provides more information about the AS configuration timing.

DATA Clock (DCLK)

Arria V devices generate the serial clock, DCLK, that provides timing to the serial interface. In the AS configuration scheme, Arria V devices drive control signals on the falling edge of DCLK and latch the configuration data on the following falling edge of this clock pin.

The maximum DCLK frequency supported by the AS configuration scheme is 100 MHz except for the AS multi-device configuration scheme. You can source DCLK using CLKUSR or the internal oscillator. If you use the internal oscillator, you can choose a 12.5, 25, 50, or 100 MHz clock under the **Device and Pin Options** dialog box, in the **Configuration** page of the Intel Quartus Prime software.

After power-up, DCLK is driven by a 12.5 MHz internal oscillator by default. The Arria V device determines the clock source and frequency to use by reading the option bit in the programming file.

Related Information

Arria V Device Datasheet
Provides more information about the DCLK frequency specification in the AS configuration scheme.

Active Serial Single-Device Configuration

To configure an Arria V device, connect the device to a serial configuration (EPCS) device or quad-serial configuration (EPCQ) device, as shown in the following figures.

Figure 8-9: Single Device AS x1 Mode Configuration

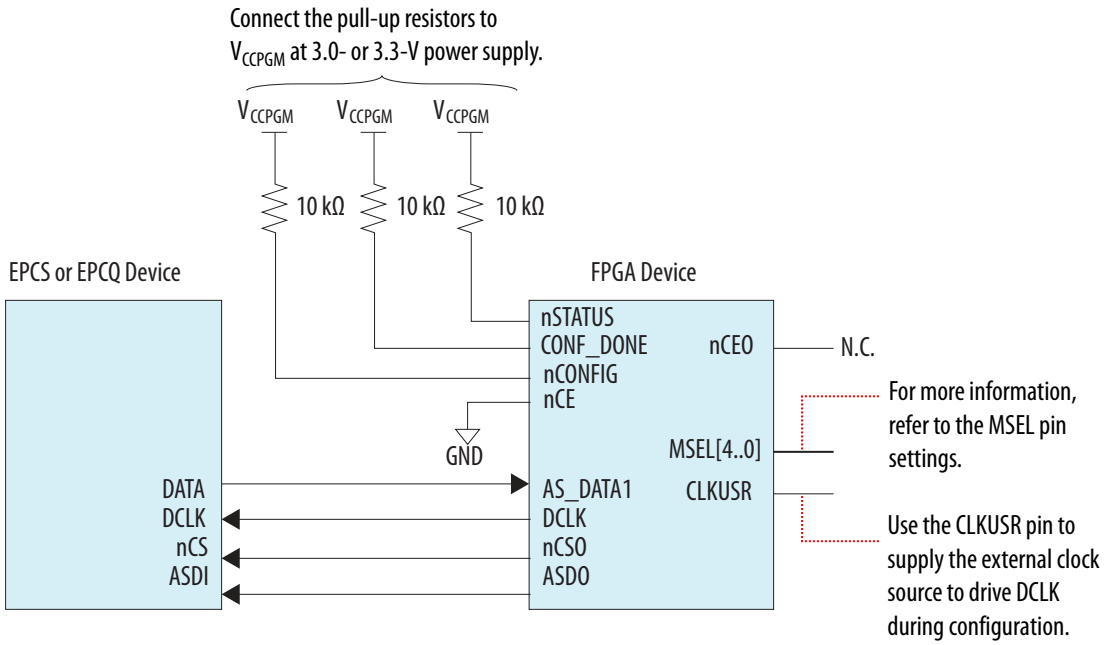
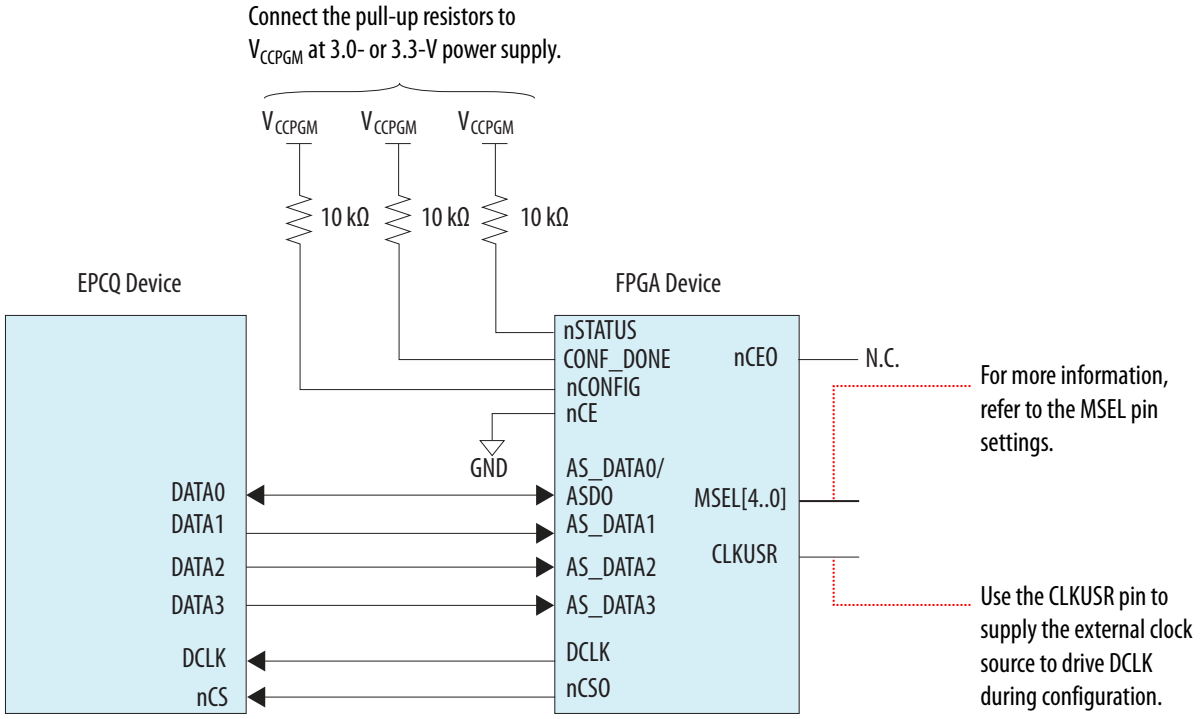


Figure 8-10: Single Device AS x4 Mode Configuration



Active Serial Multi-Device Configuration

You can configure multiple Arria V devices that are connected to a chain. Only AS x1 mode supports multi-device configuration.

The first device in the chain is the configuration master. Subsequent devices in the chain are configuration slaves.

Pin Connections and Guidelines

Observe the following pin connections and guidelines for this configuration setup:

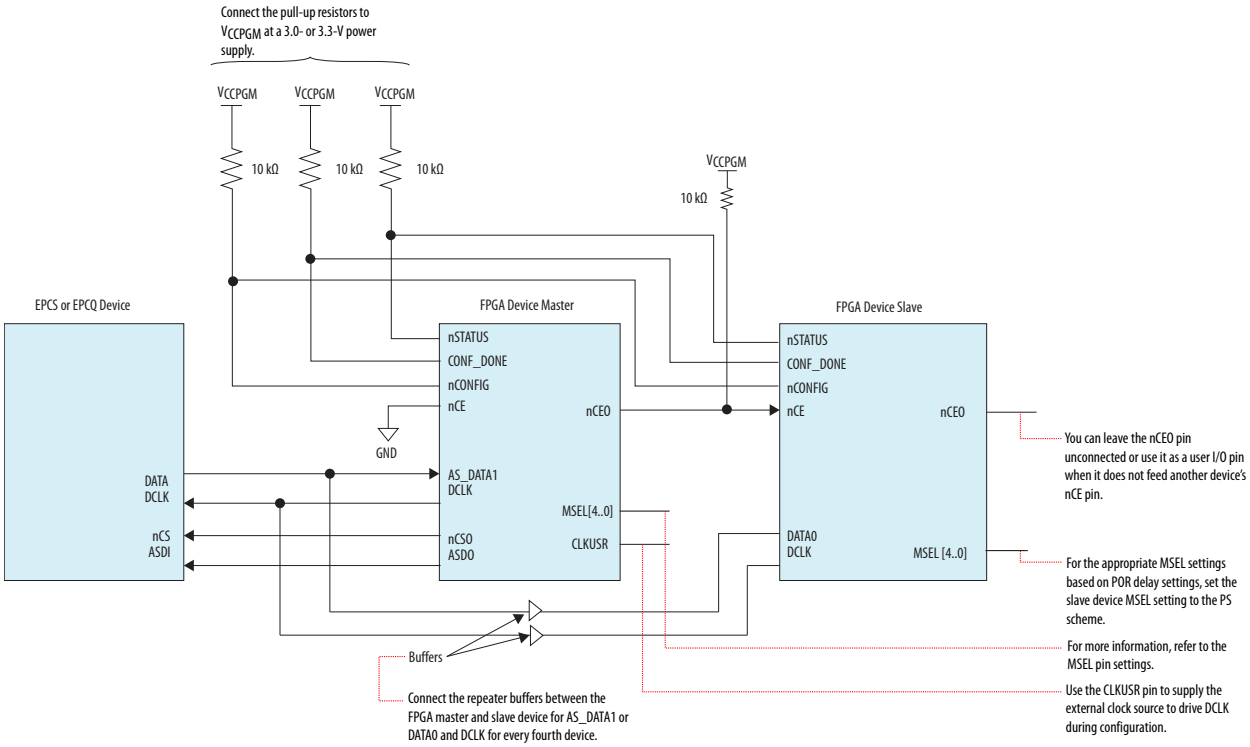
- Hardwire the MSEL pins of the first device in the chain to select the AS configuration scheme. For subsequent devices in the chain, hardwire their MSEL pins to select the PS configuration scheme. Any other Altera® devices that support the PS configuration can also be part of the chain as a configuration slave.
- Tie the following pins of all devices in the chain together:
  - nCONFIG
  - nSTATUS
  - DCLK
  - DATA[]
  - CONF_DONE

  By tying the CONF_DONE, nSTATUS, and nCONFIG pins together, the devices initialize and enter user mode at the same time. If any device in the chain detects an error, configuration stops for the entire chain and you must reconfigure all the devices. For example, if the first device in the chain flags an error on the nSTATUS pin, it resets the chain by pulling its nSTATUS pin low.
- Ensure that DCLK and DATA[] are buffered every fourth device to prevent signal integrity and clock skew problems.

Using Multiple Configuration Data

To configure multiple Arria V devices in a chain using multiple configuration data, connect the devices to an EPCS or EPCQ device, as shown in the following figure.

Figure 8-11: Multiple Device AS Configuration When Both Devices in the Chain Receive Different Sets of Configuration Data



When a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device in the chain. Configuration automatically begins for the second device in one clock cycle.

Estimating the Active Serial Configuration Time

The AS configuration time is mostly the time it takes to transfer the configuration data from an EPCS or EPCQ device to the Arria V device.

Use the following equations to estimate the configuration time:

- AS x1 mode

  .rbf Size x (minimum DCLK period / 1 bit per DCLK cycle) = estimated minimum configuration time.
- AS x4 mode

  .rbf Size x (minimum DCLK period / 4 bits per DCLK cycle) = estimated minimum configuration time.

Compressing the configuration data reduces the configuration time. The amount of reduction varies depending on your design.

Using EPCS and EPCQ Devices

EPCS devices support AS x1 mode and EPCQ devices support AS x1 and AS x4 modes.

Related Information

- Serial Configuration (EPCS) Devices Datasheet
- Quad-Serial Configuration (EPCQ) Devices Datasheet

Controlling EPCS and EPCQ Devices

During configuration, Arria V devices enable the EPCS or EPCQ device by driving its nCSO output pin low, which connects to the chip select (nCS) pin of the EPCS or EPCQ device. Arria V devices use the DCLK and ASDO pins to send operation commands and read address signals to the EPCS or EPCQ device. The EPCS or EPCQ device provides data on its serial data output (DATA[]) pin, which connects to the AS_DATA[] input of the Arria V devices.

Note: If you wish to gain control of the EPCS pins, hold the nCONFIG pin low and pull the nCE pin high. This causes the device to reset and tri-state the AS configuration pins.

Trace Length and Loading Guideline

The maximum trace length and loading apply to both single- and multi-device AS configuration setups as listed in the following table. The trace length is the length from the Arria V device to the EPCS or EPCQ device.

Table 8-8: Maximum Trace Length and Loading Guideline for AS x1 and x4 Configurations for Arria V Devices

Arria V Device AS Pins	Maximum Board Trace Length (Inches)		Maximum Board Load (pF)
	12.5/ 25/ 50 MHz	100 MHz	
DCLK	10	6	5
DATA[3..0]	10	6	10
nCSO	10	6	10

Programming EPCS and EPCQ Devices

You can program EPCS and EPCQ devices in-system using a USB-Blaster™, EthernetBlaster, EthernetBlaster II, or ByteBlaster™ II download cable. Alternatively, you can program the EPCS or EPCQ using a microprocessor with the SRunner software driver.

In-system programming (ISP) offers you the option to program the EPCS or EPCQ either using an AS programming interface or a JTAG interface. Using the AS programming interface, the configuration data is programmed into the EPCS by the Intel Quartus Prime software or any supported third-party software. Using the JTAG interface, an Altera IP called the serial flash loader (SFL) must be downloaded into the Arria V device to form a bridge between the JTAG interface and the EPCS or EPCQ. This allows the EPCS or EPCQ to be programmed directly using the JTAG interface.

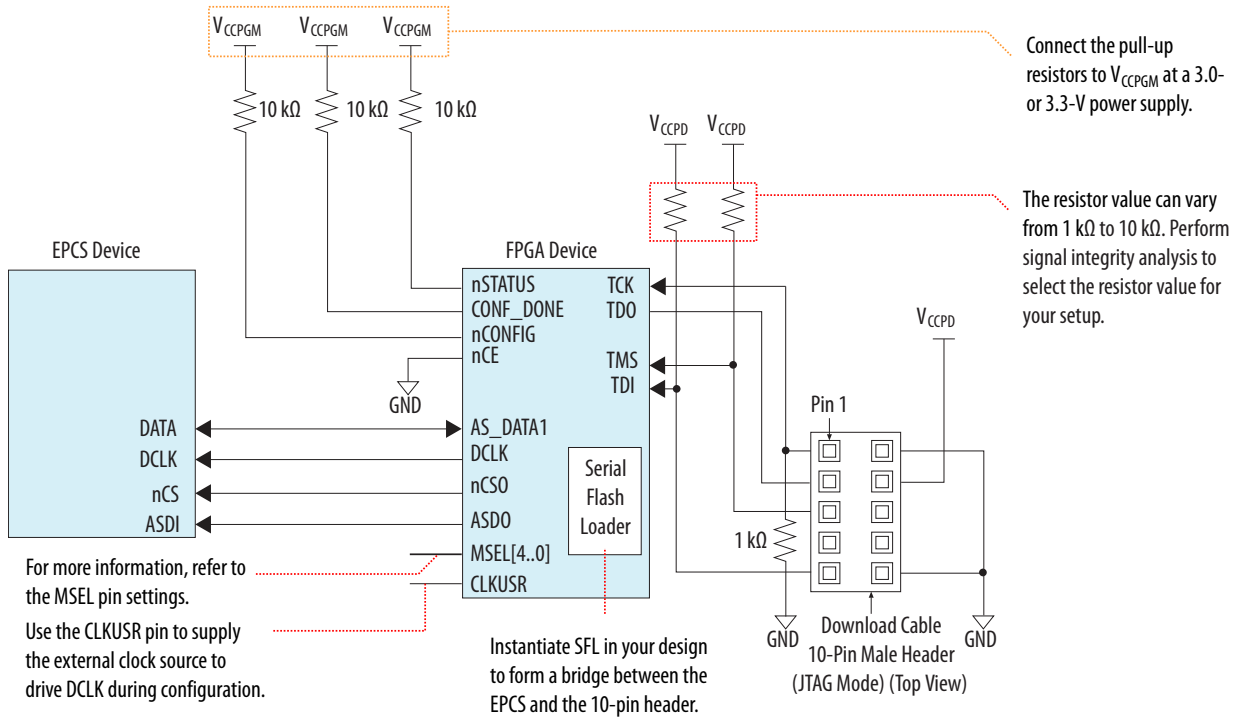
Related Information

- AN 370: Using the Serial FlashLoader with the Quartus II Software
- AN 418: SRunner: An Embedded Solution for Serial Configuration Device Programming

Programming EPCS Using the JTAG Interface

To program an EPCS device using the JTAG interface, connect the device as shown in the following figure.

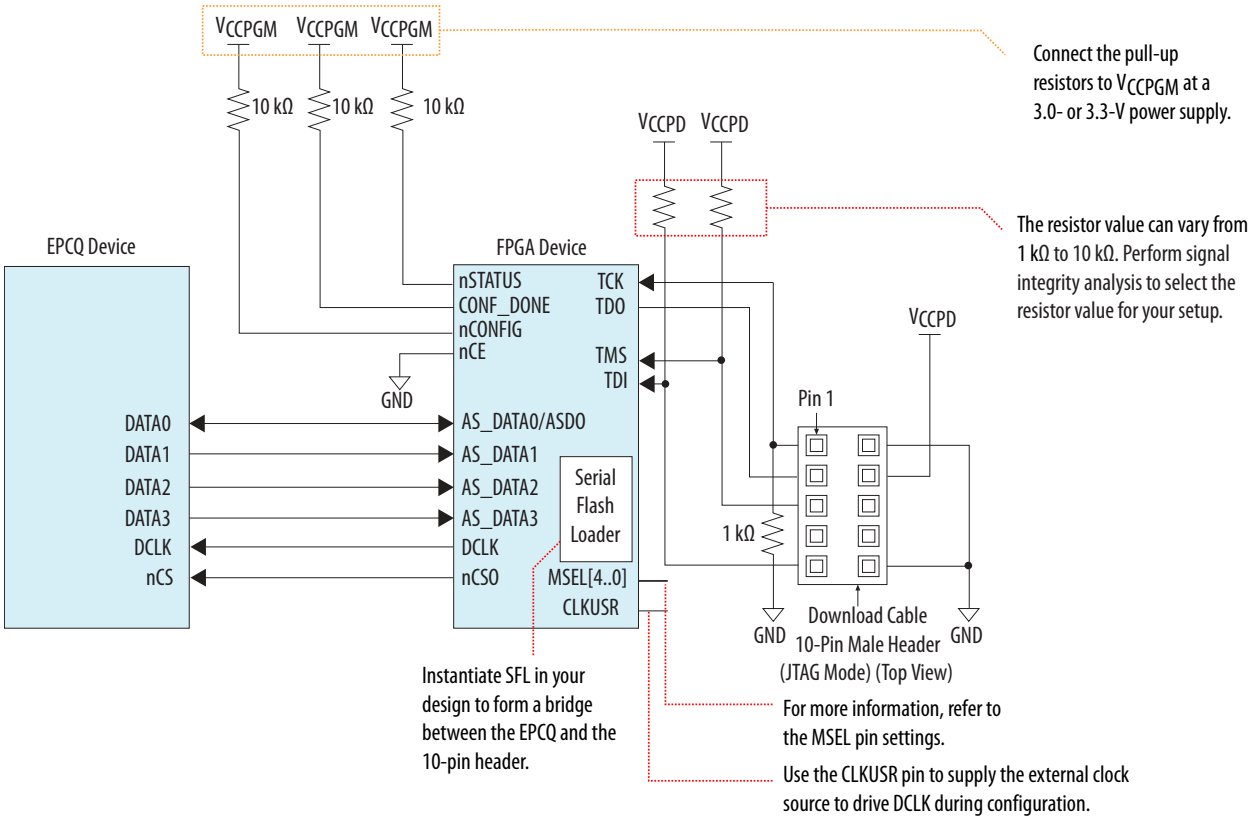
Figure 8-12: Connection Setup for Programming the EPCS Using the JTAG Interface



Programming EPCQ Using the JTAG Interface

To program an EPCQ device using the JTAG interface, connect the device as shown in the following figure.

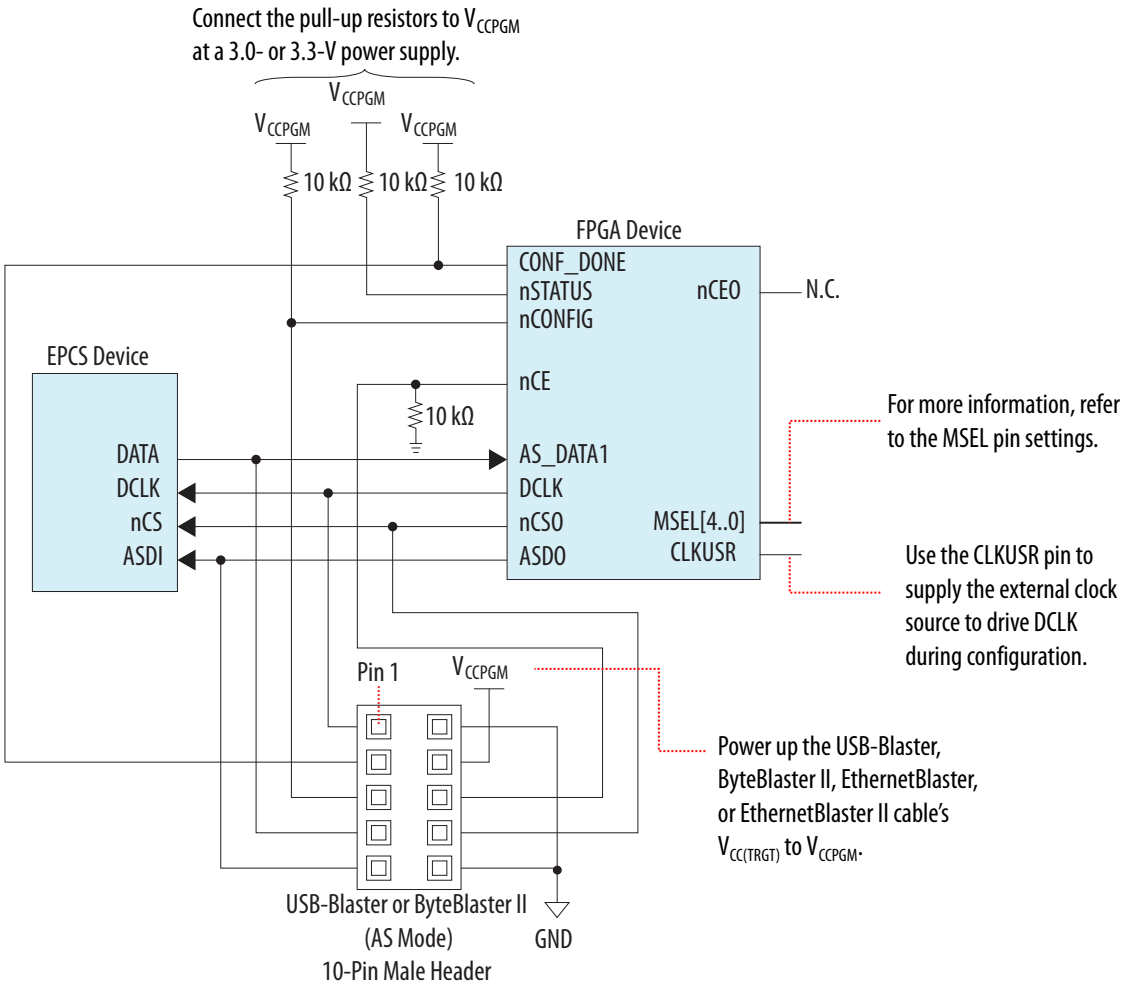
Figure 8-13: Connection Setup for Programming the EPCQ Using the JTAG Interface



Programming EPCS Using the Active Serial Interface

To program an EPCS device using the AS interface, connect the device as shown in the following figure.

Figure 8-14: Connection Setup for Programming the EPCS Using the AS Interface

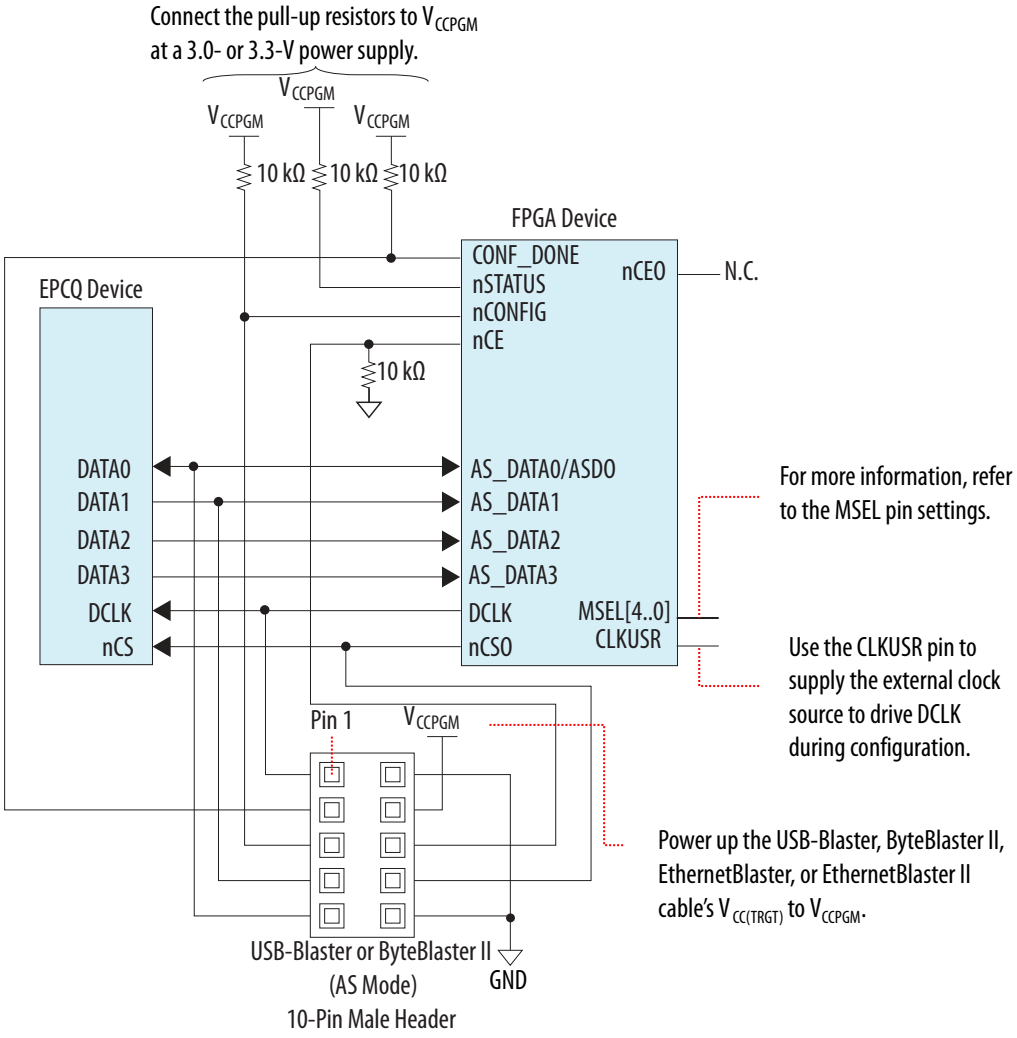


Programming EPCQ Using the Active Serial Interface

To program an EPCQ device using the AS interface, connect the device as shown in the following figure.

Figure 8-15: Connection Setup for Programming the EPCQ Using the AS Interface

Using the AS header, the programmer serially transmits the operation commands and configuration bits to the EPCQ on DATA0. This is equivalent to the programming operation for the EPCS.



When programming the EPCS and EPCQ devices, the download cable disables access to the AS interface by driving the nCE pin high. The nCONFIG line is also pulled low to hold the Arria V device in the reset stage. After programming completes, the download cable releases nCE and nCONFIG, allowing the pull-down and pull-up resistors to drive the pin to GND and V_{CCPGM}, respectively.

During the EPCQ programming using the download cable, DATA0 transfers the programming data, operation command, and address information from the download cable into the EPCQ. During the EPCQ verification using the download cable, DATA1 transfers the programming data back to the download cable.

Passive Serial Configuration

The PS configuration scheme uses an external host. You can use a microprocessor, MAX II device, MAX V device, or a host PC as the external host.

You can use an external host to control the transfer of configuration data from an external storage such as flash memory to the FPGA. The design that controls the configuration process resides in the external host.

You can store the configuration data in Programmer Object File (**.pof**), **.rbf**, **.hex**, or **.ttf**. If you are using configuration data in **.rbf**, **.hex**, or **.ttf**, send the LSB of each data byte first. For example, if the **.rbf** contains the byte sequence 02 1B EE 01 FA, the serial data transmitted to the device must be 0100-0000 1101-1000 0111-0111 1000-0000 0101-1111.

You can use the PFL IP core with a MAX II or MAX V device to read configuration data from the flash memory device and configure the Arria V device.

For a PC host, connect the PC to the device using a download cable such as the Altera USB-Blaster USB port, ByteBlaster II parallel port, EthernetBlaster, and EthernetBlaster II download cables.

The configuration data is shifted serially into the `DATA0` pin of the device.

If you are using the Intel Quartus Prime programmer and the `CLKUSR` pin is enabled, you do not need to provide a clock source for the pin to initialize your device.

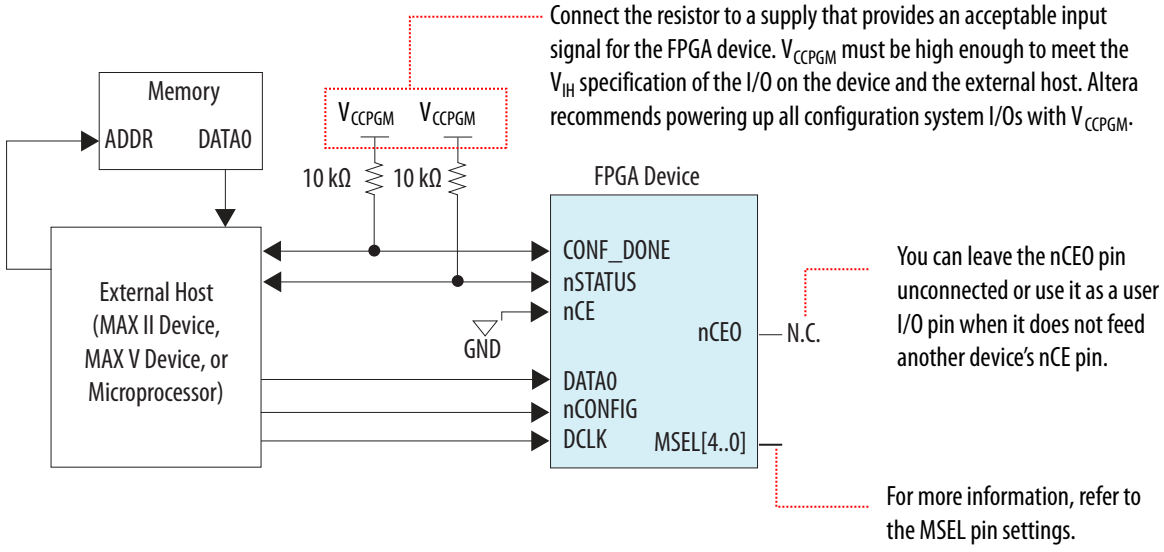
Related Information

- **Parallel Flash Loader IP Core User Guide**
- **Arria V Device Datasheet**
  Provides more information about the PS configuration timing.

Passive Serial Single-Device Configuration Using an External Host

To configure an Arria V device, connect the device to an external host, as shown in the following figure.

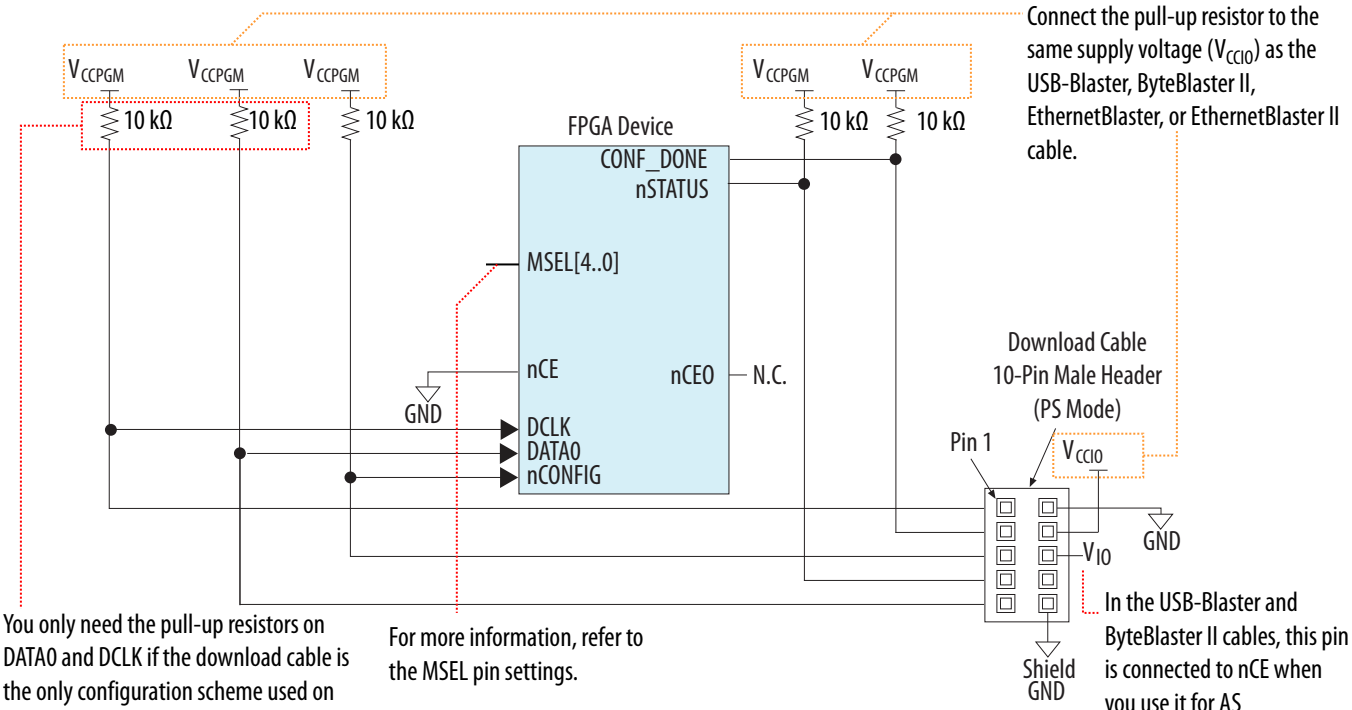
Figure 8-16: Single Device PS Configuration Using an External Host



Passive Serial Single-Device Configuration Using an Altera Download Cable

To configure an Arria V device, connect the device to a download cable, as shown in the following figure.

Figure 8-17: Single Device PS Configuration Using an Altera Download Cable



Passive Serial Multi-Device Configuration

You can configure multiple Arria V devices that are connected in a chain.

Pin Connections and Guidelines

Observe the following pin connections and guidelines for this configuration setup:

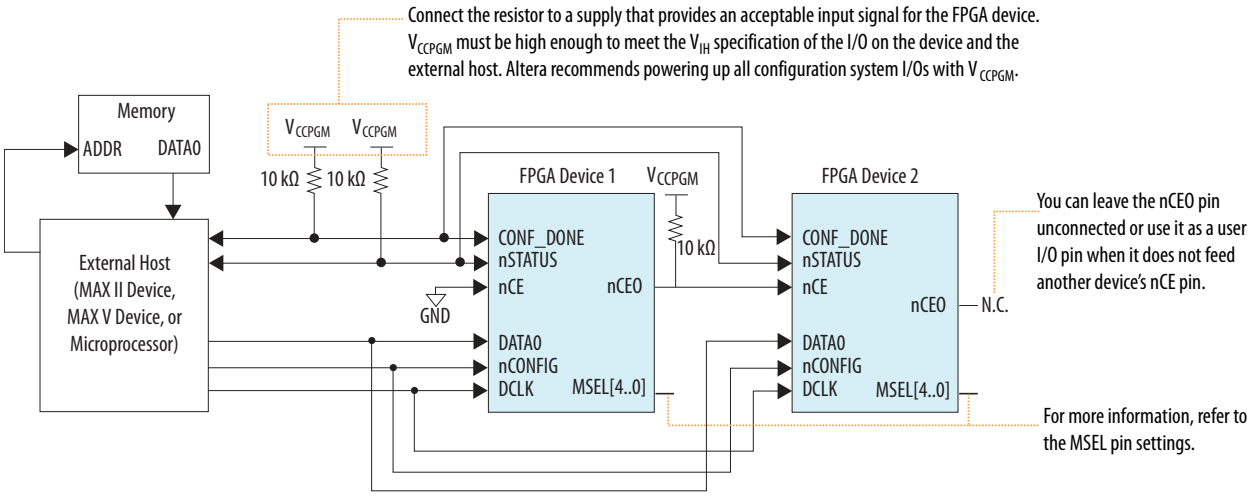
- Tie the following pins of all devices in the chain together:
  - nCONFIG
  - nSTATUS
  - DCLK
  - DATA0
  - CONF_DONE

  By tying the CONF_DONE and nSTATUS pins together, the devices initialize and enter user mode at the same time. If any device in the chain detects an error, configuration stops for the entire chain and you must reconfigure all the devices. For example, if the first device in the chain flags an error on the nSTATUS pin, it resets the chain by pulling its nSTATUS pin low.
- If you are configuring the devices in the chain using the same configuration data, the devices must be of the same package and density.

Using Multiple Configuration Data

To configure multiple Arria V devices in a chain using multiple configuration data, connect the devices to the external host as shown in the following figure.

Figure 8-18: Multiple Device PS Configuration when Both Devices Receive Different Sets of Configuration Data

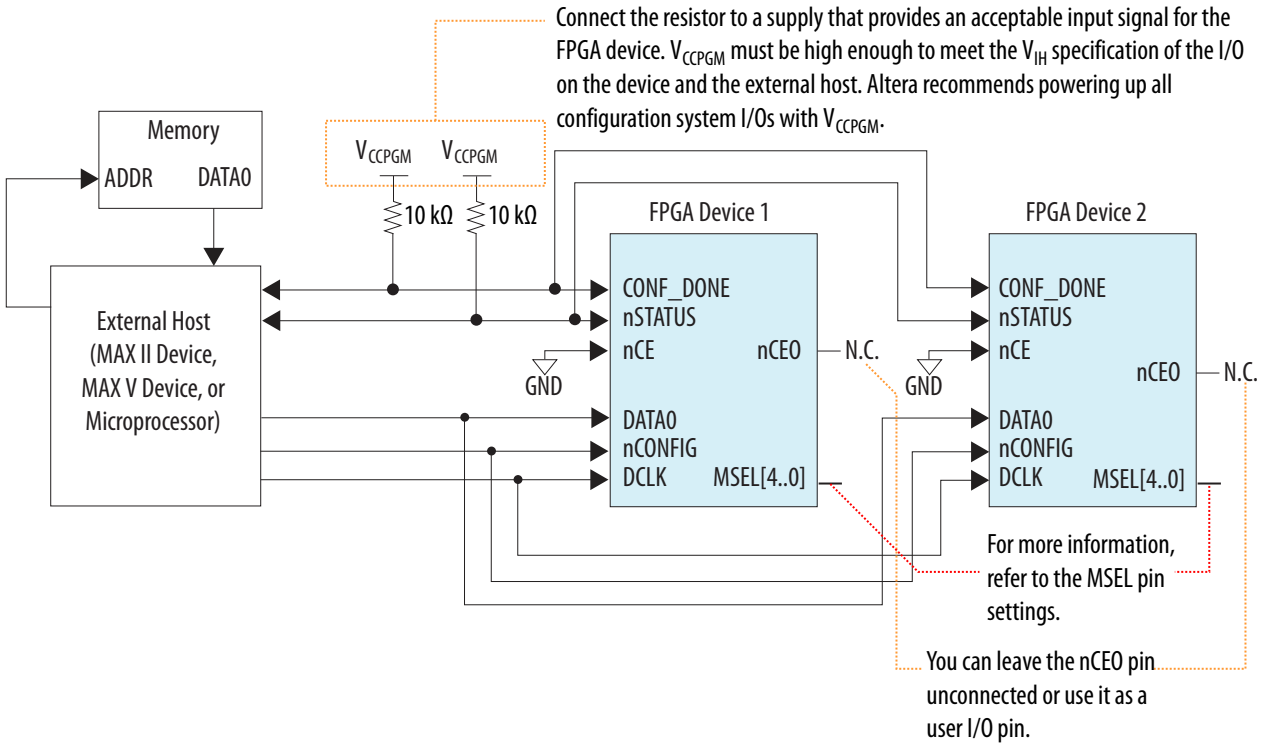


After a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device in the chain. Configuration automatically begins for the second device in one clock cycle.

Using One Configuration Data

To configure multiple Arria V devices in a chain using one configuration data, connect the devices to an external host, as shown in the following figure.

Figure 8-19: Multiple Device PS Configuration When Both Devices Receive the Same Set of Configuration Data

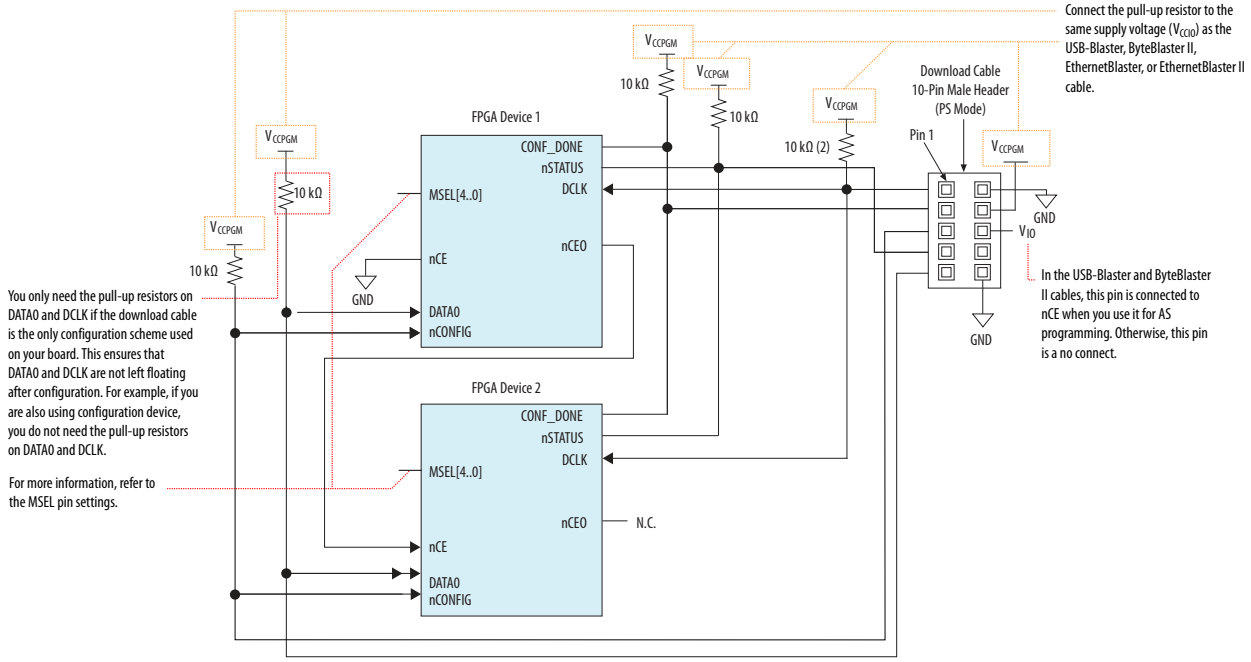


The nCE pins of the devices in the chain are connected to GND, allowing configuration for these devices to begin and end at the same time.

Using PC Host and Download Cable

To configure multiple Arria V devices, connect the devices to a download cable, as shown in the following figure.

Figure 8-20: Multiple Device PS Configuration Using an Altera Download Cable



When a device completes configuration, its nCEO pin is released low to activate the nCE pin of the next device. Configuration automatically begins for the second device.

JTAG Configuration

In Arria V devices, JTAG instructions take precedence over other configuration schemes.

The Intel Quartus Prime software generates an SRAM Object File (**.sof**) that you can use for JTAG configuration using a download cable in the Intel Quartus Prime software programmer. Alternatively, you can use the JRunner software with **.rbf** or a JAM™ Standard Test and Programming Language (STAPL) Format File (**.jam**) or JAM Byte Code File (**.jbc**) with other third-party programmer tools.

Related Information

- **JTAG Boundary-Scan Testing in Arria V Devices** on page 10-1
  Provides more information about JTAG boundary-scan testing.
- **Device Configuration Pins** on page 8-12
  Provides more information about JTAG configuration pins.
- **JTAG Secure Mode** on page 8-45
- **AN 425: Using the Command-Line Jam STAPL Solution for Device Programming**
- **Arria V Device Datasheet**
  Provides more information about the JTAG configuration timing.
- **Programming Support for Jam STAPL Language**
- **USB-Blaster Download Cable User Guide**
- **ByteBlaster II Download Cable User Guide**

- EthernetBlaster Communications Cable User Guide
- EthernetBlaster II Communications Cable User Guide

JTAG Single-Device Configuration

To configure a single device in a JTAG chain, the programming software sets the other devices to the bypass mode. A device in a bypass mode transfers the programming data from the TDI pin to the TDO pin through a single bypass register. The configuration data is available on the TDO pin one clock cycle later.

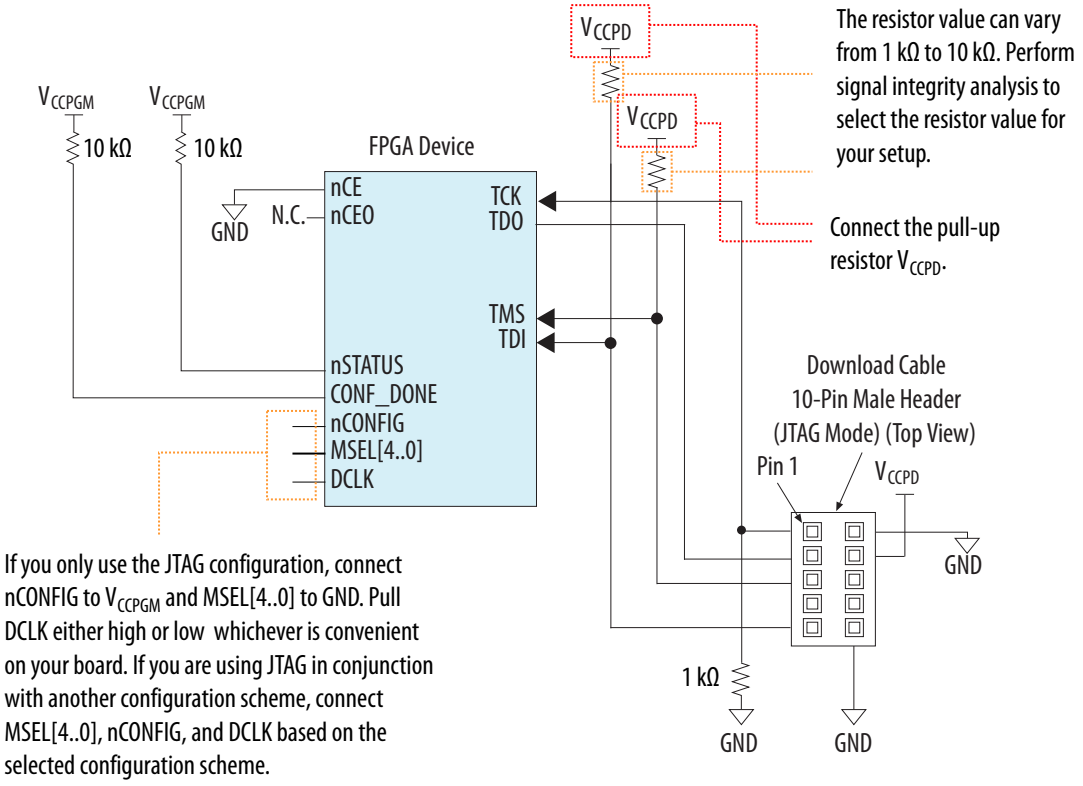
The Intel Quartus Prime software can use the CONF_DONE pin to verify the completion of the configuration process through the JTAG port:

- CONF_DONE pin is low—indicates that configuration has failed.
- CONF_DONE pin is high—indicates that configuration was successful.

After the configuration data is transmitted serially using the JTAG TDI port, the TCK port is clocked an additional 1,222 cycles to perform device initialization.

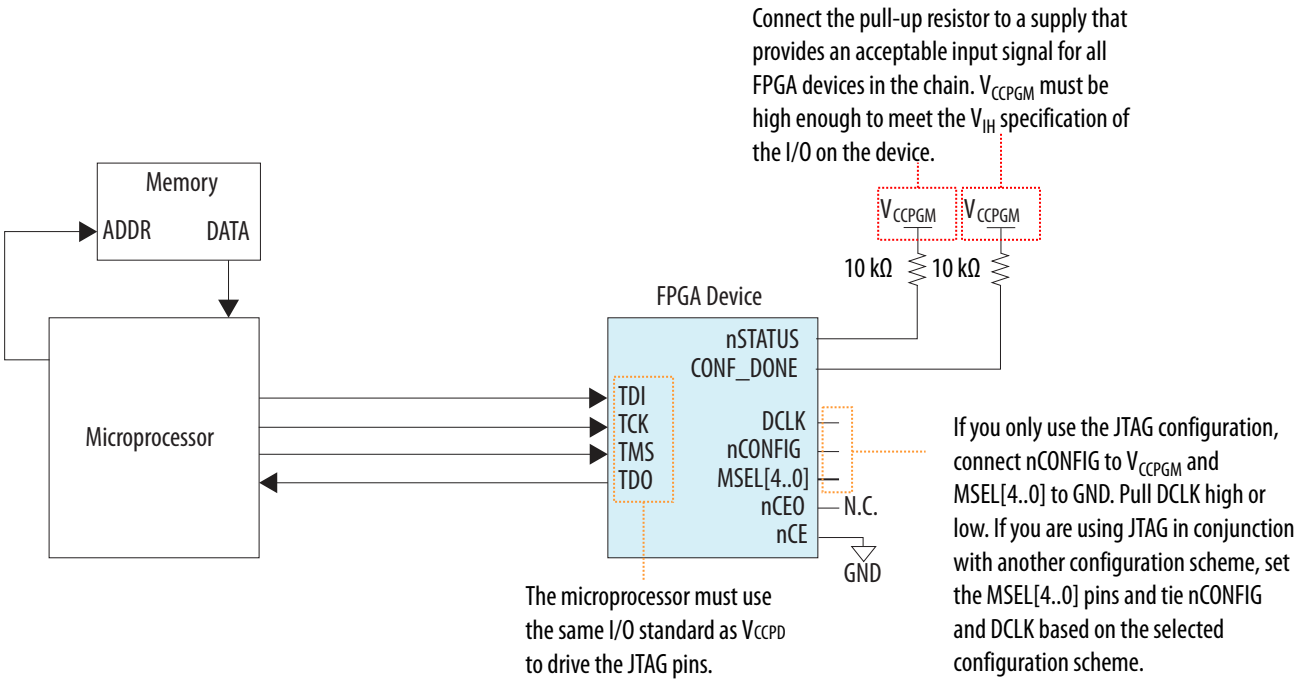
To configure an Arria V device using a download cable, connect the device as shown in the following figure.

Figure 8-21: JTAG Configuration of a Single Device Using a Download Cable



To configure Arria V device using a microprocessor, connect the device as shown in the following figure. You can use JRunner as your software driver.

Figure 8-22: JTAG Configuration of a Single Device Using a Microprocessor



Related Information

AN 414: The JRunner Software Driver: An Embedded Solution for PLD JTAG Configuration

JTAG Multi-Device Configuration

You can configure multiple devices in a JTAG chain.

Pin Connections and Guidelines

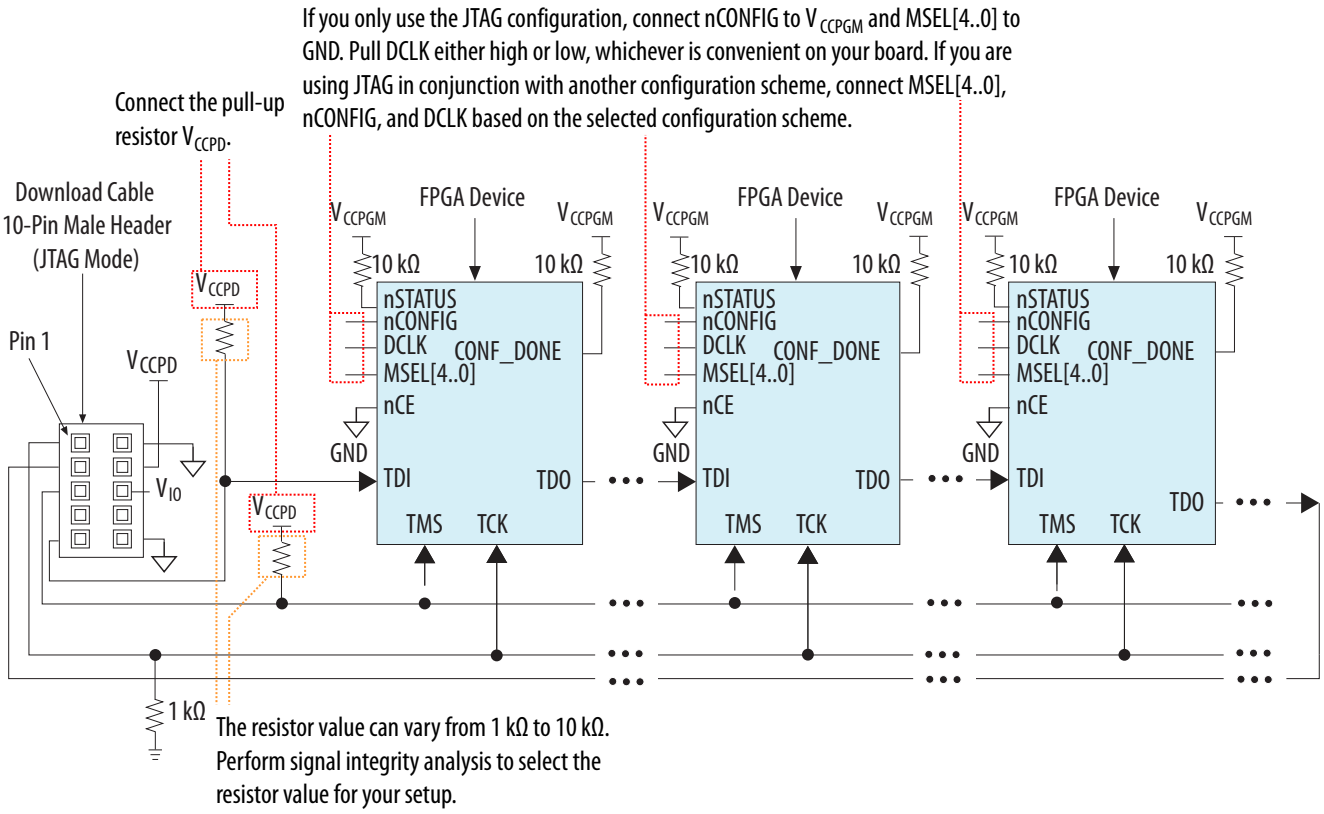
Observe the following pin connections and guidelines for this configuration setup:

- Isolate the `CONF_DONE` and `nSTATUS` pins to allow each device to enter user mode independently.
- One JTAG-compatible header is connected to several devices in a JTAG chain. The number of devices in the chain is limited only by the drive capability of the download cable.
- If you have four or more devices in a JTAG chain, buffer the `TCK`, `TDI`, and `TMS` pins with an on-board buffer. You can also connect other Altera devices with JTAG support to the chain.
- JTAG-chain device programming is ideal when the system contains multiple devices or when testing your system using the JTAG boundary-scan testing (BST) circuitry.

Using a Download Cable

The following figure shows a multi-device JTAG configuration.

Figure 8-23: JTAG Configuration of Multiple Devices Using a Download Cable



Related Information

AN 656: Combining Multiple Configuration Schemes
Provides more information about combining JTAG configuration with other configuration schemes.

CONFIG_IO JTAG Instruction

The CONFIO_IO JTAG instruction allows you to configure the I/O buffers using the JTAG port before or during device configuration. When you issue this instruction, it interrupts configuration and allows you to issue all JTAG instructions. Otherwise, you can only issue the BYPASS, IDCODE, and SAMPLE JTAG instructions.

You can use the CONFIO_IO JTAG instruction to interrupt configuration and perform board-level testing. After the board-level testing is completed, you must reconfigure your device. Use the following methods to reconfigure your device:

- JTAG interface—issue the PULSE_NCONFIG JTAG instruction.
- FPP, PS, or AS configuration scheme—pulse the nCONFIG pin low.

Configuration Data Compression

Arria V devices can receive compressed configuration bitstream and decompress the data in real-time during configuration. Preliminary data indicates that compression typically reduces the configuration file size by 30% to 55% depending on the design.

Decompression is supported in all configuration schemes except the JTAG configuration scheme.

You can enable compression before or after design compilation.

Enabling Compression Before Design Compilation

To enable compression before design compilation, follow these steps:

1. On the Assignment Menu, click **Device**.
2. Select your Arria V device and then click **Device and Pin Options**.
3. In the **Device and Pin Options** window, select **Configuration** under the **Category** list and turn on **Generate compressed bitstreams**.

Enabling Compression After Design Compilation

To enable compression after design compilation, follow these steps:

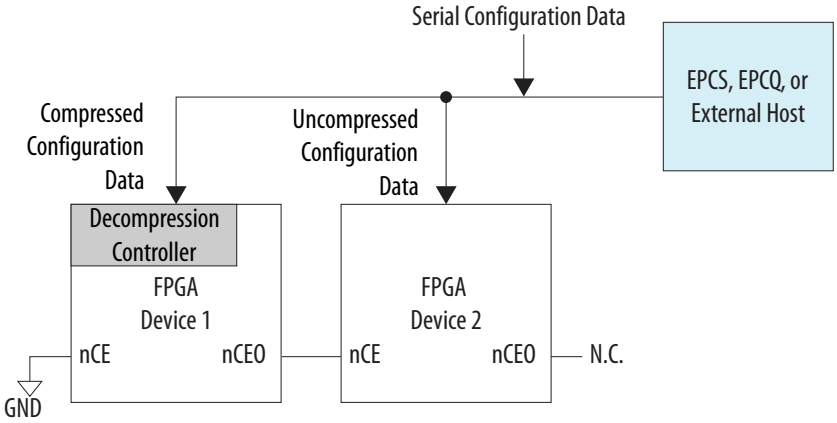
1. On the File menu, click **Convert Programming Files**.
2. Select the programming file type (**.pof**, **.sof**, **.hex**, **.hexout**, **.rbf**, or **.ttf**). For POF output files, select a configuration device.
3. Under the **Input files to convert** list, select **SOF Data**.
4. Click **Add File** and select an Arria V device **.sof**.
5. Select the name of the file you added to the **SOF Data** area and click **Properties**.
6. Turn on the **Compression** check box.

Using Compression in Multi-Device Configuration

The following figure shows a chain of two Arria V devices. Compression is only enabled for the first device.

This setup is supported by the AS or PS multi-device configuration only.

Figure 8-24: Compressed and Uncompressed Serial Configuration Data in the Same Configuration File

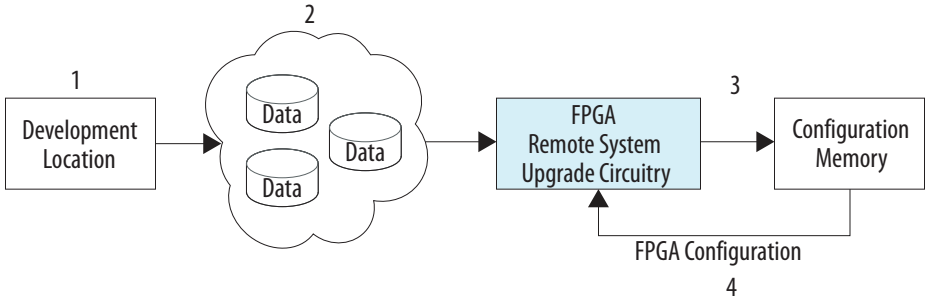


For the FPP configuration scheme, a combination of compressed and uncompressed configuration in the same multi-device configuration chain is not allowed because of the difference on the DCLK-to-DATA[] ratio.

Remote System Upgrades

Arria V devices contain dedicated remote system upgrade circuitry. You can use this feature to upgrade your system from a remote location.

Figure 8-25: Arria V Remote System Upgrade Block Diagram



You can design your system to manage remote upgrades of the application configuration images in the configuration device. The following list is the sequence of the remote system upgrade:

1. The logic (embedded processor or user logic) in the Arria V device receives a configuration image from a remote location. You can connect the device to the remote source using communication protocols such as TCP/IP, PCI, user datagram protocol (UDP), UART, or a proprietary interface.
2. The logic stores the configuration image in non-volatile configuration memory.
3. The logic starts reconfiguration cycle using the newly received configuration image.
4. When an error occurs, the circuitry detects the error, reverts to a safe configuration image, and provides error status to your design.

Configuration Images

Each Arria V device in your system requires one factory image. The factory image is a user-defined configuration image that contains logic to perform the following:

- Processes errors based on the status provided by the dedicated remote system upgrade circuitry.
- Communicates with the remote host, receives new application images, and stores the images in the local non-volatile memory device.
- Determines the application image to load into the Arria V device.
- Enables or disables the user watchdog timer and loads its time-out value.
- Instructs the dedicated remote system upgrade circuitry to start a reconfiguration cycle.

You can also create one or more application images for the device. An application image contains selected functionalities to be implemented in the target device.

Store the images at the following locations in the EPCS or EPCQ devices:

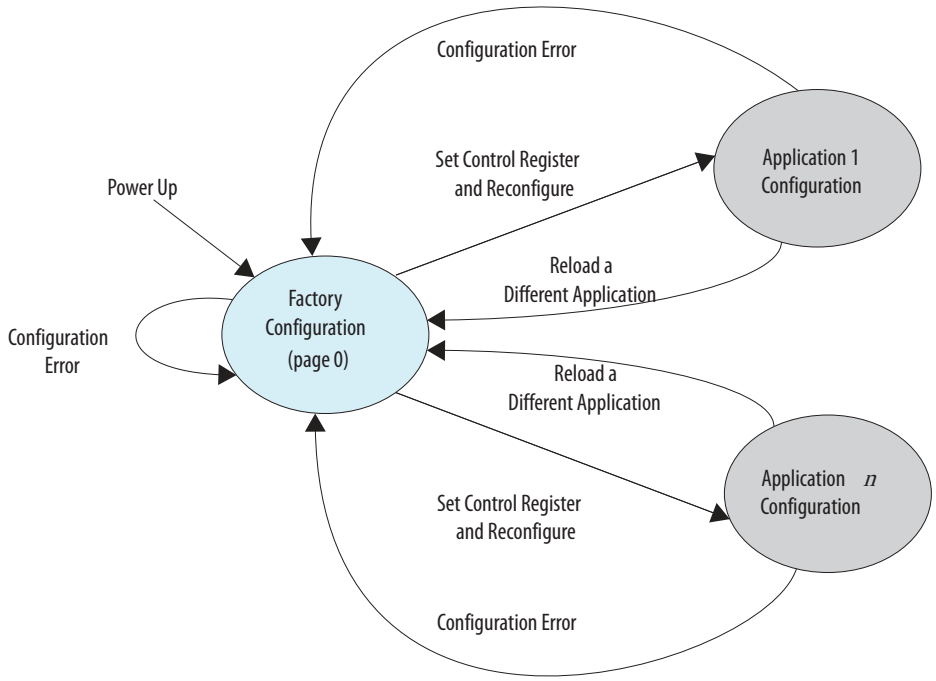
- Factory configuration image—`PGM[23..0]` = 24'h000000 start address on the EPCS or EPCQ device.
- Application configuration image—any sector boundary. Altera recommends that you store only one image at one sector boundary.

When you are using EPCQ 256, ensure that the application configuration image address granularity is 32'h00000100. The granularity requirement is having the most significant 24 bits of the 32 bits start address written to `PGM[23..0]` bits.

Note: If you are not using the Intel Quartus Prime software or SRunner software for EPCQ 256 programming, put your EPCQ 256 device into four-byte addressing mode before you program and configure your device.

Configuration Sequence in the Remote Update Mode

Figure 8-26: Transitions Between Factory and Application Configurations in Remote Update Mode



Related Information

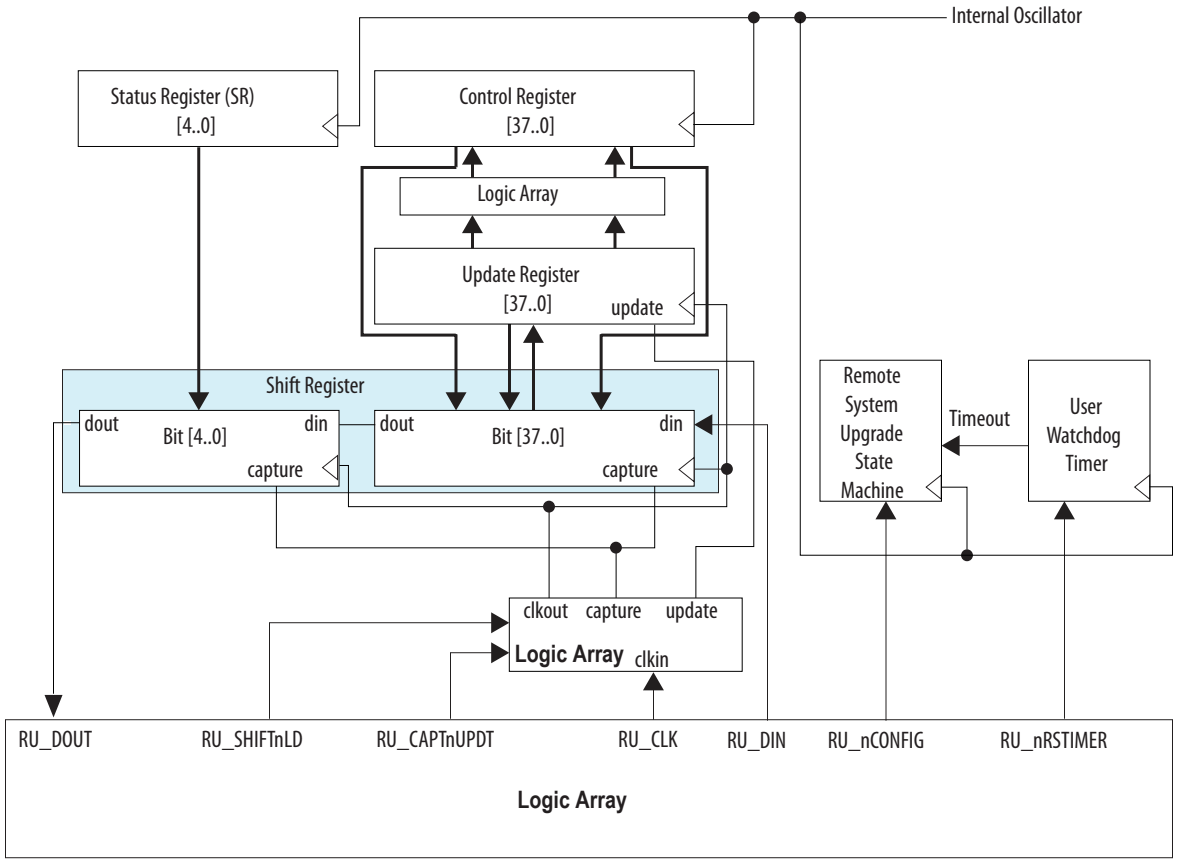
Remote System Upgrade State Machine on page 8-43
A detailed description of the configuration sequence in the remote update mode.

Remote System Upgrade Circuitry

The remote system upgrade circuitry contains the remote system upgrade registers, watchdog timer, and a state machine that controls these components.

Note: If you are using the Altera Remote Update IP core, the IP core controls the `RU_DOUT`, `RU_SHIFTnLD`, `RU_CAPTnUPDT`, `RU_CLK`, `RU_DIN`, `RU_nCONFIG`, and `RU_nRSTIMER` signals internally to perform all the related remote system upgrade operations.

Figure 8-27: Remote System Upgrade Circuitry



Related Information

Arria V Device Datasheet
Provides more information about remote system upgrade circuitry timing specifications.

Enabling Remote System Upgrade Circuitry

To enable the remote system upgrade feature, follow these steps:

1. Select **Active Serial x1/x4** or **Configuration Device** from the Configuration scheme list in the **Configuration** page of the **Device and Pin Options** dialog box in the Intel Quartus Prime software.
2. Select **Remote** from the Configuration mode list in the **Configuration** page of the **Device and Pin Options** dialog box in the Intel Quartus Prime software.

Enabling this feature automatically turns on the **Auto-restart configuration after error** option.

Remote Update Intel FPGA IP core provides a memory-like interface to the remote system upgrade circuitry and handles the shift register read and write protocol in the Arria V device logic.

Related Information

Altera Remote Update IP Core User Guide

Remote System Upgrade Registers

Table 8-9: Remote System Upgrade Registers

Register	Description
Shift	Accessible by the logic array and clocked by RU_CLK. • Bits[4..0]—Contents of the status register are shifted into these bits. • Bits[37..0]—Contents of the update and control registers are shifted into these bits.
Control	This register is clocked by the 10-MHz internal oscillator. The contents of this register are shifted to the shift register for the user logic in the application configuration to read. When reconfiguration is triggered, this register is updated with the contents of the update register.
Update	This register is clocked by RU_CLK. The factory configuration updates this register by shifting data into the shift register and issuing an update. When reconfiguration is triggered, the contents of the update register are written to the control register.
Status	After each reconfiguration, the remote system upgrade circuitry updates this register to indicate the event that triggered the reconfiguration. This register is clocked by the 10-MHz internal oscillator.

Related Information

- **Control Register** on page 8-42
- **Status Register** on page 8-43

Control Register

Table 8-10: Control Register Bits

Bit	Name	Reset Value[34]	Description
0	AnF	1'b0	Application not Factory bit. Indicates the configuration image type currently loaded in the device; **0** for factory image and **1** for application image. When this bit is **1**, the access to the control register is limited to read only and the watchdog timer is enabled. Factory configuration design must set this bit to **1** before triggering reconfiguration using an application configuration image.

[34] This is the default value after the device exits POR and during reconfiguration back to the factory configuration image.

Bit	Name	Reset Value[34]	Description
1..24	PGM[0..23]	24'h000000	Upper 24 bits of AS configuration start address (StAdd[31..8]), the 8 LSB are zero.
25	Wd_en	1'b0	User watchdog timer enable bit. Set this bit to **1** to enable the watchdog timer.
26..37	Wd_timer[11..0]	12'b000000000000	User watchdog time-out value.

Status Register

Table 8-11: Status Register Bits

Bit	Name	Reset Value[35]	Description
0	CRC	1'b0	When set to **1**, indicates CRC error during application configuration.
1	nSTATUS	1'b0	When set to **1**, indicates that nSTATUS is asserted by an external device due to error.
2	Core_nCONFIG	1'b0	When set to **1**, indicates that reconfiguration has been triggered by the logic array of the device.
3	nCONFIG	1'b0	When set to **1**, indicates that nCONFIG is asserted.
4	Wd	1'b0	When set to **1**, indicates that the user watchdog time-out.

Remote System Upgrade State Machine

The operation of the remote system upgrade state machine is as follows:

1. After power-up, the remote system upgrade registers are reset to **0** and the factory configuration image is loaded.
2. The user logic sets the AnF bit to **1** and the start address of the application image to be loaded. The user logic also writes the watchdog timer settings.
3. When the configuration reset (RU_CONFIG) goes low, the state machine updates the control register with the contents of the update register, and triggers reconfiguration using the application configuration image.
4. If error occurs, the state machine falls back to the factory image. The control and update registers are reset to **0**, and the status register is updated with the error information.
5. After successful reconfiguration, the system stays in the application configuration.

User Watchdog Timer

[34] This is the default value after the device exits POR and during reconfiguration back to the factory configuration image.

[35] After the device exits POR and power-up, the status register content is 5'b00000.

The user watchdog timer prevents a faulty application configuration from stalling the device indefinitely. You can use the timer to detect functional errors when an application configuration is successfully loaded into the device. The timer is automatically disabled in the factory configuration; enabled in the application configuration.

Note: If you do not want this feature in the application configuration, you need to turn off this feature by setting the `Wd_en` bit to **1'b0** in the update register during factory configuration user mode operation. You cannot disable this feature in the application configuration.

The counter is 29 bits wide and has a maximum count value of 2^{29}. When specifying the user watchdog timer value, specify only the most significant 12 bits. The granularity of the timer setting is 2^{17} cycles. The cycle time is based on the frequency of the user watchdog timer internal oscillator.

The timer begins counting as soon as the application configuration enters user mode. When the timer expires, the remote system upgrade circuitry generates a time-out signal, updates the status register, and triggers the loading of the factory configuration image. To reset the time, assert `RU_nRSTIMER`.

Related Information

Arria V Device Datasheet
Provides more information about the operating range of the user watchdog internal oscillator's frequency.

Design Security

The Arria V design security feature supports the following capabilities:

- Enhanced built-in advanced encryption standard (AES) decryption block to support 256-bit key industry-standard design security algorithm (FIPS-197 Certified)
- Volatile and non-volatile key programming support
- Secure operation mode for both volatile and non-volatile key through tamper protection bit setting
- Limited accessible JTAG instruction during power-up in the JTAG secure mode
- Supports board-level testing
- Supports in-socket key programming for non-volatile key
- Available in all configuration schemes except JTAG
- Supports both remote system upgrades and compression features

The Arria V design security feature provides the following security protection for your designs:

- Security against copying—the security key is securely stored in the Arria V device and cannot be read out through any interface. In addition, as configuration file read-back is not supported in Arria V devices, your design information cannot be copied.
- Security against reverse engineering—reverse engineering from an encrypted configuration file is very difficult and time consuming because the Arria V configuration file formats are proprietary and the file contains millions of bits that require specific decryption.
- Security against tampering—After you set the tamper protection bit, the Arria V device can only accept configuration files encrypted with the same key. Additionally, programming through the JTAG interface and configuration interface is blocked.

When you use compression with the design security feature, the configuration file is first compressed and then encrypted using the Intel Quartus Prime software. During configuration, the device first decrypts and then decompresses the configuration file.

When you use design security with Arria V devices in an FPP configuration scheme, it requires a different DCLK-to-DATA[] ratio.

Altera Unique Chip ID IP Core

The Altera Unique Chip ID IP core provides the following features:

- Acquiring the chip ID of an FPGA device.
- Allowing you to identify your device in your design as part of a security feature to protect your design from an unauthorized device.

Related Information

Altera Unique Chip ID IP Core User Guide

JTAG Secure Mode

When you enable the tamper-protection bit, Arria V devices are in the JTAG secure mode after power-up. During this mode, many JTAG instructions are disabled. Arria V devices only allow mandatory JTAG 1149.1 instructions to be exercised. These JTAG instructions are SAMPLE/PRELOAD, BYPASS, EXTEST, and optional instructions such as IDCODE and SHIFT_EDERROR_REG.

To enable the access of other JTAG instructions such as USERCODE, HIGHZ, CLAMP, PULSE_nCONFIG, and CONFIG_IO, you must issue the UNLOCK instruction to deactivate the JTAG secure mode. You can issue the LOCK instruction to put the device back into JTAG secure mode. You can only issue both the LOCK and UNLOCK JTAG instructions during user mode.

Related Information

Supported JTAG Instruction on page 10-3
Provides more information about JTAG binary instruction code related to the LOCK and UNLOCK instructions.

Security Key Types

Arria V devices offer two types of keys—volatile and non-volatile. The following table lists the differences between the volatile key and non-volatile keys.

Table 8-12: Security Key Types

Key Types	Key Programmability	Power Supply for Key Storage	Programming Method
Volatile	• Reprogrammable • Erasable	Required external battery, V_{CCBAT} (36)	On-board
Non-volatile	One-time programming	Does not require an external battery	On-board and in-socket programming (37)

(36) V_{CCBAT} is a dedicated power supply for volatile key storage. V_{CCBAT} continuously supplies power to the volatile register regardless of the on-chip supply condition.
(37) Third-party vendors offer in-socket programming.

Both non-volatile and volatile key programming offers protection from reverse engineering and copying. If you set the tamper-protection bit, the design is also protected from tampering.

You can perform key programming through the JTAG pins interface. Ensure that the nSTATUS pin is released high before any key-programming attempts.

Note: To clear the volatile key, issue the KEY_CLR_VREG JTAG instruction. To verify the volatile key has been cleared, issue the KEY_VERIFY JTAG instruction.

Related Information

- **Supported JTAG Instruction** on page 10-3
  Provides more information about the KEY_CLR_VREG and KEY_VERIFY instructions.
- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about the V_{CCBAT} pin connection recommendations.
- **Arria V GZ Device Family Pin Connection Guidelines**
  Provides more information about the V_{CCBAT} pin connection recommendations.
- **Arria V Device Datasheet**
  Provides more information about battery specifications.

Security Modes

Table 8-13: Supported Security Modes

There is no impact to the configuration time required when compared with unencrypted configuration schemes except FPP with AES (and/or decompression), which requires a DCLK that is up to ×4 the data rate.

Security Mode	Tamper Protection Bit Setting	Device Accepts Unencrypted File	Device Accepts Encrypted File	Security Level
No key	—	Yes	No	—
Volatile Key	—	Yes	Yes	Secure
Volatile Key with Tamper Protection Bit Set	Set	No	Yes	Secure with tamper resistant
Non-volatile Key	—	Yes	Yes	Secure
Non-volatile Key with Tamper Protection Bit Set	Set	No	Yes	Secure with tamper resistant

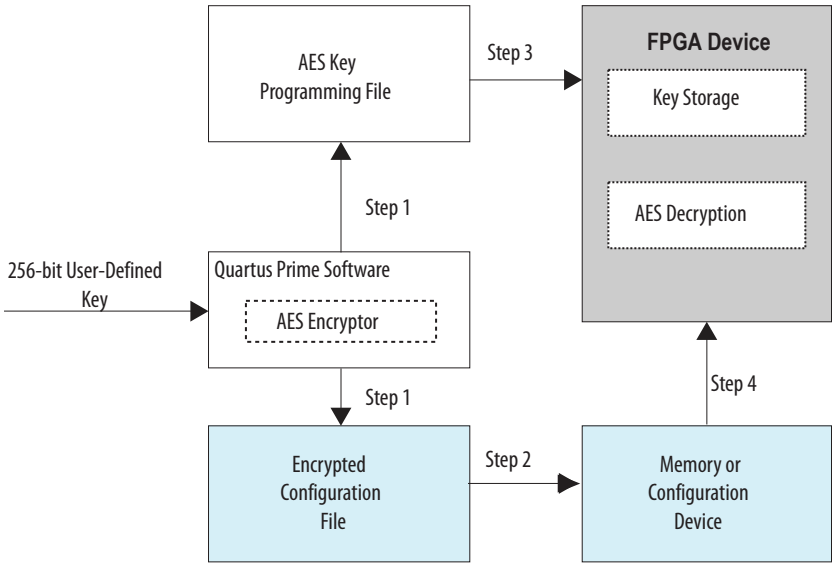
The use of unencrypted configuration bitstream in the volatile key and non-volatile key security modes is supported for board-level testing only.

Note: For the volatile key with tamper protection bit set security mode, Arria V devices do not accept the encrypted configuration file if the volatile key is erased. If the volatile key is erased and you want to reprogram the key, you must use the volatile key security mode.

Enabling the tamper protection bit disables the test mode in Arria V devices and disables programming through the JTAG interface. This process is irreversible and prevents Altera from carrying out failure analysis.

Design Security Implementation Steps

Figure 8-28: Design Security Implementation Steps



To carry out secure configuration, follow these steps:

1. The Intel Quartus Prime software generates the design security key programming file and encrypts the configuration data using the user-defined 256-bit security key.
2. Store the encrypted configuration file in the external memory.
3. Program the AES key programming file into the Arria V device through a JTAG interface.
4. Configure the Arria V device. At the system power-up, the external memory device sends the encrypted configuration file to the Arria V device.

Configuration, Design Security, and Remote System Upgrades in Arria V Devices Revision History

Document Version	Changes
2020.04.13	• Removed topic: *I/O Standards and Drive Strength for Configuration Pins*. • Updated the *User Mode* topic.
2019.10.03	Added a note to *Device Configuration Pins* to state that the DCLK, AS_DATA0/ASDO, AS_DATA1, AS_DATA2, AS_DATA3, and nCSO pins have 25 kOhm pull-up resistors when the MSEL pins are set to AS configuration scheme.

Document Version	Changes
2018.08.09	• Updated the *Active Serial Configuration* topic. • Updated Figure: *AS Configuration Timing Waveform*.

Date	Version	Changes
December 2017	2017.12.15	Added description in the *I/O Standards and Drive Strength for Configuration Pins* table.
December 2016	2016.12.09	Changed the term "configuration mode" to "configuration scheme" when referring to a configuration scheme.
August 2016	2016.08.24	Added note to Power Up and Reset states in the Configuration Sequence for Cyclone V Devices diagram.
June 2016	2016.06.10	• Added a note to specify the time between nCSO falling edge to first toggle of DCLK is more than 15ns in AS Configuration Timing figure.
December 2015	2015.12.21	• Changed instances of *Quartus II* to *Quartus Prime*. • Added the CvP_CONFDONE pin to the Configuration Pin Summary for Arria V Devices table. • Added the I/O Standards and Drive Strength for Configuration Pins table.
June 2015	2015.06.12	• Added timing waveforms for FPP, AS, and PS configuration. • Updated the Trace Length and Loading Guideline section.
January 2015	2015.01.23	• Added the Transmitting Configuration Data section. • Updated the Configuration Images section.
June 2014	2014.06.30	• Updated Figure 8-17: JTAG Configuration of a Single Device Using a Download Cable. • Updated Figure 8-19: JTAG Configuration of Multiple Devices Using a Download Cable. • Updated the maximum clock rate for Partial Reconfiguration in Table 8-1. • Updated the MSEL pin settings recommendation in the MSEL Pin Settings section.
January 2014	2014.01.10	• Added a link to the FPGA Manager chapter for details about the MSEL pin settings for the HPS in SoC FPGA devices. • Updated the V_{CCPD} Pin section. • Updated the Enabling Remote System Upgrade Circuitry section. • Updated the Configuration Pin Summary section. • Updated Figure 8-3, Figure 8-7, and Figure 8-14.

Date	Version	Changes
June 2013	2013.06.11	Updated the Configuration Error Handling section.
May 2013	2013.05.10	Removed support for active serial multi-device configuration using the same configuration data.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Added the ALTCHIP_ID megafunction section. • Updated "Connection Setup for Programming the EPCS Using the JTAG Interface" and "Connection Setup for Programming the EPCQ Using the JTAG Interface" figures. • Added the `nIO_PULLUP` pin in Table 8-3: Configuration Pin Summary for Arria V Devices. • Added links for AS, PS, FPP, and JTAG configuration timing to device datasheet. • Moved all links to the Related Information section of respective topics for easy reference.
November 2012	2012.11.19	• Added configuration modes and features for Arria V devices. • Added FPP x32 for Arria V GZ devices. • Added DATA[31..16] for Arria V GZ devices. • Reorganized content and updated template.
June 2012	2.0	Restructured the chapter.
November 2011	1.1	Minor text edits.
October 2011	1.0	Initial release.

SEU Mitigation for Arria V Devices

9

2020.04.13

AV-52009

Subscribe Send Feedback

This chapter describes the error detection features in Arria V devices. You can use these features to mitigate single event upset (SEU) or soft errors.

Related Information

Arria V Device Handbook: Known Issues
Lists the planned updates to the *Arria V Device Handbook* chapters.

Error Detection Features

The on-chip error detection CRC circuitry allows you to perform the following operations without any impact on the fitting or performance of the device:

- Auto-detection of CRC errors during configuration.
- Optional CRC error detection and identification in user mode.
- Testing of error detection functions by deliberately injecting errors through the JTAG interface.

Configuration Error Detection

When the Intel Quartus Prime software generates the configuration bitstream, the software also computes a 16-bit CRC value for each frame. A configuration bitstream can contain more than one CRC values depending on the number of data frames in the bitstream. The length of the data frame varies for each device.

When a data frame is loaded into the FPGA during configuration, the precomputed CRC value shifts into the CRC circuitry. At the same time, the CRC engine in the FPGA computes the CRC value for the data frame and compares it against the precomputed CRC value. If both CRC values do not match, the `nSTATUS` pin is set to low to indicate a configuration error.

You can test the capability of this feature by modifying the configuration bitstream or intentionally corrupting the bitstream during configuration.

User Mode Error Detection

In user mode, the contents of the configured CRAM bits may be affected by soft errors. These soft errors, which are caused by an ionizing particle, are not common in Altera devices. However, high-reliability

applications that require the device to operate error-free may require that your designs account for these errors.

You can enable the error detection circuitry to detect soft errors. Each data frame stored in the CRAM contains a 32-bit precomputed CRC value. When this feature is enabled, the error detection circuitry continuously computes a 32-bit CRC value for each frame in the CRAM and compares the CRC value against the precomputed value.

- If the CRC values match, the 32-bit CRC signature in the `syndrome` register is set to zero to indicate that no error is detected.
- Otherwise, the resulting 32-bit CRC signature in the `syndrome` register is non-zero to indicate a CRC error. The `CRC_ERROR` pin is pulled high, and the error type and location are identified.

Within a frame, the error detection circuitry can detect all single-, double-, triple-, quadruple-, and quintuple-bit errors. When a single-bit or double-adjacent error is detected, the error detection circuitry reports the bit location and determines the error type for single-bit and double-adjacent errors. The probability of other error patterns is very low and the reporting of bit location is not guaranteed. The probability of more than five CRAM bits being flipped by soft errors is very low. In general, the probability of detection for all error patterns is 99.9999%. The process of error detection continues until the device is reset by setting the `nCONFIG` signal low.

Specifications

This section lists the EMR update interval, error detection frequencies, and CRC calculation time for error detection in user mode.

Minimum EMR Update Interval

The interval between each update of the error message register depends on the device and the frequency of the error detection clock. Using a lower clock frequency increases the interval time, hence increasing the time required to recover from a single event upset (SEU).

Table 9-1: Estimated Minimum EMR Update Interval in Arria V Devices

Variant	Member Code	Timing Interval (µs)
Arria V GX	A1	2.55
	A3	2.55
	A5	2.87
	A7	2.87
	B1	3.13
	B3	3.13
	B5	3.83
	B7	3.83

Variant	Member Code	Timing Interval (µs)
Arria V GT	C3	2.55
	C7	2.87
	D3	3.13
	D7	3.83
Arria V GZ	E1	2.43
	E3	2.43
	E5	2.99
	E7	2.99
Arria V SX	B3	3.83
	B5	3.83
Arria V ST	D3	3.83
	D5	3.83

Error Detection Frequency

You can control the speed of the error detection process by setting the division factor of the clock frequency in the Intel Quartus Prime software. The divisor is 2^n, where n can be any value listed in the following table.

The speed of the error detection process for each data frame is determined by the following equation:

Figure 9-1: Error Detection Frequency Equation

$$\text{Error Detection Frequency} = \frac{\text{Internal Oscillator Frequency}}{2^n}$$

Table 9-2: Error Detection Frequency Range for Arria V Devices

The following table lists the frequencies and valid values of n.

Internal Oscillator Frequency	Error Detection Frequency		n	Divisor Range
	Maximum	Minimum		
100 MHz	100 MHz	390 kHz	0, 1, 2, 3, 4, 5, 6, 7, 8	1 – 256

CRC Calculation Time For Entire Device

While the CRC calculation is done on a per frame basis, it is important to know the time taken to complete CRC calculations for the entire device. The entire device detection time is the time taken to do CRC calculations on every frame in the device. This time depends on the device and the error detection clock frequency. The error detection clock frequency also depends on the device and on the internal oscillator frequency, which varies from 42.6 MHz to 100 MHz.

You can calculate the minimum and maximum time for any number of divisor based on the following formula:

Maximum time (n) = 2^(n-8) * t_{MAX}

Minimum time (n) = 2^n * t_{MIN}

where the range of n is from 0 to 8.

Table 9-3: Device EDCRC Detection Time in Arria V Devices

The following table lists the minimum and maximum time taken to calculate the CRC value:

- The minimum time is derived using the maximum clock frequency with a divisor of 0.
- The maximum time is derived using the minimum clock frequency with a divisor of 8.

Variant	Member Code	t_{MIN} (ms)	t_{MAX} (s)
Arria V GX	A1	13.74	8.80
	A3	13.74	8.80
	A5	21.42	13.71
	A7	21.42	13.71
	B1	30.45	19.49
	B3	30.45	19.49
	B5	40.70	26.05
	B7	40.70	26.05
Arria V GT	C3	13.74	8.80
	C7	21.42	13.71
	D3	30.45	19.49
	D7	40.70	26.05
Arria V GZ	E1	28.50	17.12
	E3	28.50	17.12
	E5	37.10	22.30
	E7	37.10	22.30
Arria V SX	B3	40.70	26.05
	B5	40.70	26.05
Arria V ST	D3	40.70	26.05
	D5	40.70	26.05

Using Error Detection Features in User Mode

This section describes the pin, registers, process flow, and procedures for error detection in user mode.

Enabling Error Detection

To enable user mode error detection and internal scrubbing in the Intel Quartus Prime software, follow these steps:

1. On the Assignments menu, click **Device**.
2. In the Device dialog box, click **Device and Pin Options**.
3. In the **Category** list, click **Error Detection CRC**.
4. Turn on **Enable Error Detection CRC_ERROR pin**.
5. To set the CRC_ERROR pin as output open drain, turn on **Enable open drain on CRC_ERROR pin**. Turning off this option sets the CRC_ERROR pin as output.
6. To enable the on-chip error correction feature, turn on **Enable internal scrubbing**.
7. In the **Divide error check frequency by** list, select a valid divisor.
8. Click OK.

CRC_ERROR Pin

Table 9-4: Pin Description

Pin Name	Pin Type	Description
CRC_ERROR	I/O or output/ output open-drain	An active-high signal, when driven high indicates that an error is detected in the CRAM bits. This pin is only used when you enable error detection in user mode. Otherwise, the pin is used as a user I/O pin. When using the WYSIWYG function, you can route the crcerror port from the WYSIWYG atom to the dedicated CRC_ERROR pin or any user I/O pin. To route the crcerror port to a user I/O pin, insert a D-type flipflop between them.

Error Detection Registers

This section describes the registers used in user mode.

Figure 9-2: Block Diagram for Error Detection in User Mode

The block diagram shows the registers and data flow in user mode.

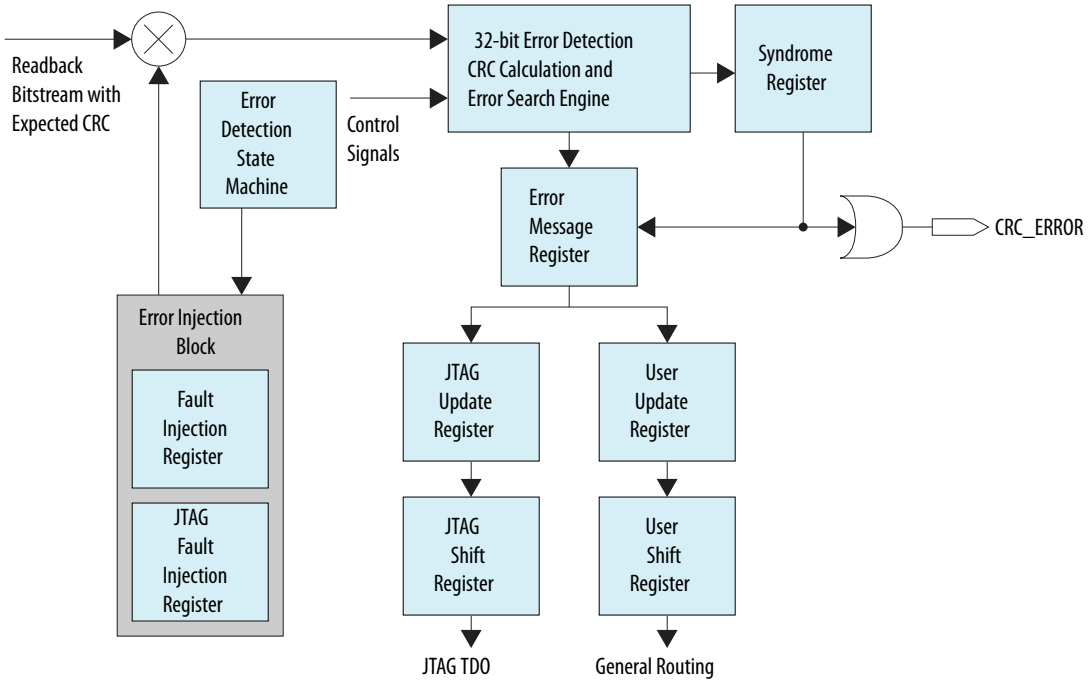


Table 9-5: Error Detection Registers

Name	Width (Bits)	Description
Syndrome register	32	Contains the 32-bit CRC signature calculated for the current frame. If the CRC value is 0, the CRC_ERROR pin is driven low to indicate no error. Otherwise, the pin is pulled high.
Error message register (EMR)	67	Contains error details for single-bit and double-adjacent errors. The error detection circuitry updates this register each time the circuitry detects an error. The Error Message Register Map figure shows the fields in this register and the Error Type in EMR table lists the possible error types.
JTAG update register	67	This register is automatically updated with the contents of the EMR one clock cycle after the content of this register is validated. The JTAG update register includes a clock enable, which must be asserted before its contents are written to the JTAG shift register. This requirement ensures that the JTAG update register is not overwritten when its contents are being read by the JTAG shift register.
JTAG shift register	67	This register allows you to access the contents of the JTAG update register via the JTAG interface using the SHIFT_EDERROR_REG JTAG instruction.

Name	Width (Bits)	Description
User update register	67	This register is automatically updated with the contents of the EMR one clock cycle after the contents of this register are validated. The user update register includes a clock enable, which must be asserted before its contents are written to the user shift register. This requirement ensures that the user update register is not overwritten when its contents are being read by the user shift register.
User shift register	67	This register allows user logic to access the contents of the user update register via the core interface.
JTAG fault injection register	46	You can use this register with the `EDERROR_INJECT JTAG` instruction to inject errors in the bitstream. The JTAG Fault Injection Register Map table lists the fields in this register.
Fault injection register	46	This register is updated with the contents of the JTAG fault injection register.

Figure 9-3: Error Message Register Map

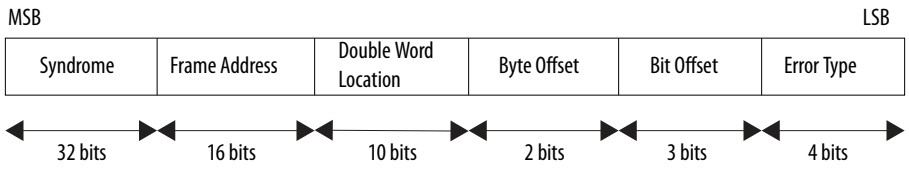


Table 9-6: Error Type in EMR

The following table lists the possible error types reported in the error type field in the EMR.

Error Type				Description
Bit 3	Bit 2	Bit 1	Bit 0	
0	0	0	0	No CRC error.
0	0	0	1	Location of a single-bit error is identified.
0	0	1	0	Location of a double-adjacent error is identified.
1	1	1	1	Error types other than single-bit and double-adjacent errors.

Table 9-7: JTAG Fault Injection Register Map

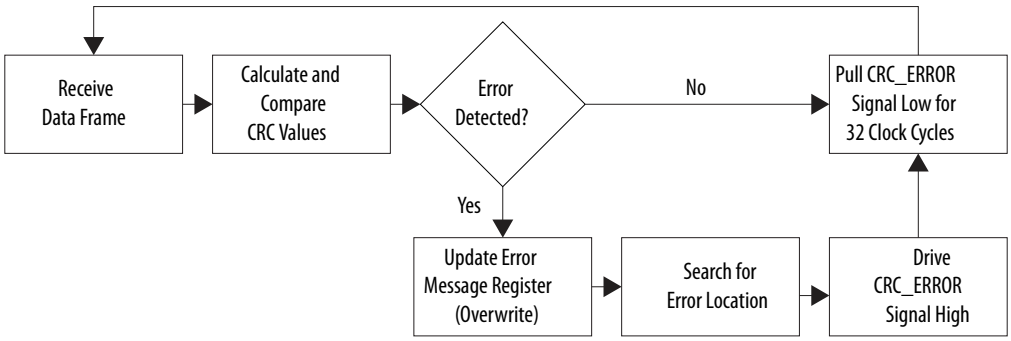
Field Name	Bit Range	Description
Error Byte Value	31:0	Contains the location of the bit error that corresponds to the error injection type to this field.
Byte Location	41:32	Contains the location of the injected error in the first data frame.

Field Name	Bit Range				Description
	45:42				Specifies the following error types.
	Bit 45	Bit 44	Bit 43	Bit 42	
Error Type	0	0	0	0	No error
	0	0	0	1	Single-bit error
	0	0	1	0	Double adjacent error

Error Detection Process

When enabled, the user mode error detection process activates automatically when the FPGA enters user mode. The process continues to run until the device is reset even when an error is detected in the current frame.

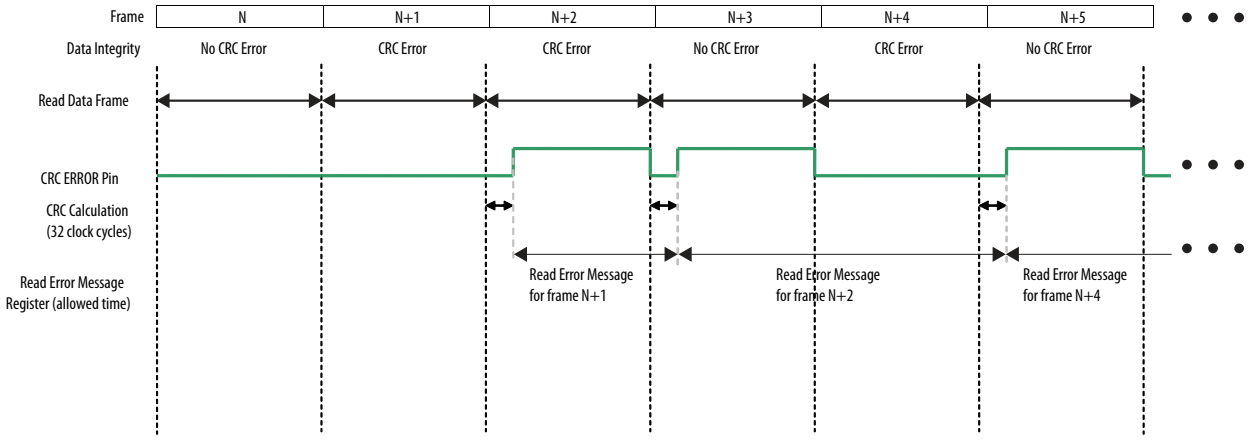
Figure 9-4: Error Detection Process Flow in User Mode



Timing

The CRC_ERROR pin is always driven low during CRC calculation. When an error occurs, the EDCRC hard block takes 32 clock cycles to update the EMR, the pin is driven high once the EMR is updated. Therefore, you can start retrieving the contents of the EMR at the rising edge of the CRC_ERROR pin. The pin stays high until the current frame is read and then driven low again for 32 clock cycles. To ensure information integrity, complete the read operation within one frame of the CRC verification. The following diagram shows the timing of these events.

Figure 9-5: Timing Requirements



Retrieving Error Information

You can retrieve the error information via the core interface or the JTAG interface using the `SHIFT_EDERROR_REG JTAG` instruction.

Recovering from CRC Errors

The system that hosts the FPGA must control device reconfiguration. To recover from a CRC error, drive the `nCONFIG` signal low. The system waits for a safe time before reconfiguring the device. When reconfiguration completes successfully, the FPGA operates as intended.

Related Information

- **Error Detection Frequency** on page 9-3
  Provides more information about the minimum and maximum error detection frequencies.
- **Minimum EMR Update Interval** on page 9-2
  Provides more information about the duration of each Arria Vdevice.
- **Test Methodology of Error Detection and Recovery using CRC in Altera FPGA Devices**
  Provides more information about how to retrieve the error information.

Testing the Error Detection Block

You can inject errors into the configuration data to test the error detection block. This error injection methodology provides design verification and system fault tolerance characterization.

Testing via the JTAG Interface

You can intentionally inject single or double-adjacent errors into the configuration data using the `EDERROR_INJECT` JTAG instruction.

Table 9-8: EDERROR_INJECT instruction

JTAG Instruction	Instruction Code	Description
EDERROR_INJECT	00 0001 0101	Use this instruction to inject errors into the configuration data. This instruction controls the JTAG fault injection register, which contains the error you want to inject into the bitstream.

You can only inject errors into the first frame of the configuration data. However, you can monitor the error information at any time. Altera recommends that you reconfigure the FPGA after the test completes.

Automating the Testing Process

You can automate the testing process by creating a Jam™ file (**.jam**). Using this file, you can verify the CRC functionality in-system and on-the-fly without reconfiguring the device. You can then switch to the CRC circuitry to check for real errors caused by an SEU.

Related Information

Test Methodology of Error Detection and Recovery using CRC in Altera FPGA Devices
Provides more information about how to test the error detection block.

SEU Mitigation for Arria V Devices Revision History

Date	Version	Changes
December 2016	2016.12.09	Updated specifications for Arria V GZ devices in Device EDCRC Detection Time in Arria V Devices table.
December 2015	2015.12.21	• Changed instances of Quartus II to Quartus Prime. • Updated the clock cycles for the CRC calculation in the Error Detection Process section.
January 2015	2015.01.23	Updated the description in the CRC Calculation Time section.
June 2014	2014.06.30	Updated the CRC Calculation Time section.
November 2013	2013.11.12	• Updated the CRC Calculation Time section to include a formula to calculate the minimum and maximum time. • Removed preliminary for the Minimum EMR Update Interval and CRC Calculation Time. • Removed related information for the Internal Scrubbing feature.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Moved all links to the Related Information section of respective topics for easy reference.

Date	Version	Changes
November 2012	2012.11.19	• Added the following specifications for Arria V GZ—Minimum EMR update interval, error detection frequency, and CRC calculation time. • Updated the width of the JTAG fault injection and fault injection registers. • Reorganized content and updated template.
June 2012	2.0	• Added the "Basic Description", "Error Detection Features", "Types of Error Detection", "Error Detection Components", "Using the Error Detection Feature", and "Testing the Error Detection Block" sections. • Updated Table 9–4, Table 9–5, and Table 9–6. • Restructured the chapter.
November 2011	1.1	Restructured chapter.
May 2011	1.0	Initial release.

JTAG Boundary-Scan Testing in Arria V Devices 10

2020.04.13

AV-52010 Subscribe Send Feedback

This chapter describes the boundary-scan test (BST) features in Arria V devices.

Related Information

- **JTAG Configuration** on page 8-33
  Provides more information about JTAG configuration.
- **Arria V Device Handbook: Known Issues**
  Lists the planned updates to the *Arria V Device Handbook* chapters.

BST Operation Control

Arria V GX, GT, SX, and ST devices support IEEE Std. 1149.1 BST. Arria V GZ devices support IEEE Std. 1149.1 and IEEE Std. 1149.6 BST. You can perform BST on Arria V devices before, after, and during configuration.

IDCODE

The `IDCODE` is unique for each Arria V device. Use this code to identify the devices in a JTAG chain.

Table 10-1: IDCODE Information for Arria V Devices

Variant	Member Code	IDCODE (32 Bits)			
		Version (4 Bits)	Part Number (16 Bits)	Manufacture Identity (11 Bits)	LSB (1 Bit)
Arria V GX	A1	0000	0010 1010 0001 0001	000 0110 1110	1
	A3	0000	0010 1010 0000 0001	000 0110 1110	1
	A5	0000	0010 1010 0001 0010	000 0110 1110	1
	A7	0000	0010 1010 0000 0010	000 0110 1110	1
	B1	0000	0010 1010 0001 0011	000 0110 1110	1
	B3	0000	0010 1010 0000 0011	000 0110 1110	1
	B5	0000	0010 1010 0001 0110	000 0110 1110	1
	B7	0000	0010 1010 0000 0110	000 0110 1110	1
Arria V GT	C3	0000	0010 1010 0000 0001	000 0110 1110	1
	C7	0000	0010 1010 0000 0010	000 0110 1110	1
	D3	0000	0010 1010 0000 0011	000 0110 1110	1
	D7	0000	0010 1010 0000 0110	000 0110 1110	1
Arria V GZ	E1	0000	0010 1001 0011 0001	000 0110 1110	1
	E3	0000	0010 1001 0111 0001	000 0110 1110	1
	E5	0000	0010 1001 0111 0111	000 0110 1110	1
	E7	0000	0010 1001 1111 0111	000 0110 1110	1

Variant	Member Code	IDCODE (32 Bits)			
		Version (4 Bits)	Part Number (16 Bits)	Manufacture Identity (11 Bits)	LSB (1 Bit)
Arria V SX	B3	0000	0010 1101 0001 0011	000 0110 1110	1
	B5	0000	0010 1101 0000 0011	000 0110 1110	1
Arria V ST	D3	0000	0010 1101 0001 0011	000 0110 1110	1
	D5	0000	0010 1101 0000 0011	000 0110 1110	1

Supported JTAG Instruction

Table 10-2: JTAG Instructions Supported by Arria V Devices

JTAG Instruction	Instruction Code	Description
SAMPLE/PRELOAD	00 0000 0101	• Allows you to capture and examine a snapshot of signals at the device pins during normal device operation and permits an initial data pattern to be an output at the device pins. • Use this instruction to preload the test data into the update registers before loading the EXTEST instruction. • Used by the Signal Tap II Embedded Logic Analyzer.
EXTEST	00 0000 1111	• Allows you to test the external circuit and board-level interconnects by forcing a test pattern at the output pins, and capturing the test results at the input pins. Forcing known logic high and low levels on output pins allows you to detect opens and shorts at the pins of any device in the scan chain. • The high-impedance state of EXTEST is overridden by bus hold and weak pull-up resistor features.

JTAG Instruction	Instruction Code	Description
BYPASS	11 1111 1111	Places the 1-bit bypass register between the TDI and TDO pins. During normal device operation, the 1-bit bypass register allows the BST data to pass synchronously through the selected devices to adjacent devices.
USERCODE	00 0000 0111	• Examines the user electronic signature (UES) within the devices along a JTAG chain. • Selects the 32-bit USERCODE register and places it between the TDI and TDO pins to allow serial shifting of USERCODE out of TDO. • The UES value is set to default value before configuration and is only user-defined after the device is configured.
IDCODE	00 0000 0110	• Identifies the devices in a JTAG chain. If you select IDCODE, the device identification register is loaded with the 32-bit vendor-defined identification code. • Selects the IDCODE register and places it between the TDI and TDO pins to allow serial shifting of IDCODE out of TDO. • IDCODE is the default instruction at power up and in the TAP RESET state. Without loading any instructions, you can go to the SHIFT_DR state and shift out the JTAG device ID.

JTAG Instruction	Instruction Code	Description
HIGHZ	00 0000 1011	• Sets all user I/O pins to an inactive drive state. • Places the 1-bit bypass register between the TDI and TDO pins. During normal operation, the 1-bit bypass register allows the BST data to pass synchronously through the selected devices to adjacent devices while tri-stating all I/O pins until a new JTAG instruction is executed. • If you are testing the device after configuration, the programmable weak pull-up resistor or the bus hold feature overrides the HIGHZ value at the pin.
CLAMP	00 0000 1010	• Places the 1-bit bypass register between the TDI and TDO pins. During normal operation, the 1-bit bypass register allows the BST data to pass synchronously through the selected devices to adjacent devices while holding the I/O pins to a state defined by the data in the boundary-scan register. • If you are testing the device after configuration, the programmable weak pull-up resistor or the bus hold feature overrides the CLAMP value at the pin. The CLAMP value is the value stored in the update register of the boundary-scan cell (BSC).
PULSE_NCONFIG	00 0000 0001	Emulates pulsing the nCONFIG pin low to trigger reconfiguration even though the physical pin is not affected.
CONFIG_IO	00 0000 1101	Allows I/O reconfiguration (after or during reconfigurations) through the JTAG ports using I/O configuration shift register (IOCSR) for JTAG testing. You can issue the CONFIG_IO instruction only after the nSTATUS pin goes high.

JTAG Instruction	Instruction Code	Description
LOCK	01 1111 0000	Put the device in JTAG secure mode. In this mode, only BYPASS, SAMPLE/PRELOAD, EXTEST, IDCODE, SHIFT_EDERROR_REG, and UNLOCK instructions are supported. This instruction can only be accessed through JTAG core access in user mode. It cannot be accessed through external JTAG pins in test or user mode.
UNLOCK	11 0011 0001	Release the device from the JTAG secure mode to enable access to all other JTAG instructions. This instruction can only be accessed through JTAG core access in user mode. It cannot be accessed through external JTAG pins in test or user mode.
KEY_CLR_VREG	00 0010 1001	Clears the volatile key.
KEY_VERIFY	00 0001 0011	Verifies the non-volatile key has been cleared.
EXTEST_PULSE (38)	00 1000 1111	Enables board-level connectivity checking between the transmitters and receivers that are AC coupled by generating three output transitions: • Driver drives data on the falling edge of TCK in the UPDATE_IR/DR state. • Driver drives inverted data on the falling edge of TCK after entering the RUN_TEST/IDLE state. • Driver drives data on the falling edge of TCK after leaving the RUN_TEST/IDLE state. The EXTEST_PULSE JTAG instruction is only supported in user mode for Arria V GZ devices.

(38) This instruction is only supported by Arria V GZ devices.

JTAG Instruction	Instruction Code	Description
EXTEST_TRAIN (38)	00 0100 1111	Behaves the same as the EXTEST_PULSE instruction except that the output continues to toggle on the TCK falling edge as long as the TAP controller is in the RUN_TEST/IDLE state. The EXTEST_TRAIN JTAG instruction is only supported in user mode for Arria V GZ devices.

Note: If the device is in a reset state and the nCONFIG or nSTATUS signal is low, the device IDCODE might not be read correctly. To read the device IDCODE correctly, you must issue the IDCODE JTAG instruction only when the nCONFIG and nSTATUS signals are high.

Note: If you use DC coupling on HSSI signals, execute the EXTEST instruction. If you use AC coupling on HSSI signals, execute the EXTEST_PULSE instruction. AC-coupled and DC-coupled HSSI are only supported in post-configuration mode.

Related Information

JTAG Secure Mode on page 8-45
Provides more information about PULSE_NCONFIG, CONFIG_IO, LOCK, and UNLOCK JTAG instructions.

JTAG Secure Mode

If you enable the tamper-protection bit, the Arria V device is in JTAG secure mode after power up. In the JTAG secure mode, the JTAG pins support only the BYPASS, SAMPLE/PRELOAD, EXTEST, IDCODE, SHIFT_EDERROR_REG, and UNLOCK instructions. Issue the UNLOCK JTAG instruction to enable support for other JTAG instructions.

JTAG Private Instruction

Caution: Never invoke the following instruction codes. These instructions can damage and render the device unusable:

- 1100010000
- 0011001001
- 0000101011 (39)
- 1100010111
- 1100010011 (40)
- 1010100001
- 0101011110
- 0000101010

(39) This JTAG private instruction is not applicable for Arria V GZ devices.
(40) This JTAG private instruction is only applicable for Arria V GZ devices.

I/O Voltage for JTAG Operation

The Arria V device operating in IEEE Std. 1149.1 BST mode uses four dedicated JTAG pins—TDI, TDO, TMS, and TCK. Arria V devices do not support the optional TRST pin.

The TCK pin has an internal weak pull-down resistor, while the TDI and TMS pins have internal weak pull-up resistors. The 3.3-, 3.0-, or 2.5-V V_{CCPD} supply of I/O bank 3A powers the TDO, TDI, TMS, and TCK pins. All user I/O pins are tri-stated during JTAG configuration.

The JTAG chain supports several different devices. Use the supported TDO and TDI voltage combinations listed in the following table if the JTAG chain contains devices that have different V_{CCIO} levels. The output voltage level of the TDO pin must meet the specification of the TDI pin it drives.

Note: Arria V GZ devices do not support 3.3-V V_{CCPD} supply.

Table 10-3: Supported TDO and TDI Voltage Combinations

The TDO output buffer for V_{CCPD} of 3.3 V or 3.0 V meets V_{OH} (MIN) of 2.4 V, and the TDO output buffer for V_{CCPD} of 2.5 V meets V_{OH} (MIN) of 2.0 V.

Device	TDI Input Buffer Power (V)	Arria V TDO V_{CCPD}		
		V_{CCPD} = 3.3 V	V_{CCPD} = 3.0 V	V_{CCPD} = 2.5 V
Arria V	V_{CCPD} = 3.3	Yes	Yes	Yes
	V_{CCPD} = 3.0	Yes	Yes	Yes
	V_{CCPD} = 2.5	Yes	Yes	Yes
Non- Arria V[41]	V_{CC} = 3.3	Yes	Yes	Yes
	V_{CC} = 2.5	Yes	Yes	Yes
	V_{CC} = 1.8	Yes	Yes	Yes
	V_{CC} = 1.5	Yes	Yes	Yes

Performing BST

You can issue BYPASS, IDCODE, and SAMPLE JTAG instructions before, after, or during configuration without having to interrupt configuration.

To issue other JTAG instructions, follow these guidelines:

- To perform testing before configuration, hold the nCONFIG pin low.
- To perform BST during configuration, issue CONFIG_IO JTAG instruction to interrupt configuration. While configuration is interrupted, you can issue other JTAG instructions to perform BST. After BST is completed, issue the PULSE_CONFIG JTAG instruction or pulse nCONFIG low to reconfigure the device.

[41] The input buffer must be tolerant to the TDO V_{CCPD} voltage.

The chip-wide reset (DEV_CLRn) and chip-wide output enable (DEV_OE) pins on Arria V devices do not affect JTAG boundary-scan or configuration operations. Toggling these pins does not disrupt BST operation (other than the expected BST behavior).

If you design a board for JTAG configuration of Arria V devices, consider the connections for the dedicated configuration pins.

Related Information

- **JTAG Configuration** on page 8-33
  Provides more information about JTAG configuration.
- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about pin connections.
- **Arria V GZ Device Family Pin Connection Guidelines**
  Provides more information about pin connections.
- **Arria V Device Datasheet**
  Provides more information about JTAG configuration timing.

Enabling and Disabling IEEE Std. 1149.1 BST Circuitry

The IEEE Std. 1149.1 BST circuitry is enabled after the Arria V device powers up. However for Arria V SoC FPGAs, you must power up both HPS and FPGA to perform BST.

The HPS should be held in reset while performing BST to stop the I/Os being accessed or setup by the HPS.

To ensure that you do not inadvertently enable the IEEE Std. 1149.1 circuitry when it is not required, disable the circuitry permanently with pin connections as listed in the following table.

Table 10-4: Pin Connections to Permanently Disable the IEEE Std. 1149.1 Circuitry for Arria V Devices

JTAG Pins[42]	Connection for Disabling
TMS	V_{CCPD} supply of Bank 3A
TCK	GND
TDI	V_{CCPD} supply of Bank 3A
TDO	Leave open

[42] The JTAG pins are dedicated. Software option is not available to disable JTAG in Arria V devices.

Guidelines for IEEE Std. 1149.1 Boundary-Scan Testing

Consider the following guidelines when you perform BST with IEEE Std. 1149.1 devices:

- If the "10..." pattern does not shift out of the instruction register through the TDO pin during the first clock cycle of the SHIFT_IR state, the TAP controller did not reach the proper state. To solve this problem, try one of the following procedures:
  - Verify that the TAP controller has reached the SHIFT_IR state correctly. To advance the TAP controller to the SHIFT_IR state, return to the RESET state and send the 01100 code to the TMS pin.
  - Check the connections to the VCC, GND, JTAG, and dedicated configuration pins on the device.
- Perform a SAMPLE/PRELOAD test cycle before the first EXTEST test cycle to ensure that known data is present at the device pins when you enter EXTEST mode. If the OEJ update register contains 0, the data in the OUTJ update register is driven out. The state must be known and correct to avoid contention with other devices in the system.
- Do not perform EXTEST testing during in-circuit reconfiguration because EXTEST is not supported during in-circuit reconfiguration. To perform testing, wait for the configuration to complete or issue the CONFIG_IO instruction to interrupt configuration.
- After configuration, you cannot test any pins in a differential pin pair. To perform BST after configuration, edit and redefine the BSC group that correspond to these differential pin pairs as an internal cell.

Related Information

IEEE 1149.1 BSDL Files

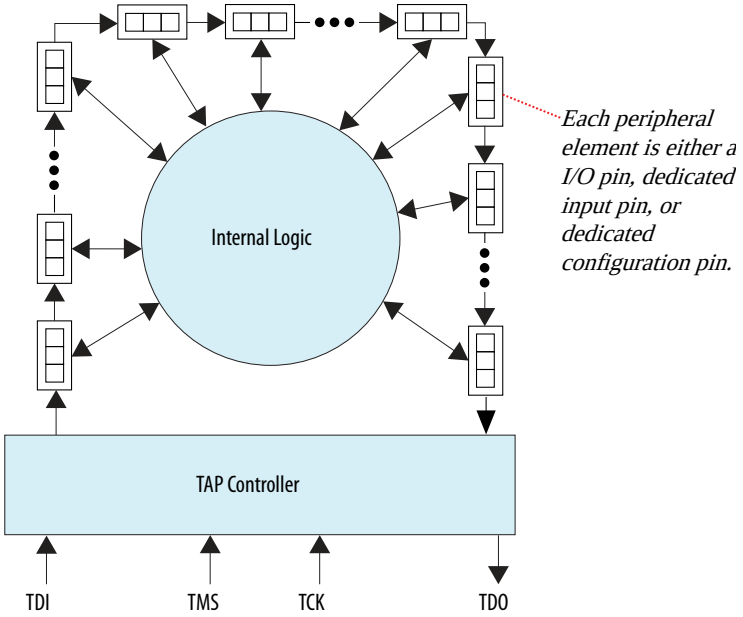
Provides more information about the BSC group definitions.

IEEE Std. 1149.1 Boundary-Scan Register

The boundary-scan register is a large serial shift register that uses the TDI pin as an input and the TDO pin as an output. The boundary-scan register consists of 3-bit peripheral elements that are associated with Arria V I/O pins. You can use the boundary-scan register to test external pin connections or to capture internal data.

Figure 10-1: Boundary-Scan Register

This figure shows how test data is serially shifted around the periphery of the IEEE Std. 1149.1 device.



Boundary-Scan Cells of an Arria V Device I/O Pin

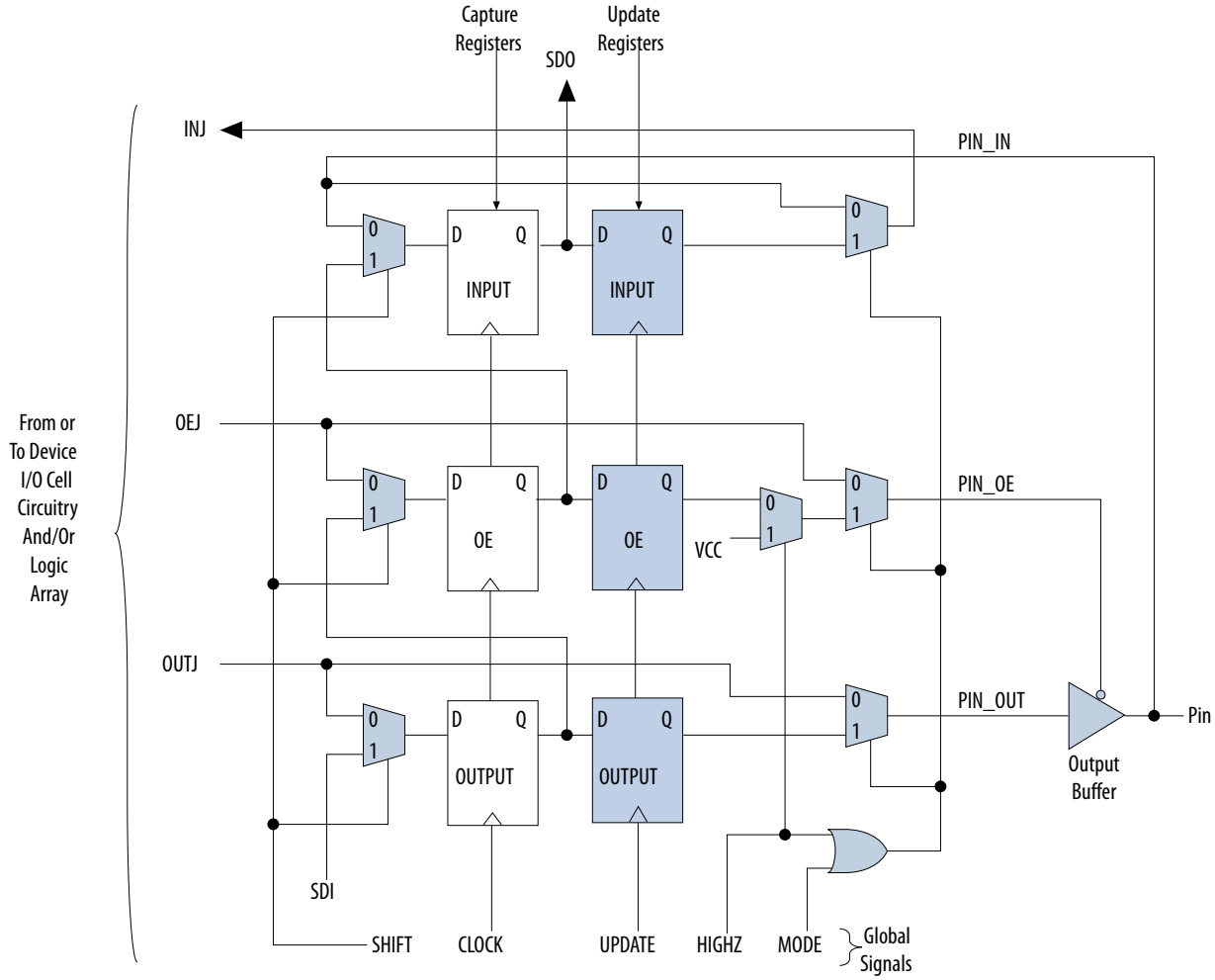
The Arria V device 3-bit BSC consists of the following registers:

- Capture registers—Connect to internal device data through the `OUTJ`, `OEJ`, and `PIN_IN` signals.
- Update registers—Connect to external data through the `PIN_OUT` and `PIN_OE` signals.

The TAP controller generates the global control signals for the IEEE Std. 1149.1 BST registers (`shift`, `clock`, and `update`) internally. A decode of the instruction register generates the `MODE` signal.

The data signal path for the boundary-scan register runs from the serial data in (`SDI`) signal to the serial data out (`SDO`) signal. The scan register begins at the `TDI` pin and ends at the `TDO` pin of the device.

Figure 10-2: User I/O BSC with IEEE Std. 1149.1 BST Circuitry for Arria V Devices



Note: `TDI`, `TDO`, `TMS`, and `TCK` pins, all `VCC` and `GND` pin types, and `VREF` pins do not have BSCs.

Table 10-5: Boundary-Scan Cell Descriptions for Arria V Devices

This table lists the capture and update register capabilities of all BSCs within Arria V devices.

Pin Type	Captures			Drives			Comments
	Output Capture Register	OE Capture Register	Input Capture Register	Output Update Register	OE Update Register	Input Update Register	
User I/O pins	`OUTJ`	`OEJ`	`PIN_IN`	`PIN_OUT`	`PIN_OE`	`INJ`	—
Dedicated clock input	0	1	`PIN_IN`	No Connect (N.C.)	N.C.	N.C.	`PIN_IN` drives to the clock network or logic array

Pin Type	Captures			Drives			Comments
	Output Capture Register	OE Capture Register	Input Capture Register	Output Update Register	OE Update Register	Input Update Register	
Dedicated input(43)	0	1	PIN_IN	N.C.	N.C.	N.C.	PIN_IN drives to the control logic
Dedicated bidirectional (open drain)(44)	0	OEJ	PIN_IN	N.C.	N.C.	N.C.	PIN_IN drives to the configuration control
Dedicated bidirectional(45)	OUTJ	OEJ	PIN_IN	N.C.	N.C.	N.C.	PIN_IN drives to the configuration control and OUTJ drives to the output buffer
Dedicated output(46)	OUTJ	0	0	N.C.	N.C.	N.C.	OUTJ drives to the output buffer

IEEE Std. 1149.6 Boundary-Scan Register

The BSCs for HSSI transmitters (GXB_TX[p,n]) and receivers/input clock buffers (GXB_RX[p,n])/(REFCLK[p,n]) in Arria V GZ devices are different from the BSCs for the I/O pins.

(43) This includes the nCONFIG, MSEL0, MSEL1, MSEL2, MSEL3, MSEL4, and nCE pins.
(44) This includes the CONF_DONE and nSTATUS pins.
(45) This includes the DCLK pin.
(46) This includes the nCEO pin.

Figure 10-3: HSSI Transmitter BSC with IEEE Std. 1149.6 BST Circuitry for Arria V GZ Devices

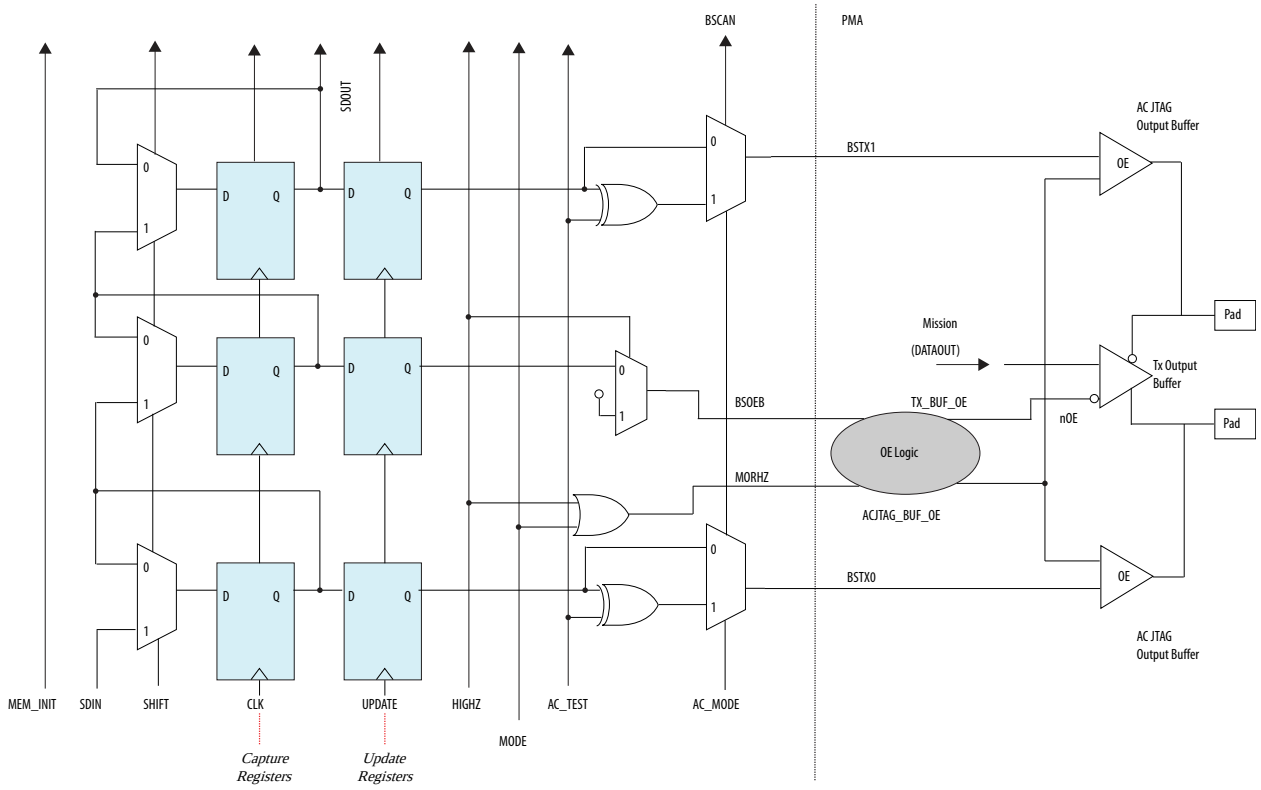
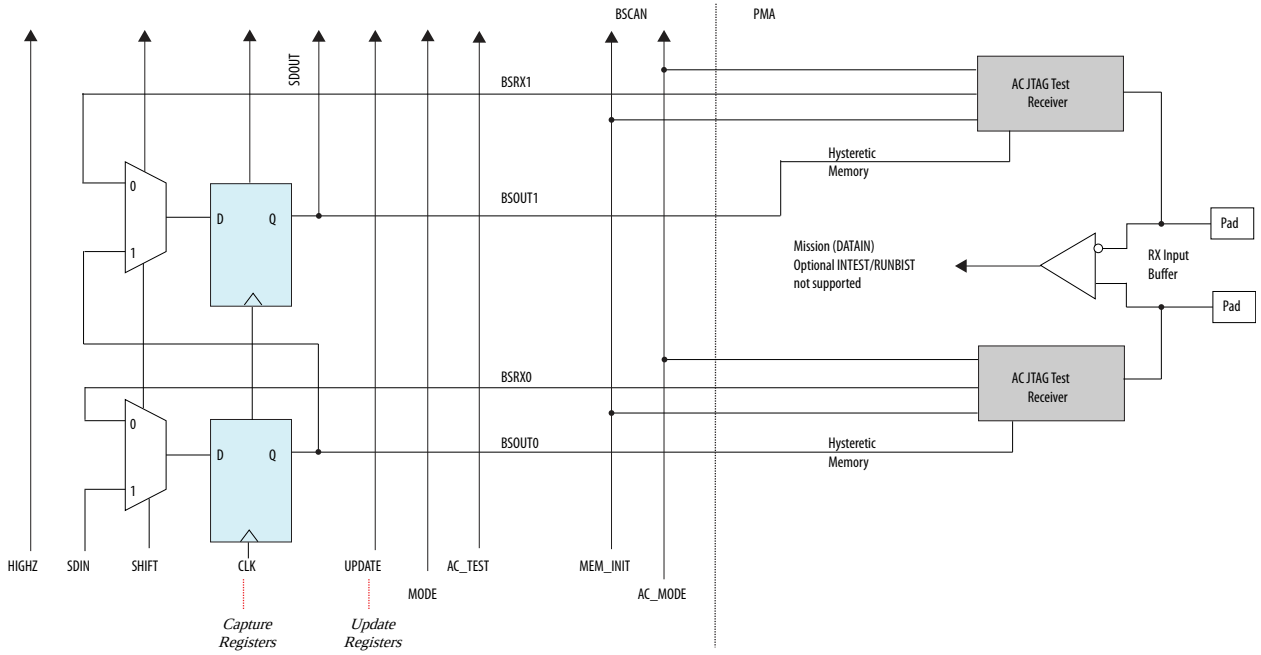


Figure 10-4: HSSI Receiver/Input Clock Buffer with IEEE Std. 1149.6 BST Circuitry for Arria V GZ Devices



JTAG Boundary-Scan Testing inArria V Devices Revision History

Date	Version	Changes
June 2016	2016.06.10	Added 0000101010 instruction code into JTAG Private Instruction section.
December 2015	2015.12.21	Changed instances of Quartus II to Quartus Prime.
June 2015	2015.06.12	Added a note in the Enabling and Disabling IEEE Std. 1149.1 BST Circuitry section.
June 2014	2014.06.30	Removed a note in the Performing BST section.
January 2014	2014.01.10	• Added a note to the Performing BST section. • Updated the Supported JTAG Instruction section. • Updated the KEY_CLR_VREG JTAG instruction.
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Updated the description for EXTEST_TRAIN and EXTEST_PULSE JTAG instructions. • Moved all links to the Related Information section of respective topics for easy reference.
November 2012	2012.11.19	• Added IDCODE for Arria V GZ devices. • Added EXTEST_PULSE and EXTEST_TRAIN JTAG instructions for Arria V GZ devices. • Added the IEEE Std. 1149.6 Boundary-Scan Register section for Arria V GZ devices. • Reorganized content and updated template.
June 2012	2.0	• Restructured the chapter. • Updated Table 10-1 and Table 10-2.
February 2012	1.2	Updated Table 10-2.
November 2011	1.1	Minor text edits.
May 2011	1.0	Initial release.

Power Management in Arria V Devices 11

2020.04.13

AV-52011 Subscribe Send Feedback

This chapter describes the hot-socketing feature, power-on reset (POR) requirements, power-up sequencing recommendation, temperature sensing diode (TSD), and their implementation in Arria V devices.

Related Information

- **Arria V Device Handbook: Known Issues**
  Lists the planned updates to the *Arria V Device Handbook* chapters.
- **PowerPlay Power Analysis**
  Provides more information about the Quartus®II PowerPlay Power Analyzer tool in volume 3 of the Quartus II Handbook.
- **Arria V Device Datasheet**
  Provides more information about the recommended operating conditions of each power supply.
- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more detailed information about power supply pin connection guidelines and power regulator sharing.
- **Arria V GZ Device Family Pin Connection Guidelines**
  Provides more detailed information about power supply pin connection guidelines and power regulator sharing.
- **Board Design Resource Center**
  Provides more detailed information about power supply design requirements.
- **Arria V and Cyclone V Design Guidelines**

Power Consumption

The total power consumption of an Arria V device consists of the following components:

- Static power—the power that the configured device consumes when powered up but no clocks are operating.
- Dynamic power— the additional power consumption of the device due to signal activity or toggling.

Dynamic Power Equation

Figure 11-1: Dynamic Power

The following equation shows how to calculate dynamic power where P is power, C is the load capacitance, and V is the supply voltage level.

$$P = \frac{1}{2} CV^2 \times frequency$$

The equation shows that power is design-dependent and is determined by the operating frequency of your design. Arria V devices minimize static and dynamic power using advanced process optimizations. This technology allows Arria V designs to meet specific performance requirements with the lowest possible power.

Programmable Power Technology

Arria V GZ devices offer the ability to configure portions of the core, called tiles, for high-speed or low-power mode of operation performed by the Intel Quartus Prime software without user intervention. Setting a tile to high-speed or low-power mode is accomplished with on-chip circuitry and does not require extra power supplies brought into the Arria V GZ device. In a design compilation, the Intel Quartus Prime software determines whether a tile should be in high-speed or low-power mode based on the timing constraints of the design.

Arria VGZ tiles consist of the following:

- Memory logic array block (MLAB)/ logic array block (LAB) pairs with routing to the pair
- MLAB/LAB pairs with routing to the pair and to adjacent digital signal processing (DSP)/ memory block routing
- TriMatrix memory blocks
- DSP blocks
- PCI Express® (PCIe®) hard IP
- Physical coding sublayer (PCS)

All blocks and routing associated with the tile share the same setting of either high-speed or low-power mode. By default, tiles that include DSP blocks or memory blocks are set to high-speed mode for optimum performance. Unused DSP blocks and memory blocks are set to low-power mode to minimize static power. Clock networks do not support programmable power technology.

With programmable power technology, faster speed grade FPGAs may require less power because there are fewer high-speed MLAB and LAB pairs, when compared with slower speed grade FPGAs. The slower speed grade device may have to use more high-speed MLAB and LAB pairs to meet performance requirements.

The Intel Quartus Prime software sets unused device resources in the design to low-power mode to reduce the static power. It also sets the following resources to low-power mode when they are not used in the design:

- LABs and MLABs
- TriMatrix memory blocks
- DSP blocks

If a phase-locked loop (PLL) is instantiated in the design, you may assert the `areset` pin high to keep the PLL in low-power mode.

Altera recommends that you power down unused PCIe HIPs, per side, by connecting the PCIe HIP power to GND on the PCB for additional power savings. All of the HIPs on a side of the device must be unused to be powered down. For additional information refer to the pin connection guidelines.

Table 11-1: Programmable Power Capabilities for Arria V GZ Devices

This table lists the available Arria V GZ programmable power capabilities. Speed grade considerations can add to the permutations to give you flexibility in designing your system.

Feature	Programmable Power Technology
LAB	Yes
Routing	Yes
Memory Blocks	Fixed setting(47)
DSP Blocks	Fixed setting(47)
Clock Networks	No

Related Information

- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides more information about powering down PCIe HIPs.
- **Arria V GZ Device Family Pin Connection Guidelines.**
  Provides more information about powering down PCIe HIPs.

Temperature Sensing Diode

The Arria V TSD uses the characteristics of a PN junction diode to determine die temperature. Knowing the junction temperature is crucial for thermal management. You can calculate junction temperature using ambient or case temperature, junction-to-ambient (ja) or junction-to-case (jc) thermal resistance, and device power consumption. Arria V devices monitor its die temperature with the internal TSD with built-in analog-to-digital converter (ADC) circuitry or the external TSD with an external temperature sensor. This allows you to control the air flow to the device.

All Arria V devices support internal TSD only except for Arria V GZ devices that support both internal and external TSDs.

(47) Tiles with DSP blocks and memory blocks that are used in the design are always set to high-speed mode. By default, unused DSP blocks and memory blocks are set to low-power mode.

Internal Temperature Sensing Diode

You can use the Arria V internal TSD in the following operations:

- Power-up mode—to read the die's temperature during configuration, enable the Altera Temperature Sensor IP core in your design.
- User mode—to read the die's temperature during user mode, assert the `clken` signal to the internal TSD circuitry.

Note: To reduce power consumption, disable the Arria V internal TSD when you are not using it.

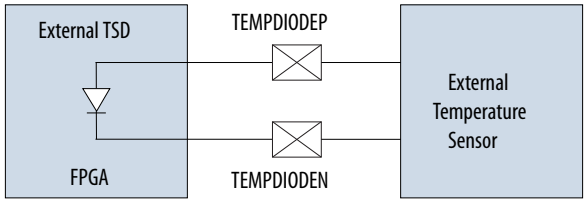
Related Information

- **Altera Temperature Sensor IP Core User Guide**
  Provides more information about using the Altera Temperature Sensor IP core.
- **Arria V Device Datasheet**
  Provides more information about the Arria V internal TSD specification.

External Temperature Sensing Diode

The Arria V GZ external TSD requires two pins for voltage reference. The following figure shows how to connect the external TSD with an external temperature sensor device, allowing external sensing of the Arria V GZ die temperature. For example, you can connect external temperature sensing devices, such as MAX1619, MAX1617A, MAX6627, and ADT7411 to the two external TSD pins for Arria V GZ device die temperature reading. The TSD diode is a substrate or common collector PNP diode type.

Figure 11-2: TSD External Pin Connections



The TSD is a very sensitive circuit that can be influenced by noise coupled from other traces on the board or within the device package itself, depending on your device usage. The interfacing signal from the Arria V GZ device to the external temperature sensor is based on millivolts (mV) of difference, as seen at the external TSD pins. Switching the I/O near the TSD pins can affect the temperature reading. Altera recommends taking temperature readings during periods of inactivity in the device or use the internal TSD with built-in ADC circuitry.

The following are board connection guidelines for the TSD external pin connections:

- The maximum trace lengths for the $TEMPDIODE_P$/$TEMPDIODE_N$ traces must be less than eight inches.
- Route both traces in parallel and place them close to each other with grounded guard tracks on each side.
- Altera recommends 10-mils width and space for both traces.
- Route traces through a minimum number of vias and crossunders to minimize the thermocouple effects.
- Ensure that the number of vias are the same on both traces.
- Ensure both traces are approximately the same length.
- Avoid coupling with toggling signals (for example, clocks and I/O) by having the GND plane between the diode traces and the high frequency signals.
- For high-frequency noise filtering, place an external capacitor (close to the external chip) between the $TEMPDIODE_P$/$TEMPDIODE_N$ trace. For Maxim devices, use an external capacitor between 2200 pF to 3300 pF.
- Place a 0.1 uF bypass capacitor close to the external device.
- You can use the internal TSD with built-in ADC circuitry and external TSD at the same time.
- If you only use internal ADC circuitry, the external TSD pins ($TEMPDIODE_P$/$TEMPDIODE_N$) can be connected to GND because the external TSD pins are not used.

For details about device specification and connection guidelines, refer to the external temperature sensor device datasheet from the device manufacturer.

Related Information

- **Arria V Device Datasheet**
  Provides more information about the external TSD specification.
- **Arria V GT and GX Device Family Pin Connection Guidelines**
  Provides details about the $TEMPDIODE_P$/$TEMPDIODE_N$ pin connection when you are not using an external TSD.
- **Arria V GZ Device Family Pin Connection Guidelines.**
  Provides details about the $TEMPDIODE_P$/$TEMPDIODE_N$ pin connection when you are not using an external TSD.

Hot-Socketing Feature

Arria V devices support hot socketing—also known as hot plug-in or hot swap.

The hot-socketing circuitry monitors the following power supplies and banks:

- Arria V GX, GT, SX, and ST devices—V_{CCIO}, V_{CCPD}, V_{CC}, and V_{CCP} power supplies and all V_{CCIO} and V_{CCPD} banks.
- Arria V GZ devices—V_{CCIO}, V_{CCPD}, and V_{CC} power supplies and all V_{CCIO} and V_{CCPD} banks.

When powering up or powering down these power supplies, refer to the Power-Up Sequence section of this handbook.

During the hot-socketing operation, the I/O pin capacitance is less than 15 pF and the clock pin capacitance is less than 20 pF.

The hot-socketing capability removes some of the difficulty that designers face when using the Arria V devices on PCBs that contain a mixture of devices with different voltage requirements.

The hot-socketing capability in Arria V devices provides the following advantages:

- You can drive signals into the I/O, dedicated input, and dedicated clock pins before or during power up or power down without damaging the device. External input signals to the I/O pins of the unpowered device will not power the power supplies through internal paths within the device.
- The output buffers are tri-stated during system power up or power down. Because the Arria V device does not drive signals out before or during power up, the device does not affect the other operating buses.
- You can insert or remove an Arria V device from a powered-up system board without damaging or interfering with the system board's operation. This capability allows you to avoid sinking current through the device signal pins to the device power supply, which can create a direct connection to GND that causes power supply failures.
- During hot socketing, Arria V devices are immune to latch up that can occur when a device is hot-socketed into an active system.

Altera uses GND as a reference for hot-socketing and I/O buffer circuitry designs. To ensure proper operation, connect GND between boards before connecting the power supplies. This prevents 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 over current condition in the Altera device.

Related Information

- **Arria V GX, GT, SX, and ST Power-Up Sequence** on page 11-7
- **Arria V GZ Power-Up Sequence** on page 11-9
- **Arria V Device Datasheet**
  Provides details about the Arria V hot-socketing specifications.

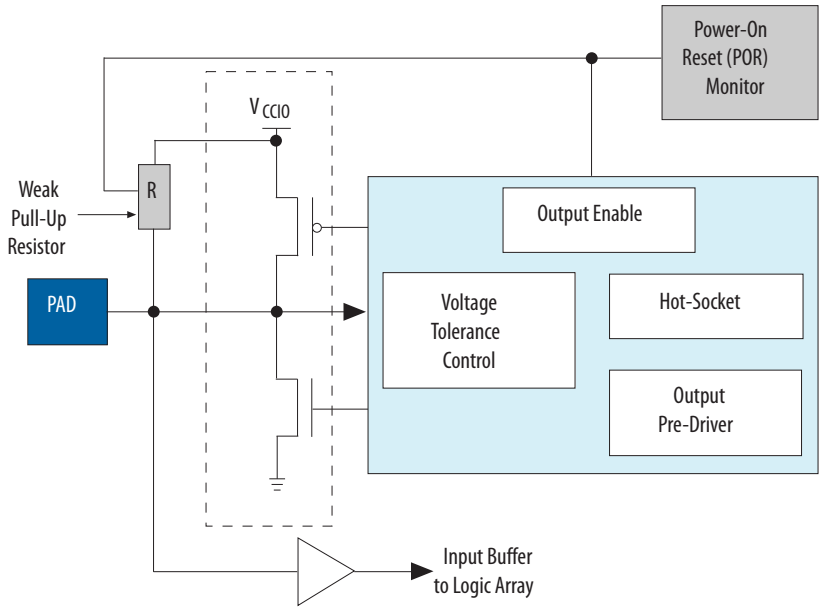
Hot-Socketing Implementation

The hot-socketing feature tri-state the output buffer during power up and power down of the power supplies. When these power supplies are below the threshold voltage, the hot-socketing circuitry generates an internal `HOTSCKT` signal.

Hot-socketing circuitry prevents excess I/O leakage during power up. When the voltage ramps up very slowly, I/O leakage is still relatively low, even after the release of the POR signal and configuration is complete.

Note: The output buffer cannot flip from the state set by the hot-socketing circuitry at very low voltage. To allow the `CONF_DONE` and `nSTATUS` pins to operate during configuration, the hot-socketing feature is not applied to these configuration pins. Therefore, these pins will drive out during power up and power down.

Figure 11-3: Hot-Socketing Circuitry for Arria V Devices



The POR circuitry monitors the voltage level of the power supplies and keeps the I/O pins tri-stated until the device is in user mode. The weak pull-up resistor (R) in the Arria V input/output element (IOE) is enabled during configuration download to keep the I/O pins from floating.

The 3.3-V tolerance control circuit allows the I/O pins to be driven by 3.3 V before the power supplies are powered and prevents the I/O pins from driving out before the device enters user mode.

Arria V GX, GT, SX, and ST Power-Up Sequence

Caution: To ensure minimum current consumption during power up, and to avoid functionality issue, follow the power-up sequence shown in the following figure. This power-up sequence is required for all Arria V GX and GT devices, except Arria V GX A5 and A7, and Arria V GT C7 devices. However, to ensure minimum current consumption during power up, Altera recommends that you also follow the power-up sequence for the Arria V GX A5 and A7, and Arria VGT C7 devices.

Note: If you plan to migrate your design from Arria V GX A5 and A7, and Arria V GT C7 devices to other Arria V devices, your design must adhere to the power-up sequence required for the other Arria V devices.

Figure 11-4: Power-Up Sequence Requirement for Arria V GX and GT Devices

Power up V_{CCBAT} at any time. Ramp up the power rails in each group to a minimum of 80% of their full rail before the next group starts. Power up V_{CCP}, V_{CCR_GXB}, V_{CCT_GXB}, and V_{CCL_GXB} together with V_{CC}.

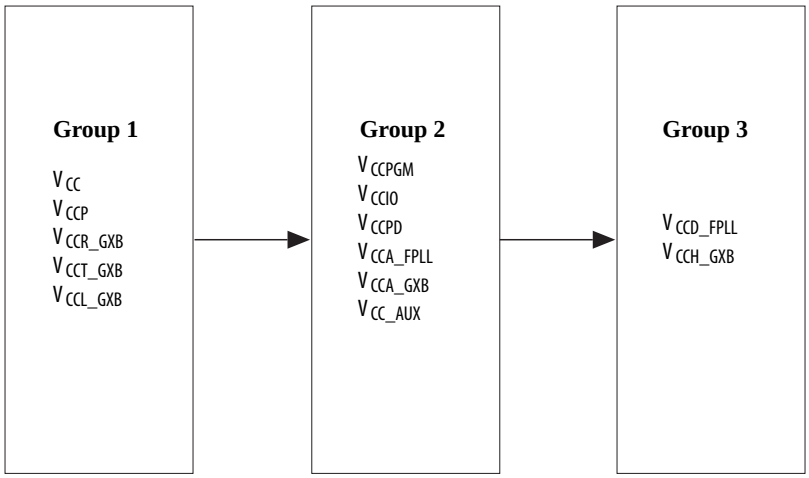


Figure 11-5: Power-Up Sequence Recommendation for Arria V SX and ST Devices

Power up the V_{ccbat} at any time. Ramp up the power rails in each group to a minimum of 80% of their recommended operating range before the next group starts.

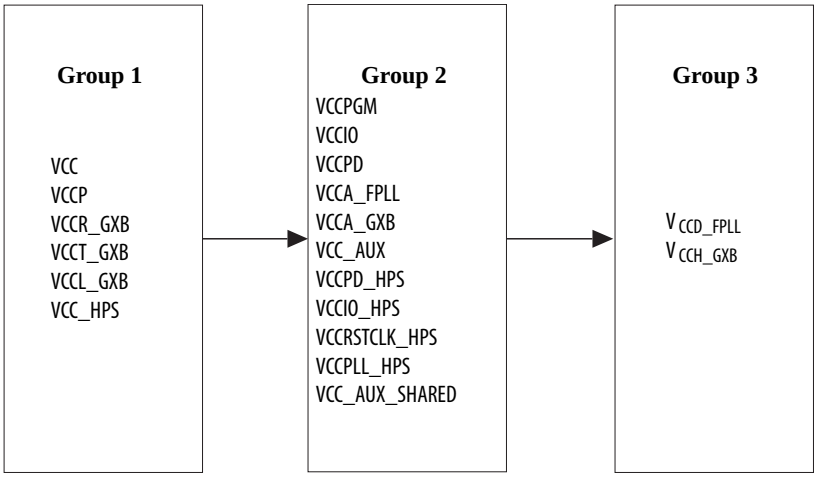


Table 11-2 lists the current transient that you may observe at the indicated power rails after powering up the Arria V device, and before configuration starts. These transients have a finite duration bounded by the time at which the device enters configuration mode. For Arria V SX and ST devices, you may observe the current transient in Table 11-2 after powering up the device, and before all the power supplies reach the recommended operating range.

Table 11-2: Maximum Power Supply Current Transient and Typical Duration

Power Rail	Maximum Power Supply Current Transient (mA)	Typical Duration (µs) [48]
V_{CCPD}[49], [50]	1250	50
V_{CCIO}[50], [51]	350	200
V_{CC_AUX}[52]	450	10
V_{CC}[52]	700	100
V_{CCPD_HPS}[53], [54], [55]	400	50
V_{CCIO_HPS}[53], [56], [55]	100	200
V_{CC_HPS}[52], [53]	420	100

For details about the minimum current requirements, refer to the Early Power Estimator (EPE), and compare to the information listed in Table 11-2. If the current transient exceeds the minimum current requirements in the EPE, you need to take the information into consideration for your power regulator design.

Related Information

PowerPlay Early Power Estimators (EPE) and Power Analyzer
Provides more information about the PowerPlay EPE support for Arria V devices.

Arria V GZ Power-Up Sequence

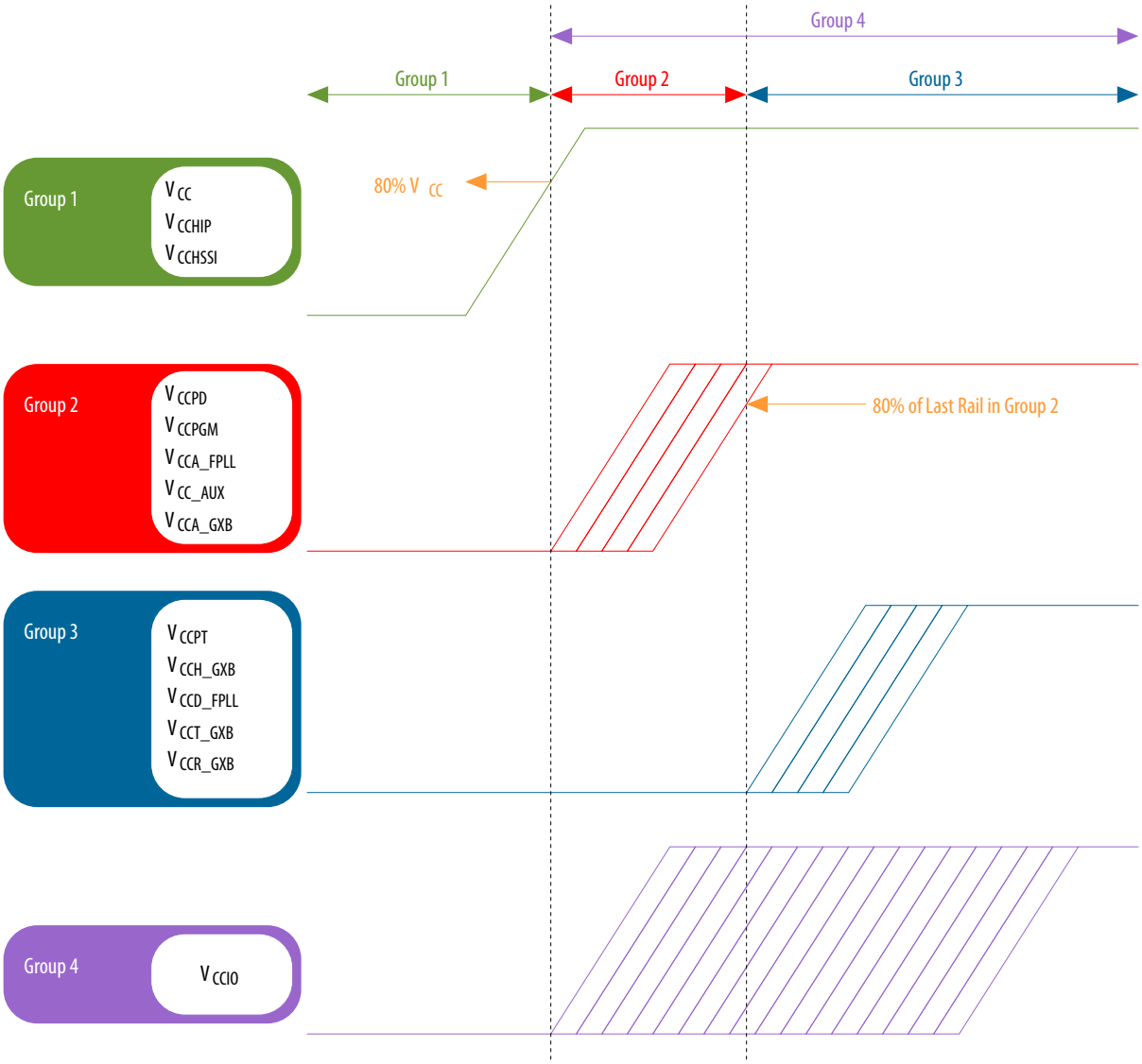
The Arria V GZ devices require a power-up sequence as shown in the following figure to prevent excessive inrush current and ensure proper transceiver functionality. This power-up sequence is divided into four power groups. Group 1 contains the first power rails to ramp. The V_{CC}, V_{CCHIP}, and V_{CCHSSI} power rails in this group must ramp to a minimum of 80% of their full rail before any other power rails may start. Group 1 power rails can continue to ramp to full rail. The power rails in Group 2 and Group 4 can start to ramp in any order after Group 1 has reached its minimum 80% threshold. When the last power rail in

[48] Only typical duration is provided as it may vary on the board design.
[49] You may observe the current transient at V_{CCPD} only when you do not follow the recommended power-up sequence. To avoid the current transient at V_{CCPD}, follow the recommended power-up sequence.
[50] The maximum current for V_{CCIO} and V_{CCPD} applies to all voltage levels supported by the Arria V device.
[51] You may observe the current transient at V_{CCIO} if you power up V_{CCIO} before V_{CCPD}. To avoid the current transient at V_{CCIO}, follow the recommended power-up sequence by powering up V_{CCIO} and V_{CCPD} together.
[52] You may observe the current transient at V_{CC_AUX}, V_{CC} and V_{CC_HPS} with any power-up sequence.
[53] These power rails are only available on Arria V SX and ST devices.
[54] You may observe the current transient at V_{CCPD_HPS} only when you do not follow the recommended power-up sequence. To avoid the current transient at V_{CCPD_HPS}, follow the recommended power-up sequence.
[55] The maximum current for V_{CCIO_HPS} and V_{CCPD_HPS} applies to all voltage levels supported by the Arria V device.
[56] You may observe the current transient at V_{CCIO_HPS} if you power up V_{CCIO_HPS} before V_{CCPD_HPS}. To avoid the current transient at V_{CCIO_HPS}, follow the recommended power-up sequence by powering up V_{CCIO_HPS} and V_{CCPD_HPS} together.

Group 2 reaches 80% of its full rail, the remaining power rails in Group 3 may start their ramp. During this time, Group 2 power rails may continue to ramp to full rail. Power rails in Group 3 may ramp in any order. All power rails must ramp monotonically. The complete power-up sequence must meet either the standard or fast POR delay time, depending on the POR delay setting that is used.

Figure 11-6: Power-Up Sequence Requirement for Arria V GZ Devices

Power up V_{CCBAT} at any time. If V_{CC}, V_{CCR_GXB}, and V_{CCT_GXB} have the same voltage level, they can be powered by the same regulator in Group 1 and ramp simultaneously.



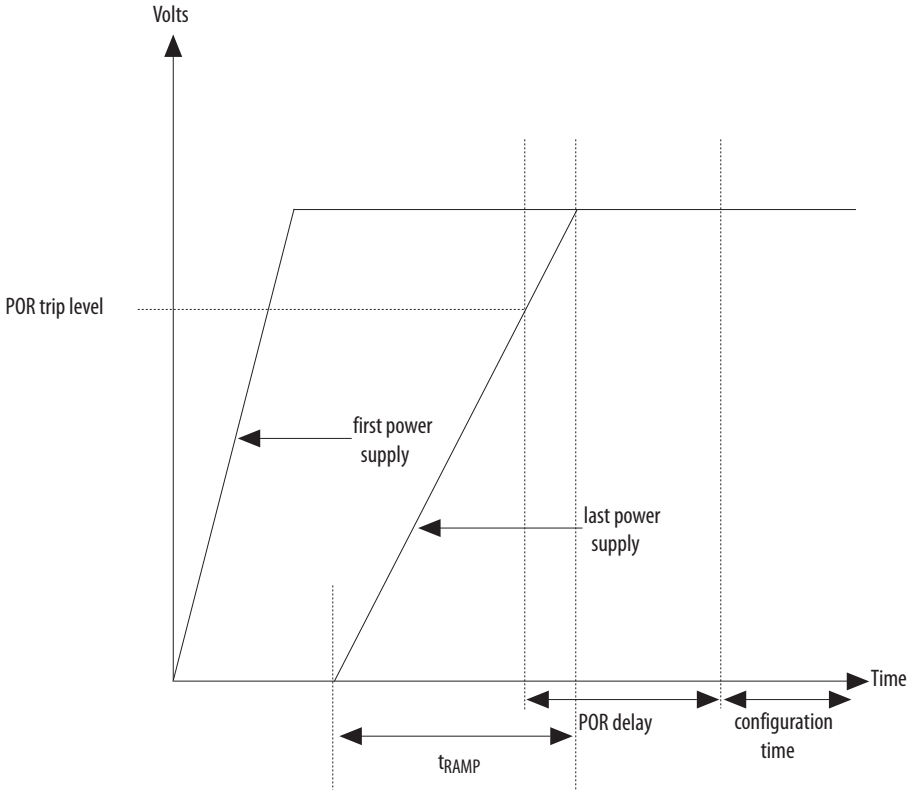
Arria V GZ devices may power down all power rails simultaneously. However, all rails must reach 0 V within 100 ms from the start of power-down.

Power-On Reset Circuitry

The POR circuitry keeps the Arria V device in the reset state until the power supply outputs are within the recommended operating range.

A POR event occurs when you power up the Arria V device until the power supplies reach the recommended operating range within the maximum power supply ramp time, t_{RAMP}. If t_{RAMP} is not met, the Arria V device I/O pins and programming registers remain tri-stated, during which device configuration could fail.

Figure 11-7: Relationship Between t_{RAMP} and POR Delay

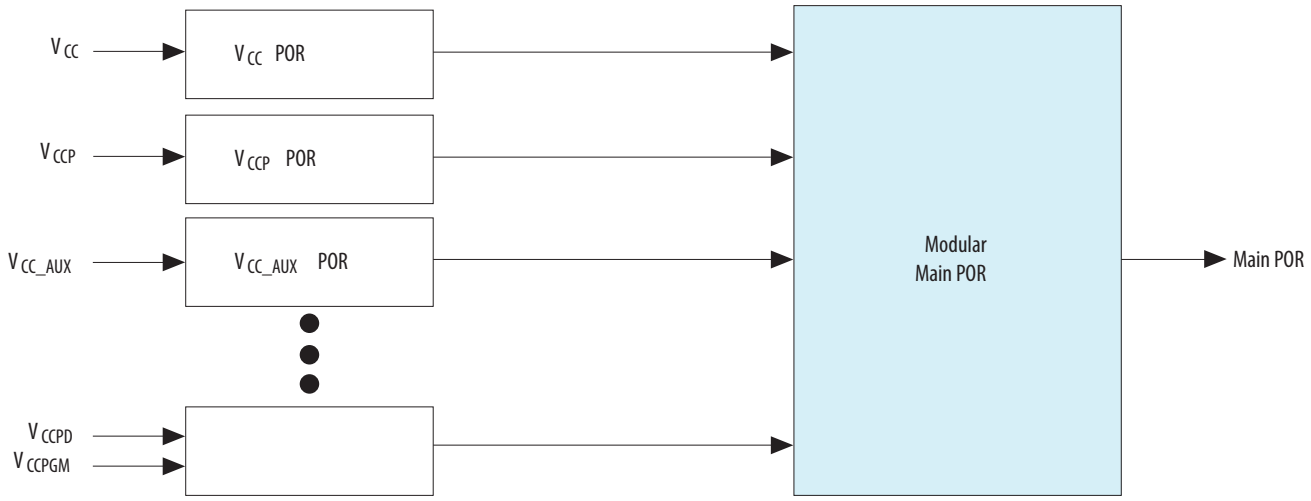


The Arria V POR circuitry uses an individual detecting circuitry to monitor each of the configuration-related power supplies independently. The main POR circuitry is gated by the outputs of all the individual detectors. The main POR signal is asserted when the power starts to ramp up. This signal is released after the last ramp-up power reaches the POR trip level during power up.

In user mode, the main POR signal is asserted when any of the monitored power goes below its POR trip level. Asserting the POR signal forces the device into the reset state.

The POR circuitry checks the functionality of the I/O level shifters powered by the V_{CCPD} and V_{CCPGM} power supplies during power-up mode. The main POR circuitry waits for all the individual POR circuitries to release the POR signal before allowing the control block to start programming the device.

Figure 11-8: Simplified POR Diagram for Arria V Devices



Related Information

Arria V Device Datasheet
Provides more information about the POR delay specification and t_{RAMP}.

Power Supplies Monitored and Not Monitored by the POR Circuitry

Table 11-3: Power Supplies Monitored and Not Monitored by the Arria V POR Circuitry

Devices	Power Supplies Monitored	Power Supplies Not Monitored
Arria V GX and GT	• V_{CC_AUX} • V_{CCBAT} • V_{CC} • V_{CCP} • V_{CCPD} • V_{CCPGM}	• V_{CCT_GXB} • V_{CCH_GXB} • V_{CCR_GXB} • V_{CCA_GXB} • V_{CCL_GXB} • V_{CCA_FPLL} • V_{CCD_FPLL} • V_{CCIO}
Arria V GZ	• V_{CC_AUX} • V_{CCBAT} • V_{CC} • V_{CCPT} • V_{CCPD} • V_{CCPGM}	• V_{CCT_GXB} • V_{CCH_GXB} • V_{CCR_GXB} • V_{CCA_GXB} • V_{CCA_FPLL} • V_{CCD_FPLL} • V_{CCIO} • V_{CCHIP}

Devices	Power Supplies Monitored	Power Supplies Not Monitored
Arria V SX and ST	• V_{CC_AUX} • V_{CCBAT} • V_{CC} • V_{CCP} • V_{CCPD} • V_{CCPGM} • V_{CC_HPS} • V_{CCPD_HPS} • $V_{CCRSTCLK\_HPS}$ • $V_{CC\_AUX\_SHARED}$	• V_{CCT_GXB} • V_{CCH_GXB} • V_{CCR_GXB} • V_{CCA_GXB} • V_{CCL_GXB} • V_{CCA_FPLL} • V_{CCD_FPLL} • V_{CCIO} • V_{CCIO_HPS} • V_{CCPLL_HPS}

Note: For the device to exit POR, you must power the V_{CCBAT} power supply even if you do not use the volatile key.

Related Information

MSEL Pin Settings
Provides information about the `MSEL` pin settings for each POR delay.

Power Management in Arria V Devices Revision History

Date	Version	Changes
December 2015	2015.12.21	• Changed instances of Quartus II to Quartus Prime. • Updated the External Temperature Sensing Diode section to provide the type of diode used. • Updated the Arria V GZ Power-Up Sequence section.
January 2015	2015.01.23	• Added $V_{CC\_AUX\_SHARED}$ to the power supplies monitored by the Arria V POR circuitry for Arria V SX and ST devices. • Added a link to the Arria V and Cyclone V Design Guidelines.
January 2014	2014.01.10	Updated the note to the V_{CCPD_HPS} power rail that current transient at V_{CCPD_HPS} is observed only when the recommended power-up sequence is not followed. To avoid the current transient at V_{CCPD_HPS}, follow the recommended power-up sequence.
June 2013	2013.06.28	• Added power-up sequences for Arria V SX and ST devices. • Added the current transient that occurs on HPS power rails during power-up

Date	Version	Changes
May 2013	2013.05.06	• Added link to the known document issues in the Knowledge Base. • Moved all links to the Related Information section of respective topics for easy reference. • Updated dynamic power in Power Consumption for improve clarity. • Added description on powering down unused PCIe HIPS in Programmable Power Technology • Updated Hot-Socketing Feature with 'When powering up these power supplies, refer to the Power-Up Sequence section of this handbook.' • Updated description about power-up sequence requirement for device migration to improve clarity. • Updated Figure 11-5 by renaming $V_{CCA\_GXB/GTB}$, $V_{CCT\_GXB/GTB}$, $V_{CCR\_GXB/GTB}$ to V_{CCA_GXB}, V_{CCT_GXB}, V_{CCR_GXB} and deleting V_{CCL_GTB}.
January 2013	2013.01.11	Updated the power-up sequence for Arria V GX and GT devices.
November 2012	2012.11.19	• Added the Programmable Power Technology section for Arria V GZ devices. • Added the External TSD section for Arria V GZ devices. • Added the Power-up sequence section for Arria V GZ devices. • Added the power supplies monitored and not monitored by the Arria V GZ devices. • Reorganized content and updated template.
June 2012	2.1	Updated the "Power-Up Sequence" section.
June 2012	2.0	• Restructured the chapter. • Added the "Power-Up Sequencing" section.
February 2012	1.3	Updated V_{CCP} description.
December 2011	1.2	• Added V_{CCP} information. • Updated Table 11-1.
November 2011	1.1	Restructured chapter.
May 2011	1.0	Initial release.