LatticeSC PURESPEED I/O Usage Guide

April 2013 Technical Note TN1088

Introduction

FPGAs are increasingly used as programmable SoCs in the middle of the system data path and therefore are expected to perform high-speed I/O translation and processing. As programmable ASSPs, they must comply with past, existing and emerging I/O standards.

Apart from established NPU, framer and module-based source synchronous I/O standards such as SPI4.2, SFI4.1, XGMII, RapidIO and CSIX, the next generation of clock-forwarded interfaces are being implemented on SRAM and DRAM memories (DDR1/2/3, RLDRAM1/2, FCRAM1/2 and QDR2) as well as ADCs and DACs. The I/O speeds required for these next-generation applications are expected to exceed 1Gbps. The LatticeSC™ PURESPEED™ I/O delivers best-in-class 2Gbps performance.

Although electrical compliance and high-speed signal integrity are required features, these alone do not address the bandwidth issue. The FPGA I/O must also have circuitry to manage and maintain the clock and data relationships of these high-speed signals as well as provide the necessary 'gearbox' functionality to slow data rates down and allow the FPGA fabric to perform the desired processing functions.

This technical note shows how a LatticeSC designer can implement legacy, contemporary and emerging parallel I/O standards reaching bandwidths up to 2Gbps covering the I/O buffers, termination schemes and I/O logic, as well as utilizing on-chip clocking resources for I/O-based clocking management. Please refer to the LatticeSC/M Family Data Sheet for more information about PURESPEED I/O.

Features

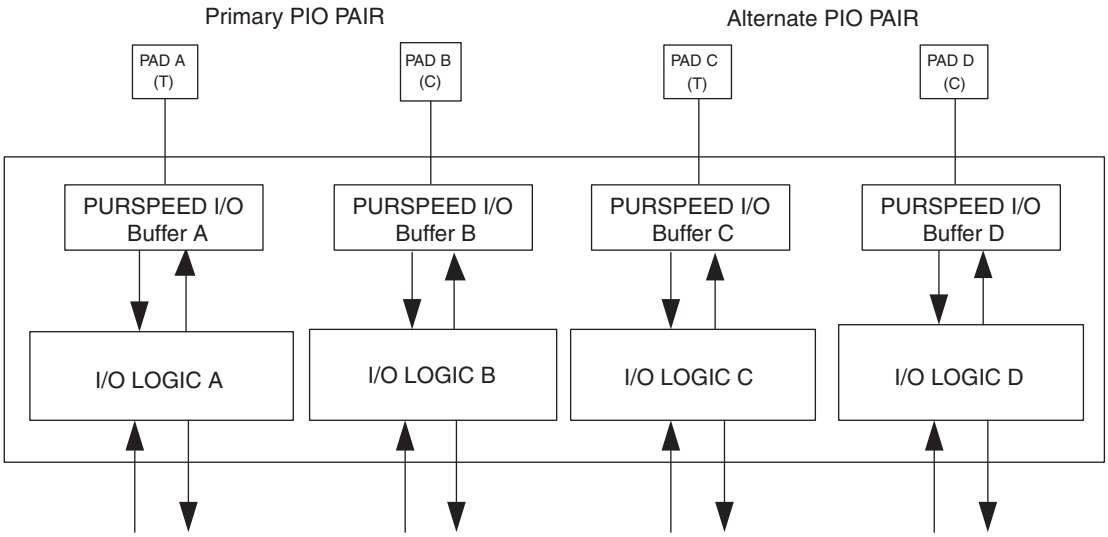
Some of the major PURESPEED I/O features supported in the LatticeSC device family are:

- Support for multiple single-ended I/O standards including LVTTL, LVCMOS33, LVCMOS25, LVCMOS18, LVCMOS12, HSTL15/18 Class I/II/III/IV, SSTL18/25/33 Class I/II, GTL, GTL+ inputs, PCI, PCIX
- Support for multiple differential I/O standards including LVDS, Mini-LVDS, Bus-LVDS, RSDS, MLVDS and LVPECL
- Programmable input and output termination
- Generic DDR up to 2Gbps using differential buffers and up to 450MHz using single-ended standards
- Generic SDR up to 1Gbps using differential buffers
- DDR, DDR2 and QDR memory interfaces that support 300MHz, 600Mbps data flow
- RLDRAM and FCRAM memory interfaces that support 400MHz, 800Mbps data flow
- Shift register provides gearbox between high-speed I/Os and FPGA logic
- Programmable adaptive input logic dynamically aligns data for robust high performance
- Programmable tap delay on every input

Functional Description

I/O Block Diagram

Figure 1. PIO Block Diagram



The LatticeSC PURESPEED I/O interface is built up from multiple Programmable I/O Cell (PIC) blocks. Each PIC contains four PIO (Programmable I/O).

Each PIC consists of a primary pair (PIOA and PIOB) and an alternate pair (PIOC and PIOD) of buffers. The primary pairs have additional capability that is not available on the alternate pairs.

When implementing single-ended signals, all the PURESPEED I/O buffers of the PIC are equivalent.

Only the primary PIO pairs on the left and right sides of the device have differential and complementary output driver capabilities. The different output drivers on the primary pair support differential PURESPEED I/O standards such as LVDS (Low Voltage Differential Signaling) and RSDS (Reduced Signal Differential Signaling). Both the primary and alternate PIO pairs will support differential inputs.

I/O Buffer Block Diagram

Figure 2. PURESPEED I/O Input Buffer

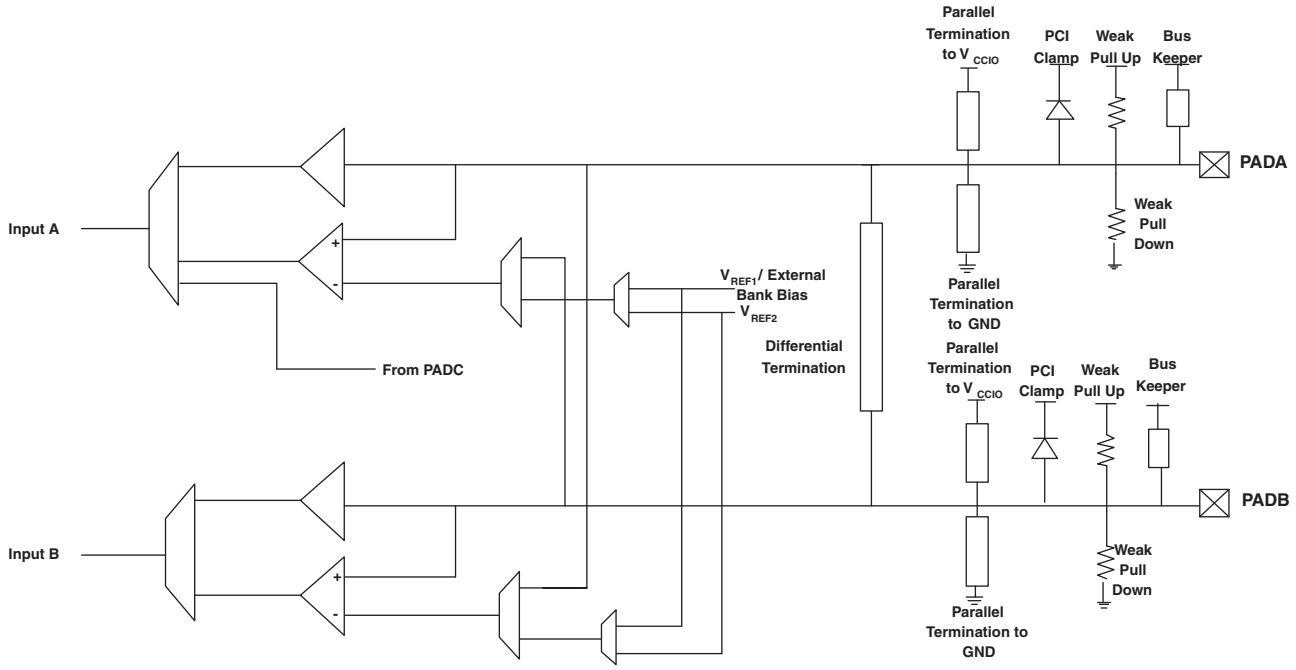
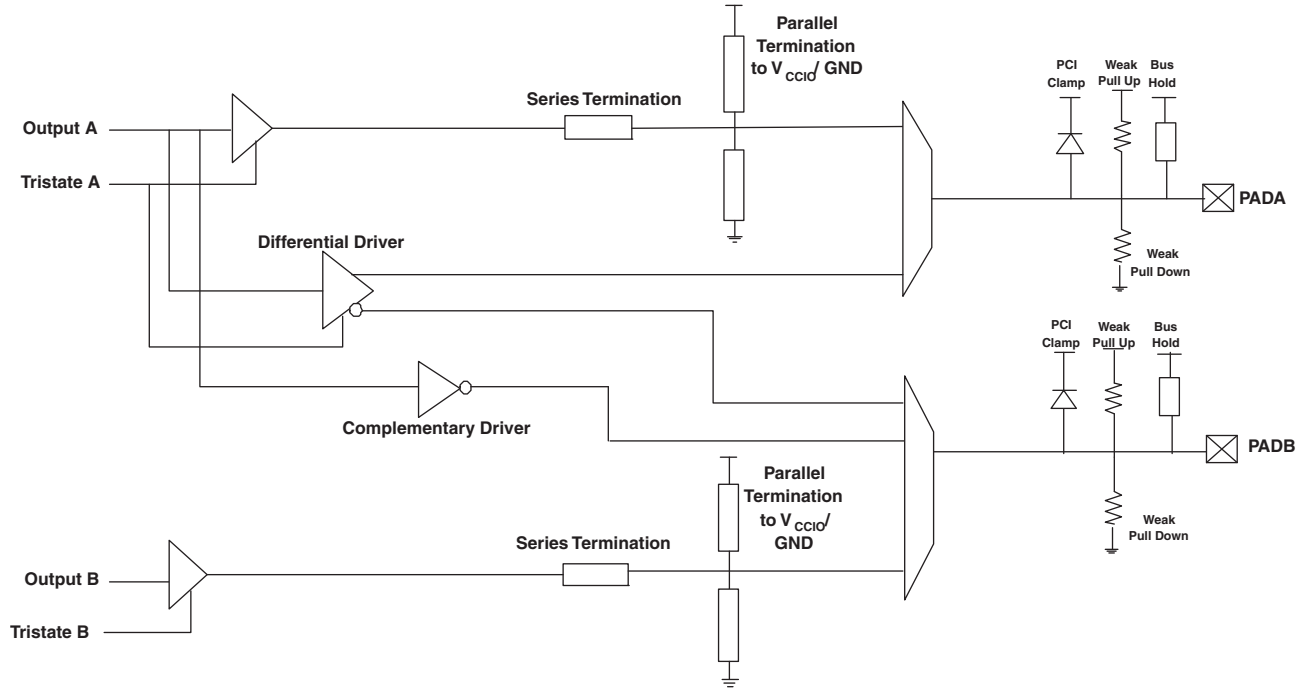


Figure 3. PURESPEED I/O Output Buffer



Figures 2 and 3 show the PURESPEED I/O input and output buffers for primary PIOA and PIOB. Each PIO has an input, output and tristate buffers connected to the I/O pad. The primary pads A and B on the left and right side banks can be combined to generate differential output drivers. The primary pads also have an extra inverter that allows complementary data to be sent to a pair of single-ended output drivers. This gives complementary output

capability from the primary pairs with minimum skew between the two drivers. The PURESPEED I/O buffers on the top and bottom banks support all V_{CCIO} levels up to 3.45V. The PURESPEED I/O buffers on the left and right sides support up to 2.625V V_{CCIO}.

Each I/O pad has programmable bus maintenance features that include weak pull-up, weak pull-down and buskeeper. I/O pads on the top and bottom sides also have a programmable PCI clamp.

Both the single-ended driver and the differential driver have programmable drive strength. Each of the single-ended and complementary output drivers also has a programmable series termination.

All of the input and output I/O pads have a dedicated programmable parallel termination to V_{CCIO}, GND or Thevenin termination to $V_{CCIO}/2$ or to V_{TT}. All the differential input pads can also be differentially terminated on all sides of the device. In addition to differential termination, common mode differential termination is available for differential signals using the VCMT node in the bank.

I/O Standards

Each PURESPEED I/O buffer can be programmable to support single-ended and differential I/O standards. Some of the single-ended standards are LVCMOS, LVTTL, SSTL, HSTL, PCI, GTL, GTLP, AGPIX and AGP2X. The LVCMOS and LVTTL PURESPEED I/O buffers have individual configurable options for drive strength, bus maintenance (weak pull-up, weak pull-down, or a bus-keeper latch) and open drain.

Some of the differential standards supported include LVDS, RSDS, BLVDS, MLVDS, LVPECL, mini-LVDS, differential SSTL and differential HSTL. Each LVDS, RSDS and mini-LVDS differential buffer has optional programmable drive strength. The tables below list the PURESPEED I/O input and output standards supported in the LatticeSC devices.

Table 1. PURESPEED I/O Standards

Standard	V_{CCIO}			V_{REF} (V)		
	Min.	Typ.	Max.	Min.	Typ.	Max.
LVCMOS 33	3.0	3.3	3.45	—	—	—
LVCMOS 25	2.375	2.5	2.625	—	—	—
LVCMOS 18	1.71	1.8	1.89	—	—	—
LVCMOS 15	1.425	1.5	1.575	—	—	—
LVCMOS 12	1.14	1.2	1.26	—	—	—
LVTTL	3.0	3.3	3.45	—	—	—
PCI33	3.0	3.3	3.3	$0.39V_{CCIO}$	$0.4V_{CCIO}$	$0.41V_{CCIO}$
PCIX33	3.0	3.3	3.3	$0.415V_{CCIO}$	$0.425V_{CCIO}$	$0.435V_{CCIO}$
PCIX15	1.425	1.5	1.575	$0.49V_{CCIO}$	$0.5V_{CCIO}$	$0.51V_{CCIO}$
AGP1X33	3.0	3.3	3.3	$0.39V_{CCIO}$	$0.4V_{CCIO}$	$0.41V_{CCIO}$
AGP2X33	3.0	3.3	3.3	$0.39V_{CCIO}$	$0.4V_{CCIO}$	$0.41V_{CCIO}$
SSTL18_I, II[3]	1.71	1.8	1.89	0.833	0.9	0.969
SSTL25_I, II[3]	2.375	2.5	2.625	1.15	1.25	1.35
SSTL33_I, II[3]	3.135	3.3	3.465	1.3	1.5	1.7
HSTL15_I, II[3]	1.425	1.5	1.575	0.68	0.75	0.9
HSTL15_III[1] and IV[1, 3]	1.425	1.5	1.575	0.68	0.9	0.9
HSTL 18_I[3], II[3]	1.71	1.8	1.89	0.816	0.9	1.08
HSTL 18_ III[1, 3], IV[1, 3]	1.71	1.8	1.89	0.816	0.9	1.08
GTL[1], GTL+[1]	—	—	—	0.882	1.0	1.122
LVDS	—	—	—	—	—	—
Mini-LVDS	—	—	—	—	—	—
RSDS	—	—	—	—	—	—

Table 1. PURESPEED I/O Standards (Continued)

Standard	V_{CCIO}			V_{REF} (V)		
	Min.	Typ.	Max.	Min.	Typ.	Max.
LVPECL33[4, 5]	3.135	3.3	3.465	—	—	—
BLVDS25[2, 3]	2.375	2.5	2.625	—	—	—
MLVDS25[2, 3]	2.375	2.5	2.625	—	—	—
SSTL18D_I[3], II[3]	1.71	1.8	1.89	—	—	—
SSTL25D_I[3], II[3]	2.375	2.5	2.625	—	—	—
SSTL33D_I[3], II[3]	3.135	3.3	3.465	—	—	—
HSTL15D_I[3], II[3]	1.425	1.5	1.575	—	—	—
HSTL18D_I[3], II[3]	1.71	1.8	1.89	—	—	—

1. Input only. Outputs can be supported by ganging multiple output buffers together.
2. Inputs on chip. Outputs are implemented with the addition of external resistors.
3. Input for this standard does not depend on the value of V_{CCIO}.
4. LVPECL33 inputs can only be used in banks with VCCIO ≤ 2.5V.
5. Outputs are implemented with the addition of external resistors and must be used in banks with VCCIO 3.3.

Programmable Terminations

Table 2 lists the various on-chip programmable terminations available at the transmitting and receiving end of the device. The termination values can be programmed as preferences in the Lattice ispLEVER® software.

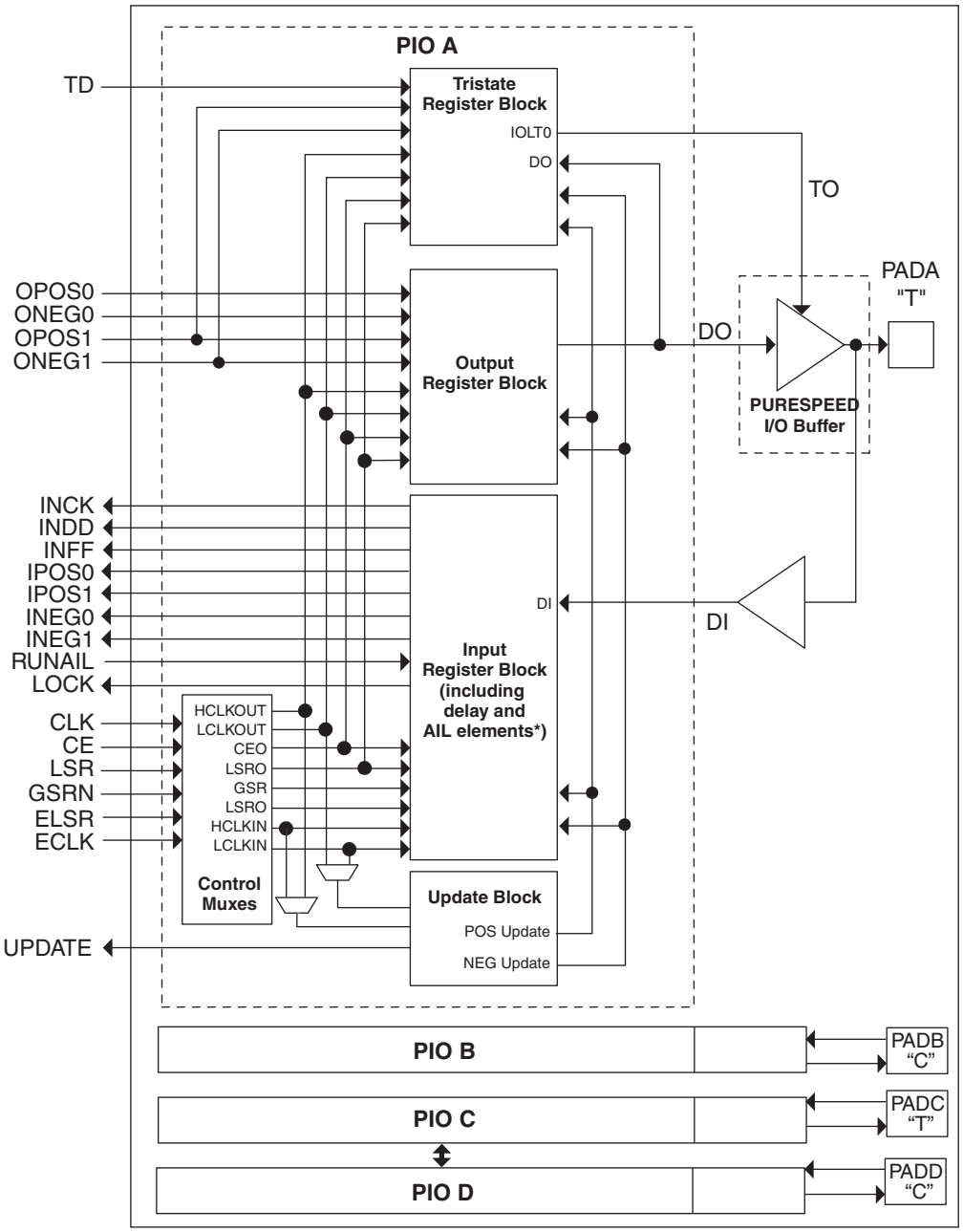
Table 2. Programmable On-chip Terminations

Termination Type	Driver/ Receiver End	Termination Type	Termination Value (Ohms)
Single-Ended Termination	Single-Ended Output Termination	Series Termination	25, 33, 50, 100
		Parallel Termination to V_{CCIO}	50, 100, 120
		Parallel termination to GND	50, 100, 120
		Parallel Termination to $V_{CCIO}/2$	25+ 60 to $V_{CCIO}/2$,
		Series Termination + Parallel Termination to $V_{CCIO}/2$	60 to $V_{CCIO}/2$, 50 to $V_{CCIO}/2$
	Single-Ended Input Terminations	Parallel Termination to V_{CCIO}	50, 100, 120
		Parallel termination to GND	50, 100, 120
		Parallel Termination to $V_{CCIO}/2$	60 to $V_{CCIO}/2$, 50 to $V_{CCIO}/2$
		Parallel Termination to V_{TT}	60, 75, 120, 150, 210
Differential Termination	Differential Input Termination	Differential Termination	120, 150, 220, 420
		Common Mode Differential Termination	120, 150, 220, 420

I/O Logic Block Diagram

Each I/O buffer is connected to an I/O logic block. The I/O logic block contains the input, output and tristate register blocks. Figure 4 shows the I/O logic block diagram.

Figure 4. I/O Logic Block Diagram



*AIL only on A or C pads located on the left, right and bottom of the device.

I/O Logic Modes

Input Register Block

Inputs coming into this block can optionally be delayed for either arbitrary delay values or to meet zero hold time requirements. See the Input Delay section of this document for details. The various input modes can be implemented in the software using IPexpress™. The I/O Logic section describes the software usage in detail.

The input register block can function in the following modes.

- Input data can bypass the input register block completely.
- SDR Register/Latch – In this mode the input data can be registered/latched using the primary clock.
- Fast Capture Latch – This mode uses the edge clock for early capture to allow zero hold but uses the primary clock for clocking the data so that transfer to the FPGA core can be on the same clock.
- DDR/Shift Register – The user can choose to operate in following modes:
  - DDR – The input serial DDR data is captured at the rising and falling edges of the high-speed edge clock.
  - DDR + Half Clock Transfer – In this mode, there is an additional half clock transfer so that all the data are presented to the FPGA core on the same clock edge.
  - DDR + Shift – The serial DDR data is captured using the high-speed edge clock, then transferred to the FPGA primary or secondary clock before the parallel data enters the FPGA core. This mode supports gearing ratios of 1x, 2x and 4x to mux/demux the I/O data rate (edge clock) to the FPGA clock rate. Note that the 1x shift simply provides a clock phase transfer to the internal FPGA clock.
  - Shift – The serial SDR data is captured using the high-speed edge clock, then transferred to the FPGA primary or secondary clock before the parallel data enters the FPGA core. This mode also supports gearing ratios of 1x, 2x and 4x to mux/demux the I/O data rate (edge clock) to the FPGA clock rate. Note that the 1x shift simply provides a clock phase transfer to the internal FPGA primary or secondary clock.

Output Register Block

The various output modes can be implemented in the software using IPexpress. The I/O Logic section of this document describes the software usage in detail. The output register block can function in the following modes.

- Output data can bypass the output registers completely
- SDR Register – In this mode the output data is registered before it is passed over to the PURESPEED I/O buffer.
- DDR/Shift Register – The I/O logic block supports the following DDR and shift register functions:
  - DDR – Outputs DDR data on both edges of the clock.
  - DDR + Shift – Receives slow-speed data from the FPGA primary or secondary clock and outputs high-speed DDR data on both sides of the high-speed edge clock. This mode supports gearing ratios of 1x, 2x and 4x to demux/mux the output data from FPGA primary or secondary clock to the edge clock. Note that the 1x shift simply provides a clock phase transfer to the edge clock.
  - Shift – Receives slow-speed data from the FPGA core primary or secondary clock and outputs high-speed SDR data clocked to the edge clock. This mode supports gearing ratios of 1x, 2x and 4x to demux/mux the output data from FPGA primary or secondary clock to the edge clock. Note that the 1x shift simply provides a clock phase transfer to the edge clock.

Tristate Register Block

The tristate register block can function in the following modes.

- Tristate data can bypass the tristate registers completely.
- SDR Register – The tristate signal is registered before it enters the output register block.
- DDR/Shift Register – The tristate signal can also be registered using the DDR/shift registers before it enters the output register block. The DDR/shift registers in this case only support a 1x gearing ratio.
- Open Drain Mode – The data associated with the output buffer is used as an input to this block, implementing an open drain function. The software will use this connection when it sees an open drain software attribute. The section on PURESPEED I/O Buffer Configurations provides further information on the open drain software attribute.

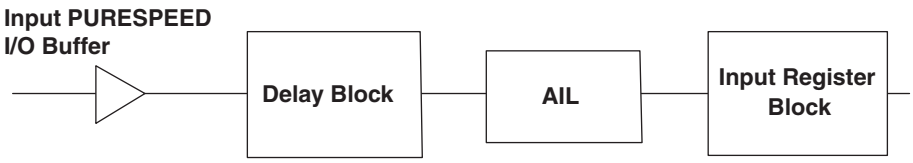
Update Block
This block is used during gearing to allow reliable transfer of data from an edge clock running at a different rate to the core primary or secondary clock. This allows all the geared I/Os to be synchronized simultaneously.

Control Block
The control logic block allows the modification of control signals before transfer to the I/O logic block. All the control signals can optionally be inverted before entering the I/O logic block. The use of Global Set/Reset (GSR) can be enabled or disabled. It can route either the edge clock or the FPGA primary or secondary clock to the high-speed clock nets. The clock provided to the I/O logic by routing is used as the slow-speed clocks. These low-speed clocks can operate at rates of up to 700MHz.

Input Delay (INDEL)

In each of the input modes described above the user can choose to delay the input signal before it is registered. The primary use for this input delay line is to achieve zero hold time for the input registers or to rotate the clock on data signals for clock-forwarded interfaces.

Figure 5. Delay Block

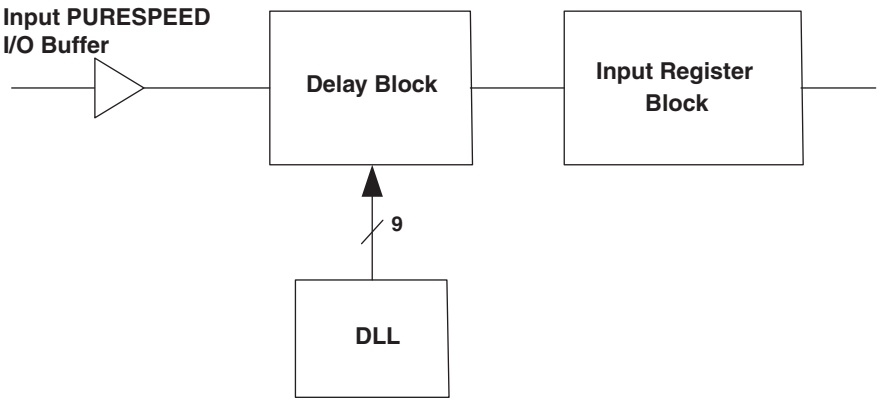


The delay block is built from a tap delay line and can be used in one of the following modes.

- **Primary Clock Injection Removal** – Removes delay caused by the primary clock injection time on the input data and guarantees zero hold time as long as primary clock input pins are used for the clocks.
- **Edge Clock Injection Removal** – Removes delay caused by the edge clock injection time on the input data and guarantees zero hold time as long as edge clock input pins in the same bank are used.
- **Edge Clock Injection Match** – Matches the edge clock injection delay as close as possible when the edge clock pins in the same bank are used. This is used on the receive of clock-forwarded interfaces so that the clock and data arrive at the same PIO input flip-flops at essentially the same time. This will allow the clock to be rotated 90 degrees, thus creating the best case data valid window for these interfaces.
- **User Defined** – Delay can be manually assigned by using the coarse delay and fine delay steps.
- **DLL** – This method combines the input delay block with a DLL to provide bus-based alignment capability for data rates up to ~400Mbps.This mode preserves a fixed clock/data phase relationship by aligning the incoming clock and data bus under DLL control. Another advantage is that this mode automatically tracks/compensates for delay variations due to process, voltage and temperature. The delay taps in the input delay block match those in the DLL for process, voltage and temperature compensation.

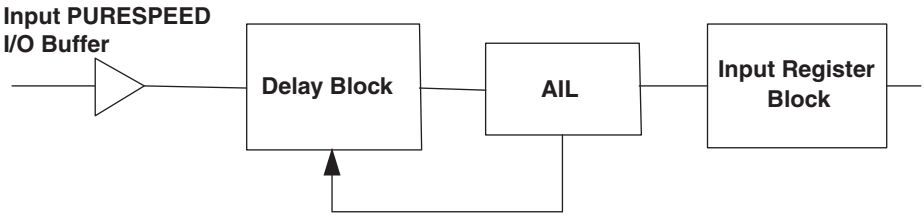
  The programmable delay block receives the shift values from the corner DLLs. Entering each bank, a pair of 9-bit buses provides the capability of extending digital codes from DLLs over to adjacent banks. This 9-bit delay code from the DLL is input to the programmable delay block.

Figure 6. Input Delay Using DLL



- **Adaptive Input Logic** – When Adaptive Input Logic (AIL) is used, it dynamically adjusts the delay of each input data path relative to the clock to ensure adequate setup and hold time margins at every input register. AIL is present on PADA (A/B differential pair) and PADC (C/D differential pair) on the left, right and bottom banks of the device. But only PADA or PADC per PIO can use the AIL block.

Figure 7. Input Delay Using Adaptive Input Logic



The Design and Usage Details section of this document explains the input delay block in detail. It also provides examples of the different modes and explains the advantages of using each of the modes

PURESPEED I/O Banking Overview

LatticeSC devices have seven general-purpose I/O banks and two SERDES banks. Each PURESPEED I/O bank has a V_{CCIO} supply voltage, two reference voltages, V_{REF1} and V_{REF2}, and dedicated V_{TT} pins. There is no V_{TT} pin on Bank 1. The top and bottom banks support all V_{CCIO} values up to and including 3.45V. The left and right banks support all values up to and including 2.625V. The banks on the left and right sides support true differential output drivers. All banks support differential input buffers and complementary output drivers. A detailed PURESPEED I/O banking description is provided in the Design and Usage Details section of this document.

Power Supplies

Table 3 lists all the power supplies required for the device. Each bank has a separate V_{CCIO} supply that powers the single-ended output drivers and the ratioed input buffers such as LVTTL, LVCMOS, PCI and PCI-X. In addition to the bank V_{CCIO} supply, this device has a V_{CC} core logic power supply, V_{CC1P2} internal 1.2V power supply, V_{CCPLL} PLL power supply and a V_{CCAUX} auxiliary supply (greater than V_{CC}) that powers the differential output drivers and the differential and referenced input buffers. The JTAG pins have a separate V_{CCJ} power supply that is independent of the bank V_{CCIO} supplies. V_{CCJ} determines the electrical characteristics of the LVCMOS JTAG pins, both the output high level and the input threshold. The extra power supplies required for SERDES are discussed in the LatticeSC/M Family Data Sheet.

Table 3. Power Supplies

Power Supply	Description	Value[1]
V_{CC}	Core power supply	1.0/1.2V
V_{CCIP2}	Internal 1.2V power supply	1.2V
V_{CCIO} (top and bottom)	I/O driver power supply for the top and bottom banks	1.2V/1.5V/1.8V/2.5V/3.3V
V_{CCIO} (left and right)	I/O driver power supply for left and right banks	1.2V/1.5V/1.8V/2.5V
V_{CCAUX}	Auxiliary power supply	2.5V
V_{CCJ}	Power supply for JTAG pins	1.8V/2.5V/3.3V

1. Refer to the LatticeSC/M Family Data Sheet for recommended min. and max. values.

Reference Voltages

Each bank can support up to two separate V_{REF} input voltages, V_{REF1} and V_{REF2}, that set the threshold for the referenced input buffers. Any I/O in the bank can potentially be used to input a V_{REF} voltage. In this case, this V_{REF1} pin can no longer be used to supply a reference voltage. Each bank has some number of dedicated V_{TT} pins (not shared with any other function) that are used to input the parallel termination voltage, V_{TT}, for the bank. Bank 1 does not have this external V_{TT} supply pin.

The Power Supplies section of this document explains the use of the different power supplies in further detail.

Clocking Block Diagram

The LatticeSC clocking structures consists of three main clock types:

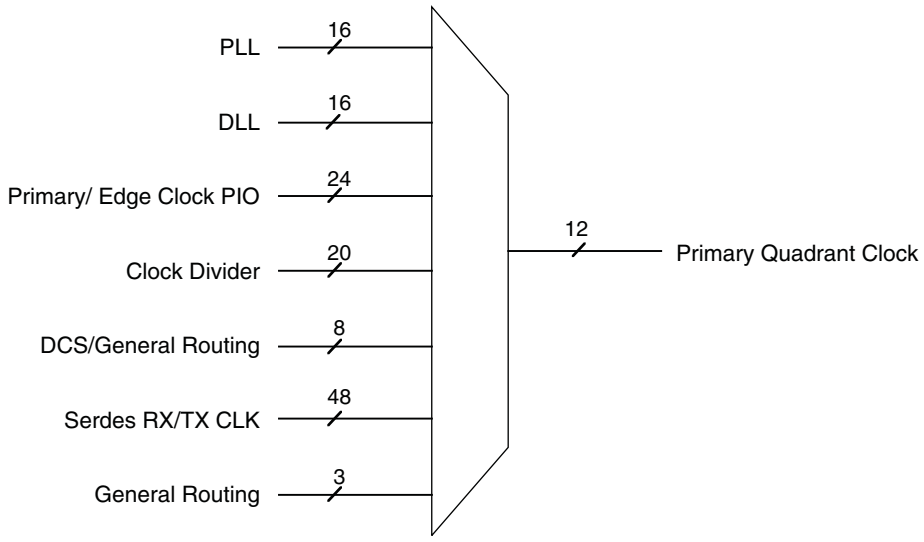
- Edge Clocks
- Primary Clocks
- Secondary Clocks

The edge clocks run along the edge of the device on all four sides and are dedicated for use with the I/O logic registers. There are eight edge clocks on the top, left and right sides of the device. The bottom side of the device has two sets of eight edge clocks. The edge clocks can be driven from edge clock PIOs in the same bank, primary clock PIOs in the same bank, routing, adjacent PLLs and DLLs and the ELSR output from the clock divider. The edge clocks can also be driven from primary clocks for low-skew clocks to all banks.

The primary clock network provides low-skew clocking to the entire device. It can also be used in the I/O logic.

Figure 8 shows the various inputs to the primary quadrant clock. Each quadrant will receive 12 primary clocks; thus, each device can have a total of 48 primary clocks.

Figure 8. Primary Clock Sources



In addition to the primary clocks and edge clocks, LatticeSC devices support secondary clocks that can be used locally to provide a slower-speed clock.

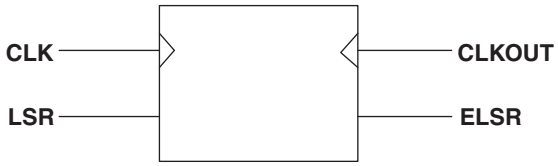
The high-speed data entering the device is clocked with the edge clock in the I/O logic registers on the I/O side. The geared data is then transferred to the primary clock before entering the FPGA logic.

The clocking structure and clock distribution is explained in detail in TN1098, LatticeSC sysCLOCK™ PLL/DLL User's Guide.

Clock Dividers

LatticeSC devices have four dedicated clock divider blocks on the left and right sides and four on each of the banks on the bottom side of the device The clock divider is used to divide the high-speed edge clock by 2 or 4. The low-speed clock is phase-matched to the high-speed clock. It can be used as a primary clock input or as the low-speed FPGA secondary clock required for DDR/SHX (x2, x4) I/O interfaces. The clock divider is also used to generate the Edge LSR (ELSR) signal. This signal is used by the UPDATE Block in each PIC to synchronize the gearing function between the high-speed I/Os and the slower-speed internal logic.

Figure 9. Clock Divider



PLL

All LatticeSC devices have eight PLLs, two on each corner of the device. Both the primary and the secondary outputs of each PLL can reach the 12 primary clocks in each quadrant. Each PLL output can also drive the edge clocks of the two banks adjacent to the PLL. Generally, PLLs have an advantage for driving signals off-chip due to low jitter and high stability.

PLLs can be used in applications such as clock insertion delay removal, clock phase adjustment, clock timing adjustment, clock skew control, frequency synthesis, and spread spectrum generation. The architecture of the PLL and its capabilities are discussed in detail in TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide.

DLL

In addition to the PLLs, LatticeSC devices have 12 DLLs. The outputs of the DLL can also feed the primary clocks in each quadrant. The DLLs are situated in the corners of the device along with the PLLs. The DLLs drive edge clocks in both banks adjacent to the corner in addition to the primary clocks. The delay block in the I/O logic receives two 9-bit dynamic delay bus from the DLL.

DLLs offer advantages for clock signals coming into the device (clock management). Because DLL clocks can be stopped, the DLLs will track changes in the input reference clock (due to low frequency wander) within a single clock cycle. DLLs can be used for clock injection removal, clock injection match, time reference delay and as a digital control setting to adjust a tracking delay element.

The architecture of the DLL and all its capabilities are discussed in detail in TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide.

Design and Usage Details

Banking

LatticeSC devices have seven PURESPEED I/O buffer banks; each is capable of supporting multiple I/O standards. Each PURESPEED I/O bank has its own I/O supply voltage (V_{CCIO}), and two voltage references, V_{REF1} and V_{REF2}, resources allowing each bank to be completely independent from each other. Figure 10 shows the seven banks and their associated supplies.

Figure 10. LatticeSC Banks

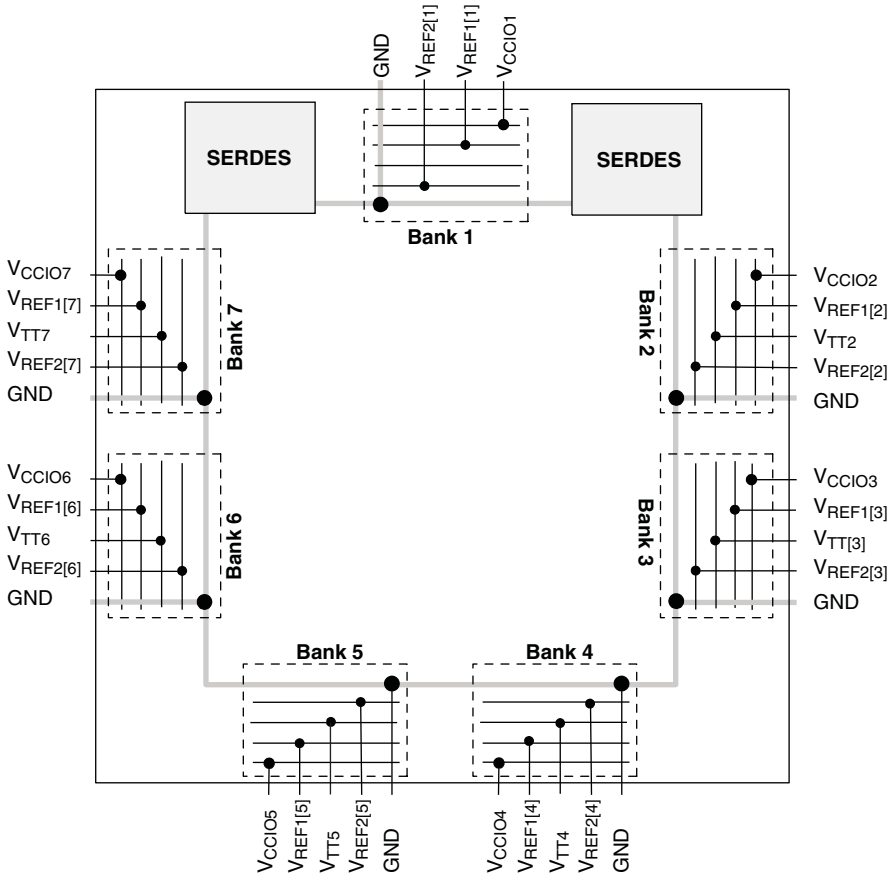


Table 4 lists the I/O standards supported in each of the banks.

Table 4. I/O Standards Supported by Different Banks

Description	Top Side Bank 1	Right Side Banks 2-3	Bottom Side Banks 4-5	Left Side Banks 6-7
I/O Buffer Type	Single-ended, Differential Receiver	Single-ended, Differential Receiver and Driver	Single-ended, Differential Receiver	Single-ended, Differential Receiver and Driver
Output Standards Supported	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 SSTL18_I SSTL25_ I, II SSTL33_ I, II HSTL15_I, II, III[1], IV[1] HSTL18_I, II,III[1], IV[1] SSTL18D_I, II SSTL25D_I, II SSTL33D_I, II HSTL15D_I, II HSTL18D_I, II PCI33 PCIX15 PCIX33 AGP1X33 AGP2X33 GTL[2], GTL+[2]	LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 SSTL18_I SSTL25_ I, II HSTL15_I,III[1] HSTL18_I,II,III[1] PCIX15 SSTL18D_I, II SSTL25D_I, II HSTL15D_I, II HSTL18D_I, II LVDS/RSDS Mini-LVDS GTL[2], GTL+[2]	LVTTL LVCMOS33 LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 SSTL18_I SSTL25_ I, II SSTL33_ I, II HSTL15_I, II, III[1], IV[1] HSTL18_I, II,III[1], IV[1] SSTL18D_I, II SSTL25D_I, II SSTL33D_I, II HSTL15D_I, II HSTL18D_I, II PCI33 PCIX15 PCIX33 AGP1X33 AGP2X33 GTL[2], GTL+[2]	LVCMOS25 LVCMOS18 LVCMOS15 LVCMOS12 SSTL18_I SSTL25_ I, II HSTL15_I,III[1] HSTL18_I,II,III[1] PCIX15 SSTL18D_I, II SSTL25D_I, II HSTL15D_I, II HSTL18D_I, II LVDS/RSDS Mini-LVDS GTL[2], GTL+[2]
Inputs	Single-ended, Differential	Single-ended, Differential	Single-ended, Differential	Single-ended, Differential
Clock Inputs	Single-ended, Differential	Single-ended, Differential	Single-ended, Differential	Single-ended, Differential
Differential Output Support via Emulation	LVDS/MLVDS/BLVDS/LVPECL	MLVDS/BLVDS/LVPECL	LVDS/MLVDS/BLVDS/LVPECL	MLVDS/BLVDS/LVPECL
AIL Support	No	Yes	Yes	Yes

1. Input only.
2. Input only. Outputs supported by bussing multiple outputs together.

Banking Rules

- If V_{CCIO} or V_{CCJ} for any bank is set to 2.5V, it is recommended that it be connected to the same power supply as V_{CCAUX}, thus minimizing leakage.
- If a bank contains differential output drivers that require a user-supplied external bias, V_{REF1} is consumed for this function, so V_{REF1} is no longer available for use by the input buffers in that bank. The Special I/O Pins section of this document explains the external bias implementation in more detail.
- When V_{TT} termination is required within a bank then all the V_{TT} pins in that bank must be connected to the same termination power supply. Pins with different IO_TYPEs can share the same V_{TT} pin as long as the termination voltage required is the same.
- Parallel termination to V_{TT} and common mode termination to V_{CMT} are mutually exclusive features within a bank. It is possible to have one or the other, but not both in the same bank.
- If V_{CMT} is used in a bank, since there is only one V_{CMT} line per bank, all the differential I/Os in a bank that use V_{CMT} must be identical. Differential termination that does not require common mode termination is always available in the bank.
- All the V_{TT} pins of the bank must be left unconnected when using common mode termination.
- Parallel termination using the Thevenin scheme is still available in the bank, only direct termination to V_{TT} is not

available when V_{CMT} is used for common mode termination.

- All the PIOs on the top and bottom sides of the device will support V_{CCIO} up to and including 3.45V. The left and right sides are 2.625V-capable and will support all V_{CCIO} up to and including 2.625V.
- Adaptive Input logic (AIL) is not available in bank 1.
- Parallel input termination to V_{CCIO} is not available in banks 1, 4 and 5 when V_{CCIO} is 3.45V. Parallel input termination to V_{TT} or ground will be available in banks 4 and 5, as long as the pad itself is not at 3.45V. Parallel input termination to ground will also be available.

I/O Placement Rules

- LVDS and RSDS outputs are available in banks 2, 3, 6 and 7 on the left and right sides of the device. The primary pads (PIOA and PIOB) are used as a differential output pair. In this case the alternate pads (PIOC and PIOD) can be used to assign differential input pair.
- You cannot mix two categories of outputs that have different biasing requirements.
- There is no such restriction on differential inputs unless the common mode termination to V_{CMT} is used. Therefore, LVDS and RSDS inputs when not common mode terminated to V_{CMT} can coexist in the same bank.
- LVPECL33 inputs can only be used in banks with VCC10 $\leq$ 2.5V.
- The differential BLVDS25, MLVDS25, LVPECL33 outputs are implemented using complementary single-ended drivers with external resistors.
- I/O standards requiring 3.3V VCCIO are only available on the top and bottom banks 1, 4 and 5.
- Only one of out of four PIO pads, PIOA or PIOC, in banks 2 through 7 will support Adaptive Input Logic (AIL). Inputs using AIL should be assigned to either of these pads. PIOC is available only if PIOA is not used as an input pin.
- Any of the I/O pins in the bank can be used to input V_{REF} voltage. If differential outputs are in the banks and require an external bias then the V_{REF1} of the bank is used to supply the external bias. The ispLEVER design tool will pick a default location for the V_{REF} pins.

Power Supplies

V_{CCIO} – Each bank has a separate V_{CCIO} supply that powers the single-ended output drivers and the ratioed input buffers such as LVTTL, LVCMOS, PCI and PCI-X. The V_{CCIO} voltage applied to the bank determines the ratioed input standards that can be supported in that bank. Banks 1, 4 and 5 support 1.2V to 3.3V single-ended driver capability. Banks 2, 3, 6 and 7 support single-ended drive from 1.2V to 2.5V. The V_{CCIO} voltage that the bank is set to will also determine the bank termination voltage and therefore determines the on-chip terminations available in that bank.

If V_{CCIO} or V_{CCJ} for any bank is set to 2.5V, it is recommended that it be connected to the same power supply as V_{CCAUX}, thus minimizing leakage.

V_{CCAUX} – In addition to the bank V_{CCIO} supply, this device has a V_{CC} core, V_{CC1P2} power supply, and a V_{CCAUX} auxiliary supply (greater than V_{CC}) that powers the differential output drivers and the differential and referenced input buffers. This voltage pin must always be connected to 2.5V. When the V_{CCIO} or V_{CCJ} of the bank are to be 2.5V, it is recommended that they be tied to the same power supply as V_{CCAUX}.

V_{TT}/V_{CMT} – Each bank has some number of dedicated V_{TT} pins (not shared with any other function) that are used to input the parallel termination voltage, V_{TT}, for the bank. The allowable V_{TT} range is from 0.5V to V_{CCAUX} - 0.5V, independent of the value of V_{CCIO}.

This supply can also be used to provide a common mode termination for differential input buffers if not used for parallel termination. When V_{TT} termination is not required, or used to provide the common mode termination voltage (V_{CMT}), these pins can be left unconnected on the device. Parallel termination to V_{TT} and common mode termination are mutually exclusive features in a bank.

Software Implementations – The V_{CMT} software attribute can be used to choose between V_{CMT}, V_{TT} and DDR_II termination.

$V_{REF1/2}$ – Each bank can support up to two separate V_{REF} input voltages, V_{REF1} and V_{REF2}, that set the threshold for the referenced input buffers. Any I/O in the bank can potentially be used to input a V_{REF} voltage. Once the I/O pin within the bank becomes the external V_{REF} input pin, it is no longer available for use as an I/O. If differential outputs that require external bias are implemented in a bank, then the V_{REF1} of that bank is used to provide this bias. This V_{REF1} pins can no longer available as a V_{REF} pin.

Software Implementations – The ispLEVER software will pick two pins as the default V_{REF} pins in a bank and these are indicated in the package pinout. These default locations can be overridden by the user in the software.

Assigning VREF/ VREF Groups for Referenced Inputs – Any I/O pin within a bank can be used to input the two V_{REF} voltages. Once an I/O pin within a bank becomes an external V_{REF} input pin, it is no longer available for use as an I/O. The user can group buffers to a particular V_{REF} rail, V_{REF1} or V_{REF2}. This grouping is done using the PGROUP V_{REF} preference along with the LOCATE PGROUP preference.

Preference syntax:

```
PGROUP <pgrp_name> [(VREF <vref_name>)+] (COMP <comp_name>)+;
LOCATE PGROUP <pgrp_name> BANK <bank_num>;
LOCATE VREF <vref_name> SITE <site_name>;
```

Example showing VREF groups:

```
PGROUP “vref_pg1” VREF “ref1” COMP “ah(0)” COMP “ah(1)” COMP “ah(2)” COMP “ah(3)”
COMP “ah(4)” COMP “ah(5)” COMP “ah(6)” COMP “ah(7)”;

PGROUP “vref_pg2” VREF “ref2” COMP “al(0)” COMP “al(1)” COMP “al(2)” COMP “al(3)”
COMP “al(4)” COMP “al(5)” COMP “al(6)” COMP “al(7)”;
LOCATE VREF “ref1” SITE “E11”;
LOCATE VREF “ref2” SITE “F12”;
Or
LOCATE PGROUP “ vref_pg1” BANK 2;
LOCATE PGROUP “ vref_pg2” BANK 2;
```

The second example shows V_{REF} groups, “vref_pg1” assigned to VREF “ref1” and “vref_pg2” assigned to “ref2”. The user must then lock these to either V_{REF1} or V_{REF2} using the LOCATE preference. The user can also designate to which bank the V_{REF} group should be located. The software will then assign these to either V_{REF1} or V_{REF2} of the bank.

If the PGROUP V_{REF} is not used, the software will automatically group all pins that need the same V_{REF} reference voltage. This preference is most useful when more than one bus uses the same reference voltage and you wish to associate each of these buses to different V_{REF} resources.

Clocking Considerations

To see the different clock considerations, refer to TN1098, LatticeSC sysCLOCK PLL/DLL User’s Guide.

I/O Assistant

The I/O Assistant process in the ispLEVER software can be used to design the I/Os before starting the design itself. The software provides an option for choosing the I/O Assistant design flow instead of the standard design flow. IPexpress can be used to generate the I/O modules for the I/O logic block. The designer can then assign the I/O attributes and also lock the I/Os pins. The DRC check with the I/O placement process will check for device-specific I/O placement and banking rules and help the user place the I/Os efficiently. The I/O Assistant thus gives the designer the advantage of have a working I/O ring with all the pins assigned before he starts the actual design. For details on using the I/O Assistant, refer to the ispLEVER Help.

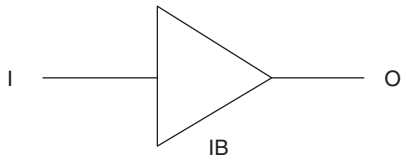
I/O Buffers

Library Elements

The ispLEVER software includes various library elements to implement the different PURESPEED I/O buffer types. These elements can be inferred in the software. The figures below show the four basic I/O buffer elements. All combinations can be made using these primitives. Detailed information on the library elements can be found in the *FPGA Libraries Manual* available through ispLEVER Help.

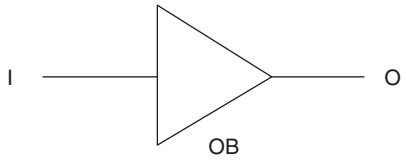
IB – The software will use the IB primitive to implement the input buffers. You can also instantiate this in your HDL to do the same function.

Figure 11. Input Buffer



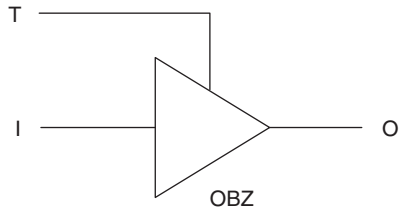
OB – The software will use the OB primitive to implement the output buffers. You can also instantiate this in your HDL to do the same function.

Figure 12. Output Buffer



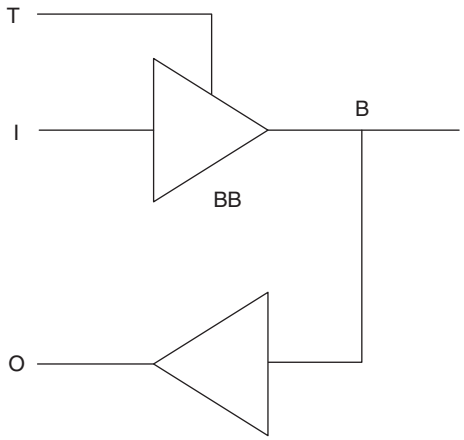
OBZ – The software will use the OBZ primitive to implement an output buffer with tristate control. You can also instantiate this in your HDL to do the same function.

Figure 13. Output Buffer with Tristate Control



BB – The software will use the BB primitive to implement the bi-directional buffers. You can also instantiate this in your HDL to do the same function.

Figure 14. Bi-directional Buffer



PURESPEED I/O Buffer Configurations

Programmable Drive Strength: All LVCMOS and LVTTL single-ended drivers have programmable drive strength. Each primary and alternate pad can operate in programmable drive strength mode. The user must consider the maximum allowable current per bank and the package thermal limit current when selecting the drive strength. Table 5 shows the available drive settings for single-ended LVCMOS and LVTTL I/O standards.

Table 5. Programmable Drive Values for Single-ended Buffers

Single-ended I/O Standard	Programmable Drive (mA)
LVCMOS12	2, 4, 8, 12
LVCMOS15	4, 8, 12, 16
LVCMOS18	4, 8, 12, 16
LVCMOS25	4, 8, 12, 16
LVCMOS33	8, 16, 24
LVTTL	8, 16, 24

In addition to the single-ended buffers, differential buffers such as LVDS and RSDS also support programmable differential drive. This gives the user additional flexibility when dealing with non-standard system configurations and termination values. Table 6 lists all the available programmable drive strengths.

Table 6. Differential Drive Setting

Differential Standard	Programmable Drive (mA)
LVDS	2, 3.5, 4, 6
RSDS	2, 3.5, 4, 6

Tristate Control – On the output side, each single-ended driver has a separate tristate control. The differential driver uses the same tristate control as the single-ended driver associated with the primary true pad.

Open Drain Control – All single-ended drivers support open drain separately on each I/O, independent of other I/Os in the bank. Software implements open drain by driving both the output data and tristate with the same logic signal, resulting in a driver that drives active low but goes into tristate rather than driving active high. You can assign an output to be an open drain driver by setting the OPENDRAIN software attribute.

Complementary Outputs – All the single-ended drivers associated with the “primary complement” pad can optionally be driven by the complement of the data that drives the single-ended driver associated with the “primary true” pad. This allows a pair of single-ended drivers to be used to drive complementary outputs with the lowest possible skew between the signals. This is used for driving complementary SSTL and HSTL signals (as required by the differential SSTL and HSTL clock inputs on synchronous DRAM and synchronous SRAM devices respectively). It can also be used in conjunction with off-chip resistors to emulate LVPECL, BLVDS and MLVDS output drivers. Complementary outputs are available on all primary pairs. You can assign complementary outputs in the software by choosing the correct IO_TYPE attribute. See the Software Attributes section for more information.

Bus Maintenance Circuit – Each pad has a weak pull-up, weak pull-down and a weak buskeeper capability. The pull-up and pull-down settings offer a fixed characteristic which is useful in creating wired logic such as wired ORs. However the current can be slightly higher than other options depending on the signal state. The bus keeper option latches the signal in the last driven state holding it at a valid level with minimal power dissipation. You can also choose to turn off the bus maintenance circuitry, minimizing power dissipation and input leakage. Note in this case, it is important to ensure that inputs are driven to a known state to avoid unnecessary power dissipation in the input buffer.

The weak bus keeper is not available when the V_{CCIO} of the bank is assigned to 3.3V. Weak internal pull-ups are available in all I/O banks. However, in I/O banks powered by supplies greater than V_{CCAUX} (2.5V), the I/O will not rise to the V_{CCIO} supply rail. This is pertinent in Banks 1, 4, and 5. If the design requires pull-ups to go to the supply rail in these banks, it is suggested that the pull-up resistor be added to the PCB.

PCI Clamp – A programmable PCI clamp is also available on the top and bottom banks of the device. These features and the PCI clamp can be turned “ON” or “OFF” on each pin independently.

The PCI clamp is used when implementing a 3.3V PCI interface. The PCI Specification, Revision 2.2 requires the use of clamping diodes for 3.3V operation. For more information on the PCI interface, please refer to the PCI Specification, Revision 2.2.

Programmable Slew Rate Control – All the output and bi-directional buffers also have an optional programmable output slew rate control that can be configured for either low-noise or high-speed performance. Each I/O pin has an individual slew rate control. This allows designers to specify slew rate control on a pin-by-pin basis. This slew rate control affects both the rising and falling edges.

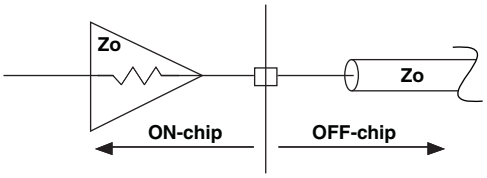
Programmable Terminations

Single-Ended Output Terminations

Series Output Termination

When driving lumped loads that are predominantly capacitive, the programmable series termination driver allows the output slew rate to be controlled. The output rise and fall times are dominated by the RC time constant formed by the output impedance and the load capacitance. When driving transmission line loads, the programmable series output termination allows the output impedance to be matched to the characteristic impedance of the transmission line. This eliminates the need for off-chip series termination resistors. The series output terminations are available for all V_{CCIO} voltages except 3.3V.

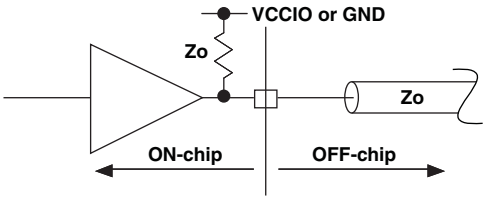
Figure 15. Programmable Series Output Termination



Parallel Output Termination to V_{CCIO} or GND

Parallel terminations for output are available using the dedicated terminations on the pads independent of other pins in the bank.

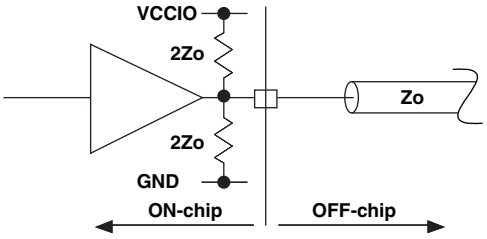
Figure 16. Parallel Termination to VCCIO or GND



Parallel to $V_{CCIO/2}$

The dedicated parallel terminations on the output pad can be used to generate a Thevenin termination by turning on both the Parallel termination to V_{CCIO} and the parallel termination to GND.

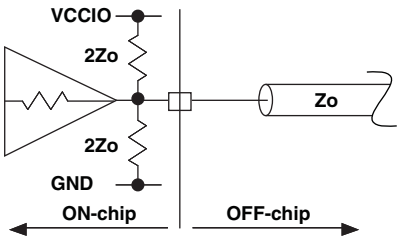
Figure 17. Parallel Terminations to $V_{CCIO}/2$



Combination Series and Parallel Output

This termination is used for SSTL_II and HSTL_II applications. Combination series/parallel termination is available for all V_{CCIO} between 1.2V and 2.5V.

Figure 18. Combined Series and Parallel Termination to $V_{CCIO/2}$

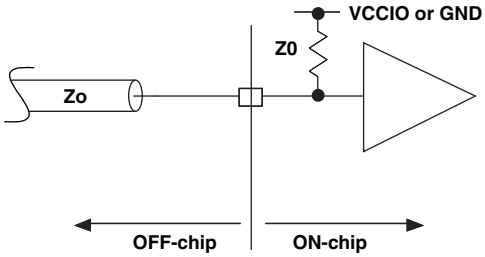


Single-Ended Input Terminations

Parallel Termination to V_{CCIO} or GND

On-chip single-ended input parallel termination is available as a programmable option on each pin, independent of all other pins in the bank V_{CCIO} or GND. When the bank V_{CCIO} is 3.3V, the parallel input termination is not available.

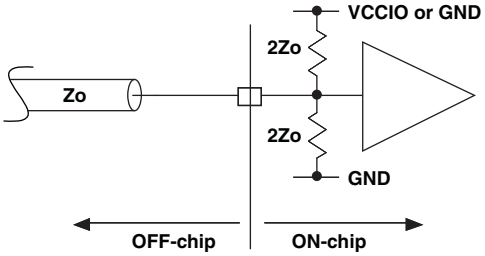
Figure 19. Parallel Input Terminations to V_{CCIO} or GND



Parallel Termination to $V_{CCIO}/2$

The dedicated parallel terminations on the input pad can be used to generate a Thevenin termination by turning on both the parallel termination to V_{CCIO} and the parallel termination to GND.

Figure 20. Parallel Input Terminations to $V_{CCIO}/2$

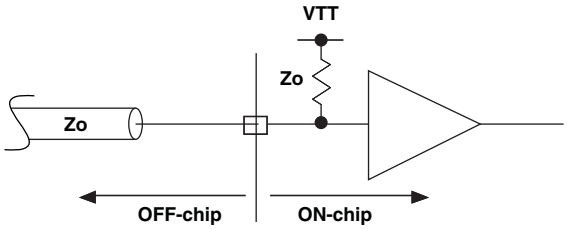


Parallel Input Termination Using V_{TT}

Some of the LatticeSC device packages have an optional V_{TT} termination voltage supply in each bank that allows parallel termination directly to V_{TT} rather than using the Thevenin scheme. The power dissipation with this type of termination is significantly less as compared to the Thevenin scheme. The user must supply V_{TT} to the device. Parallel termination to V_{TT} is available on a pin-by-pin basis within the bank. When V_{CCIO} is 3.3V, parallel input termination to V_{TT} is supported in the bank, as long as the pad itself is not at 3.3V.

All banks have some number of dedicated V_{TT} pins (not shared with any other function) that are used to input the parallel termination voltage, V_{TT}, for the bank expect for Bank 1. Bank 1 does not support V_{TT} termination. When V_{TT} termination is required, all V_{TT} pins in the bank must be connected to the termination power supply. When V_{TT} termination is not required, these pins should be left unconnected on the device.

Figure 21. Parallel Input Terminations to V_{TT}



DDR-II Memory Switchable Termination

This is a special mode for DDR-II and RLDRAM-II memory interfaces but can also be used for any high-speed bi-directional bus. This uses the input parallel termination to V_{TT}. This termination is controlled by the output tristate signal. During memory WRITE cycles when the FPGA is driving the lines, the parallel terminations are turned OFF. During READ cycles when the FPGA is receiving data, the parallel terminations are turned ON.

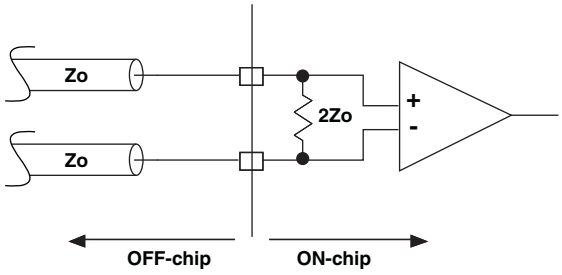
The value of the termination is programmable and can be selected based on the specific configuration of the system.

Programmable Differential Input Termination

Differential Termination

All primary and alternate pairs can be differentially terminated on all sides. When V_{CCIO} is 3.3V, differential input termination is still supported in the bank, as long as the PAD itself is not at 3.3V (SSTL33D for example).

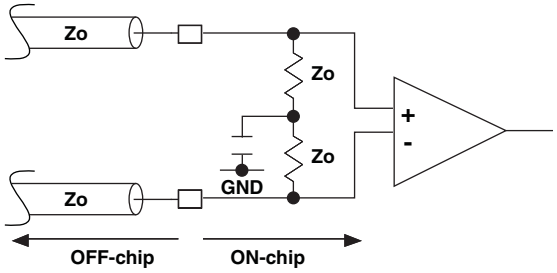
Figure 22. Differential Termination



Common Mode Termination

In addition to differential termination, common mode termination is available for differential signals using the V_{CMT} node in the bank. When common mode termination is required, V_{TT} parallel termination is not available in that bank. The V_{TT} pins should be left unconnected on the circuit board. Off-chip capacitors between V_{TT} and ground are not required, the on-chip capacitance on the V_{TT} node is sufficient to provide the necessary AC ground on the V_{CMT} node for common mode termination. Note that an extra off-chip capacitance could be connected from the V_{TT} pins to ground if necessary. Bank 1 does not support parallel termination to V_{TT}, but common mode termination is available in this bank.

Figure 23. Common Mode Termination



On-chip Termination Usage Examples

The tables below show on-chip termination examples for HSTL, SSTL and LVDS I/O standards.

Table 7. SSTL On-chip Termination Examples

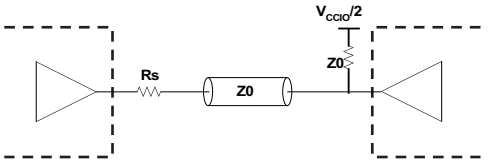
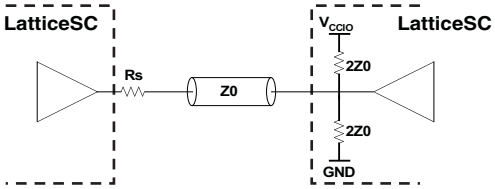
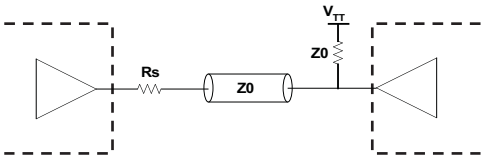
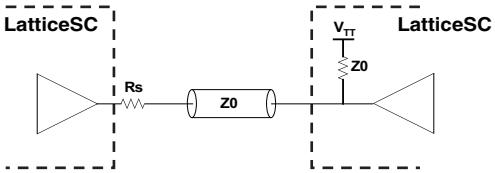
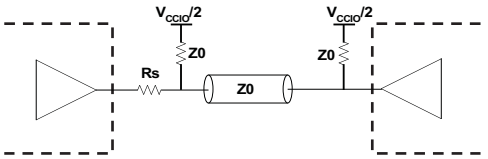
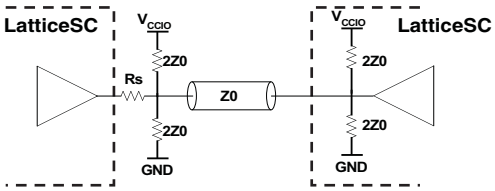
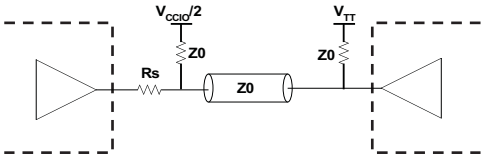
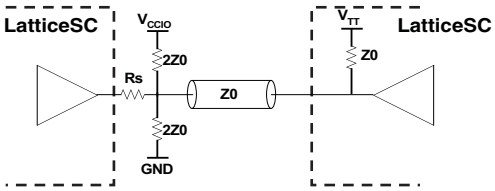
I/O Standard	Conventional Termination	LatticeSC On-Chip Termination Solution
SSTL18/25/33 Class I		
		
SSTL18/25/33 Class II		
		

Table 8. HSTL On-chip Termination Example

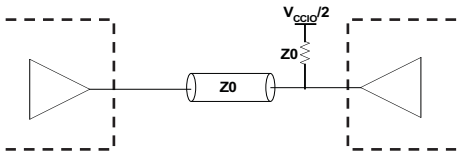
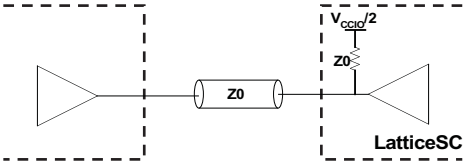
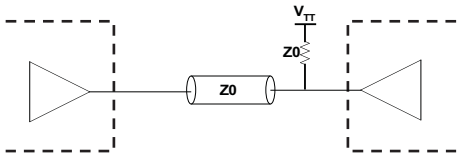
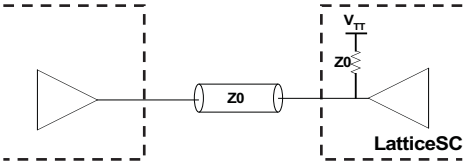
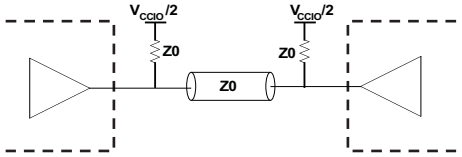
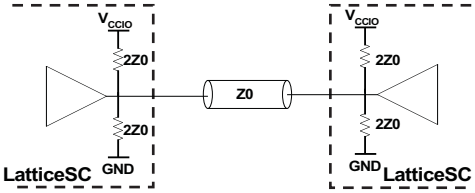
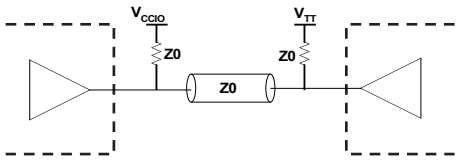
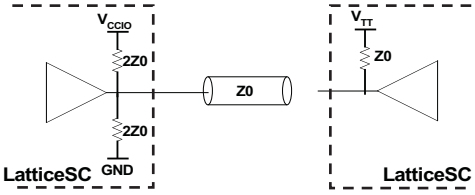
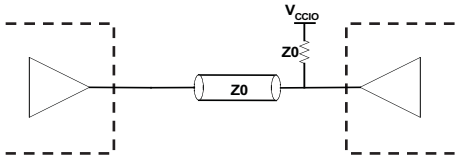
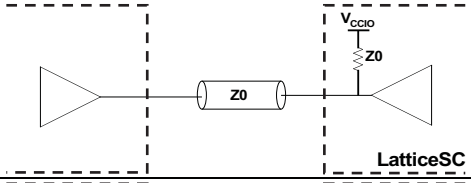
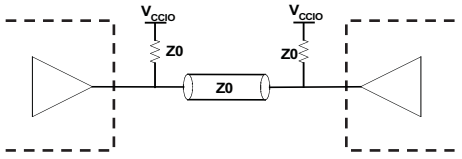
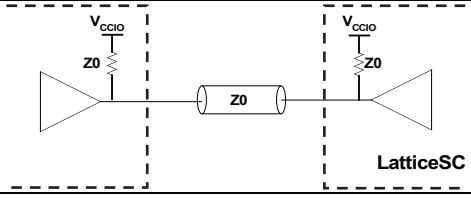
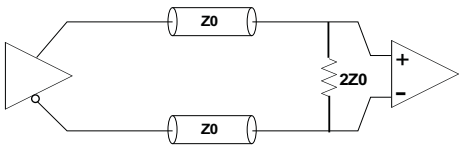
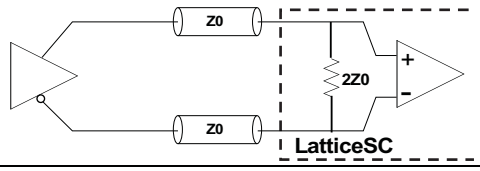
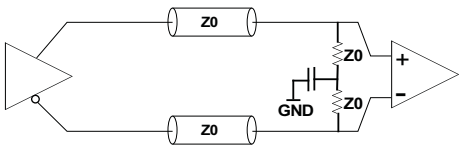
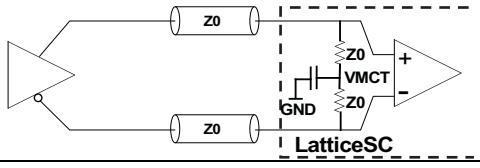
I/O Standard	Conventional Termination	LatticeSC On-Chip Termination Solution
HSTL15/ HSTL18 Class I		
		
HSTL15/ HSTL18 Class II		
		
HSTL15/ HSTL18 Class III		
HSTL15/ HSTL18 Class IV		

Table 9. Differential Termination Examples

I/O Standard	Conventional Termination	LatticeSC On-Chip Termination Solution
LVDS Differential Termination		
LVDS Common Mode Termination		

Software Attributes

The PURESPEED I/O software attributes allow the user to assign various PURESPEED I/O buffer features in the software. The user can specify these attributes in HDL, Preference Editor or as preferences in the preference file.

I/O_TYPE

This attribute is used to set the required PURESPEED I/O standard for an I/O. The V_{CCIO} required to set these I/O standards are embedded in the attribute name. There is no separate attribute to set the V_{CCIO} requirements.

Table 10. Values for Attribute IO_TYPE

I/O Standard	I/O Type
DEFAULT	LVCMOS25
LVDS[7]	LVDS
Reduced Swing Differential Signal	RSDS
Bus LVDS[6]	BLVDS25
MLVDS[6]	MLVDS25
LVPECL 3.3V[6]	LVPECL33
HSTL 15 Class I and II	HSTL15_I, HSTL15_II
HSTL 15 Class III and IV[4]	HSTL15_III, HSTL15_IV
Differential HSTL 15 Class I and II[1]	HSTL15D_I, HSTL15D_II
HSTL 18 Class I, II	HSTL18_I, HSTL18_II
HSTL 18 Class III and IV[4]	HSTL18_III, HSTL18_IV
Differential HSTL 15 Class I and II[1]	HSTL18D_I, HSTL18D_II
SSTL 18 Class I and II	SSTL18_I, SSTL18_II
Differential SSTL 18 Class I and II	SSTL18D_I, SSTL18D_II
SSTL 25 Class I and II	SSTL25_I, SSTL25_II
Differential SSTL 25 Class I and II[1]	SSTL25D_I, SSTL25D_II
SSTL 33 Class I and II	SSTL33_I, SSTL33_II
Differential SSTL 33 Class I and II[1]	SSTL33D_I, SSTL33D_II
GTL Plus 1.5V[4]	GTLPLUS15
GTL 1.2V[4]	GTL12
LVCMOS 3.3V, 2.5V, 1.8V, 1.5V, 1.2V	LVCMOS33, LVCMOS25, LVCMOS18, LVCMOS15, LVCMOS12
Differential LVCMOS Outputs[2, 5]	LVCMOS33D, LVCMOS25D, LVCMOS18D, LVCMOS15D, LVCMOS12D
LVTTL 3.3V	LVTTL33
Differential LVTLL Output[3, 5]	LVTTL33D
PCI 3.3V[8]	PCI33, PCI33_REF
PCIX 3.3V[8]	PCIX33, PCIX33_REF
PCIX 1.5V	PCIX15
AGP1X 3.3V[8]	AGP1X33, AGP1X33_REF
AGP2X 3.3V	AGP2X33

1. HSTL15D_I, HSTL15D_II, SSTL18D_I, SSTL18D_II, SSTL25D_I, SSTL25D_II SSTL33D_I, SSTL33D_II are complementary HSTL and SSTL inputs and outputs.
2. LVCMOS25D, LVCMOS33D, LVCMOS18D, LVCMOS15D, LVCMOS12D are complementary LVCMOS output drivers.
3. LVTTL33D is complementary LVTTL driver.
4. Inputs only.
5. Outputs only
6. These differential output buffers are generated using complementary LVCMOS output buffers and an external resistor network. Refer to the LatticeSC/M Family Data Sheet for information on how these are implemented.
7. Users can implement the mini-LVDS I/O standard by setting IO_TYPE to LVDS.
8. PCI33_REF, PCI33X_REF and AGP1X33_REF are internally referenced IO_TYPEs. When you select one of these IO_TYPEs, the software will consume a user I/O pin to implement the internal VREF circuit. This VREF pin should be left unconnected on the PCB.

TERMINATEVCCIO

This attribute determines the on-chip parallel input and output termination. This attribute is available for each individual pin in the bank. The V_{CCIO} to which the termination is available depends on the V_{CCIO} of the bank.

Values: OFF, 50, 100, 120

Default: OFF

TERMINATEGND

This attribute determines whether the on-chip parallel input and output termination is a programmable option that is available for all HSTL and SSTL I/O standards. This attribute is available for each individual pin in the bank, independent of all other pins in the bank.

Values: OFF, 50, 100, 120

Default: OFF

Note: To implement a split termination, for example a parallel termination of type 50 ohm to VCCIO/2, the user must turn ON the TERMINATEVCCIO to 100ohm and TERMINATEGND to 100 ohm.

TERMINATEVTT

This attribute sets the on-chip input parallel termination to $V_{TT.}$ This programmable option can be set for each I/O individually. When V_{CCIO} is 3.3V, parallel input termination to VTT is still supported in the bank, as long as the pad itself is not at 3.3V.

Values: OFF, 60, 75, 120, 150, 210

Default: OFF

Table 11 lists the available parallel terminations and their respective software attribute values.

Table 11. Parallel Termination Attribute Values

Parallel Termination	TERMINATEVCCIO	TERMINATEGND	TERMINATEVTT
Default	OFF	OFF	OFF
120 to V_{CCIO}	120	OFF	
120 to GND	OFF	120	
60 to $V_{CCIO}/2$	120	120	
100 Ohm to GND	OFF	100	
100 Ohm to V_{CCIO}	100	OFF	
50 Ohm to GND	OFF	50	
50 Ohm to V_{CCIO}	50	OFF	
50 Ohm to $V_{CCIO}/2$	100	100	
60 Ohm to V_{TT}			60
75 Ohm to V_{TT}			75
120 Ohm to V_{TT}			120
150 Ohm to V_{TT}			150
210 Ohm to V_{TT}			210

IMPEDANCE

This attribute sets the on-chip programmable series output termination. This programmable option can be set for each I/O individually.

Programmable output impedance is not supported for 3.3V drivers.

Values: OFF, 25, 33, 50, 100

Default: OFF

Note: Here IMPEDANCE= OFF means that the user does not have control of the Impedance control logic. When the parameter IMPEDANCE=OFF, the IMPEDANCE is predetermined by specification of the IO_TYPE.

Based on the specific IO_TYPE and VCCIO voltage, the driver has a specific drive characteristic. For added flexibility, the LatticeSC device allows additional specific drive characteristics for some IO_TYPES. This helps the designer adjust for system-level effects like board characteristics, termination mismatch, etc.

Table 12. Series Termination Attribute Values

Series Output Termination (Ohms)	IMPEDANCE Value
25	25
33	33
50	50
100	100

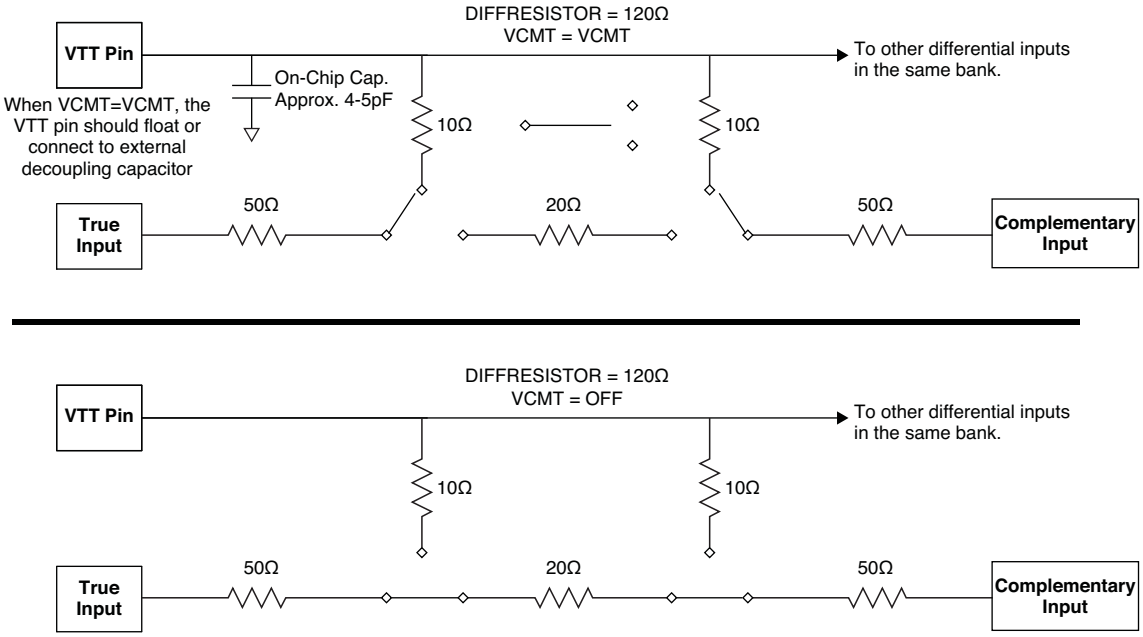
DIFFRESISTOR

The LatticeSC differential receiver input has two user-programmable options for providing on-device differential termination. Both options utilize the DIFFRESISTOR attribute to infer the presence of the resistor. This resistor network is designed to implement one of the terminating modes based on the VCMT attribute. Figure 24 shows the different termination configurations using the DIFFRESISTOR=120 attribute.

When VCMT=VCMT the resistor takes the path onto the on-chip VCMT connection. This allows an on-chip decoupling capacitor (approximately 4-5pF) to be present on the center-tap of the resistor network. It also permits off-chip filtering by utilizing a connection to the dedicated VTT pins of the device. This mode can only be used when VTT is not required for single-ended terminated inputs such as HSTL and SSTL. If VCMT=VCMT, the VTT pins must either be left floating or tied to an external decoupling capacitor. It should not be connected to board ground.

The second differential termination option is used in conjunction with the DIFFRESISTOR attribute generates a resistor network that does not use the VCMT connection. This option is available when VCMT=OFF and can be used in mixed I/O banks where differential inputs and single-ended VTT terminated inputs (such as SSTL and HSTL) must coexist.

Figure 24. Differential Termination Details



Values: OFF, 120, 150, 220, 420

Default: OFF

Table 13. Differential Termination Attribute Values

Series Output Termination (Ohms)	DIFFRESISTOR Value
120	120
150	150
220	220
420	420

VCMT

The attribute gives the user the option to choose between common mode input termination and V_{TT} parallel input termination. In the DDR II memory interface, this attribute has a DDR_II mode option. Choosing the DDR_II will allow parallel input termination to V_{TT} when receiving and series output termination while transmitting.

Table 14. VCMT Settings

VCMT	Description
OFF	Default value
VCMT	Turns on common mode differential input termination
VTT	Turns on parallel input termination to V_{TT}
DDR_II	Used for bi-directional DDR signals. Enables both the parallel input termination to V_{TT} when I/O is an input and series input termination when the I/O is an output.

DRIVE

The DRIVE attribute will set the programmable drive strength for all the LVCMOS and LVTTL output standards.

Table 15. DRIVE Settings

Output Standard	DRIVE (mA)	Default (mA)
LVTTL	8, 16, 24	8
LVCMOS33	8, 16, 24	8
LVCMOS25	4, 8, 12, 16	8
LVCMOS18	4, 8, 12, 16	8
LVCMOS15	4, 8, 12, 16	8
LVCMOS12	2, 4, 8, 12	8

SLEWRATE

The SLEWRATE attribute is available for all output and bi-directional drivers. Each I/O pin has an individual slew rate control. The user can set the SLEWRATE to be SLOW or FAST. By default, the software will set this attribute to SLOW for all the I/O standards.

DIFFCURRENT

The DIFFCURRENT attribute sets the differential output current drive for the differential drivers. All the LVDS and RSDS drivers on the left and right sides can be programmed to have different drive settings. Table 16 shows the available differential drive settings.

Table 16. DIFFCURRENT Settings

Differential Outputs	DIFFCURRENT (mA)	Default (mA)
LVDS	2, 3.5, 4, 6	3.5
RSDS	2, 3.5, 4, 6	2
Mini-LVDS	3.5, 4, 6	3.5

PULLMODE

The PULLMODE options can be enabled for each I/O independently. The KEEPER option is NOT available for the PULLMODE attribute when V_{CCIO} is 3.3V. The PULLMODE can be set on all LVCMOS, LVTTL, PCI and AGP inputs and outputs.

Values: UP, DOWN, NONE, KEEPER

Default: UP

Table 17. PULLMODE Values

PULL Options	PULLMODE VALUE
Pull-up (Default)	UP
Pull-down	DOWN
Bus Keeper	KEEPER
Pull-off	NONE

PCICLAMP

The PCICLAMP option is available for PCI and AGP inputs and outputs. It can be enabled on each I/O independently. PCICLAMP is available on all the I/Os on the top and bottom banks of the device.

Values: ON, OFF

Default: ON

REFCIRCUIT

There is an internal circuit to turn on the bias for the LVDS and RSDS I/O standards. The user can also choose to provide an external bias. The selection is made using the REFCIRCUIT attribute.

Values: OFF, INTERNAL, EXTERNAL

Default: OFF

Note: Refer to the Special I/O Pins section of this document for details on assigning this differential bias.

PWRSAVE

When using comparator type input pins (pins that use V_{REF}), such as GTL, HSTL, and SSTL, under certain frequencies the user can turn ON the PWRSAVE to reduce power dissipation on the input pins.

The PWRSAVE feature can also be used for differential input buffers including BLVDS25, MLVDS25, LVDS, and RSDS. PWRSAVE will reduce the receiver tolerance to a lower common mode voltage than what is allowed by the standard specification. This reduction in receiver tolerance on the low side allows the input receiver to use less power.

Values: ON, OFF

Default: OFF

HDL Usage

All the PURESPEED I/O attributes mentioned above can be added directly in the HDL source file. This section lists only the HDL syntax for the PURESPEED I/O buffer attributes. Refer to the Mentor Graphics® and Synplicity® user manuals for a complete list of synthesis attributes. These manuals are available through the ispLEVER software Help.

VHDL Synplicity/ Mentor Graphics

Table 18 lists syntax and examples for the PURESPEED I/O attributes in VHDL when using the Mentor Graphics® or Synplicity® synthesis tools.

Table 18. VHDL Syntax for PURESPEED I/O Attributes

Attribute	Syntax
IO_TYPE	attribute IO_TYPE: string; attribute IO_TYPE of Pinname: signal is “IO_TYPE Value”;
TERMINATEVCCIO	attribute TERMINATEVCCIO: string; attribute TERMINATEVCCIO of Pinname: signal is “TERMINATEVCCIO Value”;
TERMINATEGND	attribute TERMINATEGND: string; attribute TERMINATEGND of Pinname: signal is “TERMINATEGND Value”;
TERMINATEVTT	attribute TERMINATEVTT: string; attribute TERMINATEVTT of Pinname: signal is “TERMINATEVTT Value”;
IMPEDANCE	attribute IMPEDANCE: string; attribute IMPEDANCE of Pinname: signal is “IMPEDANCE Value”;
DIFFRESISTOR	attribute DIFFRESISTOR: string; attribute DIFFRESISTOR of Pinname: signal is “DIFFRESISTOR Value”;
VCMT	attribute VCMT: string; attribute VCMT of Pinname: signal is “VCMT Value”;
DRIVE	attribute DRIVE: string; attribute DRIVE of Pinname: signal is “Drive Value”;
DIFFCURRENT	attribute DIFFCURRENT: string; attribute DIFFCURRENT of Pinname: signal is “DIFFCURRENT Value”;
SLEWRATE	attribute SLEWRATE: string; attribute SLEWRATE of Pinname: signal is “Slewrate Value”;
PULLMODE	attribute PULLMODE: string; attribute PULLMODE of Pinname: signal is “Pullmode Value”;
PCICLAMP	attribute PCICLAMP: string; attribute PCICLAMP of Pinname: signal is “PCIClamp Value”;
REFCIRCUIT	attribute REFCIRCUIT: string; attribute REFCIRCUIT of Pinname: signal is “REFCIRCUIT Value”;
PWRSAVE	attribute PWRSAVE: string; attribute PWRSAVE of Pinname: signal is “PWRSAVE Value”;

Verilog Synplicity

This section lists syntax and examples for all the PURESPEED I/O Attributes in Verilog using the Synplicity synthesis tool.

Table 19. Verilog/Synplicity Syntax for PURESPEED I/O Attributes

Attribute	Syntax
IO_TYPE	PinType PinName /* synthesis IO_TYPE=”IO_TYPE Value”*/;
TERMINATEVCCIO	PinType PinName /* synthesis TERMINATEVCCIO =” TERMINATEVCCIO Value”*/;
TERMINATEGND	PinType PinName /* synthesis TERMINATEGND =” TERMINATEGND Value”*/;
TERMINATEVTT	PinType PinName /* synthesis TERMINATEVTT =” TERMINATEVTT Value”*/;
IMPEDANCE	PinType PinName /* synthesis IMPEDANCE =” IMPEDANCE Value”*/;
DIFFRESISTOR	PinType PinName /* synthesis DIFFRESISTOR =” DIFFRESISTOR Value”*/;
VCMT	PinType PinName /* synthesis VCMT =” VCMT Value”*/;
DRIVE	PinType PinName /* synthesis DRIVE =” DRIVE Value”*/;
SLEWRATE	PinType PinName /* synthesis SLEWRATE =“SLEWRATE Value”*/;
DIFFCURRENT	PinType PinName /* synthesis DIFFCURRENT =” DIFFCURRENT Value”*/;
PULLMODE	PinType PinName /* synthesis PULLMODE =” PULLMODE Value”*/;
PCICLAMP	PinType PinName /* synthesis PCICLAMP =” PCICLAMP Value”*/;
REFCIRCUIT	PinType PinName /* synthesis REFCIRCUIT =” REFCIRCUIT Value”*/;
PWRSAVE	PinType PinName /* synthesis PWRSAVE =” PWRSAVE Value”*/;

Verilog Mentor Graphics

This section lists syntax and examples for all the PURESPEED I/O attributes in Verilog using the Precision® RTL Synthesis tool.

Table 20. Verilog/ Mentor Graphics Syntax for PURESPEED I/O Attributes

Attribute	Syntax
IO_TYPE	//pragma attribute PinName IO_TYPE IO_TYPE Value
TERMINATEVCCIO	// pragma attribute PinName TERMINATEVCCIO TERMINATEVCCIO Value
TERMINATEGND	// pragma attribute PinName TERMINATEGND TERMINATEGND Value
TERMINATEVTT	// pragma attribute PinName TERMINATEVTT TERMINATEVTT Value
IMPEDANCE	// pragma attribute PinName IMPEDANCE IMPEDANCE Value
DIFFRESISTOR	// pragma attribute PinName DIFFRESISTOR DIFFRESISTOR Value
VCMT	// pragma attribute PinName VCMT VCMT Value
DRIVE	// pragma attribute PinName DRIVE DRIVE Value
SLEWRATE	// pragma attribute PinName SLEWRATE SLEWRATE Value
DIFFCURRENT	// pragma attribute PinName DIFFCURRENT DIFFCURRENT Value
PULLMODE	// pragma attribute PinName PULLMODE PULLMODE Value
PCICLAMP	// pragma attribute PinName PCICLAMP PCICLAMP Value
REFCIRCUIT	// pragma attribute PinName REFCIRCUIT REFCIRCUIT Value
PWRSAVE	// pragma attribute PinName PWRSAVE PWRSAVE Value

Legal Combinations

The software will only allow legal combinations of the software attributes to be used. When illegal combinations of the attributes are used, the software will issue an error message. The Preference Editor of the software will list the legal combinations of all the attributes in tabular format. Below are some of the legal combination rules to follow

when assigning the different I/O attributes in the software. Along with these rules, you also need to follow the banking and I/O placement rules specified in the Banking section of this document.

Input Legal Combinations

Table 21. Input Legal Combinations

IO_TYPE	TERMINATEVCCIO	TERMINATEGND	TERMINATEVTT	VCMT	DIFFRESISTOR	PULLMODE	PCICLAMP
LVDS	OFF	OFF	OFF	VCMT	OFF, 120, 150, 220, 420	NONE	OFF
LVDS (For mini-LVDS)	OFF	OFF	OFF	VCMT	OFF, 120, 150	NONE	OFF
BLVDS25	OFF	OFF	OFF	OFF	OFF	NONE	OFF
MLVDS25	OFF	OFF	OFF	OFF	OFF	NONE	OFF
RSDS	OFF	OFF	OFF	VCMT	OFF, 120, 150, 220, 420	NONE	OFF
LVPECL33	OFF	OFF	OFF	VCMT	OFF, 120, 150, 220, 420	NONE	OFF
HSTL15_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
HSTL15_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
HSTL15_III	OFF, 50	OFF	OFF	OFF	OFF	NONE	OFF
HSTL15_IV	OFF, 50	OFF	OFF	OFF	OFF	NONE	OFF
HSTL15D_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
HSTL15D_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
HSTL18_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
HSTL18_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
HSTL18_III	OFF, 50	OFF	OFF	OFF	OFF	NONE	OFF
HSTL18_IV	OFF, 50	OFF	OFF	OFF	OFF	NONE	OFF
HSTL18D_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
HSTL18D_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
SSTL18_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
SSTL18_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
SSTL18D_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
SSTL18D_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
SSTL25_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
SSTL25_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT	OFF	NONE	OFF
SSTL25D_I	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
SSTL25D_II	OFF, 120, 100	OFF, 120, 100	OFF, 60, 75, 120, 210	OFF, VTT, VCMT	OFF, 120, 150, 220, 420	NONE	OFF
SSTL33_I	OFF	OFF	OFF	OFF	OFF	NONE	OFF
SSTL33_II	OFF	OFF	OFF	OFF	OFF	NONE	OFF
SSTL33D_I	OFF	OFF	OFF	OFF	OFF	NONE	OFF
SSTL33D_II	OFF	OFF	OFF	OFF	OFF	NONE	OFF
GTLPLUS15	OFF, 50	OFF	OFF	OFF	OFF	NONE	OFF
GTL12	OFF, 50	OFF	OFF	OFF	OFF	NONE	OFF
LVTTL33	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE	OFF
LVCMOS33	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE	OFF
LVCMOS25	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE, KEEPER	OFF
LVCMOS18	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE, KEEPER	OFF
LVCMOS15	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE, KEEPER	OFF
LVCMOS12	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE, KEEPER	OFF
PCI33	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE	ON, OFF
PCIX33	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE	ON, OFF
PCIX15	OFF, 120, 100	OFF	OFF, 60, 75, 120, 210	OFF, VTT	OFF	UP, DOWN, NONE	ON, OFF
AGP1X33	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE	ON, OFF
AGP2X33	OFF	OFF	OFF	OFF	OFF	UP, DOWN, NONE	ON, OFF

Table 21 lists all the valid input software attributes and the allowable values for a given input PURESPEED I/O standard. Below are some of the rules to follow when assigning software attributes for inputs.

1. Users can implement mini-LVDS input by setting IO_TYPE to LVDS.
2. In order to do a common mode termination, the VCMT attribute should be set to VCMT.
3. Users can assign TERMINATEVTT or TERMINATEVCCIO/TERMINATEGND, but not both at the same time.
4. When terminating to VTT, the VCMT should be set to VTT.
5. The PULLMODE option 'KEEPER' is not available for 3.3V standards (LVCMOS33 and LVTTL).
6. IMPEDANCE is not available on inputs.
7. Parallel terminations, TERMINATEVCCIO/TERMINATEGND are not available for SSTL33_I and SSTL33_II.
8. The programmable clamp diode (on/off programmable only) is available for use with PCI33, PCIX33, PCIX15, AGP1X33 and AGP2X33 I/O standards.
9. LVPECL inputs cannot be placed in the bank when the VCCIO of the bank is set to 3.3V.

Output Legal Combinations

Table 22. Output Legal Combinations

IO_TYPE	IMPEDANCE	TERMINATEVCCIO	TERMINATEGND	DRIVE	OPENDRAIN	DIFFCURRENT	PULLMODE	PCICLAMP	SLEWRATE	REFCIRCUIT
LVDS	OFF	OFF	OFF	NA	OFF	3P5, 2, 4, 6	NONE	OFF	NA	INTERNAL, EXTERNAL
LVDS (For mini-LVDS)	OFF	OFF	OFF	NA	OFF	3P5, 4, 6	NONE	OFF	NA	INTERNAL, EXTERNAL
BLVDS25	100	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
MLVDS25	50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
RSDS	OFF	OFF	OFF	NA	OFF	2, 3P5, 4, 6	NONE	OFF	SLOW, FAST	INTERNAL, EXTERNAL
LVPECL33	OFF	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL15_I	OFF, 50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL15_II	OFF, 25	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL15D_I	OFF, 50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL15D_II	OFF, 25	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL18_I	OFF, 50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL18_II	OFF, 25	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL18D_I	OFF, 50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
HSTL18D_II	OFF, 25	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL18_I	OFF, 33	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL18_II	OFF, 33	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL18D_I	OFF, 33	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL18D_II	OFF, 33	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL25_I	OFF, 50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL25_II	OFF, 33	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL25D_I	OFF, 50	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL25D_II	OFF, 33	OFF, 120	OFF, 120	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL33_I	OFF	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL33_II	OFF	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL33D_I	OFF	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
SSTL33D_II	OFF	OFF	OFF	NA	OFF	NA	NONE	OFF	SLOW, FAST	OFF
LVTTL33	OFF	OFF	OFF	8, 16, 24	OFF, ON	NA	UP, DOWN, NONE	OFF	SLOW, FAST	OFF
LVTTL33D	OFF	OFF	OFF	8, 16, 24	OFF, ON	NA	UP, DOWN, NONE	OFF	SLOW, FAST	OFF
LVCMOS33	OFF	OFF	OFF	8, 16, 24	OFF, ON	NA	UP, DOWN, NONE	OFF	SLOW, FAST	OFF
LVCMOS33D	OFF	OFF	OFF	8, 16, 24	OFF, ON	NA	UP, DOWN, NONE	OFF	SLOW, FAST	OFF
LVCMOS25	OFF, 50, 100, 33, 25	OFF	OFF	8, 4, 12, 16	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
LVCMOS25D	OFF, 50, 100, 33, 25	OFF	OFF	8, 4, 12, 16	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF

Table 22. Output Legal Combinations (Continued)

IO_TYPE	IMPEDANCE	TERMINATEVCCIO	TERMINATEGND	DRIVE	OPENDRAIN	DIFFCURRENT	PULLMODE	PCICLAMP	SLEWRATE	REFCIRCUIT
LVCMOS18	OFF, 50, 100, 33, 25	OFF	OFF	8, 4, 12, 16	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
LVCMOS18D	OFF, 50, 100, 33, 25	OFF	OFF	8, 4, 12, 16	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
LVCMOS15	OFF, 50, 100, 33, 25	OFF	OFF	8, 4, 12, 16	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
LVCMOS15D	OFF, 50, 100, 33, 25	OFF	OFF	8, 4, 12, 16	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
LVCMOS12	OFF, 50, 100, 33, 25	OFF	OFF	8, 2, 4, 12	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
LVCMOS12D	OFF, 50, 100, 33, 25	OFF	OFF	8, 2, 4, 12	OFF, ON	NA	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST	OFF
PCI33	OFF	OFF	OFF	NA	OFF	NA	UP, DOWN, NONE	ON, OFF	SLOW, FAST	OFF
PCIX33	OFF	OFF	OFF	NA	OFF	NA	UP, DOWN, NONE	ON, OFF	SLOW, FAST	OFF
PCIX15	OFF	OFF	OFF	NA	OFF	NA	UP, DOWN, NONE	ON, OFF	SLOW, FAST	OFF
AGP1X33	OFF	OFF	OFF	NA	OFF	NA	UP, DOWN, NONE	ON, OFF	SLOW, FAST	OFF
AGP2X33	OFF	OFF	OFF	NA	OFF	NA	UP, DOWN, NONE	ON, OFF	SLOW, FAST	OFF

Table 22 lists all the valid output software attributes and the allowable values for a given output PURESPEED I/O standard. Below are some of the rules to follow when assigning software attributes for outputs.

1. Users can implement the Mini-LVDS output by setting IO_TYPE to be LVDS.
2. TERMINATEVTT, VCMT, DIFFERESISTOR are not available on outputs.
3. Parallel terminations, TERMINATEVCCIO/TERMINATEGND are not available for SSTL33_I and SSTL33_II.
4. When EXTERNAL REFCIRCUIT is used, the VREF1 of the bank is used to provide the required external bias and is no longer available as a VREF input pin.
5. BLVDS25, MLVDS25 and LVPECL drivers are implemented using complementary LVCMOS output buffers and external resistor network. See the LatticeSC/M Family Data Sheet for resistor values.
6. The DRIVE and IMPEDANCE settings can both be turned off, but if one of them is on, then they are mutually exclusive.
7. LVCMOS outputs can have PULLMODE of UP, DOWN, NONE, KEEPER, except for 3.3V where choices are limited to UP, DOWN, NONE.
8. IMPEDANCE selections are not available for 3.3V standards.
9. Only LVTTL and LVCMOS outputs can be assigned as OPENDRAIN.
10. The programmable clamp diode (on/off programmable only) is available for use with PCI33, PCIX33, PCIX15, AGP1X33 and AGP2X33 I/O standards.
11. Only LVDS and RSDS differential I/O types support the programmable drive (DIFCURRENT) attribute.

Bi-directional Legal Combinations

Table 23. Legal Combinations for Bi-directional Buffers

IO_TYPE	IMPEDANCE	TERMINATEVCCIO	TERMINATEGND	TERMINATEVTT	VCMT	DRIVE	OPENDRAIN	PULLMODE	PCICLAMP	SLEWRATE
BLVDS25	100	OFF	OFF	OFF	OFF	NA	OFF	NONE	OFF	SLOW, FAST
MLVDS25	50	OFF	OFF	OFF	OFF	NA	OFF	NONE	OFF	SLOW, FAST
HSTL15_I	50	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
HSTL15_II	OFF, 25	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
HSTL15D_I	50	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
HSTL15D_II	OFF, 25	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
HSTL18_I	50	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST

Table 23. Legal Combinations for Bi-directional Buffers (Continued)

IO_TYPE	IMPEDANCE	TERMINATEVCCIO	TERMINATEGND	TERMINATEVTT	VCMT	DRIVE	OPENDRAIN	PULLMODE	PCICLAMP	SLEWRATE
HSTL18_II	OFF, 25	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
HSTL18D_I	50	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
HSTL18D_II	OFF, 25	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL18_I	33	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL18_II	OFF, 33	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL18D_I	33	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL18D_II	OFF, 33	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL25_I	50	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL25_II	OFF, 33	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL25D_I	50	OFF	OFF	150, 60, 75	DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL25D_II	OFF, 33	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL25D_II	OFF, 33	OFF, 120	OFF, 120	OFF, 150, 60, 75	OFF, DDR_II	NA	OFF	NONE	OFF	SLOW, FAST
SSTL33_II	OFF	OFF	OFF	OFF	OFF	NA	OFF	NONE	OFF	SLOW, FAST
SSTL33D_II	OFF	OFF	OFF	OFF	OFF	NA	OFF	NONE	OFF	SLOW, FAST
LVTTL33	OFF	OFF	OFF	OFF	OFF	8, 16, 24	OFF, ON	UP, DOWN, NONE	OFF	SLOW, FAST
LVCMOS33	OFF	OFF	OFF	OFF	OFF	8, 16, 24	OFF, ON	UP, DOWN, NONE	OFF	SLOW, FAST
LVCMOS25	OFF, 50, 100, 33, 25	OFF	OFF	OFF	OFF	8, 4, 12, 16	OFF, ON	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST
LVCMOS18	OFF, 50, 100, 33, 25	OFF	OFF	OFF	OFF	8, 4, 12, 16	OFF, ON	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST
LVCMOS15	OFF, 50, 100, 33, 25	OFF	OFF	OFF	OFF	8, 4, 12, 16	OFF, ON	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST
LVCMOS12	OFF, 50, 100, 33, 25	OFF	OFF	OFF	OFF	8, 2, 4, 12	OFF, ON	UP, DOWN, NONE, KEEPER	OFF	SLOW, FAST
PCI33	OFF	OFF	OFF	OFF	OFF	NA	OFF	UP, DOWN, NONE	ON, OFF	SLOW, FAST
PCIX33	OFF	OFF	OFF	OFF	OFF	NA	OFF	UP, DOWN, NONE	ON, OFF	SLOW, FAST
PCIX15	OFF	OFF	OFF	60, 75, 150	DDR_II	NA	OFF	UP, DOWN, NONE	ON, OFF	SLOW, FAST
AGP1X33	OFF	OFF	OFF	OFF	OFF	NA	OFF	UP, DOWN, NONE	ON, OFF	SLOW, FAST
AGP2X33	OFF	OFF	OFF	OFF	OFF	NA	OFF	UP, DOWN, NONE	ON, OFF	SLOW, FAST

Table 23 lists all the valid software attributes for bi-directional I/Os and the allowable values for a given PURESPEED I/O standard. Below are some of the rules to follow when assigning software attributes for bi-directional I/Os.

1. REFCIRCUIT, DIFFCURRENT and DIFFRESISTOR are not available for bi-directional I/Os.
2. When VCMT is set to DDR_II, both the IMPEDANCE and the Parallel Termination to VTT, TERMINATEVTT can be turned on at the same time.
3. The DRIVE and IMPEDANCE settings are mutually exclusive for a given pin.

Special I/O Pins

The LatticeSC PURESPEED I/O interface has several special function pins. Some of these are shared and some are dedicated.

DIFFR

The DIFFR pin is shared with a regular I/O pin, one per bank on the left and right sides, but in a pre-determined location. If any differential drivers are required in a bank, this pin must have an off-chip resistor connected, 1K ohm +/- 1% to GND. If the bank has no differential drivers, this pin is available as a regular user I/O.

XRES

The XRES pin is a single dedicated pin located in one corner of the device. It is not associated with any of the PURESPEED I/O banks. This pin must have an external resistor of 1K+/- 1% ohm connected to GND. This resistor is used as the calibration reference for the on-chip Thevenin terminators.

The XRES calibration can be operated in the following two modes.

1. In the default mode, the update of PVT compensation will be turned off at the end of configuration. This does not allow for complete temperature compensation since the design is not yet running, but should be within a few percentage points of being accurate.
2. For increased accuracy, the user can turn on the XRES update signal. This provides PVT compensation when the design is running.

Users can select a mode of operation using one of the following bitstream generation (bitgen -g Rnet:val) options.

1. **XRES INIT** – The XRES control bits will be initialized during configuration to take a snapshot of PVT. They are then latched and will not change any more. This is the default setting.
2. **XRES OFF** – If this option is used, the XRES control lines are never powered up and all control bits are 0, resulting in minimum drive (also minimum termination). Users should be careful if on-die termination is used.
3. **XRES ON** – The XRES control lines are turned on and will be updated with PVT changes all the time.

The user can control this update by instantiating the pvtiocntl() software primitive in the HDL source file.

Configuration Pins
Some of the I/O pins are reserved for configuration. Refer to TN1080, LatticeSC sysCONFIG Usage Guide for details.

Others
VTT

As previously described, the VTT pins are dedicated for VTT use only. They are not shared with any other function. Use them for VTT or leave them unconnected. Each bank has some number of VTT pins except for Bank 1. Bank 1 does not support any external VTT parallel termination.

VREF1, VREF2

The VREF pins are shared with regular I/O pins. Any I/O pin can be programmed as a VREF pin. If the external bias is selected for a differential driver it is brought into the device using the VREF1 path, so VREF1 is no longer available for use in that bank. When one of the internally referenced IO_TYPEs are selected, the software will consume a VREF pin to implement the internal VREF circuit. This VREF will no longer be available in the bank. Users should leave this pin unconnected on the PCB.

Assigning Differential Bias

The differential drivers have user-selectable internal or external bias. If the external bias is selected, it is brought into the device using the VREF1 path, so VREF1 is no longer available for use in that bank. External bias can only be provided via the VREF1 pin.

Internal bias is selected as the default. The user can choose to use the external bias for applications that require a tighter than standard output common mode range. This can be set using the REFCIRCUIT software attribute.

All differential drivers in a bank must have the same bias. LVDS and RSDS outputs require the same bias. Hence LVDS and RSDS can be mixed in a bank.

Internal bias uses the VCCAUX of 2.5 to provide the bias. This could be noisy if you need to run at high speeds above 1 Gbps. In this case using an external bias would give you better overhead and signal integrity. When the external bias is used, the same value bias should be brought into the device for all standards. This is scaled on-chip to the correct value for LVDS or RSDS before being applied to the bank bias circuit.

Table 1-1. LatticeSC Differential Driver, External Bias Voltage

Description	Value (nom)	Tolerance	Units
Differential driver external bias	1.2	+/-5%	V

Pad File

The software generates a pad file that lists all the IO_TYPEs used, the pin assignments and the respective properties. It will also point out the special I/O pins required when using certain IO_TYPES. Figure 25 shows a pad file example from the software.

Figure 25. Pad File

Pinout by Port Name:

Port Name	Pin/Bank	Buffer Type	Site	Properties
A	AJ2/5	LVCMOS25_OUT	PB5A	DRIVE:8mA
AH_0	AC6/6	LVCMOS25_IN	PL57C	
AH_1	AJ1/5	LVCMOS25_IN	PB3B	
AH_2	AH2/5	LVCMOS25_IN	PB4B	
AH_3	AF4/5	SSTL25_I_IN	PB3C	TERMVCC:100 TERMGND:100 VREF1_
AH_4	AF16/4	SSTL33_II_IN	PB38D	VREF1_LOAD
AH_5	AD7/5	RSDS_IN	PB5C	VCMT:VCMT DIFFRES:120
AH_6	AG3/5	LVCMOS25_IN	PB4A	
AH_7	AD6/5	HSTL15_III_IN	PB4C	VREF2_LOAD
AL_0	V5/6	LVCMOS25_IN	PL43C	
AL_1	V4/6	LVCMOS25_IN	PL43D	
AL_2	V1/6	LVDS_IN	PL42A	
AL_3	J1/7	HSTL18_II_IN	PL26A	TERMVCC:120 TERMGND:120 VREF1_
AL_4	AE3/6	LVCMOS25_IN	PL56A	
AL_5	AH1/5	LVCMOS25_IN	PB3A	
AL_6	AF2/6	LVCMOS25_IN	PL57A	
AL_7	AE5/5	LVCMOS25_IN	PB3D	
AS	AF13/5	LVCMOS25_IN	PB20C	
B	L1/7	RSDS_OUT	PL27A	DIFFCUR:2 REFC:INT
BE	R8/7	HSTL18_II_OUT	PL27D	IMP:25
C	AK17/4	SSTL33_II_OUT	PB37B	
CLK	T27/3	HSTL15_IV_IN	PR40B	TERMVCC:50 VREF1_LOAD
D	AG2/6	SSTL25_I_OUT	PL57B	IMP:50
DIFFR_6	V6/6	REF_RESISTOR	PL42D	
DIFFR_7	M6/7	REF_RESISTOR	PL25D	
E	AK16/4	LVCMOS33_OUT	PB37A	DRIVE:8mA
F	AC7/6	LVCMOS25_OUT	PL57D	DRIVE:8mA
G	V2/6	LVDS_OUT	PL43A	DIFFCUR:3P5 REFC:EXT
H	P8/7	SSTL18_I_OUT	PL27C	
RST	D13/1	LVCMOS25_IN	PT25A	
VREF1_3	T26/3	VREF1_DRIVER	PR34C	
VREF1_4	AE24/4	VREF1_DRIVER	PB67C	
VREF1_5	AF9/5	VREF1_DRIVER	PB11C	
VREF1_6	R5/6	VREF1_DRIVER	PL34C	
VREF1_7	L6/7	VREF1_DRIVER	PL25C	
VREF2_5	AH13/5	VREF2_DRIVER	PB23D	

VCCIO by Bank:

Bank	VCCIO
1	2.5V
2	2.5V
3	1.5V
4	3.3V
5	2.5V
6	2.5V
7	1.8V

VREF by Bank

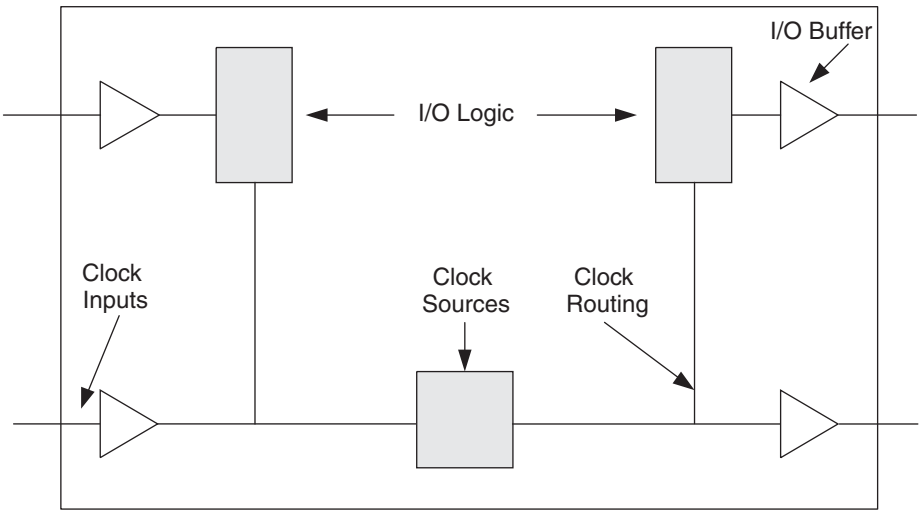
Vref	Pin	Bank / Vref #	Load
VREF1_3	T26	3 / VREF1	T27
VREF1_4	AE24	4 / VREF1	AF16
VREF1_5	AF9	5 / VREF1	AF4
VREF2_5	AH13	5 / VREF2	AD6
VREF1_6	R5	6 / VREF1	V2
VREF1_7	L6	7 / VREF1	J1

I/O Logic

Basic Building Blocks

When creating I/O interfaces in the LatticeSC architecture there are a few basic elements the user will need to understand. These elements are the I/O logic, I/O buffer, clock sources, clock routing, and clock inputs. Figure 26 provides a high-level view of these elements. Designing source synchronous, high-speed interfaces requires the user to balance the delays of the clock and the data. The LatticeSC architecture provides a rich set of features for fine tuning these delays. The following sections describe these elements in detail.

Figure 26. Basic I/O Elements



Using LatticeSC Architecture I/O Logic Elements

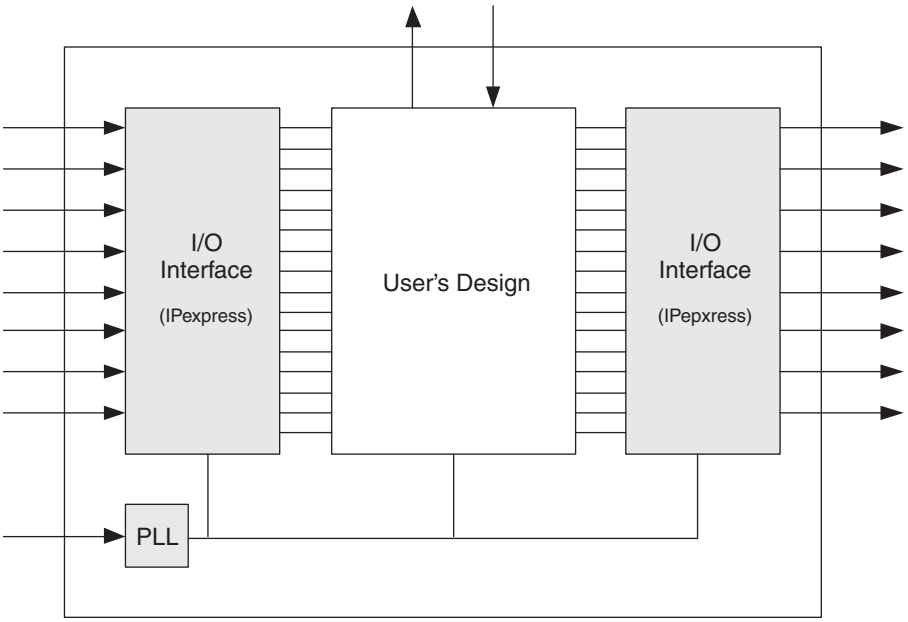
There are two methods for using I/O logic elements from the LatticeSC library: HDL inference and HDL instantiation.

HDL inference can be used for basic input and output ports that require basic I/O flip-flops. By default, the synthesis tool will implement an I/O flip-flop library element on any port that uses a flip-flop on the top level of the design. A basic I/O flip-flop library element (by default) is not programmed to take advantage of special features of the I/O logic such as edge clock routing and delay compensation.

For advanced features of the LatticeSC architecture I/O, logic special library elements will need to be instantiated in the design HDL. There are many different library elements to implement both SDR and DDR interfaces. To ease the burden on the user of selecting and instantiating these elements, the ispLEVER IPexpress™ tool can be used to create I/O interfaces. For detailed information on specific LatticeSC architecture library elements, see the ispLEVER Help system.

IPexpress can be used to create complex I/O interfaces in the LatticeSC architecture. In general, IPexpress should be the only required interface to create and instantiate any of the LatticeSC architecture I/O logic features. IPexpress supports user-customized interfaces for both SDR and DDR applications. The output of IPexpress for I/O logic interfaces is a user-defined HDL module to be instantiated in the user's HDL design. This customized module should contain all the necessary library elements, clocking connections and HDL parameters to implement the I/O interfaced design in IPexpress. Figure 27 provides a hierarchical view of an example HDL design including I/O interfaces produced by IPexpress.

Figure 27. HDL Example Hierarchy with IPexpress I/O Interfaces

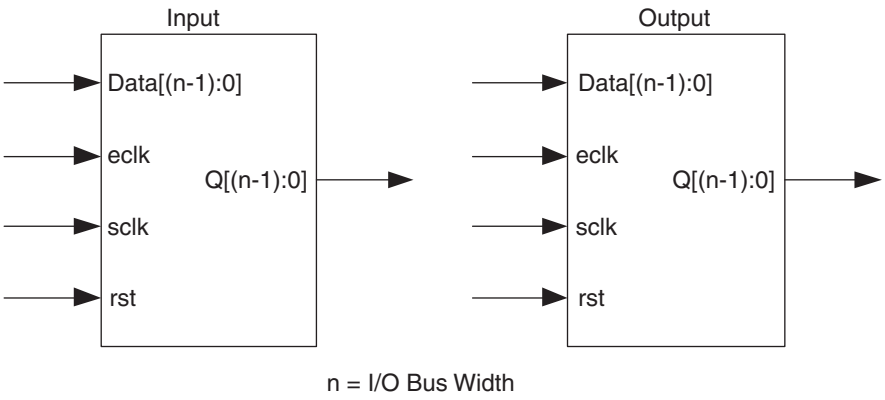


SDR

Single Data Rate (SDR) is a type of interface that is clocked on a single edge of a clock. The LatticeSC architecture supports SDR interfaces that can be clocked on the rising or falling edge of a clock. It is best to use the special SDR elements when special features of the LatticeSC architecture I/O logic will be required. Special requirements include the need for edge clock routing, data delay compensation or Mux/DeMux logic built into the I/O interface. Again, the basic inferred elements will not use edge clock routing or any special features. Therefore the SDR elements provide two clock inputs: one for the I/O data rate edge clock (eclk) and an additional sclk to clock data on the system clock network (typically on a FPGA primary clock route).

Figure 28 shows a simple SDR element. Further options are available in IPexpress to build on top of this element.

Figure 28. Basic Input and Output SDR Element (1x Mux/DeMux Gearing)



SDR Gearing: A gearing capability is provided to Mux/DeMux the I/O data rate (eclk) to the FPGA clock rate (sclk). Mux/DeMux gearing ratios are provided for 1x, 2x, and 4x. The ratio is oriented in terms of the sclk rate to the eclk rate. Table 2 provides an example to illustrate the Mux/DeMux gearing ratio.

Table 2. 8-Bit SDR Gearing Example

	Clock Frequency	Clock	Bus Width
I/O Data	200MHz on I/O	eclk	8 bits
1x Gearing	200MHz in FPGA	sclk	8 bits
2x Gearing	100MHz in FPGA	sclk	16 bits
4x Gearing	50MHz in FPGA	sclk	32 bits

For a data bus wider than a single bit a synchronous reset must be provided to the I/O logic block to maintain bus order. This reset is synchronous to the sclk of the I/O logic block.

When using the Mux/DeMux gearing feature the data of a single I/O port is serialized starting with bit 0. For example, a 4-bit data bus that is using 4x gearing will transmit bit 0 first, followed by bit 1, etc. For larger data buses that use an external data bus of greater than one bit, the serialization process will still transmit data bit 0 first on the lower end of the internal data bus. For example, a 32-bit data bus that is using 4x gearing requires an 8-bit external data bus. In this mode, bits [7:0] will be present on the first clock cycle, [15:8] on the next clock cycle, [23:16] on the next clock cycle, and finally [31:24] on the fourth clock cycle.

IPexpress: Creating an SDR interface is a function of the ispLEVER IPexpress tool. IPexpress allows the user to select from a variety of options to create a specific SDR bus for the user's application. Options for an SDR interface include interface direction, I/O bus width, FPGA Mux/DeMux gearing ratio, and options for data delay compensation. The output of IPexpress is an HDL module that can be instantiated in the user's design. This module contains all of the SDR elements, clock connections and HDL parameters required to implement the SDR interface.

Timing Diagrams: The following figures show example waveforms for input and output SDR interfaces. These examples both use a 2x Mux/DeMux gearing ratio with a 4-bit I/O interface width. A 4-bit I/O bus width and 2x gearing ratio will result in an 8-bit FPGA bus width at half the rate of eclk as shown in Figure 29.

Figure 29. Input SDR x2 Gearing Waveform

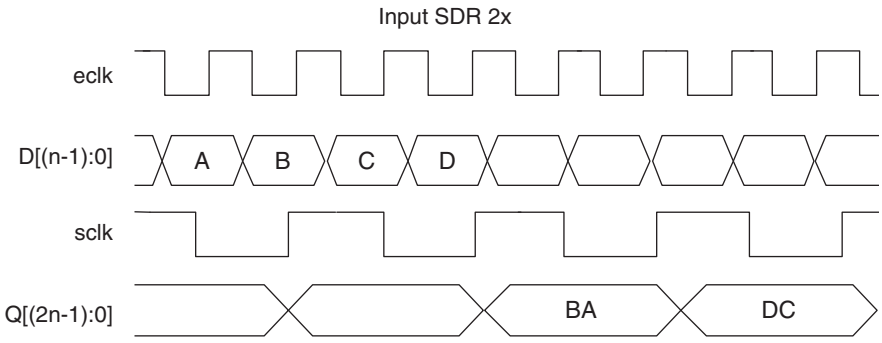
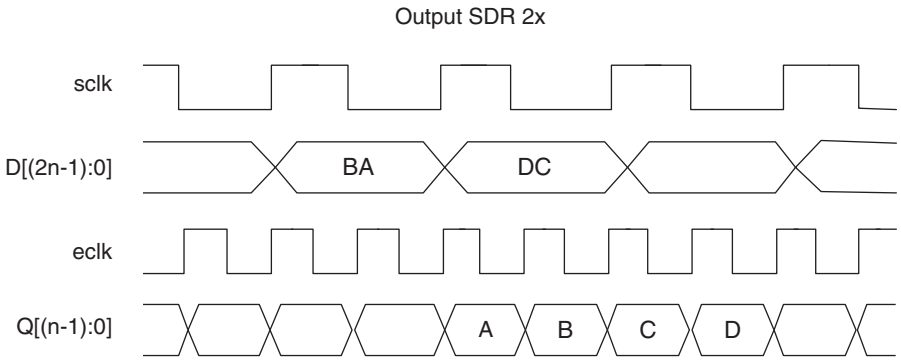


Figure 30 shows an output interface with an 8-bit FPGA bus width geared to a 4-bit I/O interface.

Figure 30. Output SDR x2 Gearing Waveform



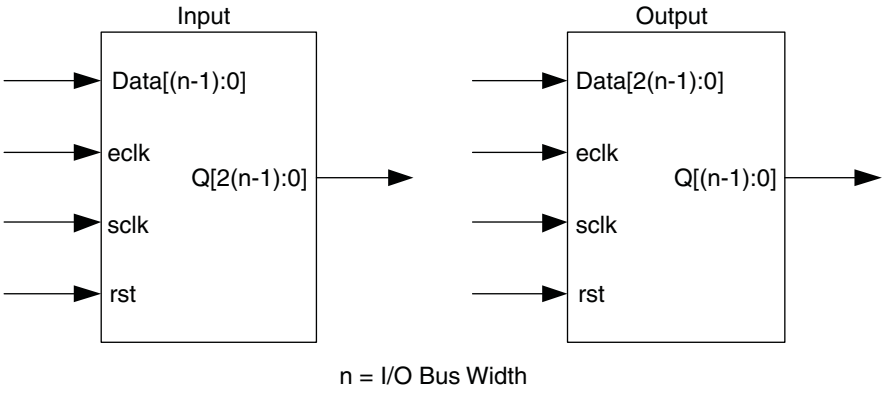
DDR

Double Data Rate (DDR) is a type of interface that is clocked on both the rising and falling edges of a clock. In the LatticeSC architecture, all DDR interfaces need to be instantiated with special DDR library elements. DDR interface library elements cannot be inferred by the synthesis tools. The basic mode of the input DDR library element will provide a single input from the I/O that is clocked on both edges of the clock with two outputs to the FPGA fabric that are clocked on only the rising edge of the clock. The LatticeSC architecture FPGA fabric supports only single data rate flip-flops and therefore it is not possible to use internal FPGA flip-flops on both edges of the clock. The basic mode of the output element will provide two inputs from the FPGA fabric both clocked on the rising edge of the clock and a single output to the I/O that is clocked on both edges of the clock.

The DDR library elements are designed for edge clock routing on the I/O side and primary clock routing on the FPGA side. To accommodate this clock domain transfer, the DDR elements provide two clock inputs. One for the I/O data rate edge clock (eclk) and another sclk port to clock data on the system clock network (typically on a FPGA primary clock route).

Figure 31 provides a basic DDR element. Further options are available to build on top of this basic element.

Figure 31. Basic Input and Output DDR Element (1x Mux/DeMux Gearing)



DDR Gearing – A gearing capability is provided to Mux/DeMux the I/O data rate (eclk) to the FPGA clock rate (sclk). Mux/DeMux gearing ratios are provided for 1x, 2x and 4x. For DDR interfaces this ratio is slightly different compared to the SDR ratio. A basic 1x DDR element provides two FPGA side bits for one I/O side bit at the same clock rate. When working with DDR gearing ratios the clock ratios are the same as SDR, but the data width is twice the ratio. Table 3 provides an example to illustrate the DDR Mux/DeMux gearing ratio.

Table 3. 8-Bit DDR Gearing Example

	Clock Frequency	Clock	Internal Bus Width
I/O Data	200MHz on I/O	eclk (both edges)	8 bits
1x Gearing	200MHz in FPGA	sclk (rising edge)	16 bits
2x Gearing	100MHz in FPGA	sclk (rising edge)	32 bits
4x Gearing	50MHz in FPGA	sclk (rising edge)	64 bits

For a data bus wider than a single bit, a synchronous reset must be provided to the I/O logic block to maintain bus order. This reset is synchronous to the sclk of the I/O logic block.

When using the Mux/DeMux gearing feature the data of a single I/O port is serialized starting with bit 0. For example, an 8-bit internal data bus that is using 4x gearing will transmit bit 0 first, followed by bit 1, etc. For larger data buses that use an external data bus of greater than one bit, the serialization process will still transmit data bit 0 first on the lower end of the internal data bus. For example, a 64-bit data bus that is using 4x gearing requires an 8-bit external data bus. In this mode bits [7:0] will be present on the first clock cycle, [15:8] a half cycle later, [23:16] on the next clock cycle, etc.

IPexpress – Creating a DDR interface is a function of the ispLEVER IPexpress tool. IPexpress allows the user to select from a variety of options to create a specific DDR bus for the user's application. Options for a DDR interface include interface direction, I/O bus width, FPGA Mux/DeMux gearing ratio, and options for data delay compensation. The output of IPexpress is an HDL module that can be instantiated in the user's design. This module contains all of the DDR elements, clock connections and HDL parameters required to implement the DDR interface.

Timing Diagrams – The following figures show example waveforms for input and output DDR interfaces. Both examples use a 2x Mux/DeMux gearing ratio with a 4-bit I/O interface width. A 4-bit I/O bus width and 2x gearing ratio will result in an 16-bit FPGA bus width at half the rate of eclk.

Figure 32. Input DDR x1 Gearing Waveform

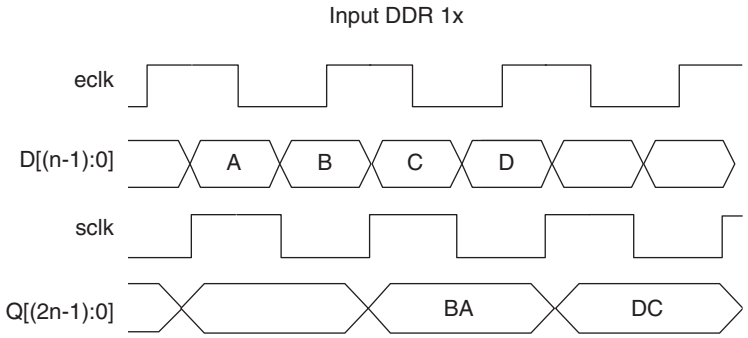


Figure 33. Output DDR x1 Gearing Waveform

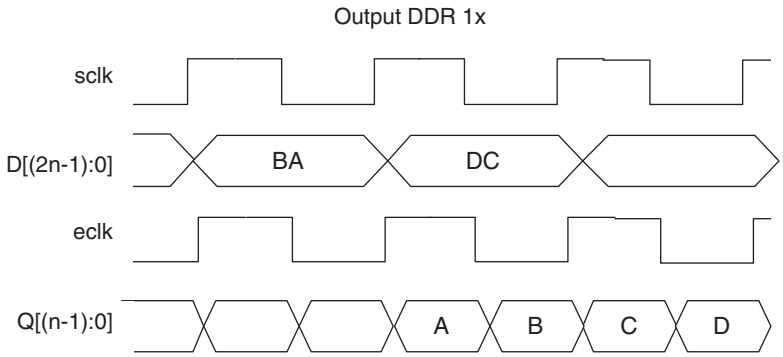


Figure 34. Input DDR x2 Gearing Waveform

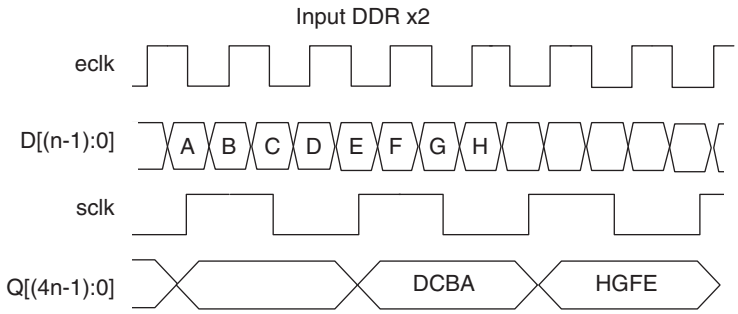
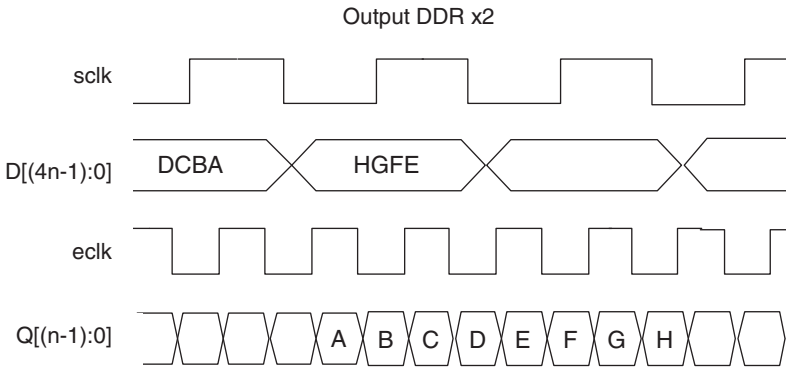


Figure 35. Output DDR x2 Gearing Waveform



I/O Architecture Rules

Table 4 shows the PIO usage for x1, x2, x4 gearing. The checkmarks in the columns show the specific PIOs that are used for each gearing mode. When using x2 or x4 gearing, any PIO which is not used for gearing can still be used as an output.

Table 4. Input/Output/Tristate Gearing Resource Rules

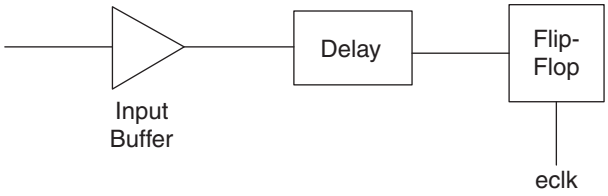
PIO	Input/Output Logic			Tristate	
	x1	x2	x4	x1	x2/x4
A	✓	✓	✓	✓	N/A
B	✓	✓	No I/O Logic	✓	N/A
C	✓	✓	✓	✓	N/A
D	✓	✓	No I/O Logic	✓	N/A

Note: Pin can still be used without I/O logic. Additional restriction applies when AIL (Adaptive Input Logic) is used. See Adaptive Input Logic (AIL) section for more information.

Input Delay Compensation

The I/O logic block provides the ability to delay an incoming signal from the pin. This is performed using a DELAY library element. Figure 36 shows the input data path. A DELAY element may also be used without the input flip-flop to delay a signal such as a clock.

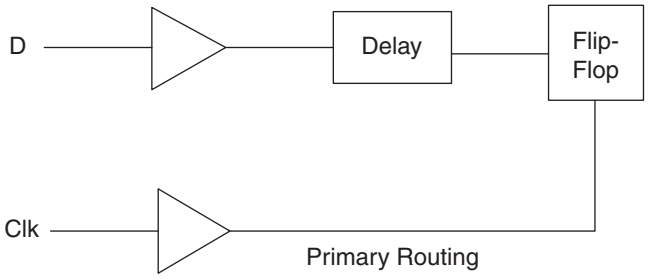
Figure 36. I/O Logic Delay



There are five methods for setting the amount of delay provided by the DELAY element. IPexpress provides delay options for the SDR and DDR interfaces.

Primary Clock Injection Delay Match – The primary clock injection delay match will provide DELAY to an incoming data signal to match the delay of the worst case primary clock injection delay to guarantee a zero hold time. Figure 37 provides a schematic view of the circuit topology for this delay. The primary clock injection delay is the latency of the clock signal from after the input buffer to the primary clock spine and to a latching flip-flop. This delay setting will allow a clock-to-data relationship at the I/O pins to be maintained inside of the device to the first flip-flop.

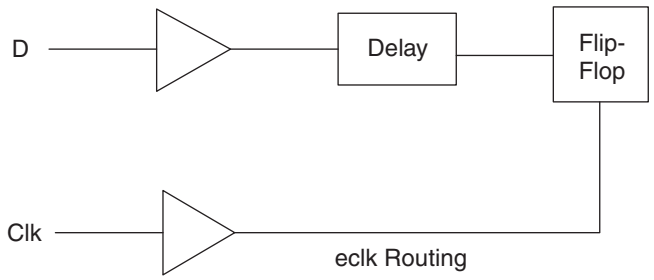
Figure 37. Primary Clock Injection Delay Match



This mode requires the data and clock input buffer to have the same I/O type. It also requires a preferred primary clock pin (discussed later) to be utilized for the input clock.

Edge Clock Injection Delay Match – The edge clock injection delay match will provide DELAY to an incoming data signal to match the delay of the worst case edge clock injection delay to guarantee a zero hold time. Figure 38 provides a schematic view of the circuit topology for this delay. The edge clock injection delay is the latency of the clock signal from after the input buffer to the edge clock spine and to a latching flip-flop. This delay setting will allow a clock-to-data relationship at the I/O pins to be maintained inside of the device to the first flip-flop.

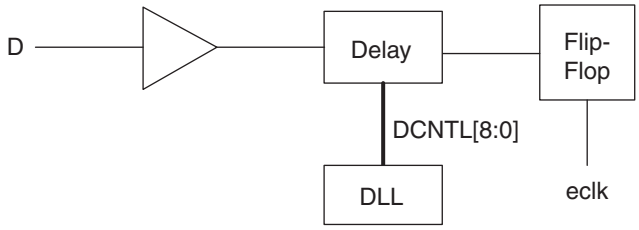
Figure 38. Edge Clock Injection Delay Match



This mode requires that the data and clock input buffer are of the same I/O type. It is also required that a preferred edge clock pin is utilized for the input clock.

DLL Controlled Delay – DLL controlled delay provides a digital vector to control the amount of delay that is added to the incoming signal. This mode is designed to work with the Time Reference Delay mode of the DLL to provide a 90 degree phase shift to a clock signal. In this mode the DELAY element uses a 9-bit DCNTL vector to control the amount of delay that is added. The Time Reference Delay DLL will set the DCNTL vector to the appropriate values to maintain a 90 degree phase shift across all environmental conditions. Figure 39 provides a schematic view input path for this delay.

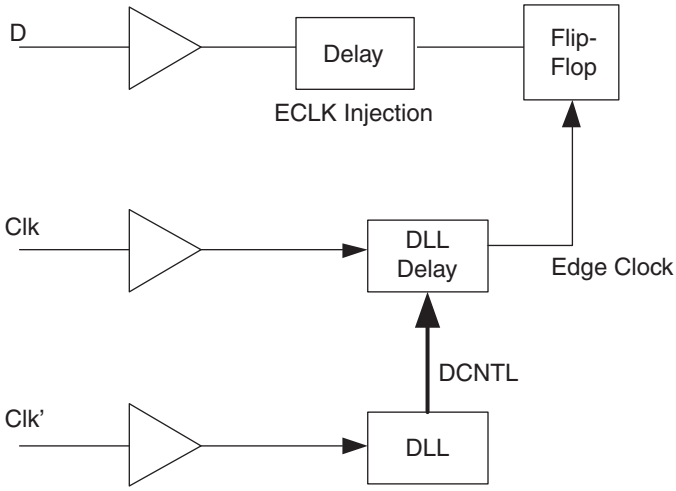
Figure 39. DLL Controlled Delay



When using a DLL to drive the DCNTL[8:0] bus there are two DCNTL busses available per bank. A single DCNTL bus can drive multiple DELAY elements across the bank.

Edge Clock Injection Delay Removal – ECLK injection delay mode is provided for clock-forwarded DDR interfaces when the clock and data arrive at the I/O edge aligned. Figure 40 provides a schematic view of the circuit topology for this delay mode. In this interface model, two types of DELAY elements are used. The input clock signal utilizes a DELAY block in DLL controlled delay mode. Here the input clock is delayed to provide a 90 degree phase shift. The data is then delayed using the ECLK injection delay mode in order to match the data and clock paths to the flip-flop. This ECLK injection delay will delay the data to ultimately provide a centered clock given the 90 degree phase shift. The clock signal into the DLL Clk (Clk' in Figure 40) can be any clock running at the same rate as the Clk signal. The DLL can either be driven from an external pin or inside the device.

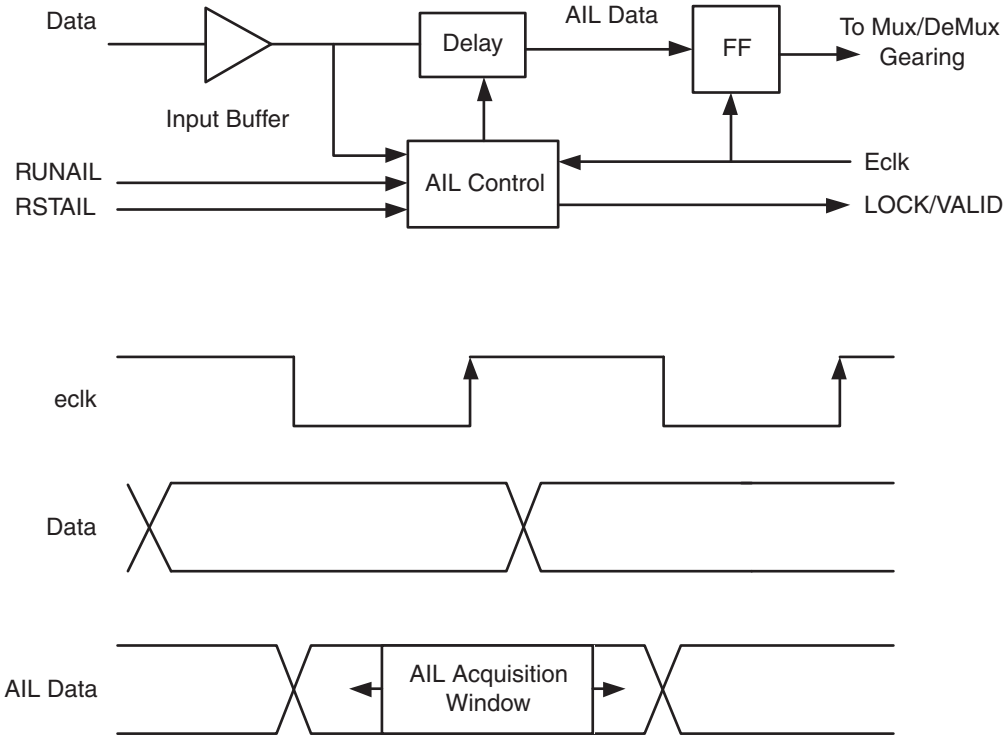
Figure 40. ECLK Injection Delay Circuit



Adaptive Input Logic (AIL)

The Adaptive Input Logic (AIL) is the ability of the I/O logic to automatically control the coarse and fine delays to guarantee setup and hold times for a single I/O. This is accomplished by continuously detecting the data transition point and delaying the data to move the sampling clock edge far enough away to correctly latch the data. The user is required to select a window size in which data transitions are not allowed. The AIL will continuously delay data to keep the window within the data pulse width away from the transition edges. Figure 41 provides a block diagram and waveform reference of the AIL feature provided in the I/O logic.

Figure 41. AIL Controlled Delay Block Diagram and Reference Waveform



The AIL features are not bus-based, but rather per input port. Since the AIL will slide the data to a particular clock edge, the bits of a bus may not be latched on the same clock edge. For this reason, a training pattern will be

required to realign the individual bits of the input data bus to the same clock edge. For single-bit instances this training pattern is not required. An AIL deskew reference design is available from Lattice. This reference design provides an example of the circuitry and training pattern required to align the bits of a bus.

The ispLEVER IPexpress tool for SDR and DDR interfaces provides options for using AIL when generating input interfaces. When used, the I/O interface will provide new ports for AIL control.

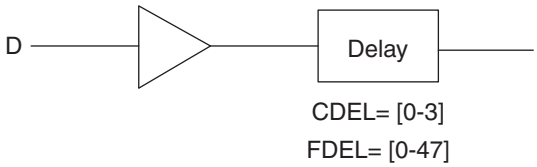
- **RUNAIL** – This active high async input signal allows the user to control when the AIL updates the delays.
- **RSTAIL** – This active high async input signal resets the AIL control to the middle of the delay chain.
- **LOCK** – This async signal is used to identify if the AIL window has stabilized on a fixed location or is moving across the delay chain.
- **VALID** – This async signal is used to identify the degree of movement in the delay chain.

Each PIC has one AIL circuit, associated with the delay block for primary true pad “A”. The data and control signals from pad “A” are input to the AIL circuit and the output from the circuit is made available to IOLogic “A”. It is also possible to select the data and control signals from alternate true pad “C” instead of those from pad “A”. In this case, the output from the circuit is made available to IOLogic “C”. If pad “A” is used as input to the AIL, then pad “C” is available for use as a general purpose input or output, there are no restrictions. However, if pad “C” is used as input to the AIL, pad “A” cannot be used as an input to the device and is only available for use as a general purpose output. This allows each PIC to be used as one input differential pair and one output differential par. No AIL support is available in I/O Bank 1. For more information regarding AIL, see TN1158 LatticeSC PURESPEED I/O Adaptive Input Logic User’s Guide.

User-Defined Delay

Static Delay – User-defined delay allows the user to statically set the amount of delay provided by the DELAY element. The DELAY element in this mode provides coarse delay (CDEL) and fine delay (FDEL) settings. The coarse and fine delay values are additive to create the total delay. Figure 42 provides a schematic view of the input data path.

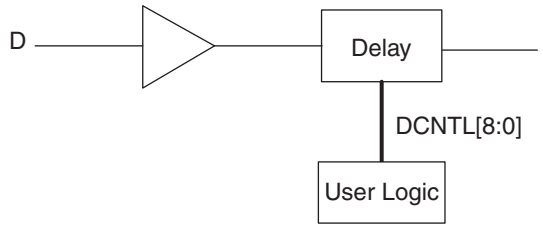
Figure 42. User-Defined Static Delay



The LatticeSC/M Family Data Sheet provides a range of delay values for each setting of CDEL and FDEL per speed grade.

Dynamic Delay – The user can also control the delay provided by the DELAY element through the DCNTL bus. Figure 43 provides a schematic view of user logic control of the DELAY. When creating an input interface for dynamic delay the user should select the DLL mode of operation of the DELAY. This will expose the DCNTL ports to connect to the user logic or a DLL.

Figure 43. User-Defined Dynamic Delay



The DCNTL bus is 9 bits wide and controls the CDEL and FDEL delays. Bits 8:7 select the CDEL value while bits 6:5 and 2:0 select the FDEL value. Bits 4:3 should be held at '0'. Table 5 provides a listing of possible DCNTL values and the resulting DELAY.

Table 5. DCNTL Bus User Logic Values

DCNTL[8:7]	CDEL
00	CDEL0
01	CDEL1
10	CDEL2
11	CDEL3

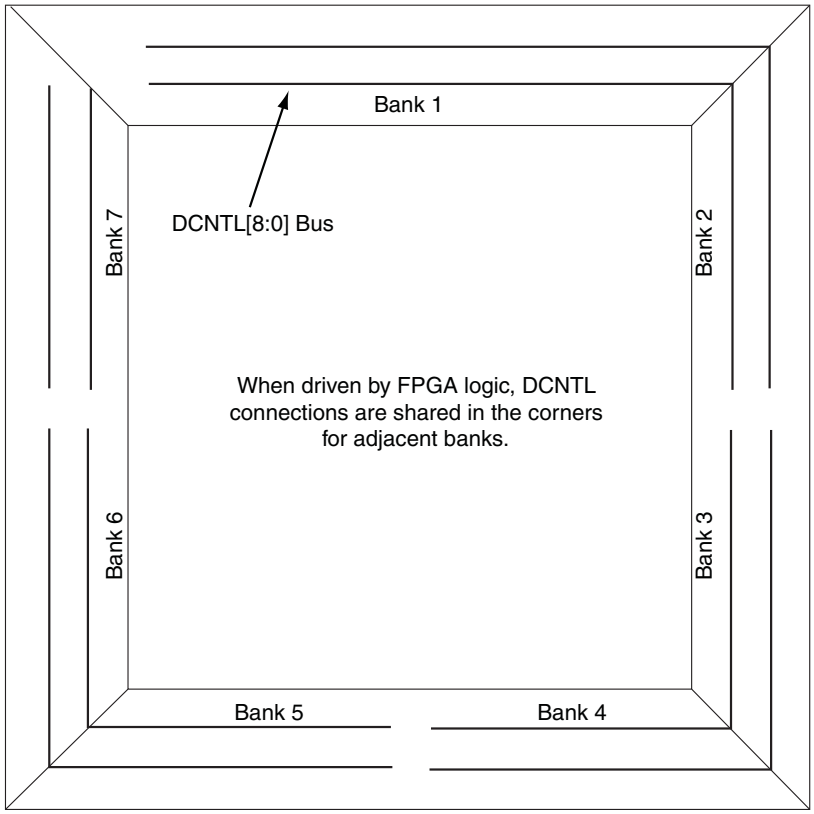
DCNTL[6:5][2:0]	FDEL
00000	FDEL0
00001	FDEL1
...	...
11111	FDEL31

As in the static mode the CDEL and FDEL value is additive to create the total delay of the DELAY element.

Note: The DELAY element behaves differently when controlled by a DLL compared to user logic. The simulation model of the DELAY element matches the DLL-controlled behavior of the DELAY element. Under user logic control the DELAY element simulation model delay will not match the hardware delay to the same degree as it does for the DLL control.

The routing of the DCNTL bus is limited to the I/O banks. Figure 44 provides a high-level view of the DCNTL bus connections.

Figure 44. DCNTL Bus Connections



As shown in Figure 44, the DCNTL bus is shared between adjacent banks in the corners. For example, there are two DCNTL busses in Bank 5 which are shared with the DCNTL busses in Bank 6.

For applications requiring dynamic delay of the DELAY element for more than two instances per I/O bank, as provided by the DCNTL bus, the AIL's delay control can be utilized. Specifically, the DCNTL bus can initialize the AIL to a desired delay value and the AIL can load the desired delay value into the DELAY block. To accomplish this capability, the following steps are required.

1. Include the AIL in the data path.
2. Tie the RUNAIL signal low so the AIL is not functioning.
3. Load the DCNTL bus with the desired CDEL and FDEL delay values.
4. Set the AIL RST to a high. This will load the desired CDEL and FDEL delay values from the DCNTL bus into the AIL.
5. Release the AIL RST to latch in the desired values from the AIL into the DELAY block.
6. Repeat steps 3, 4 and 5 for all required I/Os.

Note that this capability is only available on I/Os that support the AIL circuit as specified in the Adaptive Input Logic (AIL) section of this document.

Figure 45 provides a hardware representation of user logic control of the DELAY block utilizing the AIL.

Figure 45. Hardware Representation of User-Defined Dynamic Delay Utilizing the AIL

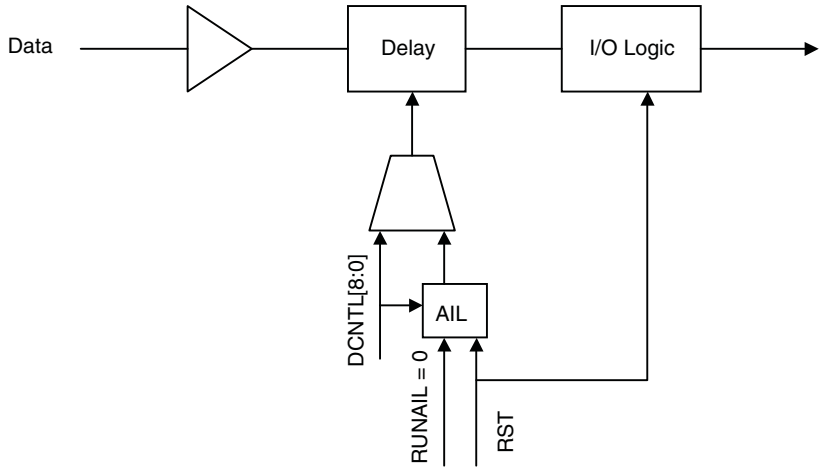
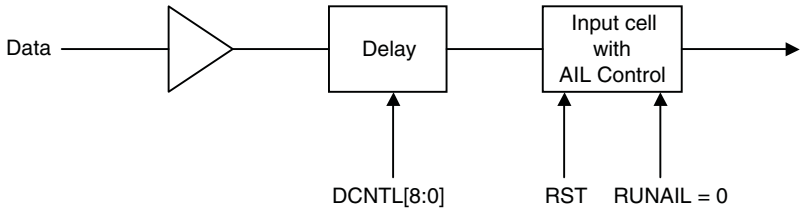


Figure 46 provides the software representation of user logic control of the DELAY block utilizing the AIL. Note that the software representation does not reflect the exact hardware representation, however the hardware will operate as expected.

Due to limitations, it is important to note that the simulation libraries do not model this behavior for this specific function. The DCTL bus directly changes the delay even if AIL is instantiated. The hardware will operate correctly as described.

Figure 46. Software Representation of User-Defined Dynamic Delay Utilizing the AIL



PLLs, DLLs and Clock Routing

The LatticeSC architecture provides PLLs, DLLs, phase matched Clock Dividers and dedicated clock routing to improve the performance of I/O interfaces.

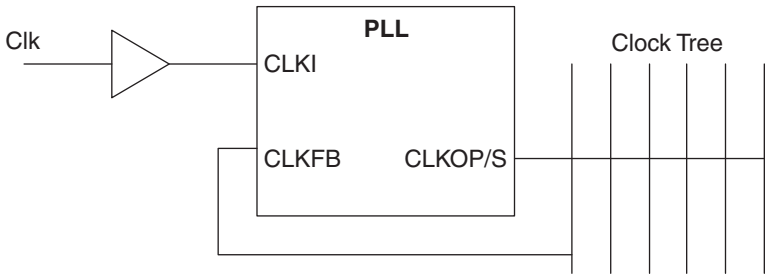
PLL

The LatticeSC architecture supports a PLL that can be utilized to fine tune clock timing for high-speed I/O interfaces. The LatticeSC PLL provides a single clock input (CLKI), two clock outputs (CLKOP and CLKOS), and a feedback clock input (CLKFB). For more information on the full capabilities of the LatticeSC PLL, refer to TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide. This section will cover the features of the LatticeSC PLL as designed for clock I/O tuning. The LatticeSC PLL can be created using IPexpress.

Clock Insertion Delay Removal

The LatticeSC PLL can be used to remove the clock insertion delay of a clock signal. Reducing clock insertion delay is useful in reducing clock-to-out latency. Figure 47 provides the configuration of the PLL to remove clock insertion delay. One of the output clocks is fed back into the CLKFB port. The delay of the PLL and clock route network is then removed by aligning the CLKFB clock with the CLKI input clock. Either of the two output clocks can be used in this mode. IPexpress allows the user to select the CLKFB clock to be either the CLKOP, CLKOS, or a user-defined clock.

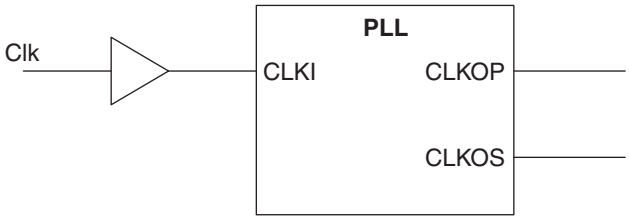
Figure 47. Clock Insertion Delay Removal PLL



Phase Adjustment

The LatticeSC PLL can be used to create a clock phase offset between the two outputs of the PLL. Figure 48 provides the PLL configuration for this mode.

Figure 48. Phase Adjustment PLL



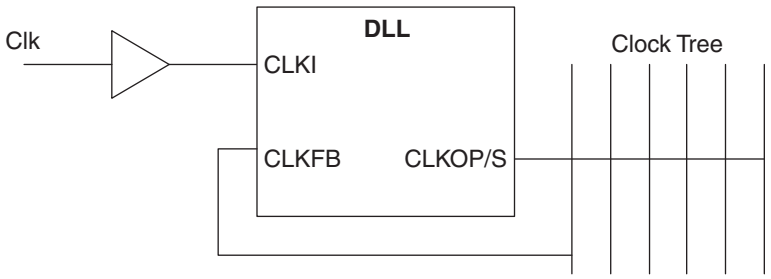
This mode is useful to provide a clock-to-data relationship for a forwarded clocking scheme. PLLs are useful in forward clock interfaces due to high stability and low jitter compared to a DLL.

DLL

The LatticeSC architecture provides a DLL that can be utilized to fine tune clock timing for high-speed I/O interfaces. The LatticeSC DLL is created using IPexpress which provides several different modes for working with clock delay. For more information on the full capabilities of the LatticeSC DLL, see TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide. This section will cover the features of the LatticeSC DLL as designed for clock I/O tuning.

Clock Insertion Delay Removal – The LatticeSC DLL can be used to remove the clock insertion delay of a clock signal. Reducing clock insertion delay is useful in reducing clock-to-out latency. Figure 49 provides the configuration of the DLL to remove clock insertion delay. One of the output clocks is fed back into the CLKFB port. The delay of the DLL and clock route network is then removed by aligning the CLKFB clock with the CLKI input clock. Either of the two output clocks can be used in this mode. IPexpress allows the user to select the CLKFB clock to be either the CLKOP, CLKOS, or a user-defined clock.

Figure 49. Clock Insertion Delay Removal DLL

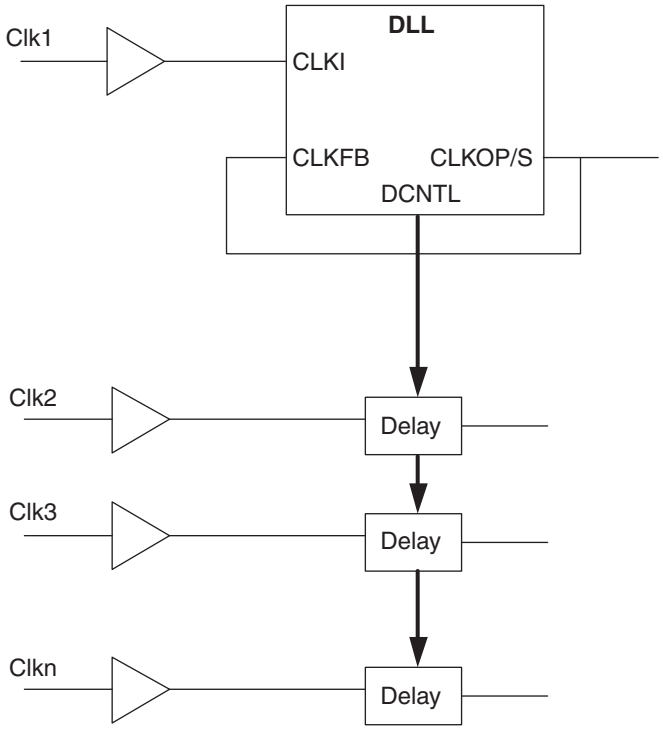


Providing the DCNTL Port

The DLL performs clock phase alignment by delaying a copy of the CLKI signal to the CLKOP/S output to match the CLKFB signal. The delay element used to do this in the DLL is identical to the delay element in the I/O used to

delay inputs to the device. This delay value is then propagated to the DCNTL bus and sent to the I/Os to delay other device input clocks by the same amount so that they are aligned to the output clock of the DLL. Figure 50 provides a schematic view of this circuit topology. When the DCNTL bus is used, the DLL only can only delay a clock signal by 3.3ns. Therefore, to cover an entire period of delay adjustment, the input clocks must be at least 300MHz. For many interfaces, only a 90 degree phase shift is needed. In this case, the low end of the DLL range of 100MHz can be used.

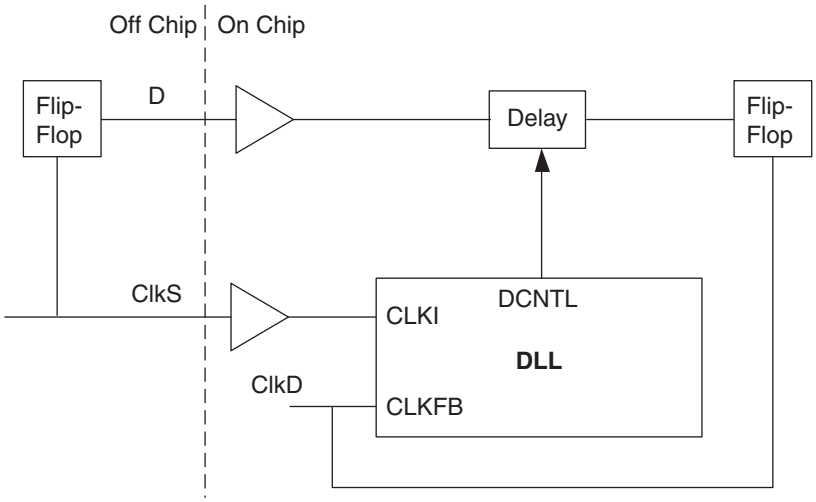
Figure 50. Clock Insertion Delay Removal via DCNTL Bus DLL Circuit



Clock Delay Match

The LatticeSC DLL provides a feature to measure the phase offset between two input clocks and produce this delay value as a DCNTL value. This feature is useful when transferring data between two frequency locked clocks of unknown phase. Figure 51 provides a schematic view of the circuit topology for this mode. In this example the D input is launched from ClkS and will be captured on ClkD. The DCNTL delay value that is produced will provide appropriate delay to the D input to produce the same clock-to-data relationship from ClkD to D that was present from ClkS to D on the input pins.

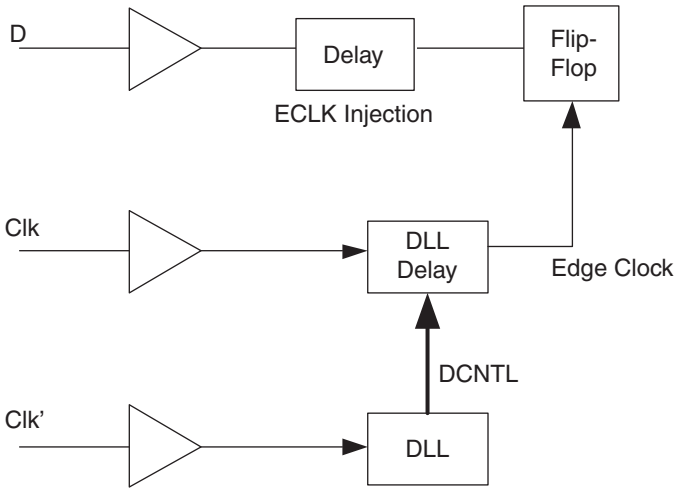
Figure 51. Clock Delay Match



Time Reference

The Time Reference mode of the LatticeSC DLL is designed to create a 90 degree phase shift only. The DCNTL value will provide a value across all environmental conditions that will create a 90 degree phase shift. Figure 52 provides a schematic view of the circuit topology for this mode. The ECLK Injection delay added to the data is required to match the clock injection time.

Figure 52. Time Reference



This DLL mode is useful in DDR interfaces where the clock-to-data relationship at the I/O pins are edge aligned. The delay on the DCNTL bus will create a 90 degree phase shift to provide valid setup and hold timing.

For applications requiring delays slightly less than or slightly greater than 90 degrees, the DCNTL 90 degree phase shift can be incremented or decremented by setting the DLL SMI offset 0x9 [7:0] register setting. This register setting will either add (DLL SMI offset 0x9 bit 7 = 0) or subtract (DLL SMI offset 0x9 bit 7 = 1) TFDEL delays from the default 90 degree phase shift. The number of TFDEL delays to be added or subtracted is specified by DLL SMI offset 0x9 bits [0:6] in the memory map section of TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide.

Note: The specification of TFDEL is located in the DC and Switching Characteristics section of the LatticeSC/M Family Data Sheet.

DLLs are useful in input interfaces due to fast tracking of the input clock phase. DLLs are also well suited for interfaces where the input clock may stop such as memory interfaces.

SMI Capable

The LatticeSC PLL and DLL also have the capability to connect to the SMI port of the system bus. This connection allows a subset of the PLL and DLL settings to be controlled by a master controller on the system bus such as a microprocessor or FPGA user logic. This feature provides the ability for a user to change the runtime behavior of the PLL and DLL through a dedicated memory map set of controls for the PLL. Features such as phase offset and clock dividers can be adjusted during runtime. See TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide, as well as TN1085, LatticeSC MPI/System Bus, for further information on this topic.

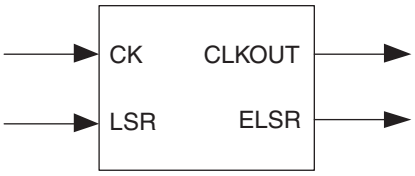
Preferred PLL/DLL Pins

There are preferred input pins for the inputs to the DLLs and PLLs. The preferred input pins provide the smallest clock insertion delay times and best signal integrity for high-speed input clocks. These pins are the best pins for the DLL and PLL inputs, but these pins are also general-purpose FPGA pins when not being used for clock inputs. This does not imply that these pins are the only input pins for the DLL and PLL, but simply the best pins (smallest clock insertion delay and best signal integrity). These pins support differential inputs as well as single-ended inputs. To find the location of these pins reference the LatticeSC/M Family Data Sheet pinout section. Under the function column are keywords such as DLL_FB, DLL_IN, PLL_FB, and PLL_IN. These correspond to the CLKFB and CLKI ports of the DLL and PLL. In addition to the PLL/DLL preferred pins any edge clock or primary clock preferred pin is also a good selection for a PLL/DLL input pin. Again, any pin can be used to source an input to the DLL or PLL, but for high-speed applications >400MHz I/O rate it is best to use these inputs for clock signal integrity and minimum insertion delay.

CLKDIV

The Lattice LatticeSC architecture provides a dedicated phase matched clock divider element (CLKDIV). This clock divider element supports dividing a clock by 2 or by 4. The clock dividers are used for providing the low-speed FPGA clocks for shift registers (x2, x4) and DDR (x2, x4) I/O logic interfaces. Both the divided and undivided clocks are matched in phase to facilitate transfers between them. Figure 53 shows the CLKDIV element.

Figure 53. CLKDIV Element



The LSR is an asynchronous reset to produce a synchronous edge reset (ELSR). The ELSR can use edge clock routing to provide a balanced reset to I/O logic elements.

There are four clock dividers on the top edge (Bank 1), four on the right side (Banks 2 and 3), four on the left side (Banks 6 and 7), and eight on the bottom (four in Bank 4 and four in Bank 5).

Clock dividers can be driven by edge clocks, primary clocks, secondary clocks, and any input pin. There are also preferred input pins to minimize clock insertion delay and signal integrity. The preferred pins for a CLKDIV are the same as the preferred pins for the primary and edge clocks to be discussed later. CLKDIV outputs can only drive primary clock networks since the undivided clock can drive the edge clock.

CLKDIV Input Connectivity

Each CLKDIV has a set of best connections for high-speed clocks (>400MHz). These best connections utilize edge clock routes or dedicated routes for clock divider inputs to maintain the best signal integrity and clock insertion delay. If the best connections are not used, the routing connectivity will be preserved, but will use routing which does not have the performance of the edge clock or dedicated routes. The following tables show these best con-

nections sorted on the source of the clock signal. The ispLEVER place and route tools should use this information and select the correct placement and routing connections for clock dividers. However, under some conditions the user will need to provide guidance to the place and route tools and locate a CLKDIV and its input pin to utilize the best connections. Physical components in ispLEVER are located by a site name. Below is an example ispLEVER preference to locate a CLKDIV:

```
LOCATE COMP "tx/ucdiv" SITE "CLKDIV2B";
```

Table 6. Top Edge (Bank 1) CLKDIV Best Connections

Pin	CLKDIV
PCLKT1_0	CLKDIV1A
PCLKT1_1	CLKDIV1B
PCLKT1_2	CLKDIV1C
PCLKT1_3	CLKDIV1D

Table 7. Right Edge (Banks 2 and 3) CLKDIV Best Connections

Pin	PLL CLKOS	PLL CLKOP	DLL CLKOS	DLL CLKOP	CLKDIV
PCLKT2_0	PLL_URCA	PLL_URCB			CLKDIV2A
	PLL_LRCA	PLL_LRCB			CLKDIV2A
PCLKT2_1	PLL_URCB		DLL_URCC		CLKDIV2B
	PLL_LRCB		DLL_LRCC		CLKDIV2B
PCLKT2_2			DLL_LRCD	DLL_URCC	CLKDIV2C
			DLL_URCD	DLL_LRCC	CLKDIV2C
PCLKT2_3		PLL_URCA	DLL_URCD	DLL_URCD	CLKDIV2D
		PLL_LRCA			CLKDIV2D

Table 8. Bottom Edge (Bank 4) CLKDIV Best Connections

Pin	PLL CLKOS	PLL CLKOP	DLL CLKOS	DLL CLKOP	CLKDIV
PCLKT4_0	PLL_LRCA	PLL_LRCB	DLL_LRCF	DLL_LRCE	CLKDIV4A
PCLKT4_1	PLL_LRCB		DLL_LRCC	DLL_LRCF	CLKDIV4B
PCLKT4_2			DLL_LRCD	DLL_LRCC	CLKDIV4C
PCLKT4_3		PLL_LRCA	DLL_LRCE	DLL_LRCD	CLKDIV4D

Table 9. Bottom Edge (Bank 5) CLKDIV Best Connections

Pin	PLL CLKOS	PLL CLKOP	DLL CLKOS	DLL CLKOP	CLKDIV
PCLKT5_0	PLL_LLCA	PLL_LLCB	DLL_LLCF	DLL_LLCE	CLKDIV5A
PCLKT5_1	PLL_LLCB		DLL_LLCC	DLL_LLCF	CLKDIV5B
PCLKT5_2			DLL_LLCD	DLL_LLCC	CLKDIV5C
PCLKT5_3		DLL_LLCA	DLL_LLCE	DLL_LLCD	CLKDIV5D

Table 10. Left Edge (Banks 6 and 7) CLKDIV Best Connections

Pin	PLL CLKOS	PLL CLKOP	DLL CLKOS	DLL CLKOP	CLKDIV
PCLKT7_0	PLL_ULCA	PLL_ULCB			CLKDIV7A
	PLL_LLCA	PLL_LLCB			CLKDIV7A
PCLKT7_1	PLL_ULCB		DLL_ULCC		CLKDIV7B
	PLL_LLCB		DLL_LLCC		CLKDIV7B
PCLKT7_2			DLL_ULCD	DLL_ULCC	CLKDIV7C
			DLL_LLCD	DLL_LLCC	CLKDIV7C
PCLKT7_3		PLL_LLCA		DLL_LLCD	CLKDIV7D
		PLL_ULCA		DLL_ULCD	CLKDIV7D

It is important in cases where I/O performance is >400MHz that the user understands the edge clock routing resources used, to ensure the high-performance connections are utilized.

CLKDIV Reset

A CLKDIV can also be used to create a balanced synchronous reset (ELSR) that can extend the entire I/O bank (both banks on the left and right side). This is especially useful for synchronously resetting the I/O logic when gearing is used to synchronize the entire data bus.

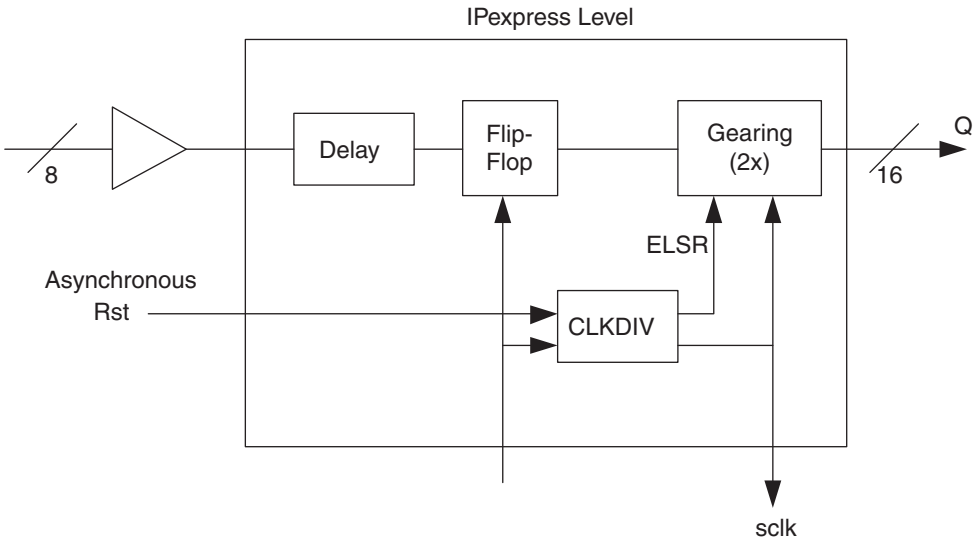
The reset input to the CLKDIV can be asynchronous to produce the synchronous reset to the I/O logic blocks. This reset will be synchronous to the sclk input to the I/O logic blocks. This is important to ensure transfers between the sclk and eclk occur such that the bit order of the geared bus is always consistent.

When a PLL or DLL feeds the CLKDIV, it is recommended to delay the reset release to the CLKDIV block until the PLL/DLL locks. This ensures proper eclk to sclk transfer in the IDDR and ODDR blocks.

IPexpress

IPexpress is used to generate I/O interface modules for SDR and DDR applications. When selecting an interface that will use the Mux/DeMux gearing logic a CLKDIV can optionally be included in the module created. When including a CLKDIV in the module generated by IPexpress, the sclk of the module will be the output of the CLKDIV. This clock is then used in the FPGA design to clock the data on the FPGA interface bus. Figure 54 provides the module generated for an 8-bit SDR bus using 2x gearing with an internal CLKDIV.

Figure 54. 8-bit x2 SDR Interface with Internal CLKDIV



In this example, the sclk will be half the rate of the eclk input since x2 gearing was selected. For example, a 200MHz input clock on eclk would produce a 100MHz clock sclk.

When Not Using a CLKDIV

If a dedicated CLKDIV is not used for the gearing operation of the I/O logic, then the SCLK and ECLK must both be sourced by the user. These clocks must be edge-aligned in order to guarantee valid transfers inside the I/O block. Edge-aligned clocks can be sourced from the PLL and DLL elements using the CLKOP and CLKOS outputs. Figure 55 provides the module generated when a CLKDIV is not used.

Figure 55. 8-bit SDR Interface Without a CLKDIV

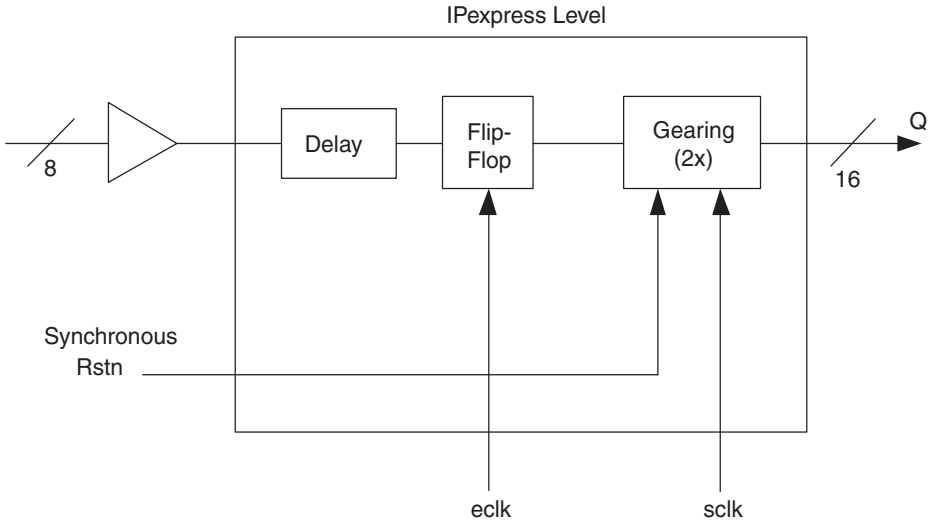
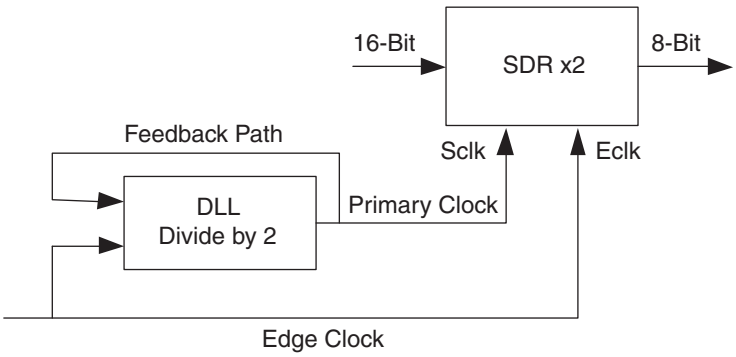


Figure 56 provides an example schematic view of the circuit topology when not using a CLKDIV. In this example, a DLL is used to source the eclk and sclk to an SDR module that was generated without a CLKDIV.

Figure 56. DLL Driving an SDR Module Without a CLKDIV



When not using a CLKDIV, a synchronous reset will also need to be provided by the user to synchronize the entire data bus. This reset needs to be synchronous to the low-speed sclk.

Clock Routing

There are three types of routing networks for clock signals in the LatticeSC architecture. Primary clocks are low skew, high fanout clock networks that provide clocks to all corners of the array. Secondary clocks are used for localized clocks that do not require a high fanout. Edge clocks are localized clocks specifically designed for registers on the I/O boundary. In most cases, the ispLEVER design tools will select the best type of routing for a given clock net.

However, it may be necessary in some designs to provide the ispLEVER design tools with guidance. At the very least, it will be helpful to know what type of clock routing is used to guarantee the best performance.

Primary Clocks

Primary clocks are used for the core clocks of the array. Primary clocks can also drive the I/O logic, but they do not offer the equivalent skew control or performance of the edge clocks. Primary clocks can be driven from a DLL, PLL, Physical Coding Sublayer (PCS), CLKDIV, DCS, and I/O pins. There are 48 primary clocks available on the LatticeSC architecture, 12 per quadrant.

There are preferred input pins for the inputs to primary clock networks. Preferred suggests that these pins are the best pins for the primary clock inputs, but these pins are also general-purpose FPGA pins. This does not imply that these pins are the only input pins for the primary clocks, but simply the lowest latency pins (smallest clock insertion delay). These pins support differential inputs as well as single-ended inputs. To find the location of these pins, reference the LatticeSC/M Family Data Sheet pinout section. Under the function column are keywords such as PCLKT7_1. In this example the pin PCLKT7_1 is the true side of a differential primary clock input in I/O bank 7. This pin can be used as the true side of a differential input or simply as a single-ended input. Primary clocks can use PCLKs 0, 1, 2 and 3, but not PCLKs 4, 5, 6 and 7. Each of the seven I/O banks have four primary clock inputs for a total of 28 on the device.

Edge Clocks

Edge clocks have been designed to be used for the I/O logic. Edge clocks are high-speed low skew clock routes that are located directly on the edge of the device. Edge clocks on the left and right sides of the device can span the entire side of the device. There are eight edge clocks on the left and eight edge clocks on the right side of the device. Edge clocks on the top and bottom can only span their respective bank. Edge clocks on the top and bottom provide eight clocks per bank. In total, there are 40 edge clocks per device. Edge clocks can also turn corners and connect banks to each other. There is a small penalty in the clock routing and skew when joining an edge clock route to another edge clock route. In general, it best to keep all pins on an edge clock in the same I/O bank. On the left and right sides of the device, the edge clock can span the entire side. In this case it is best to keep all pins on an edge clock on the same left or right side.

Edge clocks can be driven from a PLL, DLL, CLKDIV, preferred input pins, and even general routing. The inputs pins that can drive edge clocks are those found in the pinout section of the LatticeSC/M Family Data Sheet labeled PCLKT. These are the preferred inputs for edge clocks and primary clocks. Edge clocks can use any of the eight PCLKT pins per bank where as primary clocks can only use 0, 1, 2 and 3. Edge clock inputs can be single-ended or differential.

Edge clock routing can also be used for synchronous resets to synchronize I/O logic. If IPexpress is used to create an I/O interface with Mux/DeMux gearing the CLKDIV will drive the synchronous reset using edge clock routing.

Edge clocks are the only clock routes on the device that can support > 700MHz clocks. Therefore it is very important to understand when edge clocks are required and if they are being utilized correctly. In the ispLEVER place and route report the following warning message will appear if the I/O Logic block is not connected to an edge clock.

```
WARNING - par: Edge clock dllclk can`t find a dedicated branch or edge
               branch has been occupied, General routing has to be used to get
               on edge clock branch, and may suffer from excessive delay or skew.
```

There are two possible conditions in which this warning will be generated. The first is if there are not enough edge clocks left in the I/O bank to route the offending clock on an edge clock route. The second reason is that the pin location and driver location of the clock does not allow a connection to an edge clock.

Secondary Clocks

Secondary clocks are selected when primary or edge clocks cannot be used by the router. Secondary clocks are localized clock networks that do not have the tight skew control that primary and edge clocks provide for the entire device. Instead, they are low skew for a floating 6x6 array of PLCs. There is a slight increase in skew added beyond

each six rows and six columns of PLCs. Secondary clocks are on-demand clocks that are created from special high fanout buffered data routing.

Device Input Data Routing

Input data routing is the routing from the input buffer to the first flip-flop located in the I/O logic block. This routing includes the DELAY element. In a circuit that does not include the DELAY element the timing for this route is included in the input buffer delay and the input setup to the I/O logic flip-flop. In a circuit that includes the DELAY element the timing for this route is only the programmable delay that is added via the DELAY element. This is important for clarity when analyzing input I/O interfaces in the ispLEVER Trace report.

High-Level Descriptions

The following sections provide high-level descriptions of I/O interface circuits. Included in each example interface is a block schematic and ispLEVER Trace timing report to understand how the I/O timing information is provided. The module names, instance names, and net names in the schematic view match the names in the Trace report.

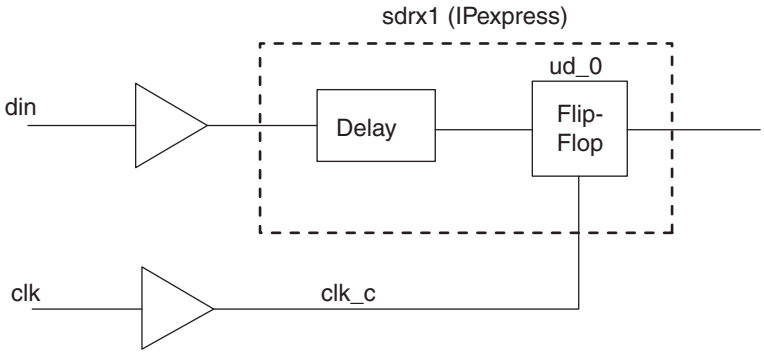
Note: These trace report segments have been captured from the ispLEVER 5.0 development tools. Slight format and timing values may change with later versions of the ispLEVER tool set.

Input Interfaces

Input Delay (Edge Clock Injection Delay Match)

This example interface uses the input delay compensation block in edge clock injection delay mode to preserve a clock-to-data relationship to the first flip-flop of the I/O logic. The I/O interface module created by IPexpress includes an SDR interface with edge clock injection delay compensation. Figure 57 provides a block level schematic for this interface.

Figure 57. ECLK Delay Compensation Example Circuit



To constrain the input setup for this I/O interface, the following preference can be used.

```
INPUT "din" SETUP 1 ns CLKPORT="clk";
```

The following Trace report segment provides the result of the input setup preference.

Figure 58. Trace Report Segment

```
Logical Details:  Cell type  Pin type       Cell name  (clock net +/-)

   Source:         Port       Pad            din
   Destination:    FF         Data in        sdrx1/ud_0  (to clk_c +)

   Max Data Path Delay:     1.246ns  (47.7% logic, 52.3% route), 1 logic levels.

   Min Clock Path Delay:    1.974ns  (28.2% logic, 71.8% route), 1 logic levels.

 Constraint Details:

      1.246ns delay din to din_IOLOGIC_S less
      1.000ns offset din to clk (totaling 0.246ns) meets
      1.974ns delay clk to din_IOLOGIC_S less
      1.024ns DIN_SET requirement (totaling 0.950ns) by 0.704ns

 Physical Path Details:

      Data path din to din_IOLOGIC_S:

   Name    Fanout   Delay (ns)          Site               Resource
IN_DEL      ---     0.594        F12.PAD to      F12.PADDI din
ROUTE         1     0.652      F12.PADDI to IOLOGICT24D.DI sdrx1/buf_Data0 (to clk_c)
                  --------
                    1.246   (47.7% logic, 52.3% route), 1 logic levels.

      Clock path clk to din_IOLOGIC_S:

   Name    Fanout   Delay (ns)          Site               Resource
IN_DEL      ---     0.556        A19.PAD to      A19.PADDI clk
ROUTE         3     1.418      A19.PADDI to OLOGICT24D.CLK clk_c
                  --------
```

The first section of this report segment provides the logical details of the interface. This section shows that the source is an input port (din) and the destination is a flip-flop (sdrx1/ud_0). These cell names match the names of the instances shown in Figure 57.

The next section provides the constraint details of how the input setup constraint is calculated. This section can be read as the input data delay of 1.246ns minus the constraint of 1ns is less than the 1.974ns clock delay minus the setup requirement of 1.024ns for the first flip-flop. The 1ns input setup constraint is met by 0.704ns, so the data can arrive 296ps before the clock.

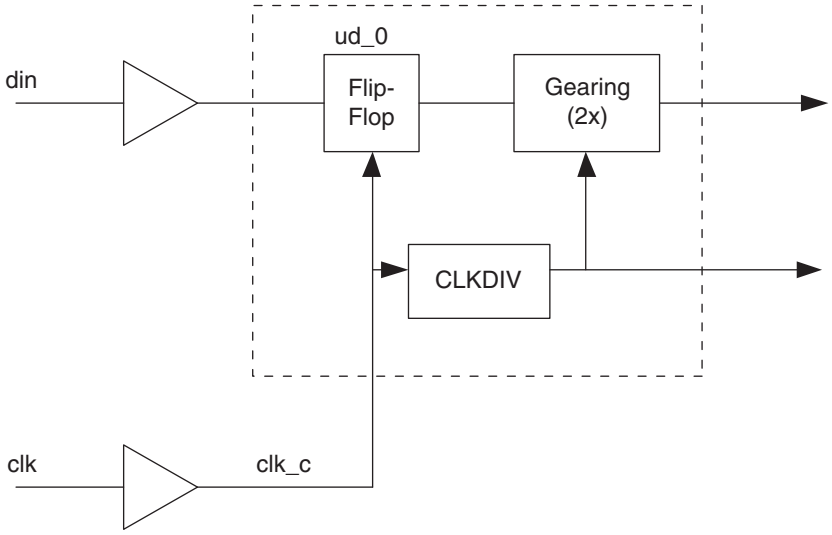
The final section covers the physical path details to show how the total data and clock paths are calculated. The data path for this circuit consists of an IN_DEL delay which is the input buffer delay and the input data routing. In this example the input data routing is 652ps which is the ECLK injection delay match value. The clock path for this circuit consists of the input buffer delay (IN_DEL) and the clock injection delay for the clock routing.

From this example it is shown that the ECLK injection delay compensation has balanced the clock and data input paths to a difference of 1.974ns - 1.246ns = 728ps.

Input DDR x2 Mux/DeMux with CLKDIV

This example interface uses an input DDR interface with x2 Mux/DeMux gearing logic and an internal CLKDIV. No delay compensation is used in this module. Figure 59 provides a block level schematic for this interface.

Figure 59. Input DDR x2 Mux/DeMux with CLKDIV



To constrain the input setup for this I/O interface the following preference can be used.

```
INPUT "din" SETUP 1 ns CLKPORT="clk";
```

This preference will cover both the rising and falling paths of the input. The FPGA array side of this interface is constrained with a FREQUENCY or PERIOD preference placed on the clock signal sourced from the sclk output of this module.

The following Trace report segment provides the result of the input setup preference.

Figure 60. Trace Report Segment

```
Logical Details:  Cell type  Pin type       Cell name  (clock net +/-)

   Source:          Port       Pad             din
   Destination:     FF         Data in         ddrx2/ud_0  (to clkin_c +)
                    FF                         ddrx2/ud_0

   Data Path Delay:     0.594ns  (100.0% logic, 0.0% route), 1 logic levels.

   Clock Path Delay:    1.476ns  (40.2% logic, 59.8% route), 1 logic levels.

 Constraint Details:

      0.594ns delay din to din_IOLOGIC less
      1.000ns offset din to clkin (totaling -0.406ns) meets
      1.476ns delay clkin to din_IOLOGIC less
      0.274ns DIN_SET requirement (totaling 1.202ns) by 1.608ns

 Physical Path Details:

      Data path din to din_IOLOGIC:

   Name     Fanout   Delay (ns)          Site               Resource
IN_DEL       ---     0.594       AF29.PAD to     AF29.PADDI din
ROUTE          1     0.000     AF29.PADDI to IOLOGICR57A.DI din_c (to clkin_c)
                   --------
                     0.594   (100.0% logic, 0.0% route), 1 logic levels.

      Clock path clkin to din_IOLOGIC:

   Name     Fanout   Delay (ns)          Site               Resource
IN_DEL       ---     0.594        K30.PAD to      K30.PADDI clkin
ROUTE          6     0.882      K30.PADDI to LOGICR57A.ECLK clkin c
```

The first section of this report segment provides the logical details of the interface. This section shows that the source is an input port (din) and the destination is a flip-flop (ddrx2/ud_0). These cell names match the names of the instances shown in Figure 59. Note that only a single setup time is calculated for both the rising clock path and the falling clock path.

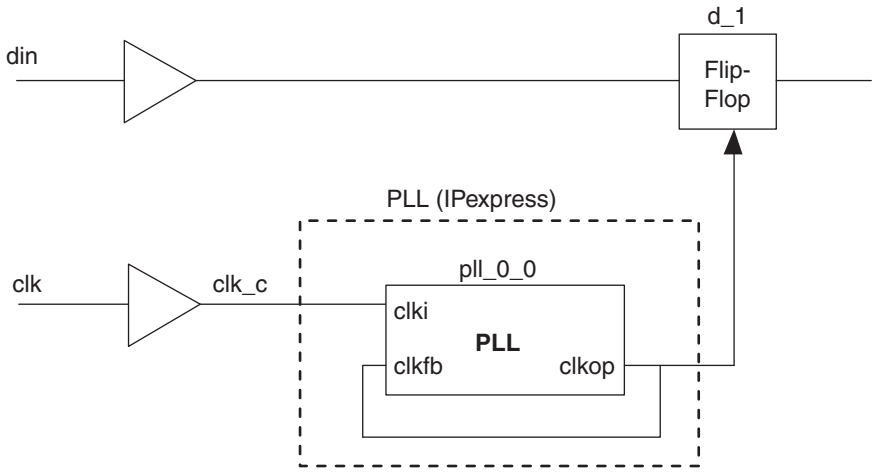
The next section provides the constraint details of how the input setup constraint is calculated. This section can be read as the input data delay of 0.594ns minus the constraint of 1ns is less than the 1.476ns clock delay minus the setup requirement of 0.274ns for the first flip-flop. The 1ns input setup constraint is met by 1.608ns so the data can arrive 608ps after the clock.

The final section covers the physical path details to show how the total data and clock paths are calculated. The data path for this circuit consists of an IN_DEL delay which is the input buffer delay and the input data routing. In this example the input data routing is 0ns because the DELAY element is not used in this circuit. The clock path for this circuit consists of the input buffer delay (IN_DEL) and the clock injection delay for the clock routing.

Input SDR Flip-Flop with PLL to Reduce Clock Insertion Delay

This example interface uses a PLL to reduce clock insertion delay. The data flip-flop in this example is a basic inferred flip-flop element. IPexpress was used to create the PLL configuration to remove clock insertion delay. Figure 61 provides a block level schematic for this interface. The CLKOP output of the PLL is used as the clock for the data flip-flop and is the feedback clock to remove the delay.

Figure 61. SDR Flip-Flop with PLL to Reduce Clock Insertion Delay



To constrain the input setup for this I/O interface the following preference can be used.

```
INPUT "din" SETUP 1 ns CLKPORT="clk";
```

The following Trace report segment provides the result of the input setup preference.

Figure 62. Trace Report Segment

```
Logical Details:  Cell type  Pin type       Cell name  (clock net +/-)

   Source:         Port       Pad            din
   Destination:    FF         Data in        d_1_0io  (to clkop +)

   Max Data Path Delay:     0.594ns  (100.0% logic, 0.0% route), 1 logic levels.

   Min Clock Path Delay:    5.260ns  (10.6% logic, 89.4% route), 2 logic levels.

 Constraint Details:

      0.594ns delay din to din_IOLOGIC_S less
      1.000ns offset din to clkin (totaling -0.406ns) meets
      5.260ns delay clkin to din_IOLOGIC_S less
      1.866ns feedback compensation less
     -0.069ns DIN_SET requirement (totaling 3.463ns) by 3.869ns

 Physical Path Details:

      Data path din to din_IOLOGIC_S:

   Name    Fanout   Delay (ns)          Site               Resource
IN_DEL      ---     0.594        H16.PAD to      H16.PADDI din
ROUTE         1     0.000      H16.PADDI to IOLOGICT35D.DI din_c (to clkop)
                  --------
                    0.594   (100.0% logic, 0.0% route), 1 logic levels.

      Clock path clkin to din_IOLOGIC_S:

   Name    Fanout   Delay (ns)          Site               Resource
IN_DEL      ---     0.556        F16.PAD to      F16.PADDI clkin
ROUTE         1     3.479      F16.PADDI to  PLL_LLCB.CLKI clkin_c
CLKOP_DEL   ---     0.000  PLL_LLCB.CLKI to PLL_LLCB.CLKOP pll/pll_0_0
ROUTE         4     1.225 PLL_LLCB.CLKOP to OLOGICT35D.CLK clkop
                  --------
                    5.260   (10.6% logic, 89.4% route), 2 logic levels.

PLL_LLCB.CLKOP attributes: CLKOP_MODE=VCO, CLKI_FDEL=0, CLKFB_FDEL=0

      Feedback path:

   Name    Fanout   Delay (ns)          Site               Resource
CLKOP_DEL   ---     0.000 PLL_LLCB.CLKFB to PLL_LLCB.CLKOP pll/pll_0_0
ROUTE         4     1.866 PLL_LLCB.CLKOP to PLL_LLCB.CLKFB clkop
                  --------
                    1.866   (0.0% logic, 100.0% route), 1 logic levels.

PLL_LLCB.CLKOP attributes: CLKOP_MODE=VCO, CLKI_FDEL=0, CLKFB_FDEL=0
```

The first section of this report segment provides the logical details of the interface. This section shows that the source is an input port (din) and the destination is a flip-flop (d_1_0io). In this example, the flip-flop is inferred by the synthesis tool and therefore the instance name of d_1_0io is not in Figure 61.

The next section provides the constraint details of how the input setup constraint is calculated. This section can be separated into two sections. The data section includes the input data delay of 0.594ns minus the constraint of 1ns which equals -0.406ns. The clock section includes the 5.260ns delay of the clock minus 1.866ns of feedback compensation and the flip-flop setup requirement of -0.069ns to equal 3.463ns. Now the input setup can be calculated via clock delay minus data delay (3.463ns - -4.06ns = 3.869ns). The 1ns input setup constraint is met by 3.869ns so the data can arrive 2.869ns after the clock.

The final section covers the physical path details to show how the total data and clock paths are calculated. When using a PLL with feedback an additional section is located in the physical path details for the feedback path. The

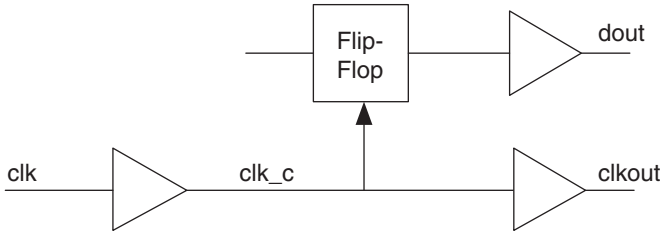
data path for this circuit consists of an IN_DEL delay which is the input buffer delay and the input data routing. In this example the input data routing is 0ns because the DELAY element is not used in this circuit. The clock path for this circuit consists of the input buffer delay (IN_DEL), a route from the input buffer to the PLL, the delay through the PLL, and the route from the PLL to a destination register. In this PLL configuration the delay through the PLL is 0ns. The feedback path consists of the delay through the PLL and the route from the output of the PLL to the CLKFB port of the PLL.

Output Interfaces

Basic Forwarded Clock

This example interface produces a data output with a forwarded clock. This example uses only basic inferred elements. Figure 63 provides a block level schematic for this interface.

Figure 63. Basic Forwarded Clock Circuit



To constrain the clock-to-data relationship for this I/O interface, the following preference can be used.

```
CLOCK_TO_OUT "dout" MAX 2 CLKPORT="clk" CLKOUT PORT "clkout";
```

The following Trace report segment provides the result of the CLOCK_TO_OUT preference.

Figure 64. Trace Report Segment

```
Logical Details:  Cell type  Pin type       Cell name  (clock net +/-)

   Source:         FF         Q              dout_0io  (from clkout_c +)
   Destination:    Port       Pad            dout

   Data Path Delay:     2.869ns  (100.0% logic, 0.0% route), 2 logic levels.

   Clock Path Delay:    1.612ns  (36.8% logic, 63.2% route), 1 logic levels.

 Constraint Details:

      1.612ns delay clk to dout_IOLOGIC_S and
      2.869ns delay dout_IOLOGIC_S to dout less
      4.193ns delay clk to clkout (totaling 0.288ns) meets
      2.000ns offset clk to dout by 1.712ns

 Physical Path Details:

      Clock path clkin to dout_IOLOGIC_S:

   Name    Fanout   Delay (ns)          Site               Resource
IN_DEL      ---     0.594        E29.PAD to      E29.PADDI clk
ROUTE         4     1.018      E29.PADDI to OLOGICR22B.CLK clkout_c
                  --------
                    1.612   (36.8% logic, 63.2% route), 1 logic levels.

      Data path dout_IOLOGIC_S to dout:

   Name    Fanout   Delay (ns)          Site               Resource
CK_DEL      ---     0.541 OLOGICR22B.CLK to OGICR22B.IOLDO dout_IOLOGIC_S (from clkout_c)
ROUTE         1     0.000 OGICR22B.IOLDO to      E30.IOLDO dout_c
OP0PAD_DEL  ---     2.328      E30.IOLDO to        E30.PAD dout
                  --------
                    2.869   (100.0% logic, 0.0% route), 2 logic levels.

      Clock out path:

   Name    Fanout   Delay (ns)          Site               Resource
IN_DEL      ---     0.594        E29.PAD to      E29.PADDI clk
ROUTE         4     1.009      E29.PADDI to      H28.PADDO clkout_c
DOPAD DEL   ---     2.590      H28.PADDO to        H28.PAD clkout
```

The first section of this report segment provides the logical details of the interface. This section shows that the source is a flip-flop (dout_0io) and the destination an output port (dout). In this example the flip-flop is inferred by the synthesis tool and therefore the instance name of dout_0io is not in Figure 63.

The next section provides the constraint details of how the CLOCK_TO_OUT preference is calculated. This calculation is basically a comparison of the data path versus the clock path. The data path includes the clock delay of 1.612ns plus the data delay of 2.869ns for a total of 4.481ns. The clock path from the clk input to the clk output is 4.193ns. The difference of 288ps meets the MAX constraint of 2ns.

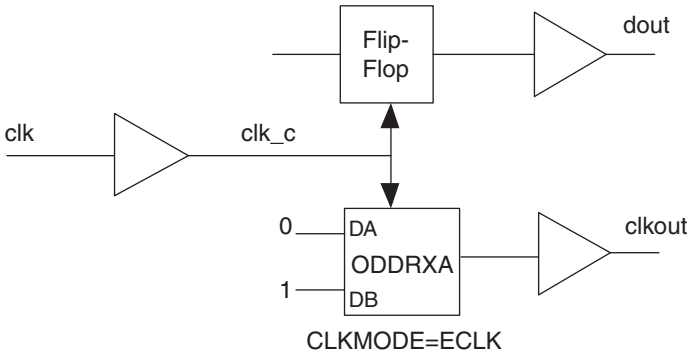
The final section covers the physical path details to show how the three components of this timing interface are created. First the clock path from input clock to data flip-flop is provided. This includes the input buffer delay (IN_DEL) and the route from the input buffer to the data flip-flop. Second the data delay is provided. This path includes the clock delay of the flip-flop, the route from the flip-flop to the output buffer, and the delay of the output buffer (OPOPAD_DEL). The data output uses a flip-flop located in the I/O logic block. The route from the flip-flop to the output buffer is internal to this block and therefore the routing delay is shown as 0ns. Last, the clock output path is provided which includes the input buffer for the clk (IN_DEL), the route from the input buffer to the output buffer, and the output buffer delay (DOPAD_DEL).

Note: An inversion of the output clock will not change the Trace timing calculations.

Strict Edge Aligned Forwarded Clock

In order to create an output clock that is edge-aligned with a data bus the same delay for the data must be matched with the clock. This example interface produces a data output with a forwarded clock that is delay matched. This example uses a basic inferred output flip-flop and an instantiated ODDRXA library element. Figure 65 provides a block-level schematic for this interface. This circuit uses a DDR output element to create a clock by connecting the inputs to logic 1 and logic 0.

Figure 65. DDR Element to Create Delay Matched Forwarded Clock



CLKMODE

An HDL parameter is available for the user to select the type of clock connection for the ODDRXA element. This parameter allows the user to select between primary and edge clock routing. In Figure 65 the ODDRXA element is set to use edge clock routing to match the data flip-flop clock routing. Table 11 shows the possible values for the CLKMODE parameter.

Table 11. CLKMODE Parameter

CLKMODE	Description
CLK (default)	The element will be clocked from primary clock routing.
ECLK	The element will be clocked from edge clock routing.

Figure 66 shows HDL examples for applying the CLKMODE parameter in both VHDL and Verilog.

Figure 66. HDL Examples of CLKMODE Parameter

```
Verilog using ECLK
ODDRXA c (.DA(1'b0), .DB(1'b1), .CLK(clk), .RST(1'b0), .Q(clkout))
/* synthesis clkmode="eclk" */;

VHDL using clk
attribute CLKMODE : string;
attribute CLKMODE of c : label is "clk";

begin

ODDRXA c port map (DA=>'0', DB=>'1', CLK=>clk, RST=>'0', Q=>clkout);
```

Polarity of the Forwarded Clock

The clock-to-data relationship from dout to clkout in Figure 65 is dependent on how the data inputs are connected on the ODDRXA element. In this example the DA port is connected to '0' and the DB port is connected to '1'. This

connection scheme maintains the phase relationship of the clock and data. In the reverse order, the clock-to-data relationship will be inverted.

Note: The Trace report will not show any difference between the connectivity of the clocks to the ODDRXA element. The inversion must be taken into account by the user.

To constrain the clock-to-data relationship for this I/O interface, the following preference can be used.

```
CLOCK_TO_OUT "dout" MAX 2 CLKPORT="clk" CLKOUT PORT "clkout";
```

The following Trace report segment provides the result of the CLOCK_TO_OUT preference.

Figure 67. Trace Report Segment

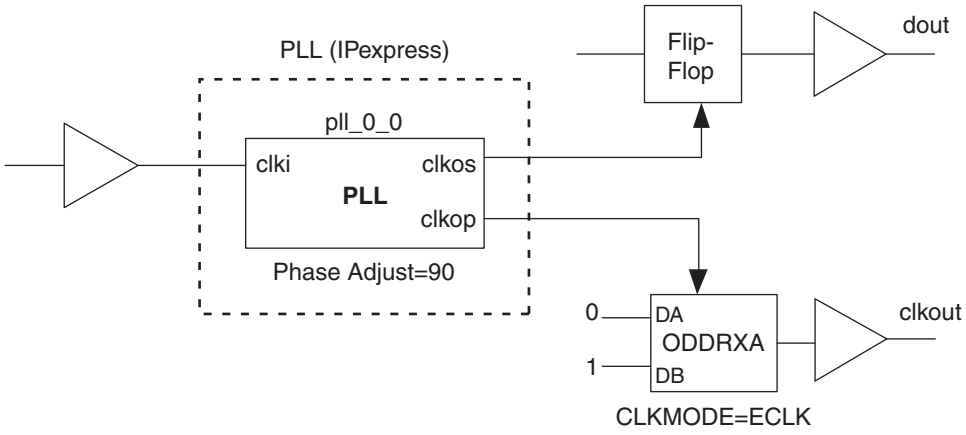
```
Physical Path Details:

      Clock path clkin to dout_IOLOGIC_S:

   Name     Fanout   Delay (ns)          Site               Resource
IN_DEL       ---     0.594         AH8.PAD to      AH8.PADDI clkin
ROUTE          4     1.148       AH8.PADDI to OLOGICB15A.CLK clkin_c
                    --------
                      1.742   (34.1% logic, 65.9% route), 1 logic levels.

      Data path dout_IOLOGIC_S to dout:

   Name     Fanout   Delay (ns)          Site               Resource
CK_DEL       ---     0.541 OLOGICB15A.CLK to OGICB15A.IOLDO dout_IOLOGIC_S (from clkin_c)
ROUTE          1     0.000 OGICB15A.IOLDO to      AH7.IOLDO dout_c
OP0PAD_DEL   ---     2.328       AH7.IOLDO to        AH7.PAD dout
                    --------
                      2.869   (100.0% logic, 0.0% route), 2 logic levels.

      Clock out path:

   Name     Fanout   Delay (ns)          Site               Resource
IN_DEL       ---     0.556         AH8.PAD to      AH8.PADDI clkin
ROUTE          4     0.833       AH8.PADDI to OLOGICB15C.CLK clkin_c
CK_DEL       ---     0.428 OLOGICB15C.CLK to OGICB15C.IOLDO clkout_IOLOGIC_S
ROUTE          1     0.000 OGICB15C.IOLDO to     AE12.IOLDO clkout_c
OP0PAD_DEL   ---     2.328      AE12.IOLDO to       AE12.PAD clkout
                    --------
                      4.145   (79.9% logic, 20.1% route), 3 logic levels.
```

This report segment provides only the physical path details for this I/O interface. Three components make up the total timing analysis. First is the clock delay to the data output flip-flop which totals 1.742ns. Second, the data delay is provided which totals 2.869ns. Last, the clock output path is provided which includes the input buffer for the clk (IN_DEL), the route from the input buffer to the output logic element, the route from the output logic to the output buffer, and the output buffer delay (OPOPAD_DEL). The total clock output path delay is 4.145ns. To compare the data and the clock output path, the total of the data path is the input clock delay (1.742ns) plus the data output delay (2.869ns) which is 4.611ns. The clock-to-data offset is then 466ps, which meets the max constraint of 2ns.

Fixed Phase Offset on Forwarded Clock

This example will utilize a PLL to create a fixed phase relationship for a forwarded clock-to-data. This example uses a basic inferred output flip-flop, an instantiated ODDRXA library element and a PLL configuration created by IPexpress. Figure 68 provides a block level schematic for this interface. This circuit uses both outputs of the PLL. The CLKOP output is connected to the data flip-flop. The CLKOS output is connected to the forwarded clock ODDRXA element. By creating a fixed relationship between CLKOP and CLKOS, this relationship is maintained at the output pins for the clock and data.

Figure 68. PLL Phase Offset Circuit



In this example, a 90 degree phase offset is selected between CLKOS and CLKOP.

To constrain the clock-to-data relationship for this I/O interface, the following preference can be used.

```
CLOCK_TO_OUT "dout" MAX 3 CLKPORT="clk" CLKOUT PORT "clkout";
```

The following Trace report segment provides the result of the CLOCK_TO_OUT preference.

Figure 69. Trace Report Segment

```
Clock path clkin to dout_IOLOGIC_S:

   Name     Fanout   Delay (ns)          Site               Resource
IN_DEL       ---     0.594        AC6.PAD to      AC6.PADDI clkin
ROUTE          1     0.369      AC6.PADDI to  PLL_LLCB.CLKI clkin_c
CLKOP_DEL    ---     0.000  PLL_LLCB.CLKI to PLL_LLCB.CLKOP pll/pll_0_0
ROUTE          3     1.536 PLL_LLCB.CLKOP to OLOGICL42D.CLK clkop
                   --------
                     2.499   (23.8% logic, 76.2% route), 2 logic levels.

PLL_LLCB.CLKOP attributes: CLKOP_MODE=VCO, CLKI_FDEL=0, CLKFB_FDEL=0

      Data path dout_IOLOGIC_S to dout:

   Name     Fanout   Delay (ns)          Site               Resource
CK_DEL       ---     0.541 OLOGICL42D.CLK to OGICL42D.IOLDO dout_IOLOGIC_S (from clkop)
ROUTE          1     0.000 OGICL42D.IOLDO to       V6.IOLDO dout_c
OP0PAD_DEL   ---     2.328       V6.IOLDO to         V6.PAD dout
                   --------
                     2.869   (100.0% logic, 0.0% route), 2 logic levels.

      Clock out path:

   Name     Fanout   Delay (ns)          Site               Resource
IN_DEL       ---     0.556        AC6.PAD to      AC6.PADDI clkin
ROUTE          1     0.358      AC6.PADDI to  PLL_LLCB.CLKI clkin_c
CLKOS_DEL    ---     2.500  PLL_LLCB.CLKI to PLL_LLCB.CLKOS pll/pll_0_0
ROUTE          1     1.225 PLL_LLCB.CLKOS to OLOGICL34D.CLK clkos
CK_DEL       ---     0.428 OLOGICL34D.CLK to OGICL34D.IOLDO clkout_IOLOGIC_S
ROUTE          1     0.000 OGICL34D.IOLDO to       R4.IOLDO clkout_c
OP0PAD_DEL   ---     2.328       R4.IOLDO to         R4.PAD clkout
                   --------
                     7.395   (78.6% logic, 21.4% route), 4 logic levels.
```

This report segment provides only the physical path details for this I/O interface. Three components make up the total timing analysis.

First is the input clock delay to the data output flip-flop. The path includes the CLKI to CLKOP to the data flip-flop. Second, the data delay is provided which totals 2.869ns. Last, the clock output path is provided which includes the input buffer for the clk (IN_DEL), the route from the input buffer to the PLL, the delay of the PLL (CLKOS_DEL), the route from the PLL to the output element, the clock delay of the output element (CK_DEL), the route from the output logic to the output buffer, and the output buffer delay (OPOPAD_DEL). The total clock output path delay is 7.395ns.

Notice that the PLL delay for CLKOS is 2.5ns and for CLKOP it is 0ns. This is the phase offset value that was selected. The FREQUENCY preference for the clk input was specified to be 100MHz (10ns) and 90 degrees of 100MHz is 2.5ns.

To compare the data and the clock output path, the total of the data path is the input clock delay (2.499ns) plus the data output delay (2.869ns) which is 5.368ns. The clock-to-data offset is now 2.027ns.

References

- LatticeSC/M Family Data Sheet
- TN1085, LatticeSC MPI/System Bus
- TN1098, LatticeSC sysCLOCK PLL/DLL User's Guide
- TN1158 LatticeSC PURESPEED I/O Adaptive Input Logic User's Guide

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
February 2006	01.0	Initial release.
October 2006	01.1	Updated headings in CLKDIV tables.
January 2007	01.2	Added section describing method to use the AIL's delay control to provide more user-defined delay capability per I/O bank.
February 2007	01.3	Updated DLL Driving an SDR Module Without a CLKDIV figure.
April 2007	01.4	Updated AIL Controlled Delay Block Diagram and Reference Waveform" diagram and Adaptive Input Logic (AIL) text section.
May 2007	01.5	Modified text in I/O Logic Modes section.
		Added I/O Architecture Rules section and Input/Output Gearing Resources Rules table.
July 2007	01.6	Expanded discussion of weak internal pull-ups in the Bus Maintenance Circuit text section.
November 2007	01.7	Added DCNTL Bus Connections figure and corresponding text.
		Modified text in DLL Controlled Delay section.
April 2008	01.8	Updated information about using PWRSAVE.
August 2008	01.9	Added footnote regarding LVPECL inputs to PURESPEED I/O Standards table.
		Updated Input Legal Combinations table and related text.
August 2008	02.0	Updated Input DDR x1 Gearing Waveform diagram.
		Updated Input DDR x2 Gearing Waveform diagram.
October 2008	02.1	Added new Vref data for HSTL 18_III and footnote 5 to PURESPEED I/O Standards table.
		Added information for LVPECL33 inputs to I/O Placement Rules.
		Added References section.
August 2009	02.2	Replaced DIFFRESISTOR text section and added corresponding Differential Termination Details figure.
October 2009	02.3	Updated Dynamic Delay text section.
March 2010	02.4	Updated Input/Output/Tristate Gearing Resource Rules table data and footnote.
May 2011	02.5	Updated information on user logic control of the DELAY element.
April 2013	02.6	Updated document with new corporate logo.