Virtex-4 Libraries Guide for HDL Designs

UG619 (v14.7) October 2, 2013

XILINX®

Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.

Chapter 1

Introduction

This HDL guide is part of the ISE® documentation collection. A separate version of this guide is available if you prefer to work with schematics.

This guide contains the following:

- Introduction.
- A list of design elements supported in this architecture, organized by functional categories.
- Descriptions of each available primitive.

About Design Elements

This version of the Libraries Guide describes the primitives that comprise the Xilinx Unified Libraries for Virtex®-4 devices, and includes examples of instantiation code for each element. Instantiation templates are also supplied in a separate ZIP file, which you can find in your installation directory under ISE/doc/usenglish/isehelp.

Primitives are Xilinx components that are native to the FPGA you are targeting. If you instantiate a primitive in your design, after the translation process you will end up with the exact same component in the back end. For example, if you instantiate the Virtex-5 element known as ISERDES_NODELAY as a user primitive, after you run translate (ngdbuild) you will end up with an ISERDES_NODELAY in the back end as well. If you were using ISERDES in a Virtex-5 device, then this will automatically retarget to an ISERDES_NODELAY for Virtex-5 in the back end. Hence, this concept of a "primitive" differs from other uses of that term in this technology.

Xilinx maintains software libraries with hundreds of functional design elements (UniMacros and primitives) for different device architectures. New functional elements are assembled with each release of development system software. This guide is one in a series of architecture-specific Libraries Guides describing functional design elements.

Design Entry Methods

For each design element in this guide, Xilinx evaluates four options for using the design element, and recommends what we believe is the best solution for you. The four options are:

- **Instantiation -** This component can be instantiated directly into the design. This method is useful if you want to control the exact placement of the individual blocks.
- **Inference -** This component can be inferred by most supported synthesis tools. You should use this method if you want to have complete flexibility and portability of the code to multiple architectures. Inference also gives the tools the ability to optimize for performance, area, or power, as specified by the user to the synthesis tool.
- **Coregen & Wizards -** This component can be used through CORE Generator or other Wizards. You should use this method if you want to build large blocks of any FPGA primitive that cannot be inferred. When using this flow, you will have to re-generate your cores for each architecture that you are targeting.
- **Macro Support -** This component has a UniMacro that can be used. These components are in the UniMacro library in the Xilinx tool, and are used to instantiate primitives that are too complex to instantiate by just using the primitives. The synthesis tools will automatically expand UniMacros to their underlying primitives.

Chapter 2

Functional Categories

This section categorizes, by function, the circuit design elements described in detail later in this guide. The elements (*primitives* and *macros*) are listed in alphanumeric order under each functional category.

Advanced

Design Element	Description
EMAC	Primitive: Fully integrated 10/100/1000 Mb/s Ethernet Media Access Controller (Ethernet MAC)

Arithmetic Functions

Design Element	Description
DSP48	Primitive: 18x18 Signed Multiplier Followed by a Three-Input Adder with Optional Pipeline Registers

Clock Components

Design Element	Description
BUFG	Primitive: Global Clock Buffer
BUFGCE	Primitive: Global Clock Buffer with Clock Enable
BUFGCE_1	Primitive: Global Clock Buffer with Clock Enable and Output State 1
BUFGCTRL	Primitive: Global Clock MUX Buffer
BUFGMUX_VIRTEX4	Primitive: Global Clock MUX Buffer
BUFIO	Primitive: Local Clock Buffer for I/O
BUFR	Primitive: Regional Clock Buffer for I/O and Logic Resources

Design Element	Description
DCM_ADV	Primitive: Advanced Digital Clock Manager Circuit
DCM_BASE	Primitive: Base Digital Clock Manager Circuit
DCM_PS	Primitive: Digital Clock Manager with Basic and Phase Shift Features
PMCD	Primitive: Phase-Matched Clock Divider

Config/BSCAN Components

Design Element	Description
BSCAN_VIRTEX4	Primitive: Virtex®-4 JTAG Boundary-Scan Logic Access Circuit
CAPTURE_VIRTEX4	Primitive: Virtex®-4 Boundary Scan Logic Control Circuit
FRAME_ECC_VIRTEX4	Primitive: Reads a Single, Virtex®-4 Configuration Frame and Computes a Hamming, Single-Error Correction, Double-Error Detection Syndrome
ICAP_VIRTEX4	Primitive: Virtex-4 Internal Configuration Access Port
STARTUP_VIRTEX4	Primitive: Virtex®-4 User Interface to Configuration Clock, Global Reset, Global 3-State Controls, and Other Configuration Signals
USR_ACCESS_VIRTEX4	Primitive: 32-Bit Register with a 32-Bit DATA Bus and a DATAVALID Port

Gigabit I/O

Design Element	Description
GT11_CUSTOM	Primitive: RocketIO MGTs with 622 Mb/s to 11.1 Gb/s Data Rates, 8 to 24 Transceivers per FPGA, and 2.5 GHz 5.55 GHz VCO, Less Than 1ns RMS Jitter
GT11_DUAL	Primitive: RocketIO MGT Tile (contains 2 GT11_CUSTOM) with 622 Mb/s to 11.1 Gb/s data rates, 8 to 24 transceivers per FPGA, and 2.5 GHz 5.55 GHz VCO, less than 1ns RMS jitter
GT11CLK	Primitive: A MUX That Can Select Fom Differential Package Input Clock, refclk From the Fabric, or rxbclk to Drive the Two Vertical Reference Clock Buses for the Column of MGTs
GT11CLK_MGT	Primitive: Allows Differential Package Input to Drive the Two Vertical Reference Clock Buses for the Column of MGTs

I/O Components

Design Element	Description
DCIRESET	Primitive: DCI State Machine Reset (After Configuration Has Been Completed)
IBUF	Primitive: Input Buffer
IBUFDS	Primitive: Differential Signaling Input Buffer
IBUFG	Primitive: Dedicated Input Clock Buffer
IBUFGDS	Primitive: Differential Signaling Dedicated Input Clock Buffer and Optional Delay
IDELAY	Primitive: Input Delay Element
IDELAYCTRL	Primitive: IDELAY Tap Delay Value Control
IOBUF	Primitive: Bi-Directional Buffer
IOBUFDS	Primitive: 3-State Differential Signaling I/O Buffer with Active Low Output Enable
ISERDES	Primitive: Dedicated I/O Buffer Input Deserializer
KEEPER	Primitive: KEEPER Symbol
OBUF	Primitive: Output Buffer
OBUFDS	Primitive: Differential Signaling Output Buffer
OBUFT	Primitive: 3-State Output Buffer with Active Low Output Enable
OBUFTDS	Primitive: 3-State Output Buffer with Differential Signaling, Active-Low Output Enable
OSERDES	Primitive: Dedicated IOB Output Serializer
PULLDOWN	Primitive: Resistor to GND for Input Pads, Open-Drain, and 3-State Outputs
PULLUP	Primitive: Resistor to VCC for Input PADs, Open-Drain, and 3-State Outputs

Processors

Design Element	Description
PPC405_ADV	Primitive: Primitive for the Power PC Core

RAM/ROM

Design Element	Description
FIFO16	Primitive: Virtex-4 Block RAM Based, Built-In FIFO
RAM16X1D	Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM
RAM16X1D_1	Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM with Negative-Edge Clock
RAM16X1S	Primitive: 16-Deep by 1-Wide Static Synchronous RAM
RAM16X1S_1	Primitive: 16-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock

Design Element	Description
RAM16X2S	Primitive: 16-Deep by 2-Wide Static Synchronous RAM
RAM32X1S	Primitive: 32-Deep by 1-Wide Static Synchronous RAM
RAM32X1S_1	Primitive: 32-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock
RAM32X2S	Primitive: 32-Deep by 2-Wide Static Synchronous RAM
RAM64X1S	Primitive: 64-Deep by 1-Wide Static Synchronous RAM
RAM64X1S_1	Primitive: 64-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock
RAMB16	Primitive: 16K-bit Data and 2K-bit Parity Single-Port Synchronous Block RAM with Configurable Port Widths
RAMB32_S64_ECC	Primitive: 512 Deep by 64-Bit Wide Synchronous, Two-Port Block RAM with Built-In Error Correction
ROM128X1	Primitive: 128-Deep by 1-Wide ROM
ROM16X1	Primitive: 16-Deep by 1-Wide ROM
ROM256X1	Primitive: 256-Deep by 1-Wide ROM
ROM32X1	Primitive: 32-Deep by 1-Wide ROM
ROM64X1	Primitive: 64-Deep by 1-Wide ROM

Registers/Latches

Design Element	Description
FDCE	Primitive: D Flip-Flop with Clock Enable and Asynchronous Clear
FDCPE	Primitive: D Flip-Flop with Clock Enable and Asynchronous Preset and Clear
FDPE	Primitive: D Flip-Flop with Clock Enable and Asynchronous Preset
FDRE	Primitive: D Flip-Flop with Clock Enable and Synchronous Reset
FDRSE	Primitive: D Flip-Flop with Synchronous Reset and Set and Clock Enable
FDSE	Primitive: D Flip-Flop with Clock Enable and Synchronous Set
IDDR	Primitive: Input Dual Data-Rate Register
LDCE	Primitive: Transparent Data Latch with Asynchronous Clear and Gate Enable
LDCPE	Primitive: Transparent Data Latch with Asynchronous Clear and Preset and Gate Enable
LDPE	Primitive: Transparent Data Latch with Asynchronous Preset and Gate Enable
ODDR	Primitive: Dedicated Dual Data Rate (DDR) Output Register

Shift Register LUT

Design Element	Description
SRL16	Primitive: 16-Bit Shift Register Look-Up Table (LUT)
SRL16_1	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Negative-Edge Clock
SRL16E	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Clock Enable
SRL16E_1	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Negative-Edge Clock and Clock Enable
SRLC16	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry
SRLC16_1	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry and Negative-Edge Clock
SRLC16E	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry and Clock Enable
SRLC16E_1	Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry, Negative-Edge Clock, and Clock Enable

Slice/CLB Primitives

Design Element	Description
BUFCF	Primitive: Fast Connect Buffer
LUT1	Primitive: 1-Bit Look-Up Table with General Output
LUT1_D	Primitive: 1-Bit Look-Up Table with Dual Output
LUT1_L	Primitive: 1-Bit Look-Up Table with Local Output
LUT2	Primitive: 2-Bit Look-Up Table with General Output
LUT2_D	Primitive: 2-Bit Look-Up Table with Dual Output
LUT2_L	Primitive: 2-Bit Look-Up Table with Local Output
LUT3	Primitive: 3-Bit Look-Up Table with General Output
LUT3_D	Primitive: 3-Bit Look-Up Table with Dual Output
LUT3_L	Primitive: 3-Bit Look-Up Table with Local Output
LUT4	Primitive: 4-Bit Look-Up-Table with General Output
LUT4_D	Primitive: 4-Bit Look-Up Table with Dual Output
LUT4_L	Primitive: 4-Bit Look-Up Table with Local Output
MULT_AND	Primitive: Fast Multiplier AND
MUXCY	Primitive: 2-to-1 Multiplexer for Carry Logic with General Output
MUXCY_D	Primitive: 2-to-1 Multiplexer for Carry Logic with Dual Output
MUXCY_L	Primitive: 2-to-1 Multiplexer for Carry Logic with Local Output
MUXF5	Primitive: 2-to-1 Look-Up Table Multiplexer with General Output
MUXF5_D	Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output

Design Element	Description
MUXF5_L	Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output
MUXF6	Primitive: 2-to-1 Look-Up Table Multiplexer with General Output
MUXF6_D	Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output
MUXF6_L	Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output
MUXF7	Primitive: 2-to-1 Look-Up Table Multiplexer with General Output
MUXF7_D	Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output
MUXF7_L	Primitive: 2-to-1 look-up table Multiplexer with Local Output
MUXF8	Primitive: 2-to-1 Look-Up Table Multiplexer with General Output
MUXF8_D	Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output
MUXF8_L	Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output
XORCY	Primitive: XOR for Carry Logic with General Output
XORCY_D	Primitive: XOR for Carry Logic with Dual Output
XORCY_L	Primitive: XOR for Carry Logic with Local Output

Chapter 3

About Design Elements

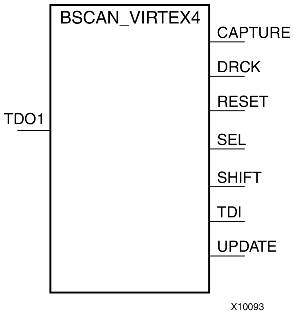
This section describes the design elements that can be used with Virtex®-4 devices. The design elements are organized alphabetically.

The following information is provided for each design element, where applicable:

- Name of element
- Brief description
- Schematic symbol (if any)
- Logic table (if any)
- Port descriptions
- Design Entry Method
- Available attributes (if any)
- Example instantiation code
- For more information

BSCAN_VIRTEX4

Primitive: Virtex®-4 JTAG Boundary-Scan Logic Access Circuit



Introduction

This design element allows access to and from internal logic by the JTAG Boundary Scan logic controller. This allows for communication between the internal running design and the dedicated JTAG pins of the FPGA.

Each instance of this design element will handle one JTAG USER instruction (USER1 through USER4) as set with the JTAG_CHAIN attribute. To handle all four USER instuctions, instantiate four of these elements and set the JTAG_CHAIN attribute appropriately.

Note For specific information on boundary scan for an architecture, see the Programmable Logic Data Sheet for this element.

Port Descriptions

Port	Direction	Width	Function
CAPTURE	Output	1	Active upon the loading of the USER instruction. Asserts High when the JTAG TAP controller is in the CAPTURE-DR state.
DRCK	Output	1	A mirror of the TCK input pin to the FPGA when the JTAG USER instruction assigned by JTAG_CHAIN is loaded and the JTAG TAP controller is in the SHIFT-DR state or in the CAPTURE-DR state.
RESET	Output	1	Active upon the loading of the USER instruction. It asserts High when the JTAG TAP controller is in the TEST-LOGIC-RESET state.
SEL	Output	1	Indicates when the USER instruction has been loaded into the JTAG Instruction Register. Becomes active in the UPDATE-IR state, and stays active until a new instruction is loaded.
SHIFT	Output	1	Active upon the loading of the USER instruction. It asserts High when the JTAG TAP controller is in the SHIFT-DR state.
TDI	Output	1	A mirror of the TDI pin.
UPDATE	Output	1	Active upon the loading of the USER instruction. It asserts High when the JTAG TAP controller is in the UPDATE-DR state.
TDO	Input	1	Active upon the loading of the USER instruction. External JTAG TDO pin will reflect data input to the macro's TDO1 pin.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
JTAG_CHAIN	Integer	1, 2, 3, 4	1	Sets the JTAG USER instruction number that this instance of the element will handle.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BSCAN_VIRTEX4: Boundary Scan primitve for connecting internal logic to
--                JTAG interface.
--                Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BSCAN_VIRTEX4_inst : BSCAN_VIRTEX4
generic map (
   JTAG_CHAIN => 1) -- Value to set BSCAN site of device. Possible values: (1,2,3 or 4)
port map (
   CAPTURE => CAPTURE, -- CAPTURE output from TAP controller
   DRCK => DRCK,       -- Data register output for USER functions
   RESET => RESET,     -- Reset output from TAP controller
   SEL => SEL,         -- USER active output
   SHIFT => SHIFT,     -- SHIFT output from TAP controller
   TDI => TDI,         -- TDI output from TAP controller
   UPDATE => UPDATE,   -- UPDATE output from TAP controller
   TDO => TDO          -- Data input for USER function
);

-- End of BSCAN_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

```
// BSCAN_VIRTEX4: Boundary Scan primitive for connecting internal logic to
//                JTAG interface.
//                Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BSCAN_VIRTEX4 #(
   .JTAG_CHAIN(1)      // Possible values: 1, 2, 3, or 4
) BSCAN_VIRTEX4_inst (
   .CAPTURE(CAPTURE), // CAPTURE output from TAP controller
   .DRCK(DRCK),       // Data register output for USER function
   .RESET(RESET),     // Reset output from TAP controller
   .SEL(SEL),         // USER active output
   .SHIFT(SHIFT),     // SHIFT output from TAP controller
   .TDI(TDI),         // TDI output from TAP controller
   .UPDATE(UPDATE),   // UPDATE output from TAP controller
   .TDO(TDO)          // Data input for USER function
);
```

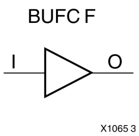
```
// End of BSCAN_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFCF

Primitive: Fast Connect Buffer



Introduction

This design element is a single fast connect buffer used to connect the outputs of the LUTs and some dedicated logic directly to the input of another LUT. Using this buffer implies CLB packing. No more than four LUTs may be connected together as a group.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFCF: Fast connect buffer used to connect the outputs of the LUTs
--        and some dedicated logic directly to the input of another LUT.
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFCF_inst: BUFCF (
port map (
   O => O, -- Connect to the output of a LUT
   I => I  -- Connect to the input of a LUT
);

-- End of BUFCF_inst instantiation
```

Verilog Instantiation Template

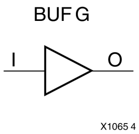
```
// BUFCF: Fast connect buffer used to connect the outputs of the LUTs
//        and some dedicated logic directly to the input of another LUT.
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFCF BUFCF_inst (
   .O(O), // Connect to the output of a LUT
   .I(I)  // Connect to the input of a LUT
);

// End of BUFCF_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFG

Primitive: Global Clock Buffer



Introduction

This design element is a high-fanout buffer that connects signals to the global routing resources for low skew distribution of the signal. BUFGs are typically used on clock nets as well other high fanout nets like sets/resets and clock enables.

Port Descriptions

Port	Direction	Width	Function
I	Input	1	Clock buffer input
O	Output	1	Clock buffer output

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFG: Global Clock Buffer (source by an internal signal)
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFG_inst : BUFG
port map (
   O => O,      -- Clock buffer output
   I => I       -- Clock buffer input
);

-- End of BUFG_inst instantiation
```

Verilog Instantiation Template

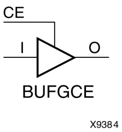
```
// BUFG: Global Clock Buffer (source by an internal signal)
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFG BUFG_inst (
   .O(O),     // Clock buffer output
   .I(I)      // Clock buffer input
);

// End of BUFG_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFGCE

Primitive: Global Clock Buffer with Clock Enable



Introduction

This design element is a global clock buffer with a single gated input. Its O output is "0" when clock enable (CE) is Low (inactive). When clock enable (CE) is High, the I input is transferred to the O output.

Logic Table

Inputs		Outputs
I	**CE**	**O**
X	0	0
I	1	I

Port Descriptions

Port	Direction	Width	Function
I	Input	1	Clock buffer input
CE	Input	1	Clock enable input
O	Output	1	Clock buffer output

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFGCE: Global Clock Buffer with Clock Enable (active high)
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFGCE_inst : BUFGCE
port map (
   O => O,   -- Clock buffer ouptput
   CE => CE, -- Clock enable input
   I => I    -- Clock buffer input
);

-- End of BUFGCE_inst instantiation
```

Verilog Instantiation Template

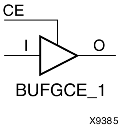
```
// BUFGCE: Global Clock Buffer with Clock Enable (active high)
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFGCE BUFGCE_inst (
   .O(O),   // Clock buffer output
   .CE(CE), // Clock enable input
   .I(I)    // Clock buffer input
);

// End of BUFGCE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFGCE_1

Primitive: Global Clock Buffer with Clock Enable and Output State 1



Introduction

This design element is a multiplexed global clock buffer with a single gated input. Its O output is High (1) when clock enable (CE) is Low (inactive). When clock enable (CE) is High, the I input is transferred to the O output.

Logic Table

Inputs		Outputs
I	**CE**	**O**
X	0	1
I	1	I

Port Descriptions

Port	Direction	Width	Function
I	Input	1	Clock buffer input
CE	Input	1	Clock enable input
O	Output	1	Clock buffer output

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFGCE_1: Global Clock Buffer with Clock Enable (active low)
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFGCE_1_inst : BUFGCE_1
port map (
   O => O,    -- Clock buffer ouptput
   CE => CE,  -- Clock enable input
   I => I     -- Clock buffer input
);
```

```
-- End of BUFGCE_1_inst instantiation
```

Verilog Instantiation Template

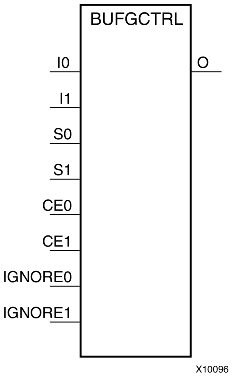
```
// BUFGCE_1: Global Clock Buffer with Clock Enable (active low)
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFGCE_1 BUFGCE_1_inst (
   .O(O),   // Clock buffer output
   .CE(CE), // Clock enable input
   .I(I)    // Clock buffer input
);

// End of BUFGCE_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFGCTRL

Primitive: Global Clock MUX Buffer



Introduction

BUFGCTRL primitive is global clock buffer that is designed as a synchronous/asynchronous "glitch free" 2:1 multiplexer with two clock inputs. Unlike global clock buffers that are found in previous generation of FPGAs, these clock buffers are designed with more control pins to provide a wider range of functionality and more robust input switching. BUFGCTRL is not limited to clocking applications.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Clock Output pin
I0, I1	Input	1 (each)	Clock Input: • I0 - Clock Input Pin • I1 - Clock Input Pin
CE0, CE1	Input	1 (each)	Clock Enable Input. The CE pins represent the clock enable pin for each clock inputs and are used to select the clock inputs. A setup/hold time must be specified when you are using the CE pin to select inputs. Failure to meet this requirement could result in a clock glitch.
S0, S1	Input	1 (each)	Clock Select Input. The S pins represent the clock select pin for each clock inputs. When using the S pin as input select, there is a setup/hold time requirement. Unlike CE pins, failure to meet this requirement will not result in a clock glitch. However, it can cause the output clock to appear one clock cycle later.
IGNORE0, IGNORE1	Input	1 (each)	Clock Ignore Input. IGNORE pins are used whenever a designer wants to bypass the switching algorithm executed by the BUFGCTRL.

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT_OUT	Integer	0, 1	0	Initializes the BUFGCTRL output to the specified value after configuration.
PRESELECT_I0	Boolean	FALSE, TRUE	FALSE	If TRUE, BUFGCTRL output uses I0 input after configuration.
PRESELECT_I1	Boolean	FALSE, TRUE	FALSE	If TRUE, BUFGCTRL output uses I1 input after configuration.

Note Both PRESELECT attributes might not be TRUE at the same time.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFGCTRL: Advanced Clock Primitive
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFGCTRL_inst : BUFGCTRL
generic map (
   INIT_OUT => 0,         -- Inital value of 0 or 1 after configuration
   PRESELECT_I0 => FALSE, -- TRUE/FALSE set the I0 input after configuration
   PRESELECT_I1 => FALSE) -- TRUE/FALSE set the I1 input after configuration
port map (
   O => O,               -- Clock MUX output
   CE0 => CE0,           -- Clock enable0 input
   CE1 => CE1,           -- Clock enable1 input
   I0 => I0,             -- Clock0 input
   I1 => I1,             -- Clock1 input
   IGNORE0 => IGNORE0,   -- Ignore clock select0 input
   IGNORE1 => IGNORE1,   -- Ignore clock select1 input
   S0 => S0,             -- Clock select0 input
   S1 => S1              -- Clock select1 input
);

-- End of BUFGCTRL_inst instantiation
```

Verilog Instantiation Template

```
// BUFGCTRL: Advanced Clock MUX Primitive
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFGCTRL #(
   .INIT_OUT(0),  // Inital value of 0 or 1 after configuration
   .PRESELECT_I0("FALSE"), // "TRUE" or "FALSE" set the I0 input after configuration
   .PRESELECT_I1("FALSE")  // "TRUE" or "FALSE" set the I1 input after configuration
) BUFGCTRL_inst (
   .O(O),     // 1-bit output
   .CE0(CE0), // 1-bit clock enable 0
   .CE1(CE1), // 1-bit clock enable 1
   .I0(I0),   // 1-bit clock 0 input
   .I1(I1),   // 1-bit clock 1 input
   .IGNORE0(IGNORE0), // 1-bit ignore 0 input
   .IGNORE1(IGNORE1), // 1-bit ignore 1 input
   .S0(S0),   // 1-bit select 0 input
   .S1(S1)    // 1-bit select 1 input
);
```

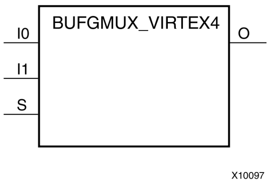
```
// End of BUFGCTRL_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFGMUX_VIRTEX4

Primitive: Global Clock MUX Buffer



Introduction

This design element is a global clock buffer with two clock inputs, one clock output, and a select line. This primitive is based on BUFGCTRL, with some pins connected to logic High or Low.

This element uses the S pins as select pins. S can switch anytime without causing a glitch. The Setup/Hold time on S is for determining whether the output will pass an extra pulse of the previously selected clock before switching to the new clock. If S changes prior to the setup time TBCCCK_S, and before I/O transitions from High to Low, then the output will not pass an extra pulse of I/O. If S changes following the hold time for S, then the output will pass an extra pulse, but it will not glitch. In any case the output will change to the new clock within three clock cycles of the slower clock.

The Setup/Hold requirements for S0 and S1 are with respect to the falling clock edge (assuming INIT_OUT = 0), not the rising edge, as for CE0 and CE1.

Switching conditions for this element are the same as the S pin of BUFGCTRL.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Clock Output
I1 : I0	Input	1	Clock Input
S0 : S1	Input	1	Clock Select Input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFGMUX_VIRTEX4: Global Clock Buffer 2-to-1 MUX
--                  Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFGMUX_VIRTEX4_inst : BUFGMUX_VIRTEX4
port map (
   O => O,     -- Clock MUX output
   I0 => I0,   -- Clock0 input
   I1 => I1,   -- Clock1 input
   S => S      -- Clock select input
);

-- End of BUFGMUX_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

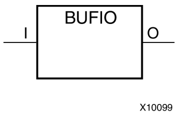
```
// BUFGMUX_VIRTEX4: Global Clock Buffer 2-to-1 MUX
//                  Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFGMUX_VIRTEX4 BUFGMUX_VIRTEX4_inst (
   .O(O),     // Clock MUX output
   .I0(I0),   // Clock0 input
   .I1(I1),   // Clock1 input
   .S(S)      // Clock select input
);

// End of BUFGMUX_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFIO

Primitive: Local Clock Buffer for I/O



Introduction

This design element is a clock buffer. It is simply a clock-in, clock-out buffer. It drives a dedicated clock net within the I/O column, independent of the global clock resources. Thus, these elements are ideally suited for source-synchronous data capture (forwarded/receiver clock distribution). They can only be driven by clock capable I/Os located in the same clock region. They drive the two adjacent I/O clock nets (for a total of up to three clock regions), as well as the regional clock buffers (BUFR). These elements cannot drive logic resources (CLB, block RAM, etc.) because the I/O clock network only reaches the I/O column.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Clock output
I	Input	1	Clock input

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFIO: Clock in, clock out buffer
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFIO_inst : BUFIO
port map (
   O => O,     -- Clock buffer output
   I => I      -- Clock buffer input
);

-- End of BUFIO_inst instantiation
```

Verilog Instantiation Template

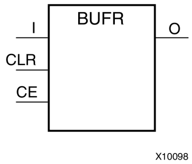
```
// BUFIO: Local Clock Buffer
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

BUFIO BUFIO_inst (
   .O(O),     // Clock buffer output
   .I(I)      // Clock buffer input
);

// End of BUFIO_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

BUFR

Primitive: Regional Clock Buffer for I/O and Logic Resources



Introduction

The BUFR is a clock buffer. BUFRs drive clock signals to a dedicated clock net within a clock region, independent from the global clock tree. Each BUFR can drive the two regional clock nets in the region in which it is located, and the two clock nets in the adjacent clock regions (up to three clock regions). Unlike BUFIOs, BUFRs can drive the I/O logic and logic resources (CLB, block RAM, etc.) in the existing and adjacent clock regions. BUFRs can be driven by either the output from BUFIOs or local interconnect. In addition, BUFRs are capable of generating divided clock outputs with respect to the clock input. The divide value is an integer between one and eight. BUFRs are ideal for source-synchronous applications requiring clock domain crossing or serial-to-parallel conversion. There are two BUFRs in a typical clock region (two regional clock networks). The center column does not have BUFRs.

Port Descriptions

Port	Direction	Width	Function
CE	Input	1	Clock enable port. When asserted low, this port disables the output clock. When asserted high, the clock is propagated out the O output port. Cannot be used in "BYPASS" mode. Connect to vcc when BUFR_DIVIDE is set to "BYPASS" or if not used.
CLR	Input	1	Counter asynchronous clear for divided clock output. When asserted high, this port resets the counter used to produce the divided clock output and the output is asserted low. Cannot be used in "BYPASS" mode. Connect to gnd when BUFR_DIVIDE is set to "BYPASS" or if not used.
I	Input	1	Clock input port. This port is the clock source port for BUFR. It can be driven by BUFIO output or local interconnect.
O	Output	1	Clock output port. This port drives the clock tracks in the clock region of the BUFR and the two adjacent clock regions. This port drives FPGA fabric, and IOBs.

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Yes
Macro support	No

Available Attributes

Attribute	Data Type	Allowed_Values	Default	Description
BUFR_DIVIDE	String	"BYPASS", "1", "2", "3", "4", "5", "6", "7", "8"	"BYPASS"	Defines whether the output clock is a divided version of input clock.
SIM_DEVICE	String	"VIRTEX4", VIRTEX5", "VIRTEX6"	"VIRTEX4"	Determine the CE latency for BUFR.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- BUFR: Regional (Local) Clock Buffer /w Enable, Clear and Division Capabilities
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

BUFR_inst : BUFR
generic map (
   BUFR_DIVIDE => "BYPASS",   -- "BYPASS", "1", "2", "3", "4", "5", "6", "7", "8"
   SIM_DEVICE => "VIRTEX4")   -- Specify target device, "VIRTEX4", "VIRTEX5", "VIRTEX6"
port map (
   O => O,      -- Clock buffer output
   CE => CE,    -- Clock enable input
   CLR => CLR, -- Clock buffer reset input
   I => I       -- Clock buffer input
);

-- End of BUFR_inst instantiation
```

Verilog Instantiation Template

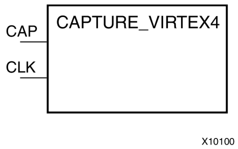
```
// BUFR: Regional Clock Buffer w/ Enable, Clear and Division Capabilities
//       Virtex-4/5, Virtex-6
// Xilinx HDL Libraries Guide, version 14.7

BUFR #(
   .BUFR_DIVIDE("BYPASS"), // "BYPASS", "1", "2", "3", "4", "5", "6", "7", "8"
   .SIM_DEVICE("VIRTEX4")  // Specify target device, "VIRTEX4", "VIRTEX5", "VIRTEX6"
) BUFR_inst (
   .O(O),      // Clock buffer output
   .CE(CE),    // Clock enable input
   .CLR(CLR), // Clock buffer reset input
   .I(I)       // Clock buffer input
);

// End of BUFR_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

CAPTURE_VIRTEX4

Primitive: Virtex®-4 Boundary Scan Logic Control Circuit



Introduction

This element provides user control and synchronization over when and how the capture register (flip-flop and latch) information task is requested. The readback function is provided through dedicated configuration port instructions. However, without this element, the readback data is synchronized to the configuration clock. Only register (flip-flop and latch) states can be captured. Although LUT RAM, SRL, and block RAM states are readback, they cannot be captured.

An asserted high CAP signal indicates that the registers in the device are to be captured at the next Low-to-High clock transition. By default, data is captured after every trigger when transition on CLK while CAP is asserted. To limit the readback operation to a single data capture, add the ONESHOT=TRUE attribute to this element.

Port Descriptions

Port	Direction	Width	Function
CAP	Input	1	Readback capture trigger
CLK	Input	1	Readback capture clock

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Connect all inputs and outputs to the design in order to ensure proper operation.

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
ONESHOT	Boolean	TRUE, FALSE	TRUE	Specifies the procedure for performing single readback per CAP trigger.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- CAPTURE_VIRTEX4: Register State Capture for Bitstream Readback
--                  Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

CAPTURE_VIRTEX4_inst : CAPTURE_VIRTEX4
generic map (
   ONESHOT => TRUE) -- TRUE or FALSE
port map (
   CAP => CAP,     -- Capture input
   CLK => CLK      -- Clock input
);
-- End of CAPTURE_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

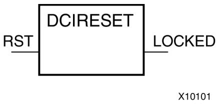
```
// CAPTURE_VIRTEX4: Register State Capture for Bitstream Readback
//                  Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

CAPTURE_VIRTEX4 #(
   .ONESHOT("TRUE") // "TRUE" or "FALSE"
) CAPTURE_VIRTEX4_inst (
   .CAP(CAP),       // Capture input
   .CLK(CLK)        // Clock input
);

// End of CAPTURE_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

DCIRESET

Primitive: DCI State Machine Reset (After Configuration Has Been Completed)



Introduction

This design element is used to reset the DCI state machine after configuration has been completed.

Port Descriptions

Port	Direction	Width	Function
LOCKED	Output	1	DCIRESET LOCK status output.
RST	Input	1	DCIRESET asynchronous reset input.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- DCIRESET: DCI reset component
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

DCIRESET_inst : DCIRESET
port map (
   LOCKED => LOCKED,     -- DCIRESET LOCK status output
   RST => RST            -- DCIRESET asynchronous reset input
);

-- End of DCIRESET_inst instantiation
```

Verilog Instantiation Template

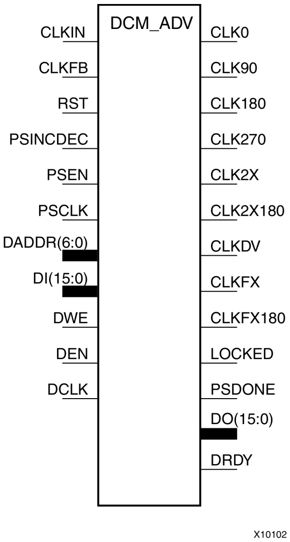
```
// DCIRESET: Digital Controlled Impedance (DCI) Reset Component
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

DCIRESET DCIRESET_inst (
   .LOCKED(LOCKED),  // 1-bit DCI LOCKED Output
   .RST(RST)         // 1-bit DCI Reset Input
);

// End of DCIRESET_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

DCM_ADV

Primitive: Advanced Digital Clock Manager Circuit



Introduction

This design element is a configurable/reconfigurable DLL with additional phase and frequency synthesis control capabilities. This component is commonly used for many FPGA applications in order to derive and control the various clocks needed within the system. If dynamic reconfiguration is not required, use either the DCM_BASE or DCM_PS components.

Port Descriptions

Port	Direction	Width	Function
Clock Outputs/Inputs			
CLK0	Output	1	The CLK0 output clock provides a clock with the same frequency as the effective CLKIN frequency. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE. When CLKFB is connected, CLK0 is phase aligned to CLKIN.
CLK90	Output	1	The CLK90 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 90 degrees.
CLK180	Output	1	The CLK180 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 180 degrees.
CLK270	Output	1	The CLK270 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 270 degrees.
CLK2X	Output	1	The CLK2X output clock provides a clock that is phase aligned to CLK0, with twice the CLK0 frequency, and with an automatic 50/50 duty-cycle correction. Until the DCM is locked, the CLK2X output appears as a 1x version of the input clock with a 25/75 duty cycle. This behavior allows the DCM to lock on the correct edge with respect to the source clock.
CLK2X180	Output	1	The CLK2X180 output clock provides a clock with the same frequency as CLK2X and phase-shifted by 180 degrees.

Port	Direction	Width	Function
CLKDV	Output	1	The frequency divide (CLKDV) output clock provides a clock that is phase aligned to CLK0 with a frequency that is a fraction of the effective CLKIN frequency. The fraction is determined by the CLKDV_DIVIDE attribute. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE.
CLKFX	Output	1	The frequency (CLKFX) output clock provides a clock with the following frequency definition: CLKFX Frequency = (M/D) x (Effective CLKIN Frequency). In this equation, M is the multiplier (numerator), with a value defined by the CLKFX_MULTIPLY attribute. D is the divisor (denominator), with a value defined by the CLKFX_DIVIDE attribute. Specifications for M and D, as well as input and output frequency ranges for the frequency synthesizer, are provided in the Data Sheet for this architecture. The rising edge of CLKFX output is phase aligned to the rising edges of CLK0, CLK2X, and CLKDV when the feedback path (CLKFB) is used. When M and D do have no common factor, the alignment occurs only once every D cycles of CLK0. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE.
CLKFX180	Output	1	The CLKFX180 output clock provides a clock with the same frequency as CLKFX and phase-shifted by 180 degrees.
CLKIN	Input	1	The source clock (CLKIN) input pin provides the source clock to the DCM. The CLKIN frequency must fall in the ranges specified in the Data Sheet for this architecture. The clock input signal comes from one of the following buffers: • IBUFG - Global Clock Input Buffer. The DCM compensates for the clock input path when an IBUFG, on the same edge (top or bottom) of the device, such as the DCM, is used. • BUFG/BUFGCTRL - Internal Global Clock Buffer. Any BUFGCTRL can drive any DCM in the device using the dedicated global routing. A BUFGCTRL can drive the DCM CLKIN pin when used to connect two DCM in series. • IBUF - Input Buffer. When IBUF drives CLKIN input, the PAD to DCM input skew is not compensated and increased jitter can occur. This configuration is generally not recommended.
CLKFB	Input	1	The feedback clock (CLKFB) input pin provides a reference or feedback signal to the DCM to delay-compensate the clock outputs and align it with the clock input. To provide the necessary feedback to the DCM, connect only the CLK0 output to the CLKFB input via a BUFG component in the case of internal feedback or an OBUF and IBUFG in the case of external feedback. Set the CLK_FEEDBACK attribute to 1X. When the CLKFB pin is connected, CLK0, CLKDV, and CLKFX are phase aligned to CLKIN. When the CLKFB pin is not connected, set CLK_FEEDBACK to "NONE" and only the CLKFX and CLKFX180 outputs are valid, however, not phase aligned to CLKIN.
Status Outputs/Control Inputs			
LOCKED	Output	1	Synchronous output from the PLL that provides you with an indication that the PLL has achieved phase alignment and is ready for operation.
PSDONE	Output	1	Dynamic CLKIN select input. When high (1), CLKIN1 is selected and while low (0), CLKIN2 is selected. If dual clock selection is not necessary, connect this input to a logic 1.

Port	Direction	Width	Function
RST	Input	1	The reset (RST) input pin resets the DCM circuitry. The RST signal is an active High asynchronous reset. Asserting the RST signal asynchronously forces all DCM outputs Low (the LOCKED signal, all status signals, and all output clocks within four source clock cycles). Because the reset is asynchronous, the last cycle of the clocks can exhibit an unintended short pulse, severely distorted duty-cycle, and no longer phase adjust with respect to one another while deasserting. The RST pin must be used when reconfiguring the device or changing the input frequency. Deasserting the RST signal synchronously starts the locking process at the next CLKIN cycle. To ensure a proper DCM reset and locking process, the RST signal must be deasserted after the CLKIN signal has been present and stable for at least three clock cycles. In all designs, the DCM must be held in reset until the clock is stable. During configuration, the DCM is automatically held in reset until GSR is released. If the clock is stable when GSR is released.
PSCLK	Input	1	The phase-shift clock (PSCLK) input pin provides the source clock for the DCM phase shift. The phase-shift clock signal can be driven by any clock source (external or internal). The frequency range of PSCLK is defined by PSCLK_FREQ_LF/HF (see the Data Sheet for this architecture). This input must be tied to ground when the CLKOUT_PHASE_SHIFT attribute is set to "NONE" or "FIXED".
PSINCDEC	Input	1	The PSINCDEC input signal is synchronous with PSCLK. The PSINCDEC input signal is used to increment or decrement the phase-shift factor when CLKOUT_PHASE_SHIFT is set to one of the variable modes. As a result, the output clock is phase shifted. the PSINCDEC signal is asserted High for increment, or deasserted Low for decrement. This input must be tied to ground when the CLKOUT_PHASE_SHIFT attribute is set to "NONE" or "FIXED".
PSEN	Input	1	The PSEN input signal is synchronous with PSCLK. A variable phase-shift operation is initiated by the PSEN input signal when CLKOUT_PHASE_SHIFT is set to a variable mode. It must be activated for one period of PSCLK. After PSEN is initiated, the phase change is effective for up to 100 CLKIN pulse cycles, plus three PSCLK cycles, and is indicated by a High pulse on PSDONE. There are no sporadic changes or glitches on any output during the phase transition. From the time PSEN is enabled until PSDONE is flagged, the DCM output clock moves bit-by-bit from its original phase shift to the target phase shift. The phase-shift is complete when PSDONE is flagged. PSEN must be tied to ground when the CLKOUT_PHASE_SHIFT attribute is set to "NONE" or "FIXED".
Dynamic Reconfiguration/DCM Status			
For more information on Dynamic Configuration, please see the Configuration User Guide.			
DO	Output	16	The DO output bus provides DCM status when not using the dynamic reconfiguration feature, and a data output when using the dynamic reconfiguration. When showing DCM status, the following mapping applies: • DO[0] - Phase-shift overflow • DO[1] - CLKIN stopped • DO[2] - CLKFX stopped • DO[3] - CLKFB stopped • DO[15:4] - Not assigned
DRDY	Output	1	The DRDY output pin provides ready status for the dynamic reconfiguration feature
DI	Input	16	The DI input bus provides reconfiguration data for dynamic reconfiguration. When not used, all bits must be assigned zeros.
DADDR	Input	7	The DADDR input bus provides a reconfiguration address for dynamic reconfiguration. When not used, all bits must be assigned zeros.

Port	Direction	Width	Function
DWE	Input	1	The DWE input pin provides the write enable control signal to write the DI data into the DADDR address. When not used, it must be tied Low.
DEN	Input	1	The DEN input pin provides the enable control signal to access the dynamic reconfiguration feature. To reflect the DCM status signals on the DO output bus when the dynamic reconfiguration feature is not used, DEN should be tied low.
DCLK	Input	1	The DCLK input pin provides the source clock for the dynamic reconfiguration circuit. The frequency of DCLK can be asynchronous (in phase and frequency) to CLKIN. The dynamic reconfiguration clock signal is driven by any clock source. The frequency range of DCLK is described in the Data Sheet for this architecture. When dynamic reconfiguration is not used, this input must be tied to ground.

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CLK_FEEDBACK	String	"1X" , or "NONE"	"1X"	Specifies the clock feedback of the allowed value.
CLKDV_DIVIDE	Float	1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0, 16.0	2.0	Specifies the extent to which the CLKDLL, CLKDLLE, CLKDLLHF, or DCM clock divider (CLKDV output) is to be frequency divided.
CLKFX_DIVIDE	Integer	1 to 32	1	Specifies the frequency divider value for the CLKFX output.
CLKFX_MULTIPLY	Integer	2 to 32	4	Specifies the frequency multiplier value for the CLKFX output.
CLKIN_DIVIDE_ BY_2	Boolean	FALSE, TRUE	FALSE	Allows for the input clock frequency to be divided in half when such a reduction is necessary to meet the DCM input clock frequency requirements.
CLKIN_PERIOD	Float	1.25 to 1000.00	10.0	Specifies period of input clock in ns from 1.25 to 1000.00.
CLKOUT_PHASE_ SHIFT	String	"NONE", "FIXED", "VARIABLE_ POSITIVE", "VARIABLE_ CENTER" or "DIRECT"	"NONE"	Specifies the phase shift mode of allowed value.

Attribute	Data Type	Allowed Values	Default	Description
DCM_ PERFORMANCE_ MODE	String	"MAX_SPEED" or "MAX_RANGE"	"MAX_SPEED"	Allows selection between maximum frequency and minimum jitter for low frequency and maximum phase shift range.
DESKEW_ADJUST	String	"SOURCE_ SYNCHRONOUS", "SYSTEM_ SYNCHRONOUS" or "0" to "15"	"SYSTEM_ SYNCHRONOUS"	Affects the amount of delay in the feedback path, and should be used for source-synchronous interfaces.
DFS_FREQUENCY_ MODE	String	"LOW" or "HIGH"	"LOW"	Specifies the frequency mode of the frequency synthesizer.
DLL_FREQUENCY_ MODE	String	"LOW" or "HIGH"	"LOW"	Specifies the DLL's frequency mode.
DUTY_CYCLE_ CORRECTION	Boolean	TRUE, FALSE	TRUE	Corrects the duty cycle of the CLK0, CLK90, CLK180, and CLK270 outputs.
FACTORY_JF	Hexa- decimal	Any 16-Bit value.	F0F0	The FACTORY_JF attribute affects the DCMs jitter filter characteristic. The default value should not be modified unless otherwise instructed by Xilinx.
PHASE_SHIFT	Integer	-255 to 1023	0	Specifies the phase shift numerator. The range depends on CLKOUT_PHASE_SHIFT.
SIM_DEVICE	String	"VIRTEX4" or "VIRTEX5"	"VIRTEX5"	Device selection.
STARTUP_WAIT	Boolean	FALSE, TRUE	FALSE	When TRUE, the configuration startup sequence waits in the specified cycle until the DCM locks.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- DCM_ADV: Digital Clock Manager Circuit
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

DCM_ADV_inst : DCM_ADV
generic map (
   CLKDV_DIVIDE => 2.0,  -- Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                         --   7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
   CLKFX_DIVIDE => 1,    -- Can be any integer from 1 to 32
   CLKFX_MULTIPLY => 4,  -- Can be any integer from 2 to 32
   CLKIN_DIVIDE_BY_2 => FALSE,    -- TRUE/FALSE to enable CLKIN divide by two feature
   CLKIN_PERIOD => 10.0,          -- Specify period of input clock in ns from 1.25 to 1000.00
   CLKOUT_PHASE_SHIFT => "NONE",  -- Specify phase shift mode of NONE, FIXED,
                                  -- VARIABLE_POSITIVE, VARIABLE_CENTER or DIRECT
   CLK_FEEDBACK => "1X",  -- Specify clock feedback of NONE or 1X
   DCM_PERFORMANCE_MODE => "MAX_SPEED",    -- Can be MAX_SPEED or MAX_RANGE
   DESKEW_ADJUST => "SYSTEM_SYNCHRONOUS", -- SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                          --   an integer from 0 to 15
   DFS_FREQUENCY_MODE => "LOW",     -- HIGH or LOW frequency mode for frequency synthesis
   DLL_FREQUENCY_MODE => "LOW",     -- LOW, HIGH, or HIGH_SER frequency mode for DLL
   DUTY_CYCLE_CORRECTION => TRUE,   -- Duty cycle correction, TRUE or FALSE
   FACTORY_JF => X"F0F0",           -- FACTORY JF Values Suggested to be set to X"F0F0"
   PHASE_SHIFT => 0,  -- Amount of fixed phase shift from -255 to 1023
```

```
   SIM_DEVICE => "VIRTEX4",        -- Set target device, "VIRTEX4" or "VIRTEX5"
   STARTUP_WAIT => FALSE)  -- Delay configuration DONE until DCM LOCK, TRUE/FALSE
port map (
   CLK0 => CLK0,         -- 0 degree DCM CLK output
   CLK180 => CLK180,     -- 180 degree DCM CLK output
   CLK270 => CLK270,     -- 270 degree DCM CLK output
   CLK2X => CLK2X,       -- 2X DCM CLK output
   CLK2X180 => CLK2X180, -- 2X, 180 degree DCM CLK out
   CLK90 => CLK90,       -- 90 degree DCM CLK output
   CLKDV => CLKDV,       -- Divided DCM CLK out (CLKDV_DIVIDE)
   CLKFX => CLKFX,       -- DCM CLK synthesis out (M/D)
   CLKFX180 => CLKFX180, -- 180 degree CLK synthesis out
   DO => DO,             -- 16-bit data output for Dynamic Reconfiguration Port (DRP)
   DRDY => DRDY,         -- Ready output signal from the DRP
   LOCKED => LOCKED,     -- DCM LOCK status output
   PSDONE => PSDONE,     -- Dynamic phase adjust done output
   CLKFB => CLKFB,       -- DCM clock feedback
   CLKIN => CLKIN,       -- Clock input (from IBUFG, BUFG or DCM)
   DADDR => DADDR,       -- 7-bit address for the DRP
   DCLK => DCLK,         -- Clock for the DRP
   DEN => DEN,           -- Enable input for the DRP
   DI => DI,             -- 16-bit data input for the DRP
   DWE => DWE,           -- Active high allows for writing configuration memory
   PSCLK => PSCLK,       -- Dynamic phase adjust clock input
   PSEN => PSEN,         -- Dynamic phase adjust enable input
   PSINCDEC => PSINCDEC, -- Dynamic phase adjust increment/decrement
   RST => RST            -- DCM asynchronous reset input
);

-- End of DCM_ADV_inst instantiation
```

Verilog Instantiation Template

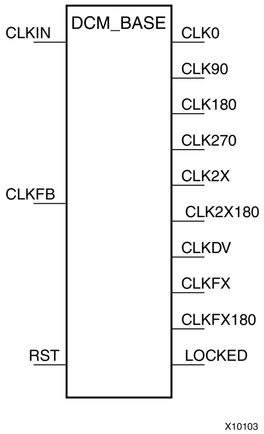
```
// DCM_ADV: Digital Clock Manager Circuit
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

DCM_ADV #(
   .CLKDV_DIVIDE(2.0), // Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                       //   7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
   .CLKFX_DIVIDE(1),   // Can be any integer from 1 to 32
   .CLKFX_MULTIPLY(4), // Can be any integer from 2 to 32
   .CLKIN_DIVIDE_BY_2("FALSE"), // TRUE/FALSE to enable CLKIN divide by two feature
   .CLKIN_PERIOD(10.0), // Specify period of input clock in ns from 1.25 to 1000.00
   .CLKOUT_PHASE_SHIFT("NONE"), // Specify phase shift mode of NONE, FIXED,
                                // VARIABLE_POSITIVE, VARIABLE_CENTER or DIRECT
   .CLK_FEEDBACK("1X"),  // Specify clock feedback of NONE, 1X or 2X
   .DCM_PERforMANCE_MODE("MAX_SPEED"), // Can be MAX_SPEED or MAX_RANGE
   .DESKEW_ADJUST("SYSTEM_SYNCHRONOUS"), // SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                         //   an integer from 0 to 15
   .DFS_FREQUENCY_MODE("LOW"), // HIGH or LOW frequency mode for frequency synthesis
   .DLL_FREQUENCY_MODE("LOW"), // LOW, HIGH, or HIGH_SER frequency mode for DLL
   .DUTY_CYCLE_CORRECTION("TRUE"), // Duty cycle correction, "TRUE"/"FALSE"
   .FACTORY_JF(16'hf0f0), // FACTORY JF value suggested to be set to 16'hf0f0
   .PHASE_SHIFT(0), // Amount of fixed phase shift from -255 to 1023
   .SIM_DEVICE("VIRTEX4"), // Set target device, "VIRTEX4" or "VIRTEX5"
   .STARTUP_WAIT("FALSE")  // Delay configuration DONE until DCM LOCK, "TRUE"/"FALSE"
) DCM_ADV_inst (
   .CLK0(CLK0),         // 0 degree DCM CLK output
   .CLK180(CLK180),     // 180 degree DCM CLK output
   .CLK270(CLK270),     // 270 degree DCM CLK output
   .CLK2X(CLK2X),       // 2X DCM CLK output
   .CLK2X180(CLK2X180), // 2X, 180 degree DCM CLK out
   .CLK90(CLK90),       // 90 degree DCM CLK output
   .CLKDV(CLKDV),       // Divided DCM CLK out (CLKDV_DIVIDE)
   .CLKFX(CLKFX),       // DCM CLK synthesis out (M/D)
   .CLKFX180(CLKFX180), // 180 degree CLK synthesis out
   .DO(DO),             // 16-bit data output for Dynamic Reconfiguration Port (DRP)
   .DRDY(DRDY),         // Ready output signal from the DRP
   .LOCKED(LOCKED),     // DCM LOCK status output
   .PSDONE(PSDONE),     // Dynamic phase adjust done output
   .CLKFB(CLKFB),       // DCM clock feedback
   .CLKIN(CLKIN),       // Clock input (from IBUFG, BUFG or DCM)
   .DADDR(DADDR),       // 7-bit address for the DRP
   .DCLK(DCLK),         // Clock for the DRP
   .DEN(DEN),           // Enable input for the DRP
   .DI(DI),             // 16-bit data input for the DRP
   .DWE(DWE),           // Active high allows for writing configuration memory
   .PSCLK(PSCLK),       // Dynamic phase adjust clock input
   .PSEN(PSEN),         // Dynamic phase adjust enable input
   .PSINCDEC(PSINCDEC), // Dynamic phase adjust increment/decrement
   .RST(RST)            // DCM asynchronous reset input
);

// End of DCM_ADV_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

DCM_BASE

Primitive: Base Digital Clock Manager Circuit



Introduction

This design element is a configurable DLL with additional phase and frequency synthesis control capabilities. This component is commonly used for many FPGA applications in order to derive and control the various clocks needed within the system. If dynamic reconfiguration is required, use the DCM_ADV component. If dynamic phase shift is required, use the DCM_PS component

Port Descriptions

Port	Direction	Width	Function
Clock Outputs/Inputs			
CLK0	Output	1	The CLK0 output clock provides a clock with the same frequency as the effective CLKIN frequency. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE. When CLKFB is connected, CLK0 is phase aligned to CLKIN.
CLK90	Output	1	The CLK90 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 90 degrees.
CLK180	Output	1	The CLK180 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 180 degrees.
CLK270	Output	1	The CLK270 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 270 degrees.
CLK2X	Output	1	The CLK2X output clock provides a clock that is phase aligned to CLK0, with twice the CLK0 frequency, and with an automatic 50/50 duty-cycle correction. Until the DCM is locked, the CLK2X output appears as a 1x version of the input clock with a 25/75 duty cycle. This behavior allows the DCM to lock on the correct edge with respect to the source clock.
CLK2X180	Output	1	The CLK2X180 output clock provides a clock with the same frequency as CLK2X and phase-shifted by 180 degrees.
CLKDV	Output	1	The frequency divide (CLKDV) output clock provides a clock that is phase aligned to CLK0 with a frequency that is a fraction of the effective CLKIN frequency. The fraction is determined by the CLKDV_DIVIDE attribute. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE.

Port	Direction	Width	Function
CLKFX	Output	1	The frequency (CLKFX) output clock provides a clock with the following frequency definition: CLKFX Frequency = (M/D) x (Effective CLKIN Frequency) In this equation, M is the multiplier (numerator) with a value defined by the CLKFX_MULTIPLY attribute. D is the divisor (denominator) with a value defined by the CLKFX_DIVIDE attribute. Specifications for M and D, as well as input and output frequency ranges for the frequency synthesizer, are provided in the Data Sheet for this architecture. The rising edge of CLKFX output is phase aligned to the rising edges of CLK0, CLK2X, and CLKDV when the feedback path (CLKFB) is used. When M and D to have no common factor, the alignment occurs only once every D cycles of CLK0. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE.
CLKFX180	Output	1	The CLKFX180 output clock provides a clock with the same frequency as CLKFX and phase-shifted by 180 degrees.
CLKIN	Input	1	The source clock (CLKIN) input pin provides the source clock to the DCM. The CLKIN frequency must fall in the ranges specified in the Data Sheet for this architecture. The clock input signal comes from one of the following buffers: • IBUFG - Global Clock Input Buffer. The DCM compensates for the clock input path when an IBUFG on the same edge (top or bottom) of the device as the DCM is used. • BUFG/BUFGCTRL - Internal Global Clock Buffer. Any BUFGCTRL can drive any DCM in the device using the dedicated global routing. A BUFGCTRL can drive the DCM CLKIN pin when used to connect two DCM in series. • IBUF - Input Buffer. When IBUF drives CLKIN input, the PAD to DCM input skew is not compensated and increased jitter can occur. This configuration is generally not recommended.
CLKFB	Input	1	The feedback clock (CLKFB) input pin provides a reference or feedback signal to the DCM to delay-compensate the clock outputs, and align it with the clock input. To provide the necessary feedback to the DCM, connect only the CLK0 output to the CLKFB input via a BUFG component in the case of internal feedback or an OBUF and IBUFG in the case of external feedback. Set the CLK_FEEDBACK attribute to 1X. When the CLKFB pin is connected, CLK0, CLKDV, and CLKFX are phase aligned to CLKIN. When the CLKFB pin is not connected, set CLK_FEEDBACK to NONE and only the CLKFX and CLKFX180 outputs are valid. However, they are not phase aligned to CLKIN.
Status Outputs/Control Inputs			
LOCKED	Output	1	Synchronous output from the PLL that provides you with an indication the PLL has achieved phase alignment and is ready for operation.
RST	Input	1	The reset (RST) input pin resets the DCM circuitry. The RST signal is an active High asynchronous reset. Asserting the RST signal asynchronously forces all DCM outputs Low (the LOCKED signal, all status signals, and all output clocks within four source clock cycles). Because the reset is asynchronous, the last cycle of the clocks can exhibit an unintended short pulse, severely distorted duty-cycle, and no longer phase adjust with respect to one another while deasserting. The RST pin must be used when reconfiguring the device or changing the input frequency. Deasserting the RST signal synchronously starts the locking process at the next CLKIN cycle. To ensure a proper DCM reset and locking process, the RST signal must be deasserted after the CLKIN signal has been present and stable for at least three clock cycles. In all designs, the DCM must be held in reset until the clock is stable. During configuration, the DCM is automatically held in reset until GSR is released. If the clock is stable when GSR is released.

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CLK_FEEDBACK	String	"1X" , "2X", or "NONE"	"1X"	Specifies the feedback input to the DCM (CLK0, or CLK2X).
CLKDV_DIVIDE	Float	1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0, 16.0	2.0	Specifies the extent to which the CLKDLL, CLKDLLE, CLKDLLHF, or DCM clock divider (CLKDV output) is to be frequency divided.
CLKFX_DIVIDE	Integer	1 to 32	1	Specifies the frequency divider value for the CLKFX output.
CLKFX_MULTIPLY	Integer	2 to 32	4	Specifies the frequency multiplier value for the CLKFX output.
CLKIN_DIVIDE_ BY_2	Boolean	FALSE, TRUE	FALSE	Allows for the input clock frequency to be divided in half when such a reduction is necessary to meet the DCM input clock frequency requirements.
CLKIN_PERIOD	Float	1.25 to 1000.00	10.0	Specifies the period of input clock in ns from 1.25 to 1000.00.
CLKOUT_PHASE_ SHIFT	String	"NONE", "FIXED", "VARIABLE_ POSITIVE", "VARIABLE_ CENTER" or "DIRECT"	"NONE"	Specifies the phase shift mode of allowed value.
DCM_ PERFORMANCE_ MODE	String	"MAX_SPEED" or "MAX_RANGE"	"MAX_SPEED"	Allows selection between maximum frequency and minimum jitter for low frequency and maximum phase shift range.
DESKEW_ADJUST	String	"SOURCE_ SYNCHRONOUS", "SYSTEM_ SYNCHRONOUS" or "0" to "15"	"SYSTEM_ SYNCHRONOUS"	Affects the amount of delay in the feedback path, and should be used for source-synchronous interfaces.
DFS_FREQUENCY_ MODE	String	"LOW" or "HIGH"	"LOW"	Specifies the frequency mode of the frequency synthesizer.
DLL_FREQUENCY_ MODE	String	"LOW" or "HIGH"	"LOW"	Specifies the DLL frequency mode
DUTY_CYCLE_ CORRECTION	Boolean	TRUE, FALSE	TRUE	Corrects the duty cycle of the CLK0, CLK90, CLK180, and CLK270 outputs.
FACTORY_JF	Hexa- decimal	Any 16-Bit Value	F0F0	The FACTORY_JF attribute affects the DCMs jitter filter characteristic. This attribute is set the default value

Attribute	Data Type	Allowed Values	Default	Description
				should not be modified unless otherwise instructed by Xilinx.
PHASE_SHIFT	Integer	-255 to 1023	0	Specifies the phase shift numerator. The range depends on CLKOUT_PHASE_SHIFT.
STARTUP_WAIT	Boolean	FALSE, TRUE	FALSE	When set to TRUE, the configuration startup sequence waits in the specified cycle until the DCM locks.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- DCM_BASE: Base Digital Clock Manager Circuit
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

DCM_BASE_inst : DCM_BASE
generic map (
   CLKDV_DIVIDE => 2.0, -- Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                        --   7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
   CLKFX_DIVIDE => 1,   -- Can be any integer from 1 to 32
   CLKFX_MULTIPLY => 4, -- Can be any integer from 2 to 32
   CLKIN_DIVIDE_BY_2 => FALSE, -- TRUE/FALSE to enable CLKIN divide by two feature
   CLKIN_PERIOD => 10.0, -- Specify period of input clock in ns from 1.25 to 1000.00
   CLKOUT_PHASE_SHIFT => "NONE", -- Specify phase shift mode of NONE or FIXED
   CLK_FEEDBACK => "1X",         -- Specify clock feedback of NONE or 1X
   DCM_PERFORMANCE_MODE => "MAX_SPEED",    -- Can be MAX_SPEED or MAX_RANGE
   DESKEW_ADJUST => "SYSTEM_SYNCHRONOUS", -- SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                          --    an integer from 0 to 15
   DFS_FREQUENCY_MODE => "LOW",   -- LOW or HIGH frequency mode for frequency synthesis
   DLL_FREQUENCY_MODE => "LOW",   -- LOW, HIGH, or HIGH_SER frequency mode for DLL
   DUTY_CYCLE_CORRECTION => TRUE, -- Duty cycle correction, TRUE or FALSE
   FACTORY_JF => X"F0F0",          -- FACTORY JF Values Suggested to be set to X"F0F0"
   PHASE_SHIFT => 0, -- Amount of fixed phase shift from -255 to 1023
   STARTUP_WAIT => FALSE) -- Delay configuration DONE until DCM LOCK, TRUE/FALSE
port map (
   CLK0 => CLK0,          -- 0 degree DCM CLK ouptput
   CLK180 => CLK180,      -- 180 degree DCM CLK output
   CLK270 => CLK270,      -- 270 degree DCM CLK output
   CLK2X => CLK2X,        -- 2X DCM CLK output
   CLK2X180 => CLK2X180, -- 2X, 180 degree DCM CLK out
   CLK90 => CLK90,        -- 90 degree DCM CLK output
   CLKDV => CLKDV,        -- Divided DCM CLK out (CLKDV_DIVIDE)
   CLKFX => CLKFX,        -- DCM CLK synthesis out (M/D)
   CLKFX180 => CLKFX180, -- 180 degree CLK synthesis out
   LOCKED => LOCKED,      -- DCM LOCK status output
   CLKFB => CLKFB,        -- DCM clock feedback
   CLKIN => CLKIN,        -- Clock input (from IBUFG, BUFG or DCM)
   RST => RST             -- DCM asynchronous reset input
);

-- End of DCM_BASE_inst instantiation
```

Verilog Instantiation Template

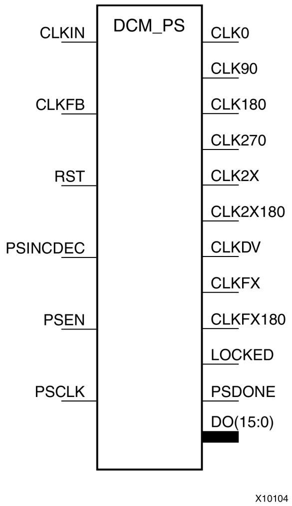
```
// DCM_BASE: Base Digital Clock Manager Circuit
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

DCM_BASE #(
   .CLKDV_DIVIDE(2.0), // Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                       //   7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
   .CLKFX_DIVIDE(1), // Can be any integer from 1 to 32
   .CLKFX_MULTIPLY(4), // Can be any integer from 2 to 32
   .CLKIN_DIVIDE_BY_2("FALSE"), // TRUE/FALSE to enable CLKIN divide by two feature
   .CLKIN_PERIOD(10.0), // Specify period of input clock in ns from 1.25 to 1000.00
   .CLKOUT_PHASE_SHIFT("NONE"), // Specify phase shift mode of NONE or FIXED
   .CLK_FEEDBACK("1X"), // Specify clock feedback of NONE, 1X or 2X
   .DCM_PERFORMANCE_MODE("MAX_SPEED"), // Can be MAX_SPEED or MAX_RANGE
   .DESKEW_ADJUST("SYSTEM_SYNCHRONOUS"), // SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                         //   an integer from 0 to 15
   .DFS_FREQUENCY_MODE("LOW"), // LOW or HIGH frequency mode for frequency synthesis
   .DLL_FREQUENCY_MODE("LOW"), // LOW, HIGH, or HIGH_SER frequency mode for DLL
   .DUTY_CYCLE_CORRECTION("TRUE"), // Duty cycle correction, TRUE or FALSE
   .FACTORY_JF(16'hf0f0), // FACTORY JF value suggested to be set to 16'hf0f0
   .PHASE_SHIFT(0), // Amount of fixed phase shift from -255 to 1023
   .STARTUP_WAIT("FALSE") // Delay configuration DONE until DCM LOCK, TRUE/FALSE
) DCM_BASE_inst (
   .CLK0(CLK0),         // 0 degree DCM CLK output
   .CLK180(CLK180),     // 180 degree DCM CLK output
   .CLK270(CLK270),     // 270 degree DCM CLK output
   .CLK2X(CLK2X),       // 2X DCM CLK output
   .CLK2X180(CLK2X180), // 2X, 180 degree DCM CLK out
   .CLK90(CLK90),       // 90 degree DCM CLK output
   .CLKDV(CLKDV),       // Divided DCM CLK out (CLKDV_DIVIDE)
   .CLKFX(CLKFX),       // DCM CLK synthesis out (M/D)
   .CLKFX180(CLKFX180), // 180 degree CLK synthesis out
   .LOCKED(LOCKED),     // DCM LOCK status output
   .CLKFB(CLKFB),       // DCM clock feedback
   .CLKIN(CLKIN),       // Clock input (from IBUFG, BUFG or DCM)
   .RST(RST)            // DCM asynchronous reset input
);

// End of DCM_BASE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

DCM_PS

Primitive: Digital Clock Manager with Basic and Phase Shift Features



Introduction

This design element is a configurable DLL with additional phase and frequency synthesis control capabilities. This component is commonly used for many FPGA applications in order to derive and control the various clocks needed within the system. If dynamic reconfiguration is requried, use DCM_ADV. If dynamic phase shift is not required, use DCM_BASE.

Port Descriptions

Port	Direction	Width	Function
Clock Outputs/Inputs			
CLK0	Output	1	The CLK0 output clock provides a clock with the same frequency as the effective CLKIN frequency. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE. When CLKFB is connected, CLK0 is phase aligned to CLKIN.
CLK90	Output	1	The CLK90 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 90 degrees.
CLK180	Output	1	The CLK180 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 180 degrees.
CLK270	Output	1	The CLK270 output clock provides a clock with the same frequency as CLK0 and phase-shifted by 270 degrees.
CLK2X	Output	1	The CLK2X output clock provides a clock that is phase aligned to CLK0, with twice the CLK0 frequency, and with an automatic 50/50 duty-cycle correction. Until the DCM is locked, the CLK2X output appears as a 1x version of the input clock with a 25/75 duty cycle. This behavior allows the DCM to lock on the correct edge with respect to the source clock.
CLK2X180	Output	1	The CLK2X180 output clock provides a clock with the same frequency as CLK2X, and phase-shifted by 180 degrees.

Port	Direction	Width	Function
CLKDV	Output	1	The frequency divide (CLKDV) output clock provides a clock that is phase aligned to CLK0 with a frequency that is a fraction of the effective CLKIN frequency. The fraction is determined by the CLKDV_DIVIDE attribute. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE.
CLKFX	Output	1	The frequency (CLKFX) output clock provides a clock with the following frequency definition: CLKFX Frequency = (M/D) x (Effective CLKIN Frequency) In this equation, M is the multiplier (numerator) with a value defined by the CLKFX_MULTIPLY attribute. D is the divisor (denominator) with a value defined by the CLKFX_DIVIDE attribute. Specifications for M and D, as well as input and output frequency ranges for the frequency synthesizer, are provided in the Data Sheet. The rising edge of CLKFX output is phase aligned to the rising edges of CLK0, CLK2X, and CLKDV when the feedback path (CLKFB) is used. When M and D to have no common factor, the alignment occurs only once every D cycles of CLK0. By default, the effective CLKIN frequency is equal to the CLKIN frequency, except when the CLKIN_DIVIDE_BY_2 attribute is set to TRUE.
CLKFX180	Output	1	The CLKFX180 output clock provides a clock with the same frequency as CLKFX and phase-shifted by 180 degrees.
CLKIN	Input	1	The source clock (CLKIN) input pin provides the source clock to the DCM. The CLKIN frequency must fall in the ranges specified in the Data Sheet. The clock input signal comes from one of the following buffers: • IBUFG - Global Clock Input Buffer. The DCM compensates for the clock input path when an IBUFG on the same edge (top or bottom) of the device as the DCM is used. • BUFG/BUFGCTRL - Internal Global Clock Buffer. Any BUFGCTRL can drive any DCM in the device using the dedicated global routing. A BUFGCTRL can drive the DCM CLKIN pin when used to connect two DCM in series. • IBUF - Input Buffer. When IBUF drives CLKIN input, the PAD to DCM input skew is not compensated and increased jitter can occur. This configuration is generally not recommended.
CLKFB	Input	1	The feedback clock (CLKFB) input pin provides a reference or feedback signal to the DCM to delay-compensate the clock outputs, and align it with the clock input. To provide the necessary feedback to the DCM, connect only the CLK0 output to the CLKFB input via a BUFG component in the case of internal feedback or an OBUF and IBUFG in the case of external feedback. Set the CLK_FEEDBACK attribute to 1X. When the CLKFB pin is connected, CLK0, CLKDV, and CLKFX are phase aligned to CLKIN. When the CLKFB pin is not connected, set CLK_FEEDBACK to "NONE" and only the CLKFX and CLKFX180 outputs are valid. However, they are not phase aligned to CLKIN.
Status Outputs/Control Inputs			
LOCKED	Output	1	Synchronous output from the PLL that provides you with an indication the PLL has achieved phase alignment and is ready for operation.
PSDONE	Output	1	Dynamic CLKIN select input. When high (1), CLKIN1 is selected and while low (0), CLKIN2 is selected. If dual clock selection is not necessary, connect this input to a logic 1.

Port	Direction	Width	Function
RST	Input	1	The reset (RST) input pin resets the DCM circuitry. The RST signal is an active High asynchronous reset. Asserting the RST signal asynchronously forces all DCM outputs Low (the LOCKED signal, all status signals, and all output clocks within four source clock cycles). Because the reset is asynchronous, the last cycle of the clocks can exhibit an unintended short pulse, severely distorted duty-cycle, and no longer phase adjust with respect to one another while deasserting. The RST pin must be used when reconfiguring the device or changing the input frequency. Deasserting the RST signal synchronously starts the locking process at the next CLKIN cycle. To ensure a proper DCM reset and locking process, the RST signal must be deasserted after the CLKIN signal has been present and stable for at least three clock cycles. In all designs, the DCM must be held in reset until the clock is stable. During configuration, the DCM is automatically held in reset until GSR is released. If the clock is stable when GSR is released.
PSCLK	Input	1	The phase-shift clock (PSCLK) input pin provides the source clock for the DCM phase shift. The phase-shift clock signal can be driven by any clock source (external or internal). The frequency range of PSCLK is defined by PSCLK_FREQ_LF/HF (see the Data Sheet). This input must be tied to ground when the CLKOUT_PHASE_SHIFT attribute is set to "NONE" or "FIXED".
PSINCDEC	Input	1	The PSINCDEC input signal is synchronous with PSCLK. The PSINCDEC input signal is used to increment or decrement the phase-shift factor when CLKOUT_PHASE_SHIFT is set to one of the variable modes. As a result, the output clock is phase shifted. the PSINCDEC signal is asserted High for increment, or deasserted Low for decrement. This input must be tied to ground when the CLKOUT_PHASE_SHIFT attribute is set to "NONE" or "FIXED".
PSEN	Input	1	The PSEN input signal is synchronous with PSCLK. A variable phase-shift operation is initiated by the PSEN input signal when CLKOUT_PHASE_SHIFT is set to a variable mode. It must be activated for one period of PSCLK. After PSEN is initiated, the phase change is effective for up to 100 CLKIN pulse cycles, plus three PSCLK cycles, and is indicated by a High pulse on PSDONE. There are no sporadic changes or glitches on any output during the phase transition. From the time PSEN is enabled until PSDONE is flagged, the DCM output clock moves bit-by-bit from its original phase shift to the target phase shift. The phase-shift is complete when PSDONE is flagged. PSEN must be tied to ground when the CLKOUT_PHASE_SHIFT attribute is set to "NONE" or "FIXED".

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CLK_FEEDBACK	String	"1X", "2X", or "NONE"	"1X"	Specifies the clock feedback of allowed value.
CLKDV_DIVIDE	FLOAT	1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0, 16.0	2.0	Specifies the extent to which the CLKDLL, CLKDLLE, CLKDLLHF, or DCM clock divider (CLKDV output) is to be frequency divided.
CLKFX_DIVIDE	Integer	1 to 32	1	Specifies the frequency divider value for the CLKFX output.
CLKFX_MULTIPLY	Integer	2 to 32	4	Specifies the frequency multiplier value for the CLKFX output.
CLKIN_DIVIDE_BY_2	Boolean	FALSE, TRUE	FALSE	Allows for the input clock frequency to be divided in half when such a reduction is necessary to meet the DCM input clock frequency requirements.
CLKIN_PERIOD	FLOAT	1.25 to 1000.00	10.0	Specifies the period of input clock in ns from 1.25 to 1000.00.
CLKOUT_PHASE_SHIFT	String	"NONE", "FIXED", "VARIABLE_POSITIVE", "VARIABLE_CENTER" or "DIRECT"	"NONE"	Specifies the phase shift mode of allowed value.
DESKEW_ADJUST	String	"SOURCE_SYNCHRONOUS", "SYSTEM_SYNCHRONOUS" or "0" to "15"	"SYSTEM_SYNCHRONOUS"	Affects the amount of delay in the feedback path, and should be used for source-synchronous interfaces.
DFS_FREQUENCY_MODE	String	"LOW" or "HIGH"	"LOW"	Specifies the frequency mode of the frequency synthesizer.
DLL_FREQUENCY_MODE	String	"LOW" or "HIGH"	"LOW"	Specifies the DLL frequency mode.
DUTY_CYCLE_CORRECTION	Boolean	TRUE, FALSE	TRUE	Corrects the duty cycle of the CLK0, CLK90, CLK180, and CLK270 outputs.
FACTORY_JF	Hexa-decimal	Any 16-Bit Value	F0F0	The FACTORY_JF attribute affects the DCM jitter filter characteristic. This attribute is set and the default value should not be modified unless otherwise instructed by Xilinx.
PHASE_SHIFT	Integer	-255 to 1023	0	Specifies the phase shift numerator. The range depends on CLKOUT_PHASE_SHIFT.
STARTUP_WAIT	Boolean	FALSE, TRUE	FALSE	When set to TRUE, the configuration startup sequence waits in the specified cycle until the DCM locks.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- DCM_PS: Digital Clock Manager Circuit
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

DCM_PS_inst : DCM_PS
generic map (
   CLKDV_DIVIDE => 2.0, -- Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                        --   7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
   CLKFX_DIVIDE => 1,   -- Can be any integer from 1 to 32
   CLKFX_MULTIPLY => 4, -- Can be any integer from 2 to 32
   CLKIN_DIVIDE_BY_2 => FALSE, -- TRUE/FALSE to enable CLKIN divide by two feature
   CLKIN_PERIOD => 10.0, -- Specify period of input clock in ns from 1.25 to 1000.00
   CLKOUT_PHASE_SHIFT => "NONE", -- Specify phase shift mode of NONE, FIXED,
                                 -- VARIABLE_POSITIVE, VARIABLE_CENTER or DIRECT
   CLK_FEEDBACK => "1X",  -- Specify clock feedback of NONE or 1X
   DCM_PERFORMANCE_MODE => "MAX_SPEED",   -- Can be MAX_SPEED or MAX_RANGE
   DESKEW_ADJUST => "SYSTEM_SYNCHRONOUS", -- SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                          --   an integer from 0 to 15
   DFS_FREQUENCY_MODE => "LOW",   -- HIGH or LOW frequency mode for frequency synthesis
   DLL_FREQUENCY_MODE => "LOW",   -- LOW, HIGH, or HIGH_SER frequency mode for DLL
   DUTY_CYCLE_CORRECTION => TRUE, -- Duty cycle correction, TRUE or FALSE
   FACTORY_JF => X"F0F0",          -- FACTORY JF Values Suggested to be set to X"F0F0"
   PHASE_SHIFT => 0,      -- Amount of fixed phase shift from -255 to 1023
   STARTUP_WAIT => FALSE) -- Delay configuration DONE until DCM LOCK, TRUE/FALSE
port map (
   CLK0 => CLK0,         -- 0 degree DCM CLK ouptput
   CLK180 => CLK180,     -- 180 degree DCM CLK output
   CLK270 => CLK270,     -- 270 degree DCM CLK output
   CLK2X => CLK2X,       -- 2X DCM CLK output
   CLK2X180 => CLK2X180, -- 2X, 180 degree DCM CLK out
   CLK90 => CLK90,       -- 90 degree DCM CLK output
   CLKDV => CLKDV,       -- Divided DCM CLK out (CLKDV_DIVIDE)
   CLKFX => CLKFX,       -- DCM CLK synthesis out (M/D)
   CLKFX180 => CLKFX180, -- 180 degree CLK synthesis out
   DO => DO,             -- 16-bit data output for Dynamic Reconfiguration Port (DRP)
   LOCKED => LOCKED,     -- DCM LOCK status output
   PSDONE => PSDONE,     -- Dynamic phase adjust done output
   CLKFB => CLKFB,       -- DCM clock feedback
   CLKIN => CLKIN,       -- Clock input (from IBUFG, BUFG or DCM)
   PSCLK => PSCLK,       -- Dynamic phase adjust clock input
   PSEN => PSEN,         -- Dynamic phase adjust enable input
   PSINCDEC => PSINCDEC, -- Dynamic phase adjust increment/decrement
   RST => RST            -- DCM asynchronous reset input
);

-- End of DCM_PS_inst instantiation
```

Verilog Instantiation Template

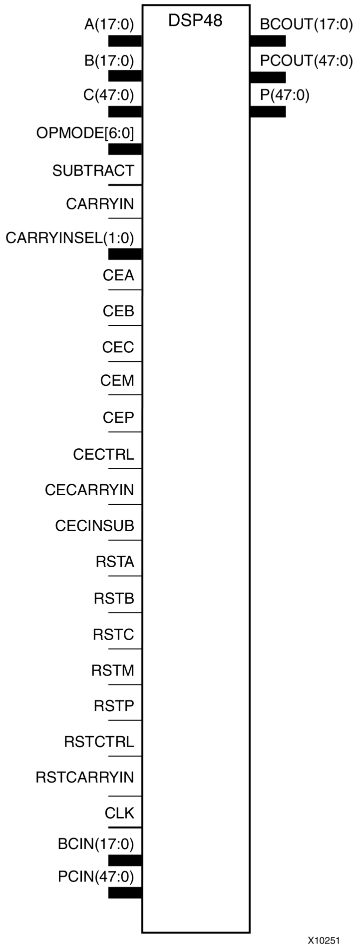
```
// DCM_PS: Dynamic Phase Shift Digital Clock Manager Circuit
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

DCM_PS #(
   .CLKDV_DIVIDE(2.0), // Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                       //   7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
   .CLKFX_DIVIDE(1), // Can be any integer from 1 to 32
   .CLKFX_MULTIPLY(4), // Can be any integer from 2 to 32
   .CLKIN_DIVIDE_BY_2("FALSE"), // TRUE/FALSE to enable CLKIN divide by two feature
   .CLKIN_PERIOD(10.0), // Specify period of input clock in ns from 1.25 to 1000.00
   .CLKOUT_PHASE_SHIFT("NONE"), // Specify phase shift mode of NONE, FIXED,
                                // VARIABLE_POSITIVE, VARIABLE_CENTER or DIRECT
   .CLK_FEEDBACK("1X"),  // Specify clock feedback of NONE, 1X or 2X
   .DCM_PERFORMANCE_MODE("MAX_SPEED"), // Can be MAX_SPEED or MAX_RANGE
   .DESKEW_ADJUST("SYSTEM_SYNCHRONOUS"), // SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                         //   an integer from 0 to 15
   .DFS_FREQUENCY_MODE("LOW"), // HIGH or LOW frequency mode for frequency synthesis
   .DLL_FREQUENCY_MODE("LOW"), // LOW, HIGH, or HIGH_SER frequency mode for DLL
   .DUTY_CYCLE_CORRECTION("TRUE"), // Duty cycle correction, TRUE or FALSE
   .FACTORY_JF(16'hf0f0), // FACTORY JF value suggested to be set to 16'hf0f0
   .PHASE_SHIFT(0), // Amount of fixed phase shift from -255 to 1023
   .STARTUP_WAIT("FALSE") // Delay configuration DONE until DCM LOCK, TRUE/FALSE
) DCM_PS_inst (
   .CLK0(CLK0),         // 0 degree DCM CLK output
   .CLK180(CLK180),     // 180 degree DCM CLK output
   .CLK270(CLK270),     // 270 degree DCM CLK output
   .CLK2X(CLK2X),       // 2X DCM CLK output
   .CLK2X180(CLK2X180), // 2X, 180 degree DCM CLK out
   .CLK90(CLK90),       // 90 degree DCM CLK output
   .CLKDV(CLKDV),       // Divided DCM CLK out (CLKDV_DIVIDE)
   .CLKFX(CLKFX),       // DCM CLK synthesis out (M/D)
   .CLKFX180(CLKFX180), // 180 degree CLK synthesis out
   .DO(DO),             // 16-bit data output for Dynamic Reconfiguration Port (DRP)
   .LOCKED(LOCKED),     // DCM LOCK status output
   .PSDONE(PSDONE),     // Dynamic phase adjust done output
   .CLKFB(CLKFB),       // DCM clock feedback
   .CLKIN(CLKIN),       // Clock input (from IBUFG, BUFG or DCM)
   .PSCLK(PSCLK),       // Dynamic phase adjust clock input
   .PSEN(PSEN),         // Dynamic phase adjust enable input
   .PSINCDEC(PSINCDEC), // Dynamic phase adjust increment/decrement
   .RST(RST)            // DCM asynchronous reset input
);

// End of DCM_PS_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

DSP48

Primitive: 18x18 Signed Multiplier Followed by a Three-Input Adder with Optional Pipeline Registers



Introduction

A slice for this design element has a 48-bit output and is primarily intended for use in digital-signal processing applications. However, the flexibility of this component means that it can be applied to many more applications than a typical MACC unit. A basic DSP48 slice consists of a multiplier followed by an adder. The multiplier accepts two, 18-bit, signed, two's complement operands producing a 36-bit, signed, two's complement result. The result is sign extended to 48 bits. The adder accepts three, 48-bit, signed, two's complement operands producing a 48-bit, singed, two's complement result.

Possible operands for the adder include the multiplier output and external source or the registered output of the adder providing an accumulate function. The 48-bit output allows for 4096 accumulations of 36-bit operands before overflow occurs.

Port Descriptions

Port	Direction	Width	Function
CLK	I	1	The DSP48 clock
A	I	18	The multiplier's A input, can also be used as adder's MSW input
B	I	18	The multiplier's B input, can also be used as adder's LSW input
BCIN	I	18	The multiplier's cascaded B input, can also be used as adder's LSW input
C	I	48	The adder's C input
PCIN	I	48	Cascaded adder's C Input from previous DSP slice
CARRYIN	I	1	The adders carry input
SUBTRACT	I	1	0= add, 1= (C, PCIN)-(mult,A:B)
OPMODE	I	7	Controls input to adder in DSP48 slices - see OPMODE table
CARRYINSEL	I	2	Selects carry source - see CARRINSEL table
CEA	I	1	Clock enable - 0=hold 1=enable AREG
CEB	I	1	Clock enable - 0=hold 1=enable BREG
CEC	I	1	Clock enable - 0=hold 1=enable CREG
CEP	I	1	Clock enable - 0=hold 1=enable PREG

Synthesis Attributes Used to Define Pipeline Registers

The following table describes the synthesis attributes used to define the pipeline registers.

Attribute	Function
AREG	0=bypass, 1=single, 2=dual
BREG	0=bypass, 1=single, 2=dual
CREG	0=bypass, 1=single
PREG	0=bypass, 1=single
MREG	0=bypass, 1=single
SUBTRACTREG	0=bypass, 1=single
OPMODEREG	0=bypass, 1=single
CARRYINSELREG	0=bypass, 1=single

Two's complement Signed Multiplier

The multiplier inside the DSP48 slice is an 18-bit x 18-bit two's complement multiplier with a 36-bit signed two's complement result. Cascading of multipliers to achieve larger products is supported. Applications such as signed-signed, signed-unsigned, and unsigned-unsigned multiplication, logical, arithmetic, barrel-shifter, two's complement and magnitude return are easily implemented. There are two independent dynamic data input ports. The input ports can represent 18-bit signed or 17-bit unsigned data.

X, Y, and Z Multiplexers

The Operational Mode (OpMode) inputs provide a way for the design to change its functionality on the fly. For example, the loading of an accumulator to restart an accumulation process. The OpMode bits can be optionally registered under the control of the configuration RAM.

The following tables list the possible values of OpMode and resulting function at the outputs of the three multiplexers supplying data to the adder/subtracter. The 7-bit OpMode control can be further broken down into multiplexer select bits. Not all possible combinations for the multiplexer select bits are allowed. If the multiplier output is selected then both the X and Y multiplexer are consumed with the multiplier output.

OpMode Control Bit Select X, Y, and Z Multiplexer Outputs

OPMODE Binary			X Multiplexer Output Fed to Add/Subtract
Z	Y	X	
XXX	XX	0	ZERO (Default)
XXX	1	1	Multiplier Output
XXX	XX	10	P
XXX	XX	11	A concatenated B

OpMode Control Bit Select X, Y, and Z Multiplexer Outputs

OPMODE Binary			Y Multiplexer Output Fed to Add/Subtract
Z	Y	X	
XXX	0	XX	ZERO (Default)
XXX	1	1	Multiplier Output
XXX	10	XX	Illegal selection
XXX	11	XX	C

OpMode Controls X, Y, and Z Multiplexer Outputs

OPMODE Binary			Y Multiplexer Output Fed to Add/Subtract
Z	Y	X	
XXX	0	XX	ZERO (Default)
XXX	1	1	Multiplier Output
XXX	10	XX	Illegal selection
XXX	11	XX	C

Three Input Adder/Subtracter Control Logic

The adder/subtracter output is a function of control and data inputs. The OpMode, as shown in the previous section, selects the inputs to the X, Y, Z multiplexer that are directed to the three adder/subtracter inputs. It also described that when the multiplier output is selected, both X and Y multiplexers are occupied. With the inputs to the adder/subtracter specified the function of the adder/subtracter itself must be examined. As with the input multiplexers, the OpMode bits specify a portion of this function. The table below shows this function. +/- in the table means either add or subtract and is specified by the state of the subtract control.

Hex OpMode	Binary OpMode	Output of Adder/Subtracter	Operation Description
[6:0]	**Z Y X**		
0x00	000 00 00	+/- CIN	Zero
0x02	000 00 10	+/- (P + CIN)	Hold P
0x03	000 00 11	+/- (A:B + CIN)	A:B select
0x05	000 01 01	+/- (A +/- B + CIN)	Multiply
0x0c	000 11 00	+/- (C + CIN)	C select
0x0e	000 11 10	+/- (C + P + CIN)	Feedback add

Hex OpMode	Binary OpMode	Output of Adder/Subtracter	Operation Description
[6:0]	Z Y X		
0x0f	000 11 11	+/- (A:B +C +CIN)	36-bit adder
0x10	001 00 00	PCIN +/- CIN	P cascade select
0x12	001 00 10	PCIN +/- (P + CIN)	P cascade feedback add
0x13	001 00 11	PCIN +/-(A:B + CIN)	P cascade add
0x15	001 01 01	PCIN +/- (A +/- B + CIN)	P cascade multiply add
0x1c	001 11 00	PCIN +/- (C + CIN)	P cascade add
0x1e	001 11 10	PCIN +/- (C+P+ CIN)	P cascade feedback add add
0x1c	001 11 11	PCIN +/- (A:B+C + CIN)	P cascade add add
0x20	010 00 00	P +/- CIN	Hold P
0x22	010 00 10	P +/- (P + CIN)	Double feedback add
0x23	010 00 11	P +/- (A:B + CIN)	Feedback add
0x25	010 01 01	P +/- (A +/- B + CIN)	Multiply-accumulate
0x2c	010 11 00	P +/- (C + CIN)	Feedback add
0x2e	010 11 10	P +/- (C + P + CIN)	Double feedback add
0x2f	010 11 11	P +/- (A:B + C + CIN)	Feedback add add
0x30	011 00 00	C +/- CIN	C Select
0x32	011 00 10	C +/- (P + CIN)	Feedback add
0x33	011 00 11	C +/- (A:B + CIN)	36-bit adder
0x35	011 01 01	C +/- (A +/- B + CIN)	Multiply add
0x3c	011 11 00	C +/- (C + CIN)	Double
0x3e	011 11 10	C +/- (C + P + CIN)	Double add feedback add
0x3f	011 11 11	C +/- (A:B + C + CIN)	Double add
0x50	101 00 00	Shift(PCIN) +/- CIN	17-bit shift P cascade select
0x52	101 00 10	Shift(PCIN) +/- (P + CIN)	17-bit shift P cascade feedback add
0x53	101 00 11	Shift(PCIN) +/- (A:B + CIN)	17-bit shift P cascade add
0x55	101 01 01	Shift(PCIN) +/- (A +/- B + CIN)	17-bit shift P cascade multiply add
0x5c	101 11 00	Shift(PCIN) +/- (C + CIN)	17-bit shift P cascade add
0x5e	101 11 10	Shift(PCIN) +/- (C + P + CIN)	17-bit shift P cascade feedback add add
0x5c	101 11 11	Shift(PCIN) +/- (A:B + C + CIN)	17-bit shift P cascade add add
0x60	110 00 00	Shift(P) +/- CIN	17-bit shift feedback
0x62	110 00 10	Shift(P) +/- (P + CIN)	17-bit shift feedback feedback add
0x63	110 00 11	Shift(P) +/- (A:B + CIN)	17-bit shift feedback add
0x65	110 01 01	Shift(P) +/- (A +/- B + CIN)	17-bit shift feedback multiply add
0x6c	110 11 00	Shift(P) +/- (C + CIN)	17-bit shift feedback add
0x6e	110 11 10	Shift(P) +/- (C + P + CIN)	17-bit shift feedback feedback add add
0x6f	110 11 11	Shift(P) +/- (A:B + C + CIN)	17-bit shift feedback add add

Rounding Modes Supported by Carry Logic

In addition to the OpMode inputs, the data inputs to the three input adder/subtracter, and the subtract control bit, the adder/subtracter output is a result of the carry-input logic.

CarryInSel signals, the Subtract control signal, and the OpMode control signals can be optionally registered under the control of the configuration RAM (denoted by the grey colored multiplexer symbol). This allows the control signals pipeline delay to match the pipeline delay for data in the design. The CarryInSel signals, the Subtract control signal, and the OpMode control signals share a common reset signal (RSTCTRL) and the Subtract control signal, and the OpMode control signals share a common clock enable signal. The clock enable allows control signals to stall along with data when needed.

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
AREG	Integer	0, 1, 2	1	Selects whether to register the A input to the DSP48.
B_INPUT	String	"DIRECT" or "CASCADE"	"DIRECT"	"DIRECT"=multiplicand is B; "CASCADE"=multiplicant is BCIN.
BREG	Integer	0, 1, 2	1	Selects whether to register the B input to the DSP48.
CARRYINREG	Integer	0,1	1	Number of pipeline registers for the CARRYIN input.
CARRYINSELREG	Integer	0, 1	1	Number of pipeline registers for the CARRYINSEL.
CREG	Integer	0, 1, 2	1	Selects whether to register the C input to the DSP48.
LEGACY_MODE	String	"NONE," "MULT18X18", or "MULT18X18S"	"MULT18X18S"	An internal attribute setting for the DCM. It should not be modified from the default value.
MREG	Integer	0, 1	1	Selects whether to register the multiplier stage of the DSP48. Enable=1/disable=0.
OPMODEREG	Integer	0, 1	1	Number of pipeline regsiters on OPMODE input, 0 or 1.
PREG	Integer	0, 1	1	Selects whether to register the C input to the DSP48.
SUBTRACTREG	Integer	0, 1	1	Number of pipeline registers on the SUBTRACT input, 0 or 1.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- DSP48: DSP Function Block
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

DSP48_inst : DSP48
generic map (
   AREG => 1,  -- Number of pipeline registers on the A input, 0, 1 or 2
   BREG => 1,  -- Number of pipeline registers on the B input, 0, 1 or 2
   B_INPUT => "DIRECT", -- B input DIRECT from fabric or CASCADE from another DSP48
   CARRYINREG => 1,     -- Number of pipeline registers for the CARRYIN input, 0 or 1
   CARRYINSELREG => 1,  -- Number of pipeline registers for the CARRYINSEL, 0 or 1
   CREG => 1,           -- Number of pipeline registers on the C input, 0 or 1
   LEGACY_MODE => "MULT18X18S", -- Backward compatibility, NONE, MULT18X18 or MULT18X18S
   MREG => 1,      -- Number of multiplier pipeline registers, 0 or 1
   OPMODEREG => 1, -- Number of pipeline registers on OPMODE input, 0 or 1
   PREG => 1,      -- Number of pipeline registers on the P output, 0 or 1
   SUBTRACTREG => 1) -- Number of pipeline registers on the SUBTRACT input, 0 or 1
port map (
   BCOUT => BCOUT,  -- 18-bit B cascade output
   P => P,          -- 48-bit product output
   PCOUT => PCOUT,  -- 48-bit cascade output
   A => A,          -- 18-bit A data input
   B => B,          -- 18-bit B data input
   BCIN => BCIN,    -- 18-bit B cascade input
   C => C,          -- 48-bit cascade input
   CARRYIN => CARRYIN,       -- Carry input signal
   CARRYINSEL => CARRYINSEL, -- 2-bit carry input select
   CEA => CEA,      -- A data clock enable input
   CEB => CEB,      -- B data clock enable input
   CEC => CEC,      -- C data clock enable input
   CECARRYIN => CECARRYIN, -- CARRYIN clock enable input
   CECINSUB => CECINSUB,   -- CINSUB clock enable input
   CECTRL => CECTRL, -- Clock Enable input for CTRL registers
   CEM => CEM,       -- Clock Enable input for multiplier registers
   CEP => CEP,       -- Clock Enable input for P registers
   CLK => CLK,       -- Clock input
   OPMODE => OPMODE, -- 7-bit operation mode input
   PCIN => PCIN,     -- 48-bit PCIN input
   RSTA => RSTA,     -- Reset input for A pipeline registers
   RSTB => RSTB,     -- Reset input for B pipeline registers
   RSTC => RSTC,     -- Reset input for C pipeline registers
   RSTCARRYIN => RSTCARRYIN, -- Reset input for CARRYIN registers
   RSTCTRL => RSTCTRL, -- Reset input for CTRL registers
   RSTM => RSTM,       -- Reset input for multiplier registers
   RSTP => RSTP,       -- Reset input for P pipeline registers
   SUBTRACT => SUBTRACT -- SUBTRACT input
);

-- End of DSP48_inst instantiation
```

Verilog Instantiation Template

```
// DSP48: DSP Function Block
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

DSP48 #(
   .AREG(1),           // Number of pipeline registers on the A input, 0, 1 or 2
   .BREG(1),           // Number of pipeline registers on the B input, 0, 1 or 2
   .B_INPUT("DIRECT"), // B input DIRECT from fabric or CASCADE from another DSP48
   .CARRYINREG(1),     // Number of pipeline registers for the CARRYIN input, 0 or 1
   .CARRYINSELREG(1),  // Number of pipeline registers for the CARRYINSEL, 0 or 1
   .CREG(1),           // Number of pipeline registers on the C input, 0 or 1
   .LEGACY_MODE("MULT18X18S"), // Backward compatibility, NONE, MULT18X18 or MULT18X18S
   .MREG(1),           // Number of multiplier pipeline registers, 0 or 1
   .OPMODEREG(1),      // Number of pipeline regsiters on OPMODE input, 0 or 1
   .PREG(1),           // Number of pipeline registers on the P output, 0 or 1
   .SUBTRACTREG(1)     // Number of pipeline registers on the SUBTRACT input, 0 or 1
) DSP48_inst (
   .BCOUT(BCOUT),  // 18-bit B cascade output
   .P(P),          // 48-bit product output
   .PCOUT(PCOUT),  // 48-bit cascade output
   .A(A),          // 18-bit A data input
   .B(B),          // 18-bit B data input
   .BCIN(BCIN),    // 18-bit B cascade input
   .C(C),          // 48-bit cascade input
   .CARRYIN(CARRYIN), // Carry input signal
   .CARRYINSEL(CARRYINSEL), // 2-bit carry input select
   .CEA(CEA),      // A data clock enable input
   .CEB(CEB),      // B data clock enable input
   .CEC(CEC),      // C data clock enable input
   .CECARRYIN(CECARRYIN), // CARRYIN clock enable input
   .CECINSUB(CECINSUB), // CINSUB clock enable input
   .CECTRL(CECTRL), // Clock Enable input for CTRL regsiters
   .CEM(CEM),       // Clock Enable input for multiplier regsiters
   .CEP(CEP),       // Clock Enable input for P regsiters
   .CLK(CLK),       // Clock input
   .OPMODE(OPMODE), // 7-bit operation mode input
   .PCIN(PCIN),     // 48-bit PCIN input
   .RSTA(RSTA),     // Reset input for A pipeline registers
   .RSTB(RSTB),     // Reset input for B pipeline registers
   .RSTC(RSTC),     // Reset input for C pipeline registers
   .RSTCARRYIN(RSTCARRYIN), // Reset input for CARRYIN registers
   .RSTCTRL(RSTCTRL), // Reset input for CTRL registers
   .RSTM(RSTM), // Reset input for multiplier registers
   .RSTP(RSTP), // Reset input for P pipeline registers
   .SUBTRACT(SUBTRACT) // SUBTRACT input
);

// End of DSP48_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

EMAC

Primitive: Fully integrated 10/100/1000 Mb/s Ethernet Media Access Controller (Ethernet MAC)

Introduction

This design element provides Ethernet connectivity to the Virtex®-4 PowerPC® Processor. The Ethernet MAC (EMAC) supports the following feature:

- Fully integrated 10/100/1000 Mb/s Ethernet MAC
- Complies with the IEEE 802.3-2002 specification
- Configurable full- or half-duplex operation
- Media Independent Interface (MII) Management (MDIO) interface to manage objects in the Physical (PHY) layer
- User-accessable raw statistics vector outputs
- Supports VLAN frames
- Configurable inter-frame gap adjustment
- Configurable in-band Frame Check Sequence (FCS) field passing on both transmit and receive paths
- Provides auto pad on transmit and FCS field stripping on receive
- Configured and monitored through a host interface
- Hardware selectable Device Control Register (DCR) bus or 1G Ethernet MAC bus host interface
- Configurable flow control through Ethernet MAC Control PAUSE frames; symmetrically or asymmetrically enabled
- Configurable support for jumbo frames of any length
- Configurable receive address filter for unicast, multicast, and broadcast addresses
- Media Independent Interface (MII), Gigabit Media Independent Interface (GMII), and Reduced Gigabit Media Independent Interface (RGMII)
- Includes a 1000BASE-X Physical Coding Sublayer (PCS) and a Physical Medium Attachment (PMA) sublayer for use with the Multi-gigabit Transceiver (MGT) to provide a complete on-chip 1000BASE-X implementation
- Serial Gigabit Media Independent Interface (SGMII) supported through MGT interface to external copper PHY layer

Port Descriptions

Inputs	Outputs
RESET	
TIEEMAC0CONFIGVEC [79:0]	
TIEEMAC1CONFIGVEC [79:0]	
TIEEMAC0UNICASTADDR [47:0]	
TIEEMAC1UNICASTADDR [47:0]	
PHYEMAC0GTXCLK	
PHYEMAC1GTXCLK	
CLIENTEMAC0DCMLOCKED	EMAC0CLIENTANINTERRUPT
CLIENTEMAC1DCMLOCKED	EMAC1CLIENTANINTERRUPT
CLIENTEMAC0RXCLIENTCLKIN	EMAC0CLIENTRXCLIENTCLKOUT
	EMAC0CLIENTRXD [15:0]

Inputs	Outputs
	EMAC0CLIENTRXDVLD
	EMAC0CLIENTRXDVLDMSW
	EMAC0CLIENTRXGOODFRAME
	EMAC0CLIENTRXBADFRAME
	EMAC0CLIENTRXFRAMEDROP
	EMAC0CLIENTRXDVREG6
	EMAC0CLIENTRXSTATS [6:0]
	EMAC0CLIENTRXSTATSBYTEVLD
	EMAC0CLIENTRXSTATSVLD
CLIENTEMAC1RXCLIENTCLKIN	EMAC1CLIENTRXCLIENTCLKOUT
	EMAC1CLIENTRXD [15:0]
	EMAC1CLIENTRXDVLD
	EMAC1CLIENTRXDVLDMSW
	EMAC1CLIENTRXGOODFRAME
	EMAC1CLIENTRXBADFRAME
	EMAC1CLIENTRXFRAMEDROP
	EMAC1CLIENTRXDVREG6
	EMAC1CLIENTRXSTATS [6:0]
	EMAC1CLIENTRXSTATSBYTEVLD
	EMAC1CLIENTRXSTATSVLD
CLIENTEMAC0TXGMIIMIICLKIN	EMAC0CLIENTTXGMIIMIICLKOUT
CLIENTEMAC0TXCLIENTCLKIN	EMAC0CLIENTTXCLIENTCLKOUT
CLIENTEMAC0TXD [15:0]	EMAC0CLIENTTXACK
CLIENTEMAC0TXDVLD	EMAC0CLIENTTXCOLLISION
CLIENTEMAC0TXDVLDMSW	EMAC0CLIENTTXRETRANSMIT
CLIENTEMAC0TXUNDERRUN	EMAC0CLIENTTXSTATS
CLIENTEMAC0TXIFGDELAY [7:0]	EMAC0CLIENTTXSTATSBYTEVLD
CLIENTEMAC0TXFIRSTBYTE	EMAC0CLIENTTXSTATSVLD
CLIENTEMAC1TXGMIIMIICLKIN	EMAC1CLIENTTXGMIIMIICLKOUT
CLIENTEMAC1TXCLIENTCLKIN	EMAC1CLIENTTXCLIENTCLKOUT
CLIENTEMAC1TXD [15:0]	EMAC1CLIENTTXACK
CLIENTEMAC1TXDVLD	EMAC1CLIENTTXCOLLISION
CLIENTEMAC1TXDVLDMSW	EMAC1CLIENTTXRETRANSMIT
CLIENTEMAC1TXUNDERRUN	EMAC1CLIENTTXSTATS
CLIENTEMAC1TXIFGDELAY [7:0]	EMAC1CLIENTTXSTATSBYTEVLD
CLIENTEMAC1TXFIRSTBYTE	EMAC1CLIENTTXSTATSVLD
CLIENTEMAC0PAUSEREQ	
CLIENTEMAC0PAUSEVAL [15:0]	

Inputs	Outputs
CLIENTEMAC1PAUSEREQ	
CLIENTEMAC1PAUSEVAL [15:0]	
HOSTADDR [9:0]	HOSTMIIMRDY
HOSTCLK	HOSTRDDATA [31:0]
HOSTMIIMSEL	
HOSTOPCODE [1:0]	
HOSTREQ	
HOSTWRDATA [31:0]	
HOSTEMAC1SEL	
DCREMACCLK	DCRHOSTDONEIR
DCREMACENABLE	EMACDCRACK
DCREMACDBUS [0:31]	EMACDCRDBUS [0:31]
DCREMACABUS [8:9]	
DCREMACREAD	
DCREMACWRITE	
PHYEMAC0RXCLK	EMAC0PHYTXCLK
PHYEMAC0RXD [7:0]	EMAC0PHYTXD [7:0]
PHYEMAC0RXDV	EMAC0PHYTXEN
PHYEMAC0RXER	EMAC0PHYTXER
PHYEMAC0MIITXCLK	
PHYEMAC0COL	
PHYEMAC0CRS	
PHYEMAC1RXCLK	EMAC1PHYTXCLK
PHYEMAC1RXD [7:0]	EMAC1PHYTXD [7:0]
PHYEMAC1RXDV	EMAC1PHYTXEN
PHYEMAC1RXER	EMAC1PHYTXER
PHYEMAC1MIITXCLK	
PHYEMAC1COL	
PHYEMAC1CRS	
PHYEMAC0SIGNALDET	EMAC0PHYENCOMMAALIGN
PHYEMAC0PHYAD [4:0]	EMAC0PHYLOOPBACKMSB
PHYEMAC0RXCLKCORCNT [2:0]	EMAC0PHYMGTRXRESET
PHYEMAC0RXBUFSTATUS [1:0]	EMAC0PHYMGTTXRESET
PHYEMAC0RXCHARISCOMMA	EMAC0PHYPOWERDOWN
PHYEMAC0RXCHARISK	EMAC0PHYSYNCACQSTATUS
PHYEMAC0RXCHECKINGCRC	EMAC0PHYTXCHARDISPMODE
PHYEMAC0RXCOMMADET	EMAC0PHYTXCHARDISPVAL
PHYEMAC0RXDISPERR	EMAC0PHYTXCHARISK

Inputs	Outputs
PHYEMAC0RXLOSSOFSYNC [1:0]	
PHYEMAC0RXNOTINTABLE	
PHYEMAC0RXRUNDISP	
PHYEMAC0RXBUFERR	
PHYEMAC0TXBUFERR	
PHYEMAC1SIGNALDET	EMAC1PHYENCOMMAALIGN
PHYEMAC1PHYAD [4:0]	EMAC1PHYLOOPBACKMSB
PHYEMAC1RXCLKCORCNT [2:0]	EMAC1PHYMGTRXRESET
PHYEMAC1RXBUFSTATUS [1:0]	EMAC1PHYMGTTXRESET
PHYEMAC1RXCHARISCOMMA	EMAC1PHYPOWERDOWN
PHYEMAC1RXCHARISK	EMAC1PHYSYNCACQSTATUS
PHYEMAC1RXCHECKINGCRC	EMAC1PHYTXCHARDISPMODE
PHYEMAC1RXCOMMADET	EMAC1PHYTXCHARDISPVAL
PHYEMAC1RXDISPERR	EMAC1PHYTXCHARISK
PHYEMAC1RXLOSSOFSYNC [1:0]	
PHYEMAC1RXNOTINTABLE	
PHYEMAC1RXRUNDISP	
PHYEMAC1RXBUFERR	
PHYEMAC1TXBUFERR	
PHYEMAC0MCLKIN	EMAC0PHYMCLKOUT
PHYEMAC0MDIN	EMAC0PHYMDOUT
	EMAC0PHYMDTRI
PHYEMAC1MCLKIN	EMAC1PHYMCLKOUT
PHYEMAC1MDIN	EMAC1PHYMDOUT
	EMAC1PHYMDTRI

Design Entry Method

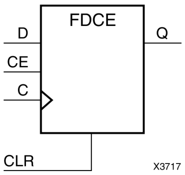
Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FDCE

Primitive: D Flip-Flop with Clock Enable and Asynchronous Clear



Introduction

This design element is a single D-type flip-flop with clock enable and asynchronous clear. When clock enable (CE) is High and asynchronous clear (CLR) is Low, the data on the data input (D) of this design element is transferred to the corresponding data output (Q) during the Low-to-High clock (C) transition. When CLR is High, it overrides all other inputs and resets the data output (Q) Low. When CE is Low, clock transitions are ignored.

This flip-flop is asynchronously cleared, outputs Low, when power is applied. For FPGA devices, power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs				Outputs
CLR	**CE**	**D**	**C**	**Q**
1	X	X	X	0
0	0	X	X	No Change
0	1	D	↑	D

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0, 1	0	Sets the initial value of Q output after configuration

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FDCE: Single Data Rate D Flip-Flop with Asynchronous Clear and
--       Clock Enable (posedge clk).
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FDCE_inst : FDCE
generic map (
   INIT => '0') -- Initial value of register ('0' or '1')
port map (
   Q => Q,      -- Data output
   C => C,      -- Clock input
   CE => CE,    -- Clock enable input
   CLR => CLR,  -- Asynchronous clear input
   D => D       -- Data input
);

-- End of FDCE_inst instantiation
```

Verilog Instantiation Template

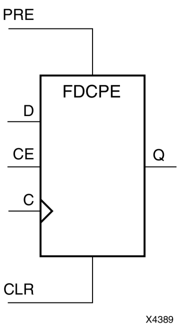
```
// FDCE: Single Data Rate D Flip-Flop with Asynchronous Clear and
//       Clock Enable (posedge clk).
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FDCE #(
   .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDCE_inst (
   .Q(Q),      // 1-bit Data output
   .C(C),      // 1-bit Clock input
   .CE(CE),    // 1-bit Clock enable input
   .CLR(CLR),  // 1-bit Asynchronous clear input
   .D(D)       // 1-bit Data input
);

// End of FDCE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FDCPE

Primitive: D Flip-Flop with Clock Enable and Asynchronous Preset and Clear



Introduction

This design element is a single D-type flip-flop with data (D), clock enable (CE), asynchronous preset (PRE), and asynchronous clear (CLR) inputs. The asynchronous active high PRE sets the Q output High; that active high CLR resets the output Low and has precedence over the PRE input. Data on the D input is loaded into the flip-flop when PRE and CLR are Low and CE is High on the Low-to-High clock (C) transition. When CE is Low, the clock transitions are ignored and the previous value is retained. The FDCPE is generally implemented as a slice or IOB register within the device.

For FPGA devices, upon power-up, the initial value of this component is specified by the INIT attribute. If a subsequent GSR (Global Set/Reset) is asserted, the flop is asynchronously set to the INIT value.

Note While this device supports the use of asynchronous set and reset, it is not generally recommended to be used for in most cases. Use of asynchronous signals pose timing issues within the design that are difficult to detect and control and also have an adverse affect on logic optimization causing a larger design that can consume more power than if a synchronous set or reset is used.

Logic Table

Inputs					Outputs
CLR	**PRE**	**CE**	**D**	**C**	**Q**
1	X	X	X	X	0
0	1	X	X	X	1
0	0	0	X	X	No Change
0	0	1	D	↑	D

Port Descriptions

Port	Direction	Width	Function
Q	Output	1	Data output
C	Input	1	Clock input
CE	Input	1	Clock enable input
CLR	Input	1	Asynchronous clear input
D	Input	1	Data input
PRE	Input	1	Asynchronous set input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0,1	0	Sets the initial value of Q output after configuration and on GSR.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FDCPE: Single Data Rate D Flip-Flop with Asynchronous Clear, Set and
--        Clock Enable (posedge clk).
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FDCPE_inst : FDCPE
generic map (
   INIT => '0') -- Initial value of register ('0' or '1')
port map (
   Q => Q,      -- Data output
   C => C,      -- Clock input
   CE => CE,    -- Clock enable input
   CLR => CLR,  -- Asynchronous clear input
   D => D,      -- Data input
   PRE => PRE   -- Asynchronous set input
);

-- End of FDCPE_inst instantiation
```

Verilog Instantiation Template

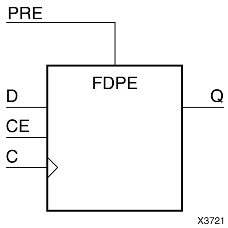
```
// FDCPE: Single Data Rate D Flip-Flop with Asynchronous Clear, Set and
//        Clock Enable (posedge clk).
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FDCPE #(
   .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDCPE_inst (
   .Q(Q),      // Data output
   .C(C),      // Clock input
   .CE(CE),    // Clock enable input
   .CLR(CLR),  // Asynchronous clear input
   .D(D),      // Data input
   .PRE(PRE)   // Asynchronous set input
);

// End of FDCPE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FDPE

Primitive: D Flip-Flop with Clock Enable and Asynchronous Preset



Introduction

This design element is a single D-type flip-flop with data (D), clock enable (CE), and asynchronous preset (PRE) inputs and data output (Q). The asynchronous PRE, when High, overrides all other inputs and sets the (Q) output High. Data on the (D) input is loaded into the flip-flop when PRE is Low and CE is High on the Low-to-High clock (C) transition. When CE is Low, the clock transitions are ignored.

For FPGA devices, this flip-flop is asynchronously preset, output High, when power is applied. Power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs				**Outputs**
PRE	**CE**	**D**	**C**	**Q**
1	X	X	X	1
0	0	X	X	No Change
0	1	D	↑	D

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0, 1	1	Sets the initial value of Q output after configuration

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FDPE: Single Data Rate D Flip-Flop with Asynchronous Preset and
--       Clock Enable (posedge clk).
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FDPE_inst : FDPE
generic map (
   INIT => '0') -- Initial value of register ('0' or '1')
port map (
   Q => Q,      -- Data output
   C => C,      -- Clock input
   CE => CE,    -- Clock enable input
   PRE => PRE,  -- Asynchronous preset input
   D => D       -- Data input
);

-- End of FDPE_inst instantiation
```

Verilog Instantiation Template

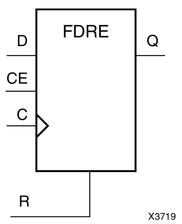
```
// FDPE: Single Data Rate D Flip-Flop with Asynchronous Preset and
//       Clock Enable (posedge clk).
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FDPE #(
   .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDPE_inst (
   .Q(Q),      // 1-bit Data output
   .C(C),      // 1-bit Clock input
   .CE(CE),    // 1-bit Clock enable input
   .PRE(PRE),  // 1-bit Asynchronous preset input
   .D(D)       // 1-bit Data input
);

// End of FDPE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FDRE

Primitive: D Flip-Flop with Clock Enable and Synchronous Reset



Introduction

This design element is a single D-type flip-flop with data (D), clock enable (CE), and synchronous reset (R) inputs and data output (Q). The synchronous reset (R) input, when High, overrides all other inputs and resets the (Q) output Low on the Low-to-High clock (C) transition. The data on the (D) input is loaded into the flip-flop when R is Low and CE is High during the Low-to-High clock transition.

This flip-flop is asynchronously cleared, outputs Low, when power is applied. For FPGA devices, power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs				Outputs
R	**CE**	**D**	**C**	**Q**
1	X	X	↑	0
0	0	X	X	No Change
0	1	D	↑	D

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0, 1	0	Sets the initial value of Q output after configuration

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FDRE: Single Data Rate D Flip-Flop with Synchronous Reset and
--       Clock Enable (posedge clk).
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FDRE_inst : FDRE
generic map (
   INIT => '0') -- Initial value of register ('0' or '1')
port map (
   Q => Q,      -- Data output
   C => C,      -- Clock input
   CE => CE,    -- Clock enable input
   R => R,      -- Synchronous reset input
   D => D       -- Data input
);

-- End of FDRE_inst instantiation
```

Verilog Instantiation Template

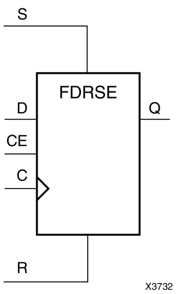
```
// FDRE: Single Data Rate D Flip-Flop with Synchronous Reset and
//       Clock Enable (posedge clk).
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FDRE #(
   .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDRE_inst (
   .Q(Q),      // 1-bit Data output
   .C(C),      // 1-bit Clock input
   .CE(CE),    // 1-bit Clock enable input
   .R(R),      // 1-bit Synchronous reset input
   .D(D)       // 1-bit Data input
);

// End of FDRE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FDRSE

Primitive: D Flip-Flop with Synchronous Reset and Set and Clock Enable



Introduction

FDRSE is a single D-type flip-flop with synchronous reset (R), synchronous set (S), clock enable (CE) inputs. The reset (R) input, when High, overrides all other inputs and resets the Q output Low during the Low-to-High clock transition. (Reset has precedence over Set.) When the set (S) input is High and R is Low, the flip-flop is set, output High, during the Low-to-High clock (C) transition. Data on the D input is loaded into the flip-flop when R and S are Low and CE is High during the Low-to-High clock transition.

Upon power-up, the initial value of this component is specified by the INIT attribute. If a subsequent GSR (Global Set/Reset) is asserted, the flop is asynchronously set to the INIT value.

Logic Table

Inputs					Outputs
R	**S**	**CE**	**D**	**C**	**Q**
1	X	X	X	↑	0
0	1	X	X	↑	1
0	0	0	X	X	No Change
0	0	1	1	↑	1
0	0	1	0	↑	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0,1	0	Sets the initial value of Q output after configuration and on GSR.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FDRSE: Single Data Rate D Flip-Flop with Synchronous Clear, Set and
--        Clock Enable (posedge clk).
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FDRSE_inst : FDRSE
generic map (
   INIT => '0') -- Initial value of register ('0' or '1')
port map (
   Q => Q,      -- Data output
   C => C,      -- Clock input
   CE => CE,    -- Clock enable input
   D => D,      -- Data input
   R => R,      -- Synchronous reset input
   S => S       -- Synchronous set input
);

-- End of FDRSE_inst instantiation
```

Verilog Instantiation Template

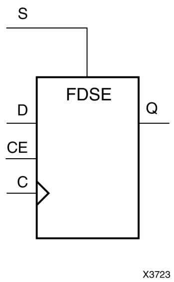
```
// FDRSE: Single Data Rate D Flip-Flop with Synchronous Clear, Set and
//        Clock Enable (posedge clk).
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FDRSE #(
   .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDRSE_inst (
   .Q(Q),      // Data output
   .C(C),      // Clock input
   .CE(CE),    // Clock enable input
   .D(D),      // Data input
   .R(R),      // Synchronous reset input
   .S(S)       // Synchronous set input
);

// End of FDRSE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FDSE

Primitive: D Flip-Flop with Clock Enable and Synchronous Set



Introduction

FDSE is a single D-type flip-flop with data (D), clock enable (CE), and synchronous set (S) inputs and data output (Q). The synchronous set (S) input, when High, overrides the clock enable (CE) input and sets the Q output High during the Low-to-High clock (C) transition. The data on the D input is loaded into the flip-flop when S is Low and CE is High during the Low-to-High clock (C) transition.

For FPGA devices, this flip-flop is asynchronously preset, output High, when power is applied. Power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs				Outputs
S	**CE**	**D**	**C**	**Q**
1	X	X	↑	1
0	0	X	X	No Change
0	1	D	↑	D

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0, 1	1	Sets the initial value of Q output after configuration

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FDSE: Single Data Rate D Flip-Flop with Synchronous Set and
--       Clock Enable (posedge clk).
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FDSE_inst : FDSE
generic map (
   INIT => '0') -- Initial value of register ('0' or '1')
port map (
   Q => Q,      -- Data output
   C => C,      -- Clock input
   CE => CE,    -- Clock enable input
   S => S,      -- Synchronous Set input
   D => D       -- Data input
);

-- End of FDSE_inst instantiation
```

Verilog Instantiation Template

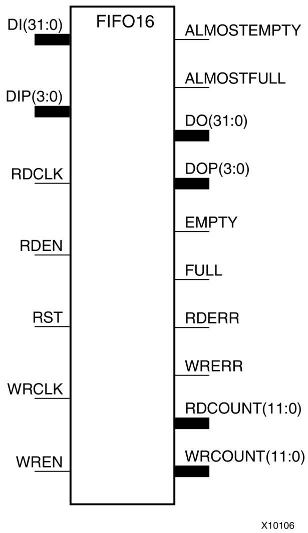
```
// FDSE: Single Data Rate D Flip-Flop with Synchronous Set and
//       Clock Enable (posedge clk).
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FDSE #(
   .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDSE_inst (
   .Q(Q),      // 1-bit Data output
   .C(C),      // 1-bit Clock input
   .CE(CE),    // 1-bit Clock enable input
   .S(S),      // 1-bit Synchronous set input
   .D(D)       // 1-bit Data input
);

// End of FDSE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FIFO16

Primitive: Virtex-4 Block RAM Based, Built-In FIFO



Introduction

A large percentage of FPGA designs implement FIFOs using block RAMs. In the Virtex®-4 architecture, additional dedicated logic in the block RAM enables you to easily implement synchronous or asynchronous FIFOs. This eliminates the need to use additional CLB logic for counter, comparator, or status flag generation and uses just one block RAM resource per FIFO. Both standard and first-word fall-through (FWFT) modes are supported.

Standard Mode -After the first word is written into an empty FIFO, the Empty flag deasserts synchronously with RDCLK. After Empty is deasserted Low and RDEN is asserted, the first word appears at DOUT on the rising edge of RDCLK.

First Word Fall Through Mode -After the first word is written into an empty FIFO, it automatically appears at DOUT after a few RDCLK cycles without asserting RDEN. Subsequent Read operations require Empty to be Low and RDEN to be High.

Note When using the dual-clock mode with independent clocks, depending on the offset between read and write clock edges, the Empty, Almost Empty, Full and Almost Full flags can deassert one cycle later. Due to the asynchronous nature of the clocks the simulation model only reflects the deassertion latency cycles listed in the architecture user guide.

The following table shows the FIFO capacity in the two modes:

FIFO Capacity Standard Mode	FWFT Mode
4k+1 entries by 4 bits	4k+2 entries by 4 bits
2k+1 entries by 9 bits	2k+2 entries by 9 bits
1k+1 entries by 18 bits	1k+2 entries by 18 bits
512+1 entries by 36 bits	512+2 entries by 36 bits

The block RAM can be configured as an asynchronous first-in/first-out (FIFO) memory with independent read and write clocks for either synchronous or asynchronous operation. Port A of the block RAM is used as a FIFO read port, and Port B is a FIFO write port. Data is read from the FIFO on the rising edge of read clock and written to the FIFO on the rising edge of write clock. Independent read and write port width selection is not supported in FIFO mode.

The available status flags are:

- **Full (FULL) -** Synchronous to WRCLK. The Full flag is asserted when there are no more available entries in the FIFO queue. When the FIFO is full, the write pointer will be frozen. This ensures the read and write pointers point to the same entry and no overflow will occur. The Full flag is registered at the output and takes one write cycle to assert. The Full flag is deasserted three clock cycles after the last entry is read, and it is synchronous to WRCLK.
- **Empty (EMPTY) -** Synchronous to RDCLK.
- **Almost Full (AFULL) -** Synchronous to WRCLK. The Almost Full flag is set when the FIFO has fewer than the number of available empty spaces specified by the ALMOST_FULL_OFFSET value. The Almost Full flag warns you to stop writing. It deasserts when the number of empty spaces in the FIFO is greater than the ALMOST_FULL_OFFSET value, and is synchronous to WRCLK.
- **Almost Empty (AEMPTY) -** Synchronous to RDCLK.
- **Write Count (WRCOUNT) -** Synchronous to WRCLK.
- **Write Error (WRERR) -** Synchronous to WRCLK. Once the Full flag has been asserted, any further write attempts will trigger the Write Error flag. The Write Error flag is deasserted when Write Enable or Full is deasserted Low. This signal is synchronous to WRCLK.
- **Read Count (RDCOUNT) -** Synchronous to RDCLK.
- **Read Error (RDERR) -** Synchronous to RDCLK.

Port Descriptions

Port	Direction	Function
DI	Input	Data input
DIP	Input	Parity-bit input
WREN	Input	Write enable. When WREN = 1, data will be written to memory. When WREN = 0, write is disabled.
WRCLK	Input	Clock for write domain operation.
RDEN	Input	Read enable. When RDEN = 1, data will be read to output register. When RDEN = 0, read is disabled.
RDCLK	Input	Clock for read domain operation.
RESET	Input	Asynchronous reset of all FIFO functions, flags, and pointers.
DO	Output	Data output, synchronous to RDCLK
DOP	Output	Parity-bit output, synchronous to RDCLK
FULL	Output	All entries in FIFO memory are filled.
ALMOSTFULL	Output	Almost all entries in FIFO memory have been filled. Synchronous to WRCLK. The value is configurable by you.
EMPTY	Output	FIFO is empty. No additional read can be performed. Synchronous to RDCLK.
ALMOSTEMPTY	Output	Almost all valid entries in FIFO are read. Synchronous with RDCLK. The value is configurable by you.
RDCOUNT	Output	The FIFO data read pointer. It is synchronous with RDCLK. The value will wrap around if the maximum read pointer value has been reached.
WRCOUNT	Output	The FIFO data write pointer. It is synchronous with WRCLK. The value will wrap around if the maximum write pointer value has been reached.
WRERR	Output	When the FIFO is full, any additional write operation generates an error flag. Synchronous with WRCLK.
RDERR	Output	When the FIFO is empty, any additional read operation generates an error flag. Synchronous with RDCLK.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
ALMOST_EMPTY_ OFFSET	Hexadecimal	Any 12-Bit Value	All zeros	Sets the almost empty threshold.
ALMOST_FULL_ OFFSET	Hexadecimal	Any 12-Bit Value	All zeros	Sets almost full threshold.
DATA_WIDTH	Integer	4, 9, 18, 36	36	Sets data width to allowed value.
FIRST_WORD_ FALL_THROUGH	Boolean	FALSE, TRUE	FALSE	Sets the FIFO FWFT to TRUE or FALSE.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FIFO16: BlockRAM Asynchronous FIFO configured fro 4k deep x 4 wide
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FIFO16_inst : FIFO16
generic map (
   ALMOST_FULL_OFFSET => X"080",  -- Sets almost full threshold
   ALMOST_EMPTY_OFFSET => X"080", -- Sets the almost empty threshold
   DATA_WIDTH => 4, -- Sets data width to 4, 9, 18, or 36
   FIRST_WORD_FALL_THROUGH => FALSE) --Sets the FIFO FWFT to TRUE or FALSE
port map (
   ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty output flag
   ALMOSTFULL => ALMOSTFULL,   -- 1-bit almost full output flag
   DO (31 DOWNTO 4) => unconnected (27 downto 0), -- Unused data output. Unconnected is a signal of 32 bits
   DO (3 DOWNTO 0) => DO,      -- 4-bit data output
   DOP => unconnected (31 downto 28), -- 4-bit Unused parity data output. Unconnected is a signal of 32 bits
   EMPTY => EMPTY,             -- 1-bit empty output flag
   FULL => FULL,               -- 1-bit full output flag
   RDCOUNT => RDCOUNT,         -- 12-bit read count output
   RDERR => RDERR,             -- 1-bit read error output
   WRCOUNT => WRCOUNT,         -- 12-bit write count output
   WRERR => WRERR,             -- 1-bit write error
   DI (31 DOWNTO 4) => X"0000000", -- Unused data inputs tied to ground
   DI (3 downto 0) => DI,      -- 4-bit data input
   DIP => X"0",       -- 4-bit Unused parity inputs tied to ground
   RDCLK => RDCLK,    -- 1-bit read clock input
   RDEN => RDEN,      -- 1-bit read enable input
   RST => RST,        -- 1-bit reset input
   WRCLK => WRCLK,    -- 1-bit write clock input
   WREN => WREN       -- 1-bit write enable input
);

-- End of FIFO16_inst instantiation
```

Verilog Instantiation Template

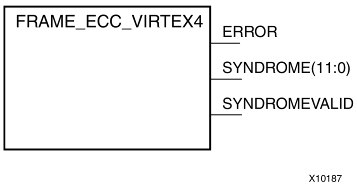
```
// FIFO16: BlockRAM Asynchronous FIFO configured for 4k deep x 4 wide
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

wire [27:0] unconnected;

FIFO16 #(
   .ALMOST_FULL_OFFSET(12'h080),     // Sets almost full threshold
   .ALMOST_EMPTY_OFFSET(12'h080),    // Sets the almost empty threshold
   .DATA_WIDTH(4),                   // Sets data width to 4, 9, 18, or 36
   .FIRST_WORD_FALL_THROUGH("FALSE") // Sets the FIFO FWFT to "TRUE" or "FALSE"
) FIFO16_4kx4_inst (
   .ALMOSTEMPTY(ALMOSTEMPTY),    // 1-bit almost empty output flag
   .ALMOSTFULL(ALMOSTFULL),      // 1-bit almost full output flag
   .DO({unconnected[27:0], DO}), // 4-bit data output
   .EMPTY(EMPTY),          // 1-bit empty output flag
   .FULL(FULL),            // 1-bit full output flag
   .RDCOUNT(RDCOUNT),      // 12-bit read count output
   .RDERR(RDERR),          // 1-bit read error output
   .WRCOUNT(WRCOUNT),      // 12-bit write count output
   .WRERR(WRERR),          // 1-bit write error
   .DI({28'h0000000, DI}), // 4-bit data input (rest tied to ground)
   .DIP(4'h0),             // Parity bits tied to Ground
   .RDCLK(RDCLK),          // 1-bit read clock input
   .RDEN(RDEN),            // 1-bit read enable input
   .RST(RST),              // 1-bit reset input
   .WRCLK(WRCLK),          // 1-bit write clock input
   .WREN(WREN)             // 1-bit write enable input
);

// End of FIFO16_4kx4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

FRAME_ECC_VIRTEX4

Primitive: Reads a Single, Virtex®-4 Configuration Frame and Computes a Hamming, Single-Error Correction, Double-Error Detection Syndrome



Introduction

This design element reads a single Virtex®-4 configuration frame of 1312-bits, 32-bits at a time. It will then compute a Hamming single error correction, double error detection "syndrome." This identifies the single frame bit (if any), which is in error and should be corrected. It also indicates the presence of two bit errors, which cannot be corrected. Note that the FRAME_ECC_VIRTEX4 primitive does not repair changed bits.

Port Descriptions

Port	Direction	Width	Function
ERROR	Output	1	Error Output. Indicates whether or not an error exists.
SYNDROME	Output	12	Indicates the location of the erroneous bit. Provides the bit location of the error and whether zero, one, or two erroneous bits are present.
SYNDROMEVALID	Output	1	When value is High, indicates the presence of zero, one or two bit errors in the frame. When asserted HIGH, SYNDROMEVALID indicates that the end of a frame readback.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Syndrome Value and Corresponding Error Status

Syndrome bit 11	Syndrome bit 10 to 0	Error Status
0	All 0s	No bit errors
0	Not equal to 0	One bit error, and syndrome value identifies the position of the erroneous bit
1	All 0s	Two bit errors, not correctable

Note SYNDROME_VALID must be HIGH for the values on the table above to be useful.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- FRAME_ECC_VIRTEX4: Configuration Frame Error Correction Circuitry
--                    Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

FRAME_ECC_VIRTEX4_inst : FRAME_ECC_VIRTEX4
port map (
   ERROR => ERROR,
   SYNDROME => SYNDROME,
   SYNDROMEVALID => SYNDROMEVALID
);

-- End of FRAME_ECC_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

```
// FRAME_ECC_VIRTEX4: Configuration Frame Error Correction Circuitry
//                    Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

FRAME_ECC_VIRTEX4 FRAME_ECC_VIRTEX4_inst (
   .ERROR(ERROR),       // 1-bit output indicating an error
   .SYNDROME(SYNDROME), // 12-bit output location of erroroneous bit
   .SYNDROMEVALID(SYNDROMEVALID) // 1-bit output indicating 0, 1 or 2 bit errors in frame
);

// End of FRAME_ECC_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

GT11_CUSTOM

Primitive: RocketIO MGTs with 622 Mb/s to 11.1 Gb/s Data Rates, 8 to 24 Transceivers per FPGA, and 2.5 GHz 5.55 GHz VCO, Less Than 1ns RMS Jitter

Introduction

This design element is a RocketIO™ MGT. RocketIO MGTs have flexible, programmable features that allow a multi-gigabit serial transceiver to be easily integrated into any Virtex®-4 design. These elements support the following features:

- 10.3 Gb/s data rates
- 8 to 24 transceivers per FPGA
- 2.5 GHz - 5.55 GHz VCO, less than 1ns RMS jitter
- Transmitter pre-emphasis
- Receiver continuous time equalization
- On-chip AC coupled receiver, with optional by-pass
- Receiver signal detect and loss of signal indicator, out of band signal receiver
- Transmit driver idle state for out of band signaling-both outputs at Vcm
- 8B/10B or 64B/66B encoding, or no data encoding (pass through mode)
- Channel bonding
- Flexible Cyclic Redundancy Check (CRC) generation and checking
- Pins for transmitter and receiver termination voltage
- You can reconfigure, using the secondary (dynamic) configuration bus
- Multiple loopback paths including PMA RX-TX path

RocketIO MGTs are only available in FX devices.

Logic Table

Inputs	Outputs
CHBONDI [4:0]	DRDY
CSUPMARESET	RXBUFERR
DADDR [7:0]	RXCALFAIL
DCLK	RXCOMMADET
DEN	RXCYCLELIMIT
DI [15:0]	RXLOCK
DWE	RXRealIGN
ENCHANSYNC	RXRECCLK1
ENMCOMMAALIGN	RXBCLK
ENPCOMMAALIGN	RXRECCLK2
GREFCLK	RXSIGDET
LOOPBACK [1:0]	TX1N
POWERDOWN	TX1P
REFCLK1	TXBUFERR
REFCLK2	TXCALFAIL

Inputs	Outputs
RX1N	TXCYCLELIMIT
RX1P	TXLOCK
RXBLOCKSYNC64B66BUSE	DO [15:0]
RXCLKSTABLE	RXLOSSOFSYNC [1:0]
RXCOMMADETUSE	RXCRCOUT [31:0]
RXCRCCLK	TXCRCOUT [31:0]
RXCRCDATAVALID	CHBONDO [4:0]
RXCRCDATAWIDTH [2:0]	RXSTATUS [5:0]
RXCRCIN [63:0]	RXDATA [63:0]
RXCRCINIT	RXCHARISCOMMA [7:0]
RXCRCINTCLK	RXCHARISK [7:0]
RXCRCPD	RXDISPERR [7:0]
RXCRCRESET	RXNOTINTABLE [7:0]
RXDATAWIDTH [1:0]	RXRUNDISP [7:0]
RXDEC64B66BUSE	TXRUNDISP [7:0]
RXDEC8B10BUSE	TXKERR [7:0]
RXDESCRAM64B66BUSE	
RXIGNOREBTF	
RXINTDATAWIDTH [1:0]	
RXPMARESET	
RXPOLARITY	
RXRESET	
RXSLIDE	
RXUSRCLK	
RXUSRCLK2	
TXBYPASS8B10B [7:0]	
TXCHARDISPMODE [7:0]	
TXCHARDISPVAL [7:0]	
TXCHARISK [7:0]	
TXCLKSTABLE	
TXCRCCLK	
TXCRCDATAVALID	
TXCRCDATAWIDTH [2:0]	
TXCRCIN [63:0]	
TXCRCINIT	
TXCRCINTCLK	
TXCRCPD	
TXCRCRESET	

Inputs	Outputs
TXDATA [63:0]	
TXDATAWIDTH [1:0]	
TXENC64B66BUSE	
TXENC8B10BUSE	
TXENOOB	
TXGEARBOX64B66BUSE	
TXINHIBIT	
TXINTDATAWIDTH [1:0]	
TXPMARESET	
TXPOLARITY	
TXRESET	
TXSCRAM64B66BUSE	
TXSYNC	
TXUSRCLK	
TXUSRCLK2	

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

GT11_DUAL

Primitive: RocketIO MGT Tile (contains 2 GT11_CUSTOM) with 622 Mb/s to 11.1 Gb/s data rates, 8 to 24 transceivers per FPGA, and 2.5 GHz 5.55 GHz VCO, less than 1ns RMS jitter

Introduction

RocketIO™ MGTs have flexible, programmable features that allow a multi-gigabit serial transceiver to be easily integrated into any Virtex®-4 design. The RocketIO MGTs support the following features:

- 622 Mb/s to 11.1 Gb/s data rates
- 8 to 24 transceivers per FPGA
- 2.5 GHz - 5.55 GHz VCO, less than 1ns RMS jitter
- Transmitter pre-emphasis (pre-equalization)
- Receiver continuous time equalization
- On-chip AC coupled receiver
- Digital oversampled receiver for data rates up to 2.5 Gb/s
- Receiver signal detect and loss of signal indicator, out-of-band signal receiver
- Transmit driver idle state for out-of-band signaling, both outputs at Vcm
- 8B/10B or 64B/66B encoding, or no data encoding (pass through mode)
- Channel bonding
- Flexible Cyclic Redundancy Check (CRC) generation and checking
- Pins for transmitter and receiver termination voltage
- You can reconfigure, using the secondary (dynamic) configuration bus
- Multiple loopback paths including PMA RX-TX path

Logic Table

Inputs	Outputs
LOOPBACK_A [1:0]	RXLOSSOFSYNC_A [1:0]
LOOPBACK_B [1:0]	RXLOSSOFSYNC_B [1:0]
RXDATAWIDTH_A [1:0]	DO_A [15:0]
RXDATAWIDTH_B [1:0]	DO_B [15:0]
RXINTDATAWIDTH_A [1:0]	RXCRCOUT_A [31:0]
RXINTDATAWIDTH_B [1:0]	RXCRCOUT_B [31:0]
TXDATAWIDTH_A [1:0]	TXCRCOUT_A [31:0]
TXDATAWIDTH_B [1:0]	TXCRCOUT_B [31:0]
TXINTDATAWIDTH_A [1:0]	CHBONDO_A [4:0]
TXINTDATAWIDTH_B [1:0]	CHBONDO_B [4:0]
DI_A [15:0]	RXSTATUS_A [5:0]
DI_B [15:0]	RXSTATUS_B [5:0]
RXCRCDATAWIDTH_A [2:0]	RXDATA_A [63:0]
RXCRCDATAWIDTH_B [2:0]	RXDATA_B [63:0]
TXCRCDATAWIDTH_A [2:0]	RXCHARISCOMMA_A [7:0]

Inputs	Outputs
TXCRCDATAWIDTH_B [2:0]	RXCHARISCOMMA_B [7:0]
CHBONDI_A [4:0]	RXCHARISK_A [7:0]
CHBONDI_B [4:0]	RXCHARISK_B [7:0]
RXCRCIN_A [63:0]	RXDISPERR_A [7:0]
RXCRCIN_B [63:0]	RXDISPERR_B [7:0]
TXCRCIN_A [63:0]	RXNOTINTABLE_A [7:0]
TXCRCIN_B [63:0]	RXNOTINTABLE_B [7:0]
TXDATA_A [63:0]	RXRUNDISP_A [7:0]
TXDATA_B [63:0]	RXRUNDISP_B [7:0]
DADDR_A [7:0]	TXKERR_A [7:0]
DADDR_B [7:0]	TXKERR_B [7:0]
TXBYPASS8B10B_A [7:0]	TXRUNDISP_A [7:0]
TXBYPASS8B10B_B [7:0]	TXRUNDISP_B [7:0]
TXCHARDISPMODE_A [7:0]	DRDY_A
TXCHARDISPMODE_B [7:0]	DRDY_B
TXCHARDISPVAL_A [7:0]	RXBUFERR_A
TXCHARDISPVAL_B [7:0]	RXBUFERR_B
TXCHARISK_A [7:0]	RXCALFAIL_A
TXCHARISK_B [7:0]	RXCALFAIL_B
DCLK_A	RXCOMMADET_A
DCLK_B	RXCOMMADET_B
DEN_A	RXCYCLELIMIT_A
DEN_B	RXCYCLELIMIT_B
DWE_A	RXLOCK_A
DWE_B	RXLOCK_B
ENCHANSYNC_A	RXMCLK_A
ENCHANSYNC_B	RXMCLK_B
ENMCOMMAALIGN_A	RXPCSHCLKOUT_A
ENMCOMMAALIGN_B	RXPCSHCLKOUT_B
ENPCOMMAALIGN_A	RXRealIGN_A
ENPCOMMAALIGN_B	RXRealIGN_B
GREFCLK_A	RXRECCLK1_A
GREFCLK_B	RXRECCLK1_B
POWERDOWN_A	RXRECCLK2_A
POWERDOWN_B	RXRECCLK2_B
REFCLK1_A	RXSIGDET_A
REFCLK1_B	RXSIGDET_B
REFCLK2_A	TX1N_A

Inputs	Outputs
REFCLK2_B	TX1N_B
RX1N_A	TX1P_A
RX1N_B	TX1P_B
RX1P_A	TXBUFERR_A
RX1P_B	TXBUFERR_B
RXBLOCKSYNC64B66BUSE_A	TXCALFAIL_A
RXBLOCKSYNC64B66BUSE_B	TXCALFAIL_B
RXCLKSTABLE_A	TXCYCLELIMIT_A
RXCLKSTABLE_B	TXCYCLELIMIT_B
RXCOMMADETUSE_A	TXLOCK_A
RXCOMMADETUSE_B	TXLOCK_B
RXCRCCLK_A	TXOUTCLK1_A
RXCRCCLK_B	TXOUTCLK1_B
RXCRCDATAVALID_A	TXOUTCLK2_A
RXCRCDATAVALID_B	TXOUTCLK2_B
RXCRCINIT_A	TXPCSHCLKOUT_A
RXCRCINIT_B	TXPCSHCLKOUT_B
RXCRCINTCLK_A	
RXCRCINTCLK_B	
RXCRCPD_A	
RXCRCPD_B	
RXCRCRESET_A	
RXCRCRESET_B	
RXDEC64B66BUSE_A	
RXDEC64B66BUSE_B	
RXDEC8B10BUSE_A	
RXDEC8B10BUSE_B	
RXDESCRAM64B66BUSE_A	
RXDESCRAM64B66BUSE_B	
RXIGNOREBTF_A	
RXIGNOREBTF_B	
RXPMARESET_A	
RXPMARESET_B	
RXPOLARITY_A	
RXPOLARITY_B	
RXRESET_A	
RXRESET_B	
RXSLIDE_A	

Inputs	Outputs
RXSLIDE_B	
RXSYNC_A	
RXSYNC_B	
RXUSRCLK_A	
RXUSRCLK_B	
RXUSRCLK2_A	
RXUSRCLK2_B	
TXCLKSTABLE_A	
TXCLKSTABLE_B	
TXCRCCLK_A	
TXCRCCLK_B	
TXCRCDATAVALID_A	
TXCRCDATAVALID_B	
TXCRCINIT_A	
TXCRCINIT_B	
TXCRCINTCLK_A	
TXCRCINTCLK_B	
TXCRCPD_A	
TXCRCPD_B	
TXCRCRESET_A	
TXCRCRESET_B	
TXENC64B66BUSE_A	
TXENC64B66BUSE_B	
TXENC8B10BUSE_A	
TXENC8B10BUSE_B	
TXENOOB_A	
TXENOOB_B	
TXGEARBOX64B66BUSE_A	
TXGEARBOX64B66BUSE_B	
TXINHIBIT_A	
TXINHIBIT_B	
TXPMARESET_A	
TXPMARESET_B	
TXPOLARITY_A	
TXPOLARITY_B	
TXRESET_A	
TXRESET_B	
TXSCRAM64B66BUSE_A	

Inputs	Outputs
TXSCRAM64B66BUSE_B	
TXSYNC_A	
TXSYNC_B	
TXUSRCLK_A	
TXUSRCLK_B	
TXUSRCLK2_A	
TXUSRCLK2_B	

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

GT11CLK

Primitive: A MUX That Can Select Fom Differential Package Input Clock, refclk From the Fabric, or rxbclk to Drive the Two Vertical Reference Clock Buses for the Column of MGTs

Introduction

This block needs to be instantiated when using the dedicated package pins for RocketIO™ clocks. There are two available per MGT column. The attributes allow this package input to drive one or both SYNCLK clock trees. Please see the *Virtex®-4 RocketIO MGT User Guide* for more details.

The attribute REFCLKSEL allows more clocking options. These options include: MGTCLK, SYNCLK1IN, SYNCLK2IN, REFCLK, RXBCLK.

Port Descriptions

Inputs are MGTCLKP, MGTCLKN

Outputs are SYNCLK1OUT, SYNCLK2OUT

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

GT11CLK_MGT

Primitive: Allows Differential Package Input to Drive the Two Vertical Reference Clock Buses for the Column of MGTs

Introduction

This block needs to be instantiated when using the dedicated package pins for RocketIO™ clocks. There are two available per MGT column.The attributes allow this package input to drive one or both SYNCLK clock trees. Please see the *Virtex®-4RocketIO MGT User Guide* for more details.

Port Description

Inputs are MGTCLKP, MGTCLKN.

Outputs are SYNCLK1OUT, SYNCLK2OUT.

Design Entry Method

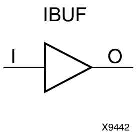
Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IBUF

Primitive: Input Buffer



Introduction

This design element is automatically inserted (inferred) by the synthesis tool to any signal directly connected to a top-level input or in-out port of the design. You should generally let the synthesis tool infer this buffer. However, it can be instantiated into the design if required. In order to do so, connect the input port (I) directly to the associated top-level input or in-out port, and connect the output port (O) to the logic sourced by that port. Modify any necessary generic maps (VHDL) or named parameter value assignment (Verilog) in order to change the default behavior of the component.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Buffer output
I	Input	1	Buffer input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

This element is usually inferred by the synthesis tool for any specified top-level input port to the design, and therefore it is generally not necessary to specify the element in source code. However, if desired, this element may be manually instantiated by copying the instantiation code from below and pasting it into the top-level entity/module of your code. Xilinx recommends that you put all I/O components on the top-level of the design to help facilitate hierarchical design methods. Connect the I port directly to the top-level input port of the design and the O port to the logic in which this input is to source. Specify the desired generic/defparam values in order to configure the proper behavior of the buffer.

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
IOSTANDARD	String	See Data Sheet.	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IBUF: Single-ended Input Buffer
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IBUF_inst : IBUF
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   IOSTANDARD => "DEFAULT")
port map (
   O => O,      -- Buffer output
   I => I       -- Buffer input (connect directly to top-level port)
);

-- End of IBUF_inst instantiation
```

Verilog Instantiation Template

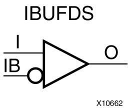
```
// IBUF: Single-ended Input Buffer
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IBUF #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .IOSTANDARD("DEFAULT")     // Specify the input I/O standard
)IBUF_inst (
   .O(O),      // Buffer output
   .I(I)       // Buffer input (connect directly to top-level port)
);

// End of IBUF_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IBUFDS

Primitive: Differential Signaling Input Buffer



Introduction

This design element is an input buffer that supports low-voltage, differential signaling. In IBUFDS, a design level interface signal is represented as two distinct ports (I and IB), one deemed the "master" and the other the "slave." The master and the slave are opposite phases of the same logical signal (for example, MYNET_P and MYNET_N). Optionally, a programmable differential termination feature is available to help improve signal integrity and reduce external components.

Logic Table

Inputs		Outputs
I	**IB**	**O**
0	0	No Change
0	1	0
1	0	1
1	1	No Change

Port Descriptions

Port	Direction	Width	Function
I	Input	1	Diff_p Buffer Input
IB	Input	1	Diff_n Buffer Input
O	Output	1	Buffer Output

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Put all I/O components on the top-level of the design to help facilitate hierarchical design methods. Connect the I port directly to the top-level "master" input port of the design, the IB port to the top-level "slave" input port, and the O port to the logic in which this input is to source. Specify the desired generic/defparam values in order to configure the proper behavior of the buffer.

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
DIFF_TERM	Boolean	TRUE or FALSE	FALSE	Enables the built-in differential termination resistor.
IOSTANDARD	String	See Data Sheet.	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IBUFDS: Differential Input Buffer
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IBUFDS_inst : IBUFDS
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   DIFF_TERM => FALSE, -- Differential Termination
   IOSTANDARD => "DEFAULT")
port map (
   O => O,  -- Buffer output
   I => I,  -- Diff_p buffer input (connect directly to top-level port)
   IB => IB -- Diff_n buffer input (connect directly to top-level port)
);

-- End of IBUFDS_inst instantiation
```

Verilog Instantiation Template

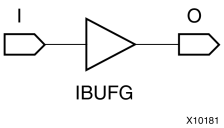
```
// IBUFDS: Differential Input Buffer
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IBUFDS #(
   .CAPACITANCE("DONT_CARE"), // "LOW", "NORMAL", "DONT_CARE" (Virtex-4 only)
   .DIFF_TERM("FALSE"),       // Differential Termination
   .IOSTANDARD("DEFAULT")     // Specify the input I/O standard
) IBUFDS_inst (
   .O(O),  // Buffer output
   .I(I),  // Diff_p buffer input (connect directly to top-level port)
   .IB(IB) // Diff_n buffer input (connect directly to top-level port)
);

// End of IBUFDS_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IBUFG

Primitive: Dedicated Input Clock Buffer



Introduction

The IBUFG is a dedicated input to the device which should be used to connect incoming clocks to the FPGA's global clock routing resources. The IBUFG provides dedicated connections to the DCM_SP and BUFG providing the minimum amount of clock delay and jitter to the device. The IBUFG input can only be driven by the global clock pins. The IBUFG output can drive CLKIN of a DCM_SP, BUFG, or your choice of logic.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Clock Buffer output
I	Input	1	Clock Buffer input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IBUFG: Single-ended global clock input buffer
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IBUFG_inst : IBUFG
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   IOSTANDARD => "DEFAULT")
port map (
   O => O, -- Clock buffer output
   I => I  -- Clock buffer input (connect directly to top-level port)
);

-- End of IBUFG_inst instantiation
```

Verilog Instantiation Template

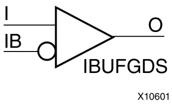
```
// IBUFG: Single-ended global clock input buffer
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IBUFG #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .IOSTANDARD("DEFAULT")  // Specify the input I/O standard
) IBUFG_inst (
   .O(O), // Clock buffer output
   .I(I)  // Clock buffer input (connect directly to top-level port)
);

// End of IBUFG_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IBUFGDS

Primitive: Differential Signaling Dedicated Input Clock Buffer and Optional Delay



Introduction

This design element is a dedicated differential signaling input buffer for connection to the clock buffer (BUFG) or DCM. In IBUFGDS, a design-level interface signal is represented as two distinct ports (I and IB), one deemed the "master" and the other the "slave." The master and the slave are opposite phases of the same logical signal (for example, MYNET_P and MYNET_N). Optionally, a programmable differential termination feature is available to help improve signal integrity and reduce external components. Also available is a programmable delay is to assist in the capturing of incoming data to the device.

Logic Table

Inputs		Outputs
I	**IB**	**O**
0	0	No Change
0	1	0
1	0	1
1	1	No Change

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Clock Buffer output
IB	Input	1	Diff_n Clock Buffer Input
I	Input	1	Diff_p Clock Buffer Input

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Put all I/O components on the top-level of the design to help facilitate hierarchical design methods. Connect the I port directly to the top-level "master" input port of the design, the IB port to the top-level "slave" input port and the O port to a DCM, BUFG or logic in which this input is to source. Some synthesis tools infer the BUFG automatically if necessary, when connecting an IBUFG to the clock resources of the FPGA. Specify the desired generic/defparam values in order to configure the proper behavior of the buffer.

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_ CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
DIFF_TERM	Boolean	TRUE or FALSE	FALSE	Enables the built-in differential termination resistor.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IBUFGDS: Differential Global Clock Input Buffer
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IBUFGDS_inst : IBUFGDS
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   DIFF_TERM => FALSE, -- Differential Termination
   IOSTANDARD => "DEFAULT")
port map (
   O => O,  -- Clock buffer output
   I => I,  -- Diff_p clock buffer input (connect directly to top-level port)
   IB => IB -- Diff_n clock buffer input (connect directly to top-level port)
);

-- End of IBUFGDS_inst instantiation
```

Verilog Instantiation Template

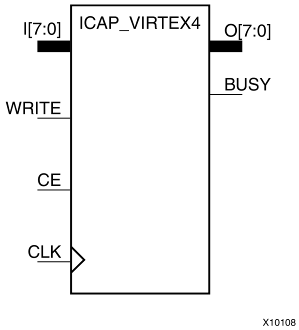
```
// IBUFGDS: Differential Global Clock Input Buffer
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IBUFGDS #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .DIFF_TERM("FALSE"),    // Differential Termination
   .IOSTANDARD("DEFAULT")  // Specify the input I/O standard
) IBUFGDS_inst (
   .O(O),  // Clock buffer output
   .I(I),  // Diff_p clock buffer input (connect directly to top-level port)
   .IB(IB) // Diff_n clock buffer input (connect directly to top-level port)
);

// End of IBUFGDS_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ICAP_VIRTEX4

Primitive: Virtex-4 Internal Configuration Access Port



Introduction

This design element provides user access to the Virtex®-4 internal configuration access port (ICAP).

Port Descriptions

Port	Direction	Width	Function
BUSY	Output	1	Busy signal
O	Output	32	32-bit data bus output
CE	Input	1	Clock enable pin
CLK	Input	1	Clock input
WRITE	Input	1	Write signal
I	Input	32	32-bit data bus input

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
ICAP_WIDTH	String	"X8" or "X32"	"X8"	Specifies the data width for the ICAP component.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ICAP_VIRTEX4: Internal Configuration Access Port
--               Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ICAP_VIRTEX4_inst : ICAP_VIRTEX4
generic map (
   ICAP_WIDTH => "X8") -- "X8" or "X32"
port map (
   BUSY => BUSY,    -- Busy output
   O => O,          -- 32-bit data output
   CE => CE,        -- Clock enable input
   CLK => CLK,      -- Clock input
   I => I,          -- 32-bit data input
   WRITE => WRITE   -- Write input
);

-- End of ICAP_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

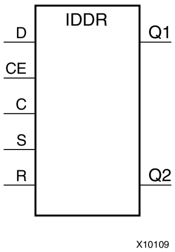
```
// ICAP_VIRTEX4: Internal Configuration Access Port
//               Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ICAP_VIRTEX4 #(
   .ICAP_WIDTH("X8") // "X8" or "X32"
) ICAP_VIRTEX4_inst (
   .BUSY(BUSY),      // Busy output
   .O(O),            // 32-bit data output
   .CE(CE),          // Clock enable input
   .CLK(CLK),        // Clock input
   .I(I),            // 32-bit data input
   .WRITE(WRITE)     // Write input
);

// End of ICAP_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IDDR

Primitive: Input Dual Data-Rate Register



Introduction

This design element is a dedicated input register designed to receive external dual data rate (DDR) signals into Xilinx® FPGAs.The IDDR is available with modes that present the data to the FPGA fabric at the time and clock edge they are captured, or on the same clock edge. This feature allows you to avoid additional timing complexities and resource usage.

- **OPPOSITE_EDGE mode -** Data is recovered in the classic DDR methodology. Given a DDR data and clock at pin D and C respectively, Q1 changes after every positive edge of clock C, and Q2 changes after every negative edge of clock C.
- **SAME_EDGE mode -** Data is still recovered by opposite edges of clock C. However, an extra register has been placed behind the negative edge data register. This extra register is clocked with positive clock edge of clock signal C. As a result, DDR data is now presented into the FPGA fabric at the same clock edge. However, because of this feature, the data pair appears to be "separated." Q1 and Q2 no longer have pair 1 and 2. Instead, the first pair presented is Pair 1 and DONT_CARE, followed by Pair 2 and 3 at the next clock cycle.
- **SAME_EDGE_PIPELINED mode -** Recovers data in a similar fashion as the SAME_EDGE mode. In order to avoid the "separated" effect of the SAME_EDGE mode, an extra register has been placed in front of the positive edge data register. A data pair now appears at the Q1 and Q2 pin at the same time. However, using this mode costs you an additional cycle of latency for Q1 and Q2 signals to change.

IDDR also works with the SelectIO™ features, such as the IODELAY.

Note For high speed interfaces, the IDDR_2CLK component can be used to specify two independent clocks to capture the data. Use this component when the performance requirements of the IDDR are not adequate, since the IDDR_2CLK requires more clocking resources and can imply placement restrictions that are not necessary when using the IDDR component.

Port Descriptions

Port	Direction	Width	Function
Q1 - Q2	Output	1	These pins are the IDDR output that connects to the FPGA fabric. Q1 is the first data pair and Q2 is the second data pair.
C	Input	1	Clock input pin.
CE	Input	1	When asserted Low, this port disables the output clock at port O.
D	Input	1	This pin is where the DDR data is presented into the IDDR module. This pin connects to a top-level input or bi-directional port, and IODELAY configured for an input delay or to an appropriate input or bidirectional buffer.
R	Input	1	Active high reset forcing Q1 and Q2 to a logic zero. Can be synchronous or asynchronous based on the SRTYPE attribute.
S	Input	1	Active high reset forcing Q1 and Q2 to a logic one. Can be synchronous or asynchronous based on the SRTYPE attribute.

Note You cannot have an active set and an active reset in this component. One or both of the signals R and S must be tied to ground.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
DDR_CLK_EDGE	String	"OPPOSITE_EDGE", "SAME_EDGE", "SAME_EDGE_PIPELINED"	"OPPOSITE_EDGE"	Sets the IDDR mode of operation with respect to clock edge.
INIT_Q1	Binary	0, 1	0	Initial value on the Q1 pin after configuration startup or when GSR is asserted.
INIT_Q2	Binary	0, 1	0	Initial value on the Q2 pin after configuration startup or when GSR is asserted.
SRTYPE	String	"SYNC" or "ASYNC"	"SYNC"	Set/reset type selection. "SYNC" specifies the behavior of the reset (R) and set (S) pins to be synchronous to the positive edge of the C clock pin. "ASYNC" specifies an asynchronous set/reset function.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IDDR: Double Data Rate Input Register with Set, Reset
--       and Clock Enable.
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IDDR_inst : IDDR
generic map (
   DDR_CLK_EDGE => "OPPOSITE_EDGE", -- "OPPOSITE_EDGE", "SAME_EDGE"
                                    -- or "SAME_EDGE_PIPELINED"
   INIT_Q1 => '0', -- Initial value of Q1: '0' or '1'
   INIT_Q2 => '0', -- Initial value of Q2: '0' or '1'
   SRTYPE => "SYNC") -- Set/Reset type: "SYNC" or "ASYNC"
port map (
   Q1 => Q1, -- 1-bit output for positive edge of clock
   Q2 => Q2, -- 1-bit output for negative edge of clock
   C => C,   -- 1-bit clock input
   CE => CE, -- 1-bit clock enable input
   D => D,   -- 1-bit DDR data input
   R => R,   -- 1-bit reset
   S => S    -- 1-bit set
   );
```

```
-- End of IDDR_inst instantiation
```

Verilog Instantiation Template

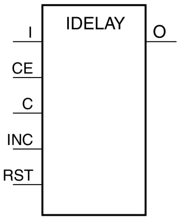
```
// IDDR: Input Double Data Rate Input Register with Set, Reset
//       and Clock Enable.
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IDDR #(
   .DDR_CLK_EDGE("OPPOSITE_EDGE"), // "OPPOSITE_EDGE", "SAME_EDGE"
                                   //    or "SAME_EDGE_PIPELINED"
   .INIT_Q1(1'b0), // Initial value of Q1: 1'b0 or 1'b1
   .INIT_Q2(1'b0), // Initial value of Q2: 1'b0 or 1'b1
   .SRTYPE("SYNC") // Set/Reset type: "SYNC" or "ASYNC"
) IDDR_inst (
   .Q1(Q1), // 1-bit output for positive edge of clock
   .Q2(Q2), // 1-bit output for negative edge of clock
   .C(C),   // 1-bit clock input
   .CE(CE), // 1-bit clock enable input
   .D(D),   // 1-bit DDR data input
   .R(R),   // 1-bit reset
   .S(S)    // 1-bit set
);

// End of IDDR_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IDELAY

Primitive: Input Delay Element



X10110

Introduction

Virtex® devices have an IDELAY module in the input path of every user I/O. IDELAY allows the implementation of deskew algorithms to correctly capture incoming data. IDELAY can be applied to data signals, clock signals, or both. IDELAY features a fully-controllable, 64-tap delay line. When used in conjunction with the IDELAYCTRL component circuitry, the IDELAY can provide precise time increments of delay independent of process, voltage, and temperature (PVT) variations. Three modes of operation are available:

- **Zero hold time delay mode -** This mode of operation allows backward compatibility for designs using the zero-hold time delay feature. When used in this mode, the IDELAYCTRL primitive does not need to be instantiated.
- **Fixed tap-delay mode -** In the fixed tap-delay mode, the delay value is set to the number determined by the attribute IOBDELAY_VALUE. This value cannot be changed during run-time. When used in this mode, the IDELAYCTRL primitive must be instantiated.
- **Variable tap-delay mode -** In the variable tap-delay mode, the delay value can be changed at run-time by manipulating the control signals CE and INC. When used in this mode, the IDELAYCTRL primitive must be instantiated.

Port Descriptions

Ports	Direction	Width	Function
I	Input	1	Serial input data from IOB
C	Input	1	Clock input
INC	Input	1	Increment/decrement number of tap delays
CE	Input	1	Enable increment/decrement function
RST	Input	1	Reset delay chain to pre-programmed value. If no value programmed, reset to 0
O	Output	1	Combinatorial output

Data Input and Output - I and O

IDELAY primitives are located in three different types of general purpose IOB locations. The input and output connectivity differs for each type of IOB location.

- **General Purpose IOBs -** The input of IDELAY in a general-purpose IOB comes directly from the input buffer, IBUF. The output of IDELAY (O) is connected directly to your logic. The input and output datapath is combinatorial and is not affected by the clock signal (C). However, you can choose to register the output signal (O) in the IOB.
- **Regional Clock-Capable IOBs -** Regional clock-capable IOBs are located in one I/O pair directly above and below an HCLK IOB. The input of IDELAY in a regional clock-capable IOB comes directly from the input buffer, IBUF. The output of IDELAY in a regional clock-capable IOB can go to one of the following locations:
  - Directly to your logic
  - BUFIO (in the case of a regional clock signal)

  The regional clock buffer, BUFIO, connects the incoming regional clock signal to the regional I/O clock tree, IOCLK. BUFIO also connects to the regional clock buffer, BUFR to connect to the regional clock tree, rclk. The input and output datapath is combinatorial and is not affected by the clock signal (C). However, you can choose to register the output signal (O) in the IOB.
- **Global clock-capable IOBs -** Global clock-capable IOBs are located in the center I/O column. The input of the IDELAY module in a global clock-capable IOB comes directly from the input global clock buffer, IBUFG. The output of the IDELAY module in a global clock-capable IOB can go to one of the following locations:
  - Directly to your logic
  - BUFG (in the case of a global clock signal)

  The global clock buffer, BUFG, connects the incoming regional clock signal to the global clock tree, gclk. The input and output datapath is combinatorial and is not affected by the clock signal (C). However, you can choose to register the output signal (O) in the IOB.

Clock Input - C

All control inputs to IDELAY (RST, CE and INC) are synchronous to the clock input (C). The data input and output (I and O) of IDELAY is not affected by this clock signal. This clock input is identical to the CLKDIV input for the ISERDES. All the clock sources used to drive CLKDIV can therefore drive the IDELAY clock input (C). The clock sources that can drive the clock input (C) are:

- Eight gclk (global clock tree)
- Two rclk (regional clock tree)

Module Reset - RST

The IDELAY reset signal, RST, resets the tap-delay line to a value set by the IOBDELAY_VALUE attribute. If the IOBDELAY_VALUE attribute is not specified, the tap-delay line is reset to 0.

Increment/Decrement Signals - CE, INC

The increment/decrement enable signal (CE) determines when the increment/decrement signal (INC) is activated. INC determines whether to increment or decrement the tap-delay line. When CE = 0, the tap delay remains constant no matter what the value of INC. When CE = 1, the tap-delay value increments or decrements depending on the value of INC. The tap delay is incremented or decremented synchronously with respect to the input clock (C). As long as CE = 1, the tap-delay increments or decrements by one every clock cycle. The increment/decrement operation is summarized in the following table:

Operation	RST	CE	INC
Reset to configured value of tap count	1	x	x
Increment tap count	0	1	1
Decrement tap count	0	1	0
No change	0	0	x

Note

1. RST resets delay chain to tap count specified by attribute IOBDELAY_VALUE. If IOBDELAY_VALUE is not specified, tap count reset to 0.
2. RST, CE, and INC are synchronous to the input clock signal (C).

When CE is raised, the increment/decrement operation begins on the next positive clock cycle. When CE is lowered, the increment/decrement operation ceases on the next positive clock cycle.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
IOBDELAY_TYPE	String	"DEFAULT", "FIXED", "VARIABLE"	"DEFAULT"	This attribute sets the type of tap delay.
IOBDELAY_VALUE	Integer	0 to 63	0	This attribute specifies the initial number of tap delays.

IOBDELAY_TYPE Attribute

The IOBDELAY_TYPE attribute sets the type of delay used. The attribute values are DEFAULT, FIXED, and VARIABLE. The default value is DEFAULT. When set to DEFAULT, the zero-hold time delay element is selected. This delay element eliminates pad-to-pad hold time. The delay is matched to the internal clock-distribution delay of the device. When used, it guarantees a pad-to-pad hold time of zero.

When set to FIXED, the tap-delay value is fixed at the number of taps determined by the IOBDELAY_VALUE attribute. This value is preset and cannot be changed dynamically.

When set to VARIABLE, the variable tap delay is selected. The tap delay can be incremented by setting CE = 1and INC = 1 or decremented by setting CE = 1 and INC = 0. The increment/decrement operation is synchronous to C, the input clock signal.

IOBDELAY_VALUE Attribute

The IOBDELAY_VALUE attribute specifies the initial number of tap delays. The possible values are any Integers from 0 to 63. The default value is 0. When set to 0, the total delay becomes the delay of the output MUX which is approximately 400 ps.

The value of the tap delay reverts to IOBDELAY_VALUE when the tap delay is reset (RST = 1), or the IOBDELAY_TYPE is set to FIXED.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IDELAY: Input Delay Element
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IDELAY_inst : IDELAY
generic map (
   IOBDELAY_TYPE => "FIXED", -- "FIXED" or "VARIABLE"
   IOBDELAY_VALUE => 0) -- Any value from 0 to 63
port map (
   O => O,      -- 1-bit output
   C => C,      -- 1-bit clock input
   CE => CE,    -- 1-bit clock enable input
   I => I,      -- 1-bit data input
   INC => INC, -- 1-bit increment input
   RST => RST  -- 1-bit reset input
);

-- End of IDELAY_inst instantiation
```

Verilog Instantiation Template

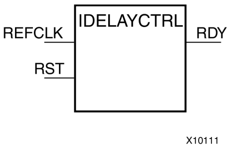
```
// IDELAY: Input Delay Element
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

(* IODELAY_GROUP = "<iodelay_group_name>" *) // Specifies group name for associated IDELAYs and IDELAYCTRL
IDELAY #(
   .IOBDELAY_TYPE("FIXED"), // "FIXED" or "VARIABLE"
   .IOBDELAY_VALUE(0)         // Any value from 0 to 63
) IDELAY_inst (
   .O(O),      // 1-bit output
   .C(C),      // 1-bit clock input
   .CE(CE),    // 1-bit clock enable input
   .I(I),      // 1-bit data input
   .INC(INC), // 1-bit increment input
   .RST(RST)  // 1-bit reset input
);

// End of IDELAY_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IDELAYCTRL

Primitive: IDELAY Tap Delay Value Control



Introduction

This design element must be instantiated when using the IODELAYE1. This occurs when the IDELAY or ISERDES primitive is instantiated with the IOBDELAY_TYPE attribute set to Fixed or Variable. The IDELAYCTRL module provides a voltage bias, independent of process, voltage, and temperature variations to the tap-delay line using a fixed-frequency reference clock, REFCLK. This enables very accurate delay tuning.

Port Descriptions

Port	Direction	Width	Function
RDY	Output	1	Indicates the validity of the reference clock input, REFCLK. When REFCLK disappears (i.e., REFCLK is held High or Low for one clock period or more), the RDY signal is deasserted.
REFCLK	Input	1	Provides a voltage bias, independent of process, voltage, and temperature variations, to the tap-delay lines in the IOBs. The frequency of REFCLK must be 200 MHz to guarantee the tap-delay value specified in the applicable data sheet.
RST	Input	1	Resets the IDELAYCTRL circuitry. The RST signal is an active-high asynchronous reset. To reset the IDELAYCTRL, assert it High for at least 50 ns.

RST (Module reset) - Resets the IDELAYCTRL circuitry. The RST signal is an active-high asynchronous reset. To reset the IDELAYCTRL, assert it High for at least 50 ns.

REFCLK (Reference Clock) - Provides a voltage bias, independent of process, voltage, and temperature variations, to the tap-delay lines in the IOBs. The frequency of REFCLK must be 200 MHz to guarantee the tap-delay value specified in the applicable data sheet.

RDY (Ready Output) - Indicates the validity of the reference clock input, REFCLK. When REFCLK disappears (i.e., REFCLK is held High or Low for one clock period or more), the RDY signal is deasserted.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IDELAYCTRL : Input Delay Element Control
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IDELAYCTRL_inst : IDELAYCTRL
port map (
   RDY => RDY,        -- 1-bit output indicates validity of the REFCLK
   REFCLK => REFCLK, -- 1-bit reference clock input
   RST => RST         -- 1-bit reset input
);

-- End of IDELAYCTRL_inst instantiation
```

Verilog Instantiation Template

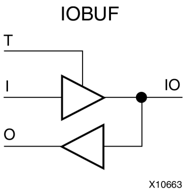
```
// IDELAYCTRL: Input Delay Control Element (Must be used in conjunction with the IDELAY
//             when used in FIXED or VARIABLE tap-delay mode)
//             Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

(* IODELAY_GROUP = "<iodelay_group_name>" *) // Specifies group name for associated IDELAYs and IDELAYCTRL
IDELAYCTRL IDELAYCTRL_inst (
   .RDY(RDY),       // 1-bit ready output
   .REFCLK(REFCLK), // 1-bit reference clock input
   .RST(RST)        // 1-bit reset input
);

// End of IDELAYCTRL_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IOBUF

Primitive: Bi-Directional Buffer



Introduction

The design element is a bidirectional single-ended I/O Buffer used to connect internal logic to an external bidirectional pin.

Logic Table

Inputs		Bidirectional	Outputs
T	**I**	**IO**	**O**
1	X	Z	IO
0	1	1	1
0	0	0	0

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Buffer output
IO	Inout	1	Buffer inout
I	Input	1	Buffer input
T	Input	1	3-State enable input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_ CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
DRIVE	Integer	2, 4, 6, 8, 12, 16, 24	12	Selects output drive strength (mA) for the SelectIO™ buffers that use the LVTTL, LVCMOS12, LVCMOS15, LVCMOS18, LVCMOS25, or LVCMOS33 interface I/O standard.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.
SLEW	String	"SLOW", "FAST", "QUIETIO"	"SLOW"	Sets the output rise and fall time. See the Data Sheet for recommendations of the best setting for this attribute.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IOBUF: Single-ended Bi-directional Buffer
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IOBUF_inst : IOBUF
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   DRIVE => 12,
   IOSTANDARD => "DEFAULT",
   SLEW => "SLOW")
port map (
   O => O,     -- Buffer output
   IO => IO,   -- Buffer inout port (connect directly to top-level port)
   I => I,     -- Buffer input
   T => T      -- 3-state enable input, high=input, low=output
);

-- End of IOBUF_inst instantiation
```

Verilog Instantiation Template

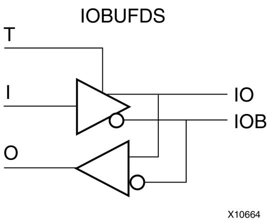
```
// IOBUF: Single-ended Bi-directional Buffer
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IOBUF #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .DRIVE(12), // Specify the output drive strength
   .IOSTANDARD("DEFAULT"), // Specify the I/O standard
   .SLEW("SLOW") // Specify the output slew rate
) IOBUF_inst (
   .O(O),     // Buffer output
   .IO(IO),   // Buffer inout port (connect directly to top-level port)
   .I(I),     // Buffer input
   .T(T)      // 3-state enable input, high=input, low=output
);

// End of IOBUF_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

IOBUFDS

Primitive: 3-State Differential Signaling I/O Buffer with Active Low Output Enable



Introduction

The design element is a bidirectional buffer that supports low-voltage, differential signaling. For the IOBUFDS, a design level interface signal is represented as two distinct ports (IO and IOB), one deemed the "master" and the other the "slave." The master and the slave are opposite phases of the same logical signal (for example, MYNET_P and MYNET_N). Optionally, a programmable differential termination feature is available to help improve signal integrity and reduce external components. Also available is a programmable delay is to assist in the capturing of incoming data to the device.

Logic Table

Inputs		Bidirectional		Outputs
I	**T**	**IO**	**IOB**	**O**
X	1	Z	Z	No Change
0	0	0	1	0
I	0	1	0	1

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Buffer output
IO	Inout	1	Diff_p inout
IOB	Inout	1	Diff_n inout
I	Input	1	Buffer input
T	Input	1	3-state enable input

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- IOBUFDS: Differential Bi-directional Buffer
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

IOBUFDS_inst : IOBUFDS
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   IOSTANDARD => "BLVDS_25")
port map (
   O => O,      -- Buffer output
   IO => IO,    -- Diff_p inout (connect directly to top-level port)
   IOB => IOB, -- Diff_n inout (connect directly to top-level port)
   I => I,      -- Buffer input
   T => T       -- 3-state enable input, high=input, low=output
);

-- End of IOBUFDS_inst instantiation
```

Verilog Instantiation Template

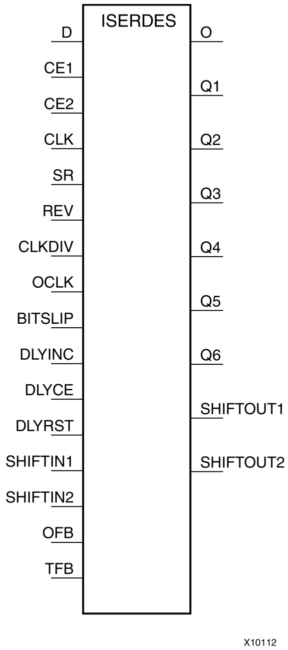
```
// IOBUFDS: Differential Bi-directional Buffer
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

IOBUFDS #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .IOSTANDARD("BLVDS_25")    // Specify the I/O standard
) IOBUFDS_inst (
   .O(O),     // Buffer output
   .IO(IO),   // Diff_p inout (connect directly to top-level port)
   .IOB(IOB), // Diff_n inout (connect directly to top-level port)
   .I(I),     // Buffer input
   .T(T)      // 3-state enable input, high=input, low=output
);

// End of IOBUFDS_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ISERDES

Primitive: Dedicated I/O Buffer Input Deserializer



Introduction

The ISERDES module provides a way for you to easily implement source synchronous solutions. ISERDES is a dedicated source synchronous I/O architecture. This module helps you by saving logic resources in the FPGA fabric for source synchronous applications. Furthermore, ISERDES also avoids additional timing complexities that can be encountered when designing such a solution in the FPGA fabric.

The ISERDES module contains or works in conjunction with the following modules: serial-to-parallel converters, serial delay chains, a word alignment unit (BITSLIP), and a clock enable (CE) module. In addition, ISERDES contains multiple clock inputs to accommodate various applications and works in conjunction with the SelectIO™ features. Following are descriptions of the ISERDES submodules.

Delay Chains Module

The Delay Chains module is a dedicated architecture that provides an adjustable or fixed timing relationship between input data and forwarded clock. This solution is achieved by placing delays in the ISERDES module that deskew the inputs. The input delay chains can be preprogrammed (fixed) or dynamically changed (variable). In addition this module works in conjunction with the IDELAYCTRL primitive.

A number of attributes are required in order to use the Delay Chains module. The attributes are as follows.

- IOBDELAY_VALUE
- IOBDELAY
- IOBDELAY_TYPE

IOBDELAY_VALUE can take values between 0 and 63. This attribute defines the number of delay taps used. Default value for this attribute is 0.

Setting the IOBDELAY attribute to "IBUF," "IFD," and "BOTH" allows the Delay Chains to be used in the combinatorial output (O output), registered output (Q1-Q6 output), and both respectively. Setting the IOBDELAY attribute to "NONE" bypasses the delay chains module.

The IOBDELAY_TYPE can take three different values: "DEFAULT", "FIXED", or "VARIABLE". "DEFAULT" lets you use the 0 hold time value. Using the "FIXED" mode, the delay taps equal to value defined by IOBDELAY_VALUE. In this mode, the value cannot be changed after the device is programmed. In the last mode, "VARIABLE," the delay value is set to an initial value defined by IOBDELAY_VALUE and adjustable after the device is programmed.

The Delay Chains module is controlled by DLYRST, DLYCE, and DLYINC pins. Each of the operations performed with these pins are synchronous to the CLKDIV clock signal. Asserting DLYRST to logic High configures the delay tap to the value defined in IOBDELAY_VALUE. To increment/decrement the delay tap value, you must use both DLYCE and DLYINC. For this operation to proceed, the DLYCE must be asserted to logic High. Setting DLYINC to 1 increments and setting DLYINC to 0 decrements the delay tap value.

The following table identifies the Delay Chains Controls.

Operation	DLYRST	DLYCE	DLYINC
Reset to IOBDELAY_VALUE	1	X	X
Increment tap value	0	1	1
Decrement tap value	0	1	0
No change	0	0	X

Note All Delay Chains operations are synchronous to CLKDIV.

Serial-to-Parallel Converter

The serial-to-parallel converter in the ISERDES module takes in serial data and convert them into data width choices from 2 to 6. Data widths larger than 6 (7, 8, and 10) is achievable by cascading two ISERDES modules for data width expansion. In order to do this, one ISERDES must be set into a MASTER mode, while another is set into SLAVE mode. Connect the SHIFTIN of "slave" and SHIFTOUT of "master" ports together. The "slave" uses Q3 to Q6 ports as its output. The serial-to-parallel converter is available for both SDR and DDR modes.

This module is primarily controlled by CLK and CLKDIV clocks. The following table describes the relationship between CLK and CLKDIV for both SDR and DDR mode.

The following table illustrates the CLK/CLKDIV relationship of the serial-to-parallel converter:

SDR Data Width	DDR Data Width	CLK	CLKDIV
2	4	2X	X
3	6	3X	X
4	8	4X	X
5	10	5X	X
6	-	6X	X
7	-	7X	X
8	-	8X	X

CE Module

CE Module is essentially a 2:1 parallel-to-serial converter. This module is controlled by CLKDIV clock input and is used to control the clock enable port of the Serial-to-Parallel Converter module.

BITSLIP Module

The BITSLIP module is a "Barrel Shifter" type function that reorders an output sequence. An output pattern only changes whenever the BITSLIP is invoked. The maximum number of BITSLIP reordering is always equal to the number of bits in the pattern length minus one (DATA_WIDTH - 1). BITSLIP is supported for both SDR and DDR operations. However, note that the output reordering for SDR and DDR greatly differs.

To use the BITSLIP, set the "BITSLIP_ENABLE" attribute to "ON." Setting this attribute to "OFF" allows you to bypass the BITSLIP module.

The BITSLIP operation is synchronous to the CLKDIV clock input. To invoke the BITSLIP module, the BITSLIP port must be asserted High for one and only one CLKDIV cycle. After one CLKDIV cycle the BITSLIP port is asserted High, the BITSLIP operation is complete. For DDR mode, a BITSLIP operation cannot be stable until after two CLKDIV cycles. All outputs of the BITSLIP appear in one of the registered output ports (Q1 to Q6) BITSLIP operations are synchronous to CLKDIV.

Additional Features

Width Expansion

It is possible to use the ISERDES modules to recover data widths larger than 6 bits. To use this feature, two ISERDES modules need to be instantiated. Both the ISERDES must be an adjacent master and slave pair. The attribute SERDES_MODE must be set to either "MASTER" or "SLAVE" in order to differentiate the modes of the ISERDES pair. In addition, you must connect the SHIFOUT ports of the MASTER to the SHIFTIN ports of the SLAVE. This feature supports data widths of 7, 8, and 10 for SDR and DDR mode. The table below lists the data width availability for SDR and DDR mode.

Mode	Widths
SDR Data Widths	2,3,4,5,6,7,8
DDR Data Widths	4,6,8,10

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Combinatorial Output - This port is an unregistered output of the ISERDES module. It is the unregistered output of the delay chain. In addition, this output port can also be configured to bypass all the submodules within ISERDES module. This output can be used to drive the BUFIOs.
Q1:6	Output	1 (each)	Registered Outputs - This port is a registered output of the ISERDES module. Using these outputs, you have a selection of the following combination of ISERDES submodules path as the inputs: • Delay chain to serial-to-parallel converter to BITSLIP module. • Delay chain to serial-to-parallel converter. These ports can be programmed from 2 to 6 bits. In the extended width mode, this port can be expanded up to 10 bits.
SHIFTOUT 1:2	Output	1 (each)	Carry out for data input expansion. Connect to SHIFTIN1/2 of slave.
BITSLIP	Input	1	Invokes the ISERDES to perform a BITSLIP operation when logic High is given and the BITSLIP module is enabled.
CE 1:2	Input	1 (each)	Clock enables input that feeds into the CE module.

Port	Direction	Width	Function
CLK	Input	1	High Speed Forwarded Clock Input - This clock input is used to drive the Serial to Parallel Converter and the BITSLIP module. The possible source for the CLK port is from one of the following clock resources: • Eight global clock lines in a clock region • Two regional clock lines • Six clock capable I/Os (within adjacent clock region) • Fabric (through bypass)
CLKDIV	Input	1	Divided High Speed Forward Clock Input - This clock input is used to drive the Serial to Parallel Converter, Delay Chain, the BITSLIP module, and CE module. This clock has to have slower frequency than the clock connected to the CLK port. The possible source for the CLKDIV port is from one of the following clock resources: • Eight global clock lines in a clock region • Two regional clock lines
D	Input	1	Serial Input Data From IOB - The D is where all the incoming data enters the ISERDES module. This port works in conjunction with SelectIO to accommodate the desired I/O standards.
DLYCE	Input	1	Enable delay chain to be incremented or decremented
DLYINC	Input	1	Delay Chain Increment/Decrement Pin - When the DLYCE pin is asserted High, the value at DLYINC pin increments/decrements the delay chain value. Logic High increments the tap value, while logic LOW decrements the tap value.
DLYRST	Input	1	Delay Chain Reset Pin - Resets delay line to programmed value of IOBDELAY_VALUE (=Tap Count). If no value programmed, resets delay line to 0 taps.
OCLK	Input	1	High Speed Clock for Memory Interfaces Applications - This clock input is used to drive the serial-to-parallel converter in the ISERDES module. The possible source for the OCLK port is from one of the following clock resources: • Eight global clock lines in a clock region • Two regional clock lines • Six clock capable I/Os (within adjacent clock region) • Fabric (through bypass) This clock is an ideal solution for memory interfaces in which strobe signals are required.
REV	Input	1	Reverse SR. For internal testing purposes. When SR is used, a second input, REV forces the storage element into the opposite state. The reset condition predominates over the set condition. The REV pin is not supported in ISERDES.
SR	Input	1	Set/Reset Input - The set/reset pin, SR forces the storage element into the state specified by the SRVAL attribute, set through your constraints file (UCF). SRVAL = 1 forces a logic 1. SRVAL = 0 forces a logic 0. When SR is used, a second input (REV) forces the storage element into the opposite state. The reset condition predominates over the set condition. The SR pin active high asynchronous reset for all registers in the ISERDES component.
SHIFTIN 1:2	Input	1 (each)	Carry input for data input expansion. Connect to SHIFTOUT1/2 of master.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
BITSLIP_ENABLE	Boolean	FALSE, TRUE	FALSE	Allows you to enable the bitslip controller.
DATA_RATE	String	"SDR" or "DDR"	"DDR"	Specify data rate of either allowed value.
DATA_WIDTH	String	If DATA_RATE = "DDR", value is limited to 4, 6, 8, or 10. If DATA_RATE = "SDR", value is limited to 2, 3, 4, 5, 6, 7, or 8.	4	Defines the serial-to-parallel converter width. This value also depends on the SDR vs. DDR and the Mode of the ISERDES.
INTERFACE_TYPE	String	"MEMORY" or "NETWORKING"	"MEMORY"	Determines which ISERDES use model is used.
IOBDELAY	String	"NONE", "IBUF", "IFD", "BOTH"	"NONE"	Defines where the ISERDES outputs the Delay Chains.
IOBDELAY_TYPE	String	"DEFAULT", "FIXED", or "VARIABLE"	"DEFAULT"	Defines whether the Delay Chains are in fixed or variable mode.
IOBDELAY_VALUE	Integer	0 to 63	0	Set initial tap delay to an Integer from 0 to 63.
NUM_CE	Integer	1 or 2	2	Define number or clock enables to an Integer of 1 or 2.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ISERDES: Input SERDES
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ISERDES_inst : ISERDES
generic map (
   BITSLIP_ENABLE => FALSE, -- TRUE/FALSE to enable bitslip controller
                             -- Must be "FALSE" in interface type is "MEMORY"
   DATA_RATE => "DDR", -- Specify data rate of "DDR" or "SDR"
   DATA_WIDTH => 4, -- Specify data width - For DDR 4,6,8, or 10
                    -- For SDR 2,3,4,5,6,7, or 8
   INTERFACE_TYPE => "MEMORY", -- Use model - "MEMORY" or "NETWORKING"
   IOBDELAY => "NONE", -- Specify outputs where delay chain will be applied
                       -- "NONE", "IBUF", "IFD", or "BOTH"
   IOBDELAY_TYPE => "DEFAULT", -- Set tap delay "DEFAULT", "FIXED", or "VARIABLE"
   IOBDELAY_VALUE => 0, -- Set initial tap delay to an integer from 0 to 63
   NUM_CE => 2, -- Define number or clock enables to an integer of 1 or 2
   SERDES_MODE => "MASTER") --Set SERDES mode to "MASTER" or "SLAVE"
```

```
port map (
   O => O,     -- 1-bit output
   Q1 => Q1,   -- 1-bit output
   Q2 => Q2,   -- 1-bit output
   Q3 => Q3,   -- 1-bit output
   Q4 => Q4,   -- 1-bit output
   Q5 => Q5,   -- 1-bit output
   Q6 => Q6,   -- 1-bit output
   SHIFTOUT1 => SHIFTOUT1, -- 1-bit output
   SHIFTOUT2 => SHIFTOUT2, -- 1-bit output
   BITSLIP => BITSLIP,     -- 1-bit input
   CE1 => CE1,        -- 1-bit input
   CE2 => CE2,        -- 1-bit input
   CLK => CLK,        -- 1-bit input
   CLKDIV => CLKDIV,  -- 1-bit input
   D => D,            -- 1-bit input
   DLYCE => DLYCE,    -- 1-bit input
   DLYINC => DLYINC,  -- 1-bit input
   DLYRST => DLYRST,  -- 1-bit input
   OCLK => OCLK,      -- 1-bit input
   REV => '0',        -- Must be tied to logic zero
   SHIFTIN1 => SHIFTIN1, -- 1-bit input
   SHIFTIN2 => SHIFTIN2, -- 1-bit input
   SR => SR           -- 1-bit input
);

-- End of ISERDES_inst instantiation
```

Verilog Instantiation Template

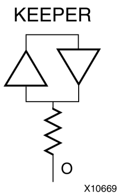
```
// ISERDES: Source Synchronous Input Deserializer
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ISERDES #(
   .BITSLIP_ENABLE("FALSE"), // "TRUE"/"FALSE" to enable bitslip controller
                             //    Must be "FALSE" if INTERFACE_TYPE set to "MEMORY"
   .DATA_RATE("DDR"), // Specify data rate of "DDR" or "SDR"
   .DATA_WIDTH(4), // Specify data width - for DDR 4,6,8, or 10
                   //    for SDR 2,3,4,5,6,7, or 8
   .INTERFACE_TYPE("MEMORY"), // Use model - "MEMORY" or "NETWORKING"
   .IOBDELAY("NONE"), // Specify outputs where delay chain will be applied
                      //    "NONE", "IBUF", "IFD", or "BOTH"
   .IOBDELAY_TYPE("DEFAULT"), // Set tap delay "DEFAULT", "FIXED", or "VARIABLE"
   .IOBDELAY_VALUE(0), // Set initial tap delay to an integer from 0 to 63
   .NUM_CE(2), // Define number or clock enables to an integer of 1 or 2
   .SERDES_MODE("MASTER")  // Set SERDES mode to "MASTER" or "SLAVE"
) ISERDES_inst (
   .O(O),    // 1-bit combinatorial output
   .Q1(Q1),  // 1-bit registered output
   .Q2(Q2),  // 1-bit registered output
   .Q3(Q3),  // 1-bit registered output
   .Q4(Q4),  // 1-bit registered output
   .Q5(Q5),  // 1-bit registered output
   .Q6(Q6),  // 1-bit registered output
   .SHIFTOUT1(SHIFTOUT1), // 1-bit carry output
   .SHIFTOUT2(SHIFTOUT2), // 1-bit carry output
   .BITSLIP(BITSLIP),     // 1-bit Bitslip input
   .CE1(CE1),        // 1-bit clock enable input
   .CE2(CE2),        // 1-bit clock enable input
   .CLK(CLK),        // 1-bit clock input
   .CLKDIV(CLKDIV),  // 1-bit divided clock input
   .D(D),            // 1-bit serial data input
   .DLYCE(DLYCE),    // 1-bit delay chain enable input
   .DLYINC(DLYINC),  // 1-bit delay increment/decrement input
   .DLYRST(DLYRST),  // 1-bit delay chain reset input
   .OCLK(OCLK),      // 1-bit high-speed clock input
   .REV(1'b0),       // Must be tied to logic zero
   .SHIFTIN1(SHIFTIN1), // 1-bit carry input
   .SHIFTIN2(SHIFTIN2), // 1-bit carry input
   .SR(SR)           // 1-bit set/reset input
);

// End of ISERDES_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

KEEPER

Primitive: KEEPER Symbol



Introduction

The design element is a weak keeper element that retains the value of the net connected to its bidirectional O pin. For example, if a logic 1 is being driven onto the net, KEEPER drives a weak/resistive 1 onto the net. If the net driver is then 3-stated, KEEPER continues to drive a weak/resistive 1 onto the net.

Port Descriptions

Name	Direction	Width	Function
O	Output	1-Bit	Keeper output

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- KEEPER: I/O Buffer Weak Keeper
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

KEEPER_inst : KEEPER
port map (
   O => O     -- Keeper output (connect directly to top-level port)
);

-- End of KEEPER_inst instantiation
```

Verilog Instantiation Template

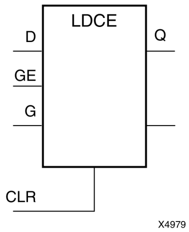
```
// KEEPER: I/O Buffer Weak Keeper
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

KEEPER KEEPER_inst (
   .O(O)     // Keeper output (connect directly to top-level port)
);

// End of KEEPER_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LDCE

Primitive: Transparent Data Latch with Asynchronous Clear and Gate Enable



Introduction

This design element is a transparent data latch with asynchronous clear and gate enable. When the asynchronous clear input (CLR) is High, it overrides the other inputs and resets the data (Q) output Low. Q reflects the data (D) input while the gate (G) input and gate enable (GE) are High and CLR is Low. If (GE) is Low, data on (D) cannot be latched. The data on the (D) input during the High-to-Low gate transition is stored in the latch. The data on the (Q) output remains unchanged as long as (G) or (GE) remains low.

This latch is asynchronously cleared, outputs Low, when power is applied. For FPGA devices, power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs				**Outputs**
CLR	**GE**	**G**	**D**	**Q**
1	X	X	X	0
0	0	X	X	No Change
0	1	1	D	D
0	1	0	X	No Change
0	1	↓	D	D

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0, 1	0	Sets the initial value of Q output after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LDCE: Transparent latch with Asynchronous Reset and
--       Gate Enable.
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LDCE_inst : LDCE
generic map (
   INIT => '0') -- Initial value of latch ('0' or '1')
port map (
   Q => Q,      -- Data output
   CLR => CLR,  -- Asynchronous clear/reset input
   D => D,      -- Data input
   G => G,      -- Gate input
   GE => GE     -- Gate enable input
);

-- End of LDCE_inst instantiation
```

Verilog Instantiation Template

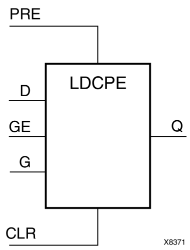
```
// LDCE:  Transparent latch with Asynchronous Reset and Gate Enable.
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LDCE #(
   .INIT(1'b0) // Initial value of latch (1'b0 or 1'b1)
) LDCE_inst (
   .Q(Q),      // Data output
   .CLR(CLR),  // Asynchronous clear/reset input
   .D(D),      // Data input
   .G(G),      // Gate input
   .GE(GE)     // Gate enable input
);

// End of LDCE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LDCPE

Primitive: Transparent Data Latch with Asynchronous Clear and Preset and Gate Enable



Introduction

This design element is a transparent data latch with data (D), asynchronous clear (CLR), asynchronous preset (PRE), and gate enable (GE). When (CLR) is High, it overrides the other inputs and resets the data (Q) output Low. When (PRE) is High and (CLR) is Low, it presets the data (Q) output High. Q reflects the data (D) input while the gate (G) input and gate enable (GE) are High and (CLR) and PRE are Low. The data on the (D) input during the High-to-Low gate transition is stored in the latch. The data on the Q output remains unchanged as long as (G) or (GE) remains Low.

This latch is asynchronously cleared, outputs Low, when power is applied. For FPGA devices, power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs					Outputs
CLR	**PRE**	**GE**	**G**	**D**	**Q**
1	X	X	X	X	0
0	1	X	X	X	1
0	0	0	X	X	No Change
0	0	1	1	0	0
0	0	1	1	1	1
0	0	1	0	X	No Change
0	0	1	↓	D	D

Port Descriptions

Port	Direction	Width	Function
Q	Output	1	Data Output
CLR	Input	1	Asynchronous clear/reset input
D	Input	1	Data Input
G	Input	1	Gate Input
GE	Input	1	Gate Enable Input
PRE	Input	1	Asynchronous preset/set input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Integer	0, 1	0	Sets the initial value of Q output after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LDCPE: Transparent latch with Asynchronous Reset, Preset and
--        Gate Enable.
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LDCPE_inst : LDCPE
generic map (
   INIT => '0') -- Initial value of latch ('0' or '1')
port map (
   Q => Q,      -- Data output
   CLR => CLR,  -- Asynchronous clear/reset input
   D => D,      -- Data input
   G => G,      -- Gate input
   GE => GE,    -- Gate enable input
   PRE => PRE   -- Asynchronous preset/set input
);

-- End of LDCPE_inst instantiation
```

Verilog Instantiation Template

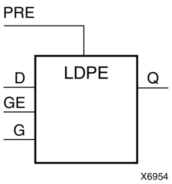
```
// LDCPE: Transparent latch with Asynchronous Reset, Preset and
//        Gate Enable.
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LDCPE #(
   .INIT(1'b0) // Initial value of latch (1'b0 or 1'b1)
) LDCPE_inst (
   .Q(Q),      // Data output
   .CLR(CLR),  // Asynchronous clear/reset input
   .D(D),      // Data input
   .G(G),      // Gate input
   .GE(GE),    // Gate enable input
   .PRE(PRE)   // Asynchronous preset/set input
);

// End of LDCPE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LDPE

Primitive: Transparent Data Latch with Asynchronous Preset and Gate Enable



Introduction

This design element is a transparent data latch with asynchronous preset and gate enable. When the asynchronous preset (PRE) is High, it overrides the other input and presets the data (Q) output High. Q reflects the data (D) input while the gate (G) input and gate enable (GE) are High. The data on the (D) input during the High-to-Low gate transition is stored in the latch. The data on the (Q) output remains unchanged as long as (G) or (GE) remains Low.

The latch is asynchronously preset, output High, when power is applied. For FPGA devices, power-on conditions are simulated when global set/reset (GSR) is active. GSR defaults to active-High but can be inverted by adding an inverter in front of the GSR input of the appropriate STARTUP_*architecture* symbol.

Logic Table

Inputs				Outputs
PRE	**GE**	**G**	**D**	**Q**
1	X	X	X	1
0	0	X	X	No Change
0	1	1	D	D
0	1	0	X	No Change
0	1	↓	D	D

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Binary	0, 1	1	Specifies the initial value upon power-up or the assertion of GSR for the (Q) port.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LDPE:  Transparent latch with Asynchronous Set and
--        Gate Enable.
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LDPE_inst : LDPE
generic map (
   INIT => '0') -- Initial value of latch ('0' or '1')
port map (
   Q => Q,      -- Data output
   CLR => CLR,  -- Asynchronous preset/set input
   D => D,      -- Data input
   G => G,      -- Gate input
   GE => GE     -- Gate enable input
);

-- End of LDPE_inst instantiation
```

Verilog Instantiation Template

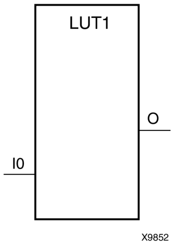
```
// LDPE:  Transparent latch with Asynchronous Preset and Gate Enable.
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LDPE #(
   .INIT(1'b0) // Initial value of latch (1'b0 or 1'b1)
) LDPE_inst (
   .Q(Q),      // Data output
   .PRE(PRE),  // Asynchronous preset/set input
   .D(D),      // Data input
   .G(G),      // Gate input
   .GE(GE)     // Gate enable input
);

// End of LDPE_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT1

Primitive: 1-Bit Look-Up Table with General Output



Introduction

This design element is a 1-bit look-up table (LUT) with general output (O).

An INIT attribute with an appropriate number of hexadecimal digits for the number of inputs must be attached to the LUT to specify its function. This element provides a look-up table version of a buffer or inverter. These elements are the basic building blocks. Two LUTs are available in each CLB slice; four LUTs are available in each CLB. Multiple variants of LUTs accommodate additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs	Outputs
I0	**O**
0	INIT[0]
1	INIT[1]
INIT = Binary number assigned to the INIT attribute	

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 2-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT1: 1-input Look-Up Table with general output
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT1_inst : LUT1
generic map (
   INIT => "00")
port map (
   O => O,   -- LUT general output
   I0 => I0  -- LUT input
);

-- End of LUT1_inst instantiation
```

Verilog Instantiation Template

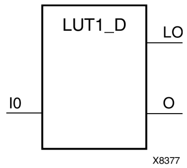
```
// LUT1: 1-input Look-Up Table with general output
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT1 #(
   .INIT(2'b00)  // Specify LUT Contents
) LUT1_inst (
   .O(O),   // LUT general output
   .I0(I0)  // LUT input
);

// End of LUT1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT1_D

Primitive: 1-Bit Look-Up Table with Dual Output



Introduction

This design element is a 1-bit look-up table (LUT) with two functionally identical outputs, O and LO. It provides a look-up table version of a buffer or inverter.

The O output is a general interconnect. The LO output is used to connect to another input within the same CLB slice and to the fast connect buffer. A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

- **The Logic Table Method -**A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.
- **The Equation Method -**Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs	Outputs	
I0	**O**	**LO**
0	INIT[0]	INIT[0]
1	INIT[1]	INIT[1]
INIT = Binary number assigned to the INIT attribute		

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 2-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT1_D: 1-input Look-Up Table with general and local outputs
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT1_D_inst : LUT1_D
generic map (
   INIT => "00")
port map (
   LO => LO, -- LUT local output
   O => O,   -- LUT general output
   I0 => I0  -- LUT input
);

-- End of LUT1_D_inst instantiation
```

Verilog Instantiation Template

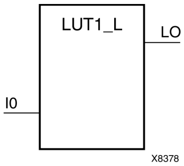
```
// LUT1_D: 1-input Look-Up Table with general and local outputs
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT1_D #(
   .INIT(2'b00)  // Specify LUT Contents
) LUT1_D_inst (
   .LO(LO), // LUT local output
   .O(O),   // LUT general output
   .I0(I0)  // LUT input
);

// End of LUT1_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT1_L

Primitive: 1-Bit Look-Up Table with Local Output



Introduction

This design element is a 1-bit look-up table (LUT) with a local output (LO) that is used to connect to another output within the same CLB slice and to the fast connect buffer. It provides a look-up table version of a buffer or inverter.

A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs	**Outputs**
I0	**LO**
0	INIT[0]
1	INIT[1]
INIT = Binary number assigned to the INIT attribute	

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 2-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT1_L: 1-input Look-Up Table with local output
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT1_L_inst : LUT1_L
generic map (
   INIT => "00")
port map (
   LO => LO, -- LUT local output
   I0 => I0  -- LUT input
);

-- End of LUT1_L_inst instantiation
```

Verilog Instantiation Template

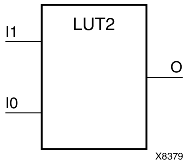
```
// LUT1_L: 1-input Look-Up Table with local output
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT1_L #(
   .INIT(2'b00)  // Specify LUT Contents
) LUT1_L_inst (
   .LO(LO), // LUT local output
   .I0(I0)  // LUT input
);

// End of LUT1_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT2

Primitive: 2-Bit Look-Up Table with General Output



Introduction

This design element is a 2-bit look-up table (LUT) with general output (O).

An INIT attribute with an appropriate number of hexadecimal digits for the number of inputs must be attached to the LUT to specify its function. This element provides a look-up table version of a buffer or inverter. These elements are the basic building blocks. Two LUTs are available in each CLB slice; four LUTs are available in each CLB. Multiple variants of LUTs accommodate additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs		Outputs
I1	**I0**	**O**
0	0	INIT[0]
0	1	INIT[1]
1	0	INIT[2]
1	1	INIT[3]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute		

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 4-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT2: 2-input Look-Up Table with general output
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT2_inst : LUT2
generic map (
   INIT => X"0")
port map (
   O => O,   -- LUT general output
   I0 => I0, -- LUT input
   I1 => I1  -- LUT input
);

-- End of LUT2_inst instantiation
```

Verilog Instantiation Template

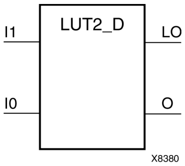
```
// LUT2: 2-input Look-Up Table with general output
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT2 #(
   .INIT(4'h0)  // Specify LUT Contents
) LUT2_inst (
   .O(O),   // LUT general output
   .I0(I0), // LUT input
   .I1(I1)  // LUT input
);

// End of LUT2_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT2_D

Primitive: 2-Bit Look-Up Table with Dual Output



Introduction

This design element is a 2-bit look-up table (LUT) with two functionally identical outputs, O and LO.

The O output is a general interconnect. The LO output is used to connect to another input within the same CLB slice and to the fast connect buffer. A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

- **The Logic Table Method -**A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.
- **The Equation Method -**Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs		Outputs	
I1	**I0**	**O**	**LO**
0	0	INIT[0]	INIT[0]
0	1	INIT[1]	INIT[1]
1	0	INIT[2]	INIT[2]
1	1	INIT[3]	INIT[3]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 4-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT2_D: 2-input Look-Up Table with general and local outputs
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT2_D_inst : LUT2_D
generic map (
   INIT => X"0")
port map (
   LO => LO, -- LUT local output
   O => O,   -- LUT general output
   I0 => I0, -- LUT input
   I1 => I1  -- LUT input
);

-- End of LUT2_D_inst instantiation
```

Verilog Instantiation Template

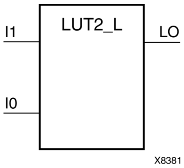
```
// LUT2_D: 2-input Look-Up Table with general and local outputs
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT2_D #(
   .INIT(4'h0)  // Specify LUT Contents
) LUT2_D_inst (
   .LO(LO), // LUT local output
   .O(O),   // LUT general output
   .I0(I0), // LUT input
   .I1(I1)  // LUT input
);

// End of LUT2_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT2_L

Primitive: 2-Bit Look-Up Table with Local Output



Introduction

This design element is a 2-bit look-up table (LUT) with a local output (LO) that is used to connect to another output within the same CLB slice and to the fast connect buffer. It provides a look-up table version of a buffer or inverter.

A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs		Outputs
I1	**I0**	**LO**
0	0	INIT[0]
0	1	INIT[1]
1	0	INIT[2]
1	1	INIT[3]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute		

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 4-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT2_L: 2-input Look-Up Table with local output
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT2_L_inst : LUT2_L
generic map (
   INIT => X"0")
port map (
   LO => LO, -- LUT local output
   I0 => I0, -- LUT input
   I1 => I1  -- LUT input
);

-- End of LUT2_L_inst instantiation
```

Verilog Instantiation Template

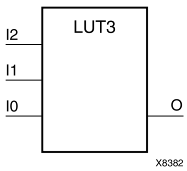
```
// LUT2_L: 2-input Look-Up Table with local output
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT2_L #(
   .INIT(4'h0)  // Specify LUT Contents
) LUT2_L_inst (
   .LO(LO), // LUT local output
   .I0(I0), // LUT input
   .I1(I1)  // LUT input
);

// End of LUT2_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT3

Primitive: 3-Bit Look-Up Table with General Output



Introduction

This design element is a 3-bit look-up table (LUT) with general output (O). A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

An INIT attribute with an appropriate number of hexadecimal digits for the number of inputs must be attached to the LUT to specify its function. This element provides a look-up table version of a buffer or inverter. These elements are the basic building blocks. Two LUTs are available in each CLB slice; four LUTs are available in each CLB. Multiple variants of LUTs accommodate additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs			**Outputs**
I2	**I1**	**I0**	**O**
0	0	0	INIT[0]
0	0	1	INIT[1]
0	1	0	INIT[2]
0	1	1	INIT[3]
1	0	0	INIT[4]
1	0	1	INIT[5]
1	1	0	INIT[6]
1	1	1	INIT[7]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 8-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT3: 3-input Look-Up Table with general output
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT3_inst : LUT3
generic map (
   INIT => X"00")
port map (
   O => O,    -- LUT general output
   I0 => I0, -- LUT input
   I1 => I1, -- LUT input
   I2 => I2  -- LUT input
);

-- End of LUT3_inst instantiation
```

Verilog Instantiation Template

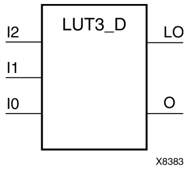
```
// LUT3: 3-input Look-Up Table with general output
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT3 #(
   .INIT(8'h00)  // Specify LUT Contents
) LUT3_inst (
   .O(O),    // LUT general output
   .I0(I0), // LUT input
   .I1(I1), // LUT input
   .I2(I2)  // LUT input
);

// End of LUT3_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT3_D

Primitive: 3-Bit Look-Up Table with Dual Output



Introduction

This design element is a 3-bit look-up table (LUT) with two functionally identical outputs, O and LO.

The O output is a general interconnect. The LO output is used to connect to another input within the same CLB slice and to the fast connect buffer. A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

- **The Logic Table Method -**A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.
- **The Equation Method -**Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs			**Outputs**	
I2	**I1**	**I0**	**O**	**LO**
0	0	0	INIT[0]	INIT[0]
0	0	1	INIT[1]	INIT[1]
0	1	0	INIT[2]	INIT[2]
0	1	1	INIT[3]	INIT[3]
1	0	0	INIT[4]	INIT[4]
1	0	1	INIT[5]	INIT[5]
1	1	0	INIT[6]	INIT[6]
1	1	1	INIT[7]	INIT[7]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute				

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 8-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT3_D: 3-input Look-Up Table with general and local outputs
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT3_D_inst : LUT3_D
generic map (
   INIT => X"00")
port map (
   LO => LO,   -- LUT local output
   O => O,     -- LUT general output
   I0 => I0,   -- LUT input
   I1 => I1,   -- LUT input
   I2 => I2    -- LUT input
);

-- End of LUT3_D_inst instantiation
```

Verilog Instantiation Template

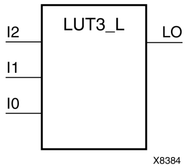
```
// LUT3_D: 3-input Look-Up Table with general and local outputs
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT3_D #(
   .INIT(8'h00)  // Specify LUT Contents
) LUT3_D_inst (
   .LO(LO), // LUT local output
   .O(O),   // LUT general output
   .I0(I0),   // LUT input
   .I1(I1),   // LUT input
   .I2(I2)    // LUT input
);

// End of LUT3_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT3_L

Primitive: 3-Bit Look-Up Table with Local Output



Introduction

This design element is a 3-bit look-up table (LUT) with a local output (LO) that is used to connect to another output within the same CLB slice and to the fast connect buffer. It provides a look-up table version of a buffer or inverter.

A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs			**Outputs**
I2	**I1**	**I0**	**LO**
0	0	0	INIT[0]
0	0	1	INIT[1]
0	1	0	INIT[2]
0	1	1	INIT[3]
1	0	0	INIT[4]
1	0	1	INIT[5]
1	1	0	INIT[6]
1	1	1	INIT[7]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 8-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT3_L: 3-input Look-Up Table with local output
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT3_L_inst : LUT3_L
generic map (
   INIT => X"00")
port map (
   LO => LO,    -- LUT local output
   I0 => I0,    -- LUT input
   I1 => I1,    -- LUT input
   I2 => I2     -- LUT input
);

-- End of LUT3_L_inst instantiation
```

Verilog Instantiation Template

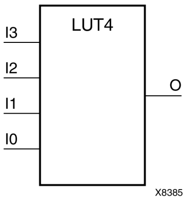
```
// LUT3_L: 3-input Look-Up Table with local output
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT3_L #(
   .INIT(8'h00)  // Specify LUT Contents
) LUT3_L_inst (
   .LO(LO), // LUT local output
   .I0(I0), // LUT input
   .I1(I1), // LUT input
   .I2(I2)  // LUT input
);

// End of LUT3_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT4

Primitive: 4-Bit Look-Up-Table with General Output



Introduction

This design element is a 4-bit look-up table (LUT) with general output (O).

An INIT attribute with an appropriate number of hexadecimal digits for the number of inputs must be attached to the LUT to specify its function. This element provides a look-up table version of a buffer or inverter. These elements are the basic building blocks. Two LUTs are available in each CLB slice; four LUTs are available in each CLB. Multiple variants of LUTs accommodate additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs				Outputs
I3	**I2**	**I1**	**I0**	**O**
0	0	0	0	INIT[0]
0	0	0	1	INIT[1]
0	0	1	0	INIT[2]
0	0	1	1	INIT[3]
0	1	0	0	INIT[4]
0	1	0	1	INIT[5]
0	1	1	0	INIT[6]
0	1	1	1	INIT[7]
1	0	0	0	INIT[8]
1	0	0	1	INIT[9]
1	0	1	0	INIT[10]
1	0	1	1	INIT[11]

Inputs				Outputs
I3	I2	I1	I0	O
1	1	0	0	INIT[12]
1	1	0	1	INIT[13]
1	1	1	0	INIT[14]
1	1	1	1	INIT[15]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute				

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT4: 4-input Look-Up Table with general output
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT4_inst : LUT4
generic map (
   INIT => X"0000")
port map (
   O => O,    -- LUT general output
   I0 => I0, -- LUT input
   I1 => I1, -- LUT input
   I2 => I2, -- LUT input
   I3 => I3  -- LUT input
);

-- End of LUT4_inst instantiation
```

Verilog Instantiation Template

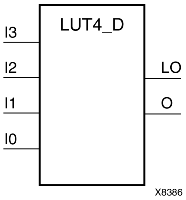
```
// LUT4: 4-input Look-Up Table with general output
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT4 #(
   .INIT(16'h0000)  // Specify LUT Contents
) LUT4_inst (
   .O(O),   // LUT general output
   .I0(I0), // LUT input
   .I1(I1), // LUT input
   .I2(I2), // LUT input
   .I3(I3)  // LUT input
);

// End of LUT4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT4_D

Primitive: 4-Bit Look-Up Table with Dual Output



Introduction

This design element is a 4-bit look-up table (LUT) with two functionally identical outputs, O and LO

The O output is a general interconnect. The LO output is used to connect to another input within the same CLB slice and to the fast connect buffer. A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

- **The Logic Table Method -**A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.
- **The Equation Method -**Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation. This method is easier to understand once you have grasped the concept and is more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs				Outputs	
I3	**I2**	**I1**	**I0**	**O**	**LO**
0	0	0	0	INIT[0]	INIT[0]
0	0	0	1	INIT[1]	INIT[1]
0	0	1	0	INIT[2]	INIT[2]
0	0	1	1	INIT[3]	INIT[3]
0	1	0	0	INIT[4]	INIT[4]
0	1	0	1	INIT[5]	INIT[5]
0	1	1	0	INIT[6]	INIT[6]
0	1	1	1	INIT[7]	INIT[7]
1	0	0	0	INIT[8]	INIT[8]
1	0	0	1	INIT[9]	INIT[9]
1	0	1	0	INIT[10]	INIT[10]
1	0	1	1	INIT[11]	INIT[11]
1	1	0	0	INIT[12]	INIT[12]
1	1	0	1	INIT[13]	INIT[13]

Inputs				Outputs	
I3	**I2**	**I1**	**I0**	**O**	**LO**
1	1	1	0	INIT[14]	INIT[14]
1	1	1	1	INIT[15]	INIT[15]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute					

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT4_D: 4-input Look-Up Table with general and local outputs
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT4_D_inst : LUT4_D
generic map (
   INIT => X"0000")
port map (
   LO => LO, -- LUT local output
   O => O,   -- LUT general output
   I0 => I0, -- LUT input
   I1 => I1, -- LUT input
   I2 => I2, -- LUT input
   I3 => I3  -- LUT input
);

-- End of LUT4_D_inst instantiation
```

Verilog Instantiation Template

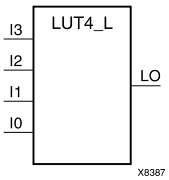
```
// LUT4_D: 4-input Look-Up Table with general and local outputs
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT4_D #(
   .INIT(16'h0000)  // Specify LUT Contents
) LUT4_D_inst (
   .LO(LO), // LUT local output
   .O(O),   // LUT general output
   .I0(I0), // LUT input
   .I1(I1), // LUT input
   .I2(I2), // LUT input
   .I3(I3)  // LUT input
);

// End of LUT4_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

LUT4_L

Primitive: 4-Bit Look-Up Table with Local Output



Introduction

This design element is a 4-bit look-up table (LUT) with a local output (LO) that is used to connect to another output within the same CLB slice and to the fast connect buffer. It provides a look-up table version of a buffer or inverter.

A mandatory INIT attribute, with an appropriate number of hexadecimal digits for the number of inputs, must be attached to the LUT to specify its function.

The INIT parameter for the FPGA LUT primitive is what gives the LUT its logical value. By default, this value is zero, thus driving the output to a zero regardless of the input values (acting as a ground). However, in most cases a new INIT value must be determined in order to specify the logic function for the LUT primitive. There are at least two methods by which the LUT value can be determined:

The Logic Table Method -A common method to determine the desired INIT value for a LUT is using a logic table. To do so, simply create a binary logic table of all possible inputs, specify the desired logic value of the output and then create the INIT string from those output values.

The Equation Method -Another method to determine the LUT value is to define parameters for each input to the LUT that correspond to their listed truth value and use those to build the logic equation you are after. This method is easier to understand once you have grasped the concept and more self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

Inputs				Outputs
I3	**I2**	**I1**	**I0**	**LO**
0	0	0	0	INIT[0]
0	0	0	1	INIT[1]
0	0	1	0	INIT[2]
0	0	1	1	INIT[3]
0	1	0	0	INIT[4]
0	1	0	1	INIT[5]
0	1	1	0	INIT[6]
0	1	1	1	INIT[7]
1	0	0	0	INIT[8]
1	0	0	1	INIT[9]
1	0	1	0	INIT[10]
1	0	1	1	INIT[11]
1	1	0	0	INIT[12]

Inputs				Outputs
I3	I2	I1	I0	LO
1	1	0	1	INIT[13]
1	1	1	0	INIT[14]
1	1	1	1	INIT[15]
INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute				

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Initializes look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- LUT4_L: 4-input Look-Up Table with local output
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

LUT4_L_inst : LUT4_L
generic map (
   INIT => X"0000")
port map (
   LO => LO, -- LUT local output
   I0 => I0, -- LUT input
   I1 => I1, -- LUT input
   I2 => I2, -- LUT input
   I3 => I3  -- LUT input
);

-- End of LUT4_L_inst instantiation
```

Verilog Instantiation Template

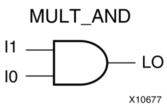
```
// LUT4_L: 4-input Look-Up Table with local output
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

LUT4_L #(
   .INIT(16'h0000)  // Specify LUT Contents
) LUT4_L_inst (
   .LO(LO), // LUT local output
   .I0(I0), // LUT input
   .I1(I1), // LUT input
   .I2(I2), // LUT input
   .I3(I3)  // LUT input
);

// End of LUT4_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MULT_AND

Primitive: Fast Multiplier AND



Introduction

The design element is an AND component located within the slice where the two inputs are shared with the 4-input LUT and the output drives into the carry logic. This added logic is especially useful for building fast and smaller multipliers. However, it can be used for other purposes as well. The I1 and I0 inputs must be connected to the I1 and I0 inputs of the associated LUT. The LO output must be connected to the DI input of the associated MUXCY, MUXCY_D, or MUXCY_L.

Logic Table

Inputs		Outputs
I1	**I0**	**LO**
0	0	0
0	1	0
1	0	0
1	1	1

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MULT_AND: 2-input AND gate connected to Carry chain
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MULT_AND_inst : MULT_AND
port map (
   LO => LO,   -- MULT_AND output (connect to MUXCY DI)
   I0 => I0,   -- MULT_AND data[0] input
   I1 => I1    -- MULT_AND data[1] input
);

-- End of MULT_AND_inst instantiation
```

Verilog Instantiation Template

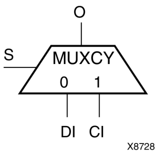
```
// MULT_AND: 2-input AND gate connected to Carry chain
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MULT_AND MULT_AND_inst (
   .LO(LO),   // MULT_AND output (connect to MUXCY DI)
   .I0(I0),   // MULT_AND data[0] input
   .I1(I1)    // MULT_AND data[1] input
);

// End of MULT_AND_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXCY

Primitive: 2-to-1 Multiplexer for Carry Logic with General Output



Introduction

The direct input (DI) of a slice is connected to the (DI) input of the MUXCY. The carry in (CI) input of an LC is connected to the CI input of the MUXCY. The select input (S) of the MUXCY is driven by the output of the look-up table (LUT) and configured as a MUX function. The carry out (O) of the MUXCY reflects the state of the selected input and implements the carry out function of each LC. When Low, S selects DI; when High, S selects CI.

The variants MUXCY_D and MUXCY_L provide additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

Logic Table

Inputs			Outputs
S	**DI**	**CI**	**O**
0	1	X	1
0	0	X	0
1	X	1	1
1	X	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXCY: Carry-Chain MUX with general output
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXCY_inst : MUXCY
port map (
   O => O,    -- Carry output signal
   CI => CI,  -- Carry input signal
   DI => DI,  -- Data input signal
   S => S     -- MUX select, tie to '1' or LUT4 out
);

-- End of MUXCY_inst instantiation
```

Verilog Instantiation Template

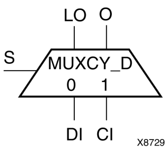
```
// MUXCY: Carry-Chain MUX with general output
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXCY MUXCY_inst (
   .O(O),    // Carry output signal
   .CI(CI), // Carry input signal
   .DI(DI), // Data input signal
   .S(S)    // MUX select, tie to '1' or LUT4 out
);

// End of MUXCY_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXCY_D

Primitive: 2-to-1 Multiplexer for Carry Logic with Dual Output



Introduction

This design element implements a 1-bit, high-speed carry propagate function. One such function can be implemented per logic cell (LC), for a total of 4-bits per configurable logic block (CLB). The direct input (DI) of an LC is connected to the DI input of the MUXCY_D. The carry in (CI) input of an LC is connected to the CI input of the MUXCY_D. The select input (S) of the MUX is driven by the output of the look-up table (LUT) and configured as an XOR function. The carry out (O and LO) of the MUXCY_D reflects the state of the selected input and implements the carry out function of each LC. When Low, S selects DI; when High, S selects CI.

Outputs O and LO are functionally identical. The O output is a general interconnect. See also MUXCY and MUXCY_L.

Logic Table

Inputs			Outputs	
S	**DI**	**CI**	**O**	**LO**
0	1	X	1	1
0	0	X	0	0
1	X	1	1	1
1	X	0	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXCY_D: Carry-Chain MUX with general and local outputs
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXCY_D_inst : MUXCY_D
port map (
   LO => LO, -- Carry local output signal
   O => O,   -- Carry general output signal
   CI => CI, -- Carry input signal
   DI => DI, -- Data input signal
```

```
   S => S    -- MUX select, tie to '1' or LUT4 out
);

-- End of MUXCY_D_inst instantiation
```

Verilog Instantiation Template

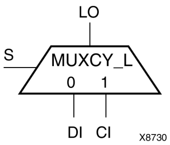
```
// MUXCY_D: Carry-Chain MUX with general and local outputs
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXCY_D MUXCY_D_inst (
   .LO(LO), // Carry local output signal
   .O(O),   // Carry general output signal
   .CI(CI), // Carry input signal
   .DI(DI), // Data input signal
   .S(S)    // MUX select, tie to '1' or LUT4 out
);

// End of MUXCY_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXCY_L

Primitive: 2-to-1 Multiplexer for Carry Logic with Local Output



Introduction

This design element implements a 1-bit high-speed carry propagate function. One such function is implemented per logic cell (LC), for a total of 4-bits per configurable logic block (CLB). The direct input (DI) of an LC is connected to the DI input of the MUXCY_L. The carry in (CI) input of an LC is connected to the CI input of the MUXCY_L. The select input (S) of the MUXCY_L is driven by the output of the look-up table (LUT) and configured as an XOR function. The carry out (LO) of the MUXCY_L reflects the state of the selected input and implements the carry out function of each (LC). When Low, (S) selects DI; when High, (S) selects (CI).

See also MUXCY and MUXCY_D.

Logic Table

Inputs			Outputs
S	**DI**	**CI**	**LO**
0	1	X	1
0	0	X	0
1	X	1	1
1	X	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXCY_L: Carry-Chain MUX with local output
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXCY_L_inst : MUXCY_L
port map (
   LO => LO, -- Carry local output signal
   CI => CI, -- Carry input signal
   DI => DI, -- Data input signal
   S => S    -- MUX select, tie to '1' or LUT4 out
);
```

```
-- End of MUXCY_L_inst instantiation
```

Verilog Instantiation Template

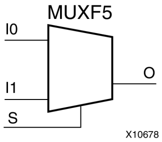
```
// MUXCY_L: Carry-Chain MUX with local output
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXCY_L MUXCY_L_inst (
   .LO(LO), // Carry local output signal
   .CI(CI), // Carry input signal
   .DI(DI), // Data input signal
   .S(S)    // MUX select, tie to '1' or LUT4 out
);

// End of MUXCY_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF5

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element is a two input multiplexer for creating a function-of-5 look-up table or a 4-to-1 multiplexer when connected to LUT4 look-up tables. The local outputs (LO) from two LUT4 look-up tables are connected to the I0 and I1 inputs of the MUXF5. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The O output is a general interconnect.

The variants MUXF5_D and MUXF5_L provide additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

Logic Table

Inputs			Outputs
S	**I0**	**I1**	**O**
0	1	X	1
0	0	X	0
1	X	1	1
1	X	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF5: Slice MUX to tie two LUT4's together with general output
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF5_inst : MUXF5
port map (
   O => O,    -- Output of MUX to general routing
   I0 => I0,  -- Input (tie directly to the output of LUT4)
   I1 => I1,  -- Input (tie directoy to the output of LUT4)
   S => S     -- Input select to MUX
);
```

```
-- End of MUXF5_inst instantiation
```

Verilog Instantiation Template

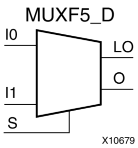
```
// MUXF5: Slice MUX to tie two LUT4's together with general output
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF5 MUXF5_inst (
   .O(O),    // Output of MUX to general routing
   .I0(I0),  // Input (tie directly to the output of LUT4)
   .I1(I1),  // Input (tie directoy to the output of LUT4)
   .S(S)     // Input select to MUX
);

// End of MUXF5_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF5_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element is a two input multiplexer for creating a function-of-5 look-up table or a 4-to-1 multiplexer when connected to LUT4 look-up tables. The local outputs (LO) from two LUT4 look-up tables are connected to the I0 and I1 inputs of the MUXF5. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

Outputs O and LO are functionally identical. The O output is a general interconnect. The LO output connects to other inputs in the same CLB slice.

See also MUXF5 and MUXF5_L.

Logic Table

Inputs			Outputs	
S	**I0**	**I1**	**O**	**LO**
0	1	X	1	1
0	0	X	0	0
1	X	1	1	1
1	X	0	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF5_D: Slice MUX to tie two LUT4's together with general and local outputs
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF5_D_inst : MUXF5_D
port map (
   LO => LO,   -- Output of MUX to local routing
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie directly to the output of LUT4)
   I1 => I1,   -- Input (tie directoy to the output of LUT4)
   S => S      -- Input select to MUX
```

```
);

-- End of MUXF5_D_inst instantiation
```

Verilog Instantiation Template

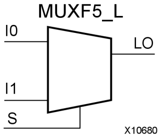
```
// MUXF5_D: Slice MUX to tie two LUT4's together with general and local outputs
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF5_D MUXF5_D_inst (
   .LO(LO),  // Output of MUX to local routing
   .O(O),    // Output of MUX to general routing
   .I0(I0),  // Input (tie directly to the output of LUT4)
   .I1(I1),  // Input (tie directoy to the output of LUT4)
   .S(S)     // Input select to MUX
);

// End of MUXF5_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF5_L

Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output



Introduction

This design element is a two input multiplexer for creating a function-of-5 look-up table or a 4-to-1 multiplexer when connected to LUT4 look-up tables. The local outputs (LO) from two LUT4 look-up tables are connected to the I0 and I1 inputs of the MUXF5. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The LO output connects to other inputs in the same CLB slice.

See also MUXF5 and MUXF5_D.

Logic Table

Inputs			**Output**
S	**I0**	**I1**	**LO**
0	1	X	1
0	0	X	0
1	X	1	1
1	X	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF5_L: Slice MUX to tie two LUT4's together with local output
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF5_L_inst : MUXF5_L
port map (
   LO => LO,   -- Output of MUX to local routing
   I0 => I0,   -- Input (tie directly to the output of LUT4)
   I1 => I1,   -- Input (tie directoy to the output of LUT4)
   S => S      -- Input select to MUX
);

-- End of MUXF5_L_inst instantiation
```

Verilog Instantiation Template

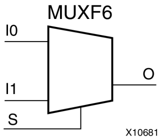
```
// MUXF5_L: Slice MUX to tie two LUT4's together with local output
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF5_L MUXF5_L_inst (
   .LO(LO),  // Output of MUX to local routing
   .I0(I0),  // Input (tie directly to the output of LUT4)
   .I1(I1),  // Input (tie directoy to the output of LUT4)
   .S(S)     // Input select to MUX
);

// End of MUXF5_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF6

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element is a two input multiplexer in two slices for creating a function-of-6 look-up table or an 8-to-1 multiplexer in combination with the associated four LUT4 look-up tables and two MUXF5 multiplexers. The local outputs (LO) from two MUXF5 multiplexers in the CLB are connected to the I0 and I1 inputs of the MUXF6. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The O output is a general interconnect.

The variants MUXF6_D and MUXF6_L provide additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

Logic Table

Inputs			Outputs
S	**I0**	**I1**	**O**
0	1	X	1
0	0	X	0
1	X	1	1
1	X	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF6: CLB MUX to tie two MUXF5's together with general output
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF6_inst : MUXF6
port map (
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie to MUXF5 LO out)
   I1 => I1,   -- Input (tie to MUXF5 LO out)
   S => S      -- Input select to MUX
);
```

```
-- End of MUXF6_inst instantiation
```

Verilog Instantiation Template

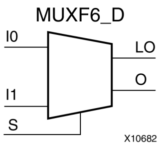
```
// MUXF6: CLB MUX to tie two MUXF5's together with general output
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF6 MUXF6_inst (
   .O(O),    // Output of MUX to general routing
   .I0(I0),  // Input (tie to MUXF5 LO out)
   .I1(I1),  // Input (tie to MUXF5 LO out)
   .S(S)     // Input select to MUX
);

// End of MUXF6_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF6_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element is a two input multiplexer in two slices for creating a function-of-6 look-up table or an 8-to-1 multiplexer in combination with the associated four LUT4 look-up tables and two MUXF5 multiplexers. The local outputs (LO) from two MUXF5 multiplexers in the CLB are connected to the I0 and I1 inputs of the MUXF6. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

Outputs O and LO are functionally identical. The O output is a general interconnect. The LO output connects to other inputs in the same CLB slice.

See also MUXF6 and MUXF6_L.

Logic Table

Inputs			Outputs	
S	**I0**	**I1**	**O**	**LO**
0	1	X	1	1
0	0	X	0	0
1	X	1	1	1
1	X	0	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF6_D: CLB MUX to tie two MUXF5's together with general and local outputs
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF6_D_inst : MUXF6_D
port map (
   LO => LO,   -- Output of MUX to local routing
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie to MUXF5 LO out)
   I1 => I1,   -- Input (tie to MUXF5 LO out)
   S => S      -- Input select to MUX
```

```
);

-- End of MUXF6_D_inst instantiation
```

Verilog Instantiation Template

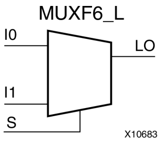
```
// MUXF6_D: CLB MUX to tie two MUXF5's together with general and local outputs
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF6_D MUXF6_D_inst (
   .LO(LO),  // Output of MUX to local routing
   .O(O),    // Output of MUX to general routing
   .I0(I0),  // Input (tie to MUXF5 LO out)
   .I1(I1),  // Input (tie to MUXF5 LO out)
   .S(S)     // Input select to MUX
);

// End of MUXF6_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF6_L

Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output



Introduction

This design element is a two input multiplexer in two slices for creating a function-of-6 look-up table or an 8-to-1 multiplexer in combination with the associated four LUT4 look-up tables and two MUXF5 multiplexers. The local outputs (LO) from two MUXF5 multiplexers in the CLB are connected to the I0 and I1 inputs of the MUXF6. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The LO output connects to other inputs in the same CLB slice.

See also MUXF6 and MUXF6_D.

Logic Table

Inputs			Output
S	**I0**	**I1**	**LO**
0	1	X	1
0	0	X	0
1	X	1	1
1	X	0	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF6_L: CLB MUX to tie two MUXF5's together with local output
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF6_L_inst : MUXF6_L
port map (
   LO => LO,   -- Output of MUX to local routing
   I0 => I0,   -- Input (tie to MUXF5 LO out)
   I1 => I1,   -- Input (tie to MUXF5 LO out)
   S => S      -- Input select to MUX
);
```

```
-- End of MUXF6_L_inst instantiation
```

Verilog Instantiation Template

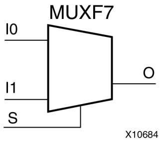
```
// MUXF6_L: CLB MUX to tie two MUXF5's together with local output
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF6_L MUXF6_L_inst (
   .LO(LO),  // Output of MUX to local routing
   .I0(I0),  // Input (tie to MUXF5 LO out)
   .I1(I1),  // Input (tie to MUXF5 LO out)
   .S(S)     // Input select to MUX
);

// End of MUXF6_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF7

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element is a two input multiplexer for creating a function-of-7 look-up table or a 16-to-1 multiplexer in combination with the associated MUXF6 and MUXF5 multiplexers, and LUT4 look-up tables. Local outputs (LO) of two MUXF6 are connected to the I0 and I1 inputs of the MUXF7. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The O output is a general interconnect.

The variants MUXF7_D and MUXF7_L provide additional types of outputs that can be used by different timing models for more accurate pre-layout timing estimation.

Logic Table

Inputs			Outputs
S	**I0**	**I1**	**O**
0	I0	X	I0
1	X	I1	I1
X	0	0	0
X	1	1	1

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Output of MUX to general routing
I0	Input	1	Input (tie to MUXF6 LO out)
I1	Input	1	Input (tie to MUXF6 LO out)
S	Input	1	Input select to MUX

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF7: CLB MUX to tie two MUXF6's together with general output
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF7_inst : MUXF7
port map (
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie to MUXF6 LO out or LUT6 O6 pin)
   I1 => I1,   -- Input (tie to MUXF6 LO out or LUT6 O6 pin)
   S => S      -- Input select to MUX
);

-- End of MUXF7_inst instantiation
```

Verilog Instantiation Template

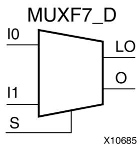
```
// MUXF7: CLB MUX to tie two MUXF6's together with general output
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF7 MUXF7_inst (
   .O(O),     // Output of MUX to general routing
   .I0(I0),   // Input (tie to MUXF6 LO out or LUT6 O6 pin)
   .I1(I1),   // Input (tie to MUXF6 LO out or LUT6 O6 pin)
   .S(S)      // Input select to MUX
);

// End of MUXF7_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF7_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element is a two input multiplexer for creating a function-of-7 look-up table or a 16-to-1 multiplexer in combination with the associated MUXF6 and MUXF5 multiplexers, and LUT4 look-up tables. Local outputs (LO) of two MUXF6 are connected to the I0 and I1 inputs of the MUXF7. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

Outputs O and LO are functionally identical. The O output is a general interconnect. The LO output connects to other inputs in the same CLB slice.

See also MUXF7 and MUXF7_L.

Logic Table

Inputs			Outputs	
S	**I0**	**I1**	**O**	**LO**
0	I0	X	I0	I0
1	X	I1	I1	I1
X	0	0	0	0
X	1	1	1	1

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Output of MUX to general routing
LO	Output	1	Output of MUX to local routing
I0	Input	1	Input (tie to MUXF6 LO out)
I1	Input	1	Input (tie to MUXF6 LO out)
S	Input	1	Input select to MUX

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF7_D: CLB MUX to tie two MUXF6's together with general and local outputs
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF7_D_inst : MUXF7_D
port map (
   LO => LO,   -- Output of MUX to local routing
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie to MUXF6 LO out or LUT6 O6 pin)
   I1 => I1,   -- Input (tie to MUXF6 LO out or LUT6 O6 pin)
   S => S      -- Input select to MUX
);

-- End of MUXF7_D_inst instantiation
```

Verilog Instantiation Template

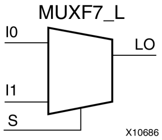
```
// MUXF7_D: CLB MUX to tie two MUXF6's together with general and local outputs
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF7_D MUXF7_D_inst (
   .LO(LO),   // Output of MUX to local routing
   .O(O),     // Output of MUX to general routing
   .I0(I0),   // Input (tie to MUXF6 LO out or LUT6 O6 pin)
   .I1(I1),   // Input (tie to MUXF6 LO out or LUT6 O6 pin)
   .S(S)      // Input select to MUX
);

// End of MUXF7_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF7_L

Primitive: 2-to-1 look-up table Multiplexer with Local Output



Introduction

This design element is a two input multiplexer for creating a function-of-7 look-up table or a 16-to-1 multiplexer in combination with the associated MUXF6 and MUXF5 multiplexers, and LUT4 look-up tables. Local outputs (LO) of two MUXF6 are connected to the I0 and I1 inputs of the MUXF7. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The LO output connects to other inputs in the same CLB slice.

See also MUXF7 and MUXF7_D.

Logic Table

Inputs			Output
S	**I0**	**I1**	**LO**
0	I0	X	I0
1	X	I1	I1
X	0	0	0
X	1	1	1

Port Descriptions

Port	Direction	Width	Function
LO	Output	1	Output of MUX to local routing
I0	Input	1	Input
I1	Input	1	Input
S	Input	1	Input select to MUX

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF7_L: CLB MUX to tie two MUXF6's together with local output
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF7_L_inst : MUXF7_L
port map (
   LO => LO,   -- Output of MUX to local routing
   I0 => I0,   -- Input (tie to MUXF6 LO out or LUT6 O6 pin)
   I1 => I1,   -- Input (tie to MUXF6 LO out or LUT6 O6 pin)
   S => S      -- Input select to MUX
);

-- End of MUXF7_L_inst instantiation
```

Verilog Instantiation Template

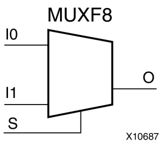
```
// MUXF7_L: CLB MUX to tie two MUXF6's together with local output
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF7_L MUXF7_L_inst (
   .LO(LO),   // Output of MUX to local routing
   .I0(I0),   // Input (tie to MUXF6 LO out or LUT6 O6 pin)
   .I1(I1),   // Input (tie to MUXF6 LO out or LUT6 O6 pin)
   .S(S)      // Input select to MUX
);

// End of MUXF7_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF8

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element provides a multiplexer function in eight slices for creating a function-of-8 look-up table or a 32-to-1 multiplexer in combination with the associated look-up tables, MUXF5s, MUXF6s, and MUXF7s. Local outputs (LO) of MUXF7 are connected to the I0 and I1 inputs of the MUXF8. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

Logic Table

Inputs			Outputs
S	**I0**	**I1**	**O**
0	I0	X	I0
1	X	I1	I1
X	0	0	0
X	1	1	1

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Output of MUX to general routing
I0	Input	1	Input (tie to MUXF7 LO out)
I1	Input	1	Input (tie to MUXF7 LO out)
S	Input	1	Input select to MUX

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF8: CLB MUX to tie two MUXF7's together with general output
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF8_inst : MUXF8
port map (
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie to MUXF7 LO out)
   I1 => I1,   -- Input (tie to MUXF7 LO out)
   S => S      -- Input select to MUX
);

-- End of MUXF8_inst instantiation
```

Verilog Instantiation Template

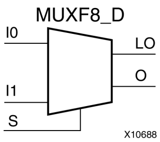
```
// MUXF8: CLB MUX to tie two MUXF7's together with general output
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF8 MUXF8_inst (
   .O(O),     // Output of MUX to general routing
   .I0(I0),   // Input (tie to MUXF7 LO out)
   .I1(I1),   // Input (tie to MUXF7 LO out)
   .S(S)      // Input select to MUX
);

// End of MUXF8_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF8_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element provides a multiplexer function in eight slices for creating a function-of-8 look-up table or a 32-to-1 multiplexer in combination with the associated look-up tables, MUXF5s, MUXF6s, and MUXF7s. Local outputs (LO) of MUXF7 are connected to the I0 and I1 inputs of the MUXF8. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

Outputs O and LO are functionally identical. The O output is a general interconnect. The LO output connects to other inputs in the same CLB slice.

Logic Table

Inputs			Outputs	
S	**I0**	**I1**	**O**	**LO**
0	I0	X	I0	I0
1	X	I1	I1	I1
X	0	0	0	0
X	1	1	1	1

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Output of MUX to general routing
LO	Output	1	Output of MUX to local routing
I0	Input	1	Input (tie to MUXF7 LO out)
I1	Input	1	Input (tie to MUXF7 LO out)
S	Input	1	Input select to MUX

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF8_D: CLB MUX to tie two MUXF7's together with general and local outputs
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF8_D_inst : MUXF8_D
port map (
   LO => LO,   -- Output of MUX to local routing
   O => O,     -- Output of MUX to general routing
   I0 => I0,   -- Input (tie to MUXF7 LO out)
   I1 => I1,   -- Input (tie to MUXF7 LO out)
   S => S      -- Input select to MUX
);

-- End of MUXF8_D_inst instantiation
```

Verilog Instantiation Template

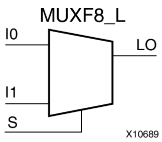
```
// MUXF8_D: CLB MUX to tie two MUXF7's together with general and local outputs
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF8_D MUXF8_D_inst (
   .LO(LO),   // Output of MUX to local routing
   .O(O),     // Output of MUX to general routing
   .I0(I0),   // Input (tie to MUXF7 LO out)
   .I1(I1),   // Input (tie to MUXF7 LO out)
   .S(S)      // Input select to MUX
);

// End of MUXF8_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

MUXF8_L

Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output



Introduction

This design element provides a multiplexer function in eight slices for creating a function-of-8 look-up table or a 32-to-1 multiplexer in combination with the associated look-up tables, MUXF5s, MUXF6s, and MUXF7s. Local outputs (LO) of MUXF7 are connected to the I0 and I1 inputs of the MUXF8. The S input is driven from any internal net. When Low, S selects I0. When High, S selects I1.

The LO output connects to other inputs in the same CLB slice.

Logic Table

Inputs			Output
S	**I0**	**I1**	**LO**
0	I0	X	I0
1	X	I1	I1
X	0	0	0
X	1	1	1

Port Descriptions

Port	Direction	Width	Function
LO	Output	1	Output of MUX to local routing
I0	Input	1	Input (tie to MUXF7 LO out)
I1	Input	1	Input (tie to MUXF7 LO out)
S	Input	1	Input select to MUX

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- MUXF8_L: CLB MUX to tie two MUXF7's together with local output
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

MUXF8_L_inst : MUXF8_L
port map (
   LO => LO,   -- Output of MUX to local routing
   I0 => I0,   -- Input (tie to MUXF7 LO out)
   I1 => I1,   -- Input (tie to MUXF7 LO out)
   S => S      -- Input select to MUX
);

-- End of MUXF8_L_inst instantiation
```

Verilog Instantiation Template

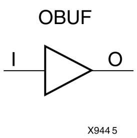
```
// MUXF8_L: CLB MUX to tie two MUXF7's together with local output
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

MUXF8_L MUXF8_L_inst (
   .LO(LO),   // Output of MUX to local routing
   .I0(I0),   // Input (tie to MUXF7 LO out)
   .I1(I1),   // Input (tie to MUXF7 LO out)
   .S(S)      // Input select to MUX
);

// End of MUXF8_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

OBUF

Primitive: Output Buffer



Introduction

This design element is a simple output buffer used to drive output signals to the FPGA device pins that do not need to be 3-stated (constantly driven). Either an OBUF, OBUFT, OBUFDS, or OBUFTDS must be connected to every output port in the design.

This element isolates the internal circuit and provides drive current for signals leaving a chip. It exists in input/output blocks (IOB). Its output (O) is connected to an OPAD or an IOPAD. The interface standard used by this element is LVTTL. Also, this element has selectable drive and slew rates using the DRIVE and SLOW or FAST constraints. The defaults are DRIVE=12 mA and SLOW slew.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Output of OBUF to be connected directly to top-level output port.
I	Input	1	Input of OBUF. Connect to the logic driving the output port.

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
DRIVE	Integer	2, 4, 6, 8, 12, 16, 24	12	Specifies the output current drive strength of the I/O. It is suggested that you set this to the lowest setting tolerable for the design drive and timing requirements.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.
SLEW	String	"SLOW" or "FAST"	"SLOW"	Specifies the slew rate of the output driver. Consult the product Data Sheet for recommendations of the best setting for this attribute.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- OBUF: Single-ended Output Buffer
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

OBUF_inst : OBUF
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   DRIVE => 12,
   IOSTANDARD => "DEFAULT",
   SLEW => "SLOW")
port map (
   O => O,     -- Buffer output (connect directly to top-level port)
   I => I      -- Buffer input
);

-- End of OBUF_inst instantiation
```

Verilog Instantiation Template

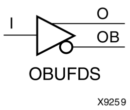
```
// OBUF: Single-ended Output Buffer
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

OBUF #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .DRIVE(12),   // Specify the output drive strength
   .IOSTANDARD("DEFAULT"), // Specify the output I/O standard
   .SLEW("SLOW") // Specify the output slew rate
) OBUF_inst (
   .O(O),     // Buffer output (connect directly to top-level port)
   .I(I)      // Buffer input
);

// End of OBUF_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

OBUFDS

Primitive: Differential Signaling Output Buffer



Introduction

This design element is a single output buffer that supports low-voltage, differential signaling (1.8 v CMOS). OBUFDS isolates the internal circuit and provides drive current for signals leaving the chip. Its output is represented as two distinct ports (O and OB), one deemed the "master" and the other the "slave." The master and the slave are opposite phases of the same logical signal (for example, MYNET and MYNETB).

Logic Table

Inputs	Outputs	
I	**O**	**OB**
0	0	1
1	1	0

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Diff_p output (connect directly to top level port)
OB	Output	1	Diff_n output (connect directly to top level port)
I	Input	1	Buffer input

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_ CARE"	"DONT_CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- OBUFDS: Differential Output Buffer
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

OBUFDS_inst : OBUFDS
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   IOSTANDARD => "DEFAULT")
port map (
   O => O,      -- Diff_p output (connect directly to top-level port)
   OB => OB,    -- Diff_n output (connect directly to top-level port)
   I => I       -- Buffer input
);

-- End of OBUFDS_inst instantiation
```

Verilog Instantiation Template

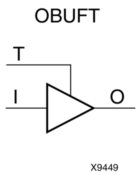
```
// OBUFDS: Differential Output Buffer
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

OBUFDS #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .IOSTANDARD("DEFAULT") // Specify the output I/O standard
) OBUFDS_inst (
   .O(O),      // Diff_p output (connect directly to top-level port)
   .OB(OB),    // Diff_n output (connect directly to top-level port)
   .I(I)       // Buffer input
);

// End of OBUFDS_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

OBUFT

Primitive: 3-State Output Buffer with Active Low Output Enable



Introduction

This design element is a single, 3-state output buffer with input I, output O, and active-Low output enables (T). This element uses the LVTTL standard and has selectable drive and slew rates using the DRIVE and SLOW or FAST constraints. The defaults are DRIVE=12 mA and SLOW slew.

When T is Low, data on the inputs of the buffers is transferred to the corresponding outputs. When T is High, the output is high impedance (off or Z state). OBUFTs are generally used when a single-ended output is needed with a 3-state capability, such as the case when building bidirectional I/O.

Logic Table

Inputs		Outputs
T	**I**	**O**
1	X	Z
0	1	1
0	0	0

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Buffer output (connect directly to top-level port)
I	Input	1	Buffer input
T	Input	1	3-state enable input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_ CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
DRIVE	Integer	2, 4, 6, 8, 12, 16, 24	12	Specifies the output current drive strength of the I/O. You should set this to the lowest setting tolerable for the design drive and timing requirements.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.
SLEW	String	"SLOW" or "FAST"	"SLOW"	Specifies the slew rate of the output driver. See the Data Sheet for recommendations of the best setting for this attribute.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- OBUFT: Single-ended 3-state Output Buffer
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

OBUFT_inst : OBUFT
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   DRIVE => 12,
   IOSTANDARD => "DEFAULT",
   SLEW => "SLOW")
port map (
   O => O,     -- Buffer output (connect directly to top-level port)
   I => I,     -- Buffer input
   T => T      -- 3-state enable input
);

-- End of OBUFT_inst instantiation
```

Verilog Instantiation Template

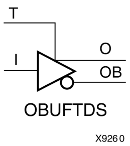
```
// OBUFT: Single-ended 3-state Output Buffer
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

OBUFT #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .DRIVE(12),   // Specify the output drive strength
   .IOSTANDARD("DEFAULT"), // Specify the output I/O standard
   .SLEW("SLOW") // Specify the output slew rate
) OBUFT_inst (
   .O(O),     // Buffer output (connect directly to top-level port)
   .I(I),     // Buffer input
   .T(T)      // 3-state enable input
);

  // End of OBUFT_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

OBUFTDS

Primitive: 3-State Output Buffer with Differential Signaling, Active-Low Output Enable



Introduction

This design element is an output buffer that supports low-voltage, differential signaling. For the OBUFTDS, a design level interface signal is represented as two distinct ports (O and OB), one deemed the "master" and the other the "slave." The master and the slave are opposite phases of the same logical signal (for example, MYNET_P and MYNET_N).

Logic Table

Inputs		**Outputs**	
I	**T**	**O**	**OB**
X	1	Z	Z
0	0	0	1
1	0	1	0

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Diff_p output (connect directly to top level port)
OB	Output	1	Diff_n output (connect directly to top level port)
I	Input	1	Buffer input
T	Input	1	3-state enable input

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
CAPACITANCE	String	"LOW", "NORMAL", "DONT_CARE"	"DONT_ CARE"	Specified whether the I/O should be used with lower or normal intrinsic capacitance.
IOSTANDARD	String	See Data Sheet	"DEFAULT"	Assigns an I/O standard to the element.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- OBUFTDS: Differential 3-state Output Buffer
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

OBUFTDS_inst : OBUFTDS
generic map (
   CAPACITANCE => "DONT_CARE", -- "LOW", "NORMAL", "DONT_CARE"
   IOSTANDARD => "DEFAULT")
port map (
   O => O,      -- Diff_p output (connect directly to top-level port)
   OB => OB,    -- Diff_n output (connect directly to top-level port)
   I => I,      -- Buffer input
   T => T       -- 3-state enable input
);

-- End of OBUFTDS_inst instantiation
```

Verilog Instantiation Template

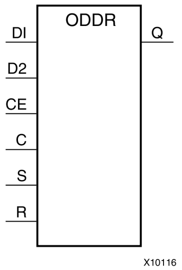
```
// OBUFTDS: Differential 3-state Output Buffer
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

OBUFTDS #(
   .CAPACITANCE("DONT_CARE"), // "DONT_CARE". "LOW" or "NORMAL" (Virtex-4 only)
   .IOSTANDARD("DEFAULT") // Specify the output I/O standard
) OBUFTDS_inst (
   .O(O),      // Diff_p output (connect directly to top-level port)
   .OB(OB),    // Diff_n output (connect directly to top-level port)
   .I(I),      // Buffer input
   .T(T)       // 3-state enable input
);

// End of OBUFTDS_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ODDR

Primitive: Dedicated Dual Data Rate (DDR) Output Register



Introduction

This design element is a dedicated output register for use in transmitting dual data rate (DDR) signals from FPGA devices. The ODDR interface with the FPGA fabric is not limited to opposite clock edges. The ODDR is available with modes that allow data to be presented from the FPGA fabric at the same clock edge. This feature allows designers to avoid additional timing complexities and CLB usage. In addition, the ODDR works in conjunction with SelectIO™ features.

ODDR Modes

This element has two modes of operation. These modes are set by the DDR_CLK_EDGE attribute.

- **OPPOSITE_EDGE mode -** The data transmit interface uses the classic DDR methodology. Given a data and clock at pin D1-2 and C respectively, D1 is sampled at every positive edge of clock C, and D2 is sampled at every negative edge of clock C. Q changes every clock edge.
- **SAME_EDGE mode -** Data is still transmitted at the output of the ODDR by opposite edges of clock C. However, the two inputs to the ODDR are clocked with a positive clock edge of clock signal C and an extra register is clocked with a negative clock edge of clock signal C. Using this feature, DDR data can now be presented into the ODDR at the same clock edge.

Port Descriptions

Port	Direction	Width	Function
Q	Output	1	Data Output (DDR) - The ODDR output that connects to the IOB pad.
C	Input	1	Clock Input - The C pin represents the clock input pin.
CE	Input	1	Clock Enable Input - When asserted High, this port enables the clock input on port C.
D1 : D2	Input	1 (each)	Data Input - This pin is where the DDR data is presented into the ODDR module.
R	Input	1	Reset - Depends on how SRTYPE is set.
S	Input	1	Set - Active High asynchronous set pin. This pin can also be Synchronous depending on the SRTYPE attribute.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
DDR_CLK_EDGE	String	"OPPOSITE_EDGE", "SAME_EDGE"	"OPPOSITE_EDGE"	DDR clock mode recovery mode selection.
INIT	Binary	0, 1	1	Q initialization value.
SRTYPE	String	"SYNC", "ASYNC"	"SYNC"	Set/Reset type selection.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ODDR: Output Double Data Rate Output Register with Set, Reset
--       and Clock Enable.
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ODDR_inst : ODDR
generic map(
   DDR_CLK_EDGE => "OPPOSITE_EDGE", -- "OPPOSITE_EDGE" or "SAME_EDGE"
   INIT => '0',    -- Initial value for Q port ('1' or '0')
   SRTYPE => "SYNC") -- Reset Type ("ASYNC" or "SYNC")
port map (
   Q => Q,    -- 1-bit DDR output
   C => C,     -- 1-bit clock input
   CE => CE,  -- 1-bit clock enable input
   D1 => D1,  -- 1-bit data input (positive edge)
   D2 => D2,  -- 1-bit data input (negative edge)
   R => R,    -- 1-bit reset input
   S => S     -- 1-bit set input
);

-- End of ODDR_inst instantiation
```

Verilog Instantiation Template

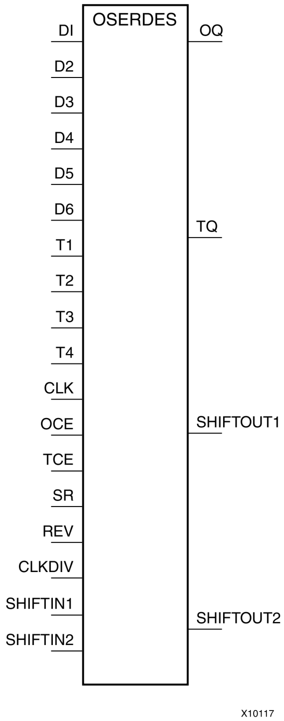
```
// ODDR: Output Double Data Rate Output Register with Set, Reset
//       and Clock Enable.
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ODDR #(
   .DDR_CLK_EDGE("OPPOSITE_EDGE"), // "OPPOSITE_EDGE" or "SAME_EDGE"
   .INIT(1'b0),    // Initial value of Q: 1'b0 or 1'b1
   .SRTYPE("SYNC") // Set/Reset type: "SYNC" or "ASYNC"
) ODDR_inst (
   .Q(Q),   // 1-bit DDR output
   .C(C),   // 1-bit clock input
   .CE(CE), // 1-bit clock enable input
   .D1(D1), // 1-bit data input (positive edge)
   .D2(D2), // 1-bit data input (negative edge)
   .R(R),   // 1-bit reset
   .S(S)    // 1-bit set
);

// End of ODDR_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

OSERDES

Primitive: Dedicated IOB Output Serializer



Introduction

Use the OSERDES primitive to easily implement a source synchronous interface. This device helps you by saving logic resources that would otherwise be implemented in the FPGA fabric. It also avoids additional timing complexities that you might encounter when you are designing circuitry in the FPGA fabric. This element contains multiple clock inputs to accommodate various applications, and will work in conjunction with SelectIO™ features.

Port Descriptions

Port	Direction	Width	Function
OQ	Output	1	Data Path Output - This port is the data output of the OSERDES module. This port connects the output of the data parallel-to-serial converter to the data input of the IOB pad. In addition, this output port can also be configured to bypass all the submodules within the OSERDES module.
SHIFTOUT1-2	Output	1 (each)	Carry Out for data input expansion. Connect to SHIFTIN1/2 of master.
TQ	Output	1	3-State Path Output - This port is the 3-state output of the OSERDES module. This port connects the output of the 3-state parallel-to-serial converter to the control input of the IOB pad.
CLK	Input	1	High Speed Clock Input - This clock input is used to drive the parallel-to-serial converters. The possible source for the CLK port is from one of the following clock resources: • Ten global clock lines in a clock region • Four regional clock lines

Port	Direction	Width	Function
			• Four clock capable I/Os (within adjacent clock region) • Fabric (through bypass)
CLKDIV	Input	1	Divided High Speed Clock Input - This clock input is used to drive the parallel-to-serial converter. This clock must be a divided down version of the clock connected to the CLK port. One of the following clock resources can be used as a source for CLKDIV: • Ten global clock lines in a clock region • Four regional clock lines
D1-D6	Input	1	Parallel Data Inputs - Ports D1 to D6 are the location in which all incoming parallel data enters the OSERDES module. This port is connected to the FPGA fabric, and can be configured from 2 to 6 bits. In the extended width mode, this port can be expanded up to 10 bits.
OCE	Input	1	Parallel to serial converter (data) clock enable - This port is used to enables the output of the data parallel-to-serial converter when asserted High.
SR	Input	1	Set/Reset Input - The set/reset (SR) pin forces the storage element into the state specified by the SRVAL attribute. SRVAL = 1 forces a logic 1. SRVAL = 0 forces a logic 0. The reset condition predominates over the set condition.
SHIFTIN1-2	Input	1 (each)	Carry Input for Data Input Expansion. Connect to SHIFTOUT1/2 of slave.
T1 - T4	Input	1 (each)	Parallel 3-State Inputs - Ports T1 to T4 are the location in which all parallel 3-state signals enters the OSERDES module. This port is connected to the FPGA fabric, and can be configured from 1 to 4 bits. This feature is not supported in the extended width mode.
TCE	Input	1	Parallel to serial converter (3-state) clock enable - This port is used to enable the output of the 3-state signal parallel-to-serial converter when asserted High.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

The data parallel-to-serial converter in the OSERDES module takes in 2 to 6 bits of parallel data and converts them into serial data. Data input widths larger than 6 (7, 8, and 10) are achievable by cascading two OSERDES modules for data width expansion. In order to do this, one OSERDES must be set into a MASTER mode, while another is set into SLAVE mode. You must connect the SHIFTOUT of "slave" and SHIFTIN of "master" ports together. The "slave" only uses D3 to D6 ports as its input. The parallel-to-serial converter is available for both SDR and DDR modes.

This module is designed such that the data input at D1 port is the first output bit. This module is controlled by CLK and CLKDIV clocks. The following table describes the relationship between CLK and CLKDIV for both SDR and DDR mode.

SDR Data Width	DDR Data Width	CLK	CLKDIV
2	4	2X	X
3	6	3X	X
4	8	4X	X
5	10	5X	X
6	-	6X	X
7	-	7X	X
8	-	8X	X

Output of this block is connected to the data input of an IOB pad of the FPGA. This IOB pad can be configured to a desired standard using SelectIO.

Parallel-to-Serial Converter (3-state)

The 3-state parallel-to-serial converter in the OSERDES module takes in up to 4 bits of parallel 3-state signals and converts them into serial 3-state signal. Unlike the data parallel-to-serial converter, the 3-state parallel-to-serial converter is not extendable to more than 4-bit, 3-state signals. This module is primarily controlled by CLK and CLKDIV clocks. In order to use this module, the following attributes must be declared: DATA_RATE_TQ and TRISTATE_WIDTH. In certain cases, you can also need to declare DATA_RATE_OQ and DATA_WIDTH. The following table lists the attributes needed for the desired functionality.

Mode of Operation	DATA_RATE_TQ	TRISTATE_WIDTH
4-bit DDR*	DDR	4
1-bit SDR	SDR	1
Buffer	BUF	1

Output of this block is connected to the 3-state input of an IOB pad of the FPGA. This IOB pad can be configured to a desired standard using SelectIO.

Width Expansion

It is possible to use this element to transmit parallel data widths larger than six. However, the 3-state output is not expandable. In order to use this feature, *two* of these elements need to be instantiated, and the two must be an adjacent master and slave pair. The attribute MODE must be set to either "MASTER" or "SLAVE" in order to differentiate the modes of the OSERDES pair. In addition, you must connect the SHIFTIN ports of the MASTER to the SHIFTOUT ports of the SLAVE. This feature supports data widths of 7, 8, and 10 for SDR and DDR mode. The table below lists the data width availability for SDR and DDR mode.

Mode	Widths
SDR Data Widths	2,3,4,5,6,7,8
DDR Data Widths	4,6,8,10

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
DATA_RATE_OQ	String	"SDR", "DDR"	"DDR"	Defines whether the data changes at every clock edge or every positive clock edge with respect to CLK.
DATA_RATE_TQ	String	"BUF", "SDR", "DDR"	"DDR"	Defines whether the 3-state changes at every clock edge, every positive clock edge, or buffer configuration with respect to CLK.

Attribute	Data Type	Allowed Values	Default	Description
DATA_WIDTH	Integer	2, 3, 4, 5, 6, 7, 8, or 10	4	If DATA_RATE_OQ = DDR, value is limited to 4, 6, 8, or 10. If DATA_RATE_OQ = SDR, value is limited to 2, 3, 4, 5, 6, 7, or 8.
INIT_OQ	Binary	0, 1	0	Defines the initial value of OQ output
INIT_TQ	Binary	0, 1	0	Defines the initial value of TQ output
SERDES_MODE	String	"MASTER", "SLAVE"	"MASTER"	Defines whether the OSERDES module is a master or slave when width expansion is used.
SRVAL_OQ	Binary	0, 1	0	Defines the value of OQ output when reset is invoked.
SRVAL_TQ	Binary	0, 1	0	Defines the value of TQ output when reset is invoked.
TRISTATE_WIDTH	Integer	1, 2, 4	4	If DATA_RATE_TQ = DDR, value is limited to 2 or 4. The value can only be set to 1 when DATA_RATE_TQ = SDR or BUF.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- OSERDES: Output SERDES
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

OSERDES_inst : OSERDES
generic map (
   DATA_RATE_OQ => "DDR", -- Specify data rate to "DDR" or "SDR"
   DATA_RATE_TQ => "DDR", -- Specify data rate to "DDR", "SDR", or "BUF"
   DATA_WIDTH => 4, -- Specify data width - For DDR: 4,6,8, or 10
                    -- For SDR or BUF: 2,3,4,5,6,7, or 8
   INIT_OQ => '0',  -- INIT for Q1 register - '1' or '0'
   INIT_TQ => '0',  -- INIT for Q2 register - '1' or '0'
   SERDES_MODE => "MASTER", --Set SERDES mode to "MASTER" or "SLAVE"
   SRVAL_OQ => '0', -- Define Q1 output value upon SR assertion - '1' or '0'
   SRVAL_TQ => '0', -- Define Q1 output value upon SR assertion - '1' or '0'
   TRISTATE_WIDTH => 4) -- Specify parallel to serial converter width
                        -- When DATA_RATE_TQ = DDR: 2 or 4
                        -- When DATA_RATE_TQ = SDR or BUF: 1 "
port map (
   OQ => OQ,     -- 1-bit output
   SHIFTOUT1 => SHIFTOUT1, -- 1-bit data expansion output
   SHIFTOUT2 => SHIFTOUT2, -- 1-bit data expansion output
   TQ => TQ,     -- 1-bit 3-state control output
   CLK => CLK,  -- 1-bit clock input
   CLKDIV => CLKDIV,  -- 1-bit divided clock input
   D1 => D1,     -- 1-bit parallel data input
   D2 => D2,     -- 1-bit parallel data input
   D3 => D3,     -- 1-bit parallel data input
   D4 => D4,     -- 1-bit parallel data input
   D5 => D5,     -- 1-bit parallel data input
   D6 => D6,     -- 1-bit parallel data input
   OCE => OCE,  -- 1-bit clcok enable input
   REV => '0',  -- Must be tied to logic zero
   SHIFTIN1 => SHIFTIN1, -- 1-bit data expansion input
   SHIFTIN2 => SHIFTIN2, -- 1-bit data expansion input
   SR => SR,   -- 1-bit set/reset input
   T1 => T1,   -- 1-bit parallel 3-state input
```

```
   T2 => T2,    -- 1-bit parallel 3-state input
   T3 => T3,    -- 1-bit parallel 3-state input
   T4 => T4,    -- 1-bit parallel 3-state input
   TCE => TCE   -- 1-bit 3-state signal clock enable input
);

-- End of OSERDES_inst instantiation
```

Verilog Instantiation Template

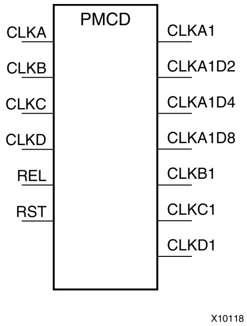
```
// OSERDES: Source Synchronous Output Serializer
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

OSERDES #(
   .DATA_RATE_OQ("DDR"), // Specify data rate to "DDR" or "SDR"
   .DATA_RATE_TQ("DDR"), // Specify data rate to "DDR", "SDR", or "BUF"
   .DATA_WIDTH(4), // Specify data width - for DDR: 4,6,8, or 10
                   //     for SDR or BUF: 2,3,4,5,6,7, or 8
   .INIT_OQ(1'b0), // INIT for OQ register - 1'b1 or 1'b0
   .INIT_TQ(1'b0), // INIT for OQ register - 1'b1 or 1'b0
   .SERDES_MODE("MASTER"), // Set SERDES mode to "MASTER" or "SLAVE"
   .SRVAL_OQ(1'b0), // Define OQ output value upon SR assertion - 1'b1 or 1'b0
   .SRVAL_TQ(1'b0), // Define TQ output value upon SR assertion - 1'b1 or 1'b0
   .TRISTATE_WIDTH(4)  // Specify parallel to serial converter width
                       //    When DATA_RATE_TQ = DDR: 2 or 4
                       //    When DATA_RATE_TQ = SDR or BUF: 1
) OSERDES_inst (
   .OQ(OQ),     // 1-bit data path output
   .SHIFTOUT1(SHIFTOUT1), // 1-bit data expansion output
   .SHIFTOUT2(SHIFTOUT2), // 1-bit data expansion output
   .TQ(TQ),     // 1-bit 3-state control output
   .CLK(CLK),        // 1-bit clock input
   .CLKDIV(CLKDIV), // 1-bit divided clock input
   .D1(D1),     // 1-bit parallel data input
   .D2(D2),     // 1-bit parallel data input
   .D3(D3),     // 1-bit parallel data input
   .D4(D4),     // 1-bit parallel data input
   .D5(D5),     // 1-bit parallel data input
   .D6(D6),     // 1-bit parallel data input
   .OCE(OCE),   // 1-bit clock enable input
   .REV(1'b0), // Must be tied to logic zero
   .SHIFTIN1(SHIFTIN1), // 1-bit data expansion input
   .SHIFTIN2(SHIFTIN2), // 1-bit data expansion input
   .SR(SR),     // 1-bit set/reset input
   .T1(T1),     // 1-bit parallel 3-state input
   .T2(T2),     // 1-bit parallel 3-state input
   .T3(T3),     // 1-bit parallel 3-state input
   .T4(T4),     // 1-bit parallel 3-state input
   .TCE(TCE)    // 1-bit 3-state signal clock enable input
);

// End of OSERDES_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

PMCD

Primitive: Phase-Matched Clock Divider



Introduction

This design element is one of the clock resources available in the Virtex®-4 architecture. It provides the following clock management features:

Phase-Aligned Divided Clocks

The phase-aligned divided clocks create up to four frequency-divided and phase-aligned versions of an input clock, CLKA. The output clocks are a function of the input clock frequency: divided-by-1 (CLKA1), divided-by-2 (CLKAD2), divided-by-4 (CLKA1D4), and divided-by-8 (CLKA1D8). CLKA1, CLKA1D2, CLKA1D4, CLKA1D8 output clocks are rising-edge aligned.

Matched-Clock Phase

The matched-clock phase preserves edge alignments, phase relations, or skews between the input clock CLKA and other PMCD input clocks. Three additional input clocks (CLKB, CLKC, and CLKD) and three corresponding delayed output clocks (CLKB1, CLKC1, and CLKD1) are available. The same delay is inserted to CLKA, CLKB, CLKC, and CLKD; thus, the delayed CLKA1, CLKB1, CLKC1, and CLKD1 clock outputs maintain edge alignments, phase relations, and the skews of the respective inputs.

This design element can be used with other clock resources, including global buffers and the digital clock management feature. Together, these clock resources provide flexibility in managing complex clock networks in designs

Port Descriptions

Port	Direction	Function
CLKA	Input	CLKA is a clock input to the PMCD. The CLKA frequency can be divided by 1, 2, 4, and 8.
CLKB CLKC CLKD	Input	CLKB, CLKC, and CLKD are clock inputs to the PMCD. These clock are not divided by PMCD, however, they are delayed by the PMCD to maintain the phase alignment and phase relationship to CLKA.
RST	Input	RST is the reset input to the PMCD. Asserting the RST signal asynchronously forces all outputs Low. Deasserting RST synchronously allows all outputs to toggle.
REL	Input	REL is the release input to the PMCD. Asserting the REL signal releases the divided output synchronous to CLKA.
CLKA1	Output	The CLKA1 output has the same frequency as the CLKA input. It is a delayed version of CLKA.
CLKA1D2	Output	The CLKA1D2 output has the frequency of CLKA divided by two. CLKA1D2 is rising-edge aligned to CLKA1.

Port	Direction	Function
CLKA1D4	Output	The CLKA1D4 output has the frequency of CLKA divided by four. CLKA1D4 is rising-edge aligned to CLKA1.
CLKA1D8	Output	The CLKA1D8 output has the frequency of CLKA divided by eight, CLKA1D8 is rising-edge aligned to CLKA1.
CLKB1 CLKC1 CLKD1	Output	The CLKB1 output is has the same frequency as the CLKB input, a delayed version of CLKB. The skew between CLKB1 and CLKA1 is the same as the skew between CLKB and CLKA inputs. Similarly, CLKC1 is a delayed version of CLKC, and CLKD1 is a delayed version of CLKD.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
EN_REL	Boolean	FALSE, TRUE	FALSE	This attribute allows for CLKA1D2, CLKA1D4, and CLKA1D8 outputs to be released at REL signal assertion. **Note** REL is synchronous to CLKA input.
RST_DEASSERT_ CLK	String	"CLKA," "CLKB", "CLKC", "CLKD"	"CLKA"	This attribute allows the deassertion of the RST signal to be synchronous to a selected PMCD input clock.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- PMCD: Phase-Matched Clock Divider Circuit
--       Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

PMCD_inst : PMCD
generic map (
   EN_REL => FALSE,              -- TRUE/FALSE to allow synchronous deassertion of RST
   RST_DEASSERT_CLK => "CLKA")  -- Reset syncronization to which clock: CLKA, CLKB, CLKC or CLKD
port map (
   CLKA1 => CLKA1, -- Output CLKA divided by 1
   CLKA1D2 => CLKA1D2, -- Output CLKA divided by 2
   CLKA1D4 => CLKA1D4, -- Output CLKA divided by 4
   CLKA1D8 => CLKA1D8, -- Output CLKA divided by 8
   CLKB1 => CLKB1, -- Output phase matched CLKB
   CLKC1 => CLKC1, -- Output phase matched CLKC
   CLKD1 => CLKD1, -- Output phase matched CLKD
   CLKA => CLKA,   -- Input CLKA
   CLKB => CLKB,   -- Input CLKB
   CLKC => CLKC,   -- Input CLKC
   CLKD => CLKD,   -- Input CLKD
   REL => REL,     -- PCMD release input
   RST => RST      -- Active high reset input
```

```
);

-- End of PMCD_inst instantiation
```

Verilog Instantiation Template

```
// PMCD: Phase-Matched Clock Divider Circuit
//       Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

PMCD #(
   .EN_REL("FALSE"),  // TRUE/FALSE to allow synchronous deassertion of RST
   .RST_DEASSERT_CLK("CLKA") // Reset syncronization to which clock: CLKA, CLKB, CLKC or CLKD
) PMCD_inst (
   .CLKA1(CLKA1),     // Output CLKA divided by 1
   .CLKA1D2(CLKA1D2), // Output CLKA divided by 2
   .CLKA1D4(CLKA1D4), // Output CLKA divided by 4
   .CLKA1D8(CLKA1D8), // Output CLKA divided by 8
   .CLKB1(CLKB1),     // Output phase matched CLKB
   .CLKC1(CLKC1),     // Output phase matched CLKC
   .CLKD1(CLKD1),     // Output phase matched CLKD
   .CLKA(CLKA),       // Input CLKA
   .CLKB(CLKB),       // Input CLKB
   .CLKC(CLKC),       // Input CLKC
   .CLKD(CLKD),       // Input CLKD
   .REL(REL),         // PCMD release input
   .RST(RST)          // Active high reset input
);

// End of PMCD_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

PPC405_ADV

Primitive: Primitive for the Power PC Core

Introduction

This design element is a 32-bit implementation of the PowerPC® embedded environment architecture that is derived from the PowerPC architecture. Specifically, the PowerPC 405 is an embedded PowerPC 405F6, for Virtex®-4 devices, processor core. The processor core also contains on-chip memory logic (OCM), an APU controller (Virtex-4 devices only), and the gasket logic and interface.

The PowerPC architecture provides a software model that ensures compatibility between implementations of the PowerPC family of microprocessors. The PowerPC architecture defines parameters that guarantee compatible processor implementations at the application-program level, allowing broad flexibility in the development derivative PowerPC implementations that meet specific market requirements.

Design Entry Method

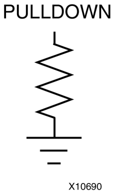
Instantiation	Yes
Inference	No
CORE Generator™ and wizards	Recommended
Macro support	No

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

PULLDOWN

Primitive: Resistor to GND for Input Pads, Open-Drain, and 3-State Outputs



Introduction

This resistor element is connected to input, output, or bidirectional pads to guarantee a logic Low level for nodes that might float.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Pulldown output (connect directly to top level port)

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- PULLDOWN: I/O Buffer Weak Pull-down
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

PULLDOWN_inst : PULLDOWN
port map (
   O => O     -- Pulldown output (connect directly to top-level port)
);

-- End of PULLDOWN_inst instantiation
```

Verilog Instantiation Template

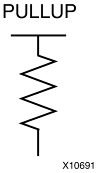
```
// PULLDOWN: I/O Buffer Weak Pull-down
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

PULLDOWN PULLDOWN_inst (
   .O(O)     // Pulldown output (connect directly to top-level port)
);

// End of PULLDOWN_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

PULLUP

Primitive: Resistor to VCC for Input PADs, Open-Drain, and 3-State Outputs



Introduction

This design element allows for an input, 3-state output or bi-directional port to be driven to a weak high value when not being driven by an internal or external source. This element establishes a High logic level for open-drain elements and macros when all the drivers are off.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Pullup output (connect directly to top level port)

Design Entry Method

Instantiation	Yes
Inference	No
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- PULLUP: I/O Buffer Weak Pull-up
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

PULLUP_inst : PULLUP
port map (
   O => O     -- Pullup output (connect directly to top-level port)
);

-- End of PULLUP_inst instantiation
```

Verilog Instantiation Template

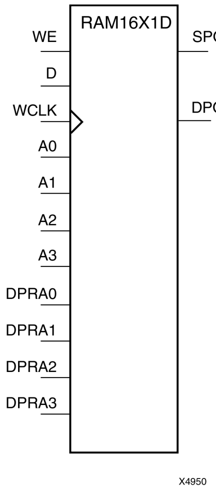
```
// PULLUP: I/O Buffer Weak Pull-up
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

PULLUP PULLUP_inst (
   .O(O)     // Pullup output (connect directly to top-level port)
);

// End of PULLUP_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM16X1D

Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM



Introduction

This element is a 16-word by 1-bit static dual port random access memory with synchronous write capability. The device has two address ports: the read address (DPRA3:DPRA0) and the write address (A3:A0). These two address ports are asynchronous. The read address controls the location of the data driven out of the output pin (DPO), and the write address controls the destination of a valid write transaction. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected.

When WE is High, any positive transition on (WCLK) loads the data on the data input (D) into the word selected by the 4-bit write address. For predictable performance, write address and data inputs must be stable before a Low-to-High (WCLK) transition. This RAM block assumes an active-High (WCLK). (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The SPO output reflects the data in the memory cell addressed by A3:A0. The DPO output reflects the data in the memory cell addressed by DPRA3:DPRA0.

Note The write process is not affected by the address on the read address port.

You can use the INIT attribute to directly specify an initial value. The value must be a hexadecimal number, for example, INIT=ABAC. If the INIT attribute is not specified, the RAM is initialized with all zeros.

Logic Table

Mode selection is shown in the following logic table:

Inputs			**Outputs**	
WE (mode)	**WCLK**	**D**	**SPO**	**DPO**
0 (read)	X	X	data_a	data_d
1 (read)	0	X	data_a	data_d
1 (read)	1	X	data_a	data_d
1 (write)	↑	D	D	data_d
1 (read)	↓	X	data_a	data_d
data_a = word addressed by bits A3-A0				
data_d = word addressed by bits DPRA3-DPRA0				

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros.	Initializes RAMs, registers, and look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM16X1D: 16 x 1 positive edge write, asynchronous read dual-port distributed RAM
--             Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM16X1D_inst : RAM16X1D
generic map (
   INIT => X"0000")
port map (
   DPO => DPO,      -- Read-only 1-bit data output for DPRA
   SPO => SPO,      -- R/W 1-bit data output for A0-A3
   A0 => A0,        -- R/W address[0] input bit
   A1 => A1,        -- R/W address[1] input bit
   A2 => A2,        -- R/W address[2] input bit
   A3 => A3,        -- R/W ddress[3] input bit
   D => D,          -- Write 1-bit data input
   DPRA0 => DPRA0, -- Read-only address[0] input bit
   DPRA1 => DPRA1, -- Read-only address[1] input bit
   DPRA2 => DPRA2, -- Read-only address[2] input bit
   DPRA3 => DPRA3, -- Read-only address[3] input bit
   WCLK => WCLK,    -- Write clock input
   WE => WE         -- Write enable input
);

-- End of RAM16X1D_inst instantiation
```

Verilog Instantiation Template

```
// RAM16X1D: 16 x 1 positive edge write, asynchronous read dual-port distributed RAM
//            Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM16X1D #(
   .INIT(16'h0000) // Initial contents of RAM
) RAM16X1D_inst (
   .DPO(DPO),     // Read-only 1-bit data output for DPRA
   .SPO(SPO),     // Rw/ 1-bit data output for A0-A3
   .A0(A0),       // Rw/ address[0] input bit
   .A1(A1),       // Rw/ address[1] input bit
   .A2(A2),       // Rw/ address[2] input bit
   .A3(A3),       // Rw/ address[3] input bit
   .D(D),         // Write 1-bit data input
   .DPRA0(DPRA0), // Read address[0] input bit
```

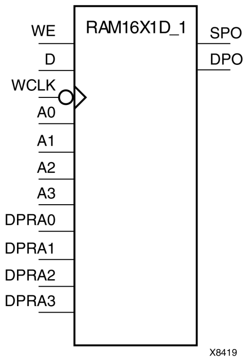
```
   .DPRA1(DPRA1), // Read address[1] input bit
   .DPRA2(DPRA2), // Read address[2] input bit
   .DPRA3(DPRA3), // Read address[3] input bit
   .WCLK(WCLK),   // Write clock input
   .WE(WE)        // Write enable input
);

// End of RAM16X1D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM16X1D_1

Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM with Negative-Edge Clock



Introduction

This is a 16-word by 1-bit static dual port random access memory with synchronous write capability and negative-edge clock. The device has two separate address ports: the read address (DPRA3:DPRA0) and the write address (A3:A0). These two address ports are asynchronous. The read address controls the location of the data driven out of the output pin (DPO), and the write address controls the destination of a valid write transaction.

When the write enable (WE) is set to Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 4-bit write address. For predictable performance, write address and data inputs must be stable before a High-to-Low WCLK transition. This RAM block assumes an active-Low (WCLK). (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

You can initialize RAM16X1D_1 during configuration using the INIT attribute.

The SPO output reflects the data in the memory cell addressed by A3:A0. The DPO output reflects the data in the memory cell addressed by DPRA3:DPRA0.

Note The write process is not affected by the address on the read address port.

Logic Table

Mode selection is shown in the following logic table:

Inputs			**Outputs**	
WE (mode)	**WCLK**	**D**	**SPO**	**DPO**
0 (read)	X	X	data_a	data_d
1 (read)	0	X	data_a	data_d
1 (read)	1	X	data_a	data_d
1 (write)	↓	D	D	data_d
1 (read)	↑	X	data_a	data_d
data_a = word addressed by bits A3:A0 data_d = word addressed by bits DPRA3:DPRA0				

Port Descriptions

Port	Direction	Width	Function
DPO	Output	1	Read-only 1-Bit data output
SPO	Output	1	R/W 1-Bit data output
A0	Input	1	R/W address[0] input
A1	Input	1	R/W address[1] input
A2	Input	1	R/W address[2] input
A3	Input	1	R/W address[3] input
D	Input	1	Write 1-Bit data input
DPRA0	Input	1	Read-only address[0] input
DPRA1	Input	1	Read-only address[1] input
DPRA2	Input	1	Read-only address[2] input
DPRA3	Input	1	Read-only address[3] input
WCLK	Input	1	Write clock input
WE	Input	1	Write enable input

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Initializes RAMs, registers, and look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM16X1D_1: 16 x 1 negative edge write, asynchronous read dual-port distributed RAM
--            Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM16X1D_1_inst : RAM16X1D_1
generic map (
   INIT => X"0000")
port map (
   DPO => DPO,      -- Read-only 1-bit data output for DPRA
   SPO => SPO,      -- R/W 1-bit data output for A0-A3
   A0 => A0,        -- R/W address[0] input bit
   A1 => A1,        -- R/W address[1] input bit
   A2 => A2,        -- R/W address[2] input bit
```

```
   A3 => A3,        -- R/W ddress[3] input bit
   D => D,          -- Write 1-bit data input
   DPRA0 => DPRA0, -- Read-only address[0] input bit
   DPRA1 => DPRA1, -- Read-only address[1] input bit
   DPRA2 => DPRA2, -- Read-only address[2] input bit
   DPRA3 => DPRA3, -- Read-only address[3] input bit
   WCLK => WCLK,    -- Write clock input
   WE => WE         -- Write enable input
);

-- End of RAM16X1D_1_inst instantiation
```

Verilog Instantiation Template

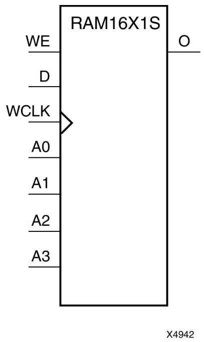
```
// RAM16X1D_1: 16 x 1 negative edge write, asynchronous read dual-port distributed RAM
//             Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM16X1D_1 #(
   .INIT(16'h0000) // Initial contents of RAM
) RAM16X1D_1_inst (
   .DPO(DPO),       // Read-only 1-bit data output
   .SPO(SPO),       // Rw/ 1-bit data output
   .A0(A0),         // Rw/ address[0] input bit
   .A1(A1),         // Rw/ address[1] input bit
   .A2(A2),         // Rw/ address[2] input bit
   .A3(A3),         // Rw/ address[3] input bit
   .D(D),           // Write 1-bit data input
   .DPRA0(DPRA0), // Read-only address[0] input bit
   .DPRA1(DPRA1), // Read-only address[1] input bit
   .DPRA2(DPRA2), // Read-only address[2] input bit
   .DPRA3(DPRA3), // Read-only address[3] input bit
   .WCLK(WCLK),    // Write clock input
   .WE(WE)         // Write enable input
);

// End of RAM16X1D_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM16X1S

Primitive: 16-Deep by 1-Wide Static Synchronous RAM



Introduction

This element is a 16-word by 1-bit static random access memory with synchronous write capability. When the write enable (WE) is set Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is set High, any positive transition on WCLK loads the data on the data input (D) into the word selected by the 4-bit address (A3:A0). This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins. You can initialize RAM16X1S during configuration using the INIT attribute.

Logic Table

Inputs			Outputs
WE(mode)	**WCLK**	**D**	**O**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↑	D	D
1 (read)	↓	X	Data
Data = word addressed by bits A3:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Specifies initial contents of the RAM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM16X1S: 16 x 1 posedge write distributed (LUT) RAM
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM16X1S_inst : RAM16X1S
generic map (
   INIT => X"0000")
port map (
   O => O,        -- RAM output
   A0 => A0,      -- RAM address[0] input
   A1 => A1,      -- RAM address[1] input
   A2 => A2,      -- RAM address[2] input
   A3 => A3,      -- RAM address[3] input
   D => D,        -- RAM data input
   WCLK => WCLK,  -- Write clock input
   WE => WE       -- Write enable input
);

-- End of RAM16X1S_inst instantiation
```

Verilog Instantiation Template

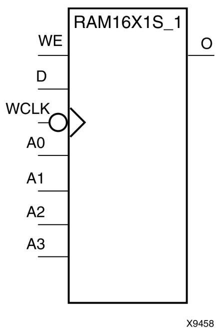
```
// RAM16X1S: 16 x 1 posedge write distributed (LUT) RAM
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM16X1S #(
   .INIT(16'h0000) // Initial contents of RAM
) RAM16X1S_inst (
   .O(O),        // RAM output
   .A0(A0),      // RAM address[0] input
   .A1(A1),      // RAM address[1] input
   .A2(A2),      // RAM address[2] input
   .A3(A3),      // RAM address[3] input
   .D(D),        // RAM data input
   .WCLK(WCLK), // Write clock input
   .WE(WE)       // Write enable input
);

// End of RAM16X1S_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM16X1S_1

Primitive: 16-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock



Introduction

This element is a 16-word by 1-bit static random access memory with synchronous write capability and negative-edge clock. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 4-bit address (A3:A0). For predictable performance, address and data inputs must be stable before a High-to-Low WCLK transition. This RAM block assumes an active-Low (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can initialize this element during configuration using the INIT attribute.

Logic Table

Inputs			**Outputs**
WE(mode)	**WCLK**	**D**	**O**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↓	D	D
1 (read)	↑	X	Data
Data = word addressed by bits A3:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Specifies initial contents of the RAM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM16X1S_1: 16 x 1 negedge write distributed (LUT) RAM
--            Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM16X1S_1_inst : RAM16X1S_1
generic map (
   INIT => X"0000")
port map (
   O => O,       -- RAM output
   A0 => A0,     -- RAM address[0] input
   A1 => A1,     -- RAM address[1] input
   A2 => A2,     -- RAM address[2] input
   A3 => A3,     -- RAM address[3] input
   D => D,       -- RAM data input
   WCLK => WCLK, -- Write clock input
   WE => WE      -- Write enable input
);

-- End of RAM16X1S_1_inst instantiation
```

Verilog Instantiation Template

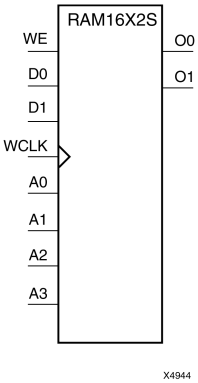
```
// RAM16X1S_1: 16 x 1 negedge write distributed (LUT) RAM
//            Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM16X1S_1 #(
   .INIT(16'h0000) // Initial contents of RAM
) RAM16X1S_1_inst (
   .O(O),       // RAM output
   .A0(A0),     // RAM address[0] input
   .A1(A1),     // RAM address[1] input
   .A2(A2),     // RAM address[2] input
   .A3(A3),     // RAM address[3] input
   .D(D),       // RAM data input
   .WCLK(WCLK), // Write clock input
   .WE(WE)      // Write enable input
);

// End of RAM16X1S_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM16X2S

Primitive: 16-Deep by 2-Wide Static Synchronous RAM



Introduction

This element is a 16-word by 2-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on the data input (D1:D0) into the word selected by the 4-bit address (A3:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O1:O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can use the INIT_xx properties to specify the initial contents of a wide RAM. INIT_00 initializes the RAM cells corresponding to the O0 output, INIT_01 initializes the cells corresponding to the O1 output, etc. For example, a RAM16X2S instance is initialized by INIT_00 and INIT_01 containing 4 hex characters each. A RAM16X8S instance is initialized by eight properties INIT_00 through INIT_07 containing 4 hex characters each. A RAM64x2S instance is completely initialized by two properties INIT_00 and INIT_01 containing 16 hex characters each.

Except for Virtex-4 devices, the initial contents of this element cannot be specified directly.

Logic Table

Inputs			Outputs
WE (mode)	**WCLK**	**D1:D0**	**O1:O0**
0 (read)	X	X	Data
1(read)	0	X	Data
1(read)	1	X	Data
1(write)	↑	D1:D0	D1:D0
1(read)	↓	X	Data
Data = word addressed by bits A3:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT_00 to INIT_01	Hexadecimal	Any 16-Bit Value	All zeros	Initializes RAMs, registers, and look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM16X2S: 16 x 2 posedge write distributed (LUT) RAM
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM16X2S_inst : RAM16X2S
generic map (
   INIT_00 => X"0000", --  INIT for bit 0 of RAM
   INIT_01 => X"0000") --  INIT for bit 1 of RAM
port map (
   O0 => O0,      -- RAM data[0] output
   O1 => O1,      -- RAM data[1] output
   A0 => A0,      -- RAM address[0] input
   A1 => A1,      -- RAM address[1] input
   A2 => A2,      -- RAM address[2] input
   A3 => A3,      -- RAM address[3] input
   D0 => D0,      -- RAM data[0] input
   D1 => D1,      -- RAM data[1] input
   WCLK => WCLK, -- Write clock input
   WE => WE       -- Write enable input
);

-- End of RAM16X2S_inst instantiation
```

Verilog Instantiation Template

```
// RAM16X2S: 16 x 2 posedge write distributed (LUT) RAM
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM16X2S #(
   .INIT_00(16'h0000), // Initial contents of bit 0 of RAM
   .INIT_01(16'h0000)  // Initial contents of bit 1 of RAM
) RAM16X2S_inst (
   .O0(O0),     // RAM data[0] output
   .O1(O1),     // RAM data[1] output
   .A0(A0),     // RAM address[0] input
   .A1(A1),     // RAM address[1] input
   .A2(A2),     // RAM address[2] input
   .A3(A3),     // RAM address[3] input
   .D0(D0),     // RAM data[0] input
   .D1(D1),     // RAM data[1] input
   .WCLK(WCLK), // Write clock input
```

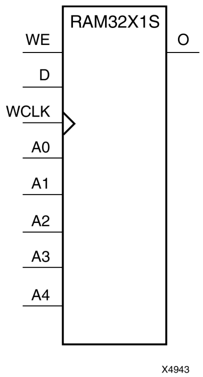
```
   .WE(WE)      // Write enable input
);

// End of RAM16X2S_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM32X1S

Primitive: 32-Deep by 1-Wide Static Synchronous RAM



Introduction

The design element is a 32-word by 1-bit static random access memory with synchronous write capability. When the write enable is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any positive transition on (WCLK) loads the data on the data input (D) into the word selected by the 5-bit address (A4-A0). For predictable performance, address and data inputs must be stable before a Low-to-High (WCLK) transition. This RAM block assumes an active-High (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins. You can initialize RAM32X1S during configuration using the INIT attribute.

Logic Table

Inputs			Outputs
WE (Mode)	**WCLK**	**D**	**O**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↑	D	D
1 (read)	↓	X	Data

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Descriptions
INIT	Hexadecimal	Any 32-Bit Value	All zeros	Specifies initial contents of the RAM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM32X1S: 32 x 1 posedge write distributed (LUT) RAM
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM32X1S_inst : RAM32X1S
generic map (
   INIT => X"00000000")
port map (
   O => O,        -- RAM output
   A0 => A0,      -- RAM address[0] input
   A1 => A1,      -- RAM address[1] input
   A2 => A2,      -- RAM address[2] input
   A3 => A3,      -- RAM address[3] input
   A4 => A4,      -- RAM address[4] input
   D => D,        -- RAM data input
   WCLK => WCLK, -- Write clock input
   WE => WE       -- Write enable input
);

-- End of RAM32X1S_inst instantiation
```

Verilog Instantiation Template

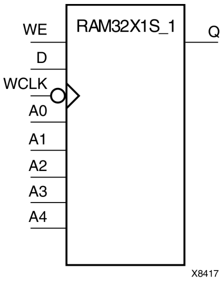
```
// RAM32X1S: 32 x 1 posedge write distributed (LUT) RAM
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM32X1S #(
   .INIT(32'h00000000)  // Initial contents of RAM
) RAM32X1S_inst (
   .O(O),        // RAM output
   .A0(A0),      // RAM address[0] input
   .A1(A1),      // RAM address[1] input
   .A2(A2),      // RAM address[2] input
   .A3(A3),      // RAM address[3] input
   .A4(A4),      // RAM address[4] input
   .D(D),        // RAM data input
   .WCLK(WCLK), // Write clock input
   .WE(WE)       // Write enable input
);

// End of RAM32X1S_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM32X1S_1

Primitive: 32-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock



Introduction

The design element is a 32-word by 1-bit static random access memory with synchronous write capability. When the write enable is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 5-bit address (A4:A0). For predictable performance, address and data inputs must be stable before a High-to-Low (WCLK) transition. This RAM block assumes an active-Low (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins. You can initialize RAM32X1S_1 during configuration using the INIT attribute.

Logic Table

Inputs			**Outputs**
WE (Mode)	**WCLK**	**D**	**O**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↓	D	D
1 (read)	↑	X	Data
Data = word addressed by bits A4:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	**Data Type**	**Allowed Values**	**Default**	**Descriptions**
INIT	Hexadecimal	Any 32-Bit Value	0	Initializes RAMs, registers, and look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM32X1S_1: 32 x 1 negedge write distributed (LUT) RAM
--             Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM32X1S_1_inst : RAM32X1S_1
generic map (
   INIT => X"00000000")
port map (
   O => O,        -- RAM output
   A0 => A0,      -- RAM address[0] input
   A1 => A1,      -- RAM address[1] input
   A2 => A2,      -- RAM address[2] input
   A3 => A3,      -- RAM address[3] input
   A4 => A4,      -- RAM address[4] input
   D => D,        -- RAM data input
   WCLK => WCLK, -- Write clock input
   WE => WE       -- Write enable input
);

-- End of RAM32X1S_1_inst instantiation
```

Verilog Instantiation Template

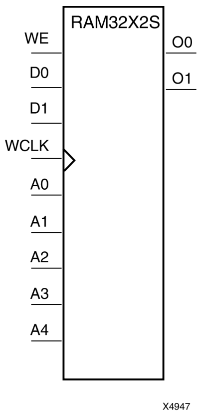
```
// RAM32X1S_1: 32 x 1 negedge write distributed (LUT) RAM
//             Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM32X1S_1 #(
   .INIT(32'h00000000)  // Initial contents of RAM
)RAM32X1S_1_inst (
   .O(O),        // RAM output
   .A0(A0),      // RAM address[0] input
   .A1(A1),      // RAM address[1] input
   .A2(A2),      // RAM address[2] input
   .A3(A3),      // RAM address[3] input
   .A4(A4),      // RAM address[4] input
   .D(D),        // RAM data input
   .WCLK(WCLK), // Write clock input
   .WE(WE)       // Write enable input
);

// End of RAM32X1S_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM32X2S

Primitive: 32-Deep by 2-Wide Static Synchronous RAM



Introduction

The design element is a 32-word by 2-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any positive transition on (WCLK) loads the data on the data input (D1-D0) into the word selected by the 5-bit address (A4-A0). For predictable performance, address and data inputs must be stable before a Low-to-High (WCLK) transition. This RAM block assumes an active-High (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block. The signal output on the data output pins (O1-O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can use the INIT_00 and INIT_01 properties to specify the initial contents of RAM32X2S.

Logic Table

Inputs			**Outputs**
WE (Mode)	**WCLK**	**D**	**O0-O1**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↑	D1:D0	D1:D0
1 (read)	↓	X	Data
Data = word addressed by bits A4:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Send Feedback

Available Attributes

Attribute	Data Type	Allowed Values	Default	Descriptions
INIT_00	Hexadecimal	Any 32-Bit Value	All zeros	INIT for bit 0 of RAM.
INIT_01	Hexadecimal	Any 32-Bit Value	All zeros	INIT for bit 1 of RAM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM32X2S: 32 x 2 posedge write distributed (LUT) RAM
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM32X2S_inst : RAM32X2S
generic map (
   INIT_00 => X"00000000", --  INIT for bit 0 of RAM
   INIT_01 => X"00000000") --  INIT for bit 1 of RAM
port map (
   O0 => O0,      -- RAM data[0] output
   O1 => O1,      -- RAM data[1] output
   A0 => A0,      -- RAM address[0] input
   A1 => A1,      -- RAM address[1] input
   A2 => A2,      -- RAM address[2] input
   A3 => A3,      -- RAM address[3] input
   A4 => A4,      -- RAM address[4] input
   D0 => D0,      -- RAM data[0] input
   D1 => D1,      -- RAM data[1] input
   WCLK => WCLK, -- Write clock input
   WE => WE      -- Write enable input
);

-- End of RAM32X2S_inst instantiation
```

Verilog Instantiation Template

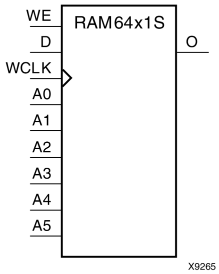
```
// RAM32X2S: 32 x 2 posedge write distributed (LUT) RAM
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM32X2S #(
   .INIT_00(32'h00000000), // INIT for bit 0 of RAM
   .INIT_01(32'h00000000)  // INIT for bit 1 of RAM
) RAM32X2S_inst (
   .O0(O0),     // RAM data[0] output
   .O1(O1),     // RAM data[1] output
   .A0(A0),     // RAM address[0] input
   .A1(A1),     // RAM address[1] input
   .A2(A2),     // RAM address[2] input
   .A3(A3),     // RAM address[3] input
   .A4(A4),     // RAM address[4] input
   .D0(D0),     // RAM data[0] input
   .D1(D1),     // RAM data[1] input
   .WCLK(WCLK), // Write clock input
   .WE(WE)      // Write enable input
);

// End of RAM32X2S_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM64X1S

Primitive: 64-Deep by 1-Wide Static Synchronous RAM



Introduction

This design element is a 64-word by 1-bit static random access memory (RAM) with synchronous write capability. When the write enable is set Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is set High, any positive transition on WCLK loads the data on the data input (D) into the word selected by the 6-bit address (A5:A0). This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can initialize this element during configuration using the INIT attribute.

Logic Table

Mode selection is shown in the following logic table

Inputs			Outputs
WE (mode)	**WCLK**	**D**	**O**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↑	D	D
1 (read)	↓	X	Data
Data = word addressed by bits A5:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 64-Bit Value	All zeros	Initializes ROMs, RAMs, registers, and look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM64X1S: 64 x 1 positive edge write, asynchronous read single-port distributed RAM
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM64X1S_inst : RAM64X1S
generic map (
   INIT => X"0000000000000000")
port map (
   O => O,        -- 1-bit data output
   A0 => A0,      -- Address[0] input bit
   A1 => A1,      -- Address[1] input bit
   A2 => A2,      -- Address[2] input bit
   A3 => A3,      -- Address[3] input bit
   A4 => A4,      -- Address[4] input bit
   A5 => A5,      -- Address[5] input bit
   D => D,        -- 1-bit data input
   WCLK => WCLK,  -- Write clock input
   WE => WE       -- Write enable input
);

-- End of RAM64X1S_inst instantiation
```

Verilog Instantiation Template

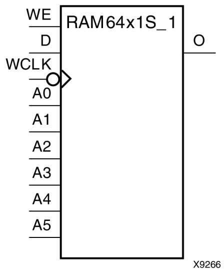
```
// RAM64X1S: 64 x 1 positive edge write, asynchronous read single-port distributed RAM
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM64X1S #(
   .INIT(64'h0000000000000000) // Initial contents of RAM
) RAM64X1S_inst (
   .O(O),        // 1-bit data output
   .A0(A0),      // Address[0] input bit
   .A1(A1),      // Address[1] input bit
   .A2(A2),      // Address[2] input bit
   .A3(A3),      // Address[3] input bit
   .A4(A4),      // Address[4] input bit
   .A5(A5),      // Address[5] input bit
   .D(D),        // 1-bit data input
   .WCLK(WCLK),  // Write clock input
   .WE(WE)       // Write enable input
);

// End of RAM64X1S_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAM64X1S_1

Primitive: 64-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock



Introduction

This design element is a 64-word by 1-bit static random access memory with synchronous write capability. When the write enable is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 6-bit address (A5:A0). For predictable performance, address and data inputs must be stable before a High-to-Low (WCLK) transition. This RAM block assumes an active-Low (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can initialize this element during configuration using the INIT attribute.

Logic Table

Inputs			Outputs
WE (mode)	**WCLK**	**D**	**O**
0 (read)	X	X	Data
1 (read)	0	X	Data
1 (read)	1	X	Data
1 (write)	↓	D	D
1 (read)	↑	X	Data
Data = word addressed by bits A5:A0			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 64-Bit Value	All zeros	Initializes ROMs, RAMs, registers, and look-up tables.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAM64X1S_1: 64 x 1 negative edge write, asynchronous read single-port distributed RAM
--             Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAM64X1S_1_inst : RAM64X1S_1
generic map (
   INIT => X"0000000000000000")
port map (
   O => O,        -- 1-bit data output
   A0 => A0,      -- Address[0] input bit
   A1 => A1,      -- Address[1] input bit
   A2 => A2,      -- Address[2] input bit
   A3 => A3,      -- Address[3] input bit
   A4 => A4,      -- Address[4] input bit
   A5 => A5,      -- Address[5] input bit
   D => D,        -- 1-bit data input
   WCLK => WCLK,  -- Write clock input
   WE => WE       -- Write enable input
);

-- End of RAM64X1S_1_inst instantiation
```

Verilog Instantiation Template

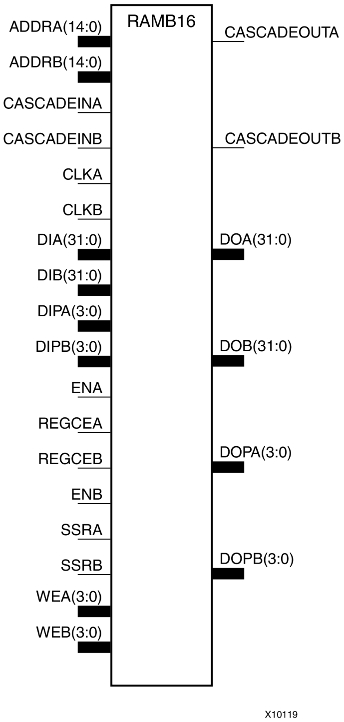
```
// RAM64X1S_1: 64 x 1 negative edge write, asynchronous read single-port distributed RAM
//             Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAM64X1S_1 #(
   .INIT(64'h0000000000000000) // Initial contents of RAM
) RAM64X1S_1_inst (
   .O(O),       // 1-bit data output
   .A0(A0),     // Address[0] input bit
   .A1(A1),     // Address[1] input bit
   .A2(A2),     // Address[2] input bit
   .A3(A3),     // Address[3] input bit
   .A4(A4),     // Address[4] input bit
   .A5(A5),     // Address[5] input bit
   .D(D),       // 1-bit data input
   .WCLK(WCLK), // Write clock input
   .WE(WE)      // Write enable input
);

// End of RAM64X1S_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAMB16

Primitive: 16K-bit Data and 2K-bit Parity Single-Port Synchronous Block RAM with Configurable Port Widths



Introduction

In addition to distributed RAM memory, Virtex®-4 and above devices feature a large number of 18 kB block RAM memories. This block RAM memory is a True Dual-Port RAM, offering fast, discrete, and large blocks of memory in the device. The memory is organized in columns, and the total amount of block RAM memory depends on the size of the device. The 18 kB blocks are cascadable to enable a deeper and wider memory implementation, with a minimal timing penalty incurred through specialized routing resources.

Read Operation	The read operation uses one clock edge. The read address is registered on the read port, and the stored data is loaded into the output latches after the RAM access interval passes.
Write Operation	A write operation is a single clock-edge operation. The write address is registered on the write port, and the data input is stored in memory.

Write Operating Modes

There are three options for the behavior of the data output during a write operation on its port. The "read during write" mode offers the flexibility of using the data output bus during a write operation on the same port. Output behavior is determined by the configuration. This choice increases the efficiency of block RAM memory at each clock cycle and allows designs that use maximum bandwidth.

Three different modes are used to determine data available on the output latches after a write clock edge.

WRITE_FIRST or Transparent Mode (Default)	The input data is simultaneously written into memory and stored in the data output (transparent write).
READ_FIRST or Read-Before-Write Mode	Data previously stored at the write address appears on the output latches, while the input data is being stored in memory (read before write).
NO_CHANGE Mode	The output latches remain unchanged during a write operation.

Mode selection is set by configuration. One of these three modes is set individually for each port by an attribute. The default mode is WRITE_FIRST.

Port Descriptions

Output Latches Synchronous Set/Reset - SRVAL (SRVAL_A & SRVAL_B)

The SRVAL_A and SRVAL_B (dual-port) attributes define output latch values when the SSR input is asserted. The width of the SRVAL (SRVAL_A and SRVAL_B) attribute is the port width, as shown in the following table:

Port Width Values and Data Width	DOP Bus	DO Bus	SRVAL
1	NA	<0>	1
2	NA	<1:0>	2
4	NA	<3:0>	4
9	<0>	<7:0>	(1+8) = 9
18	<1:0>	<15:0>	(2+16) = 18
36	<3:0>	<31:0>	(4 + 32) = 36

Optional Output Register On/Off Switch - DO[A/B]_REG

This attribute sets the number of pipeline register at A/B output of RAMB16. The valid values are 0 (default) or 1.

Clock Inversion at Output Register Switch - INVERT_CLK_DO[A/B]_REG

When set to TRUE, the clock input to the pipeline register at A/B output of RAMB16 is inverted. The default value is FALSE.

Extended Mode Address Determinant - RAM_EXTENSION_[A/B]

This attribute determines whether the block RAM of interest has its A/B port as UPPER/LOWER address when using the cascade mode. In the cascading mode, READ_WIDTH_[A/B] and WRITE_WIDTH_[A/B] should be set to 1. When the block RAM is not used in cascade mode, the default value is NONE.

Read Width - READ_WIDTH_[A/B]

This attribute determines the A/B read port width of the block RAM. The valid values are: 0 (default), 1, 2, 4, 9, 18, and 36. The READ_WIDTH_[A/B] for both the ports should not be set to zero at the same time.

Write Width - WRITE_WIDTH_[A/B]

This attribute determines the A/B write port width of the block RAM. The valid values are: 0 (default), 1, 2, 4, 9, 18, and 36.

Write Mode - WRITE_MODE_[A/B]

This attribute determines the write mode of the A/B input ports. The possible values are WRITE_FIRST (default), READ_FIRST, and NO_CHANGE.

RAMB16 Location Constraints

Block RAM instances can have LOC properties attached to them to constrain placement. Block RAM placement locations differ from the convention used for naming CLB locations, allowing LOC properties to transfer easily from array to array. The LOC properties use the following form: LOC = RAMB16_X#Y#

The RAMB16_X0Y0 is the bottom-left block RAM location on the device.

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	Yes
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
DOA_REG	Integer	0, 1	0	Optional output registers on A port
DOB_REG	Integer	0, 1	0	Optional output registers on B port.
INIT_00 to INIT_39	Hexa-decimal	Any 256-Bit Value	All zeros	To change the initial contents of the RAM to anything other than all zero's.
INIT_0A to INIT_0F	Hexa-decimal	Any 256-Bit Value	All zeros	To change the initial contents of the RAM to anything other than all zero's.
INIT_1A to INIT_1F	Hexa-decimal	Any 256-Bit Value	All zeros	To change the initial contents of the RAM to anything other than all zero's.
INIT_2A to INIT_2F	Hexa-decimal	Any 256-Bit Value	All zeros	To change the initial contents of the RAM to anything other than all zero's.
INIT_3A to INIT_3F	Hexa-decimal	Any 256-Bit Value	All zeros	To change the initial contents of the RAM to anything other than all zeros.
INIT_A	Hexa-decimal	Any 36-Bit Value	All zeros	Initial values on A output port.
INIT_B	Hexa-decimal	Any 36-Bit Value	All zeros	Initial values on B output port.
INITP_00 to INITP_07	Hexa-decimal	Any 256-Bit Value	All zeros	Applied for the parity bits.
INVERT_CLK_DOA_REG	Boolean	FALSE, TRUE	FALSE	Invert clock on A port output registers.
INVERT_CLK_DOB_REG	Boolean	FALSE, TRUE	FALSE	Invert clock on B port output registers.
RAM_EXTENSION_A	String	"LOWER", "NONE" or "UPPER"	"NONE"	Allowed value when cascaded.
RAM_EXTENSION_B	String	"LOWER", "NONE" or "UPPER"	"NONE"	Allowed value when cascaded.
READ_WIDTH_A	Integer	0, 1, 2, 4, 9, 18 or 36	0	Determines the A read port width of the block RAM.

READ_WIDTH_B	Integer	0, 1, 2, 4, 9, 18 or 36	0	Determines the B read port width of the block RAM.
SIM_COLLISION_ CHECK	String"	"ALL", "WARNING_ ONLY", "GENERATE_X_ ONLY", or "NONE"	"ALL"	Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows: • "ALL" - Warning produced and affected outputs/memory location go unknown (X). • "WARNING_ONLY" - Warning produced and affected outputs/memory retain last value. • "GENERATE_X_ONLY" - No warning. However, affected outputs/memory go unknown (X). • "NONE" - No warning and affected outputs/memory retain last value. **Note** Setting this to a value other than "ALL" can allow problems in the design go unnoticed during simulation. Care should be taken when changing the value of this attribute. Please see the *Synthesis and Simulation Design Guide* for more information.
SRVAL_A	Hexa- decimal	Any 36-Bit Value.	All zeros	Use to set/reset value for A port output.
SRVAL_B	Hexa- decimal	Any 36-Bit Value.	All zeros	Use to set/reset value for B port output.
WRITE_MODE_A	String	"WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"	"WRITE_FIRST"	Configures Port A (Sn) of a dual-port RAMB16 to support one of three write modes.
WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"	"WRITE_FIRST"	Configures Port B (Sn) of a dual-port RAMB16 to support one of three write modes.
WRITE_WIDTH_A	Integer	0, 1, 2, 4, 9, 18 or 36	0	Determines the A write port width of the block RAM.
WRITE_WIDTH_B	Integer	0, 1, 2, 4, 9, 18 or 36	0	Determines the B write port width of the block RAM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAMB16: 16k+2k Parity Parameterizable BlockRAM
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAMB16_inst : RAMB16
generic map (
   DOA_REG => 0, -- Optional output registers on the A port (0 or 1)
```

```
DOB_REG => 0, -- Optional output registers on the B port (0 or 1)
INIT_A => X"000000000", --  Initial values on A output port
INIT_B => X"000000000", --  Initial values on B output port
INVERT_CLK_DOA_REG => FALSE, -- Invert clock on A port output registers (TRUE or FALSE)
INVERT_CLK_DOB_REG => FALSE, -- Invert clock on B port output registers (TRUE or FALSE)
RAM_EXTENSION_A => "NONE", -- "UPPER", "LOWER" or "NONE" when cascaded
RAM_EXTENSION_B => "NONE", -- "UPPER", "LOWER" or "NONE" when cascaded
READ_WIDTH_A => 0, -- Valid values are 1,2,4,9,18 or 36
READ_WIDTH_B => 0, -- Valid values are 1,2,4,9,18 or 36
SIM_COLLISION_CHECK => "ALL", -- Collision check enable "ALL", "WARNING_ONLY",
                              -- "GENERATE_X_ONLY" or "NONE"
SRVAL_A => X"000000000", --  Port A output value upon SSR assertion
SRVAL_B => X"000000000", --  Port B output value upon SSR assertion
WRITE_MODE_A => "WRITE_FIRST", --  WRITE_FIRST, READ_FIRST or NO_CHANGE
WRITE_MODE_B => "WRITE_FIRST", --  WRITE_FIRST, READ_FIRST or NO_CHANGE
WRITE_WIDTH_A => 0, -- Valid values are 1,2,4,9,18 or 36
WRITE_WIDTH_B => 0, -- Valid values are 1,2,4,9,18 or 36
-- The following INIT_xx declarations specify the initial contents of the RAM
INIT_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_07 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_0F => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_10 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_11 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_12 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_13 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_14 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_15 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_16 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_17 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_18 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_19 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1F => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_20 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_21 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_22 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_23 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_24 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_25 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_26 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_27 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_28 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_29 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2F => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_30 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_31 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_32 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_33 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_34 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_35 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_36 => X"0000000000000000000000000000000000000000000000000000000000000000",
```

```
   INIT_37 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_38 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_39 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_3A => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_3B => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_3C => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_3D => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_3E => X"0000000000000000000000000000000000000000000000000000000000000000",
   INIT_3F => X"0000000000000000000000000000000000000000000000000000000000000000",
   -- The next set of INITP_xx are for the parity bits
   INITP_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
   INITP_07 => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
   CASCADEOUTA => CASCADEOUTA, -- 1-bit cascade output
   CASCADEOUTB => CASCADEOUTB, -- 1-bit cascade output
   DOA => DOA,       -- 32-bit A port Data Output
   DOB => DOB,       -- 32-bit B port Data Output
   DOPA => DOPA,     -- 4-bit  A port Parity Output
   DOPB => DOPB,     -- 4-bit  B port Parity Output
   ADDRA => ADDRA,   -- 15-bit A port Address Input
   ADDRB => ADDRB,   -- 15-bit B port Address Input
   CASCADEINA => CASCADEINA, -- 1-bit cascade A input
   CASCADEINB => CASCADEINB, -- 1-bit cascade B input
   CLKA => CLKA,     -- Port A Clock
   CLKB => CLKB,     -- Port B Clock
   DIA => DIA,       -- 32-bit A port Data Input
   DIB => DIB,       -- 32-bit B port Data Input
   DIPA => DIPA,     -- 4-bit  A port parity Input
   DIPB => DIPB,     -- 4-bit  B port parity Input
   ENA => ENA,       -- 1-bit  A port Enable Input
   ENB => ENB,       -- 1-bit  B port Enable Input
   REGCEA => REGCEA, -- 1-bit A port register enable input
   REGCEB => REGCEB, -- 1-bit B port register enable input
   SSRA => SSRA,     -- 1-bit  A port Synchronous Set/Reset Input
   SSRB => SSRB,     -- 1-bit  B port Synchronous Set/Reset Input
   WEA => WEA,       -- 4-bit  A port Write Enable Input
   WEB => WEB        -- 4-bit  B port Write Enable Input
);

-- End of RAMB16_inst instantiation
```

Verilog Instantiation Template

```
// RAMB16: 16k+2k Parity parameterizable BlockRAM
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAMB16 #(
   .DOA_REG(0),  // Optional output registers on A port (0 or 1)
   .DOB_REG(0),  // Optional output registers on B port (0 or 1)
   .INIT_A(36'h000000000),  // Initial values on A output port
   .INIT_B(36'h000000000),  // Initial values on B output port
   .INVERT_CLK_DOA_REG("FALSE"),  // Invert clock on A port output registers ("TRUE" or "FALSE")
   .INVERT_CLK_DOB_REG("FALSE"),  // Invert clock on A port output registers ("TRUE" or "FALSE")
   .RAM_EXTENSION_A("NONE"),  // "UPPER", "LOWER" or "NONE" when cascaded
   .RAM_EXTENSION_B("NONE"),  // "UPPER", "LOWER" or "NONE" when cascaded
   .READ_WIDTH_A(0),  // Valid values are 1, 2, 4, 9, 18, or 36
   .READ_WIDTH_B(0),  // Valid values are 1, 2, 4, 9, 18, or 36
   .SIM_COLLISION_CHECK("ALL"),  // Collision check enable "ALL", "WARNING_ONLY",
                                 //   "GENERATE_X_ONLY" or "NONE"
   .SRVAL_A(36'h000000000), // Set/Reset value for A port output
   .SRVAL_B(36'h000000000),  // Set/Reset value for B port output
   .WRITE_MODE_A("WRITE_FIRST"),  // "WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"
   .WRITE_MODE_B("WRITE_FIRST"),  // "WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"
   .WRITE_WIDTH_A(2),  // Valid values are 1, 2, 4, 9, 18, or 36
```

```
.WRITE_WIDTH_B(0),  // Valid values are 1, 2, 4, 9, 18, or 36

// The forllowing INIT_xx declarations specify the initial contents of the RAM
.INIT_00(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_01(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_02(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_03(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_04(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_05(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_06(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_07(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_08(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_09(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_0A(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_0B(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_0C(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_0D(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_0E(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_0F(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_10(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_11(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_12(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_13(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_14(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_15(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_16(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_17(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_18(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_19(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_1A(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_1B(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_1C(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_1D(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_1E(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_1F(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_20(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_21(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_22(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_23(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_24(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_25(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_26(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_27(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_28(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_29(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_2A(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_2B(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_2C(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_2D(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_2E(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_2F(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_30(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_31(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_32(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_33(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_34(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_35(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_36(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_37(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_38(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_39(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_3A(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_3B(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_3C(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_3D(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_3E(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INIT_3F(256'h0000000000000000000000000000000000000000000000000000000000000000),

// The next set of INITP_xx are for the parity bits
.INITP_00(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INITP_01(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INITP_02(256'h0000000000000000000000000000000000000000000000000000000000000000),
.INITP_03(256'h0000000000000000000000000000000000000000000000000000000000000000),
```

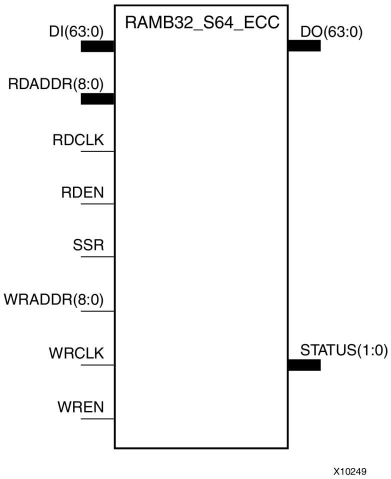
```
   .INITP_04(256'h0000000000000000000000000000000000000000000000000000000000000000),
   .INITP_05(256'h0000000000000000000000000000000000000000000000000000000000000000),
   .INITP_06(256'h0000000000000000000000000000000000000000000000000000000000000000),
   .INITP_07(256'h0000000000000000000000000000000000000000000000000000000000000000)
) RAMB16_inst (
   .CASCADEOUTA(CASCADEOUTA),  // 1-bit cascade output
   .CASCADEOUTB(CASCADEOUTB),  // 1-bit cascade output
   .DOA(DOA),      // 32-bit A port data output
   .DOB(DOB),      // 32-bit B port data output
   .DOPA(DOPA),    // 4-bit A port parity data output
   .DOPB(DOPB),    // 4-bit B port parity data output
   .ADDRA(ADDRA),  // 15-bit A port address input
   .ADDRB(ADDRB),  // 15-bit B port address input
   .CASCADEINA(CASCADEINA), // 1-bit cascade A input
   .CASCADEINB(CASCADEINB), // 1-bit cascade B input
   .CLKA(CLKA),     // 1-bit A port clock input
   .CLKB(CLKB),     // 1-bit B port clock input
   .DIA(DIA),       // 32-bit A port data input
   .DIB(DIB),       // 32-bit B port data input
   .DIPA(DIPA),     // 4-bit A port parity data input
   .DIPB(DIPB),     // 4-bit B port parity data input
   .ENA(ENA),       // 1-bit A port enable input
   .ENB(ENB),       // 1-bit B port enable input
   .REGCEA(REGCEA), // 1-bit A port register enable input
   .REGCEB(REGCEB), // 1-bit B port register enable input
   .SSRA(SSRA),     // 1-bit A port set/reset input
   .SSRB(SSRB),     // 1-bit B port set/reset input
   .WEA(WEA),       // 4-bit A port write enable input
   .WEB(WEB)        // 4-bit B port write enable input
);

// End of RAMB16_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

RAMB32_S64_ECC

Primitive: 512 Deep by 64-Bit Wide Synchronous, Two-Port Block RAM with Built-In Error Correction



Introduction

Two vertically adjacent block RAMs can be configured as a single 512 x 64 RAM with built in Hamming error correction, using the extra eight bits in the 72-bit wide RAM. The operation is transparent to you. The eight protection bits are generated during each write operation, and are used during each read operation to correct any single error, or to detect (but not correct) any double error. Two status outputs indicate the three possible read results: No error, single error corrected, double error detected. The read operation does not correct the error in the memory array, it only presents corrected data on DOUT.

This error correction code (ECC) configuration option is available with any block RAM pair, but cannot use the one block RAM immediately above or below the Virtex®-4 PowerPC®™ blocks.

Port Descriptions

Port	Direction	Function
DIN<63:0>	Input	Data input bus
WRADDR<8:0>	Input	Write address bus
RDADDR<8:0>	Input	Read address bus
WREN	Input	Write enable. When WREN = 1, data will be written into memory. When WREN = 0, write is disabled.
RDEN	Input	Read enable. When RDEN = 1, data will be read from memory. When RDEN = 0, read is disabled.
SSR	Input	Set/Reset output registers (not the memory content)
WRCLK	Input	Clock for write operations
RDCLK	Input	Clock for read operations
DOUT<63:0>	Output	Data output bus
STATUS<1:0>(1)	Output	Error status bus

Note Hamming code implemented in the block RAM ECC logic detects one of three conditions: no detectable error, single-bit error detected and corrected on DOUT (but not corrected in the memory), and double-bit error detected without correction. The result of STATUS<1:0> indicates these three conditions.

STATUS[1:0]	Function
0	No bit error.
1	Single-bit error. The block RAM ECC macro detects and automatically corrects a single-bit error.
10	Double-bit error. The block RAM ECC macro detects a double-bit error.
11	Indeterminate state. The Hamming code implemented in the block RAM ECC cannot generate a predictable status if STATUS<1:0> is equal to three. Designers must ensure that the data has at most double-bit errors for the STATUS<1:0> to generate the proper indicator.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
DO_REG	Integer	0, 1	0	Optional output registers on A port .
SIM_COLLISION_ CHECK	String	"ALL", "NONE", "WARNING_ ONLY" or "GENERATE_X_ ONLY"	"ALL"	Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows: • "ALL" - Warning produced and affected outputs/memory location go unknown (X). • "WARNING_ONLY" - Warning produced and affected outputs/memory retain last value. • "GENERATE_X_ONLY" - No warning. However, affected outputs/memory go unknown (X). • "NONE" - No warning and affected outputs/memory retain last value. **Note** Setting this to a value other than "ALL" can allow problems in the design go unnoticed during simulation. Care should be taken when changing the value of this attribute. Please see the *Synthesis and Simulation Design Guide* for more information.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- RAMB32_S64_ECC: 512 x 64 Error Correction BlockRAM
--                 Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

RAMB32_S64_ECC_inst: RAMB32_S64_ECC_inst
   generic map(
      DO_REG => 0, -- Optional output registers (0 or 1)
      SIM_COLLISION_CHECK => "ALL") -- Collision check enable "ALL", "WARNING_ONLY", "GENERATE_X_ONLY"
   port map (
      DO => DO,          -- 64-bit output data
      STATUS => STATUS, -- 2-bit status output
      DI => DI,          -- 64-bit data input
      RDADDR => RDADDR, -- 9-bit data address input
      RDCLK => RDCLK,    -- 1-bit read clock input
      RDEN => RDEN,      -- 1-bit read enable input
      SSR => '0',        -- Always tie to ground
      WRADDR =>WRADDR,  -- 9-bit write address input
      WRCLK => WRCLK,    -- 1-bit write clock input
      WREN => WREN       -- 1-bit write enable input
   );

-- End of RAMB32_S64_ECC_inst instantiation
```

Verilog Instantiation Template

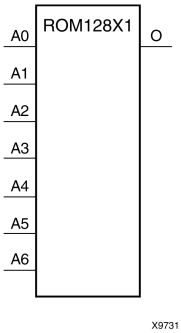
```
// RAMB32_S64_ECC: 512 x 64 Error Correction BlockRAM
//                 Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

RAMB32_S64_ECC #(
   .DO_REG(0),      // Optional output registers (0 or 1)
   .SIM_COLLISION_CHECK("ALL") // Collision check enable "ALL",
                               //"WARNING_ONLY", "GENERATE_X_ONLY"
) RAMB32_S64_ECC_inst (
   .DO(DO),          // 64-bit output data
   .STATUS(STATUS), // 2-bit status output
   .DI(DI),          // 64-bit data input
   .RDADDR(RDADDR), // 9-bit data address input
   .RDCLK(RDCLK),    // 1-bit read clock input
   .RDEN(RDEN),      // 1-bit read enable input
   .SSR(1'b0),       // Always tie to ground
   .WRADDR(WRADDR), // 9-bit write address input
   .WRCLK(WRCLK),    // 1-bit write clock input
   .WREN(WREN)       // 1-bit write enable input
);

// End of RAMB32_S64_ECC_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ROM128X1

Primitive: 128-Deep by 1-Wide ROM



Introduction

This design element is a 128-word by 1-bit read-only memory. The data output (O) reflects the word selected by the 7-bit address (A6:A0). The ROM is initialized to a known value during configuration with the INIT=value parameter. The value consists of 32 hexadecimal digits that are written into the ROM from the most-significant digit A=FH to the least-significant digit A=0H. An error occurs if the INIT=value is not specified.

Logic Table

Input				Output
I0	I1	I2	I3	O
0	0	0	0	INIT(0)
0	0	0	1	INIT(1)
0	0	1	0	INIT(2)
0	0	1	1	INIT(3)
0	1	0	0	INIT(4)
0	1	0	1	INIT(5)
0	1	1	0	INIT(6)
0	1	1	1	INIT(7)
1	0	0	0	INIT(8)
1	0	0	1	INIT(9)
1	0	1	0	INIT(10)
1	0	1	1	INIT(11)
1	1	0	0	INIT(12)
1	1	0	1	INIT(13)
1	1	1	0	INIT(14)
1	1	1	1	INIT(15)

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 128-Bit Value	All zeros	Specifies the contents of the ROM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ROM128X1: 128 x 1 Asynchronous Distributed (LUT) ROM
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ROM128X1_inst : ROM128X1
generic map (
   INIT => X"00000000000000000000000000000000")
port map (
   O => O,   -- ROM output
   A0 => A0, -- ROM address[0]
   A1 => A1, -- ROM address[1]
   A2 => A2, -- ROM address[2]
   A3 => A3, -- ROM address[3]
   A4 => A4, -- ROM address[4]
   A5 => A5, -- ROM address[5]
   A6 => A6  -- ROM address[6]
);

-- End of ROM128X1_inst instantiation
```

Verilog Instantiation Template

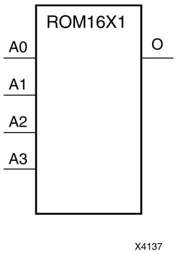
```
// ROM128X1: 128 x 1 Asynchronous Distributed (LUT) ROM
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ROM128X1 #(
   .INIT(128'h00000000000000000000000000000000) // Contents of ROM
) ROM128X1_inst (
   .O(O),   // ROM output
   .A0(A0), // ROM address[0]
   .A1(A1), // ROM address[1]
   .A2(A2), // ROM address[2]
   .A3(A3), // ROM address[3]
   .A4(A4), // ROM address[4]
   .A5(A5), // ROM address[5]
   .A6(A6)  // ROM address[6]
);

// End of ROM128X1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ROM16X1

Primitive: 16-Deep by 1-Wide ROM



Introduction

This design element is a 16-word by 1-bit read-only memory. The data output (O) reflects the word selected by the 4-bit address (A3:A0). The ROM is initialized to a known value during configuration with the INIT=value parameter. The value consists of four hexadecimal digits that are written into the ROM from the most-significant digit A=FH to the least-significant digit A=0H. For example, the INIT=10A7 parameter produces the data stream: 0001 0000 1010 0111 An error occurs if the INIT=value is not specified.

Logic Table

Input				Output
I0	**I1**	**I2**	**I3**	**O**
0	0	0	0	INIT(0)
0	0	0	1	INIT(1)
0	0	1	0	INIT(2)
0	0	1	1	INIT(3)
0	1	0	0	INIT(4)
0	1	0	1	INIT(5)
0	1	1	0	INIT(6)
0	1	1	1	INIT(7)
1	0	0	0	INIT(8)
1	0	0	1	INIT(9)
1	0	1	0	INIT(10)
1	0	1	1	INIT(11)
1	1	0	0	INIT(12)
1	1	0	1	INIT(13)
1	1	1	0	INIT(14)
1	1	1	1	INIT(15)

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Specifies the contents of the ROM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ROM16X1: 16 x 1 Asynchronous Distributed (LUT) ROM
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ROM16X1_inst : ROM16X1
generic map (
   INIT => X"0000")
port map (
   O => O,    -- ROM output
   A0 => A0, -- ROM address[0]
   A1 => A1, -- ROM address[1]
   A2 => A2, -- ROM address[2]
   A3 => A3  -- ROM address[3]
);

-- End of ROM16X1_inst instantiation
```

Verilog Instantiation Template

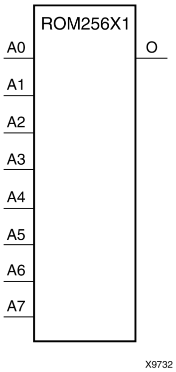
```
// ROM16X1: 16 x 1 Asynchronous Distributed (LUT) ROM
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ROM16X1 #(
   .INIT(16'h0000) // Contents of ROM
) ROM16X1_inst (
   .O(O),    // ROM output
   .A0(A0), // ROM address[0]
   .A1(A1), // ROM address[1]
   .A2(A2), // ROM address[2]
   .A3(A3)  // ROM address[3]
);

// End of ROM16X1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ROM256X1

Primitive: 256-Deep by 1-Wide ROM



Introduction

This design element is a 256-word by 1-bit read-only memory. The data output (O) reflects the word selected by the 8-bit address (A7:A0). The ROM is initialized to a known value during configuration with the INIT=value parameter. The value consists of 64 hexadecimal digits that are written into the ROM from the most-significant digit A=FH to the least-significant digit A=0H.

An error occurs if the INIT=value is not specified.

Logic Table

Input				Output
I0	**I1**	**I2**	**I3**	**O**
0	0	0	0	INIT(0)
0	0	0	1	INIT(1)
0	0	1	0	INIT(2)
0	0	1	1	INIT(3)
0	1	0	0	INIT(4)
0	1	0	1	INIT(5)
0	1	1	0	INIT(6)
0	1	1	1	INIT(7)
1	0	0	0	INIT(8)
1	0	0	1	INIT(9)
1	0	1	0	INIT(10)
1	0	1	1	INIT(11)
1	1	0	0	INIT(12)
1	1	0	1	INIT(13)
1	1	1	0	INIT(14)
1	1	1	1	INIT(15)

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 256-Bit Value	All zeros	Specifies the contents of the ROM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ROM256X1: 256 x 1 Asynchronous Distributed (LUT) ROM
--             Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ROM256X1_inst : ROM256X1
generic map (
   INIT => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
   O => O,   -- ROM output
   A0 => A0, -- ROM address[0]
   A1 => A1, -- ROM address[1]
   A2 => A2, -- ROM address[2]
   A3 => A3, -- ROM address[3]
   A4 => A4, -- ROM address[4]
   A5 => A5, -- ROM address[5]
   A6 => A6,  -- ROM address[6]
   A7 => A7  -- ROM address[7]
);

-- End of ROM256X1_inst instantiation
```

Verilog Instantiation Template

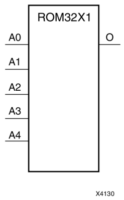
```
// ROM256X1: 256 x 1 Asynchronous Distributed (LUT) ROM
//             Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ROM256X1 #(
   .INIT(256'h0000000000000000000000000000000000000000000000000000000000000000) // Contents of ROM
) ROM256X1_inst (
   .O(O),   // ROM output
   .A0(A0), // ROM address[0]
   .A1(A1), // ROM address[1]
   .A2(A2), // ROM address[2]
   .A3(A3), // ROM address[3]
   .A4(A4), // ROM address[4]
   .A5(A5), // ROM address[5]
   .A6(A6), // ROM address[6]
   .A7(A7)  // ROM address[7]
);

// End of ROM256X1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ROM32X1

Primitive: 32-Deep by 1-Wide ROM



Introduction

This design element is a 32-word by 1-bit read-only memory. The data output (O) reflects the word selected by the 5-bit address (A4:A0). The ROM is initialized to a known value during configuration with the INIT=value parameter. The value consists of eight hexadecimal digits that are written into the ROM from the most-significant digit A=1FH to the least-significant digit A=00H.

For example, the INIT=10A78F39 parameter produces the data stream: 0001 0000 1010 0111 1000 1111 0011 1001. An error occurs if the INIT=value is not specified.

Logic Table

Input				Output
I0	**I1**	**I2**	**I3**	**O**
0	0	0	0	INIT(0)
0	0	0	1	INIT(1)
0	0	1	0	INIT(2)
0	0	1	1	INIT(3)
0	1	0	0	INIT(4)
0	1	0	1	INIT(5)
0	1	1	0	INIT(6)
0	1	1	1	INIT(7)
1	0	0	0	INIT(8)
1	0	0	1	INIT(9)
1	0	1	0	INIT(10)
1	0	1	1	INIT(11)
1	1	0	0	INIT(12)
1	1	0	1	INIT(13)
1	1	1	0	INIT(14)
1	1	1	1	INIT(15)

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 32-Bit Value	All zeros	Specifies the contents of the ROM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ROM32X1: 32 x 1 Asynchronous Distributed (LUT) ROM
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ROM32X1_inst : ROM32X1
generic map (
   INIT => X"00000000")
port map (
   O => O,    -- ROM output
   A0 => A0, -- ROM address[0]
   A1 => A1, -- ROM address[1]
   A2 => A2, -- ROM address[2]
   A3 => A3, -- ROM address[3]
   A4 => A4  -- ROM address[4]
);
-- End of ROM32X1_inst instantiation
```

Verilog Instantiation Template

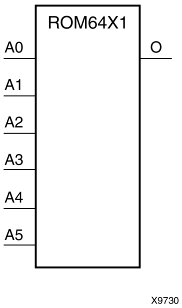
```
// ROM32X1: 32 x 1 Asynchronous Distributed (LUT) ROM
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ROM32X1 #(
   .INIT(32'h00000000) // Contents of ROM
) ROM32X1_inst (
   .O(O),    // ROM output
   .A0(A0), // ROM address[0]
   .A1(A1), // ROM address[1]
   .A2(A2), // ROM address[2]
   .A3(A3), // ROM address[3]
   .A4(A4)  // ROM address[4]
);

// End of ROM32X1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

ROM64X1

Primitive: 64-Deep by 1-Wide ROM



Introduction

This design element is a 64-word by 1-bit read-only memory. The data output (O) reflects the word selected by the 6-bit address (A5:A0). The ROM is initialized to a known value during configuration with the INIT=value parameter. The value consists of 16 hexadecimal digits that are written into the ROM from the most-significant digit A=FH to the least-significant digit A=0H. An error occurs if the INIT=value is not specified.

Logic Table

Input				Output
I0	**I1**	**I2**	**I3**	**O**
0	0	0	0	INIT(0)
0	0	0	1	INIT(1)
0	0	1	0	INIT(2)
0	0	1	1	INIT(3)
0	1	0	0	INIT(4)
0	1	0	1	INIT(5)
0	1	1	0	INIT(6)
0	1	1	1	INIT(7)
1	0	0	0	INIT(8)
1	0	0	1	INIT(9)
1	0	1	0	INIT(10)
1	0	1	1	INIT(11)
1	1	0	0	INIT(12)
1	1	0	1	INIT(13)
1	1	1	0	INIT(14)
1	1	1	1	INIT(15)

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 64-Bit Value	All zeros	Specifies the contents of the ROM.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- ROM64X1: 64 x 1 Asynchronous Distributed (LUT) ROM
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

ROM64X1_inst : ROM64X1
generic map (
   INIT => X"0000000000000000")
port map (
   O => O,   -- ROM output
   A0 => A0, -- ROM address[0]
   A1 => A1, -- ROM address[1]
   A2 => A2, -- ROM address[2]
   A3 => A3, -- ROM address[3]
   A4 => A4, -- ROM address[4]
   A5 => A5  -- ROM address[5]
);

-- End of ROM64X1_inst instantiation
```

Verilog Instantiation Template

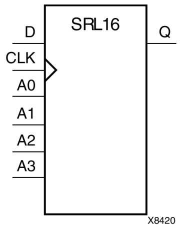
```
// ROM64X1: 64 x 1 Asynchronous Distributed (LUT) ROM
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

ROM64X1 #(
   .INIT(64'h0000000000000000) // Contents of ROM
) ROM64X1_inst (
   .O(O),   // ROM output
   .A0(A0), // ROM address[0]
   .A1(A1), // ROM address[1]
   .A2(A2), // ROM address[2]
   .A3(A3), // ROM address[3]
   .A4(A4), // ROM address[4]
   .A5(A5)  // ROM address[5]
);

// End of ROM64X1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRL16

Primitive: 16-Bit Shift Register Look-Up Table (LUT)



Introduction

This design element is a shift register look-up table (LUT). The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

The data (D) is loaded into the first bit of the shift register during the Low-to-High clock (CLK) transition. During subsequent Low-to-High clock transitions data shifts to the next highest bit position while new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached.

Logic Table

Inputs			Output
Am	**CLK**	**D**	**Q**
Am	X	X	Q(Am)
Am	↑	D	Q(Am - 1)
m= 0, 1, 2, 3			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of Q output after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRL16: 16-bit shift register LUT operating on posedge of clock
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRL16_inst : SRL16
generic map (
   INIT => X"0000")
port map (
   Q => Q,       -- SRL data output
   A0 => A0,     -- Select[0] input
   A1 => A1,     -- Select[1] input
   A2 => A2,     -- Select[2] input
   A3 => A3,     -- Select[3] input
   CLK => CLK,   -- Clock input
   D => D        -- SRL data input
);

-- End of SRL16_inst instantiation
```

Verilog Instantiation Template

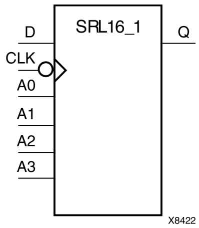
```
// SRL16: 16-bit shift register LUT operating on posedge of clock
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRL16 #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRL16_inst (
   .Q(Q),       // SRL data output
   .A0(A0),     // Select[0] input
   .A1(A1),     // Select[1] input
   .A2(A2),     // Select[2] input
   .A3(A3),     // Select[3] input
   .CLK(CLK),   // Clock input
   .D(D)        // SRL data input
);

// End of SRL16_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRL16_1

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Negative-Edge Clock



Introduction

This design element is a shift register look-up table (LUT). The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

The data (D) is loaded into the first bit of the shift register during the High-to-Low clock (CLK) transition. During subsequent High-to-Low clock transitions data shifts to the next highest bit position as new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached.

Logic Table

Inputs			Output
Am	**CLK**	**D**	**Q**
Am	X	X	Q(Am)
Am	↓	D	Q(Am - 1)
m= 0, 1, 2, 3			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of Q output after configuration

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRL16_1: 16-bit shift register LUT operating on negedge of clock
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRL16_1_inst : SRL16_1
generic map (
   INIT => X"0000")
port map (
   Q => Q,        -- SRL data output
   A0 => A0,      -- Select[0] input
   A1 => A1,      -- Select[1] input
   A2 => A2,      -- Select[2] input
   A3 => A3,      -- Select[3] input
   CLK => CLK,    -- Clock input
   D => D         -- SRL data input
);

-- End of SRL16_1_inst instantiation
```

Verilog Instantiation Template

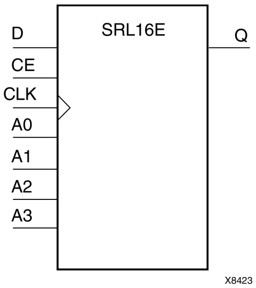
```
// SRL16_1: 16-bit shift register LUT operating on negedge of clock
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRL16_1 #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRL16_1_inst (
   .Q(Q),        // SRL data output
   .A0(A0),      // Select[0] input
   .A1(A1),      // Select[1] input
   .A2(A2),      // Select[2] input
   .A3(A3),      // Select[3] input
   .CLK(CLK),    // Clock input
   .D(D)         // SRL data input
);

// End of SRL16_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRL16E

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Clock Enable



Introduction

This design element is a shift register look-up table (LUT). The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

When CE is High, the data (D) is loaded into the first bit of the shift register during the Low-to-High clock (CLK) transition. During subsequent Low-to-High clock transitions, when CE is High, data shifts to the next highest bit position as new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached. When CE is Low, the register ignores clock transitions.

Logic Table

Inputs				Output
Am	**CE**	**CLK**	**D**	**Q**
Am	0	X	X	Q(Am)
Am	1	↑	D	Q(Am - 1)
m= 0, 1, 2, 3				

Port Descriptions

Port	Direction	Width	Function
Q	Output	1	Shift register data output
D	Input	1	Shift register data input
CLK	Input	1	Clock
CE	Input	1	Active high clock enable
A	Input	4	Dynamic depth selection of the SRL • A=0000 ==> 1-bit shift length • A=1111 ==> 16-bit shift length

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexa- decimal	Any 16-Bit Value	All zeros	Sets the initial value of content and output of shift register after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRL16E: 16-bit shift register LUT with clock enable operating on posedge of clock
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRL16E_inst : SRL16E
generic map (
   INIT => X"0000")
port map (
   Q => Q,        -- SRL data output
   A0 => A0,      -- Select[0] input
   A1 => A1,      -- Select[1] input
   A2 => A2,      -- Select[2] input
   A3 => A3,      -- Select[3] input
   CE => CE,      -- Clock enable input
   CLK => CLK,    -- Clock input
   D => D         -- SRL data input
);

-- End of SRL16E_inst instantiation
```

Verilog Instantiation Template

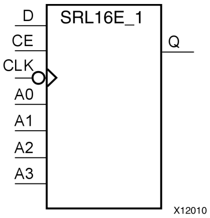
```
// SRL16E: 16-bit shift register LUT with clock enable operating on posedge of clock
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRL16E #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRL16E_inst (
   .Q(Q),       // SRL data output
   .A0(A0),     // Select[0] input
   .A1(A1),     // Select[1] input
   .A2(A2),     // Select[2] input
   .A3(A3),     // Select[3] input
   .CE(CE),     // Clock enable input
   .CLK(CLK),   // Clock input
   .D(D)        // SRL data input
);

// End of SRL16E_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRL16E_1

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Negative-Edge Clock and Clock Enable



Introduction

This design element is a shift register look-up table (LUT) with clock enable (CE). The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

When CE is High, the data (D) is loaded into the first bit of the shift register during the High-to-Low clock (CLK) transition. During subsequent High-to-Low clock transitions, when CE is High, data is shifted to the next highest bit position as new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached. When CE is Low, the register ignores clock transitions.

Logic Table

Inputs				**Output**
Am	**CE**	**CLK**	**D**	**Q**
Am	0	X	X	Q(Am)
Am	1	↓	D	Q(Am - 1)
m= 0, 1, 2, 3				

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of content and output of shift register after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRL16E_1: 16-bit shift register LUT with clock enable operating on negedge of clock
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRL16E_1_inst : SRL16E_1
generic map (
   INIT => X"0000")
port map (
   Q => Q,        -- SRL data output
   A0 => A0,      -- Select[0] input
   A1 => A1,      -- Select[1] input
   A2 => A2,      -- Select[2] input
   A3 => A3,      -- Select[3] input
   CE => CE,      -- Clock enable input
   CLK => CLK,    -- Clock input
   D => D         -- SRL data input
);

-- End of SRL16E_1_inst instantiation
```

Verilog Instantiation Template

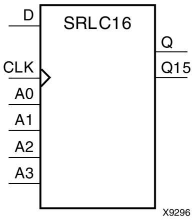
```
// SRL16E_1: 16-bit shift register LUT with clock enable operating on negedge of clock
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRL16E_1 #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRL16E_1_inst (
   .Q(Q),       // SRL data output
   .A0(A0),     // Select[0] input
   .A1(A1),     // Select[1] input
   .A2(A2),     // Select[2] input
   .A3(A3),     // Select[3] input
   .CE(CE),     // Clock enable input
   .CLK(CLK),   // Clock input
   .D(D)        // SRL data input
);

// End of SRL16E_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRLC16

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry



Introduction

This design element is a shift register look-up table (LUT) with Carry. The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

The data (D) is loaded into the first bit of the shift register during the Low-to-High clock (CLK) transition. During subsequent Low-to-High clock transitions data shifts to the next highest bit position as new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached.

Note The Q15 output is available for you in cascading to multiple shift register LUTs to create larger shift registers.

Logic Table

Inputs			Output
Am	**CLK**	**D**	**Q**
Am	X	X	Q(Am)
Am	↑	D	Q(Am - 1)
m= 0, 1, 2, 3			

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of content and output of shift register after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRLC16: 16-bit cascadable shift register LUT operating on posedge of clock
--         Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRLC16_inst : SRLC16
generic map (
   INIT => X"0000")
port map (
   Q => Q,         -- SRL data output
   Q15 => Q15,     -- Carry output (connect to next SRL)
   A0 => A0,       -- Select[0] input
   A1 => A1,       -- Select[1] input
   A2 => A2,       -- Select[2] input
   A3 => A3,       -- Select[3] input
   CLK => CLK,     -- Clock input
   D => D          -- SRL data input
);

-- End of SRLC16_inst instantiation
```

Verilog Instantiation Template

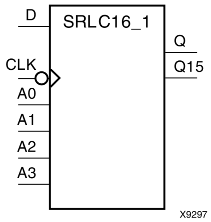
```
// SRLC16: 16-bit cascadable shift register LUT operating on posedge of clock
//         Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRLC16 #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRLC16_inst (
   .Q(Q),         // SRL data output
   .Q15(Q15),     // Carry output (connect to next SRL)
   .A0(A0),       // Select[0] input
   .A1(A1),       // Select[1] input
   .A2(A2),       // Select[2] input
   .A3(A3),       // Select[3] input
   .CLK(CLK),     // Clock input
   .D(D)          // SRL data input
);

// End of SRLC16_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRLC16_1

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry and Negative-Edge Clock



Introduction

This design element is a shift register look-up table (LUT) with carry and a negative-edge clock. The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

Note The Q15 output is available for your use in cascading multiple shift register LUTs to create larger shift registers.

Logic Table

Inputs			Output	
Am	**CLK**	**D**	**Q**	**Q15**
Am	X	X	Q(Am)	No Change
Am	↓	D	Q(Am - 1)	Q14
m= 0, 1, 2, 3				

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of content and output of shift register after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRLC16_1: 16-bit cascadable shift register LUT operating on negedge of clock
--           Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRLC16_1_inst : SRLC16_1
generic map (
   INIT => X"0000")
port map (
   Q => Q,        -- SRL data output
   Q15 => Q15,    -- Carry output (connect to next SRL)
   A0 => A0,      -- Select[0] input
   A1 => A1,      -- Select[1] input
   A2 => A2,      -- Select[2] input
   A3 => A3,      -- Select[3] input
   CLK => CLK,    -- Clock input
   D => D         -- SRL data input
);

-- End of SRLC16_1_inst instantiation
```

Verilog Instantiation Template

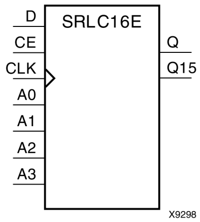
```
// SRLC16_1: 16-bit cascadable shift register LUT operating on negedge of clock
//           Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRLC16_1 #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRLC16_1_inst (
   .Q(Q),        // SRL data output
   .Q15(Q15),    // Carry output (connect to next SRL)
   .A0(A0),      // Select[0] input
   .A1(A1),      // Select[1] input
   .A2(A2),      // Select[2] input
   .A3(A3),      // Select[3] input
   .CLK(CLK),    // Clock input
   .D(D)         // SRL data input
);

// End of SRLC16_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRLC16E

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry and Clock Enable



Introduction

This design element is a shift register look-up table (LUT) with carry and clock enable. The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

The data (D) is loaded into the first bit of the shift register during the Low-to-High clock (CLK) transition. When CE is High, during subsequent Low-to-High clock transitions, data shifts to the next highest bit position as new data is loaded. The data appears on the Q output when the shift register length determined by the address inputs is reached.

Note The Q15 output is available for you in cascading to multiple shift register LUTs to create larger shift registers.

Logic Table

Inputs				Output	
Am	**CLK**	**CE**	**D**	**Q**	**Q15**
Am	X	0	X	Q(Am)	Q(15)
Am	X	1	X	Q(Am)	Q(15)
Am	↑	1	D	Q(Am - 1)	Q15
m= 0, 1, 2, 3					

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of content and output of shift register after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRLC16E: 16-bit cascable shift register LUT with clock enable operating on posedge of clock
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRLC16E_inst : SRLC16E
generic map (
   INIT => X"0000")
port map (
   Q => Q,          -- SRL data output
   Q15 => Q15,      -- Carry output (connect to next SRL)
   A0 => A0,        -- Select[0] input
   A1 => A1,        -- Select[1] input
   A2 => A2,        -- Select[2] input
   A3 => A3,        -- Select[3] input
   CE => CE,        -- Clock enable input
   CLK => CLK,      -- Clock input
   D => D           -- SRL data input
);

-- End of SRLC16E_inst instantiation
```

Verilog Instantiation Template

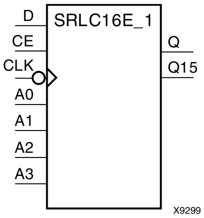
```
// SRLC16E: 16-bit cascadable shift register LUT with clock enable operating on posedge of clock
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRLC16E #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRLC16E_inst (
   .Q(Q),         // SRL data output
   .Q15(Q15),     // Carry output (connect to next SRL)
   .A0(A0),       // Select[0] input
   .A1(A1),       // Select[1] input
   .A2(A2),       // Select[2] input
   .A3(A3),       // Select[3] input
   .CE(CE),       // Clock enable input
   .CLK(CLK),     // Clock input
   .D(D)          // SRL data input
);

// End of SRLC16E_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

SRLC16E_1

Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Carry, Negative-Edge Clock, and Clock Enable



Introduction

This design element is a shift register look-up table (LUT) with carry, clock enable, and negative-edge clock. The inputs A3, A2, A1, and A0 select the output length of the shift register.

The shift register can be of a fixed, static length or it can be dynamically adjusted.

- **To create a fixed-length shift register -**Drive the A3 through A0 inputs with static values. The length of the shift register can vary from 1 bit to 16 bits, as determined by the following formula: Length = (8 x A3) +(4 x A2) + (2 x A1) + A0 +1 If A3, A2, A1, and A0 are all zeros (0000), the shift register is one bit long. If they are all ones (1111), it is 16 bits long.
- **To change the length of the shift register dynamically -**Change the values driving the A3 through A0 inputs. For example, if A2, A1, and A0 are all ones (111) and A3 toggles between a one (1) and a zero (0), the length of the shift register changes from 16 bits to 8 bits. Internally, the length of the shift register is always 16 bits and the input lines A3 through A0 select which of the 16 bits reach the output.

The shift register LUT contents are initialized by assigning a four-digit hexadecimal number to an INIT attribute. The first, or the left-most, hexadecimal digit is the most significant bit. If an INIT value is not specified, it defaults to a value of four zeros (0000) so that the shift register LUT is cleared during configuration.

When CE is High, the data (D) is loaded into the first bit of the shift register during the High-to-Low clock (CLK) transition. During subsequent High-to-Low clock transitions data shifts to the next highest bit position as new data is loaded when CE is High. The data appears on the Q output when the shift register length determined by the address inputs is reached.

Note The Q15 output is available for your use in cascading multiple shift register LUTs to create larger shift registers.

Logic Table

Inputs				Output	
Am	**CE**	**CLK**	**D**	**Q**	**Q15**
Am	0	X	X	Q(Am)	No Change
Am	1	X	X	Q(Am)	No Change
Am	1	↓	D	Q(Am -1)	Q14
m= 0, 1, 2, 3					

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

Available Attributes

Attribute	Data Type	Allowed Values	Default	Description
INIT	Hexadecimal	Any 16-Bit Value	All zeros	Sets the initial value of content and output of shift register after configuration.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- SRLC16E_1: 16-bit shift register LUT with clock enable operating on negedge of clock
--            Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

SRLC16E_1_inst : SRLC16E_1
generic map (
   INIT => X"0000")
port map (
   Q => Q,         -- SRL data output
   Q15 => Q15,     -- Carry output (connect to next SRL)
   A0 => A0,       -- Select[0] input
   A1 => A1,       -- Select[1] input
   A2 => A2,       -- Select[2] input
   A3 => A3,       -- Select[3] input
   CE => CE,       -- Clock enable input
   CLK => CLK,     -- Clock input
   D => D          -- SRL data input
);

-- End of SRLC16E_1_inst instantiation
```

Verilog Instantiation Template

```
// SRLC16E_1: 16-bit shift register LUT with clock enable operating on negedge of clock
//            Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

SRLC16E_1 #(
   .INIT(16'h0000) // Initial Value of Shift Register
) SRLC16E_1_inst (
   .Q(Q),         // SRL data output
   .Q15(Q15),     // Carry output (connect to next SRL)
   .A0(A0),       // Select[0] input
   .A1(A1),       // Select[1] input
   .A2(A2),       // Select[2] input
   .A3(A3),       // Select[3] input
   .CE(CE),       // Clock enable input
   .CLK(CLK),     // Clock input
   .D(D)          // SRL data input
);
```

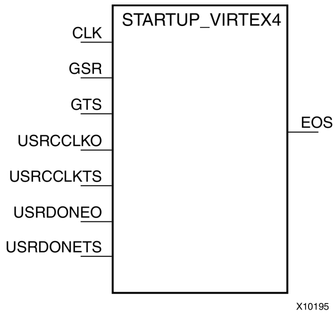
```
// End of SRLC16E_1_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

STARTUP_VIRTEX4

Primitive: Virtex®-4 User Interface to Configuration Clock, Global Reset, Global 3-State Controls, and Other Configuration Signals



Introduction

This design element lets you activate Global Set/Reset (GSR), Global Tristate (GTS) control, and your configuration clock. It also allows you to control the DONE and CLK pins after configuration.

Port Descriptions

Port	Direction	Width	Function
EOS	Output	1	EOS signal
CLK	Input	1	Clock input
GTS	Input	1	Global Tristate (GTS) control
GSR	Input	1	Global Set/Reset (GSR)
USRCCLKO	Input	1	Allows you to drive external CCLK pin.
USRCCLKTS	Input	1	Tristates CCLK pin when asserted
USRDONEO	Input	1	Allows you to drive eternal DONE pin.
USRDONETS	Input	1	Tristates DONE pin when asserted.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

Note Block RAM content, LUT RAMs, the Digital Clock Manager (DCM), and shift register LUTs (SRL16, SRL16_1, SRL16E, SRL16E_1, SRLC16, SRLC16_1, SRLC16E, and SRLC16E_1) are not set/reset.

Following configuration, the Global Tristate (GTS), when High--and when BSCAN, is not enabled and executing an EXTEST instruction--forces all the IOB outputs into high-impedance mode, which isolates the device outputs from the circuit but leaves the inputs active.

CLK input allows you to clock through configuration startup sequence with a user-specified IO, rather than having to provide clock on JTAGs TCK or CCCLK pin. To enable this, Bitgen must also have the startup clk set to userclk when generating your bitstream.

USRCLKO/TS and USRDONEO/TS are used to control the external DONE and CCLK pins. Using the STARTUP_VIRTEX4 in combination with the USR_ACCESS_VIRTEX4 primitive supports a variety of applications, such as loading PROM data into the FPGA for various uses. Refer to USR_ACCESS_VIRTEX4 for more information.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- STARTUP_VIRTEX4: Startup primitive for GSR, GTS or startup sequence
--                  control.
--                  Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

STARTUP_VIRTEX4_inst : STARTUP_VIRTEX4
port map (
   EOS => EOS,       -- End of Startup 1-bit output
   CLK => CLK,       -- Clock input for start-up sequence
   GSR => GSR_PORT, -- Global Set/Reset input (GSR cannot be used for the port name)
   GTS => GTS_PORT, -- Global 3-state input (GTS cannot be used for the port name)
   USRCCLKO => USRCCLKO,   -- USRCCLKO 1-bit input
   USRCCLKTS => USRCCLKTS, -- USRCCLKTS 1-bit input
   USRDONEO => USRDONEO,   -- USRDONEO 1-bit input
   USRDONETS => USRDONETS  -- USRDONETS 1-bit input
);

-- End of STARTUP_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

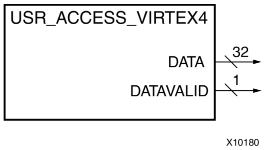
```
// STARTUP_VIRTEX4: Startup primitive for GSR, GTS or startup sequence
//                  control.
//                  Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

STARTUP_VIRTEX4 STARTUP_VIRTEX4_inst (
   .EOS(EOS), // End Of Startup 1-bit output
   .CLK(CLK), // Clock input for start-up sequence
   .GSR(GSR_PORT), // Global Set/Reset input (GSR cannot be used as a port name)
   .GTS(GTS_PORT), // Global 3-state input (GTS cannot be used as a port name)
   .USRCCLKO(USRCCLKO), // USERCLK0 1-bit input
   .USRCCLKTS(USRCCLKTS), // USERCLKTS 1-bit input
   .USRDONEO(USRDONEO), // USRDONE0 1-bit input
   .USRDONETS(USRDONETS) // USRDONETS 1-bit input
);

// End of STARTUP_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

USR_ACCESS_VIRTEX4

Primitive: 32-Bit Register with a 32-Bit DATA Bus and a DATAVALID Port



Introduction

This design element is a 32-bit register that allows data from the bitstream to be directly accessible by the FPGA fabric. This module has two outputs: the 32-bit DATA bus and DATAVALID. The configuration data source clock can be CCLK or TCK.

The use model for this block is that it allows data from a bitstream data storage source (e.g., PROM) to be accessed by the fabric after the FPGA has been configured. To accomplish this the STARTUP_VIRTEX4 block should also be instantiated. The STARTUP_VIRTEX4 block has inputs that allow you to take over the CCLK and DONE pins after the EOS (End-Of-Startup) signal has been asserted. These pins are USR_CCLK_O, USR_CCLK_TS, USR_DONE_O, and USR_DONE_TS. The Bitgen option -g DONE_cycle: 7 should be used to prevent the DONE pin from going high since that would reset the PROM. The USR_CCLK_O pin should be connected to a controlled clock in the fabric. The PROM should contain a packet of data with the USR_ACCESS register as the target. After EOS has been asserted, the data packet can be loaded by clocking the USR_CCLK_O pin while keeping USR_CCLK_TS low (it can be tied low in this usage).

Alternatively, the USR_ACCESS register can be used to provide a single 32-bit constant value to the fabric as an alternative to using a BRAM or LUTRAM to hold the constant.

Port Descriptions

Port	Direction	Width	Function
DATA	Output	32	The 32-bit register that allows the FPGA fabric to access data from bitstream data storage source.
DATAVALID	Output	1	Indicates whether the value in the DATA bus is new or valid. When this condition is true, this port is asserted HIGH for one cycle of the configuration data source clock.

Design Entry Method

Instantiation	Recommended
Inference	No
CORE Generator™ and wizards	No
Macro support	No

When using this module to access data from bitstream data storage source (e.g., PROM) to FPGA fabric after configuration, the STARTUP_VIRTEX4 block should also be instantiated. This element contains inputs that allow the designer to utilize the CCLK and DONE pins after the EOS (End-Of-Startup) signal have been asserted. These pins are USR_CCLK_O, USR_CCLK_TS, USR_DONE_O, and USR_DONE_TS.

The USR_CCLK_O pin should be connected to a controlled clock in the fabric. The data storage source should contain a packet of data with the USR_ACCESS_VIRTEX4 register as the target. After EOS has been asserted, the data packet can be loaded by clocking the USR_CCLK_O pin while keeping USR_CCLK_TS to logic Low. The USR_CCLK_TS can be tied to logic Low when using this application.

In addition, when using this module, the bitgen option -g DONE_cycle: 7 should be used to prevent the High assertion of DONE pin. Should the DONE pin be asserted High, the PROM will be reset.

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- USR_ACCESS_VIRTEX4: Configuration Data Memory Access Port
--                     Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

USR_ACCESS_VIRTEX4_inst : USR_ACCESSS_VIRTEX4
port map (
   DATA => DATA,           -- 32-bit config data output
   DATAVALID => DATAVALID -- 1-bit data valid output
);

-- End of USR_ACCESS_VIRTEX4_inst instantiation
```

Verilog Instantiation Template

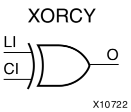
```
// USR_ACCESS_VIRTEX4: Configuration Data Memory Access Port
//                     Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

USR_ACCESS_VIRTEX4 USR_ACCESS_VIRTEX4_inst (
   .DATA(DATA),          // 32-bit config data output
   .DATAVALID(DATAVALID) // 1-bit data valid output
);

// End of USR_ACCESS_VIRTEX4_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

XORCY

Primitive: XOR for Carry Logic with General Output



Introduction

This design element is a special XOR with general O output that generates faster and smaller arithmetic functions. The XORCY primitive is a dedicated XOR function within the carry-chain logic of the slice. It allows for fast and efficient creation of arithmetic (add/subtract) or wide logic functions (large AND/OR gate).

Logic Table

Input		Output
LI	**CI**	**O**
0	0	0
0	1	1
1	0	1
1	1	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- XORCY: Carry-Chain XOR-gate with general output
--        Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

XORCY_inst : XORCY
port map (
   O => O,    -- XOR output signal
   CI => CI, -- Carry input signal
   LI => LI  -- LUT4 input signal
);

-- End of XORCY_inst instantiation
```

Verilog Instantiation Template

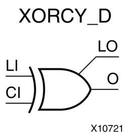
```
// XORCY: Carry-Chain XOR-gate with general output
//        Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

XORCY XORCY_inst (
   .O(O),    // XOR output signal
   .CI(CI), // Carry input signal
   .LI(LI)  // LUT4 input signal
);

// End of XORCY_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

XORCY_D

Primitive: XOR for Carry Logic with Dual Output



Introduction

This design element is a special XOR that generates faster and smaller arithmetic functions.

Logic Table

Input		Output
LI	CI	O and LO
0	0	0
0	1	1
1	0	1
1	1	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- XORCY_D: Carry-Chain XOR-gate with local and general outputs
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

XORCY_D_inst : XORCY_D
port map (
   LO => LO, -- XOR local output signal
   O => O,   -- XOR general output signal
   CI => CI, -- Carry input signal
   LI => LI  -- LUT4 input signal
);

-- End of XORCY_D_inst instantiation
```

Verilog Instantiation Template

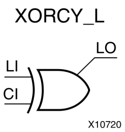
```
// XORCY_D: Carry-Chain XOR-gate with local and general outputs
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

XORCY_D XORCY_D_inst (
   .LO(LO), // XOR local output signal
   .O(O),   // XOR general output signal
   .CI(CI), // Carry input signal
   .LI(LI)  // LUT4 input signal
);

// End of XORCY_D_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.

XORCY_L

Primitive: XOR for Carry Logic with Local Output



Introduction

This design element is a special XOR with local LO output that generates faster and smaller arithmetic functions.

Logic Table

Input		Output
LI	**CI**	**LO**
0	0	0
0	1	1
1	0	1
1	1	0

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	No
Macro support	No

VHDL Instantiation Template

Unless they already exist, copy the following two statements and paste them before the entity declaration.

```
Library UNISIM;
use UNISIM.vcomponents.all;


-- XORCY_L: Carry-Chain XOR-gate with local  => direct-connect output
--          Virtex-4
-- Xilinx HDL Libraries Guide, version 14.7

XORCY_L_inst : XORCY_L
port map (
   LO => LO, -- XOR local output signal
   CI => CI, -- Carry input signal
   LI => LI  -- LUT4 input signal
);

-- End of XORCY_L_inst instantiation
```

Verilog Instantiation Template

```
// XORCY_L: Carry-Chain XOR-gate with local (direct-connect) output
//          Virtex-4
// Xilinx HDL Libraries Guide, version 14.7

XORCY_L XORCY_L_inst (
   .LO(LO), // XOR local output signal
   .CI(CI), // Carry input signal
   .LI(LI)  // LUT4 input signal
);

// End of XORCY_L_inst instantiation
```

For More Information

- See the *Virtex-4 FPGA User Guide (UG070)*.
- See the *Virtex-4 FPGA Data Sheet DC and Switching Characteristics (DS302)*.