

Spartan-3A and Spartan-3A DSP Libraries Guide for HDL Designs

UG613 (v 12.2) July 23, 2010



Xilinx is disclosing this user guide, manual, release note, and/or specification (the “Documentation”) to you solely for use in the development of designs to operate with Xilinx hardware devices. You may not reproduce, distribute, republish, download, display, post, or transmit the Documentation in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx. Xilinx expressly disclaims any liability arising out of your use of the Documentation. Xilinx reserves the right, at its sole discretion, to change the Documentation without notice at any time. Xilinx assumes no obligation to correct any errors contained in the Documentation, or to advise you of any corrections or updates. Xilinx expressly disclaims any liability in connection with technical support or assistance that may be provided to you in connection with the Information.

THE DOCUMENTATION IS DISCLOSED TO YOU “AS-IS” WITH NO WARRANTY OF ANY KIND. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE DOCUMENTATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT OF THIRD-PARTY RIGHTS. IN NO EVENT WILL XILINX BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOSS OF DATA OR LOST PROFITS, ARISING FROM YOUR USE OF THE DOCUMENTATION.

© Copyright 2002-2010 Xilinx Inc. All Rights Reserved. XILINX, the Xilinx logo, the Brand Window and other designated brands included herein are trademarks of Xilinx, Inc. All other trademarks are the property of their respective owners.

About this Guide

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 retargeted elements.
- 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 this architecture, 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.

Table of Contents

Preface About this Guide	3
About Design Elements	3
Design Entry Methods	4
Chapter 1 Design Element Retargeting	9
Chapter 2 Functional Categories	13
Chapter 3 About Design Elements	19
BSCAN_SPARTAN3A.....	20
BUFCF.....	23
BUFG	25
BUFGCE.....	27
BUFGMUX	29
CAPTURE_SPARTAN3A.....	31
DCM_SP	33
DNA_PORT	40
DSP48A.....	42
FDCPE.....	48
FDCPE_1.....	51
FDRSE.....	53
FDRSE_1.....	55
IBUF	57
IBUF_DLY_ADJ.....	59
IBUFDS.....	61
IBUFDS_DLY_ADJ	64
IBUFG	66
IBUFGDS.....	68
ICAP_SPARTAN3A.....	70
IDDR2.....	72
IOBUF.....	75
IOBUFDS.....	78
KEEPER	81
LDCPE	83
LUT1	86
LUT1_D.....	88
LUT1_L	90
LUT2	92

LUT2_D.....	94
LUT2_L	96
LUT3	98
LUT3_D.....	100
LUT3_L	102
LUT4	104
LUT4_D.....	107
LUT4_L	110
MULT_AND	113
MULT18X18SIO	115
MUXCY.....	118
MUXCY_D	120
MUXCY_L.....	122
MUXF5.....	124
MUXF5_D	126
MUXF5_L	128
MUXF6.....	130
MUXF6_D	132
MUXF6_L	134
MUXF7.....	136
MUXF7_D	138
MUXF7_L	140
MUXF8.....	142
MUXF8_D	144
MUXF8_L	146
OBUF	148
OBUFDS	150
OBUFT	152
OBUFTDS.....	154
ODDR2	156
PULLDOWN	159
PULLUP	161
RAM16X1D.....	163
RAM16X1S.....	166
RAM32X1S.....	168
RAM64X1S.....	170
RAMB16BWE	173
RAMB16BWE_S18.....	176

RAMB16BWE_S18_S18.....	179
RAMB16BWE_S18_S9.....	183
RAMB16BWE_S36.....	186
RAMB16BWE_S36_S18.....	189
RAMB16BWE_S36_S36.....	193
RAMB16BWE_S36_S9.....	196
RAMB16BWER.....	199
ROM128X1.....	207
ROM16X1.....	210
ROM256X1.....	212
ROM32X1.....	215
ROM64X1.....	218
SPI_ACCESS	221
SRL16	223
SRL16_1	225
SRL16E.....	227
SRL16E_1	230
SRLC16	232
SRLC16_1.....	234
SRLC16E	236
SRLC16E_1.....	238
STARTUP_SPARTAN3A.....	241
XORCY.....	243
XORCY_D.....	245
XORCY_L.....	247

Design Element Retargeting

To ensure that Xilinx customers are able to take full advantage of the latest circuit design advances, certain design elements are automatically changed by the ISE software tools when they are used in this architecture.

The following table lists these elements and the more advanced elements into which they are transformed.

Original Element	Modern Equivalent
BUFGCE_1	BUFGCE + INV
BUFGDLL	DCM_SP + BUFG
BUFGMUX_1	BUFGMUX + INV
BUFGP	BUFG
CAPTURE_SPARTAN3	CAPTURE_SPARTAN3a
CLKDLL	DCM_SP
CLKDLLE	DCM_SP
CLKDLLHF	DCM_SP
FD	FDCPE
FD_1	FDCPE + INV
FDC	FDCPE
FDC_1	FDCPE + INV
FDCE	FDCPE
FDCE_1	FDCPE + INV
FDCP	FDCPE
FDCP_1	FDCPE + INV
FDE	FDCPE
FDE_1	FDCPE + INV
FDPE	FDCPE
FDPE_1	FDCPE + INV
FDR	FDRSE
FDR_1	FDRSE + INV
FDRE	FDRSE
FDRE_1	FDRSE + INV
FDRS	FDRSE

Original Element	Modern Equivalent
FDRS_1	FDRSE + INV
FDS	FDRSE
FDS_1	FDRSE + INV
FDSE	FDRSE
FDSE_1	FDRSE + INV
LD	LDCPE
LD_1	LDCPE + INV
LDC	LDCPE
LDC_1	LDCPE + INV
LDCE	LDCPE
LDCE_1	LDCPE + INV
LDE	LDCPE
LDE_1	LDCPE + INV
LDP	LDCPE
LDP_1	LDCPE + INV
LDPE	LDCPE
LDPE_1	LDCPE + INV
RAM128X1S_1	RAM128x1s + INV on clock
RAM16X1D_1	RAM16X1D + INV on clock
RAM16X1S_1	RAM16X1S + INV on clock
RAM16X2S	2=RAM16x1s
RAM16X4S	4=RAM16x1s
RAM16X8S	8=RAM16x1s
RAM32X1D_1	RAM32x1d + INV on clock
RAM32X1S_1	RAM32x1s + INV on clock
RAM32X2S	2 RAM32x1s
RAM32X4S	4 RAM32x1s
RAM32X8S	8 RAM32x1s
RAM64X1S_1	RAM64x1s + INV on clock
RAM64X2S	2 RAM64x1s
RAMB16_S1_S1	RAMB16BWE
RAMB16_S1_S18	RAMB16BWE
RAMB16_S1_S2	RAMB16BWE
RAMB16_S1_S36	RAMB16BWE
RAMB16_S1_S4	RAMB16BWE
RAMB16_S1_S9	RAMB16BWE
RAMB16_S1	RAMB16BWE
RAMB16_S18_S18	RAMB16BWE

Original Element	Modern Equivalent
RAMB16_S18_S36	RAMB16BWE
RAMB16_S18	RAMB16BWE
RAMB16_S2_S18	RAMB16BWE
RAMB16_S2_S2	RAMB16BWE
RAMB16_S2_S36	RAMB16BWE
RAMB16_S2_S4	RAMB16BWE
RAMB16_S2_S9	RAMB16BWE
RAMB16_S2	RAMB16BWE
RAMB16_S36_S36	RAMB16BWE
RAMB16_S36	RAMB16BWE
RAMB16_S4_S18	RAMB16BWE
RAMB16_S4_S36	RAMB16BWE
RAMB16_S4_S4	RAMB16BWE
RAMB16_S4_S9	RAMB16BWE
RAMB16_S4	RAMB16BWE
RAMB16_S9_S18	RAMB16BWE
RAMB16_S9_S36	RAMB16BWE
RAMB16_S9_S9	RAMB16BWE
RAMB16_S9	RAMB16BWE
RAMB4_S1_S1	RAMB16BWE
RAMB4_S1_S16	RAMB16BWE
RAMB4_S1_S2	RAMB16BWE
RAMB4_S1_S4	RAMB16BWE
RAMB4_S1_S8	RAMB16BWE
RAMB4_S1	RAMB16BWE
RAMB4_S16_S16	RAMB16BWE
RAMB4_S16	RAMB16BWE
RAMB4_S2_S16	RAMB16BWE
RAMB4_S2_S2	RAMB16BWE
RAMB4_S2_S4	RAMB16BWE
RAMB4_S2_S8	RAMB16BWE
RAMB4_S2	RAMB16BWE
RAMB4_S4_S16	RAMB16BWE
RAMB4_S4_S4	RAMB16BWE
RAMB4_S4_S8	RAMB16BWE
RAMB4_S4	RAMB16BWE
RAMB4_S8_S16	RAMB16BWE
RAMB4_S8_S8	RAMB16BWE

Original Element	Modern Equivalent
STARTUP_SPARTAN3	STARTUP_SPARTAN3a

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.

Arithmetic Functions	RAM/ROM
Clock Components	Registers/Latches
Config/BSCAN Components	Shift Register LUT
I/O Components	Slice/CLB Primitives

Arithmetic Functions

Design Element	Description
DSP48A	Primitive: Multi-Functional, Cascadable, 48-bit Output, Arithmetic Block
MULT18X18SIO	Primitive: 18 x 18 Cascadable Signed Multiplier with Optional Input and Output Registers, Clock Enable, and Synchronous Reset

Clock Components

Design Element	Description
BUFG	Primitive: Global Clock Buffer
BUFGCE	Primitive: Global Clock Buffer with Clock Enable
BUFGMUX	Primitive: Global Clock MUX Buffer
DCM_SP	Primitive: Digital Clock Manager
IBUFG	Primitive: Dedicated Input Clock Buffer
IBUFGDS	Primitive: Differential Signaling Dedicated Input Clock Buffer and Optional Delay

Config/BSCAN Components

Design Element	Description
BSCAN_SPARTAN3A	Primitive: Spartan®-3A JTAG Boundary Scan Logic Access Circuit
CAPTURE_SPARTAN3A	Primitive: Spartan®-3A Register State Capture for Bitstream Readback
DNA_PORT	Primitive: Device DNA Data Access Port
ICAP_SPARTAN3A	Primitive: Internal Configuration Access Port
SPI_ACCESS	Primitive: Internal Logic Access to the Serial Peripheral Interface (SPI) PROM Data
STARTUP_SPARTAN3A	Primitive: Spartan®-3A Global Set/Reset, Global 3-State and Configuration Start-Up Clock Interface

I/O Components

Design Element	Description
IBUF	Primitive: Input Buffer
IBUF_DLY_ADJ	Primitive: Dynamically Adjustable Input Delay Buffer
IBUFDS	Primitive: Differential Signaling Input Buffer
IBUFDS_DLY_ADJ	Primitive: Dynamically Adjustable Differential Input Delay Buffer
IBUFG	Primitive: Dedicated Input Clock Buffer
IBUFGDS	Primitive: Differential Signaling Dedicated Input Clock Buffer and Optional Delay
IOBUF	Primitive: Bi-Directional Buffer
IOBUFDS	Primitive: 3-State Differential Signaling I/O Buffer with Active Low Output Enable
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
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

RAM/ROM

Design Element	Description
RAM16X1D	Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM
RAM16X1S	Primitive: 16-Deep by 1-Wide Static Synchronous RAM
RAM32X1S	Primitive: 32-Deep by 1-Wide Static Synchronous RAM
RAM64X1S	Primitive: 64-Deep by 1-Wide Static Synchronous RAM
RAMB16BWE	Primitive: 16K-bit Data and 2K-bit Parity Configurable Synchronous Dual Port Block RAM
RAMB16BWE_S18	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Single Port Block RAM with 18-bit Port
RAMB16BWE_S18_S18	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 18-bit Ports
RAMB16BWE_S18_S9	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 18-bit and 9-bit Ports
RAMB16BWE_S36	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Single Port Block RAM with 36-Bit Port
RAMB16BWE_S36_S18	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 36-bit and 18-bit Ports
RAMB16BWE_S36_S36	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 36-bit Ports
RAMB16BWE_S36_S9	Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 36-bit and 9-bit Ports
RAMB16BWER	Primitive: 16K-bit Data and 2K-bit Parity Configurable Synchronous Dual Port Block RAM with Optional Output Registers
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
FDCPE	Primitive: D Flip-Flop with Clock Enable and Asynchronous Preset and Clear
FDCPE_1	Primitive: D Flip-Flop with Negative-Edge Clock, Clock Enable, and Asynchronous Preset and Clear
FDRSE	Primitive: D Flip-Flop with Synchronous Reset and Set and Clock Enable
FDRSE_1	Primitive: D Flip-Flop with Negative-Clock Edge, Synchronous Reset and Set, and Clock Enable
IDDR2	Primitive: Double Data Rate Input D Flip-Flop with Optional Data Alignment, Clock Enable and Programmable Synchronous or Asynchronous Set/Reset
LDCPE	Primitive: Transparent Data Latch with Asynchronous Clear and Preset and Gate Enable
ODDR2	Primitive: Dual Data Rate Output D Flip-Flop with Optional Data Alignment, Clock Enable and Programmable Synchronous or Asynchronous Set/Reset

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

Design Element	Description
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
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

About Design Elements

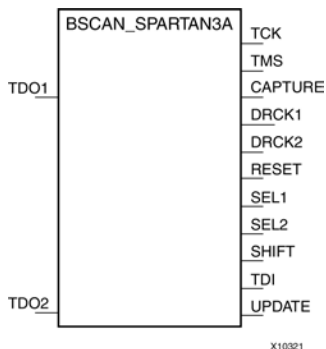
This section describes the design elements that can be used with this architecture. The design elements are organized alphabetically.

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_SPARTAN3A

Primitive: Spartan®-3A 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.

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
TDI	Output	1	A mirror of the TDI input pin to the FPGA.
DRCK1, DRK2	Output	1	A mirror of the TCK input pin to the FPGA when the JTAG USER instruction is loaded and the JTAG TAP controller is in the SHIFT-DR state. DRK1 applies to the USER1 logic while DRK2 applies to USER2.
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.
SEL1, SEL2	Output	1	Indicates when the USER1 or USER2 instruction has been loaded into the JTAG Instruction Register. SEL1 or SEL2 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.
CAPTURE	Output	1	Active upon the loading of the USER instruction. Asserts High when the JTAG TAP controller is in the CAPTURE-DR state.
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.
TCK	Output	1	TCK output from TAP controller
TMS	Output	1	TMS output from TAP controller
TDO1, TDO2	Input	1	Active upon the loading of the USER1 or USER2 instruction. External JTAG TDO pin reflects data input to the component's TDO1 (USER1) or TDO2 (USER2) pin.

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;

-- BSCAN_SPARTAN3A: Boundary Scan primitive for connecting internal logic to
--                   JTAG interface.
--                   Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

BSCAN_SPARTAN3A_inst : BSCAN_SPARTAN3A
port map (
    CAPTURE => CAPTURE, -- CAPTURE output from TAP controller
    DRCK1 => DRCK1,      -- Data register output for USER1 functions
    DRCK2 => DRCK2,      -- Data register output for USER2 functions
    RESET => RESET,      -- Reset output from TAP controller
    SEL1 => SEL1,         -- USER1 active output
    SEL2 => SEL2,         -- USER2 active output
    SHIFT => SHIFT,      -- SHIFT output from TAP controller
    TCK => TCK,          -- TCK output from TAP controller
    TDI => TDI,          -- TDI output from TAP controller
    TMS => TMS,          -- TMS output from TAP controller
    UPDATE => UPDATE,    -- UPDATE output from TAP controller
    TDO1 => TDO1,        -- Data input for USER1 function
    TDO2 => TDO2         -- Data input for USER2 function
);

-- End of BSCAN_SPARTAN3A_inst instantiation
```

Verilog Instantiation Template

```
// BSCAN_SPARTAN3A: Boundary Scan primitive for connecting internal logic to
//                   JTAG interface.
//                   Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

BSCAN_SPARTAN3A BSCAN_SPARTAN3A_inst (
    .CAPTURE(CAPTURE), // CAPTURE output from TAP controller
    .DRCK1(DRCK1),     // Data register output for USER1 functions
    .DRCK2(DRCK2),     // Data register output for USER2 functions
    .RESET(RESET),     // Reset output from TAP controller
    .SEL1(SEL1),       // USER1 active output
    .SEL2(SEL2),       // USER2 active output
    .SHIFT(SHIFT),     // SHIFT output from TAP controller
    .TCK(TCK),         // TCK output from TAP controller
    .TDI(TDI),         // TDI output from TAP controller
    .TMS(TMS),         // TMS output from TAP controller
    .UPDATE(UPDATE),   // UPDATE output from TAP controller
    .TDO1(TDO1),       // Data input for USER1 function
    .TDO2(TDO2)        // Data input for USER2 function
);

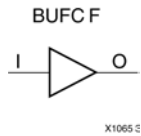
// End of BSCAN_SPARTAN3A_inst instantiation
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

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.
--       Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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.
//       Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

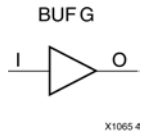
// End of BUFCF_inst instantiation
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

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	Type	Width	Function
I	Input	1	Clock buffer output
O	Output	1	Clock buffer 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;

-- BUFG: Global Clock Buffer (source by an internal signal)
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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)
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

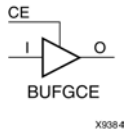
// End of BUFG_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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

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)
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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)
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

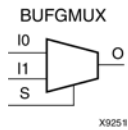
// End of BUFGCE_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

BUFGMUX

Primitive: Global Clock MUX Buffer



Introduction

BUFGMUX is a multiplexed global clock buffer that can select between two input clocks: I0 and I1. When the select input (S) is Low, the signal on I0 is selected for output (O). When the select input (S) is High, the signal on I1 is selected for output.

BUFGMUX and BUFGMUX_1 are distinguished by the state the output assumes when that output switches between clocks in response to a change in its select input. BUFGMUX assumes output state 0 and BUFGMUX_1 assumes output state 1.

Note BUFGMUX guarantees that when S is toggled, the state of the output remains in the inactive state until the next active clock edge (either I0 or I1) occurs.

Logic Table

Inputs			Outputs
I0	I1	S	O
I0	X	0	I0
X	I1	1	I1
X	X	↑	0
X	X	↓	0

Port Descriptions

Port	Type	Width	Function
I0	Input	1	Clock0 input
I1	Input	1	Clock1 input
O	Output	1	Clock MUX output
S	Input	1	Clock select 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;

-- BUFGMUX: Global Clock Buffer 2-to-1 MUX
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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

-- End of BUFGMUX_inst instantiation
```

Verilog Instantiation Template

```
// BUFGMUX: Global Clock Buffer 2-to-1 MUX
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

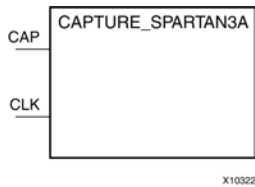
// End of BUFGMUX_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

CAPTURE_SPARTAN3A

Primitive: Spartan®-3A Register State Capture for Bitstream Readback



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	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_SPARTAN3A: Register State Capture for Bitstream Readback
--                      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

CAPTURE_SPARTAN3A_inst : CAPTURE_SPARTAN3A
generic map (
    ONESHOT => TRUE) -- TRUE or FALSE
port map (
    CAP => CAP,      -- Capture input
    CLK => CLK       -- Clock input
);

-- End of CAPTURE_SPARTAN3A_inst instantiation
```

Verilog Instantiation Template

```
// CAPTURE_SPARTAN3A: Register State Capture for Bitstream Readback
//                      Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

CAPTURE_SPARTAN3A #(
    .ONESHOT("TRUE") // "TRUE" or "FALSE"
) CAPTURE_SPARTAN3A_inst (
    .CAP(CAP),        // Capture input
    .CLK(CLK)         // Clock input
);

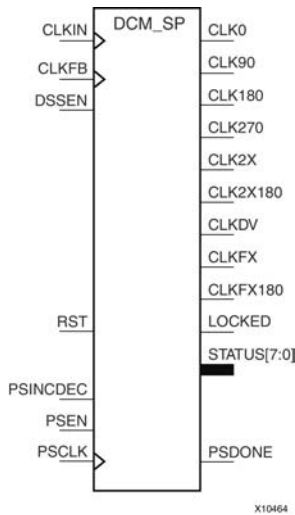
// End of CAPTURE_SPARTAN3A_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

DCM_SP

Primitive: Digital Clock Manager



Introduction

This design element is a digital clock manager that provides multiple functions. It can implement a clock delay locked loop (DLL), a digital frequency synthesizer (DFS), and a digital phase shifter (DPS). DCM_SPs are useful for eliminating the clock delay coming on and off the chip, shifting the clock phase to improve data capture, deriving different frequency clocks, as well as other useful clocking functions.

Port Descriptions

Port	Type	Width	Function
CLKDV	Output	1	Divided clock output, controlled by the CLKDV_DIVIDE attribute. The CLKDV output has a 50% duty cycle unless the CLKDV_DIVIDE attribute is a non-integer value.
CLKFB	Input	1	Clock feedback input to DCM. The feedback input is required unless the DFS outputs, CLKFX or CLKFX180, are used standalone. The source of the CLKFB input must be the CLK0 or CLK2X output from the DCM and the CLK_FEEDBACK must be set to 1X or 2X accordingly. When set to NONE, CLKFB is unused and should be tied low. Ideally, the feedback point includes the delay added by the clock distribution network, either internally or externally.
CLKFX	Output	1	Synthesized clock output, controlled by the CLKFX_MULTIPLY and CLKFX_DIVIDE attributes. Always has a 50% duty cycle. If no phase relationship is necessary, then no clock feedback is required.
CLKFX180	Output	1	Synthesized clock output CLKFX, 180 phase shift (appears to be an inverted version of CLKFX). Always has a 50% duty cycle. If no phase relationship is necessary, then no feedback loop is required.
CLKIN	Input	1	Clock input to DCM. Always required. The CLKIN frequency and jitter must fall within the limits specified in the data sheet.
CLK0	Output	1	Same frequency as CLKIN, 0 phase shift (i.e., not phase shifted). Always conditioned to a 50% duty cycle on Spartan®-6 FPGAs. CLK_FEEDBACK must be set to 1X or 2X to deskew CLK0.
CLK2X	Output	1	Double-frequency clock output, 0 phase shift. When available, the CLK2X output always has a 50% duty cycle. Either CLK0 or CLK2X is required as a feedback source for DLL functions.

Port	Type	Width	Function
CLK2X180	Output	1	Double-frequency clock output, 180 phase shift. When available, the CLK2X180 output always has a 50% duty cycle.
CLK90	Output	1	Same frequency as CLKIN, 90 phase shift (quarter period). Always conditioned to a 50% duty cycle on Spartan®-6 FPGAs.
CLK180	Output	1	Same frequency as CLKIN, 180 phase shift (half period). Always conditioned to a 50% duty cycle on Spartan®-6 FPGAs.
CLK270	Output	1	Same frequency as CLKIN, 270 phase shift (three-quarters period). Always conditioned to a 50% duty cycle on Spartan®-6 FPGAs.
LOCKED	Output	1	<p>All DCM features have locked onto the CLKIN frequency. Clock outputs are now valid, assuming CLKIN is within specified limits.</p> <ul style="list-style-type: none"> 0 - DCM is attempting to lock onto CLKIN frequency. DCM clock outputs are not valid. 1 - DCM is locked onto CLKIN frequency. DCM clock outputs are valid. 1-to-0 - DCM lost lock. Reset DCM.
PSCLK	Input	1	Clock input to variable phase shifter, clocked on rising edge. When using a global clock buffer, only the upper eight BUFGMUXs can drive PSCLK: BUFGMUX_X2Y1, BUFGMUX_X2Y2, BUFGMUX_X2Y3, BUFGMUX_X2Y4, BUFGMUX_X3Y5, BUFGMUX_X3Y6, BUFGMUX_X3Y7 and BUFGMUX_X3Y8.
PSDONE	Output	1	<p>Variable phase shift operation complete.</p> <ul style="list-style-type: none"> 0 - No phase shift operation is active or phase shift operation is in progress. 1 - Requested phase shift operation is complete. Output High for one PSCLK cycle. Next variable phase shift operation can commence.
PSEN	Input	1	<p>Variable phase-shift enable. Can be inverted within a DCM block. Non-inverted behavior shown below.</p> <ul style="list-style-type: none"> 0 - Disable variable phase shift. Ignore inputs to phase shifter. 1 - Enable variable phase shift operations on next rising PSCLK clock edge.
PSINCDEC	Input	1	<p>Increment/decrement variable phase shift. Can be inverted within a DCM block. Non-inverted behavior shown below.</p> <ul style="list-style-type: none"> 0 - Decrement phase shift value on next enabled, rising PSCLK clock edge. 1 - Increment phase shift value on next enabled, rising PSCLK clock edge.
RST	Input	1	<p>Asynchronous reset input. Resets the DCM logic to its postconfiguration state. Causes DCM to reacquire and relock to the CLKIN input. Invertible within DCM block. Non-inverted behavior shown below.</p> <ul style="list-style-type: none"> 0 - No effect. 1 - Reset DCM block. Hold RST pulse High for at least three valid CLKIN cycles.

Port	Type	Width	Function
STATUS[7:0]	Output	8	<p>The status output bus provides DCM status.</p> <ul style="list-style-type: none"> STATUS[0] - Variable phase shift overflow. Control output for variable fine phase shifting. The variable phase shifter has reached a minimum or maximum limit value. The limit value is either +/-255 or a lesser value if the phase shift has reached the end of the delay line. <ul style="list-style-type: none"> 0 - The phase shift has not yet reached its limit value. 1 - The phase shift has reached its limited value. STATUS[1] - CLKIN Input Stopped Indicator. Available only when the CLKFB feedback input is connected. Held in reset until the LOCKED output is asserted. Requires at least one CLKIN cycle to become active. Never asserted if CLKIN never toggles. <ul style="list-style-type: none"> 0 - CLKIN input is toggling. 1 - CLKIN input is not toggling even though the locked output can still be High. STATUS[2] - CLKFX or CLKFX180 output stopped indicator. <ul style="list-style-type: none"> 0 - CLKFX and CLKFX180 outputs are toggling. 1 - CLKFX and CLKFX180 outputs are not toggling, even though the LOCKED output can still be High. STATUS[7:3] - Reserved.

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed_Values	Default	Description
CLK_FEEDBACK	String	"1X", "2X", "NONE"	"1X"	<p>Defines the DCM feedback mode.</p> <ul style="list-style-type: none"> 1X: CLK0 as feedback. 2X: CLK2X as feedback.
CLKDV_DIVIDE	1 significant digit Float	2.0, 1.5, 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_SP 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	Enables CLKIN divide by two features.

Attribute	Type	Allowed_Values	Default	Description
CLKIN_PERIOD	String	0 bit String	"10.0"	Specifies the input period to the DCM_SP CLKIN input in ns.
CLKOUT_PHASE_SHIFT	String	"NONE", "FIXED", "VARIABLE"	"NONE"	<p>This attribute specifies the phase shift mode.</p> <ul style="list-style-type: none"> NONE: No phase shift capability. Any set value has no effect. FIXED: DCM outputs are a fixed phase shift from CLKIN. Value is specified by PHASE_SHIFT attribute. VARIABLE: Allows the DCM outputs to be shifted in a positive and negative range relative to CLKIN. Starting value is specified by PHASE_SHIFT.
DESKEW_ADJUST	String	"SYSTEM_SYNCHRONOUS", "SOURCE_SYNCHRONOUS"	"SYSTEM_SYNCHRONOUS"	Sets configuration bits affecting the clock delay alignment between the DCM_SP output clocks and an FPGA clock input pin.
DFS_FREQUENCY_MODE	String	"LOW", "HIGH"	"LOW"	This is a legacy attribute. The DCM is always in the automatic frequency search mode. Setting High or Low makes no effect.
DLL_FREQUENCY_MODE	String	"LOW", "HIGH"	"LOW"	This is a legacy attribute. The DCM is always in the automatic frequency search mode. Setting High or Low makes no effect.
DSS_MODE	String	"NONE", "SPREAD_2", "SPREAD_4", "SPREAD_6", "SPREAD_8"	"NONE"	<p>Note Unused in Spartan®-6.</p> <p>Specifies a frequency spread for output clocks.</p> <ul style="list-style-type: none"> NONE - The default, specifies no spread factors. The digital spread spectrum function is disabled. SPREAD_2 - Creates a new clock period that is +/- 50 ps of the current clock period SPREAD_4 - Creates a new clock period that is +/- 100 ps of the current clock period. SPREAD_6 - Creates a new clock period that is +/- 150 ps of the current clock period. SPREAD_8 - Creates a new clock period that is +/- 200 ps of the current clock period. <p>The spreading is cumulative as the SPREAD_# is increased. For example, SPREAD_2 creates two additional clock frequencies at +/-50 ps relative to the input clock frequency; SPREAD_4 does the same as SPREAD_2, plus it creates two additional clock frequencies at +/-100 ps.</p>
DUTY_CYCLE_CORRECTION	Boolean	TRUE, FALSE	TRUE	Unsupported

Attribute	Type	Allowed_Values	Default	Description
FACTORY_JF	Hexadecimal	16'h8080 to 16'hffff	16'hc080	Unsupported
PHASE_SHIFT	Integer	-255 to 255	0	<p>The PHASE_SHIFT attribute is applicable only if the CLKOUT_PHASE_SHIFT attribute is set to FIXED or VARIABLE. Defines the rising-edge skew between CLKIN and all the DCM clock outputs at configuration and consequently phase shifts the DCM clock outputs. The skew or phase shift value is specified as an integer that represents a fraction of the clock period as expressed in the equations in Fine Phase Shifting. Actual allowable values depends on input clock frequency. The actual range is less when $TCLKIN > FINE_SHIFT_RANGE$. The FINE_SHIFT_RANGE specification represents the total delay of all taps in the delay line.</p>
STARTUP_WAIT	Boolean	FALSE, TRUE	FALSE	<p>Controls whether the FPGA configuration signal DONE waits for the DCM to assert its LOCKED signal before going High.</p> <ul style="list-style-type: none"> FALSE - Default. DONE is asserted at the end of configuration without waiting for the DCM to assert LOCKED. TRUE - The DONE signal does not transition High until the LOCKED signal transitions High on the associated DCM. <p>STARTUP_WAIT does not prevent LOCKED from transitioning High. The FPGA startup sequence must also be modified to insert a LCK (lock) cycle before the postponed cycle. The DONE cycle or the GWE cycle are typical choices. When more than one DCM is configured, the FPGA waits until all DCMs are LOCKED.</p>

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_SP: Digital Clock Manager Circuit
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

DCM_SP_inst : DCM_SP
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 interger from 1 to 32
    CLKFX_MULTIPLY => 4, -- Can be any integer from 1 to 32
    CLKIN_DIVIDE_BY_2 => FALSE, -- TRUE/FALSE to enable CLKIN divide by two feature
    CLKIN_PERIOD => 0.0, -- Specify period of input clock
    CLKOUT_PHASE_SHIFT => "NONE", -- Specify phase shift of "NONE", "FIXED" or "VARIABLE"
    CLK_FEEDBACK => "1X", -- Specify clock feedback of "NONE", "1X" or "2X"
    DESKEW_ADJUST => "SYSTEM_SYNCHRONOUS", -- "SOURCE_SYNCHRONOUS", "SYSTEM_SYNCHRONOUS" or
                        -- an integer from 0 to 15
    DLL_FREQUENCY_MODE => "LOW", -- "HIGH" or "LOW" frequency mode for DLL
    DUTY_CYCLE_CORRECTION => TRUE, -- Duty cycle correction, TRUE or FALSE
    PHASE_SHIFT => 0, -- Amount of fixed phase shift from -255 to 255
    STARTUP_WAIT => FALSE) -- Delay configuration DONE until DCM_SP 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
    PSDONE => PSDONE, -- Dynamic phase adjust done output
    STATUS => STATUS, -- 8-bit DCM status bits 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_SP_inst instantiation

```

Verilog Instantiation Template

```
// DCM_SP: Digital Clock Manager Circuit
//      Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

DCM_SP #(
    .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(0.0), // Specify period of input clock
    .CLKOUT_PHASE_SHIFT("NONE"), // Specify phase shift of NONE, FIXED or VARIABLE
    .CLK_FEEDBACK("1X"), // Specify clock feedback of NONE, 1X or 2X
    .DESKEW_ADJUST("SYSTEM_SYNCHRONOUS"), // SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                          // an integer from 0 to 15
    .DLL_FREQUENCY_MODE("LOW"), // HIGH or LOW frequency mode for DLL
    .DUTY_CYCLE_CORRECTION("TRUE"), // Duty cycle correction, TRUE or FALSE
    .PHASE_SHIFT(0), // Amount of fixed phase shift from -255 to 255
    .STARTUP_WAIT("FALSE") // Delay configuration DONE until DCM LOCK, TRUE/FALSE
) DCM_SP_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
    .PSDONE(PSDONE), // Dynamic phase adjust done output
    .STATUS(STATUS), // 8-bit DCM status bits 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
);

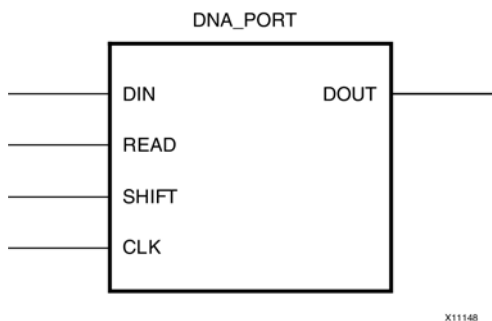
// End of DCM_SP_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

DNA_PORT

Primitive: Device DNA Data Access Port



Introduction

This element allows access to a dedicated shift register that can be loaded with the Device DNA data bits (unique ID) for a given Virtex®-6 device. In addition to shifting out the DNA data bits, this component allows for the inclusion of supplemental bits of your data, or allows for the DNA data to rollover (repeat DNA data after initial data has been shifted out). This component is primarily used in conjunction with other circuitry to build added copy protection for the FPGA bitstream from possible theft. Connect all inputs and outputs to the design to ensure proper operation. To access the Device DNA data, you must first load the shift register by setting the active high READ signal for one clock cycle. After the shift register is loaded, the data can be synchronously shifted out by enabling the active high SHIFT input and capturing the data out the DOUT output port. Additional data can be appended to the end of the 57-bit shift register by connecting the appropriate logic to the DIN port. If DNA data rollover is desired, connect the DOUT port directly to the DIN port to allow for the same data to be shifted out after completing the 57-bit shift operation. If no additional data is necessary, the DIN port can be tied to a logic zero. The attribute SIM_DNA_VALUE can be optionally set to allow for simulation of a possible DNA data sequence. By default, the Device DNA data bits are all zeros in the simulation model.

Port Descriptions

Port	Type	Width	Function
CLK	Input	1	Clock input.
DIN	Input	1	User data input pin.
DOUT	Output	1	DNA output data.
READ	Input	1	Active high load DNA, active low read input.
SHIFT	Input	1	Active high shift enable input.

Design Entry Method

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

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

To access the Device DNA data, you must first load the shift register by setting the active high READ signal for one clock cycle. After the shift register is loaded, the data can be synchronously shifted out by enabling the active high SHIFT input and capturing the data out the DOUT output port. Additional data can be appended to the end of the 57-bit shift register by connecting the appropriate logic to the DIN port. If DNA data rollover is desired, connect the DOUT port directly to the DIN port to allow for the same data to be shifted out after completing the 57-bit shift operation. If no additional data is necessary, the DIN port can be tied to a logic zero. The attribute SIM_DNA_VALUE can be optionally set to allow for simulation of a possible DNA data sequence. By default, the Device DNA data bits are all zeros in the simulation model.

Available Attributes

Attribute	Type	Allowed Values	Default	Description
SIM_DNA_VALUE	Hexa-decimal	57'h00000000 0000000 to 57'h1fffffffffffff	57'h00000000 0000000	Specifies the Pre-programmed factory ID value.

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;

-- DNA_PORT: Access to the device-specific DNA value
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

DNA_PORT_inst : DNA_PORT
generic map (
    SIM_DNA_VALUE => X"0000000000000000") -- Specifies the Pre-programmed factory ID value
port map (
    DOUT => DOUT,      -- 1-bit DNA output data
    CLK => CLK,        -- 1-bit clock input
    DIN => DIN,        -- 1-bit user data input pin
    READ => READ,      -- 1-bit input, active high load DNA, active low read
    SHIFT => SHIFT     -- 1-bit input, active high shift enable
);

-- End of DNA_PORT_inst instantiation
```

Verilog Instantiation Template

```
// DNA_PORT: Access to the device-specific DNA value
//           Spartan-3A, Virtex-6, Spartan-6
// Xilinx HDL Libraries Guide, version 12.2

DNA_PORT #(
    .SIM_DNA_VALUE(57'h0000000000000000) // Specifies the unique DNA value
                                           // for simulation test
) DNA_PORT_inst (

    .DOUT(DOUT), // 1-bit DNA output data
    .CLK(CLK),   // 1-bit clock input
    .DIN(DIN),   // 1-bit user data input pin
    .READ(READ), // 1-bit input, active high load DNA, active low read
    .SHIFT(SHIFT) // 1-bit input, active high shift enable
);

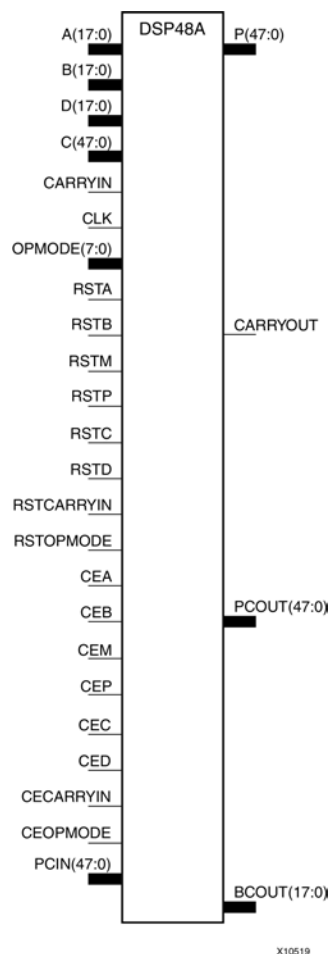
// End of DNA_PORT_inst instantiation
```

For More Information

See the [Spartan-3A FPGA Family Data Sheet](#).

DSP48A

Primitive: Multi-Functional, Cascadable, 48-bit Output, Arithmetic Block



Introduction

The DSP48A is a versatile, scalable, hard IP block that allows for the creation of compact, high-speed, arithmetic-intensive operations, such as those seen for many DSP algorithms. The block consists of a configurable, 18-bit, pre-add/sub, followed by an 18x18 signed multiplier, followed by a 48-bit post-add/sub/accum. Several configurable pipeline registers exist within the block, allowing for higher clock speeds with the trade-off of added latency. Opmode pins allow the block operation to change from one clock-cycle to the next, thus allowing a single block to serve several arithmetic functions within a design. Furthermore, multiple MSP1 blocks can be cascaded to efficiently form larger multiplication and addition functions.

Port Descriptions

Port	Direction	Width	Function
Data Ports			
A	Input	18	18-bit data input to multiplier or post add/sub depending on the value of OPMODE[1:0].
B	Input	18	18-bit data input to multiplier, pre-add/sub and, perhaps, a post-add/sub depending on the value of OPMODE[3:0].
C	Input	48	48-bit data input to post-add/sub.

Port	Direction	Width	Function
D	Input	18	18-bit data input to pre-add/sub.
CARRYIN	Input	1	External carry input to the post-add/sub. Should only be connected to the CARRYOUT pin of another DSP48A block.
P	Output	48	Primary data output.
CARRYOUT	Output	1	Carry out signal for post-add/sub. Should only be connected to the CARRYIN pin of another DSP48A.
Control Inputs			
CLK	Input	1	DSP48A clock
OPMODE	Input	8	Control input to select the arithmetic operations of the DSP48A.
OPMODE[1:0]			Specifies the source of the X input to the post-add/sub <ul style="list-style-type: none"> 0 - Specifies to place all zeroes (disable the post-add/sub). 1 - Use the POUT output signal. 2 - Use the concatenated D, B, A input signals. 3 - Use the multiplier product.
OPMODE[3:2]			Specifies the source of the Y input to the post-add/sub <ul style="list-style-type: none"> 0 - Specifies to place all zeroes (disable the post-add/sub and propagate the multiplier product to POUT). 1 - Use the PCIN. 2 - Use the POUT port (accumulator). 3 - Use the C port.
OPMODE[4]			Specifies the use of the pre-add/sub <ul style="list-style-type: none"> 0 - Selects to use the pre-adder adding or subtracting the values on the B and D ports prior to the multiplier. 1 - Bypass the pre-adder, supplying the data on Port B directly to the multiplier.
OPMODE[5]			Force a value on carry-in to the post-adder. Only applicable when CARRYINSEL = "OPMODE5".
OPMODE[6]			Specifies whether the pre-add/sub is an adder or subtracter <ul style="list-style-type: none"> 0 - Specifies pre-add/sub to perform an addition operation. 1 - Specifies pre-add/sub to perform a subtract operation.
OPMODE[7]			Specifies whether the post-add/sub is an adder or subtracter <ul style="list-style-type: none"> 0 - Specifies post-add/sub to perform an addition operation. 1 - Specifies post-add/sub to perform a subtract operation.
Reset/Clock Enable Inputs			
RSTA	Input	1	Active high, reset for the A port registers (A0REG = 1 or A1REG = 1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous, depending on the value of the RSTTYPE attribute.
RSTB	Input	1	Active high, reset for the B port registers (B0REG = 1 or B1REG = 1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.

Port	Direction	Width	Function
RSTC	Input	1	Active high, reset for the C port registers (CREG=1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.
RSTD	Input	1	Active high, reset for the D port registers (DREG=1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.
RSTM	Input	1	Active high, reset for the multiplier registers (MREG=1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.
RSTP	Input	1	Active high, reset for the P output registers (PREG=1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.
RSTCARRYIN	Input	1	Active high, reset for the carry-in register (CARRYINREG=1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.
RSTOPMODE	Input	1	Active High, reset for the OPMODE registers (OPMODEREG=1). Tie to logic zero if not used. This reset is configurable to be synchronous or asynchronous depending on the value of the RSTTYPE attribute.
CEA	Input	1	Active High, clock enable for the A port registers (A0REG = 1 or A1REG = 1). Tie to logic one if not used and A0REG = 1 or A1REG = 1. Tie to logic zero if A0REG = 0, and A1REG = 0.
CEB	Input	1	Active High, clock enable for the B port registers (B0REG = 1 or B1REG = 1). Tie to logic one if not used and B0REG = 1 or B1REG = 1. Tie to logic zero if B0REG = 0 and B1REG = 0.
CEC	Input	1	Active high, clock enable for the C port registers (CREG=1). Tie to logic one if not used and CREG=1. Tie to a logic zero if CREG=0..
CED	Input	1	Active high, clock enable for the D port registers (DREG=1). Tie to logic one if not used and DREG=1. Tie to a logic zero if DREG=0.
CEM	Input	1	Active high, clock enable for the multiplier registers (MREG=1). Tie to logic one if not used and MREG=1. Tie to a logic zero if MREG=0.
CEP	Input	1	Active high, clock enable for the output port registers (PREG=1). Tie to logic one if not used and PREG=1. Tie to a logic zero if PREG=0.
CECARRYIN	Input	1	Active high, clock enable for the carry-in registers (CARRYINREG=1). Tie to logic one if not used and CARRYINREG=1. Tie to a logic zero if CARRYINREG=0.
CEOPMODE	Input	1	Clock enable for the OPMODE input registers (OPMODEREG=1). Tie to logic one if not used and OPMODEREG=1. Tie to a logic zero if OPMODEREG=0.
Cascade Ports			
PCIN	Input	48	Cascade input for Port P. If used, connect to PCOUT of upstream cascaded DSP48A. If not used, tie port to all zeros.
PCOUT	Output	48	Cascade output for Port P. If used, connect to PCIN of downstream cascaded DSP48A. If not used, leave unconnected.
BCOUT	Output	18	Cascade output for Port B. If used, connect to the B port of downstream cascaded DSP48A. If not used, leave unconnected.

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed Values	Default	Description
A0REG	Integer	0, 1	0	Selects whether to register the first stage A input to the DSP48A.
A1REG	Integer	0, 1	1	Selects whether to register the second stage A input to the DSP48A.
B0REG	Integer	0, 1	0	Selects whether to register the first stage B input to the DSP48A.
B1REG	Integer	0, 1	1	Selects whether to register the second stage B input to the DSP48A.
CARRYINREG	Integer	0, 1	1	Selects whether to register the CARRYIN input to the DSP48A. This should only be used when CARRYINSEL is set to "CARRYIN" and the CARRYIN pin is used.
CARRYINSEL	String	"CARRYIN", "OPMODE5"	"CARRYIN"	Selects whether the post add/sub carry-in signal should be sourced from the CARRYIN pin (connected to the CARRYOUT of another DSP48A) or dynamically controlled from the FPGA fabric by the OPMODE[5] input.
CREG	Integer	0, 1	1	Selects whether to register the C input to the DSP48A.
DREG	Integer	0, 1	1	Selects whether to register the D input to the DSP48A.
MREG	Integer	0, 1	1	Selects whether to register the multiplier stage of the DSP48A. Enable=1/disable=0.
OPMODEREG	Integer	0, 1	1	Selects whether to register the OPMODE inputs to the DSP48A.
PREG	Integer	0, 1	1	Selects whether to register the C input to the DSP48A.
RSTTYPE	String	"ASYNC", "SYNC"	"SYNC"	Selects whether all resets for the DSP48A should have a synchronous or asynchronous reset capability. Due to improved timing and circuit stability, it is recommended to always have this set to "SYNC" unless an asynchronous reset is absolutely necessary.

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;

-- DSP48A: DSP Function Block
--      Spartan-3A DSP
-- Xilinx HDL Libraries Guide, version 12.2

DSP48A_inst : DSP48A
generic map (
  AOREG => 1,          -- Enable=1/disable=0 first stage A input pipeline register
  A1REG => 1,          -- Enable=1/disable=0 second stage A input pipeline register
  BOREG => 1,          -- Enable=1/disable=0 first stage B input pipeline register
  B1REG => 1,          -- Enable=1/disable=0 second stage B input pipeline register
  CARRYINREG => 1,     -- Enable=1/disable=0 CARRYIN input pipeline register
  CARRYINSEL => "CARRYIN", -- Specify carry-in source, "CARRYIN" or "OPMODE5"
  CREG => 1,          -- Enable=1/disable=0 C input pipeline register
  DREG => 1,          -- Enable=1/disable=0 D pre-adder input pipeline register
  MREG => 1,          -- Enable=1/disable=0 M pipeline register
  OPMODEREG => 1,     -- Enable=1/disable=0 OPMODE input pipeline register
  PREG => 1,          -- Enable=1/disable=0 P output pipeline register
  RSTTYPE => "SYNC") -- Specify reset type, "SYNC" or "ASYN"
port map (
  BCOUT => BCOUT, -- 18-bit B port cascade output
  CARRYOUT => CARRYOUT, -- 1-bit carry output
  P => P,          -- 48-bit output
  PCOUT => PCOUT, -- 48-bit cascade output
  A => A,          -- 18-bit A data input
  B => B,          -- 18-bit B data input (can be connected to fabric or BCOUT of adjacent DSP48A)
  C => C,          -- 48-bit C data input
  CARRYIN => CARRYIN, -- 1-bit carry input signal
  CEA => CEA,      -- 1-bit active high clock enable input for A input registers
  CEB => CEB,      -- 1-bit active high clock enable input for B input registers
  CEC => CEC,      -- 1-bit active high clock enable input for C input registers
  CECARRYIN => CECARRYIN, -- 1-bit active high clock enable input for CARRYIN registers
  CED => CED,      -- 1-bit active high clock enable input for D input registers
  CEM => CEM,      -- 1-bit active high clock enable input for multiplier registers
  CEOPMODE => CEOPMODE, -- 1-bit active high clock enable input for OPMODE registers
  CEP => CEP,      -- 1-bit active high clock enable input for P output registers
  CLK => CLK,      -- Clock input
  D => D,          -- 18-bit B pre-adder data input
  OPMODE => OPMODE, -- 8-bit operation mode input
  PCIN => PCIN,    -- 48-bit P cascade input
  RSTA => RSTA,     -- 1-bit reset input for A input pipeline registers
  RSTB => RSTB,     -- 1-bit reset input for B input pipeline registers
  RSTC => RSTC,     -- 1-bit reset input for C input pipeline registers
  RSTCARRYIN => RSTCARRYIN, -- 1-bit reset input for CARRYIN input pipeline registers
  RSTD => RSTD,     -- 1-bit reset input for D input pipeline registers
  RSTM => RSTM,     -- 1-bit reset input for M pipeline registers
  RSTOPMODE => RSTOPMODE, -- 1-bit reset input for OPMODE input pipeline registers
  RSTP => RSTP,     -- 1-bit reset input for P pipeline registers
);

-- End of DSP48A_inst instantiation

```

Verilog Instantiation Template

```
// DSP48A: DSP Function Block
//      Spartan-3A DSP
// Xilinx HDL Libraries Guide, version 12.2

DSP48A #(
    .AOREG(0), // Enable=1/disable=0 first stage A input pipeline register
    .AIREG(1), // Enable=1/disable=0 second stage A input pipeline register
    .BOREG(0), // Enable=1/disable=0 first stage B input pipeline register
    .BIREG(1), // Enable=1/disable=0 second stage B input pipeline register
    .CARRYINREG(1), // Enable=1/disable=0 CARRYIN input pipeline register
    .CARRYINSEL("CARRYIN"), // Specify carry-in source, "CARRYIN" or "OPMODE5"
    .CREG(1), // Enable=1/disable=0 C input pipeline register
    .DREG(1), // Enable=1/disable=0 D pre-adder input pipeline register
    .MREG(1), // Enable=1/disable=0 M pipeline register
    .OPMODEREG(1), // Enable=1/disable=0 OPMODE input pipeline register
    .PREG(1), // Enable=1/disable=0 P output pipeline register
    .RSTTYPE("SYNC") // Specify reset type, "SYNC" or "ASYNC"
) DSP48A_inst (
    .BCOUT(BCOUT), // 18-bit B port cascade output
    .CARRYOUT(CARRYOUT), // 1-bit carry output
    .P(P), // 48-bit output
    .PCOUT(PCOUT), // 48-bit cascade output
    .A(A), // 18-bit A data input
    .B(B), // 18-bit B data input (can be connected to fabric or BCOUT of adjacent DSP48A)
    .C(C), // 48-bit C data input
    .CARRYIN(CARRYIN), // 1-bit carry input signal
    .CEA(CEA), // 1-bit active high clock enable input for A input registers
    .CEB(CEB), // 1-bit active high clock enable input for B input registers
    .CEC(CEC), // 1-bit active high clock enable input for C input registers
    .CECARRYIN(CECARRYIN), // 1-bit active high clock enable input for CARRYIN registers
    .CED(CED), // 1-bit active high clock enable input for D input registers
    .CEM(CEM), // 1-bit active high clock enable input for multiplier registers
    .CEOPMODE(CEOPMODE), // 1-bit active high clock enable input for OPMODE registers
    .CEP(CEP), // 1-bit active high clock enable input for P output registers
    .CLK(CLK), // Clock input
    .D(D), // 18-bit B pre-adder data input
    .OPMODE(OPMODE), // 8-bit operation mode input
    .PCIN(PCIN), // 48-bit P cascade input
    .RSTA(RSTA), // 1-bit reset input for A input pipeline registers
    .RSTB(RSTB), // 1-bit reset input for B input pipeline registers
    .RSTC(RSTC), // 1-bit reset input for C input pipeline registers
    .RSTCARRYIN(RSTCARRYIN), // 1-bit reset input for CARRYIN input pipeline registers
    .RSTD(RSTD), // 1-bit reset input for D input pipeline registers
    .RSTM(RSTM), // 1-bit reset input for M pipeline registers
    .RSTOPMODE(RSTOPMODE), // 1-bit reset input for OPMODE input pipeline registers
    .RSTP(RSTP) // 1-bit reset input for P output pipeline registers
);

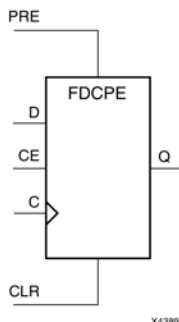
// End of DSP48A_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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).
--       Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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).
//       Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

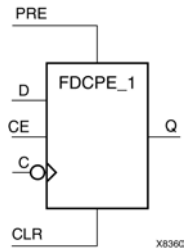
// End of FDCPE_inst instantiation
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

FDCPE_1

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



Introduction

FDCPE_1 is a single D-type flip-flop with data (D), clock enable (CE), asynchronous preset (PRE), and asynchronous clear (CLR) inputs and data output (Q). The asynchronous PRE, when High, sets the (Q) output High; CLR, when High, resets the output Low. Data on the (D) input is loaded into the flip-flop when PRE and CLR are Low and CE is High on the High-to-Low clock (C) transition. When CE is Low, the 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	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	Type	Allowed Values	Default	Description
INIT	Binary	0,1	0	Sets the initial value of Q output after configuration.

Verilog Instantiation Template

```
// FDCPE_1: Single Data Rate D Flip-Flop with Asynchronous Clear, Set and
//          Clock Enable (negedge clock).
//          Spartan-3A DSP
// Xilinx HDL Libraries Guide, version 12.2

FDCPE_1 #(
    .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)
) FDCPE_1_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
);

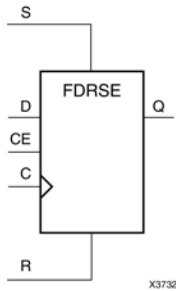
// End of FDCPE_1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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).
--       Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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).
//       Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

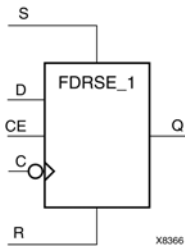
// End of FDRSE_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

FDRSE_1

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



Introduction

FDRSE_1 is a single D-type flip-flop with synchronous reset (R), synchronous set (S), and clock enable (CE) inputs and data output (Q). The reset (R) input, when High, overrides all other inputs and resets the (Q) output Low during the High-to-Low 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 High-to-Low 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 High-to-Low 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	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	D	↓	D

Design Entry Method

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

Available Attributes

Attribute	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;
```

Verilog Instantiation Template

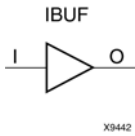
```
// FDRSE_1: Single Data Rate D Flip-Flop with Synchronous Clear, Set and  
//          Clock Enable (negedge clock).  
//          Spartan-3A DSP  
// Xilinx HDL Libraries Guide, version 12.2  
  
FDRSE_1 #(   
    .INIT(1'b0) // Initial value of register (1'b0 or 1'b1)  
) FDRSE_1_inst (   
    .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_1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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

In general, this element is inferred by the synthesis tool for any specified top-level input port to the design. It is generally not necessary to specify them in the source code. However, if desired, they be manually instantiated by either copying the instantiation code from the appropriate Libraries Guide HDL template and pasting it into the top-level entity/module of your code. It is recommended to always 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	Type	Allowed Values	Default	Description
IBUF_DELAY_VALUE	String	"0" through "16"	"0"	Specifies the amount of additional delay to add to the non-registered path out of the IOB.
IFD_DELAY_VALUE	String	"AUTO", "0" through "8"	"AUTO"	Specifies the amount of additional delay to add to the registered path within the IOB.
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IBUF_inst : IBUF
generic map (
    IBUF_DELAY_VALUE => "0", -- Specify the amount of added input delay for buffer,
                           -- "0"-"16"
    IFD_DELAY_VALUE => "AUTO", -- Specify the amount of added delay for input register,
                           -- "AUTO", "0"-"8"
    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
//      Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

IBUF #(
    .IBUF_DELAY_VALUE("0"), // Specify the amount of added input delay for
                           // the buffer: "0"-"16" (Spartan-3A)
    .IFD_DELAY_VALUE("AUTO"), // Specify the amount of added delay for input
                           // register: "AUTO", "0"-"8" (Spartan-3A)
    .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
```

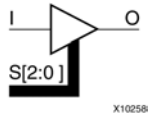
For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

IBUF_DLY_ADJ

Primitive: Dynamically Adjustable Input Delay Buffer

IBUF_DLY_AD J



Introduction

This design element is an input buffer with an adjustable delay element allowing dynamic delay adjustment (delay tuning) of an input signal into the FPGA. This is particularly useful for data aligning and capturing of high-speed input signals into the FPGA over process, voltage, and temperature variations. This component consists of a 3-bit select bus, which allows 8 unique values of delay to be added to the incoming signal. Additionally, the component can be programmed with a delay offset to delay adjustment within either the lower 8 or upper 8 of the 16 contiguous delay values.

See "For More Information" for details on the amount of delay and further details about usage of this component.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Delayed output from the buffer
I	Input	1	Differential input data (positive)
IB	Input	1	Differential input data (negative)
S	Input	3	Dynamic delay adjustment select lines

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed Values	Default	Description
DELAY_OFFSET	String	"OFF", "ON"	"OFF"	When set to "OFF", the IBUFDS_DLY_ADJ operates at the lower range of delay values. This should be used when a smaller amount of additional delay is needed. When set to "ON", the component operates at the upper (longer) range of delay values. This should be used when a larger amount of additional delay is needed.
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_DLY_ADJ: Single-ended Input Buffer
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IBUF_DLY_ADJ_inst : IBUF_DLY_ADJ
generic map (
    DELAY_OFFSET => "OFF", -- Enable initial Delay Offset, "OFF" or "ON"
    IOSTANDARD => "DEFAULT") -- Specify the input I/O standard
port map (
    O => O, -- Buffer output
    I => I, -- Buffer input (connect directly to top-level port)
    S => S -- 3-bit buffer delay select input
);

-- End of IBUF_DLY_ADJ_inst instantiation
```

Verilog Instantiation Template

```
// IBUF_DLY_ADJ: Dynamically Adjustable Delay, Single-ended Input Buffer
//           Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

IBUF_DLY_ADJ #(
    .DELAY_OFFSET("OFF"), // Enable Initial Delay Offset, "OFF" or "ON"
    .IOSTANDARD("DEFAULT") // Specify the input I/O standard
)IBUF_DLY_ADJ_inst (
    .O(O), // Buffer output
    .I(I), // Buffer input (connect directly to top-level port)
    .S(S) // 3-bit buffer delay select input
);

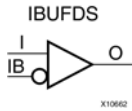
// End of IBUF_DLY_ADJ_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	Type	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	Type	Allowed Values	Default	Description
DIFF_TERM	Boolean	TRUE or FALSE	FALSE	Enables the built-in differential termination resistor.
IBUF_DELAY_VALUE	String	"0" through "16"	"0"	Specifies the amount of additional delay to add to the non-registered path out of the IOB.
IFD_DELAY_VALUE	String	"AUTO", "0" thru "8"	"AUTO"	Specifies the amount of additional delay to add to the registered path within the IOB.
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IBUFDS_inst : IBUFDS
generic map (
    DIFF_TERM => FALSE, -- Differential Termination
    IBUF_DELAY_VALUE => "0", -- Specify the amount of added input delay for buffer,
                        -- "0"-"16"
    IFD_DELAY_VALUE => "AUTO", -- Specify the amount of added delay for input register,
                        -- "AUTO", "0"-"8"
    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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

IBUFDS #(
    .IBUF_DELAY_VALUE("0"), // Specify the amount of added input delay for
                        // the buffer: "0"-"16" (Spartan-3A)
    .IFD_DELAY_VALUE("AUTO"), // Specify the amount of added delay for input
                        // register: "AUTO", "0"-"8" (Spartan-3A)
    .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

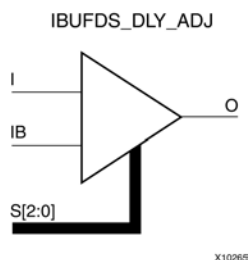
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

IBUFDS_DLY_ADJ

Primitive: Dynamically Adjustable Differential Input Delay Buffer



Introduction

This design element is a differential input buffer with an adjustable delay element allowing dynamic delay adjustment (delay tuning) of an input signal into the FPGA. This is particularly useful for data aligning and capturing of high-speed input signals into the FPGA over process, voltage, and temperature variations. This component consists of a 3-bit select bus, which allows 8 unique values of delay to be added to the incoming signal. Additionally, the component can be programmed with a delay offset to delay adjustment within either the lower 8 or upper 8 of the 16 contiguous delay values.

Port Descriptions

Port	Direction	Width	Function
O	Output	1	Delayed output from the buffer
I	Input	1	Differential input data (positive)
IB	Input	1	Differential input data (negative)
S	Input	3	Dynamic delay adjustment select lines

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed Values	Default	Description
DELAY_OFFSET	String	"OFF", "ON"	"OFF"	When set to OFF, the IBUFDS_DLY_ADJ operates at the lower range of delay values. This should be used when a smaller amount of additional delay is needed. When set to "ON", the component operates at the upper (longer) range of delay values. This should be used when a larger amount of additional delay is needed.
DIFF_TERM	Boolean	TRUE, FALSE	FALSE	Specifies the procedure for enabling or disabling (default) the internal differential termination capability.
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_DLY_ADJ: Differential Input Buffer
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IBUFDS_DLY_ADJ_inst : IBUFDS_DLY_ADJ
generic map (
    DIFF_TERM => FALSE, -- Differential Termination
    DELAY_OFFSET => "OFF", -- Enable initial Delay Offset, "OFF" or "ON"
    IOSTANDARD => "DEFAULT") -- Specify the input I/O standard
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)
    S => S, -- 3-bit buffer delay select input
);

-- End of IBUFDS_DLY_ADJ_inst instantiation
```

Verilog Instantiation Template

```
// IBUFDS_DLY_ADJ: Dynamically Adjustable Delay, Differential Input Buffer
//           Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

IBUFDS_DLY_ADJ #(
    .DELAY_OFFSET("OFF"), // Enable Initial Delay Offset, "OFF" or "ON"
    .DIFF_TERM("FALSE"), // Differential Termination
    .IOSTANDARD("DEFAULT") // Specify the input I/O standard
) IBUFDS_DLY_ADJ_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)
    .S(S) // 3-bit buffer delay select input
);

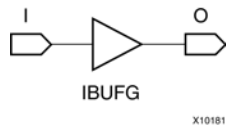
// End of IBUFDS_DLY_ADJ_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	Type	Allowed Values	Default	Description
IBUF_DELAY_VALUE	String	"0" through "16"	"0"	Specifies the amount of additional delay to add to the non-registered path out of the IOB
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IBUFG_inst : IBUFG
generic map (
    IBUF_DELAY_VALUE => "0", -- Specify the amount of added input delay for buffer,
                           -- "0"-"16"
    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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

IBUFG #(
    .IBUF_DELAY_VALUE("0"), // Specify the amount of added input delay for
                           // the buffer: "0"-"16" (Spartan-3A)
    .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)
);

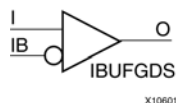
// End of IBUFG_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	Type	Allowed Values	Default	Description
DIFF_TERM	Boolean	TRUE or FALSE	FALSE	Enables the built-in differential termination resistor.
IBUF_DELAY_VALUE	String	"0" through "16"	"0"	Specifies the amount of additional delay to add to the non-registered path out of the IOB
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IBUFGDS_inst : IBUFGDS
generic map (
    DIFF_TERM => FALSE, -- Differential Termination
    IBUF_DELAY_VALUE => "0", -- Specify the amount of added input delay for buffer,
                           -- "0"-"16"
    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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

IBUFGDS #(
    .IBUF_DELAY_VALUE("0"), // Specify the amount of added input delay for
                           // the buffer, "0"-"16" (Spartan-3A)
    .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)
);

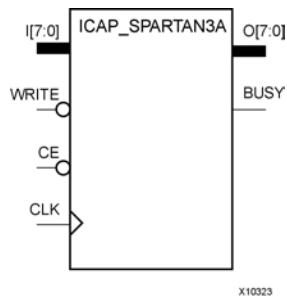
// End of IBUFGDS_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

ICAP_SPARTAN3A

Primitive: Internal Configuration Access Port



Introduction

This primitive works similar to the Slave Parallel (SelectMAP) configuration interface except it is available to the FPGA application using internal routing connections. Furthermore, the ICAP primitive has separate read and write data ports, as opposed to the bidirectional bus on the Slave Parallel (SelectMAP) interface. ICAP allows the FPGA application to access configuration registers, readback configuration data, or to trigger a MultiBoot event after configuration successfully completes.

Port Descriptions

Port	Direction	Width	Function
O	Output	8	Configuration data output bus
Busy	Output	8	Busy output
I	Input	8	Configuration data input bus
WRITE	Input	8	Active Low Write input
CE	Input	8	Active Low Clock Enable Input
CLK	Input	8	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;

-- ICAP_SPARTAN3A: Internal Configuration Access Port
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

ICAP_SPARTAN3A_inst : ICAP_SPARTAN3A
port map (
    BUSY => BUSY,    -- Busy output
    O => O,          -- 8-bit data output
    CE => CE,        -- Clock enable input
    CLK => CLK,      -- Clock input
    I => I,          -- 8-bit data input
    WRITE => WRITE   -- Write input
);

-- End of ICAP_SPARTAN3A_inst instantiation
```

Verilog Instantiation Template

```
// ICAP_SPARTAN3A: Internal Configuration Access Port
//           Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

ICAP_SPARTAN3A ICAP_SPARTAN3A_inst (
    .BUSY(BUSY),    // Busy output
    .O(O),         // 8-bit data output
    .CE(CE),       // Clock enable input
    .CLK(CLK),     // Clock input
    .I(I),         // 8-bit data input
    .WRITE(WRITE)  // Write input
);

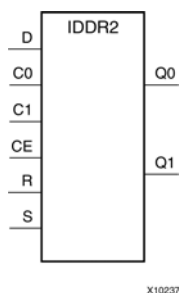
// End of ICAP_SPARTAN3A_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

IDDR2

Primitive: Double Data Rate Input D Flip-Flop with Optional Data Alignment, Clock Enable and Programmable Synchronous or Asynchronous Set/Reset



Introduction

This design element is a dedicated input register designed to receive external dual data rate (DDR) signals into Xilinx® FPGAs. The IDDR2 requires two clocks to be connected to the component, C0 and C1, so that data is captured at the positive edge of both C0 and C1 clocks. The IDDR2 features an active high clock enable port, CE, which be used to suspend the operation of the registers, and both set and reset ports that be configured to be synchronous or asynchronous to the respective clocks. The IDDR2 has an optional alignment feature that allows both output data ports to the component to be aligned to a single clock.

Logic Table

Input						Output	
S	R	CE	D	C0	C1	Q0	Q1
1	x	x	x	x	x	INIT_Q0	INIT_Q1
0	1	x	x	x	x	not INIT_Q0	not INIT_Q1
0	0	0	x	x	x	No Change	No Change
0	0	1	D	↑	x	D	No Change
0	0	1	D	x	↑	No Change	D

Set/Reset can be synchronous via SRTYPE value

Design Entry Method

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

To change the default behavior of the IDDR2, modify attributes via the generic map (VHDL) or named parameter value assignment (Verilog) as a part of the instantiated component. The IDDR2 can be connected directly to a top-level input port in the design, where an appropriate input buffer can be inferred, or directly to an instantiated IBUF, IOBUF, IBUFDS or IOBUFDS. All inputs and outputs of this component should either be connected or properly tied off.

Available Attributes

Attribute	Type	Allowed Values	Default	Description
DDR_ALIGNMENT	String	NONE, "C0", "C1"	"NONE"	Sets the output alignment more for the DDR register <ul style="list-style-type: none"> NONE - Makes the data available on the Q0 and Q1 outputs shortly after the corresponding C0 or C1 positive clock edge. C0 - Makes the data on both Q0 and Q1 align to the positive edge of the C0 clock. C1 - Makes the data on both Q0 and Q1 align to the positive edge of the C1 clock.
INIT_Q0	Integer	0, 1	0	Sets initial state of the Q0 output to 0 or 1.
INIT_Q1	Integer	0, 1	0	Sets initial state of the Q1 output to 0 or 1.
SRTYPE	String	"SYNC", "ASYNC"	"SYNC"	Specifies SYNC" or "ASYNC" set/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;

-- IDDR2: Input Double Data Rate Input Register with Set, Reset
--      and Clock Enable.
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IDDR2_inst : IDDR2
generic map(
  DDR_ALIGNMENT => "NONE", -- Sets output alignment to "NONE", "C0", "C1"
  INIT_Q0 => '0', -- Sets initial state of the Q0 output to '0' or '1'
  INIT_Q1 => '0', -- Sets initial state of the Q1 output to '0' or '1'
  SRTYPE => "SYNC") -- Specifies "SYNC" or "ASYNC" set/reset
port map (
  Q0 => Q0, -- 1-bit output captured with C0 clock
  Q1 => Q1, -- 1-bit output captured with C1 clock
  C0 => C0, -- 1-bit clock input
  C1 => C1, -- 1-bit clock input
  CE => CE, -- 1-bit clock enable input
  D => D,   -- 1-bit data input
  R => R,   -- 1-bit reset input
  S => S    -- 1-bit set input
);

-- End of IDDR2_inst instantiation
```

Verilog Instantiation Template

```
// IDDR2: Input Double Data Rate Input Register with Set, Reset
//      and Clock Enable.
//      Spartan-3E/3A/6
// Xilinx HDL Libraries Guide, version 12.2

IDDR2 #(
  .DDR_ALIGNMENT("NONE"), // Sets output alignment to "NONE", "C0" or "C1"
  .INIT_Q0(1'b0), // Sets initial state of the Q0 output to 1'b0 or 1'b1
  .INIT_Q1(1'b0), // Sets initial state of the Q1 output to 1'b0 or 1'b1
  .SRTYPE("SYNC") // Specifies "SYNC" or "ASYNC" set/reset
) IDDR2_inst (
  .Q0(Q0), // 1-bit output captured with C0 clock
```

```
.Q1(Q1), // 1-bit output captured with C1 clock
.C0(C0), // 1-bit clock input
.C1(C1), // 1-bit clock input
.CE(CE), // 1-bit clock enable input
.D(D),   // 1-bit DDR data input
.R(R),   // 1-bit reset input
.S(S)    // 1-bit set input
);

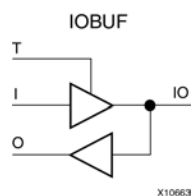
// End of IDDR2_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	Type	Allowed Values	Default	Description
DRIVE	Integer	2, 4, 6, 8, 12, 16, 24	12	Selects output drive strength (mA) for the SelectIO™ buffers that use the LVTTTL, LVC MOS12, LVC MOS15, LVC MOS18, LVC MOS25, or LVC MOS33 interface I/O standard.
IBUF_DELAY_VALUE	String	"0" through "16"	"0"	Specifies the amount of additional delay to add to the non-registered path out of the IOB
IFD_DELAY_VALUE	String	"AUTO", "0" through "8"	"AUTO"	Specifies the amount of additional delay to add to the registered path within the IOB
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IOBUF_inst : IOBUF
generic map (
    DRIVE => 12,
    IBUF_DELAY_VALUE => "0", -- Specify the amount of added input delay for buffer,
                           -- "0"-"16"
    IFD_DELAY_VALUE => "AUTO", -- Specify the amount of added delay for input register,
                           -- "AUTO", "0"-"8"
    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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

IOBUF #(
    .DRIVE(12), // Specify the output drive strength
    .IBUF_DELAY_VALUE("0"), // Specify the amount of added input delay for the buffer,
                          // "0"-"16" (Spartan-3A only)
    .IFD_DELAY_VALUE("AUTO"), // Specify the amount of added delay for input register,
                          // "AUTO", "0"-"8" (Spartan-3A only)
    .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
);

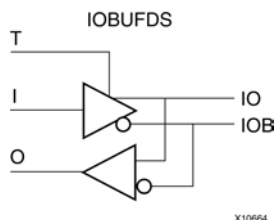
// End of IOBUF_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 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
1	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	Type	Allowed Values	Default	Description
IBUF_DELAY_VALUE	String	"0" through "16"	"0"	Specifies the amount of additional delay to add to the non-registered path out of the IOB
IFD_DELAY_VALUE	String	"AUTO", "0" through "8"	"AUTO"	Specifies the amount of additional delay to add to the registered path within the IOB
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

IOBUFDS_inst : IOBUFDS
generic map (
    IBUF_DELAY_VALUE => "0", -- Specify the amount of added input delay for buffer,
                           -- "0"-"12"
    IFD_DELAY_VALUE => "AUTO", -- Specify the amount of added delay for input register,
                           -- "AUTO", "0"-"8"
    IOSTANDARD => "DEFAULT")
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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

IOBUFDS #(
    .IBUF_DELAY_VALUE("0"), // Specify the amount of added input delay for the buffer,
                           // "0"-"16" (Spartan-3A only)
    .IFD_DELAY_VALUE("AUTO"), // Specify the amount of added delay for input register,
                           // "AUTO", "0"-"8" (Spartan-3A only)
    .IOSTANDARD("DEFAULT") // 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

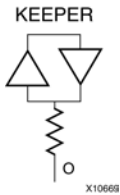
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

KEEPER KEEPER_inst (
    .O(0)      // Keeper output (connect directly to top-level port)
);

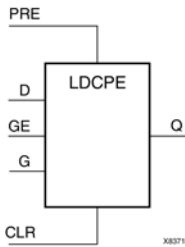
// End of KEEPER_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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.
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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.
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

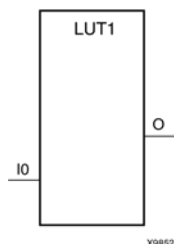
// End of LDCPE_inst instantiation
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

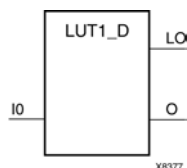
// End of LUT1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 output 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 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	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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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

-- End of LUT1_D_inst instantiation
```

Verilog Instantiation Template

```
// LUT1_D: 1-input Look-Up Table with general and local outputs
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

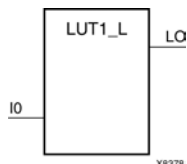
// End of LUT1_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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

-- End of LUT1_L_inst instantiation
```

Verilog Instantiation Template

```
// LUT1_L: 1-input Look-Up Table with local output
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

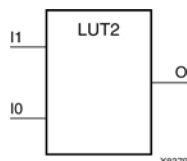
// End of LUT1_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

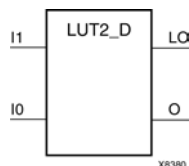
// End of LUT2_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 output 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 LogicTable 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	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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

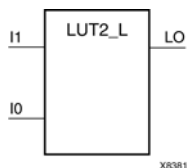
// End of LUT2_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--          Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//          Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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

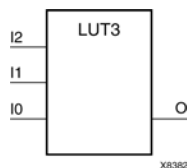
// End of LUT2_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

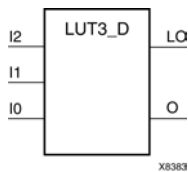
// End of LUT3_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 output 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 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	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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

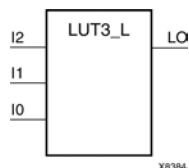
// End of LUT3_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

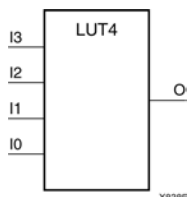
// End of LUT3_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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]
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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

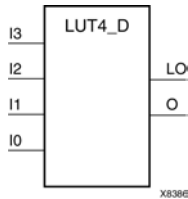
// End of LUT4_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 output 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 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	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]
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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

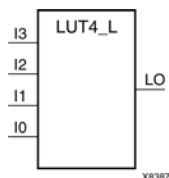
// End of LUT4_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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]
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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

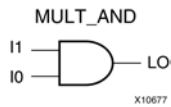
// End of LUT4_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//          Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

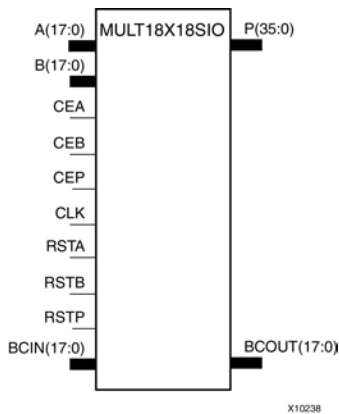
// End of MULT_AND_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MULT18X18SIO

Primitive: 18 x 18 Cascadable Signed Multiplier with Optional Input and Output Registers, Clock Enable, and Synchronous Reset



Introduction

This design element is a 36-bit output, 18x18-bit input dedicated signed multiplier. This component can perform asynchronous multiplication operations when the attributes AREG, BREG and PREG are all set to 0. Alternatively, synchronous multiplication operations of different latency and performance characteristics can be performed when any combination of those attributes is set to 1. When using the multiplier in synchronous operation, the MULT18X18SIO features active high clock enables for each set of register banks in the multiplier, CEA, CEB and CEP, as well as synchronous resets, RSTA, RSTB, and RSTP. Multiple MULT18X18SIOs can be cascaded to create larger multiplication functions using the BCIN and BCOUT ports in combination with the B_INPUT attribute.

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed Values	Default	Descriptions
AREG	Integer	0, 1	1	Specifies the use of the input registers on the A port. A zero disables the use of the register; a one enables the register.
BREG	Integer	0, 1	1	Specifies the use of the input registers on the B port. A zero disables the use of the register; a one enables the register.
B_INPUT	String	"DIRECT" or "CASCADE"	"DIRECT"	Specifies whether the B port is connected to the general FPGA fabric, "DIRECT" or is connected to the BCOUT port of another MULT18X18SIO.
PREG	Integer	0, 1	1	Specifies the use of the output registers of the multiplier. A zero disables the use of the register; a one enables the register.

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;

-- MULT18X18SIO: 18 x 18 cascadable, signed synchronous/asynchronous multiplier
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MULT18X18SIO_inst : MULT18X18SIO
generic map (
    AREG => 1, -- Enable the input registers on the A port (1=on, 0=off)
    BREG => 1, -- Enable the input registers on the B port (1=on, 0=off)
    B_INPUT => "DIRECT", -- B cascade input "DIRECT" or "CASCADE"
    PREG => 1) -- Enable the input registers on the P port (1=on, 0=off)
port map (
    BCOUT => BCOUT, -- 18-bit cascade output
    P => P, -- 36-bit multiplier output
    A => A, -- 18-bit multiplier input
    B => B, -- 18-bit multiplier input
    BCIN => BCIN, -- 18-bit cascade input
    CEA => CEA, -- Clock enable input for the A port
    CEB => CEB, -- Clock enable input for the B port
    CEP => CEP, -- Clock enable input for the P port
    CLK => CLK, -- Clock input
    RSTA => RSTA, -- Synchronous reset input for the A port
    RSTB => RSTB, -- Synchronous reset input for the B port
    RSTP => RSTP, -- Synchronous reset input for the P port
);

-- End of MULT18X18SIO_inst instantiation
```

Verilog Instantiation Template

```
// MULT18X18SIO: 18 x 18 cascadable, signed synchronous/asynchronous multiplier
//           Spartan-3E/3A
// Xilinx HDL Libraries Guide, version 12.2

MULT18X18SIO #(
    .AREG(1), // Enable the input registers on the A port (1=on, 0=off)
    .BREG(1), // Enable the input registers on the B port (1=on, 0=off)
    .B_INPUT("DIRECT"), // B cascade input "DIRECT" or "CASCADE"
    .PREG(1) // Enable the input registers on the P port (1=on, 0=off)
) MULT18X18SIO_inst (
    .BCOUT(BCOUT), // 18-bit cascade output
    .P(P), // 36-bit multiplier output
    .A(A), // 18-bit multiplier input
    .B(B), // 18-bit multiplier input
    .BCIN(BCIN), // 18-bit cascade input
    .CEA(CEA), // Clock enable input for the A port
    .CEB(CEB), // Clock enable input for the B port
    .CEP(CEP), // Clock enable input for the P port
    .CLK(CLK), // Clock input
    .RSTA(RSTA), // Synchronous reset input for the A port
    .RSTB(RSTB), // Synchronous reset input for the B port
    .RSTP(RSTP) // Synchronous reset input for the P port
);

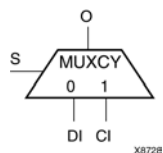
// End of MULT18X18SIO_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

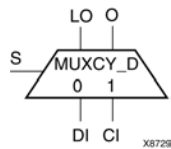
// End of MUXCY_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

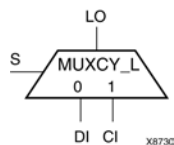
// End of MUXCY_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

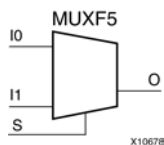
// End of MUXCY_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF5

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element provides a multiplexer function in a CLB slice for creating a function-of-5 look-up table or a 4-to-1 multiplexer in combination with the associated look-up tables. The local outputs (LO) from the two 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 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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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 directly 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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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 directly to the output of LUT4)
    .S(S)     // Input select to MUX
);

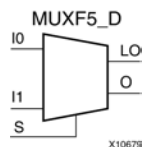
// End of MUXF5_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF5_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element provides a multiplexer function in a CLB slice for creating a function-of-5 look-up table or a 4-to-1 multiplexer in combination with the associated look-up tables. The local outputs (LO) from the two 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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF5_D_inst : MUXF5_D
port map (
    LO => LO,  -- Ouput of MUX to local routing
    O  => O,   -- Output of MUX to general routing
    IO => IO,  -- Input (tie directly to the output of LUT4)
    I1 => I1,  -- Input (tie directly 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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF5_D MUXF5_D_inst (
    .LO(LO),  // Ouput of MUX to local routing
    .O(O),   // Output of MUX to general routing
    .IO(IO), // Input (tie directly to the output of LUT4)
    .I1(I1), // Input (tie directly to the output of LUT4)
    .S(S)    // Input select to MUX
);

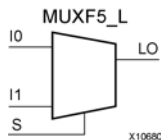
// End of MUXF5_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF5_L

Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output



Introduction

This design element provides a multiplexer function in a CLB slice for creating a function-of-5 look-up table or a 4-to-1 multiplexer in combination with the associated look-up tables. The local outputs (LO) from the two 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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF5_L_inst : MUXF5_L
port map (
    LO => LO,  -- Output of MUX to local routing
    IO => IO,  -- Input (tie directly to the output of LUT4)
    I1 => I1,  -- Input (tie directly 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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF5_L MUXF5_L_inst (
    .LO(LO),  // Output of MUX to local routing
    .IO(IO),  // Input (tie directly to the output of LUT4)
    .I1(I1),  // Input (tie directly to the output of LUT4)
    .S(S)     // Input select to MUX
);

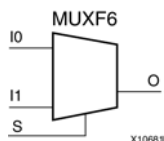
// End of MUXF5_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF6

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element provides a multiplexer function in two slices for creating a function-of-6 look-up table or an 8-to-1 multiplexer in combination with the associated four look-up tables and two MUXF5s. The local outputs (LO) from the two MUXF5s 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 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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF6 MUXF6_inst (
    .O(O),    // Output of MUX to general routing
    .IO(I0),  // Input (tie to MUXF5 LO out)
    .I1(I1),  // Input (tie to MUXF5 LO out)
    .S(S)     // Input select to MUX
);

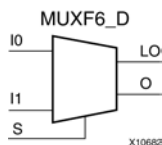
// End of MUXF6_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF6_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element provides a multiplexer function in a two slices for creating a function-of-6 look-up table or an 8-to-1 multiplexer in combination with the associated four look-up tables and two MUXF5s. The local outputs (LO) from the two MUXF5s 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.

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF6_D_inst : MUXF6_D
port map (
    LO => LO,  -- Ouput of MUX to local routing
    O  => O,   -- Output of MUX to general routing
    IO => IO,  -- 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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF6_D MUXF6_D_inst (
    .LO(LO),  // Ouput of MUX to local routing
    .O(O),   // Output of MUX to general routing
    .IO(IO), // Input (tie to MUXF5 LO out)
    .I1(I1), // Input (tie to MUXF5 LO out)
    .S(S)    // Input select to MUX
);

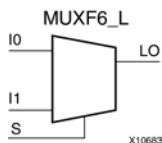
// End of MUXF6_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF6_L

Primitive: 2-to-1 Look-Up Table Multiplexer with Local Output



Introduction

This design element provides a multiplexer function for use in creating a function-of-6 look-up table or an 8-to-1 multiplexer in combination with the associated four look-up tables and two MUXF5s. The local outputs (LO) from the two MUXF5s 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.

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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF6_L MUXF6_L_inst (
    .LO(LO), // Output of MUX to local routing
    .IO(IO), // Input (tie to MUXF5 LO out)
    .I1(I1), // Input (tie to MUXF5 LO out)
    .S(S)    // Input select to MUX
);

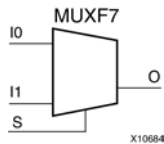
// End of MUXF6_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF7

Primitive: 2-to-1 Look-Up Table Multiplexer with General Output



Introduction

This design element provides a multiplexer function for use in creating a function-of-7 look-up table or an 8-to-1 multiplexer in combination with the associated look-up tables. Local outputs (LO) of 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 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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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 LUT6's or MUXF6's together with general output
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

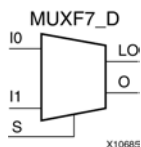
// End of MUXF7_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF7_D

Primitive: 2-to-1 Look-Up Table Multiplexer with Dual Output



Introduction

This design element provides a multiplexer function for use in creating a function-of-7 look-up table or a 16-to-1 multiplexer in combination with the associated look-up tables. Local outputs (LO) of 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.

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF7_D_inst : MUXF7_D
port map (
    LO => LO,  -- Ouput of MUX to local routing
    O  => O,   -- Output of MUX to general routing
    IO => IO,  -- 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 LUT6's or MUXF6's together with general and local outputs
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF7_D MUXF7_D_inst (
    .LO(LO),  // Ouput of MUX to local routing
    .O(O),   // Output of MUX to general routing
    .IO(IO), // 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
);

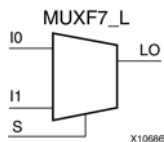
// End of MUXF7_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

MUXF7_L

Primitive: 2-to-1 look-up table Multiplexer with Local Output



Introduction

This design element provides a multiplexer function for use in creating a function-of-7 look-up table or a 16-to-1 multiplexer in combination with the associated look-up tables. Local outputs (LO) of 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.

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF7_L_inst : MUXF7_L
port map (
    LO => LO,  -- Output of MUX to local routing
    IO => IO,  -- 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 LUT6's or MUXF6's together with local output
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF7_L MUXF7_L_inst (
    .LO(LO),  // Output of MUX to local routing
    .IO(IO),  // 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
);

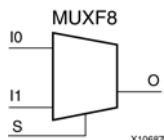
// End of MUXF7_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 16-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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

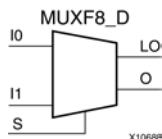
// End of MUXF8_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 four look-up tables and two MUXF8s. 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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF8_D_inst : MUXF8_D
port map (
    LO => LO,  -- Ouput of MUX to local routing
    O  => O,   -- Output of MUX to general routing
    IO => IO,  -- 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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF8_D MUXF8_D_inst (
    .LO(LO),  // Ouput of MUX to local routing
    .O(O),   // Output of MUX to general routing
    .IO(IO), // Input (tie to MUXF7 LO out)
    .I1(I1), // Input (tie to MUXF7 LO out)
    .S(S)    // Input select to MUX
);

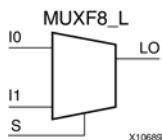
// End of MUXF8_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 four look-up tables and two MUXF8s. 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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

MUXF8_L_inst : MUXF8_L
port map (
    LO => LO,  -- Output of MUX to local routing
    IO => IO,  -- 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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

MUXF8_L MUXF8_L_inst (
    .LO(LO),  // Output of MUX to local routing
    .IO(IO),  // Input (tie to MUXF7 LO out)
    .I1(I1),  // Input (tie to MUXF7 LO out)
    .S(S)     // Input select to MUX
);

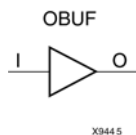
// End of MUXF8_L_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 LVTTTL. 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	Type	Allowed Values	Default	Description
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

OBUF_inst : OBUF
generic map (
    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
//      Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

OBUF #(
    .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
);

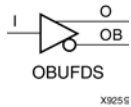
// End of OBUF_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	Type	Allowed Values	Default	Description
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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

OBUFDS_inst : OBUFDS
generic map (
    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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

OBUFDS #(
    .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
);

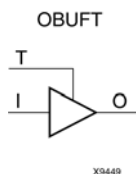
// End of OBUFDS_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 LVTTTL 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	Type	Allowed Values	Default	Description
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. 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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

OBUFT_inst : OBUFT
generic map (
    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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

OBUFT #(
    .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
);

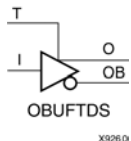
// End of OBUFT_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	Type	Allowed Values	Default	Description
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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

OBUFTDS_inst : OBUFTDS
generic map (
    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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

OBUFTDS #(
    .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
);

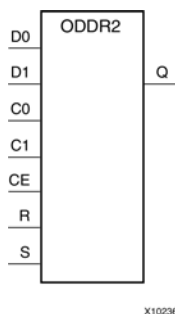
// End of OBUFTDS_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

ODDR2

Primitive: Dual Data Rate Output D Flip-Flop with Optional Data Alignment, Clock Enable and Programmable Synchronous or Asynchronous Set/Reset



Introduction

The design element is an output double data rate (DDR) register useful in producing double data rate signals exiting the FPGA. The ODDR2 requires two clocks (C0 and C1) to be connected to the component so that data is provided at the positive edge of both clocks. The ODDR2 features an active high clock enable port, CE, which can be used to suspend the operation of the registers and both set and reset ports that can be configured to be synchronous or asynchronous to the respective clocks. The ODDR2 has an optional alignment feature, which allows data to be captured by a single clock and clocked out by two clocks.

Logic Table

Inputs							Outputs
S	R	CE	D0	D1	C0	C1	O
1	X	X	X	X	X	X	1
0	1	X	X	X	X	X	not INIT
0	0	0	X	X	X	X	No Change
0	0	1	D0	X	↑	X	D0
0	0	1	X	D1	X	↑	D1
Set/Reset can be synchronous via SRTYPE value							

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed Values	Default	Descriptions
DDR_ALIGNMENT	String	"NONE", "C0", "C1"	"NONE"	Sets the input capture behavior for the DDR register. "NONE" clocks in data to the D0 input on the positive transition of the C0 clock and D1 on the positive transition of the C1 clock. "C0" allows the input clocking of both D0 and D1 align to the positive edge of the C0 clock. "C1" allows the input clocking of both D0 and D1 align to the positive edge of the C1 clock.
INIT	Integer	0, 1	0	Sets initial state of the Q0 output to 0 or 1.
SRTYPE	String	"SYNC", "ASYN"	"SYNC"	Specifies "SYNC" or "ASYN" set/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;

-- ODDR2: Output Double Data Rate Output Register with Set, Reset
--      and Clock Enable.
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

ODDR2_inst : ODDR2
generic map(
    DDR_ALIGNMENT => "NONE", -- Sets output alignment to "NONE", "C0", "C1"
    INIT => '0', -- Sets initial state of the Q output to '0' or '1'
    SRTYPE => "SYNC") -- Specifies "SYNC" or "ASYN" set/reset
port map (
    Q => Q, -- 1-bit output data
    C0 => C0, -- 1-bit clock input
    C1 => C1, -- 1-bit clock input
    CE => CE, -- 1-bit clock enable input
    D0 => D0, -- 1-bit data input (associated with C0)
    D1 => D1, -- 1-bit data input (associated with C1)
    R => R, -- 1-bit reset input
    S => S -- 1-bit set input
);

-- End of ODDR2_inst instantiation

```

Verilog Instantiation Template

```

// ODDR2: Output Double Data Rate Output Register with Set, Reset
//      and Clock Enable.
//      Spartan-3E/3A/6
// Xilinx HDL Libraries Guide, version 12.2

ODDR2 #(
    .DDR_ALIGNMENT("NONE"), // Sets output alignment to "NONE", "C0" or "C1"
    .INIT(1'b0), // Sets initial state of the Q output to 1'b0 or 1'b1
    .SRTYPE("SYNC")) // Specifies "SYNC" or "ASYN" set/reset
) ODDR2_inst (
    .Q(Q), // 1-bit DDR output data
    .C0(C0), // 1-bit clock input
    .C1(C1), // 1-bit clock input
    .CE(CE), // 1-bit clock enable input
    .D0(D0), // 1-bit data input (associated with C0)

```

```
.D1(D1), // 1-bit data input (associated with C1)
.R(R),   // 1-bit reset input
.S(S)    // 1-bit set input
);

// End of ODDR2_inst instantiation
```

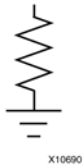
For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

PULLDOWN

Primitive: Resistor to GND for Input Pads, Open-Drain, and 3-State Outputs

PULLDOWN



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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

PULLDOWN PULLDOWN_inst (
    .O(0)      // Pulldown output (connect directly to top-level port)
);

// End of PULLDOWN_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

PULLUP PULLUP_inst (
    .O(0)      // Pullup output (connect directly to top-level port)
);

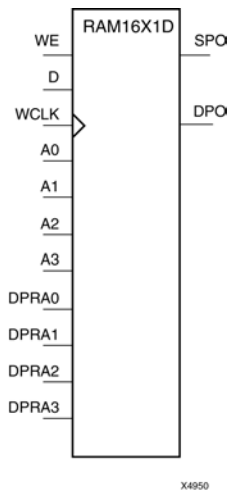
// End of PULLUP_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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 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_inst instantiation

```

Verilog Instantiation Template

```
// RAM16X1D: 16 x 1 positive edge write, asynchronous read dual-port distributed RAM
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

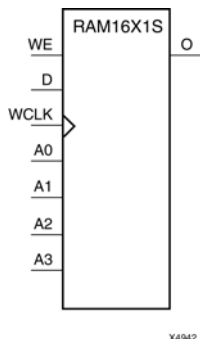
// End of RAM16X1D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

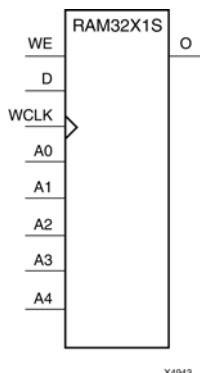
// End of RAM16X1S_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--          Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//          Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

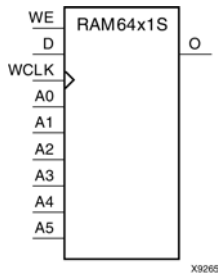
// End of RAM32X1S_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

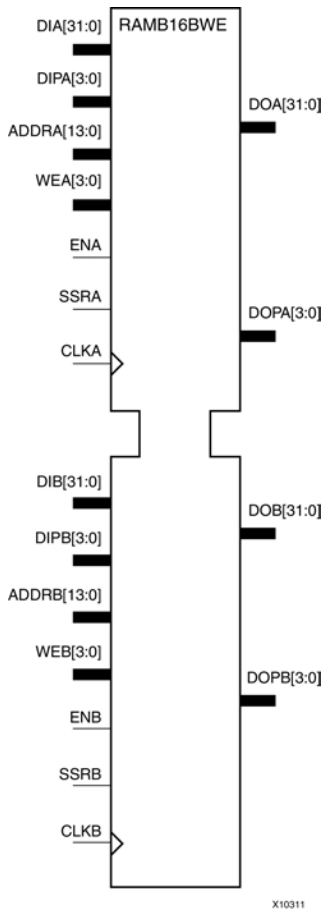
// End of RAM64X1S_inst instantiation
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

RAMB16BWE

Primitive: 16K-bit Data and 2K-bit Parity Configurable Synchronous Dual Port Block RAM



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR[13:0]	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.

Port	Direction	Width	Function
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

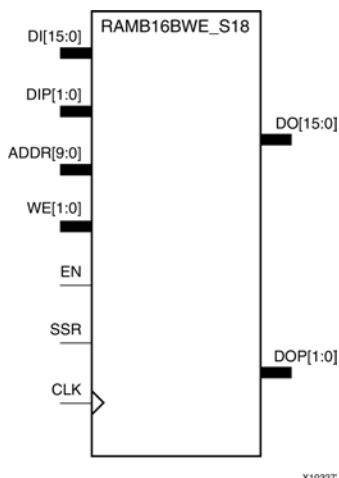
Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_ CHECK	String	ALL, "WARNING_ ONLY", "GENERATE_X_ ONLY" or "NONE"	"ALL"	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"	"WRITE_ FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST" = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port. NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S18

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Single Port Block RAM with 18-bit Port



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR B	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

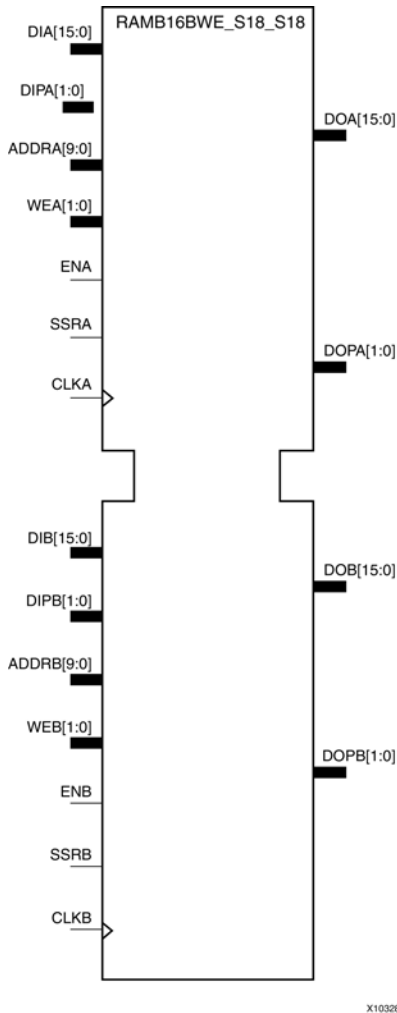
Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_ CHECK	String	ALL, "WARNING_ ONLY", "GENERATE_X_ ONLY" or "NONE"	"ALL"	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"	"WRITE_ FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST" = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port. NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S18_S18

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 18-bit Ports



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR B	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_CHECK	String	ALL, "WARNING_ONLY", "GENERATE_X_ONLY" or "NONE"	"ALL"	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"	"WRITE_FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port.

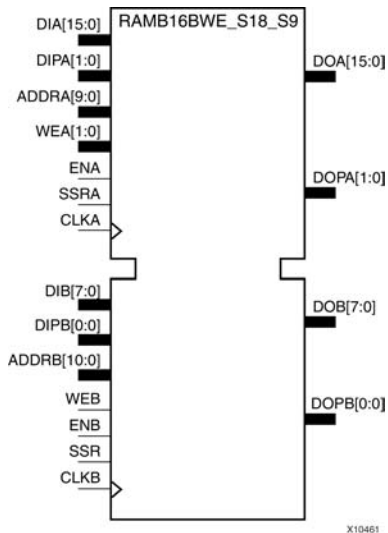
Attribute	Type	Allowed Values	Default	Description
				<ul style="list-style-type: none">NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S18_S9

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 18-bit and 9-bit Ports



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR[10:0]	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

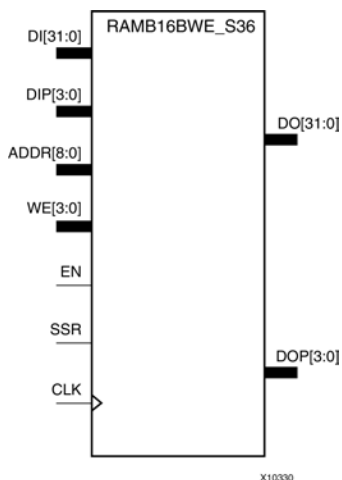
Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_ CHECK	String	ALL, "WARNING_ ONLY", "GENERATE_X_ ONLY" or "NONE"	"ALL"	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"	"WRITE_ FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST" = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port. NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S36

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Single Port Block RAM with 36-Bit Port



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR B	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

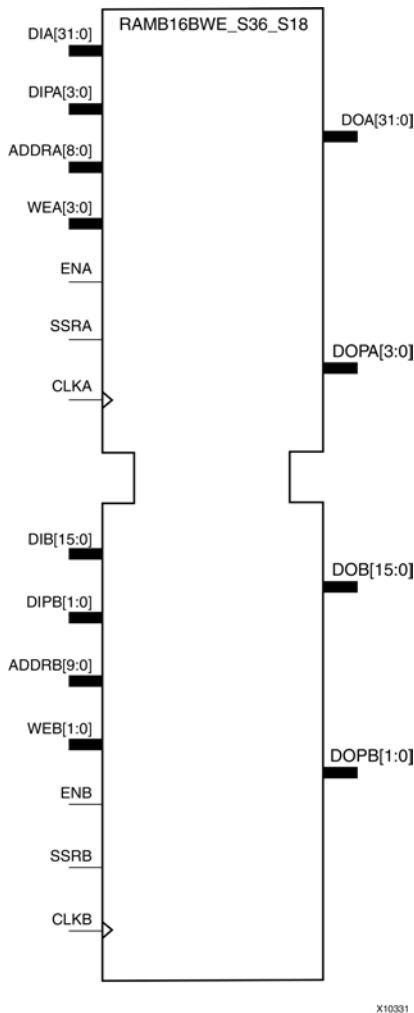
Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_ CHECK	String	ALL, "WARNING_ ONLY", "GENERATE_X_ ONLY" or "NONE"	"ALL"	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"	"WRITE_ FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST" = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port. NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S36_S18

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 36-bit and 18-bit Ports



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR B	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_CHECK	String	ALL, "WARNING_ONLY", "GENERATE_X_ONLY" or "NONE"	"ALL"	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"	"WRITE_FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port.

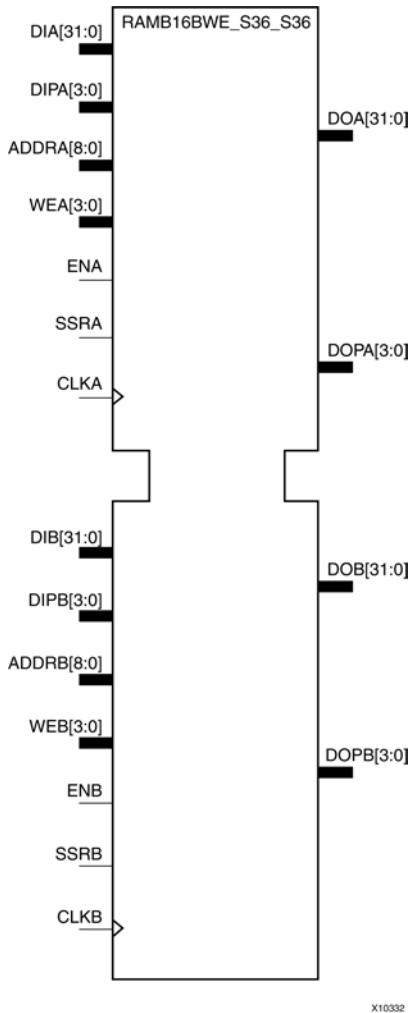
Attribute	Type	Allowed Values	Default	Description
				<ul style="list-style-type: none">NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S36_S36

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 36-bit Ports



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR B	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

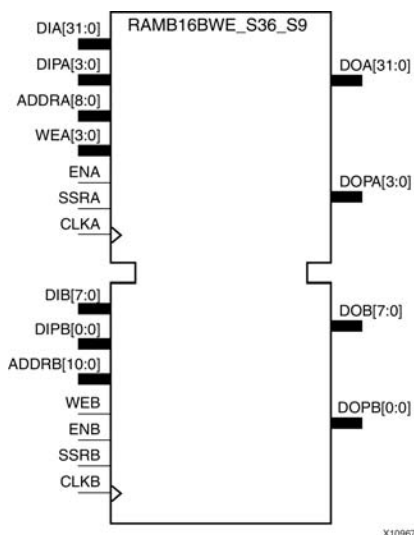
Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWE_S36_S9

Primitive: 16K-bit Data and 2K-bit Parity Synchronous Dual Port Block RAM with 36-bit and 9-bit Ports



Introduction

This design element can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B may operate fully independently and asynchronously to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible. This block RAM memory offers fast and flexible storage of large amounts of on-chip data.

Port Descriptions

Port	Direction	Width	Function
DOA, DOB	Output	32	Port A/B data output bus.
DOPA, DOPB	Output	4	Port A/B parity output bus.
DIA, DIB	Input	32	Port A/B data input bus.
DIPA, DIPB	Input	4	Port A/B parity input bus.
ADDRA, ADDR[10:0]	Input	14	Port A/B address input bus. MSB always exists on ADDRA/B while the LSB is determined by the settings for DATA_WIDTH_A/B.
WEA, WEB	Input	4	Port A/B byte-wide write enable.
ENA, ENB	Input	1	Port A/B enable
SSRA, SSRB	Input	1	Port A/B output registers synchronous reset.
CLKA, CLKB	Input	1	Port A/B clock input.

Design Entry Method

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

This element can be inferred by most synthesis tools by properly describing the RAM behavior in standard RTL code (consult synthesis tool documentation for details). Alternatively, CORE Generator™ can also create the desired macro for this RAM. If it is desired to have more control over the implementation or placement of this component, it may also be directly instantiated. To instantiate this component, use the HDL templates in the ISE® software or the instantiation template below and paste into your code. Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation and the SSRA/SSRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. Refer to the DATA_WIDTH table below for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting as the necessary connections for these signals change based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH values for either Port A or Port B:

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections	DO, DOP Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal	DO[0]
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal	DO[1:0]
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal	DO[3:0]
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal	DO[7:0], DOP[0]
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1]	DO[15:0], DOP[1:0]
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.	DO[31:0], DOP[3:0]

Alternatively, the prior RAMB16_Sm_Sn design elements can be instantiated if a byte-enable operation is not necessary. Also, new convenience macros called RAMB16BWE_Sm_Sn are provided to allow for easier instantiation of this RAM with byte-enable operation. If either of these components is used, the software automatically re-targets it to a properly configured RAMB16BWE component.

Available Attributes

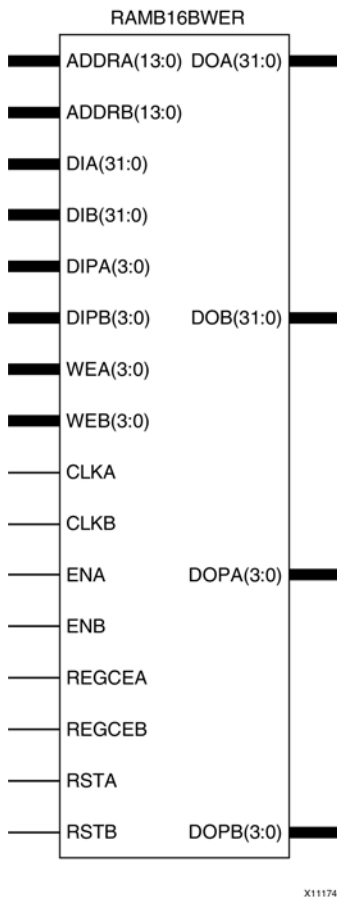
Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A, DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, or 36	0	Specifies the configurable data width for Ports A and B.
INIT_A, INIT_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the initial value on the Port B output after configuration.
SIM_COLLISION_ CHECK	String	ALL, "WARNING_ ONLY", "GENERATE_X_ ONLY" or "NONE	"ALL	<p>Allows modification of the simulation behavior if a memory collision occurs. The output is affected as follows:</p> <ul style="list-style-type: none"> "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. <p>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 <i>Synthesis and Simulation Design Guide</i> for more information.</p>
SRVAL_A, SRVAL_B	Hexadecimal	Any 36-Bit Value	All zeros	Specifies the output value of Port B upon the assertion of the synchronous reset (SSRB) signal.
WRITE_MODE_A, WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", or "NO_CHANGE	"WRITE_FIRST	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> WRITE_FIRST" = written value appears on output port of the RAM. READ_FIRST = previous RAM contents for that memory location appear on the output port. NO_CHANGE = previous value on the output port remains the same.
INIT_00 to INIT_3F	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 16kb data memory array.
INITP_00 to INITP_07	Hexadecimal	Any 256-Bit Value	All zeros	Allows specification of the initial contents of the 2kb parity data memory array.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

RAMB16BWER

Primitive: 16K-bit Data and 2K-bit Parity Configurable Synchronous Dual Port Block RAM with Optional Output Registers



Introduction

This design element contains several block RAM memories that can be configured as general-purpose 16kb data + 2kb parity RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. This component can be configured and used as a 1-bit wide by 16K deep to a 36-bit wide by 512 deep, single-port or dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, Port A and Port B can operate fully independently and asynchronously to each other, accessing the same memory array. When these ports are configured in the wider data width modes, byte-enable write operations are possible. This RAM also offers a configurable output register that can be enabled to improve clock-to-out times of the RAM while incurring an extra clock cycle of latency during the read operation.

Port Descriptions

The following table shows the necessary input and output connections for the variable input ports for each DATA_WIDTH value for either Port A or Port B.

DATA_WIDTH Value	DI, DIP Connections	ADDR Connections	WE Connections
1	DI[0]	ADDR[13:0]	Connect WE[3:0] to single user WE signal.
2	DI[1:0]	ADDR[13:1]	Connect WE[3:0] to single user WE signal.
4	DI[3:0]	ADDR[13:2]	Connect WE[3:0] to single user WE signal.
9	DI[7:0], DIP[0]	ADDR[13:3]	Connect WE[3:0] to single user WE signal.
18	DI[15:0], DIP[1:0]	ADDR[13:4]	Connect WE[0] and WE[2] to user WE[0] and WE[1] and WE[3] to user WE[1].
36	DI[31:0], DIP[3:0]	ADDR[13:5]	Connect each WE[3:0] signal to the associated byte write enable.

Alternatively, the older RAMB16_Sm_Sn and RAMB16BWER_Sm_Sn elements can be instantiated if the output registers are not necessary. If any of these components are used, the software will automatically retarget them to a properly configured RAMB16BWER element.

Port	Direction	Width	Function
ADDRA[13:0]	Input	14	Port A address input bus. MSB always exists on ADDRA[13] while the LSB is determined by the settings for DATA_WIDTH_A.
ADDRB[13:0]	Input	14	Port B address input bus. MSB always exists on ARDRB[13] while the LSB is determined by the settings for DATA_WIDTH_B.
CLKA	Input	1	Port A clock input.
CLKB	Input	1	Port B clock input.
DIA[31:0]	Input	32	Port A data input bus.
DIB[31:0]	Input	32	Port B data input bus.
DIPA[3:0]	Input	4	Port A parity input bus.
DIPB[3:0]	Input	4	Port B parity input bus.
DOA[31:0]	Output	32	Port A data output bus.
DOB[31:0]	Output	32	Port B data output bus.
DOPA[3:0]	Output	4	Port A parity output bus.
DOPB[3:0]	Output	4	Port B parity output bus.
ENA	Input	1	Port A enable.
ENB	Input	1	Port B enable.
REGCEA	Input	1	Output register clock enable.
REGCEB	Input	1	Output register clock enable.
RSTA	Input	1	Port A output registers set/reset. This reset is configurable to be synchronous or asynchronous, depending on the value of the RSTTYPE attribute.
RSTB	Input	1	Port B output registers set/reset. This reset is configurable to be synchronous or asynchronous, depending on the value of the RSTTYPE attribute.
WEA[3:0]	Input	4	Port A byte-wide write enable.
WEB[3:0]	Input	4	Port B byte-wide write enable.

Design Entry Method

Instantiation	Yes
Inference	Recommended
CORE Generator™ and wizards	Yes
Macro support	Yes

Connect all necessary inputs to the desired signals in the design. The CLKA/CLKB clock signals must be tied to an active clock for RAM operation, and the SRA/SRB reset signals must be either tied to a logic zero or to the proper reset signal. ENA/ENB must either be tied to a logic one or a proper RAM port enable signal. REGCEA and REGCEB must be tied to the proper output register clock enable, or a logic one if the respective DOA_REG or DOB_REG attribute is set to 1. If DOA_REG is set to 0, then REGCEA and REGCEB must be set to a logic 0.

Refer to the DATA_WIDTH column in the “Port Description” table (above) for the necessary data input, data output, write enable and address connection information for each DATA_WIDTH setting, since the necessary connections for these signals change, based on this attribute. All other output signals can be left unconnected (open) and all unused input signals should be tied to a logic zero.

Available Attributes

Attribute	Type	Allowed Values	Default	Description
DATA_WIDTH_A	Integer	0, 1, 2, 4, 9, 18, 36	0	Specifies the configurable data width for port A. Need not equal the width for port B.
DATA_WIDTH_B	Integer	0, 1, 2, 4, 9, 18, 36	0	Specifies the configurable data width for port B. Need not equal the width for port A.
DOA_REG	Integer	0, 1	0	Set to 1 to use the A port output registers.
DOB_REG	Integer	0, 1	0	Set to 1 to use the B port output registers.
INIT_A	Hexa-decimal	36'h000000000 to 36'hfffffff	All zeros	Specifies the initial value on the port A output after configuration.
INIT_B	Hexa-decimal	36'h000000000 to 36'hfffffff	All zeros	Specifies the initial value on the Port B output after configuration.
INIT_FILE	String	0 bit String	NONE	File name of file used to specify initial RAM contents.
INIT_00 to INIT_3F	Hexa-decimal	Any 256 bit value	All zeros	Specifies the initial contents of the 16 kb data memory array.
INITP_01 to INITP_07	Hexa-decimal	Any 256 bit value	All zeros	Specifies the initial contents of the 2 kb parity data memory array.
RSTTYPE	String	“SYNC”, “ASYNC”	“SYNC”	Selects whether the RAM outputs should have a synchronous or asynchronous reset capability. Due to improved timing and circuit stability, it is recommended to always have this set to “SYNC” unless an asynchronous reset is absolutely necessary.

Attribute	Type	Allowed Values	Default	Description
SIM_COLLISION_CHECK	String	"ALL", "GENERATE_X_ONLY", "WARNING_ONLY", "NONE"	"ALL"	<p>Allows modification of the simulation behavior so that if a memory collision occurs:</p> <ul style="list-style-type: none"> • ALL - Warning produced and affected outputs/memory go unknown (X). • WARNING_ONLY - Warning produced and affected outputs/memory retain last value. • GENERATE_X_ONLY - No warning, but affected outputs/memory go unknown (X). • NONE - No warning and affected outputs/memory retain last value. <p>Note Setting this to a value other than "ALL" can allow problems in the design to go unnoticed during simulation. Care should be taken when changing the value of this attribute.</p>
SRVAL_A	Hexa-decimal	36'h00000000 to 36'hffffffff	All zeros	Specifies the output value of Port A upon the assertion of the reset (RSTA) signal.
SRVAL_B	Hexa-decimal	36'h00000000 to 36'hffffffff	All zeros	Specifies the output value of Port B upon the assertion of the reset (RSTB) signal.
WRITE_MODE_A	String	"WRITE_FIRST", "READ_FIRST", "NO_CHANGE"	"WRITE_FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> • WRITE_FIRST - Written value appears on output port of the RAM. • READ_FIRST - Previous RAM contents for that memory location appear on the output port. • NO_CHANGE - Previous value on the output port remains the same.
WRITE_MODE_B	String	"WRITE_FIRST", "READ_FIRST", "NO_CHANGE"	"WRITE_FIRST"	<p>Specifies output behavior of the port being written to:</p> <ul style="list-style-type: none"> • WRITE_FIRST - Written value appears on output port of the RAM. • READ_FIRST - Previous RAM contents for that memory location appear on the output port. • NO_CHANGE - Previous value on the output port remains the same.

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;

-- RAMB16BWER: 16k+2k Parity Paramatizable, byte-wide enable BlockRAM, output registers
--           Spartan-3A DSP
-- Xilinx HDL Libraries Guide, version 12.2

RAMB16BWER_inst : RAMB16BWER
generic map (

```

[illegible]

```

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 (
  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,   -- 14-bit A port address input
  ADDRb => ADDRb,   -- 14-bit B port address 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 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 output register enable input
  REGCEB => REGCEB, -- 1-bit B port output register enable input
  RSTA => RSTA,     -- 1-bit A port reset input
  RSTB => RSTB,     -- 1-bit B port reset input
  WEA => WEA,      -- 4-bit A port Write Enable Input
  WEB => WEB       -- 4-bit B port Write Enable Input
);

-- End of RAMB16BWER_inst instantiation

```

Verilog Instantiation Template

```

// RAMB16BWER: 16k+2k Parity Paramatizable, byte-wide enable BlockRAM, output registers
//      Spartan-3A DSP
// Xilinx HDL Libraries Guide, version 12.2

RAMB16BWER #(
  .DATA_WIDTH_A(0), // Valid values are 0, 1, 2, 4, 9, 18, or 36
  .DATA_WIDTH_B(0), // Valid values are 0, 1, 2, 4, 9, 18, or 36
  .DOA_REG(0), // Specifies to enable=1/disable=0 port A output registers
  .DOB_REG(0), // Specifies to enable=1/disable=0 port B output registers
  .INIT_A(36'h000000000), // Initial values on A output port
  .INIT_B(36'h000000000), // Initial values on B output port
  .RSTTYPE("SYNC"), // Specifies reset type to be "SYNC" or "ASYNc"
  .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"

  // The following 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),

```

Spartan-3A and Spartan-3A DSP Libraries Guide for HDL Designs
UG613 (v 12.2) July 23, 2010 www.xilinx.com

```
.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 output register enable input
.REGCEB(REGCEB),  // 1-bit B port output register enable input
.RSTA(RSTA),      // 1-bit A port reset input
.RSTB(RSTB),      // 1-bit B port reset input
.WEA(WEA),        // 4-bit A port write enable input
.WEB(WEB)         // 4-bit B port write enable input
);

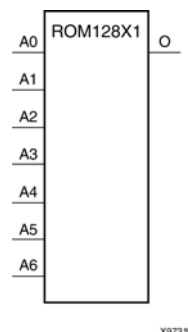
// End of RAMB16BWER_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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]
);

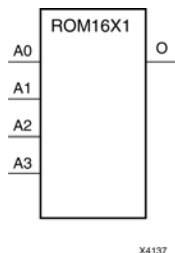
// End of ROM128X1_inst instantiation
```


For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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]
);

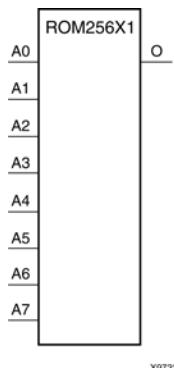
// End of ROM16X1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--          Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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

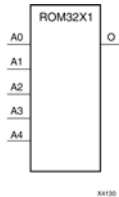
[illegible]

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

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
--          Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//          Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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]
);

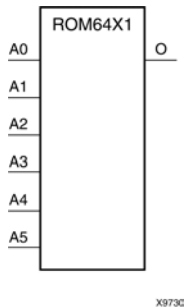
// End of ROM32X1_inst instantiation
```


For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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	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
--          Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//          Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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]
);

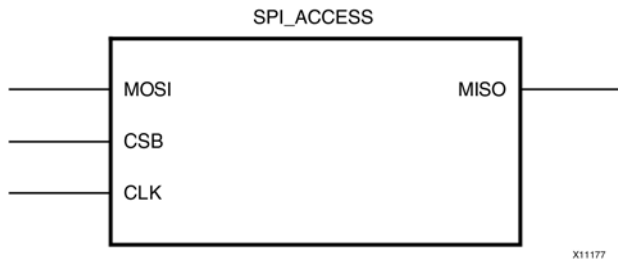
// End of ROM64X1_inst instantiation
```

For More Information

- See the [*Spartan-3 Generation FPGA User Guide*](#).
- See the [*Spartan-3A FPGA Family Data Sheet*](#).

SPI_ACCESS

Primitive: Internal Logic Access to the Serial Peripheral Interface (SPI) PROM Data



Introduction

This design element allows connection from the internal logic of the FPGA to an In-System Flash (ISF) Memory contained within the device through an SPI serial protocol. For information regarding the initialization of SPI_ACCESS model see the *Synthesis and Simulation Design Guide*

Port Descriptions

Port	Direction	Width	Function
MISO	Output	1	Serial output data from the ISF Memory.
MOSI	Input	1	Serial input instructions/data to the ISF Memory.
CSB	Input	1	ISF Memory enable.
CLK	Input	1	ISF Memory clock.

Design Entry Method

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

Available Attributes

Attribute	Type	Allowed Values	Default	Description
SIM_DEVICE	String	"3S50AN", "3S200AN", "3S400AN", "3S700AN", "3S1400AN"	"UNSPECIFIED"	Specifies the target device so that the proper size ISF Memory is used. This attribute <i>must</i> be set.
SIM_USER_ID	Hexadecimal	Any 64-Byte Value	All locations default to 0xFF	Specifies the programmed USER ID in the Security Register for the ISF Memory.
SIM_MEM_FILE	String	Specified file and directory name.	"NONE"	Optionally specifies a hex file containing the initialization memory content for the ISF Memory.

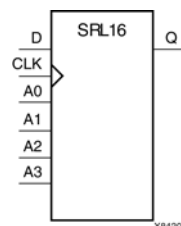
Attribute	Type	Allowed Values	Default	Description
SIM_FACTORY_ID	Hexadecimal	Any 64-Byte Value	All locations default to 0xFF	Specifies the Unique Identifier value in the Security Register for simulation purposes (the actual HW value will be specific to the particular device used).
SIM_DELAY_TYPE	String	"ACCURATE", "SCALED"	"SCALED"	<p>Scales down some timing delays for faster simulation run.</p> <ul style="list-style-type: none"> • ACCURATE - timing and delays consistent with datasheet specs. • SCALED - timing numbers scaled back to run faster simulation, behavior not affected.

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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
A _m	CLK	D	Q
A _m	X	X	Q(A _m)
A _m	↑	D	Q(A _m - 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 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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

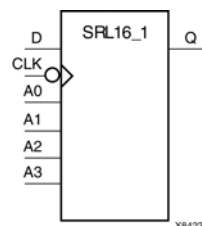
// End of SRL16_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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
A _m	CLK	D	Q
A _m	X	X	Q(A _m)
A _m	↓	D	Q(A _m - 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 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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

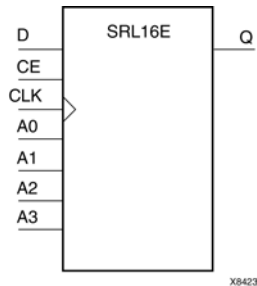
// End of SRL16_1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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 <ul style="list-style-type: none"> 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	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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

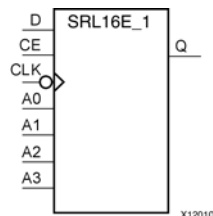
// End of SRL16E_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

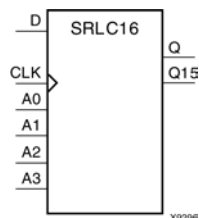
// End of SRL16E_1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

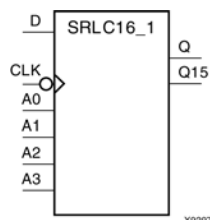
// End of SRLC16_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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	
A _m	CLK	D	Q	Q15
A _m	X	X	Q(A _m)	No Change
A _m	↓	D	Q(A _m - 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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

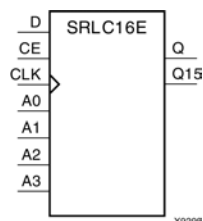
// End of SRLC16_1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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 cascable shift register LUT with clock enable operating on posedge of clock
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

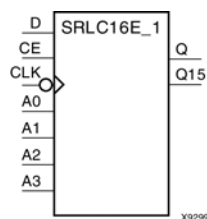
// End of SRLC16E_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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: $\text{Length} = (8 \times A3) + (4 \times A2) + (2 \times A1) + A0 + 1$. 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	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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//           Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

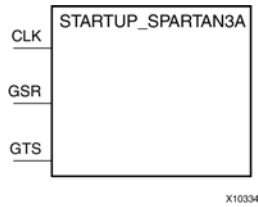
// End of SRLC16E_1_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

STARTUP_SPARTAN3A

Primitive: Spartan®-3A Global Set/Reset, Global 3-State and Configuration Start-Up Clock Interface



Introduction

This design element is used to either interface device pins and logic to the Global Set/Reset (GSR) signal, or for Global Tristate (GTS) dedicated routing. This primitive can also be used to specify a different clock for the device startup sequence at the end of configuring the device.

Port Descriptions

Port	Direction	Width	Function
GSR	Input	1	Input connection to the global set / reset (GSR) routing.
GTS	Input	1	Input connection to the global 3-state (GTS) routing.
CLK	Input	1	Input connection to the configuration startup sequence clock (GSR) routing.

Design Entry Method

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

To use the dedicated GSR circuitry, connect the sourcing pin or logic to the GSR pin. However, avoid using the GSR circuitry of this component unless certain precautions are taken first. Since the skew of the GSR net cannot be guaranteed, either use general routing for the set/reset signal in which routing delays and skew can be calculated as a part of the timing analysis of the design or to take preventative measures to ensure that possible skew on the release of the clock cycle won't interfere with circuit operation.

Similarly, if the dedicated global 3-state is used, connect the appropriate sourcing pin or logic to the GTS input pin of the primitive. In order to specify a clock for the startup sequence of configuration, connect a clock from the design to the CLK pin of this design 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;

-- STARTUP_SPARTAN3A: Startup primitive for GSR, GTS or startup sequence
--                      control.
--                      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

STARTUP_SPARTAN3A_inst : STARTUP_SPARTAN3A
port map (
    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)
);

-- End of STARTUP_SPARTAN3A_inst instantiation
```

Verilog Instantiation Template

```
// STARTUP_SPARTAN3A: Startup primitive for GSR, GTS or startup sequence
//                      control.
//                      Spartan-3A
// Xilinx HDL Libraries Guide, version 12.2

STARTUP_SPARTAN3A STARTUP_SPARTAN3A_inst (
    .CLK(CLK),           // Clock input for start-up sequence
    .GSR(GSR_PORT),     // Global Set/Reset input (GSR can not be used as a port name)
    .GTS(GTS_PORT)      // Global 3-state input (GTS can not be used as a port name)
);

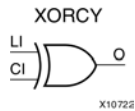
// End of STARTUP_SPARTAN3A_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--      Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

XORCY XORCY_inst (
    .O(O),    // XOR output signal
    .CI(CI),  // Carry input signal
    .LI(LI)   // LUT4 input signal
);

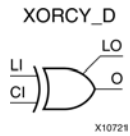
// End of XORCY_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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
--           Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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
);

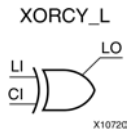
// End of XORCY_D_inst instantiation
```

For More Information

- See the [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).

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 ouput
--          Spartan-3A
-- Xilinx HDL Libraries Guide, version 12.2

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
//      Spartan-3
// Xilinx HDL Libraries Guide, version 12.2

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 [Spartan-3 Generation FPGA User Guide](#).
- See the [Spartan-3A FPGA Family Data Sheet](#).