

Virtex-5 Libraries Guide for HDL Designs

UG621 (v 12.3) September 21, 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.
- Descriptions of each available macro.
- 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 valid design elements 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.

Design elements are divided into three main categories:

- **Retargeted Elements** - These elements are automatically changed by the ISE software tools when they are used in this architecture. Retargeting ensures that your design takes advantage of the latest circuit design advances.
- **Macros** - These elements are in the UniMacro library in the Xilinx tool, and are used to instantiate primitives that are complex to instantiate by just using the primitives. The synthesis tools will automatically expand the unimacros to their underlying primitives.
- **Primitives** - Xilinx components that are native to the FPGA you are targeting. If you instantiate a primitive in your design, after the translation process (ngdbuild) 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.

CORE Generator 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. In addition to a comprehensive Unified Library containing all design elements, this guide is one in a series of architecture-specific libraries.

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 About Unimacros | 13 |
| BRAM_SDP_MACRO | 14 |
| BRAM_SINGLE_MACRO | 23 |
| BRAM_TDP_MACRO | 33 |
| FIFO_DUALCLOCK_MACRO | 43 |
| FIFO_SYNC_MACRO | 47 |
| Chapter 3 Functional Categories | 51 |
| Chapter 4 About Design Elements | 57 |
| BSCAN_VIRTEX5 | 58 |
| BUFCF | 60 |
| BUFG | 62 |
| BUFGCE | 64 |
| BUFGCTRL | 66 |
| BUFGMUX_CTRL | 69 |
| BUFIO | 71 |
| BUFR | 73 |
| CAPTURE_VIRTEX5 | 75 |
| CARRY4 | 77 |
| CFGLUT5 | 79 |
| CRC32 | 82 |
| CRC64 | 84 |
| DCIRESET | 86 |
| DCM_ADV | 88 |
| DSP48E | 95 |
| FDCE | 103 |
| FDCPE | 105 |
| FDRSE | 108 |
| FIFO18 | 110 |
| FIFO18_36 | 114 |
| FIFO36 | 118 |

| | |
|------------------------|-----|
| FIFO36_72 | 122 |
| FRAME_ECC_VIRTEX5..... | 126 |
| GTP_DUAL | 128 |
| GTX_DUAL | 129 |
| IBUF | 130 |
| IBUFDS..... | 132 |
| IBUFG | 134 |
| IBUFGDS..... | 136 |
| ICAP_VIRTEX5 | 138 |
| IDDR..... | 140 |
| IDDR_2CLK | 143 |
| IDELAY | 146 |
| IDELAYCTRL | 149 |
| IOBUF | 151 |
| IOBUFDS..... | 153 |
| IODELAY | 155 |
| ISERDES_NODELAY | 159 |
| KEEPER | 163 |
| KEY_CLEAR..... | 165 |
| LDCPE | 167 |
| LUT5 | 170 |
| LUT5_D..... | 174 |
| LUT5_L | 178 |
| LUT6 | 182 |
| LUT6_2..... | 186 |
| LUT6_D..... | 190 |
| LUT6_L | 194 |
| MUXF7..... | 198 |
| MUXF7_D | 200 |
| MUXF7_L..... | 202 |
| MUXF8..... | 204 |
| MUXF8_D | 206 |
| MUXF8_L..... | 208 |
| OBUF | 210 |
| OBUFDS | 212 |
| OBUFT | 214 |
| OBUFTDS..... | 216 |
| ODDR..... | 218 |

| | |
|-------------------------|-----|
| OSERDES | 221 |
| PLL_ADV..... | 226 |
| PLL_BASE..... | 234 |
| PPC440 | 238 |
| PULLDOWN | 239 |
| PULLUP | 241 |
| RAM128X1D..... | 243 |
| RAM16X1D_1..... | 246 |
| RAM16X1S_1..... | 248 |
| RAM16X2S..... | 250 |
| RAM16X4S..... | 252 |
| RAM16X8S..... | 254 |
| RAM256X1S..... | 256 |
| RAM32M | 258 |
| RAM32X1D..... | 262 |
| RAM32X1S_1..... | 265 |
| RAM32X2S..... | 267 |
| RAM32X4S..... | 270 |
| RAM32X8S..... | 272 |
| RAM64M | 274 |
| RAM64X1D..... | 278 |
| RAM64X1S..... | 281 |
| RAM64X1S_1..... | 283 |
| RAM64X2S..... | 285 |
| RAMB18..... | 287 |
| RAMB18SDP | 294 |
| RAMB36..... | 300 |
| RAMB36SDP | 311 |
| SRL16 | 319 |
| SRL16_1 | 321 |
| SRL16E..... | 323 |
| SRL16E_1 | 326 |
| SRLC32E | 328 |
| STARTUP_VIRTEX5 | 331 |
| SYSMON | 334 |
| TEMAC..... | 340 |
| USR_ACCESS_VIRTEX5..... | 341 |

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® Design Suite 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 |
| BUFGMUX | BUFGMUX_CTRL |
| BUFGMUX_1 | BUFGMUX_CTRL + INV |
| BUFGMUX_VIRTEX4 | BUFGMUX_CTRL |
| BUFGP | BUFG |
| DCM_BASE | DCM_ADV |
| DCM_PS | DCM_ADV |
| DSP48 | DSP48E |
| 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 |
| FIFO16 | FIFO18 |
| ISERDES | ISERDES_NODELAY |
| JTAGPPC | JTAG_PPC440 |
| LD | LDCPE |
| LD_1 | LDCPE + INV |
| LDC | LDCPE |
| LDC_1 | LDCPE + INV |
| LDCE | LDCPE |
| LDCE_1 | LDCPE + INV |
| LDCP | LDCPE |
| LDCP_1 | LDCPE + INV |
| LDE | LDCPE |
| LDE_1 | LDCPE + INV |
| LDP | LDCPE |
| LDP_1 | LDCPE + INV |
| LDPE | LDCPE |
| LDPE_1 | LDCPE + INV |
| LUT1 | LUT5 |
| LUT1_L | LUT5_L |
| LUT1_D | LUT5_D |
| LUT2 | LUT5 |
| LUT2_L | LUT5_L |
| LUT2_D | LUT5_D |
| LUT3 | LUT5 |
| LUT3_L | LUT5_L |
| LUT3_D | LUT56_D |
| LUT4 | LUT5 |
| LUT4_L | LUT5_L |
| LUT4_D | LUT5_D |
| MULT_AND | LUT6 |
| MULT18X18 | DSP48E |
| MULT18X18S | DSP48E |
| MUXCY | CARRY4 |

| Original Element | Modern Equivalent |
|------------------|--------------------|
| MUXCY_D | CARRY4 |
| MUXCY_L | CARRY4 |
| MUXF5 | LUT5 |
| MUXF5_D | LUT5_D |
| MUXF5_L | LUT5_L |
| MUXF6 | LUT6 |
| MUXF6_D | LUT6_D |
| MUXF6_L | LUT6_L |
| PMCD | PLL_ADV |
| RAM16X1D | RAM64X1D |
| RAM16X1S | RAM64X1S |
| RAM32X1S | RAM64X1S |
| RAMB16 | RAMB18 |
| RAMB16BWE | RAMB18 |
| ROM128X1 | 2 LUT6'S + MUXF7 |
| ROM16X1 | LUT5 |
| ROM256X1 | 4 LUT6'S + MUXF6/7 |
| ROM32X1 | LUT5 |
| ROM64X1 | LUT6 |
| SRLC16 | SRLC32E |
| SRLC16_1 | SRLC32E + INV |
| SRLC16E | SRLC32E |
| SRLC16E_1 | SRLC32E + INV |
| XORCY | CARRY4 |
| XORCY_D | CARRY4 |
| XORCY_L | CARRY4 |

About Unimacros

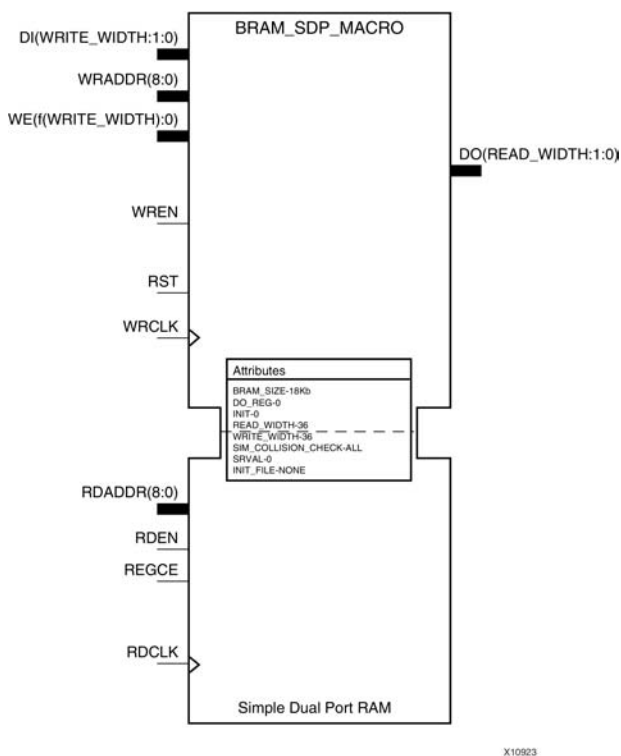
This section describes the unimacros that can be used with this architecture. The unimacros are organized alphabetically.

The following information is provided for each unimacro, where applicable:

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

BRAM_SDP_MACRO

Macro: Simple Dual Port RAM



X10923

Introduction

FPGA devices contain several block RAM memories that can be configured as general-purpose 36Kb or 18Kb RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. Both read and write operations are fully synchronous to the supplied clock(s) of the component. However, READ and WRITE ports can operate fully independently and asynchronously to each other, accessing the same memory array. Byte-enable write operations are possible, and an optional output register can be used to reduce the clock-to-out times of the RAM.

Note This element, must be configured so that read and write ports have the same width.

Port Description

| Name | Direction | Width (Bits) | Function |
|-------------------|-----------|-------------------------|--------------------------------------|
| Output Ports | | | |
| DO | Output | See Configuration Table | Data output bus addressed by RDADDR. |
| Input Ports | | | |
| DI | Input | See Configuration Table | Data input bus addressed by WRADDR. |
| WRADDR, RDADDR | Input | See Configuration Table | Write/Read address input buses. |
| WE | Input | See Configuration Table | Byte-Wide Write enable. |

| Name | Direction | Width (Bits) | Function |
|-----------------|-----------|--------------|---|
| WREN, RDEN | Input | 1 | Write/Read enable |
| SSR | Input | 1 | Output registers synchronous reset. |
| REGCE | Input | 1 | Output register clock enable input (valid only when DO_REG=1) |
| WRCLK, RDCLK | Input | 1 | Write/Read clock input. |

Configuration Table

| DATA_WIDTH | BRAM_SIZE | ADDR | WE |
|------------|-----------|------|----|
| 72 - 37 | 36Kb | 9 | 8 |
| 36 - 19 | 36Kb | 10 | 4 |
| | 18Kb | 9 | |
| 18 - 10 | 36Kb | 11 | 2 |
| | 18Kb | 10 | |
| 9 - 5 | 36Kb | 12 | 1 |
| | 18Kb | 11 | |
| 4 - 3 | 36Kb | 13 | 1 |
| | 18Kb | 12 | |
| 2 | 36Kb | 14 | 1 |
| | 18Kb | 13 | |
| 1 | 36Kb | 15 | 1 |
| | 18Kb | 14 | |

Design Entry Method

This unimacro can be instantiated only. It is a parameterizable version of the primitive. Consult the Configuration Table above to correctly configure it to meet your design needs.

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|--------|----------------|---------|--|
| BRAM_SIZE | String | 36Kb, 18Kb | 18Kb | Configures RAM as 36Kb or 18Kb memory. |

| Attribute | Type | Allowed Values | Default | Description |
|----------------------------|-------------|---|------------|---|
| DO_REG | Integer | 0, 1 | 0 | A value of 1 enables to the output registers to the RAM enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle but will have slower clock to out timing. |
| INIT | Hexadecimal | Any 72-Bit Value | All zeros | Specifies the initial value on the output after configuration. |
| READ_WIDTH, WRITE_WIDTH | Integer | 1-72 | 36 | Specifies size of DI/DO bus. READ_WIDTH and WRITE_WIDTH must be equal. |
| INIT_FILE | String | 0 bit string | NONE | Name of the file containing initial values. |
| SIM_COLLISION_CHECK | String | ALL, "WARNING_ONLY", "GENERATE_X_ONLY", "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> |
| SIM_MODE | String | "SAFE" or "FAST" . | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| SRVAL | Hexadecimal | Any 72-Bit Value | All zeroes | Specifies the output value of on the DO port upon the assertion of the synchronous reset (RST) signal. |
| INIT_00 to INIT_7F | Hexadecimal | Any 256-Bit Value | All zeroes | Allows specification of the initial contents of the 16Kb or 32Kb data memory array. |
| INITP_00 to INITP_0F | Hexadecimal | Any 256-Bit Value | All zeroes | Allows specification of the initial contents of the 2Kb or 4Kb parity data memory array. |

VHDL Instantiation Template

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

[illegible]

[illegible]

```

INIT_72 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_73 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_74 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_75 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_76 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_77 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_78 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_79 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7F => 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",

-- The next set of INIT_xx are valid when configured as 36Kb
INITP_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0F => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
DO => DO,          -- Output read data port
DI => DI,          -- Input write data port
RDADDR => RDADDR,  -- Input read address
RDCLK => RDCLK,    -- Input read clock
RDEN => RDEN,      -- Input read port enable
REGCE => REGCE,    -- Input read output register enable
RST => RST,        -- Input reset
WE => WE,          -- Input write enable
WRADDR => WRADDR,  -- Input write address
WRCLK => WRCLK,    -- Input write clock
WREN => WREN       -- Input write port enable
);
-- End of BRAM_SDP_MACRO_inst instantiation

```

Verilog Instantiation Template

```

// BRAM_SDP_MACRO: Simple Dual Port RAM
//                      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

BRAM_SDP_MACRO #(
    .BRAM_SIZE("18Kb"), // Target BRAM, "18Kb" or "36Kb"
    .DEVICE("VIRTEX5"), // Target device: "VIRTEX5", "VIRTEX6", "SPARTAN6"
    .WRITE_WIDTH(0),    // Valid values are 1-72 (37-72 only valid when BRAM_SIZE="36Kb")
    .READ_WIDTH(0),     // Valid values are 1-72 (37-72 only valid when BRAM_SIZE="36Kb")
    .DO_REG(0),         // Optional output register (0 or 1)
    .INIT_FILE("NONE"),
    .SIM_COLLISION_CHECK("ALL"), // Collision check enable "ALL", "WARNING_ONLY",
                                // "GENERATE_X_ONLY" or "NONE"
    .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
    .SRVAL(72'h00000000000000000000), // Set/Reset value for port output
    .INIT(72'h00000000000000000000), // Initial values on output port
    .INIT_00(256'h0000000000000000000000000000000000000000000000000000000000000000),
    .INIT_01(256'h0000000000000000000000000000000000000000000000000000000000000000),
    .INIT_02(256'h0000000000000000000000000000000000000000000000000000000000000000),
    .INIT_03(256'h0000000000000000000000000000000000000000000000000000000000000000),

```

```
// The next set of INIT_xx are valid when configured as 36Kb
```

[illegible]

21

```
) BRAM_SDP_MACRO_inst (
  .DO(DO),           // Output read data port
  .DI(DI),           // Input write data port
  .RDADDR(RDADDR),   // Input read address
  .RDCLK(RDCLK),      // Input read clock
  .RDEN(RDEN),        // Input read port enable
  .REGCE(REGCE),      // Input read output register enable
  .RST(RST),          // Input reset
  .WE(WE),            // Input write enable
  .WRADDR(WRADDR),    // Input write address
  .WRCLK(WRCLK),      // Input write clock
  .WREN(WREN)         // Input write port enable
);

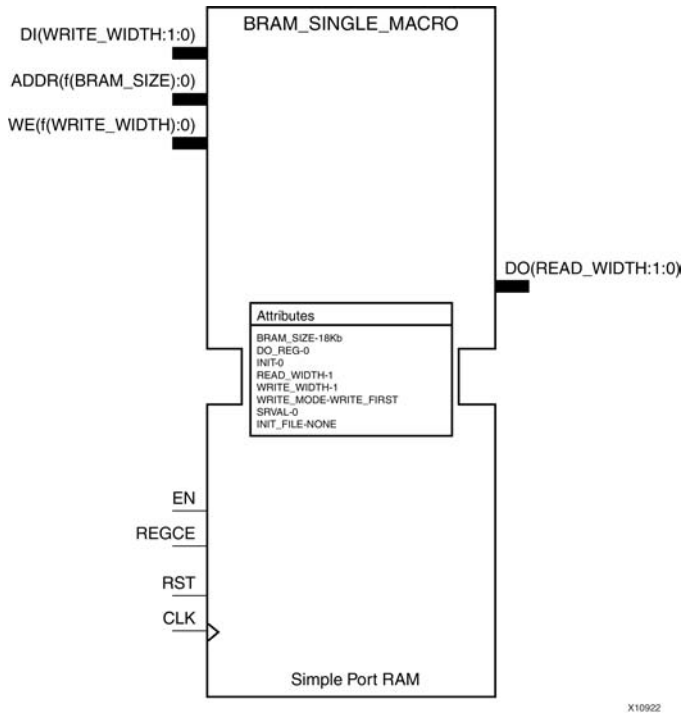
// End of BRAM_SDP_MACRO_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BRAM_SINGLE_MACRO

Macro: Single Port RAM



Introduction

FPGA devices contain several block RAM memories that can be configured as general-purpose 36Kb or 18Kb RAM/ROM memories. These single-port, block RAM memories offer fast and flexible storage of large amounts of on-chip data. Byte-enable write operations are possible, and an optional output register can be used to reduce the clock-to-out times of the RAM.

Port Description

| Name | Direction | Width | Function |
|--------------|-----------|--------------------------------|---|
| Output Ports | | | |
| DO | Output | See Configuration Table below. | Data output bus addressed by ADDR. |
| Input Ports | | | |
| DI | Input | See Configuration Table below. | Data input bus addressed by ADDR. |
| ADDR | Input | See Configuration Table below. | Address input bus. |
| WE | Input | See Configuration Table below. | Byte-Wide Write enable. |
| EN | Input | 1 | Write/Read enables. |
| RST | Input | 1 | Output registers synchronous reset. |
| REGCE | Input | 1 | Output register clock enable input (valid only when DO_REG=1) |
| CLK | Input | 1 | Clock input. |

Configuration Table

| WRITE_WIDTH | READ_WIDTH | BRAM_SIZE | ADDR | WE |
|-------------|------------|-----------|------|----|
| 37 - 72 | 37 - 72 | 36Kb | 9 | 8 |
| | 36 - 19 | | 10 | |
| | 18 - 10 | | 11 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 36 - 19 | 36 - 19 | 36Kb | 10 | 4 |
| | 18-10 | | 11 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 18 - 10 | 36 - 19 | 36Kb | 11 | 2 |
| | 18-10 | | 11 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 9 - 5 | 36-19 | 36Kb | 12 | 1 |
| | 18-10 | | 12 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 4 - 3 | 36-19 | 36Kb | 13 | 1 |
| | 18-10 | | 13 | |
| | 9 - 5 | | 13 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 2 | 36-19 | 36Kb | 14 | 1 |
| | 18-10 | | 14 | |
| | 9 - 5 | | 14 | |
| | 4 - 3 | | 14 | |
| | 2 | | 14 | |
| | 1 | | 15 | |

| WRITE_WIDTH | READ_WIDTH | BRAM_SIZE | ADDR | WE |
|-------------|------------|-----------|------|----|
| 1 | 36 - 19 | 36Kb | 15 | 1 |
| | 18 - 10 | | 15 | |
| | 9 - 5 | | 15 | |
| | 3 - 4 | | 15 | |
| | 2 | | 15 | |
| | 1 | | 15 | |
| 18-10 | 18-10 | 18Kb | 10 | 2 |
| | 9 - 5 | | 11 | |
| | 4 - 3 | | 12 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 9 - 5 | 18-10 | 18Kb | 11 | 1 |
| | 9 - 5 | | 11 | |
| | 4 - 3 | | 12 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 4 - 3 | 18-10 | 18Kb | 12 | 1 |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 12 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 2 | 18-10 | 18Kb | 13 | 1 |
| | 9 - 5 | | 13 | |
| | 4 - 3 | | 13 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 1 | 18-10 | 18Kb | 14 | 1 |
| | 9 - 5 | | 14 | |
| | 4 - 3 | | 14 | |
| | 2 | | 14 | |
| | 1 | | 14 | |

Design Entry Method

This unimacro can be instantiated only. It is a parameterizable version of the primitive. Consult the above Configuration Table in correctly configuring this element to meet your design needs.

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------|-------------|--|-------------|--|
| BRAM_SIZE | String | 36Kb, 18Kb | 18Kb | Configures RAM as 36Kb or 18Kb memory. |
| DO_REG | Integer | 0, 1 | 0 | A value of 1 enables to the output registers to the RAM enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle but will have slower clock to out timing. |
| READ_WIDTH | Integer | 1 - 36 | 1 | Specifies size of output bus. |
| WRITE_WIDTH | Integer | 1 - 36 | 1 | Specifies size of input bus. |
| INIT_FILE | String | 0 bit string | NONE | Name of the file containing initial values. |
| WRITE_MODE | String | READ_FIRST, WRITE_FIRST, NO_CHANGE | WRITE_FIRST | Specifies write mode to the memory |
| INIT | Hexadecimal | Any 72-Bit Value | All zeros | Specifies the initial value on the output after configuration. |
| SRVAL | Hexadecimal | Any 72-Bit Value | All zeroes | Specifies the output value of on the DO port upon the assertion of the synchronous reset (RST) signal. |
| SIM_MODE | String | "SAFE", "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| INIT_00 to INIT_FF | Hexadecimal | Any 256-Bit Value | All zeroes | Allows specification of the initial contents of the 16Kb or 32Kb data memory array. |
| INITP_00 to INITP_0F | Hexadecimal | Any 256-Bit Value | All zeroes | Allows specification of the initial contents of the 2Kb or 4Kb parity data memory array. |

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;
```

```
-- BRAM_SINGLE_MACRO: Single Port RAM
--                      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
-- Note - This Unimacro model assumes the port directions to be "downto".
```

-- Simulation of this model with "to" in the port directions could lead to erroneous results.

```
BRAM_SINGLE_MACRO_inst : BRAM_SINGLE_MACRO
generic map (
    BRAM_SIZE => "18Kb", -- Target BRAM, "18Kb" or "36Kb"
    DEVICE => "VIRTEX5", -- Target Device: "VIRTEX5", "VIRTEX6", "SPARTAN6"
    DO_REG => 0, -- Optional output register (0 or 1)
    INIT_A => X"00000000", -- Initial values on output port
    INIT_FILE => "NONE",
    WRITE_WIDTH => 0, -- Valid values are 1-72 (37-72 only valid when BRAM_SIZE="36Kb")
    READ_WIDTH => 0, -- Valid values are 1-72 (37-72 only valid when BRAM_SIZE="36Kb")
    SIM_MODE => "SAFE", -- Simulation: "SAFE" vs "FAST",
    -- see "Synthesis and Simulation Design Guide" for details
    SRVAL => X"00000000", -- Set/Reset value for port output
    WRITE_MODE => "WRITE_FIRST", -- "WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"
    -- The following INIT_xx declarations specify the initial contents of the RAM
    INIT_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_07 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0F => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_10 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_11 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_12 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_13 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_14 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_15 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_16 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_17 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_18 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_19 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1A => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1B => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1C => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1D => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1E => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1F => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_20 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_21 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_22 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_23 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_24 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_25 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_26 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_27 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_28 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_29 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_2A => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_2B => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_2C => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_2D => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_2E => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_2F => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_30 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_31 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_32 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_33 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_34 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_35 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_36 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_37 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_38 => X"0000000000000000000000000000000000000000000000000000000000000000",
```

UG621 (v 12.3) September 21, 2010

```

-- 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",

-- The next set of INIT_xx are valid when configured as 36Kb
INITP_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0F => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
  DO => DO,      -- Output data
  ADDR => ADDR,   -- Input address
  CLK => CLK,     -- Input clock
  DI => DI,       -- Input data port
  EN => EN,       -- Input RAM enable
  REGCE => REGCE, -- Input output register enable
  RST => RST,     -- Input reset
  WE => WE        -- Input write enable
);

-- End of BRAM_SINGLE_MACRO_inst instantiation

```

Verilog Instantiation Template

```

// BRAM_SINGLE_MACRO: Single Port RAM
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

BRAM_SINGLE_MACRO #(
  .BRAM_SIZE("18Kb"), // Target BRAM, "18Kb" or "36Kb"
  .DEVICE("VIRTEX5"), // Target Device: "VIRTEX5", "VIRTEX6", "SPARTAN6"
  .DO_REG(0), // Optional output register (0 or 1)
  .INIT(36'h000000000), // Initial values on output port
  .INIT_FILE ("NONE"),
  .WRITE_WIDTH(0), // Valid values are 1-72 (37-72 only valid when BRAM_SIZE="36Kb")
  .READ_WIDTH(0), // Valid values are 1-72 (37-72 only valid when BRAM_SIZE="36Kb")
  .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
  .SRVAL(36'h000000000), // Set/Reset value for port output
  .WRITE_MODE("WRITE_FIRST"), // "WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"
  .INIT_00(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_01(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_02(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_03(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_04(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_05(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_06(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_07(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_08(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_09(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_0A(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_0B(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_0C(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_0D(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_0E(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_0F(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_10(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_11(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_12(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_13(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_14(256'h0000000000000000000000000000000000000000000000000000000000000000),

```

```
// The next set of INIT_xx are valid when configured as 36Kb
```

[illegible]

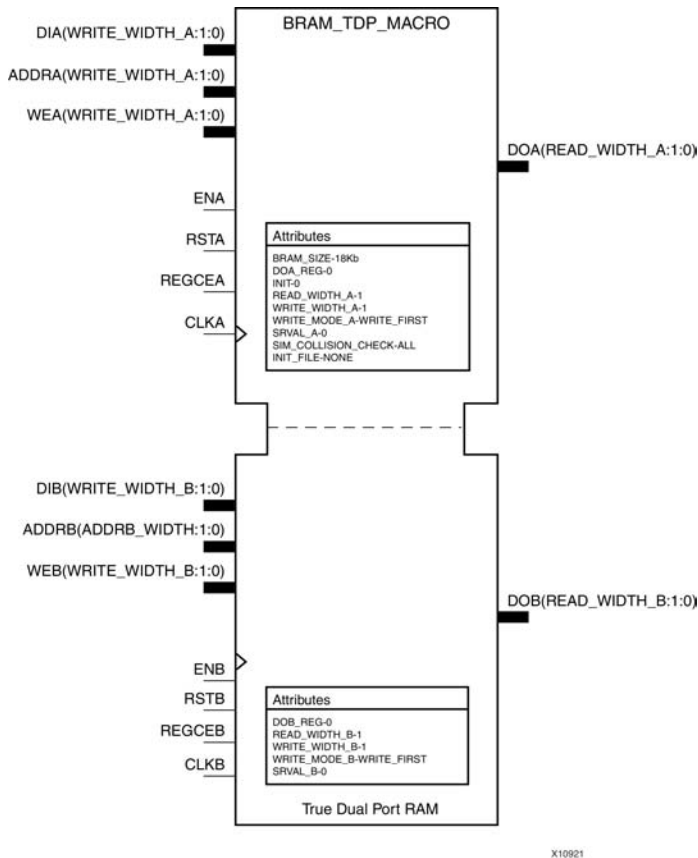
31

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BRAM_TDP_MACRO

Macro: True Dual Port RAM



Introduction

FPGA devices contain several block RAM memories that can be configured as general-purpose 36kb or 18kb RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. Both read and write operations are fully synchronous to the supplied clock(s) of the component. However, READ and WRITE ports can operate fully independently and asynchronous to each other, accessing the same memory array. Byte-enable write operations are possible, and an optional output register can be used to reduce the clock-to-out times of the RAM.

Port Description

| Name | Direction | Width | Function |
|--------------|-----------|--------------------------------|--------------------------------------|
| Output Ports | | | |
| DOA | Output | See Configuration Table below. | Data output bus addressed by ADDRA. |
| DOB | Output | See Configuration Table below. | Data output bus addressed by ADDRDB. |
| Input Ports | | | |
| DIA | Input | See Configuration Table below. | Data input bus addressed by ADDRA. |

| Name | Direction | Width | Function |
|----------------|-----------|--------------------------------|---|
| DIB | Input | See Configuration Table below. | Data input bus addressed by ADDRb. |
| ADDRA, ADDRb | Input | See Configuration Table below. | Address input buses for Port A, B. |
| WEA, WEB | Input | See Configuration Table below. | Write enable for Port A, B. |
| ENA, ENB | Input | 1 | Write/Read enables for Port A, B. |
| RSTA, RSTB | Input | 1 | Output registers synchronous reset for Port A, B. |
| REGCEA, REGCEB | Input | 1 | Output register clock enable input for Port A, B (valid only when DO_REG=1) |
| CLKA, CLKB | Input | 1 | Write/Read clock input for Port A, B. |

Configuration Table

| WRITE_WIDTH_A/B-DIA/DIB | READ_WIDTH_A/B-DOA/DOB | BRAM_SIZE | ADDRA/B | WEA/B |
|-------------------------|------------------------|-----------|---------|-------|
| 36 - 19 | 36 - 19 | 36Kb | 10 | 4 |
| | 18-10 | | 11 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 18 - 10 | 36 - 19 | 36Kb | 11 | 2 |
| | 18-10 | | 11 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 9 - 5 | 36-19 | 36Kb | 12 | 1 |
| | 18-10 | | 12 | |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 4 - 3 | 36-19 | 36Kb | 13 | 1 |
| | 18-10 | | 13 | |
| | 9 - 5 | | 13 | |
| | 4 - 3 | | 13 | |
| | 2 | | 14 | |
| | 1 | | 15 | |

| WRITE_WIDTH_A/B-DIA/DIB | READ_WIDTH_A/B-DOA/DOB | BRAM_SIZE | ADDRA/B | WEA/B |
|-------------------------|------------------------|-----------|---------|-------|
| 2 | 36-19 | 36Kb | 14 | 1 |
| | 18-10 | | 14 | |
| | 9 - 5 | | 14 | |
| | 4 - 3 | | 14 | |
| | 2 | | 14 | |
| | 1 | | 15 | |
| 1 | 36-19 | 36Kb | 15 | 1 |
| | 18-10 | | 15 | |
| | 9 - 5 | | 15 | |
| | 4 - 3 | | 15 | |
| | 2 | | 15 | |
| | 1 | | 15 | |
| 18-10 | 18-10 | 18Kb | 10 | 2 |
| | 9 - 5 | | 11 | |
| | 4 - 3 | | 12 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 9 - 5 | 18-10 | 18Kb | 11 | 1 |
| | 9 - 5 | | 11 | |
| | 4 - 3 | | 12 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 4 - 3 | 18-10 | 18Kb | 12 | 1 |
| | 9 - 5 | | 12 | |
| | 4 - 3 | | 12 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 2 | 18-10 | 18Kb | 13 | 1 |
| | 9 - 5 | | 13 | |
| | 4 - 3 | | 13 | |
| | 2 | | 13 | |
| | 1 | | 14 | |
| 1 | 18-10 | 18Kb | 14 | 1 |
| | 9 - 5 | | 14 | |
| | 4 - 3 | | 14 | |
| | 2 | | 14 | |
| | 1 | | 14 | |

Design Entry Method

This unimacro can be instantiated only. It is a parameterizable version of the primitive. Consult the Configuration Table above to correctly configure it to meet your design needs.

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

Available Attributes

| Attribute(s) | Type | Allowed Values | Default | Description |
|-------------------------|--------------|---|------------|---|
| BRAM_SIZE | String | 36Kb, 18Kb | 18Kb | Configures RAM as 36Kb or 18Kb memory. |
| DO_REG | Integer | 0, 1 | 0 | A value of 1 enables to the output registers to the RAM enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle but will have slower clock to out timing. |
| INIT | Hexa-decimal | Any 72-Bit Value | All zeros | Specifies the initial value on the output after configuration. |
| INIT_FILE | String | 0 bit string | NONE | Name of file containing initial values. |
| READ_WIDTH, WRITE_WIDTH | Integer | 1 - 72 | 36 | Specifies size of DI/DO bus. READ_WIDTH and WRITE_WIDTH must be equal. |
| SIM_COLLISION_CHECK | String | ALL, "WARNING_ONLY", "GENERATE_X_ONLY", "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> |
| SIM_MODE | String | "SAFE", "FAST" . | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| SRVAL A, SRVAL_B | Hexa-decimal | Any 72-Bit Value | All zeroes | Specifies the output value of on the DO port upon the assertion of the synchronous reset (RST) signal. |

| Attribute(s) | Type | Allowed Values | Default | Description |
|----------------------|--------------|-------------------|------------|--|
| INIT_00 to INIT_FF | Hexa-decimal | Any 256-Bit Value | All zeroes | Allows specification of the initial contents of the 16Kb or 32Kb data memory array. |
| INITP_00 to INITP_0F | Hexa-decimal | Any 256-Bit Value | All zeroes | Allows specification of the initial contents of the 2Kb or 4Kb parity data memory array. |

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;

-- BRAM_TDP_MACRO: True Dual Port RAM
--               Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

-- Note - This Unimacro model assumes the port directions to be "downto".
--        Simulation of this model with "to" in the port directions could lead to erroneous results.

BRAM_TDP_MACRO_inst : BRAM_TDP_MACRO
generic map (
  BRAM_SIZE => "18Kb", -- Target BRAM, "18Kb" or "36Kb"
  DEVICE => "VIRTEX5", -- Target Device: "VIRTEX5", "VIRTEX6", "SPARTAN6"
  DOA_REG => 0, -- Optional port A output register (0 or 1)
  DOB_REG => 0, -- Optional port B output register (0 or 1)
  INIT_A => X"000000000", -- Initial values on A output port
  INIT_B => X"000000000", -- Initial values on B output port
  INIT_FILE => "NONE",
  READ_WIDTH_A => 0, -- Valid values are 1-36 (19-36 only valid when BRAM_SIZE="36Kb")
  READ_WIDTH_B => 0, -- Valid values are 1-36 (19-36 only valid when BRAM_SIZE="36Kb")
  SIM_COLLISION_CHECK => "ALL", -- Collision check enable "ALL", "WARNING_ONLY",
  -- "GENERATE_X_ONLY" or "NONE"
  SIM_MODE => "SAFE", -- Simulation: "SAFE" vs "FAST",
  -- see "Synthesis and Simulation Design Guide" for details
  SRVAL_A => X"000000000", -- Set/Reset value for A port output
  SRVAL_B => X"000000000", -- Set/Reset value for B port output
  WRITE_MODE_A => "WRITE_FIRST", -- "WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"
  WRITE_MODE_B => "WRITE_FIRST", -- "WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"
  WRITE_WIDTH_A => 0, -- Valid values are 1, 2, 4, 9, 18 or 36 (36 only valid when BRAM_SIZE="36Kb")
  WRITE_WIDTH_B => 0, -- Valid values are 1, 2, 4, 9, 18 or 36 (36 only valid when BRAM_SIZE="36Kb")
  -- The following INIT_xx declarations specify the initial contents of the RAM
  INIT_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_07 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_0F => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_10 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_11 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_12 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_13 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_14 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_15 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_16 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_17 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_18 => X"0000000000000000000000000000000000000000000000000000000000000000",
  INIT_19 => X"0000000000000000000000000000000000000000000000000000000000000000",

```

```
-- The next set of INIT xx are valid when configured as 36Kb
```

[illegible]

```

INIT_61 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_62 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_63 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_64 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_65 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_66 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_67 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_68 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_69 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6F => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_70 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_71 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_72 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_73 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_74 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_75 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_76 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_77 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_78 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_79 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7F => 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",

-- The next set of INIT_xx are valid when configured as 36Kb
INITP_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0F => X"0000000000000000000000000000000000000000000000000000000000000000")

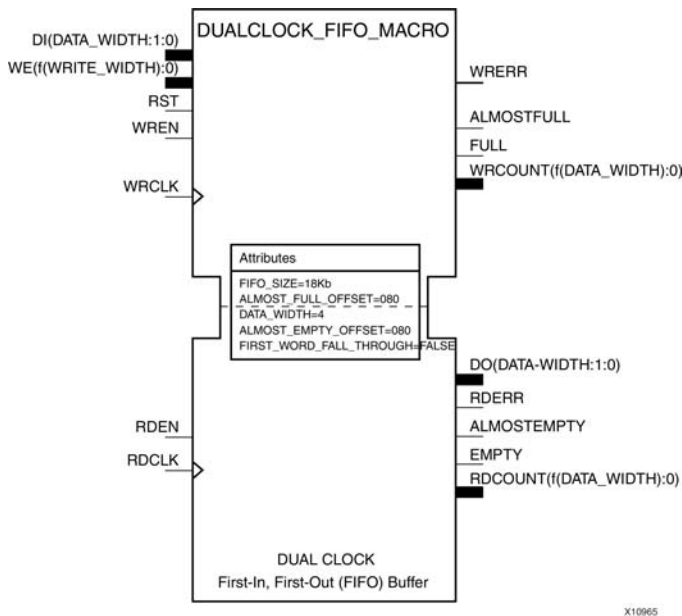
port map (
  DOA => DOA,      -- Output port-A data
  DOB => DOB,      -- Output port-B data
  ADDRA => ADDRA,   -- Input port-A address
  ADDR8 => ADDR8,   -- Input port-B address
  CLKA => CLKA,     -- Input port-A clock
  CLKB => CLKB,     -- Input port-B clock
  DIA => DIA,      -- Input port-A data
  DIB => DIB,      -- Input port-B data
  ENA => ENA,      -- Input port-A enable
  ENB => ENB,      -- Input port-B enable
  REGCEA => REGCEA, -- Input port-A output register enable
  REGCEB => REGCEB, -- Input port-B output register enable
  RSTA => RSTA,     -- Input port-A reset
  RSTB => RSTB,     -- Input port-B reset
  WEA => WEA,      -- Input port-A write enable
  WEB => WEB       -- Input port-B write enable
);

-- End of BRAM_TDP_MACRO_inst instantiation

```


FIFO_DUALCLOCK_MACRO

Macro: Dual Clock First-In, First-Out (FIFO) RAM Buffer



Introduction

FPGA devices contain several block RAM memories that can be configured as general-purpose 36kb or 18kb RAM/ROM memories. Dedicated logic in the block RAM enables you to easily implement FIFOs. The FIFO can be configured as an 18 kb or 36 kb memory. This unimacro configures the FIFO for using independent read and writes clocks. Data is read from the FIFO on the rising edge of read clock and written to the FIFO on the rising edge of write clock.

Depending on the offset between read and write clock edges, the Empty, Almost Empty, Full and Almost Full flags can deassert one cycle later. Due to the asynchronous nature of the clocks the simulation model only reflects the deassertion latency cycles listed in the User Guide.

Port Description

| Name | Direction | Width | Function |
|--------------|-----------|--------------------------------|---|
| Output Ports | | | |
| ALMOSTEMPTY | Output | 1 | Almost all valid entries in FIFO have been read. |
| ALMOSTFULL | Output | 1 | Almost all entries in FIFO memory have been filled. |
| DO | Output | See Configuration Table below. | Data output bus addressed by ADDR. |
| EMPTY | Output | 1 | FIFO is empty. |
| FULL | Output | 1 | All entries in FIFO memory are filled. |
| RDCOUNT | Output | See Configuration Table below. | FIFO data read pointer. |

| Name | Direction | Width | Function |
|-------------|-----------|--------------------------------|--|
| RDERR | Output | 1 | When the FIFO is empty, any additional read operation generates an error flag. |
| WRCOUNT | Output | See Configuration Table below. | FIFO data write pointer. |
| WRERR | Output | 1 | When the FIFO is full, any additional write operation generates an error flag. |
| Input Ports | | | |
| DI | Input | See Configuration Table below. | Data input bus addressed by ADDR. |
| RDCLK | Input | 1 | Clock for Read domain operation. |
| RDEN | Input | 1 | Read Enable |
| RST | Input | 1 | Asynchronous reset. |
| WRCLK | Input | 1 | Clock for Write domain operation. |
| WREN | Input | 1 | Write Enable |

Configuration Table

This unimacro can be instantiated only. The unimacro is a parameterizable version of the primitive. Please use the Configuration Table below to correctly configure the unimacro to meet design needs.

| DATA_WIDTH | FIFO_SIZE | WRCOUNT | RDCOUNT |
|------------|-----------|---------|---------|
| 72 - 37 | 36kb | 9 | 9 |
| 36 - 19 | 36kb | 10 | 10 |
| | 18kb | 9 | 9 |
| 18 - 10 | 36kb | 11 | 11 |
| | 18kb | 10 | 10 |
| 9-5 | 36kb | 12 | 12 |
| | 18kb | 11 | 11 |
| 1-4 | 36kb | 13 | 13 |
| | 18kb | 12 | 12 |

Design Entry Method

This unimacro can be instantiated only. It is a parameterizable version of the primitive. Consult the above Configuration Table to correctly configure this element to meet your design needs.

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------|-------------|------------------|-----------|--|
| ALMOST_EMPTY_OFFSET | Hexadecimal | 13-Bit Value | All zeros | Setting determines the difference between EMPTY and ALMOSTEMPTY conditions. Must be set using hexadecimal notation. |
| ALMOST_FULL_OFFSET | Hexadecimal | 13-Bit Value | All zeros | Setting determines the difference between FULL and ALMOSTFULL conditions. Must be set using hexadecimal notation. |
| DATA_WIDTH | Integer | 1 - 72 | 4 | Width of DI/DO bus. |
| FIFO_SIZE | String | 18kb, 36kb | 18kb | Configures FIFO as 18kb or 36kb memory. |
| FIRST_WORD_FALL_THROUGH | Boolean | FALSE, TRUE | FALSE | If TRUE, the first word written into the empty FIFO appears at the FIFO output without RDEN asserted. |
| SIM_MODE | String | "SAFE" or "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- FIFO_DUALCLOCK_MACRO: Dual-Clock First-In, First-Out (FIFO) RAM Buffer
--                               Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

-- Note - This Unimacro model assumes the port directions to be "downto".
-- Simulation of this model with "to" in the port directions could lead to erroneous results.

FIFO_DUALCLOCK_MACRO_inst : FIFO_DUALCLOCK_MACRO
generic map (
    DEVICE => "VIRTEX5",           -- Target Device: "VIRTEX5", "VIRTEX6"
    ALMOST_FULL_OFFSET => X"0080", -- Sets almost full threshold
    ALMOST_EMPTY_OFFSET => X"0080", -- Sets the almost empty threshold
    DATA_WIDTH => 0,              -- Valid values are 1-72 (37-72 only valid when FIFO_SIZE="36Kb")
    FIFO_SIZE => "18Kb",           -- Target BRAM, "18Kb" or "36Kb"
    FIRST_WORD_FALL_THROUGH => FALSE, -- Sets the FIFO FWFT to TRUE or FALSE
    SIM_MODE => "SAFE") -- Simulation "SAFE" vs "FAST",
                        -- see "Synthesis and Simulation Design Guide" for details
port map (
    ALMOSTEMPTY => ALMOSTEMPTY,    -- Output almost empty
    ALMOSTFULL => ALMOSTFULL,      -- Output almost full
    DO => DO,                      -- Output data
    EMPTY => EMPTY,               -- Output empty
    FULL => FULL,                 -- Output full
    RDCOUNT => RDCOUNT,           -- Output read count
    RDERR => RDERR,              -- Output read error
    WRCOUNT => WRCOUNT,          -- Output write count
    WRERR => WRERR,              -- Output write error
    DI => DI,                    -- Input data
    RDCLK => RDCLK,              -- Input read clock
    RDEN => RDEN,                -- Input read enable
    RST => RST,                  -- Input reset
    WRCLK => WRCLK,              -- Input write clock
    WREN => WREN                 -- Input write enable
);
-- End of FIFO_DUALCLOCK_MACRO_inst instantiation

```

Verilog Instantiation Template

```
// FIFO_DUALCLOCK_MACRO: Dual Clock First-In, First-Out (FIFO) RAM Buffer
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FIFO_DUALCLOCK_MACRO #(
    .ALMOST_EMPTY_OFFSET(9'h080), // Sets the almost empty threshold
    .ALMOST_FULL_OFFSET(9'h080),  // Sets almost full threshold
    .DATA_WIDTH(0),               // Valid values are 1-72 (37-72 only valid when FIFO_SIZE="36Kb")
    .DEVICE("VIRTEX5"),           // Target device: "VIRTEX5", "VIRTEX6"
    .FIFO_SIZE ("18Kb"),           // Target BRAM: "18Kb" or "36Kb"
    .FIRST_WORD_FALL_THROUGH ("FALSE"), // Sets the FIFO FWFT to "TRUE" or "FALSE"
    .SIM_MODE("SAFE") // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
) FIFO_DUALCLOCK_MACRO_inst (
    .ALMOSTEMPTY(ALMOSTEMPTY), // Output almost empty
    .ALMOSTFULL(ALMOSTFULL),   // Output almost full
    .DO(DO),                   // Output data
    .EMPTY(EMPTY),             // Output empty
    .FULL(FULL),               // Output full
    .RDCOUNT(RDCOUNT),         // Output read count
    .RDERR(RDERR),             // Output read error
    .WRCOUNT(WRCOUNT),         // Output write count
    .WRERR(WRERR),             // Output write error
    .DI(DI),                   // Input data
    .RDCLK(RDCLK),             // Input read clock
    .RDEN(RDEN),               // Input read enable
    .RST(RST),                 // Input reset
    .WRCLK(WRCLK),             // Input write clock
    .WREN(WREN)                // Input write enable
);

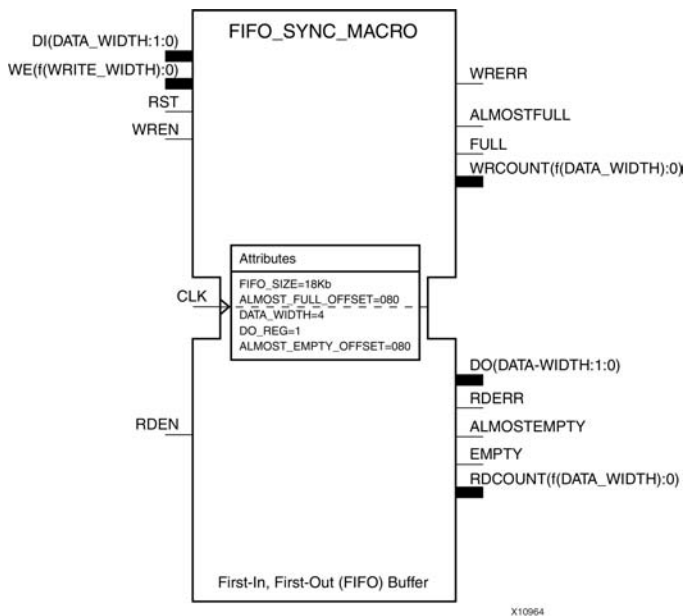
// End of FIFO_DUALCLOCK_MACRO_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FIFO_SYNC_MACRO

Macro: Synchronous First-In, First-Out (FIFO) RAM Buffer



Introduction

FPGA devices contain several block RAM memories that can be configured as general-purpose 36kb or 18kb RAM/ROM memories. Dedicated logic in the block RAM enables you to easily implement FIFOs. The FIFO can be configured as an 18 kb or 36 kb memory. This unimacro configures the FIFO such that it uses one clock for reading as well as writing.

Port Description

| Name | Direction | Width | Function |
|--------------|-----------|--------------------------------|--|
| Output Ports | | | |
| ALMOSTEMPTY | Output | 1 | Almost all valid entries in FIFO have been read. |
| ALMOSTFULL | Output | 1 | Almost all entries in FIFO memory have been filled. |
| DO | Output | See Configuration Table. | Data output bus addressed by ADDR. |
| EMPTY | Output | 1 | FIFO is empty. |
| FULL | Output | 1 | All entries in FIFO memory are filled. |
| RDCOUNT | Output | See Configuration Table below. | FIFO data read pointer. |
| RDERR | Output | 1 | When the FIFO is empty, any additional read operation generates an error flag. |
| WRCOUNT | Output | See Configuration Table. | FIFO data write pointer. |

| Name | Direction | Width | Function |
|-------------|-----------|--------------------------|--|
| WRERR | Output | 1 | When the FIFO is full, any additional write operation generates an error flag. |
| Input Ports | | | |
| CLK | Input | 1 | Clock for Read/Write domain operation. |
| DI | Input | See Configuration Table. | Data input bus addressed by ADDR. |
| RDEN | Input | 1 | Read Enable |
| RST | Input | 1 | Asynchronous reset. |
| WREN | Input | 1 | Write Enable |

Configuration Table

This unimacro can be instantiated only. The unimacro is a parameterizable version of the primitive. Please use the configuration table below to correctly configure the unimacro to meet design needs.

| DATA_WIDTH | FIFO_SIZE | WRCOUNT | RDCOUNT |
|------------|-----------|---------|---------|
| 72 - 37 | 36kb | 9 | 9 |
| 36 - 19 | 36kb | 10 | 10 |
| | 18kb | 9 | 9 |
| 18 - 10 | 36kb | 11 | 11 |
| | 18kb | 10 | 10 |
| 9-5 | 36kb | 12 | 12 |
| | 18kb | 11 | 11 |
| 1-4 | 36kb | 13 | 13 |
| | 18kb | 12 | 12 |

Design Entry Method

This unimacro can be instantiated only. It is a parameterizable version of the primitive. Consult the above Configuration Table to correctly configure this element to meet your design needs.

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|---------------------|-------------|------------------|-----------|--|
| ALMOST_EMPTY_OFFSET | Hexadecimal | Any 13-Bit Value | All zeros | Setting determines the difference between EMPTY and ALMOSTEMPTY conditions. Must be set using hexadecimal notation. |
| ALMOST_FULL_OFFSET | Hexadecimal | Any 13-Bit Value | All zeros | Setting determines the difference between FULL and ALMOSTFULL conditions. Must be set using hexadecimal notation. |
| DATA_WIDTH | Integer | 1 - 72 | 4 | Width of DI/DO bus. |
| DO_REG | Binary | 0,1 | 1 | DO_REG must be set to 0 for flags and data to follow a standard synchronous FIFO operation. When DO_REG is set to 1, effectively a pipeline register is added to the output of the synchronous FIFO. Data then has a one clock cycle latency. However, the clock-to-out timing is improved. |
| FIFO_SIZE | String | 18kb, 36kb | 18kb | Configures FIFO as 18kb or 36kb memory. |
| SIM_MODE | String | "SAFE" or "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- FIFO_SYNC_MACRO: Synchronous First-In, First-Out (FIFO) RAM Buffer
--               Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

-- Note - This Unimacro model assumes the port directions to be "downto".
--        Simulation of this model with "to" in the port directions could lead to erroneous results.

FIFO_SYNC_MACRO_inst : FIFO_SYNC_MACRO
generic map (
    DEVICE => "VIRTEX5",           -- Target Device: "VIRTEX5", "VIRTEX6"
    ALMOST_FULL_OFFSET => X"0080", -- Sets almost full threshold
    ALMOST_EMPTY_OFFSET => X"0080", -- Sets the almost empty threshold
    DATA_WIDTH => 0,             -- Valid values are 1-72 (37-72 only valid when FIFO_SIZE="36Kb")
    FIFO_SIZE => "18Kb",          -- Target BRAM, "18Kb" or "36Kb"
    SIM_MODE => "SAFE")           -- Simulation) "SAFE" vs "FAST",
                                -- see "Synthesis and Simulation Design Guide" for details
port map (
    ALMOSTEMPTY => ALMOSTEMPTY,    -- Output almost empty
    ALMOSTFULL => ALMOSTFULL,      -- Output almost full
    DO => DO,                      -- Output data
    EMPTY => EMPTY,               -- Output empty
    FULL => FULL,                 -- Output full
    RDCOUNT => RDCOUNT,           -- Output read count
    RDERR => RDERR,              -- Output read error
    WRCOUNT => WRCOUNT,          -- Output write count
    WRERR => WRERR,              -- Output write error
    CLK => CLK,                  -- Input clock
    DI => DI,                    -- Input data
    RDEN => RDEN,                -- Input read enable
    RST => RST,                  -- Input reset
    WREN => WREN                 -- Input write enable
);
-- End of FIFO_SYNC_MACRO_inst instantiation

```

Verilog Instantiation Template

```
// FIFO_SYNC_MACRO: Synchronous First-In, First-Out (FIFO) RAM Buffer
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FIFO_SYNC_MACRO #(
    .DEVICE("VIRTEX5"), // Target device: "VIRTEX5", "VIRTEX6"
    .ALMOST_EMPTY_OFFSET(9'h080), // Sets the almost empty threshold
    .ALMOST_FULL_OFFSET(9'h080), // Sets almost full threshold
    .DATA_WIDTH(0), // Valid values are 1-72 (37-72 only valid when FIFO_SIZE="36Kb")
    .DO_REG(0), // Optional output register (0 or 1)
    .FIFO_SIZE ("18Kb"), // Target BRAM: "18Kb" or "36Kb"
    .SIM_MODE("SAFE") // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
) FIFO_SYNC_MACRO_inst (
    .ALMOSTEMPTY(ALMOSTEMPTY), // Output almost empty
    .ALMOSTFULL(ALMOSTFULL), // Output almost full
    .DO(DO), // Output data
    .EMPTY(EMPTY), // Output empty
    .FULL(FULL), // Output full
    .RDCOUNT(RDCOUNT), // Output read count
    .RDERR(RDERR), // Output read error
    .WRCOUNT(WRCOUNT), // Output write count
    .WRERR(WRERR), // Output write error
    .CLK(CLK), // Input clock
    .DI(DI), // Input data
    .RDEN(RDEN), // Input read enable
    .RST(RST), // Input reset
    .WREN(WREN) // Input write enable
);

// End of FIFO_SYNC_MACRO_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

Functional Categories

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

| | | |
|-------------------------|--------------------|----------------------|
| Advanced | I/O Components | Slice/CLB Primitives |
| Arithmetic Functions | Processors | |
| Clock Components | RAM/ROM | |
| Config/BSCAN Components | Registers/Latches | |
| Gigabit I/O | Shift Register LUT | |

Advanced

| Design Element | Description |
|------------------------|--|
| CRC32 | Primitive: Cyclic Redundancy Check Calculator for 32 bits |
| CRC64 | Primitive: Cyclic Redundancy Check Calculator for 64 bits |
| SYSMON | Primitive: System Monitor |
| TEMAC | Primitive: Tri-mode Ethernet Media Access Controller (MAC) |

Arithmetic Functions

| Design Element | Description |
|------------------------|---|
| DSP48E | Primitive: 25x18 Two's Complement Multiplier with Integrated 48-Bit, 3-Input Adder/Subtractor/Accumulator or 2-Input Logic Unit |

Clock Components

| Design Element | Description |
|------------------------------|--|
| BUFG | Primitive: Global Clock Buffer |
| BUFGCE | Primitive: Global Clock Buffer with Clock Enable |
| BUFGCTRL | Primitive: Global Clock MUX Buffer |
| BUFGMUX_CTRL | Primitive: 2-to-1 Global Clock MUX Buffer |
| BUFIO | Primitive: Local Clock Buffer for I/O |
| BUFR | Primitive: Regional Clock Buffer for I/O and Logic Resources |
| DCM_ADV | Primitive: Advanced Digital Clock Manager Circuit |
| PLL_ADV | Primitive: Advanced Phase Locked Loop Clock Circuit |
| PLL_BASE | Primitive: Basic Phase Locked Loop Clock Circuit |

Config/BSCAN Components

| Design Element | Description |
|------------------------------------|---|
| BSCAN_VIRTEX5 | Primitive: Virtex®-5 JTAG Boundary-Scan Logic Access Circuit |
| CAPTURE_VIRTEX5 | Primitive: Virtex®-5 Readback Register Capture Control |
| FRAME_ECC_VIRTEX5 | Primitive: Virtex®-5 Configuration Frame Error Detection and Correction Circuitry |
| ICAP_VIRTEX5 | Primitive: Internal Configuration Access Port |
| KEY_CLEAR | Primitive: Virtex-5 Configuration Encryption Key Erase |
| STARTUP_VIRTEX5 | Primitive: Virtex®-5 Configuration Start-Up Sequence Interface |
| USR_ACCESS_VIRTEX5 | Primitive: Virtex-5 User Access Register |

Gigabit I/O

| Design Element | Description |
|--------------------------|-------------------------------------|
| GTP_DUAL | Primitive: Dual Gigabit Transceiver |
| GTX_DUAL | Primitive: Dual Gigabit Transceiver |

I/O Components

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

Processors

| Design Element | Description |
|----------------|----------------------------------|
| PPC440 | Primitive: Power PC 440 CPU Core |

RAM/ROM

| Design Element | Description |
|----------------|---|
| FIFO18 | Primitive: 18kb FIFO (First In, First Out) Block RAM Memory |
| FIFO18_36 | Primitive: 36-bit Wide by 512 Deep 18kb FIFO (First In, First Out) Block RAM Memory |
| FIFO36 | Primitive: 36kb FIFO (First In, First Out) Block RAM Memory |

| Design Element | Description |
|----------------------------|---|
| FIFO36_72 | Primitive: 72-Bit Wide by 512 Deep 36kb FIFO (First In, First Out) Block RAM Memory with ECC (Error Detection and Correction Circuitry) |
| RAM128X1D | Primitive: 128-Deep by 1-Wide Dual Port Random Access Memory (Select RAM) |
| RAM16X1D_1 | Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM with Negative-Edge Clock |
| RAM16X1S_1 | Primitive: 16-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock |
| RAM16X2S | Primitive: 16-Deep by 2-Wide Static Synchronous RAM |
| RAM16X4S | Primitive: 16-Deep by 4-Wide Static Synchronous RAM |
| RAM16X8S | Primitive: 16-Deep by 8-Wide Static Synchronous RAM |
| RAM256X1S | Primitive: 256-Deep by 1-Wide Random Access Memory (Select RAM) |
| RAM32M | Primitive: 32-Deep by 8-bit Wide Multi Port Random Access Memory (Select RAM) |
| RAM32X1D | Primitive: 32-Deep by 1-Wide Static Dual Port Synchronous RAM |
| RAM32X1S_1 | Primitive: 32-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock |
| RAM32X2S | Primitive: 32-Deep by 2-Wide Static Synchronous RAM |
| RAM32X4S | Primitive: 32-Deep by 4-Wide Static Synchronous RAM |
| RAM32X8S | Primitive: 32-Deep by 8-Wide Static Synchronous RAM |
| RAM64M | Primitive: 64-Deep by 4-bit Wide Multi Port Random Access Memory (Select RAM) |
| RAM64X1D | Primitive: 64-Deep by 1-Wide Dual Port Static Synchronous RAM |
| RAM64X1S | Primitive: 64-Deep by 1-Wide Static Synchronous RAM |
| RAM64X1S_1 | Primitive: 64-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock |
| RAM64X2S | Primitive: 64-Deep by 2-Wide Static Synchronous RAM |
| RAMB18 | Primitive: 18K-bit Configurable Synchronous True Dual Port Block RAM |
| RAMB18SDP | Primitive: 36-bit by 512 Deep, 18kb Synchronous Simple Dual Port Block RAM |
| RAMB36 | Primitive: 36kb Configurable Synchronous True Dual Port Block RAM |
| RAMB36SDP | Primitive: 72-bit by 512 Deep, 36kb Synchronous Simple Dual Port Block RAM with ECC (Error Correction Circuitry) |

Registers/Latches

| Design Element | Description |
|---------------------------|--|
| FDCE | Primitive: D Flip-Flop with Clock Enable and Asynchronous Clear |
| FDCPE | Primitive: D Flip-Flop with Clock Enable and Asynchronous Preset and Clear |
| FDRSE | Primitive: D Flip-Flop with Synchronous Reset and Set and Clock Enable |
| IDDR | Primitive: Input Dual Data-Rate Register |
| IDDR_2CLK | Primitive: Input Dual Data-Rate Register with Dual Clock Inputs |
| LDCPE | Primitive: Transparent Data Latch with Asynchronous Clear and Preset and Gate Enable |
| ODDR | Primitive: Dedicated Dual Data Rate (DDR) Output Register |

Shift Register LUT

| Design Element | Description |
|--------------------------|---|
| SRL16 | Primitive: 16-Bit Shift Register Look-Up Table (LUT) |
| SRL16_1 | Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Negative-Edge Clock |
| SRL16E | Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Clock Enable |
| SRL16E_1 | Primitive: 16-Bit Shift Register Look-Up Table (LUT) with Negative-Edge Clock and Clock Enable |
| SRLC32E | Primitive: 32 Clock Cycle, Variable Length Shift Register Look-Up Table (LUT) with Clock Enable |

Slice/CLB Primitives

| Design Element | Description |
|----------------|---|
| BUFCF | Primitive: Fast Connect Buffer |
| CARRY4 | Primitive: Fast Carry Logic with Look Ahead |
| CFGLUT5 | Primitive: 5-input Dynamically Reconfigurable Look-Up Table (LUT) |
| LUT5 | Primitive: 5-Input Lookup Table with General Output |
| LUT5_D | Primitive: 5-Input Lookup Table with General and Local Outputs |
| LUT5_L | Primitive: 5-Input Lookup Table with Local Output |
| LUT6 | Primitive: 6-Input Lookup Table with General Output |
| LUT6_2 | Primitive: Six-input, 2-output, Look-Up Table |
| LUT6_D | Primitive: 6-Input Lookup Table with General and Local Outputs |
| LUT6_L | Primitive: 6-Input Lookup Table 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 |

About Design Elements

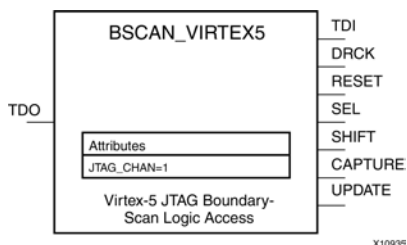
This section describes the design elements that can be used with this architecture. The design elements are organized alphabetically.

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_VIRTEX5

Primitive: Virtex®-5 JTAG Boundary-Scan Logic Access Circuit



Introduction

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

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

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

Port Descriptions

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

Design Entry Method

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

Available Attributes

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

VHDL Instantiation Template

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

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

-- BSCAN_VIRTEX5: Boundary Scan primitive for connecting internal logic to
--                JTAG interface.
--                Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

BSCAN_VIRTEX5_inst : BSCAN_VIRTEX5
generic map (
    JTAG_CHAIN => 1) -- Value for USER command. Possible values: (1,2,3 or 4)
port map (
    CAPTURE => CAPTURE, -- CAPTURE output from TAP controller
    DRCK => DRCK,       -- Data register output for USER functions
    RESET => RESET,     -- Reset output from TAP controller
    SEL => SEL,         -- USER active output
    SHIFT => SHIFT,     -- SHIFT output from TAP controller
    TDI => TDI,         -- TDI output from TAP controller
    UPDATE => UPDATE,   -- UPDATE output from TAP controller
    TDO => TDO         -- Data input for USER function
);

-- End of BSCAN_VIRTEX5_inst instantiation
```

Verilog Instantiation Template

```
// BSCAN_VIRTEX5: Boundary Scan primitive for connecting internal
//                logic to JTAG interface.
//                Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

BSCAN_VIRTEX5 #(
    .JTAG_CHAIN(1) // Value for USER command. Possible values: (1,2,3 or 4)
) BSCAN_VIRTEX5_inst (
    .CAPTURE(CAPTURE), // CAPTURE output from TAP controller
    .DRCK(DRCK),       // Data register output for USER function
    .RESET(RESET),     // Reset output from TAP controller
    .SEL(SEL),         // USER active output
    .SHIFT(SHIFT),     // SHIFT output from TAP controller
    .TDI(TDI),         // TDI output from TAP controller
    .UPDATE(UPDATE),   // UPDATE output from TAP controller
    .TDO(TDO)         // Data input for USER function
);

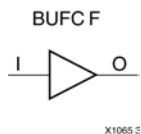
// End of BSCAN_VIRTEX5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BUFCF

Primitive: Fast Connect Buffer



Introduction

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

Design Entry Method

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

VHDL Instantiation Template

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

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

-- BUFCF: Fast connect buffer used to connect the outputs of the LUTs
--       and some dedicated logic directly to the input of another LUT.
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of BUFCF_inst instantiation
```

Verilog Instantiation Template

```
// BUFCF: Fast connect buffer used to connect the outputs of the LUTs
//       and some dedicated logic directly to the input of another LUT.
//       Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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

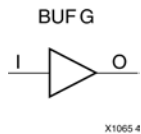
// End of BUFCF_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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 input |
| O | Output | 1 | Clock buffer output |

Design Entry Method

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

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- BUFG: Global Clock Buffer (source by an internal signal)
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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)
//      All FPGAs
// Xilinx HDL Libraries Guide, version 12.3

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

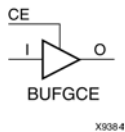
// End of BUFG_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BUFGCE

Primitive: Global Clock Buffer with Clock Enable



Introduction

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

Logic Table

| Inputs | | Outputs |
|--------|----|---------|
| I | CE | O |
| X | 0 | 0 |
| I | 1 | I |

Port Descriptions

| Port | Type | Width | Function |
|------|--------|-------|---------------------|
| I | Input | 1 | Clock buffer input |
| CE | Input | 1 | Clock enable input |
| O | Output | 1 | Clock buffer output |

Design Entry Method

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

VHDL Instantiation Template

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

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

-- BUFGCE: Global Clock Buffer with Clock Enable (active high)
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of BUFGCE_inst instantiation
```

Verilog Instantiation Template

```
// BUFGCE: Global Clock Buffer with Clock Enable (active high)
//      Virtex-5/6, Spartan-3/3E/3A/6
// Xilinx HDL Libraries Guide, version 12.3

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

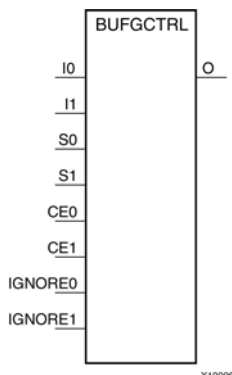
// End of BUFGCE_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BUFGCTRL

Primitive: Global Clock MUX Buffer



Introduction

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

Port Descriptions

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

Design Entry Method

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

Available Attributes

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

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

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- BUFGCTRL: Advanced Clock Primitive
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of BUFGCTRL_inst instantiation

```

Verilog Instantiation Template

```
// BUFGCTRL: Advanced Clock MUX Primitive
//          Virtex-5/6
// Xilinx HDL Libraries Guide, version 12.3

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

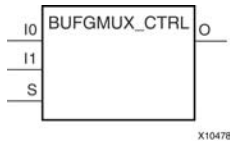
// End of BUFGCTRL_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BUFGMUX_CTRL

Primitive: 2-to-1 Global Clock MUX Buffer



Introduction

This design element is a global clock buffer with two clock inputs, one clock output, and a select line used to clearly select between one of two clocks driving the global clocking resource. This component is based on BUFGCTRL, with some pins connected to logic High or Low. This element uses the S pin as the select pin for the 2-to-1 MUX. S can switch anytime without causing a glitch on the output clock of the buffer.

Port Descriptions

| Port | Direction | Width | Function |
|------|-----------|-------|--|
| O | Output | 1 | Clock Output |
| I0 | Input | 1 | One of two Clock Inputs |
| I1 | Input | 1 | One of two Clock Inputs |
| S | Input | 1 | Select for I0 (S=0) or I1 (S=1) Clock Output |

Design Entry Method

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

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- BUFGMUX_CTRL: Global Clock Buffer 2-to-1 MUX
--               Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of BUFGMUX_CTRL_inst instantiation

```

Verilog Instantiation Template

```
// BUFGMUX_CTRL: Global Clock Buffer 2-to-1 MUX
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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

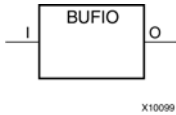
// End of BUFGMUX_CTRL_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BUFIO

Primitive: Local Clock Buffer for I/O



Introduction

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

Port Descriptions

| Port | Type | Width | Function |
|------|--------|-------|--------------|
| O | Output | 1 | Clock output |
| I | Input | 1 | Clock input |

Design Entry Method

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

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- BUFIO: Clock in, clock out buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of BUFIO_inst instantiation

```

Verilog Instantiation Template

```
// BUFIO: Local Clock Buffer
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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

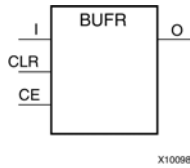
// End of BUFIO_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

BUFR

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



Introduction

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

Port Descriptions

| Port | Type | Width | Function |
|------|--------|-------|---|
| CE | Input | 1 | Clock enable port. When asserted Low, this port disables the output clock at port O. When asserted High, this port resets the counter used to produce the divided clock output. |
| CLR | Input | 1 | Counter reset for divided clock output. When asserted High, this port resets the counter used to produce the divided clock output. |
| I | Input | 1 | Clock input port. This port is the clock source port for BUFR. It can be driven by BUFIO output or local interconnect. |
| O | Output | 1 | Clock output port. This port drives the clock tracks in the clock region of the BUFR and the two adjacent clock regions. This port drives FPGA fabric, and IOBs. |

Design Entry Method

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

Available Attributes

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

VHDL Instantiation Template

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

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

-- BUFR: Regional (Local) Clock Buffer /w Enable, Clear and Division Capabilities
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of BUFR_inst instantiation
```

Verilog Instantiation Template

```
// BUFR: Regional Clock Buffer w/ Enable, Clear and Division Capabilities
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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

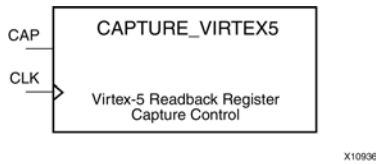
// End of BUFR_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

CAPTURE_VIRTEX5

Primitive: Virtex®-5 Readback Register Capture Control



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_VIRTEX5: Register State Capture for Bitstream Readback
--                               Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

CAPTURE_VIRTEX5_inst : CAPTURE_VIRTEX5
generic map (
    ONESHOT => TRUE) -- TRUE or FALSE
port map (
    CAP => CAP,      -- Capture input
    CLK => CLK       -- Clock input
);
-- End of CAPTURE_VIRTEX5_inst instantiation
```

Verilog Instantiation Template

```
// CAPTURE_VIRTEX5: Register State Capture for Bitstream Readback
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

CAPTURE_VIRTEX5 #(
    .ONESHOT("TRUE") // "TRUE" or "FALSE"
) CAPTURE_VIRTEX5_inst (
    .CAP(CAP),        // Capture input
    .CLK(CLK)         // Clock input
);

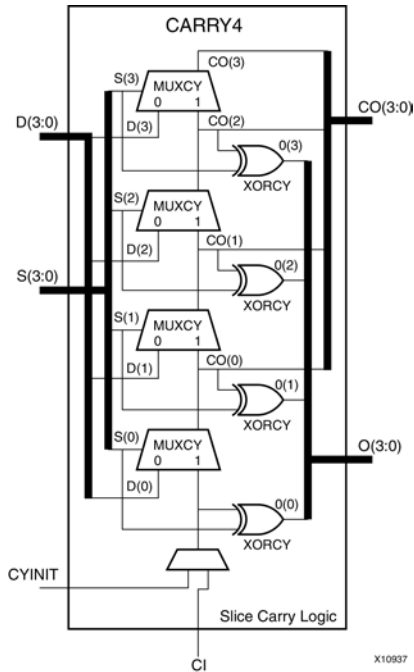
// End of CAPTURE_VIRTEX5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

CARRY4

Primitive: Fast Carry Logic with Look Ahead



Introduction

This circuit design represents the fast carry logic for a slice. The carry chain consists of a series of four MUXes and four XORs that connect to the other logic (LUTs) in the slice via dedicated routes to form more complex functions. The fast carry logic is useful for building arithmetic functions like adders, counters, subtractors and add/subs, as well as such other logic functions as wide comparators, address decoders, and some logic gates (specifically, AND and OR).

Port Descriptions

| Port | Direction | Width | Function |
|--------|-----------|-------|--|
| O | Output | 4 | Carry chain XOR general data out |
| CO | Output | 4 | Carry-out of each stage of the carry chain |
| DI | Input | 4 | Carry-MUX data input |
| S | Input | 4 | Carry-MUX select line |
| CYINIT | Input | 1 | Carry-in initialization input |
| CI | Input | 1 | Carry cascade 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;

-- CARRY4: Fast Carry Logic Component
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

CARRY4_inst : CARRY4
port map (
    CO => CO,          -- 4-bit carry out
    O => O,            -- 4-bit carry chain XOR data out
    CI => CI,          -- 1-bit carry cascade input
    CYINIT => CYINIT,  -- 1-bit carry initialization
    DI => DI,          -- 4-bit carry-MUX data in
    S => S             -- 4-bit carry-MUX select input
);

-- End of CARRY4_inst instantiation

```

Verilog Instantiation Template

```

// CARRY4: Fast Carry Logic Component
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

CARRY4 CARRY4_inst (
    .CO(CO),           // 4-bit carry out
    .O(O),             // 4-bit carry chain XOR data out
    .CI(CI),           // 1-bit carry cascade input
    .CYINIT(CYINIT),  // 1-bit carry initialization
    .DI(DI),           // 4-bit carry-MUX data in
    .S(S)              // 4-bit carry-MUX select input
);

// End of CARRY4_inst instantiation

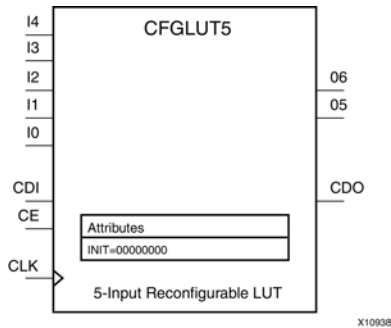
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

CFGLUT5

Primitive: 5-input Dynamically Reconfigurable Look-Up Table (LUT)



Introduction

This element is a runtime, dynamically reconfigurable, 5-input look-up table (LUT) that enables the changing of the logical function of the LUT during circuit operation. Using the CDI pin, a new INIT value can be synchronously shifted in serially to change the logical function. The O6 output pin produces the logical output function, based on the current INIT value loaded into the LUT and the currently selected I0-I4 input pins. Optionally, you can use the O5 output in combination with the O6 output to create two individual 4-input functions sharing the same inputs or a 5-input function and a 4-input function that uses a subset of the 5-input logic (see tables below). This component occupies one of the four 6-LUT components within a slice.

To cascade this element, connect the CDO pin from each element to the CDI input of the next element. This will allow a single serial chain of data (32-bits per LUT) to reconfigure multiple LUTs.

Port Descriptions

| Port | Direction | Width | Function |
|--------------------|-----------|-------|--|
| O6 | Output | 1 | 5-LUT output |
| O5 | Output | 1 | 4-LUT output |
| I0, I1, I2, I3, I4 | Input | 1 | LUT inputs |
| CDO | Output | 1 | Reconfiguration data cascaded output (optionally connect to the CDI input of a subsequent LUT) |
| CDI | Input | 1 | Reconfiguration data serial input |
| CLK | Input | 1 | Reconfiguration clock |
| CE | Input | 1 | Active high reconfiguration clock enable |

Design Entry Method

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

- Connect the CLK input to the clock source used to supply the reconfiguration data.
- Connect the CDI input to the source of the reconfiguration data.
- Connect the CE pin to the active high logic if you need to enable/disable LUT reconfiguration.
- Connect the I4-I0 pins to the source inputs to the logic equation. The logic function is output on O6 and O5.
- To cascade this element, connect the CDO pin from each element to the CDI input of the next element to allow a single serial chain of data to reconfigure multiple LUTs.

The INIT attribute should be placed on this design element to specify the initial logical function of the LUT. A new INIT can be loaded into the LUT any time during circuit operation by shifting in 32-bits per LUT in the chain, representing the new INIT value. Disregard the O6 and O5 output data until all 32-bits of new INIT data has been clocked into the LUT. The logical function of the LUT changes as new INIT data is shifted into it. Data should be shifted in MSB (INIT[31]) first and LSB (INIT[0]) last.

In order to understand the O6 and O5 logical value based on the current INIT, see the table below:

| I4 I3 I2 I1 I0 | O6 Value | O5 Value |
|----------------|----------|----------|
| 1 1 1 1 1 | INIT[31] | INIT[15] |
| 1 1 1 1 0 | INIT[30] | INIT[14] |
| ... | ... | ... |
| 1 0 0 0 1 | INIT[17] | INIT[1] |
| 1 0 0 0 0 | INIT[16] | INIT[0] |
| 0 1 1 1 1 | INIT[15] | INIT[15] |
| 0 1 1 1 0 | INIT[14] | INIT[14] |
| ... | ... | ... |
| 0 0 0 0 1 | INIT[1] | INIT[1] |
| 0 0 0 0 0 | INIT[0] | INIT[0] |

For instance, the INIT value of FFFF8000 would represent the following logical equations:

- O6 = I4 or (I3 and I2 and I1 and I0)
- O5 = I3 and I2 and I1 and I0

To use these elements as two, 4-input LUTs with the same inputs but different functions, tie the I4 signal to a logical one. The INIT[31:16] values apply to the logical values of the O6 output and INIT [15:0] apply to the logical values of the O5 output.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|---|
| INIT | Hexadecimal | Any 32-bit Value | All zeros | Specifies the initial logical expression of this 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;

-- CFGLUT5: Reconfigurable 5-input LUT
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

CFGLUT5_inst : CFGLUT5
generic map (
    INT => X"00000000")
port map (
    CDO => CDO, -- Reconfiguration cascade output
    O5 => O5,   -- 4-LUT output
    O6 => O6,   -- 5-LUT output
    CDI => CDI, -- Reconfiguration data input
    CE  => CE,  -- Reconfiguration enable input
    CLK => CLK, -- Clock input
    I0  => I0,  -- Logic data input
    I1  => I1,  -- Logic data input
    I2  => I2,  -- Logic data input
    I3  => I3,  -- Logic data input
    I4  => I4,  -- Logic data input
);

-- End of CFGLUT5_inst instantiation
```

Verilog Instantiation Template

```
// CFGLUT5: Reconfigurable 5-input LUT
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

CFGLUT5 #(
    .INIT(32'h00000000) // Specify initial LUT contents
) CFGLUT5_inst (
    .CDO(CDO), // Reconfiguration cascade output
    .O5(O5),  // 4-LUT output
    .O6(O6),  // 5-LUT output
    .CDI(CDI), // Reconfiguration data input
    .CE(CE),  // Reconfiguration enable input
    .CLK(CLK), // Clock input
    .I0(I0),  // Logic data input
    .I1(I1),  // Logic data input
    .I2(I2),  // Logic data input
    .I3(I3),  // Logic data input
    .I4(I4),  // Logic data input
);

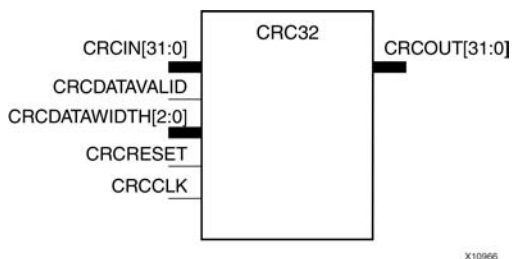
// End of CFGLUT5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

CRC32

Primitive: Cyclic Redundancy Check Calculator for 32 bits



Introduction

This design element is computed for the contents of a frame and appended to the end of the frame before transmission or storage. Each CRC block computes a 32-bit CRC using the CRC-32 polynomial specified for PCI EXPRESS®, Gigabit Ethernet, and other common protocols. The 32-bit CRC primitive, CRC32, can process 8, 16, 24 or 32-bit input data and generates a 32-bit CRC.

Port Descriptions

| Port | Direction | Width | Function | | |
|-------------------|-----------|-------|---|------------|-------------------|
| CRCIN[31:0] | Input | 32 | CRC input data, max datapath width is 4 bytes | | |
| CRCDATAVALID | Input | 1 | Indicates valid data on CRCIN inputs. | | |
| | | | 1'b1: data valid | | |
| | | | 1'b0: data invalid | | |
| | | | De-asserting this signal will cause the CRC value to be held for the number of cycles that the signal is de-asserted | | |
| CRCDATAWIDTH[2:0] | Input | 3 | Indicates how many input data bytes are valid. | | |
| | | | CRCDATAWIDTH[2:0] | Data Width | CRC Data Bus bits |
| | | | 0 | 8-bit | CRCIN[31:24] |
| | | | 1 | 16-bit | CRCIN[31:16] |
| | | | 10 | 24-bit | CRCIN[31:8] |
| | | | 11 | 32-bit | CRCIN[31:0] |
| CRCRESET | Input | 1 | Synchronous reset of CRC registers. When CRCRESET is asserted, the CRC block is initialized to the CRC_INIT value | | |
| CRCCLK | Input | 1 | CRC Clock | | |
| CRCOUT[31:0] | Output | 32 | 32-bit CRC output. CRCOUT is the byte-reversed, bit inverted CRC value corresponding to the CRC calculation on valid bytes from the previous clock cycle and the previous CRC value. Note that input CRCDATAVALID must be set to "1". | | |

Design Entry Method

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

Available Attributes

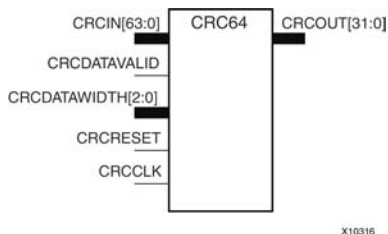
| Attribute | Type | Allowed values | Default | Description |
|----------------|-------------|------------------|------------|--|
| CRC_INIT[31:0] | Hexadecimal | Any 32-Bit Value | 0xFFFFFFFF | Sets the initial value of CRC internal registers. For LX30T & LX50T ES silicon the value is fixed as 0xFFFFFFFF. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

CRC64

Primitive: Cyclic Redundancy Check Calculator for 64 bits



Introduction

This design element is computed for the contents of a frame, and appended to the end of the frame before transmission or storage. Each CRC block computes a 32-bit CRC using the CRC-32 polynomial specified for PCI EXPRESS®, Gigabit Ethernet, and other common protocols. The 64-bit CRC primitive, CRC64, can process 8, 16, 24, 32, 40, 56 or 64-bit input data and generates a 32-bit CRC. Using the CRC64 primitive consumes both CRC hard blocks paired with a given transceiver tile.

Port Descriptions

| Port | Direction | Width | Function |
|-------------------|-----------|-------|---|
| CRCIN[63:0] | Input | 64 | CRC input data, max datapath width is 8 bytes |
| CRCDATAVALID | Input | 1 | Indicates valid data on CRCIN inputs. |
| | | | 1'b1: data valid |
| | | | 1'b0: data invalid |
| | | | De-asserting this signal will cause the CRC value to be held for the number of cycles that the signal is de-asserted. |
| CRCDATAWIDTH[2:0] | Input | 3 | Indicates how many input data bytes are valid. |
| | | | CRCDATAWIDTH [2:0] Data Width CRC Data Bus bits |
| | | | 0 8-bit CRCIN[63:56] |
| | | | 1 16-bit CRCIN[63:48] |
| | | | 10 24-bit CRCIN[63:40] |
| | | | 11 32-bit CRCIN[63:32] |
| | | | 100 40-bit CRCIN[63:24] |
| | | | 101 48-bit CRCIN[63:16] |
| | | | 110 56-bit CRCIN[63:8] |
| | | | 111 64-bit CRCIN[63:0] |
| CRCRESET | Input | 1 | Synchronous reset of CRC registers. When CRCRESET is asserted, the CRC block is initialized to the CRC_INIT value. |
| CRCCLK | Input | 1 | CRC Clock |
| CRCOUT[31:0] | Output | 32 | 32-bit CRC output. CRCOUT is the byte-reversed, bit inverted CRC value corresponding to the CRC calculation on valid bytes from the previous clock cycle and the previous CRC value. Note that CRCDATAVALID must be set to "1". |

Design Entry Method

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

Available Attributes

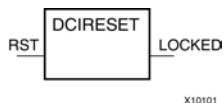
| Attribute | Type | Allowed values | Default | Description |
|----------------|-------------|------------------|------------|--|
| CRC_INIT[31:0] | Hexadecimal | Any 32-Bit Value | 0xFFFFFFFF | Sets the initial value of CRC internal registers. For LX30T & LX50T ES silicon the value is fixed as 0xFFFFFFFF. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

DCIRESET

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



Introduction

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

Port Descriptions

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

Design Entry Method

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

VHDL Instantiation Template

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

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

-- DCIRESET: DCI reset component
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of DCIRESET_inst instantiation
```

Verilog Instantiation Template

```
// DCIRESET: Digital Controlled Impedance (DCI) Reset Component
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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

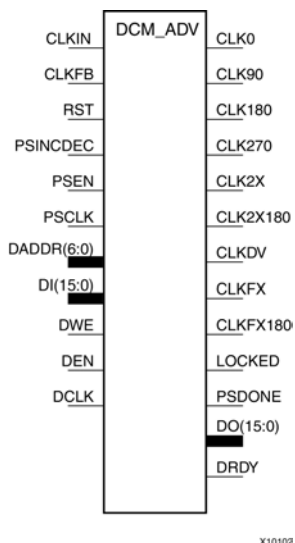
// End of DCIRESET_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

DCM_ADV

Primitive: Advanced Digital Clock Manager Circuit



Introduction

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

Port Descriptions

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

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

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

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

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|--------------------|---------|---|---------|---|
| CLK_FEEDBACK | String | "1X" , or "NONE" | "1X" | Specifies the clock feedback of the allowed value. |
| CLKDV_DIVIDE | Float | 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0, 16.0 | 2.0 | Specifies the extent to which the CLKDLL, CLKDLLE, CLKDLLHF, or DCM clock divider (CLKDV output) is to be frequency divided. |
| CLKFX_DIVIDE | Integer | 1 to 32 | 1 | Specifies the frequency divider value for the CLKFX output. |
| CLKFX_MULTIPLY | Integer | 2 to 32 | 4 | Specifies the frequency multiplier value for the CLKFX output. |
| CLKIN_DIVIDE_BY_2 | Boolean | FALSE, TRUE | FALSE | Allows for the input clock frequency to be divided in half when such a reduction is necessary to meet the DCM input clock frequency requirements. |
| CLKIN_PERIOD | Float | 1.25 to 1000.00 | 10.0 | Specifies period of input clock in ns from 1.25 to 1000.00. |
| CLKOUT_PHASE_SHIFT | String | "NONE", "FIXED", "VARIABLE_POSITIVE", "VARIABLE_CENTER" or "DIRECT" | "NONE" | Specifies the phase shift mode of allowed value. |

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

VHDL Instantiation Template

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

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

```
-- DCM_ADV: Digital Clock Manager Circuit
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
DCM_ADV_inst : DCM_ADV
generic map (
  CLKDV_DIVIDE => 2.0,  -- Divide by: 1.5,2.0,2.5,3.0,3.5,4.0,4.5,5.0,5.5,6.0,6.5
                        --      7.0,7.5,8.0,9.0,10.0,11.0,12.0,13.0,14.0,15.0 or 16.0
  CLKFX_DIVIDE => 1,    -- Can be any integer from 1 to 32
  CLKFX_MULTIPLY => 4,  -- Can be any integer from 2 to 32
  CLKIN_DIVIDE_BY_2 => FALSE,  -- TRUE/FALSE to enable CLKIN divide by two feature
  CLKIN_PERIOD => 10.0,  -- Specify period of input clock in ns from 1.25 to 1000.00
  CLKOUT_PHASE_SHIFT => "NONE",  -- Specify phase shift mode of NONE, FIXED,
                                --      VARIABLE_POSITIVE, VARIABLE_CENTER or DIRECT
  CLK_FEEDBACK => "1X",  -- Specify clock feedback of NONE or 1X
  DCM_PERFORMANCE_MODE => "MAX_SPEED",  -- Can be MAX_SPEED or MAX_RANGE
  DESKEW_ADJUST => "SYSTEM_SYNCHRONOUS",  -- SOURCE_SYNCHRONOUS, SYSTEM_SYNCHRONOUS or
                                --      an integer from 0 to 15
  DFS_FREQUENCY_MODE => "LOW",  -- HIGH or LOW frequency mode for frequency synthesis
  DLL_FREQUENCY_MODE => "LOW",  -- LOW, HIGH, or HIGH_SER frequency mode for DLL
  DUTY_CYCLE_CORRECTION => TRUE,  -- Duty cycle correction, TRUE or FALSE
  FACTORY_JF => X"F0F0",  -- FACTORY JF Values Suggested to be set to X"F0F0"
  PHASE_SHIFT => 0,  -- Amount of fixed phase shift from -255 to 1023
  SIM_DEVICE => "VIRTEX5",  -- Set target device, "VIRTEX4" or "VIRTEX5"
```

```

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

-- End of DCM_ADV_inst instantiation

```

Verilog Instantiation Template

```
// DCM_ADV: Digital Clock Manager Circuit
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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

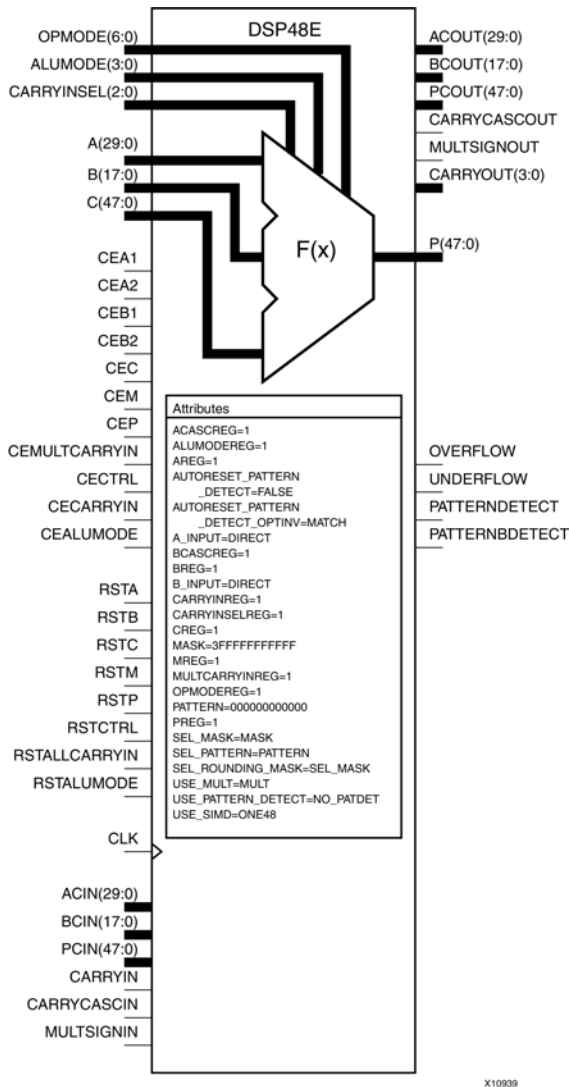
// End of DCM_ADV_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

DSP48E

Primitive: 25x18 Two's Complement Multiplier with Integrated 48-Bit, 3-Input Adder/Subtractor/Accumulator or 2-Input Logic Unit



Introduction

This design element 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. Some of the functions capable within the block include multiplication, addition, subtraction, accumulation, shifting, logical operations, and pattern detection.

Port Descriptions

| Port | Direction | Width | Function |
|------------|-----------|-------|--|
| Data Ports | | | |
| A | Input | 30 | 25-bit data input to multiplier or 30-bit MSB Data input to Adder/Logic Unit (LU). |

| Port | Direction | Width | Function |
|----------------------------|-----------|-------|---|
| B | Input | 18 | 18-bit data input to multiplier or 18-bit LSB Data input to Adder/Logic Unit. |
| C | Input | 48 | 48-bit data input to adder/Logic Unit and Pattern Detector. |
| CARRYIN | Input | 1 | External carry input to the adder/Logic Unit. |
| P | Output | 48 | Primary data output. |
| CARRYOUT | Output | 4 | Carry out signal for arithmetic operations (addition, subtraction, etc.). <ul style="list-style-type: none"> If USE_SIMD="FOUR12", CARRYOUT[3:0] represents the carryout of each 12 bit field of the Accumulate/Adder/Logic Unit. If USE_SIMD="TWO24", CARRYOUT[3] and CARRYOUT[1] represents the carryout of each 24-bit field of the Accumulator/Adder. If USE_SIMD="ONE48", CARRYOUT[3] is the only valid carry out from the Accumulate/Adder/Logic Unit. |
| Control Inputs/Status Bits | | | |
| CLK | Input | 1 | DSP48E clock input. |
| OPMODE | Input | 7 | Control input to select the arithmetic operation of the DSP48E in conjunction with ALUMODE. |
| ALUMODE | Input | 4 | Control input to select Logic Unit functions including addition and subtraction. |
| CARRYINSEL | Input | 3 | Selects carry in source to the DSP48E. |
| OVERFLOW | Output | 1 | Active High output detects overflow in addition/accumulate if pattern detector is used and PREG = 1. |
| UNDERFLOW | Output | 1 | Active High output detects underflow in addition/accumulate if pattern detector is used and PREG = 1. |
| PATTERNDETECT | Output | 1 | Active High pattern detection. Detects match of P and the selected PATTERN gated by the MASK. Result arrives on the same cycle as P. |
| PATTERN BDETECT | Output | 1 | Active High pattern detection. Detects match of P and the bar of the selected PATTERN gated by the MASK. Result arrives on the same cycle as P. |
| Reset/Clock Enable Inputs | | | |
| RSTA | Input | 1 | Active High, synchronous reset for the A port registers (AREG=1 or 2). Tie to logic zero if not used. |
| RSTB | Input | 1 | Active High, synchronous reset for the B port registers (BREG=1 or 2). Tie to logic zero if not used. |
| RSTC | Input | 1 | Active High, synchronous reset for the C port registers (CREG=1). Tie to logic zero if not used. |
| RSTM | Input | 1 | Active High, synchronous reset for the multiplier registers (MREG=1). Tie to logic zero if not used. |
| RSTP | Input | 1 | Active High, synchronous reset for the P, UNDERFLOW, OVERFLOW, PATTERNDETECT and PATTERNBDETECT and CARRYOUT output registers (PREG=1). Tie to logic zero if not used. |
| RSTCTRL | Input | 1 | Active High, synchronous reset for the OPMODE and CARRYINSEL registers (OPMODEREG=1 and CARRYINSELREG=1). Tie to logic zero if not used. |

| Port | Direction | Width | Function |
|---------------|-----------|-------|---|
| RSTALLCARRYIN | Input | 1 | Active High, synchronous reset for all carry-in registers (CARRYINREG=1) or MULTCARRYINREG=1. Tie to logic zero if not used. |
| RSTALUMODE | Input | 1 | Active High, synchronous reset for the ALUMODE registers (ALUMODEREG=1). Tie to logic zero if not used. |
| CEA1 | Input | 1 | Active High, clock enable for the A port registers (AREG=2). Tie to logic one if not used and AREG=2. Tie to logic zero if AREG=0 or 1. When two registers are used, this is the first sequentially. |
| CEA2 | Input | 1 | Active High, clock enable for the A port registers. Tie to logic one if not used and AREG=1 or 2. Tie to logic zero if AREG=0. When two registers are used, this is the second sequentially. |
| CEB1 | Input | 1 | Active High, clock enable for the B port registers (BREG=2). Tie to logic one if not used and BREG=2. Tie to logic zero if BREG=0 or 1. When two registers are used, this is the first sequentially. |
| CEB2 | Input | 1 | Active High, clock enable for the B port registers. Tie to logic one if not used and BREG=1 or 2. Tie to logic zero if BREG=0. When two registers are used, this is the second sequentially. |
| CEC | Input | 1 | Active High, clock enable for the C port registers (CREG=1). Tie to logic one if not used. |
| CEM | Input | 1 | Active High, clock enable for the multiplier registers (MREG=1). Tie to logic one if not used. |
| CEP | Input | 1 | Active High, clock enable for the output port registers (PREG=1). Tie to logic one if not used. |
| CECTRL | Input | 1 | Active High, clock enable for the OPMODE and Carry-in Select registers (CTRLREG=1). Tie to logic one if not used. |
| CECARRYIN | Input | 1 | Active High, clock enable for the Carry-in registers (CARRYINREG=1). Tie to logic one if not used. |
| CEMULTCARRYIN | Input | 1 | Clock enable for internal multiply symmetric rounding carry register. (MULTCARRYINREG=1). |
| CEALUMODE | Input | 1 | Clock enable for the ALUMODE input registers (ALUMODEREG=1). |
| Cascade Ports | | | |
| ACIN | Input | 30 | Cascade input for Port A. If used, connect to ACOUT of upstream cascaded DSP48E. If not used, tie port to all zeros. |
| BCIN | Input | 18 | Cascade input for Port B. If used, connect to BCOUT of upstream cascaded DSP48E. If not used, tie port to all zeros. |
| PCIN | Input | 48 | Cascade input for Port P. If used, connect to PCOUT of upstream cascaded DSP48E. If not used, tie port to all zeros. |
| CARRYCASCIN | Input | 1 | Cascaded Carryout[2] from previous DSP48E. |
| MULTSIGNIN | Input | 1 | Communicates multiplier sign output of a cascaded DSP48E slice for the purpose of sign extending the adder/accumulator output when greater than a 48-bit output is necessary. Should only be connected to the MULTSIGNOUT output pin. |
| ACOUT | Output | 30 | Cascade output for Port A. If used, connect to ACIN of downstream cascaded DSP48E. If not used, leave unconnected. |
| BCOUT | Output | 18 | Cascade output for Port B. If used, connect to BCIN of downstream cascaded DSP48E. If not used, leave unconnected. |
| PCOUT | Output | 48 | Cascade output for Port P. If used, connect to PCIN of downstream cascaded DSP48E. If not used, leave unconnected. |

| Port | Direction | Width | Function |
|--------------|-----------|-------|---|
| CARRYCASCOUT | Output | 1 | Cascaded Carryout[3] to next DSP48. |
| MULTSIGNOUT | Output | 1 | Communicates multiplier sign output to a cascaded DSP48E element for the purpose of sign extending the adder/accumulator output. Should only be connected to the MULTISIGNIN input pin. |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|---------------------------------|---------|-----------------------|----------|---|
| ACASCREG | Integer | 0, 1 or 2 | 1 | In conjunction with AREG, selects number of A input registers on the ACIN cascade input. Must be equal to or one less than AREG value. |
| AREG | Integer | 0, 1 or 2 | 1 | Selects whether to register the A input to the DSP48E. |
| ALUMODEREG | Integer | 0, 1 | 1 | Selects whether to register the ALUMODE input pins or not. |
| AUTORESET_PATTERN_DETECT | Boolean | TRUE or FALSE | FALSE | Automatically reset DSP48E P Register (accumulated value or Counter Value) on next clock cycle if pattern detect event as determined by AUTORESET_PATTERN_DETECT_OPTINV has occurred on this clock cycle. |
| AUTORESET_PATTERN_DETECT_OPTINV | String | "MATCH", "NOT_MATCH" | "MATCH" | Determines if AUTORESET_PATTERN_DETECT should cause auto reset of P Register on the next cycle A) if pattern is matched or B) whenever pattern is not matched on the current cycle but was matched on the last clock cycle. |
| A_INPUT | String | "DIRECT" or "CASCADE" | "DIRECT" | Selects between A ("DIRECT") and ACIN ("CASCADE") inputs. |
| BCASCREG | Integer | 0, 1, 2 | 1 | In conjunction with BREG, selects number of B input registers on BCIN cascade input. |
| BREG | Integer | 0, 1, 2 | 1 | Selects whether to register the B input to the DSP48E. |
| B_INPUT | String | "DIRECT" or "CASCADE" | "DIRECT" | Selects between B ("DIRECT") and BCIN ("CASCADE") inputs. |
| CARRYINREG | Integer | 0, 1 | 1 | Selects whether to register the CARRYIN input to the DSP48E. |
| CARRYINSELREG | Integer | 0, 1 | 1 | Selects whether to register the CARRYINSEL input to the DSP48E. |

| Attribute | Type | Allowed Values | Default | Description |
|--------------------|-------------|------------------------------|-------------|---|
| CREG | Integer | 0, 1 | 1 | Selects whether to register the C input to the DSP48E. |
| MASK | Hexadecimal | Any 48-Bit Value | 3FFF | Mask to be used for pattern detector. |
| MREG | Integer | 0, 1 | 1 | Selects whether to register the multiplier stage of the DSP48. Enable=1/disable=0. |
| MULTCARRYINREG | Integer | 0, 1 | 1 | Selects number of Internal Carry registers (used for Multiply Symmetric Rounding only). |
| OPMODEREG | Integer | 0, 1 | 1 | Selects whether to register the OPMODE inputs to the DSP48E. |
| PATTERN | Hexadecimal | Any 48-Bit Value | All zeros | Pattern to be used for pattern detector. |
| PREG | Integer | 0, 1 | 1 | Selects whether to register the P input to the DSP48E. |
| SEL_MASK | String | "MASK", "C" | "MASK" | Selects whether to use the static MASK or the C input for the mask of the pattern detector. |
| SEL_PATTERN | String | "PATTERN", "C" | "PATTERN" | Selects whether to use the static PATTERN or the C input for the pattern of the pattern detector. |
| SEL_ROUNDING_MASK | String | "SEL_MASK", "MODE1", "MODE2" | "SEL_MASK" | Selects special mask to be used for symmetric and convergent rounding uses of the pattern detector. If set to "MODE1" or "MODE2" SEL_MASK attribute is overridden. These are used for convergent rounding. |
| SIM_MODE | String | "SAFE" or "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| USE_MULT | String | "MULT", "MULT_S", "NONE" | "MULT_S" | Selects usage of the Multiplier. Set to "NONE" to save power when using only the Adder/Logic Unit. Set to "MULT" if MREG is set to 0 and set to "MULT_S" if MREG is set to 1. |
| USE_SIMD | String | "ONE48", "TWO24", "FOUR12" | "ONE48" | Selects usage of the SIMD (Single Instruction Multiple Data) Adder/Logic Unit. Select between one 48-bit Logic Unit, two 24-bit Logic Unit, or four 12-bit Logic Unit. Note that all four 12 bit Logic Unit share the same Instruction (i.e. all can subtract on the same cycle or add on the same cycle). This does allow the 48 bit adder to be broken up into smaller adders for less computationally intensive applications. SIMD only has an effect on arithmetic operation (add, accumulate, subtract, etc.) and has no effect on logical operations. |
| USE_PATTERN_DETECT | String | "PAT_DET", "NO_PATDET" | "NO_PATDET" | Enables pattern detection. Only affects simulation model and speed files. |

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;

-- DSP48E: DSP Function Block
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

DSP48E_inst : DSP48E
generic map (
    ACASCREG => 1,          -- Number of pipeline registers between
                           -- A/ACIN input and ACOUT output, 0, 1, or 2
    ALUMODEREG => 1,        -- Number of pipeline registers on ALUMODE input, 0 or 1
    AREG => 1,              -- Number of pipeline registers on the A input, 0, 1 or 2
    AUTORESET_PATTERN_DETECT => FALSE, -- Auto-reset upon pattern detect, TRUE or FALSE
    AUTORESET_PATTERN_DETECT_OPTINV => "MATCH", -- Reset if "MATCH" or "NOMATCH"
    A_INPUT => "DIRECT",    -- Selects A input used, "DIRECT" (A port) or "CASCADE" (ACIN port)
    BCASCREG => 1,          -- Number of pipeline registers between B/BCIN input and BCOUT output, 0, 1, or 2
    BREG => 1,              -- Number of pipeline registers on the B input, 0, 1 or 2
    B_INPUT => "DIRECT",    -- Selects B input used, "DIRECT" (B port) or "CASCADE" (BCIN port)
    CARRYINREG => 1,        -- Number of pipeline registers for the CARRYIN input, 0 or 1
    CARRYINSELREG => 1,     -- Number of pipeline registers for the CARRYINSEL input, 0 or 1
    CREG => 1,              -- Number of pipeline registers on the C input, 0 or 1
    MASK => X"3FFFFFFF",    -- 48-bit Mask value for pattern detect
    MREG => 1,              -- Number of multiplier pipeline registers, 0 or 1
    MULTCARRYINREG => 1,    -- Number of pipeline registers for multiplier carry in bit, 0 or 1
    OPMODEREG => 1,         -- Number of pipeline registers on OPMODE input, 0 or 1
    PATTERN => X"000000000000", -- 48-bit Pattern match for pattern detect
    PREG => 1,              -- Number of pipeline registers on the P output, 0 or 1
    SIM_MODE => "SAFE",     -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
                           -- Design Guide" for details
    SEL_MASK => "MASK",     -- Select mask value between the "MASK" value or the value on the "C" port
    SEL_PATTERN => "PATTERN", -- Select pattern value between the "PATTERN" value or the value on the "C" port
    SEL_ROUNDING_MASK => "SEL_MASK", -- "SEL_MASK", "MODE1", "MODE2"
    USE_MULT => "MULT_S",   -- Select multiplier usage, "MULT" (MREG => 0),
                           -- "MULT_S" (MREG => 1), "NONE" (not using multiplier)
    USE_PATTERN_DETECT => "NO_PATDET", -- Enable pattern detect, "PATDET", "NO_PATDET"
    USE_SIMD => "ONE48") -- SIMD selection, "ONE48", "TWO24", "FOUR12"
port map (
    ACOUT => ACOUT, -- 30-bit A port cascade output
    BCOUT => BCOUT, -- 18-bit B port cascade output
    CARRYCASCOUT => CARRYCASCOUT, -- 1-bit cascade carry output
    CARRYOUT => CARRYOUT, -- 4-bit carry output
    MULTSIGNOUT => MULTSIGNOUT, -- 1-bit multiplier sign cascade output
    OVERFLOW => OVERFLOW, -- 1-bit overflow in add/acc output
    P => P, -- 48-bit output
    PATTERNBDETECT => PATTERNBDETECT, -- 1-bit active high pattern bar detect output
    PATTERNDETECT => PATTERNDETECT, -- 1-bit active high pattern detect output
    PCOUT => PCOUT, -- 48-bit cascade output
    UNDERFLOW => UNDERFLOW, -- 1-bit active high underflow in add/acc output
    A => A, -- 30-bit A data input
    ACIN => ACIN, -- 30-bit A cascade data input
    ALUMODE => ALUMODE, -- 4-bit ALU control input
    B => B, -- 18-bit B data input
    BCIN => BCIN, -- 18-bit B cascade input
    C => C, -- 48-bit C data input
    CARRYCASCIN => CARRYCASCIN, -- 1-bit cascade carry input
    CARRYIN => CARRYIN, -- 1-bit carry input signal
    CARRYINSEL => CARRYINSEL, -- 3-bit carry select input
    CE1 => CE1, -- 1-bit active high clock enable input for 1st stage A registers
    CE2 => CE2, -- 1-bit active high clock enable input for 2nd stage A registers
    CEALUMODE => CEALUMODE, -- 1-bit active high clock enable input for ALUMODE registers
    CE1 => CE1, -- 1-bit active high clock enable input for 1st stage B registers
    CE2 => CE2, -- 1-bit active high clock enable input for 2nd stage B registers
    CEC => CEC, -- 1-bit active high clock enable input for C registers
    CECARRYIN => CECARRYIN, -- 1-bit active high clock enable input for CARRYIN register
    CECTRL => CECTRL, -- 1-bit active high clock enable input for OPMODE and carry registers
    CEM => CEM, -- 1-bit active high clock enable input for multiplier registers
    CEMULTCARRYIN => CEMULTCARRYIN, -- 1-bit active high clock enable for multiplier carry in register

```

```

CEP => CEP,          -- 1-bit active high clock enable input for P registers
CLK => CLK,          -- Clock input
MULTSIGNIN => MULTSIGNIN, -- 1-bit multiplier sign input
OPMODE => OPMODE,    -- 7-bit operation mode input
PCIN => PCIN,        -- 48-bit P cascade input
RSTA => RSTA,        -- 1-bit reset input for A pipeline registers
RSTALLCARRYIN => RSTALLCARRYIN, -- 1-bit reset input for carry pipeline registers
RSTALUMODE => RSTALUMODE, -- 1-bit reset input for ALUMODE pipeline registers
RSTB => RSTB,        -- 1-bit reset input for B pipeline registers
RSTC => RSTC,        -- 1-bit reset input for C pipeline registers
RSTCTRL => RSTCTRL,  -- 1-bit reset input for OPMODE pipeline registers
RSTM => RSTM,        -- 1-bit reset input for multiplier registers
RSTP => RSTP,        -- 1-bit reset input for P pipeline registers
);

-- End of DSP48E_inst instantiation

```

Verilog Instantiation Template

```

// DSP48E: DSP Function Block
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

DSP48E #(
    .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
    .ACASCREG(1),       // Number of pipeline registers between A/ACIN input and ACOUT output, 0, 1, or 2
    .ALUMODEREG(1),     // Number of pipeline registers on ALUMODE input, 0 or 1
    .AREG(1),           // Number of pipeline registers on the A input, 0, 1 or 2
    .AUTORESET_PATTERN_DETECT("FALSE"), // Auto-reset upon pattern detect, "TRUE" or "FALSE"
    .AUTORESET_PATTERN_DETECT_OPTINV("MATCH"), // Reset if "MATCH" or "NOMATCH"
    .A_INPUT("DIRECT"), // Selects A input used, "DIRECT" (A port) or "CASCADE" (ACIN port)
    .BCASCREG(1),       // Number of pipeline registers between B/BCIN input and BCOUT output, 0, 1, or 2
    .BREG(1),           // Number of pipeline registers on the B input, 0, 1 or 2
    .B_INPUT("DIRECT"), // Selects B input used, "DIRECT" (B port) or "CASCADE" (BCIN port)
    .CARRYINREG(1),     // Number of pipeline registers for the CARRYIN input, 0 or 1
    .CARRYINSELREG(1),  // Number of pipeline registers for the CARRYINSEL input, 0 or 1
    .CREG(1),           // Number of pipeline registers on the C input, 0 or 1
    .MASK(48'h3fffffff), // 48-bit Mask value for pattern detect
    .MREG(1),           // Number of multiplier pipeline registers, 0 or 1
    .MULTCARRYINREG(1), // Number of pipeline registers for multiplier carry in bit, 0 or 1
    .OPMODEREG(1),      // Number of pipeline registers on OPMODE input, 0 or 1
    .PATTERN(48'h000000000000), // 48-bit Pattern match for pattern detect
    .PREG(1),           // Number of pipeline registers on the P output, 0 or 1
    .SEL_MASK("MASK"),  // Select mask value between the "MASK" value or the value on the "C" port
    .SEL_PATTERN("PATTERN"), // Select pattern value between the "PATTERN" value or the value on the "C" port
    .SEL_ROUNDING_MASK("SEL_MASK"), // "SEL_MASK", "MODE1", "MODE2"
    .USE_MULT("MULT_S"), // Select multiplier usage, "MULT" (MREG => 0), "MULT_S" (MREG => 1), "NONE" (no multiplier)
    .USE_PATTERN_DETECT("NO_PATDET"), // Enable pattern detect, "PATDET", "NO_PATDET"
    .USE_SIMD("ONE48") // SIMD selection, "ONE48", "TWO24", "FOUR12"
) DSP48E_inst (
    .ACOUT(ACOUT), // 30-bit A port cascade output
    .BCOUT(BCOUT), // 18-bit B port cascade output
    .CARRYCASCOUT(CARRYCASCOUT), // 1-bit cascade carry output
    .CARRYOUT(CARRYOUT), // 4-bit carry output
    .MULTSIGNOUT(MULTSIGNOUT), // 1-bit multiplier sign cascade output
    .OVERFLOW(OVERFLOW), // 1-bit overflow in add/acc output
    .P(P), // 48-bit output
    .PATTERNBDETECT(PATTERNBDETECT), // 1-bit active high pattern bar detect output
    .PATTERNDETECT(PATTERNDETECT), // 1-bit active high pattern detect output
    .PCOUT(PCOUT), // 48-bit cascade output
    .UNDERFLOW(UNDERFLOW), // 1-bit active high underflow in add/acc output
    .A(A), // 30-bit A data input
    .ACIN(ACIN), // 30-bit A cascade data input
    .ALUMODE(ALUMODE), // 4-bit ALU control input
    .B(B), // 18-bit B data input
    .BCIN(BCIN), // 18-bit B cascade input
    .C(C), // 48-bit C data input
    .CARRYCASCIN(CARRYCASCIN), // 1-bit cascade carry input
    .CARRYIN(CARRYIN), // 1-bit carry input signal
    .CARRYINSEL(CARRYINSEL), // 3-bit carry select input
    .CEA1(CEA1), // 1-bit active high clock enable input for 1st stage A registers

```

```
.CEA2(CEA2), // 1-bit active high clock enable input for 2nd stage A registers
.CEALUMODE(CEALUMODE), // 1-bit active high clock enable input for ALUMODE registers
.CEB1(CEB1), // 1-bit active high clock enable input for 1st stage B registers
.CEB2(CEB2), // 1-bit active high clock enable input for 2nd stage B registers
.CEC(CEC), // 1-bit active high clock enable input for C registers
.CECARRYIN(CECARRYIN), // 1-bit active high clock enable input for CARRYIN register
.CECTRL(CECTRL), // 1-bit active high clock enable input for OPMODE and carry registers
.CEM(CEM), // 1-bit active high clock enable input for multiplier registers
.CEMULTCARRYIN(CEMULTCARRYIN), // 1-bit active high clock enable for multiplier carry in register
.CEP(CEP), // 1-bit active high clock enable input for P registers
.CLK(CLK), // Clock input
.MULTSIGNIN(MULTSIGNIN), // 1-bit multiplier sign input
.OPMODE(OPMODE), // 7-bit operation mode input
.PCIN(PCIN), // 48-bit P cascade input
.RSTA(RSTA), // 1-bit reset input for A pipeline registers
.RSTALLCARRYIN(RSTALLCARRYIN), // 1-bit reset input for carry pipeline registers
.RSTALUMODE(RSTALUMODE), // 1-bit reset input for ALUMODE pipeline registers
.RSTB(RSTB), // 1-bit reset input for B pipeline registers
.RSTC(RSTC), // 1-bit reset input for C pipeline registers
.RSTCTRL(RSTCTRL), // 1-bit reset input for OPMODE pipeline registers
.RSTM(RSTM), // 1-bit reset input for multiplier registers
.RSTP(RSTP) // 1-bit reset input for P pipeline registers
);

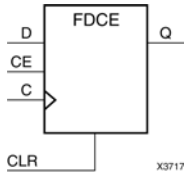
// End of DSP48E_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FDCE

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



Introduction

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

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

Logic Table

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

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|--------|----------------|---------|--|
| INIT | Binary | 0, 1 | 1 | Sets the initial value of Q output after configuration For Spartan®-6, Xilinx recommends that the INIT value always matches the polarity of the set or reset. For this element, the INIT should be 1. If set to 0, additional asynchronous circuitry will be created to correctly model the behavior. |

VHDL Instantiation Template

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

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

-- FDCE: Single Data Rate D Flip-Flop with Asynchronous Clear and
--       Clock Enable (posedge clk).
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of FDCE_inst instantiation
```

Verilog Instantiation Template

```
// FDCE: Single Data Rate D Flip-Flop with Asynchronous Clear and
//       Clock Enable (posedge clk).
//       All families.
// Xilinx HDL Libraries Guide, version 12.3

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

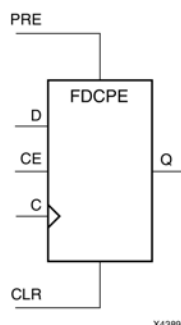
// End of FDCE_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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).
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of FDCPE_inst instantiation
```

Verilog Instantiation Template

```
// FDCPE: Single Data Rate D Flip-Flop with Asynchronous Clear, Set and
//       Clock Enable (posedge clk).
//       Virtex-5, Spartan-3/3E/3A/3A DSP
// Xilinx HDL Libraries Guide, version 12.3

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

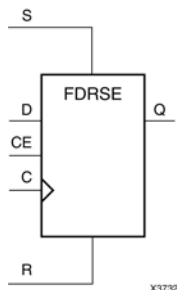
// End of FDCPE_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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).
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of FDRSE_inst instantiation
```

Verilog Instantiation Template

```
// FDRSE: Single Data Rate D Flip-Flop with Synchronous Clear, Set and
//       Clock Enable (posedge clk).
//       Virtex-5, Spartan-3/3E/3A/3A DSP
// Xilinx HDL Libraries Guide, version 12.3

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

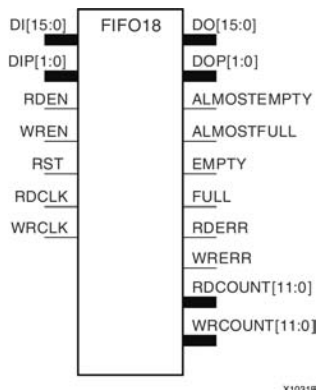
// End of FDRSE_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FIFO18

Primitive: 18kb FIFO (First In, First Out) Block RAM Memory



Introduction

Virtex®-5 and above devices contain several block RAM memories, each of which can be separately configured as a FIFO, an automatic error-correction RAM, or as a general-purpose 36kb or 18kb RAM/ROM memory. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. The FIFO18 uses the FIFO control logic and the 18kb block RAM. This primitive can be used in a 4-bit wide by 4K deep, 9-bit wide by 2K deep, or an 18-bit wide by 1K deep configuration. The primitive can be configured in either synchronous or multirate (asynchronous) mode, with all associated FIFO flags and status signals.

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

Note For a 36-bit wide by 512 deep FIFO, use the FIFO18_36. For deeper or wider configurations of the FIFO, use the FIFO36 or FIFO36_72. If error-correction circuitry is desired, use the FIFO36_72.

Port Descriptions

| Port | Direction | Width | Function |
|------------------|-----------|----------|---|
| DO | Output | 4, 8, 16 | FIFO data output bus. |
| DOP | Output | 0, 1, 2 | FIFO parity data output bus. |
| FULL | Output | 1 | Active high logic indicates that the FIFO is full. |
| ALMOSTFULL | Output | 1 | Programmable flag to indicate that the FIFO is almost full. The ALMOST_FULL_OFFSET attribute specifies the threshold where this flag is triggered relative to full/empty. |
| EMPTY | Output | 1 | Active high logic to indicate that the FIFO is currently empty. |
| ALMOSTEMPTY | Output | 1 | Programmable flag to indicate the FIFO is almost empty. ALMOST_EMPTY_OFFSET attribute specifies the threshold where this flag is triggered relative to full/empty. |
| WRERR, RDERR | Output | 1 | WRERR indicates that a write occurred while the FIFO was full and RDERR indicates that a read occurred while the FIFO was empty |
| WRCOUNT, RDCOUNT | Output | 12 | FIFO write/read pointer. |
| DI | Input | 4, 8, 16 | FIFO data input bus. |
| DIP | Input | 0, 1, 2 | FIFO parity data input bus. |

| Port | Direction | Width | Function |
|--------------|-----------|-------|---|
| WREN | Input | 1 | Active high FIFO write enable. |
| RDEN | Input | 1 | Active high FIFO read enable. |
| RST | Input | 1 | Asynchronous reset (active high) of all FIFO functions, flags, and pointers. RESET must be asserted for three clock cycles. |
| WRCLK, RDCLK | Input | 1 | FIFO read and write clocks (positive edge triggered). |

Design Entry Method

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

When you want to instantiate this primitive configured in the 4-bit WIDTH mode, connect the DIP port to logic zeros and leave the DOP port unconnected. Connect DI[3:0] and DO[3:0] to the appropriate input and output signals and tie DI[15:4] to logic zeros and leave DO[15:4] unconnected.

When configuring in the 9-bit WIDTH mode, connect the DIP[0] port to the appropriate data input and the DIP[1] to a logic zero. Connect DOP[0] to the appropriate data out and leave DOP[1] unconnected. Connect DI[7:0] and DO[7:0] to the appropriate input and output signals and tie DI[15:8] to logic zeros and leave DO[15:8] unconnected.

When configuring in the 18-bit WIDTH mode, all DI, DIP, DO and DOP signals can be connected.

For any configuration, any unused DI or DIP inputs should be tied to a logic zero, and any unused DO or DOP pins should be left unconnected. When the FIFO is set to be synchronous (EN_SYN attribute is set to TRUE), the same clock source must be tied to WRCLK and RDCLK. When in asynchronous mode (EN_SYN is set to FALSE), unique clock signals can be used. Depending on the offset between read and write clock edges, the Empty, Almost Empty, Full and Almost Full flags can deassert one cycle later. Due to the asynchronous nature of the clocks the simulation model only reflects the deassertion latency cycles listed in the architecture user guide.

The FIFO must be RST after power up. The FULL, ALMOSTFULL, EMPTY and ALMOSTEMPTY output flags should be connected to the appropriate destination logic or left unconnected if not used. The WRERR, RDERR, WRCOUNT and RDCOUNT are optional outputs and can be left unconnected if not needed. Set all attributes to the FIFO to enable the desired behavior of the primitive by adjusting the generics (VHDL) or in-line defparams (Verilog) in the instantiation.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------|-------------|------------------|-----------|---|
| ALMOST_FULL_OFFSET | Hexadecimal | Any 12-Bit Value | All zeros | Specifies the amount of data contents in the RAM to trigger the ALMOST_FULL flag. |
| ALMOST_EMPTY_OFFSET | Hexadecimal | Any 12-Bit Value | All zeros | Specifies the amount of data contents in the RAM to trigger the ALMOST_EMPTY flag. |
| FIRST_WORD_FALL_THROUGH | Boolean | TRUE, FALSE | FALSE | If TRUE, the first write to the FIFO will appear on DO without a first RDEN assertion. |
| DATA_WIDTH | Integer | 4, 9, 18 | 4 | Specifies the desired data width for the FIFO. |
| EN_SYN | Boolean | TRUE, FALSE | FALSE | EN_SYN denotes whether the FIFO is operating in either multirate (two independent clocks) or synchronous (a |

| Attribute | Type | Allowed Values | Default | Description |
|-----------|---------|-----------------|---------|---|
| | | | | single clock) mode. Multirate must use DO_REG=1. |
| DO_REG | Integer | 0, 1 | 1 | Data pipeline register for EN_SYN. |
| SIM_MODE | String | "SAFE", "FAST". | "SAFE" | <p>This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST".</p> <p>Note Not all features are supported when set to "FAST". Please see the <i>Synthesis and Simulation Design Guide</i> for more information.</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;

-- FIFO18: 16k+2k Parity Synchronous/Asynchronous BlockRAM FIFO BlockRAM Memory
--          Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

FIFO18_inst : FIFO18
generic map (
    ALMOST_FULL_OFFSET => X"080", -- Sets almost full threshold
    ALMOST_EMPTY_OFFSET => X"080", -- Sets the almost empty threshold
    DATA_WIDTH => 4, -- Sets data width to 4, 9, or 18
    DO_REG => 1, -- Enable output register ( 0 or 1)
    -- Must be 1 if the EN_SYN = FALSE
    EN_SYN => FALSE, -- Specified FIFO as Asynchronous (FALSE) or
    -- Synchronous (TRUE)
    FIRST_WORD_FALL_THROUGH => FALSE, -- Sets the FIFO FWFT to TRUE or FALSE
    SIM_MODE => "SAFE") -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
    -- Design Guide" for details
port map (
    ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty output flag
    ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
    DO => DO, -- 32-bit data output
    DOP => DOP, -- 2-bit parity data output
    EMPTY => EMPTY, -- 1-bit empty output flag
    FULL => FULL, -- 1-bit full output flag
    RDCOUNT => RDCOUNT, -- 12-bit read count output
    RDERR => RDERR, -- 1-bit read error output
    WRCOUNT => WRCOUNT, -- 12-bit write count output
    WRERR => WRERR, -- 1-bit write error
    DI => DI, -- 16-bit data input
    DIP => DIP, -- 2-bit parity input
    RDCLK => RDCLK, -- 1-bit read clock input
    RDEN => RDEN, -- 1-bit read enable input
    RST => RST, -- 1-bit reset input
    WRCLK => WRCLK, -- 1-bit write clock input
    WREN => WREN -- 1-bit write enable input
);

-- End of FIFO18_inst instantiation

```


Verilog Instantiation Template

```
// FIFO18: 16k+2k Parity Synchronous/Asynchronous BlockRAM FIFO
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FIFO18 #(
    .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
    .ALMOST_FULL_OFFSET(12'h080), // Sets almost full threshold
    .ALMOST_EMPTY_OFFSET(12'h080), // Sets the almost empty threshold
    .DATA_WIDTH(4), // Sets data width to 4, 9 or 18
    .DO_REG(1), // Enable output register (0 or 1)
    // Must be 1 if EN_SYN = "FALSE"
    .EN_SYN("FALSE"), // Specifies FIFO as Asynchronous ("FALSE")
    // or Synchronous ("TRUE")
    .FIRST_WORD_FALL_THROUGH("FALSE") // Sets the FIFO FWFT to "TRUE" or "FALSE"
) FIFO18_inst (
    .ALMOSTEMPTY(ALMOSTEMPTY), // 1-bit almost empty output flag
    .ALMOSTFULL(ALMOSTFULL), // 1-bit almost full output flag
    .DO(DO), // 16-bit data output
    .DOP(DOP), // 2-bit parity data output
    .EMPTY(EMPTY), // 1-bit empty output flag
    .FULL(FULL), // 1-bit full output flag
    .RDCOUNT(RDCOUNT), // 12-bit read count output
    .RDERR(RDERR), // 1-bit read error output
    .WRCOUNT(WRCOUNT), // 12-bit write count output
    .WRERR(WRERR), // 1-bit write error
    .DI(DI), // 16-bit data input
    .DIP(DIP), // 2-bit parity input
    .RDCLK(RDCLK), // 1-bit read clock input
    .RDEN(RDEN), // 1-bit read enable input
    .RST(RST), // 1-bit reset input
    .WRCLK(WRCLK), // 1-bit write clock input
    .WREN(WREN) // 1-bit write enable input
);

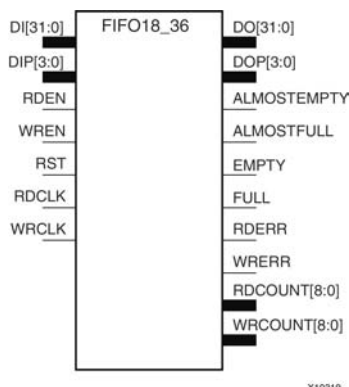
// End of FIFO18_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FIFO18_36

Primitive: 36-bit Wide by 512 Deep 18kb FIFO (First In, First Out) Block RAM Memory



Introduction

Virtex®-5 devices contain several block RAM memories that can be configured as FIFOs, automatic error-correction RAM, or general-purpose 36kb or 18kb RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. The FIFO18_36 allows access to the block RAM in the 18kb FIFO configuration when a wide data path is needed. This component is set to a 36-bit wide, 512 deep ration with configurable synchronous or asynchronous operation. This FIFO RAM also supplies all associated FIFO flags and status signals.

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

Note For an 18kb FIFO in a deeper, less wide configuration, use the FIFO18 component. For deeper or wider configurations of the FIFO, use the FIFO36 or FIFO36_72 components. If you want error-correction circuitry, use FIFO36_72.

Port Descriptions

| Port | Direction | Width | Function |
|------------------|-----------|-------|--|
| DO | Output | 32 | FIFO data output bus. |
| DOP | Output | 4 | FIFO parity data output bus. |
| FULL | Output | 1 | Active high logic indicates that the FIFO contents are full. |
| ALMOSTFULL | Output | 1 | Programmable flag to indicate the FIFO is almost full. ALMOST_FULL_OFFSET attribute specifies where to trigger this flag. |
| EMPTY | Output | 1 | Active high logic indicates the FIFO is currently empty. |
| ALMOSTEMPTY | Output | 1 | Programmable flag to indicate the FIFO is almost empty. ALMOST_EMPTY_OFFSET attribute specifies where to trigger this flag. |
| WRERR, RDERR | Output | 1 | WRERR indicates that a write occurred while the FIFO was full while RDERR indicated a read occurred while the FIFO was empty |
| WRCOUNT, RDCOUNT | Output | 9 | FIFO write/read pointer. |

| Port | Direction | Width | Function |
|-----------------|-----------|-------|---|
| DI | Input | 32 | FIFO data input bus. |
| DIP | Input | 4 | FIFO parity data input bus. |
| WREN | Input | 1 | Active high FIFO write enable. |
| RDEN | Input | 1 | Active high FIFO read enable. |
| RST | Input | 1 | Asynchronous reset (active high) of all FIFO functions, flags, and pointers. RESET must be asserted for three clock cycles. |
| WRCLK, RDCLK | Input | 1 | FIFO read and write clocks (positive edge triggered). |

Design Entry Method

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

DI, DIP, DO and DOP should be connected to their respective input and output data sources. When you are using fewer than 36-bits, connect any unused DI or DIP inputs to a logic zero and any unused DO or DOP pins should be left unconnected. When you are the FIFO set to be synchronous (EN_SYN attribute is set to TRUE), the same clock source should be tied to WRCLK and RDCLK. When you are in asynchronous mode (EN_SYN is set to FALSE), unique clock signals should be used. Depending on the offset between read and write clock edges, the Empty, Almost Empty, Full and Almost Full flags can deassert one cycle later. Due to the asynchronous nature of the clocks the simulation model only reflects the deassertion latency cycles listed in the architecture user guide. WREN and RDEN should be connected to the respective write enable and read enable signal/ logic. RST should be either tied to the appropriate reset signal/logic, or connected to a logic zero if unused.

The FULL, ALMOSTFULL, EMPTY and ALMOSTEMPTY output flags should be connected to the appropriate destination logic, or left unconnected, if not used. The WRERR, RDERR, WRCOUNT and RDCOUNT are optional outputs that can be left unconnected, if not needed. Set all attributes to the FIFO to enable the desired behavior of the component by adjusting the generics (VHDL) or in-line defparams (Verilog) in the instantiation code.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------|-------------|-----------------|-----------|--|
| ALMOST_FULL_OFFSET | Hexadecimal | Any 9-Bit Value | All zeros | Specifies the amount of data contents in the RAM to trigger the ALMOST_FULL flag. |
| ALMOST_EMPTY_OFFSET | Hexadecimal | Any 9-Bit Value | All zeros | Specifies the amount of data contents in the RAM to trigger the ALMOST_EMPTY flag. |
| FIRST_WORD_FALL_THROUGH | Boolean | TRUE, FALSE | FALSE | If TRUE, the first write to the FIFO will appear on DO without an RDEN assertion. |
| EN_SYN | Boolean | TRUE, FALSE | FALSE | When FALSE, specifies the FIFO to be used in asynchronous mode (two independent clock) or when TRUE in synchronous (a single clock) operation. |
| DO_REG | Integer | 0, 1 | 1 | Enable output register to the FIFO for improved clock-to-out timing at the expense of added read latency (one |

| Attribute | Type | Allowed Values | Default | Description |
|-----------|--------|------------------|---------|--|
| | | | | pipeline delay). DO_REG must be 1 when EN_SYN is set to FALSE. |
| SIM_MODE | String | "SAFE", "FAST" . | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- FIFO18_36: 36x18k Synchronous/Asynchronous BlockRAM FIFO
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

FIFO18_36_inst : FIFO18_36
generic map (
    ALMOST_FULL_OFFSET => X"0080", -- Sets almost full threshold
    ALMOST_EMPTY_OFFSET => X"0080", -- Sets the almost empty threshold
    DO_REG => 1, -- Enable output register (0 or 1)
    -- Must be 1 if EN_SYN = FALSE
    EN_SYN => FALSE, -- Specifies FIFO as Asynchronous (FALSE)
    -- or Synchronous (TRUE)
    FIRST_WORD_FALL_THROUGH => FALSE, -- Sets the FIFO FWFT to TRUE or FALSE
    SIM_MODE => "SAFE") -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
    -- Design Guide" for details
port map (
    ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty output flag
    ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
    DO => DO, -- 32-bit data output
    DOP => DOP, -- 4-bit parity data output
    EMPTY => EMPTY, -- 1-bit empty output flag
    FULL => FULL, -- 1-bit full output flag
    RDCOUNT => RDCOUNT, -- 9-bit read count output
    RDERR => RDERR, -- 1-bit read error output
    WRCOUNT => WRCOUNT, -- 9-bit write count output
    WRERR => WRERR, -- 1-bit write error
    DI => DI, -- 32-bit data input
    DIP => DIP, -- 4-bit parity input
    RDCLK => RDCLK, -- 1-bit read clock input
    RDEN => RDEN, -- 1-bit read enable input
    RST => RST, -- 1-bit reset input
    WRCLK => WRCLK, -- 1-bit write clock input
    WREN => WREN -- 1-bit write enable input
);

-- End of FIFO18_36_inst instantiation

```

Verilog Instantiation Template

```
// FIFO18_36: 36x18k Synchronous/Asynchronous BlockRAM FIFO
//          Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FIFO18_36 #(
    .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
    .ALMOST_FULL_OFFSET(9'h080), // Sets almost full threshold
    .ALMOST_EMPTY_OFFSET(9'h080), // Sets the almost empty threshold
    .DO_REG(1), // Enable output register (0 or 1)
    // Must be 1 if EN_SYN = "FALSE"
    .EN_SYN("FALSE"), // Specifies FIFO as Asynchronous ("FALSE")
    // or Synchronous ("TRUE")
    .FIRST_WORD_FALL_THROUGH("FALSE") // Sets the FIFO FWFT to "TRUE" or "FALSE"
) FIFO18_36_inst (
    .ALMOSTEMPTY(ALMOSTEMPTY), // 1-bit almost empty output flag
    .ALMOSTFULL(ALMOSTFULL), // 1-bit almost full output flag
    .DO(DO), // 32-bit data output
    .DOP(DOP), // 4-bit parity data output
    .EMPTY(EMPTY), // 1-bit empty output flag
    .FULL(FULL), // 1-bit full output flag
    .RDCOUNT(RDCOUNT), // 9-bit read count output
    .RDERR(RDERR), // 1-bit read error output
    .WRCOUNT(WRCOUNT), // 9-bit write count output
    .WRERR(WRERR), // 1-bit write error
    .DI(DI), // 32-bit data input
    .DIP(DIP), // 4-bit parity input
    .RDCLK(RDCLK), // 1-bit read clock input
    .RDEN(RDEN), // 1-bit read enable input
    .RST(RST), // 1-bit reset input
    .WRCLK(WRCLK), // 1-bit write clock input
    .WREN(WREN) // 1-bit write enable input
);

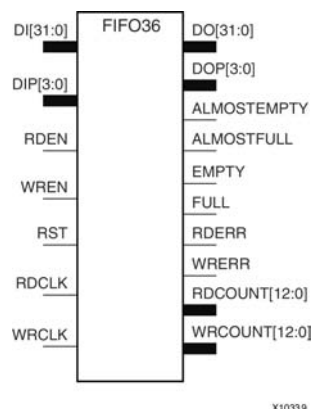
// End of FIFO18_36_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FIFO36

Primitive: 36kb FIFO (First In, First Out) Block RAM Memory



Introduction

Virtex®-5 and above devices contain several block RAM memories that can be configured as FIFOs, automatic error-correction RAM, or general-purpose 36kb or 18kb RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. The FIFO36 allows access to the block RAM in the 36kb FIFO configurations. This component can be configured and used as a 4-bit wide by 8K deep, 9-bit by 4K deep, 18-bit by 2K deep or a 36-bit wide by 1K deep synchronous or multirate (asynchronous) FIFO RAM with all associated FIFO flags.

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

Note For a 72-bit wide by 512 deep use the FIFO, the FIFO36_72 component. For smaller configurations of the FIFO, use the FIFO18 or FIFO18_36. If error-correction circuitry is desired, use the FIFO36_72.

Port Descriptions

| Port | Direction | Width | Function |
|--------------|-----------|--------------|---|
| DO | Output | 4, 8, 16, 32 | FIFO data output bus. |
| DOP | Output | 0, 1, 2, 4 | FIFO parity data output bus. |
| FULL | Output | 1 | Active high logic indicates that the FIFO contents are full. |
| ALMOSTFULL | Output | 1 | Programmable flag to indicate that the FIFO is almost full. The ALMOST_FULL_OFFSET attribute specifies the threshold where this flag is triggered relative to full/empty. |
| EMPTY | Output | 1 | Active high logic indicates that the FIFO is currently empty. |
| ALMOSTEMPTY | Output | 1 | Programmable flag to indicate the FIFO is almost empty. ALMOST_EMPTY_OFFSET attribute specifies the threshold where this flag is triggered relative to full/empty. |
| WRERR, RDERR | Output | 1 | WRERR indicates that a write occurred while the FIFO was full and RDERR indicates that a read occurred while the FIFO was empty. |

| Port | Direction | Width | Function |
|------------------|-----------|--------------|---|
| WRCOUNT, RDCOUNT | Output | 13 | FIFO write/read pointer. |
| DI | Input | 4, 8, 16, 32 | FIFO data input bus. |
| DIP | Input | 0, 1, 2, 4 | FIFO parity data bus. |
| WREN | Input | 1 | Active high FIFO write enable. |
| RDEN | Input | 1 | Active high FIFO read enable. |
| RST | Input | 1 | Asynchronous reset (active high) of all FIFO functions, flags, and pointers. RESET must be asserted for three clock cycles. |
| WRCLK, RDCLK | Input | 1 | FIFO read and write clocks (positive edge triggered). |

Design Entry Method

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

When you are instantiating the primitive configured in the 4-bit WIDTH mode, connect the DIP port to logic zeros and leave the DOP port unconnected. Connect DI[3:0] and DO[3:0] to the appropriate input and output signals and tie DI[31:4] to logic zeros and leave DO[31:4] unconnected.

When you are configuring in the 9-bit WIDTH mode, connect the DIP[0] port to the appropriate data input and the DIP[3:1] to a logic zero. Connect DOP[0] to the appropriate data out and leave DOP[3:1] unconnected. Connect DI[7:0] and DO[7:0] to the appropriate input and output signals and tie DI[31:8] to logic zeros and leave DO[31:8] unconnected.

When you are configuring in the 18-bit WIDTH mode, connect the DIP[1:0] port to the appropriate data input and the DIP[3:2] to a logic zero. Connect DOP[1:0] to the appropriate data out and leave DOP[3:2] unconnected. Connect DI[15:0] and DO[15:0] to the appropriate input and output signals and tie DI[31:16] to logic zeros and leave DO[31:16] unconnected.

When you are configuring in the 36-bit WIDTH mode, all DI, DIP, DO and DOP signals can be connected.

For any configuration, any unused DI or DIP inputs should be tied to a logic zero and any unused DO or DOP pins should be left unconnected. When the FIFO is set to be synchronous (EN_SYN attribute is set to TRUE), the same clock source should be tied to WRCLK and RDCLK.

When you are in asynchronous mode (EN_SYN is set to FALSE), unique clock signals should be used. Depending on the offset between read and write clock edges, the Empty, Almost Empty, Full and Almost Full flags can deassert one cycle later. Due to the asynchronous nature of the clocks the simulation model only reflects the deassertion latency cycles listed in the architecture user guide. WREN and RDEN should be connected to the respective write enable and read enable signal/ logic. RST should be either tied to the appropriate reset signal/logic or connected to a logic zero if unused. The FULL, ALMOSTFULL, EMPTY and ALMOSTEMPTY output flags should be connected to the appropriate destination logic or left unconnected if not used. The WRERR, RDERR, WRCOUNT and RDCOUNT are optional outputs and can be left unconnected if not needed. Set all attributes to the FIFO to enable the desired behavior of the component by adjusting the generics (VHDL) or in-line defparams (Verilog) in the instantiation code.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------|-------------|------------------|-----------|--|
| ALMOST_FULL_OFFSET | Hexadecimal | Any 13-Bit Value | All zeros | Specifies the amount of data contents in the RAM to trigger the ALMOST_FULL flag. |
| ALMOST_EMPTY_OFFSET | Hexadecimal | Any 13-Bit Value | All zeros | Specifies the amount of data contents in the RAM to trigger the ALMOST_EMPTY flag. |
| FIRST_WORD_FALL_THROUGH | Boolean | TRUE, FALSE | FALSE | If TRUE, the first write to the FIFO will appear on DO without an RDEN assertion. |
| DATA_WIDTH | Integer | 4 to 36 | 4 | Specifies the desired data width for the FIFO. |
| EN_SYN | Boolean | TRUE, FALSE | FALSE | EN_SYN denotes whether the FIFO is operating in either multirate (two independent clocks) or synchronous (a single clock) mode. Multirate must use DO_REG=1. |
| DO_REG | Integer | 0, 1 | 1 | Data pipeline register for EN_SYN. |
| SIM_MODE | String | "SAFE", "FAST". | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |

VHDL Instantiation Template

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

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

```
-- FIFO36: 32k+4k Parity Synchronous/Asynchronous BlockRAM FIFO BlockRAM Memory
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
FIFO36_inst : FIFO36
generic map (
    ALMOST_FULL_OFFSET => X"0080", -- Sets almost full threshold
    ALMOST_EMPTY_OFFSET => X"0080", -- Sets the almost empty threshold
    DATA_WIDTH => 4, -- Sets data width to 4, 9, 18, or 36
    DO_REG => 1, -- Enable output register ( 0 or 1)
    -- Must be 1 if the EN_SYN = FALSE
    EN_SYN => FALSE, -- Specified FIFO as Asynchronous (FALSE) or
    -- Synchronous (TRUE)
    FIRST_WORD_FALL_THROUGH => FALSE, -- Sets the FIFO FWFT to TRUE or FALSE
    SIM_MODE => "SAFE") -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
    -- Design Guide" for details
```

```
port map (
    ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty output flag
    ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
    DO => DO, -- 32-bit data output
    DOP => DOP, -- 4-bit parity data output
    EMPTY => EMPTY, -- 1-bit empty output flag
    FULL => FULL, -- 1-bit full output flag
    RDCOUNT => RDCOUNT, -- 13-bit read count output
    RDERR => RDERR, -- 1-bit read error output
    WRCOUNT => WRCOUNT, -- 13-bit write count output
    WRERR => WRERR, -- 1-bit write error
    DI => DI, -- 32-bit data input
    DIP => DIP, -- 4-bit parity input
    RDCLK => RDCLK, -- 1-bit read clock input
    RDEN => RDEN, -- 1-bit read enable input
    RST => RST, -- 1-bit reset input
    WRCLK => WRCLK, -- 1-bit write clock input
    WREN => WREN -- 1-bit write enable input
```



```
);
-- End of FIFO36_inst instantiation
```

Verilog Instantiation Template

```
// FIFO36: 32k+4k Parity Synchronous/Asynchronous BlockRAM FIFO
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FIFO36 #(
    .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
    .ALMOST_FULL_OFFSET(13'h0080), // Sets almost full threshold
    .ALMOST_EMPTY_OFFSET(13'h0080), // Sets the almost empty threshold
    .DATA_WIDTH(4), // Sets data width to 4, 9, 18 or 36
    .DO_REG(1), // Enable output register (0 or 1)
    // Must be 1 if EN_SYN = "FALSE"
    .EN_SYN("FALSE"), // Specifies FIFO as Asynchronous ("FALSE")
    // or Synchronous ("TRUE")
    .FIRST_WORD_FALL_THROUGH("FALSE") // Sets the FIFO FWFT to "TRUE" or "FALSE"
) FIFO36_inst (
    .ALMOSTEMPTY(ALMOSTEMPTY), // 1-bit almost empty output flag
    .ALMOSTFULL(ALMOSTFULL), // 1-bit almost full output flag
    .DO(DO), // 32-bit data output
    .DOP(DOP), // 4-bit parity data output
    .EMPTY(EMPTY), // 1-bit empty output flag
    .FULL(FULL), // 1-bit full output flag
    .RDCOUNT(RDCOUNT), // 13-bit read count output
    .RDERR(RDERR), // 1-bit read error output
    .WRCOUNT(WRCOUNT), // 13-bit write count output
    .WRERR(WRERR), // 1-bit write error
    .DI(DI), // 32-bit data input
    .DIP(DIP), // 4-bit parity input
    .RDCLK(RDCLK), // 1-bit read clock input
    .RDEN(RDEN), // 1-bit read enable input
    .RST(RST), // 1-bit reset input
    .WRCLK(WRCLK), // 1-bit write clock input
    .WREN(WREN) // 1-bit write enable input
);

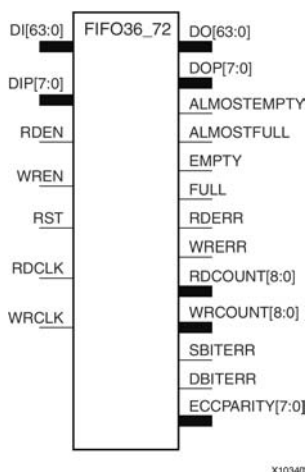
// End of FIFO36_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FIFO36_72

Primitive: 72-Bit Wide by 512 Deep 36kb FIFO (First In, First Out) Block RAM Memory with ECC (Error Detection and Correction Circuitry)



Introduction

Virtex®-5 devices contain several block RAM memories that can be configured as FIFOs, automatic error-correction RAM, or general-purpose 36kb or 18kb RAM/ROM memories. These Block RAM memories offer fast and flexible storage of large amounts of on-chip data. This element allows access to the Block RAM in the 36kB FIFO configuration when a wide data path is needed. This component is set to a 72-bit wide, 512 deep ration, with configurable synchronous or asynchronous operation. Error detection and correction circuitry can also be enabled to uncover and rectify possible memory corruptions. This FIFO RAM also supplies all associated FIFO flags and status signals.

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

Note For a 36kb FIFO in a deeper, less wide configuration, use the FIFO36 component. For smaller configurations of the FIFO, use the FIFO18 or FIFO18_36.

Port Descriptions

| Port | Direction | Width | Function |
|-------------|-----------|-------|---|
| DO | Output | 64 | FIFO data output bus. |
| DOP | Output | 8 | FIFO parity data output bus. |
| FULL | Output | 1 | Active high logic indicates that the FIFO contents are full. |
| ALMOSTFULL | Output | 1 | Programmable flag to indicate the FIFO is almost full. ALMOST_FULL_OFFSET attribute specifies where to trigger this flag. |
| EMPTY | Output | 1 | Active high logic indicates the FIFO is currently empty. |
| ALMOSTEMPTY | Output | 1 | Programmable flag to indicate the FIFO is almost empty. ALMOST_EMPTY_OFFSET attribute specifies where to trigger this flag. |

| Port | Direction | Width | Function |
|------------------|-----------|-------|---|
| WRERR, RDERR | Output | 1 | WRERR indicates that a write occurred while the FIFO was full while RDERR indicated a read occurred while the FIFO was empty. |
| WRCOUNT, RDCOUNT | Output | 9 | FIFO write/read pointer. |
| SBITTERR | Output | 1 | Status output from ECC function to indicate a single bit error was detected. EN_ECC_READ needs to be TRUE in order to use this functionality. |
| DBITTERR | Output | 1 | Status output from ECC function to indicate a double bit error was detected. EN_ECC_READ needs to be TRUE in order to use this functionality. |
| ECCPARITY | Output | 8 | 8-bit data generated by the ECC encoder used by the ECC decoder for memory error detection and correction. |
| DI | Input | 64 | FIFO data input bus. |
| DIP | Input | 8 | FIFO parity data input bus. |
| WREN | Input | 1 | Active high FIFO write enable. |
| RDEN | Input | 1 | Active high FIFO read enable. |
| RST | Input | 1 | Asynchronous reset (active high) of all FIFO functions, flags, and pointers. RESET must be asserted for three clock cycles. |
| WRCLK, RDCLK | Input | 1 | FIFO read and write clocks (positive edge triggered). |

Design Entry Method

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

DI, DIP, DO and DOP should be connected to their respective input and output data sources unless the FIFO is operating in ECC mode in which only the DI and DO ports should be used, since the parity bits are necessary for the ECC functionality. When you are using fewer than available data bits, connect any unused DI or DIP inputs to a logic zero and any unused DO or DOP pins should be left unconnected. When the FIFO is set to be synchronous (EN_SYN attribute is set to TRUE), the same clock source should be tied to WRCLK and RDCLK.

When you are in asynchronous mode (EN_SYN is set to FALSE), unique clock signals should be used. Depending on the offset between read and write clock edges, the Empty, Almost Empty, Full, and Almost Full flags can deassert one cycle later. Due to the asynchronous nature of the clocks the simulation model only reflects the deassertion latency cycles listed in the architecture user guide. WREN and RDEN should be connected to the respective write enable and read enable signal/logic. RST should be either tied to the appropriate reset signal/logic or connected to a logic zero if unused.

The FULL, ALMOSTFULL, EMPTY and ALMOSTEMPTY output flags should be connected to the appropriate destination logic or left unconnected if not used. The WRERR, RDERR, WRCOUNT and RDCOUNT are optional outputs and can be left unconnected, if not needed. In order to use the ECC function, the EN_ECC_READ and the EN_ECC_WRITE must be set to TRUE. If you want to monitor the error detection circuit operation, connect the SBITTERR, DBITTERR and the ECCPARITY signals to the appropriate logic. Set all attributes to the FIFO to enable the desired behavior in the component by adjusting the generics (VHDL) or in-line defparams (Verilog) in the instantiation.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------|-------------|-----------------|---------|--|
| ALMOST_FULL_OFFSET | Hexadecimal | Any 9-Bit Value | 080 | Specifies the amount of data contents in the RAM to trigger the ALMOST_FULL flag. |
| ALMOST_EMPTY_OFFSET | Hexadecimal | Any 9-Bit Value | 080 | Specifies the amount of data contents in the RAM to trigger the ALMOST_EMPTY flag. |
| FIRST_WORD_FALL_THROUGH | Boolean | TRUE, FALSE | FALSE | If TRUE, the first write to the FIFO will appear on DO without an RDEN assertion. |
| EN_SYN | Boolean | TRUE, FALSE | FALSE | When FALSE, specifies the FIFO to be used in asynchronous mode (two independent clock) or when TRUE in synchronous (a single clock) operation. |
| DO_REG | Integer | 0, 1 | 1 | Enable output register to the FIFO for improved clock-to-out timing at the expense of added read latency (one pipeline delay). DO_REG must be 1 when EN_SYN is set to FALSE. |
| EN_ECC_READ | Boolean | TRUE, FALSE | FALSE | Enable the ECC decoder circuitry. |
| EN_ECC_WRITE | Boolean | TRUE, FALSE | FALSE | Enable the ECC encoder circuitry. |
| SIM_MODE | String | "SAFE", "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |

VHDL Instantiation Template

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

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

```
-- FIFO36_72: 72x36k Synchronous/Asynchronous BlockRAM FIFO /w ECC
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
FIFO36_72_inst : FIFO36_72
generic map (
  ALMOST_FULL_OFFSET => X"0080", -- Sets almost full threshold
  ALMOST_EMPTY_OFFSET => X"0080", -- Sets the almost empty threshold
  DO_REG => 1, -- Enable output register (0 or 1)
  -- Must be 1 if EN_SYN = FALSE
  EN_ECC_READ => FALSE, -- Enable ECC decoder, TRUE or FALSE
  EN_ECC_WRITE => FALSE, -- Enable ECC encoder, TRUE or FALSE
  EN_SYN => FALSE, -- Specifies FIFO as Asynchronous (FALSE)
  -- or Synchronous (TRUE)
  FIRST_WORD_FALL_THROUGH => FALSE, -- Sets the FIFO FWFT to TRUE or FALSE
  SIM_MODE => "SAFE") -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
  -- Design Guide" for details
```

```
port map (
  ALMOSTEMPTY => ALMOSTEMPTY, -- 1-bit almost empty output flag
  ALMOSTFULL => ALMOSTFULL, -- 1-bit almost full output flag
  DBITERR => DBITERR -- 1-bit double bit error status output
  DO => DO, -- 64-bit data output
  DOP => DOP, -- 4-bit parity data output
  ECCPARITY => ECCPARITY -- 8-bit generated error correction parity
  EMPTY => EMPTY, -- 1-bit empty output flag
  FULL => FULL, -- 1-bit full output flag
  RDCOUNT => RDCOUNT, -- 9-bit read count output
  RDERR => RDERR, -- 1-bit read error output
  WRCOUNT => WRCOUNT, -- 9-bit write count output
  WRERR => WRERR, -- 1-bit write error
```

```

DI => DI,                -- 64-bit data input
DIP => DIP,               -- 4-bit parity input
RDCLK => RDCLK,           -- 1-bit read clock input
RDEN => RDEN,             -- 1-bit read enable input
RST => RST,               -- 1-bit reset input
WRCLK => WRCLK,           -- 1-bit write clock input
WREN => WREN              -- 1-bit write enable input
);

-- End of FIFO36_72_inst instantiation

```

Verilog Instantiation Template

```

// FIFO36_72: 72x36k Synchronous/Asynchronous BlockRAM FIFO w/ ECC
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FIFO36_72 #(
    .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
    .ALMOST_FULL_OFFSET(9'h080), // Sets almost full threshold
    .ALMOST_EMPTY_OFFSET(9'h080), // Sets the almost empty threshold
    .DO_REG(1), // Enable output register (0 or 1)
    // Must be 1 if EN_SYN = "FALSE"
    .EN_ECC_READ("FALSE"), // Enable ECC decoder, "TRUE" or "FALSE"
    .EN_ECC_WRITE("FALSE"), // Enable ECC encoder, "TRUE" or "FALSE"
    .EN_SYN("FALSE"), // Specifies FIFO as Asynchronous ("FALSE")
    // or Synchronous ("TRUE")
    .FIRST_WORD_FALL_THROUGH("FALSE") // Sets the FIFO FWFT to "TRUE" or "FALSE"
) FIFO36_72_inst (
    .ALMOSTEMPTY(ALMOSTEMPTY), // 1-bit almost empty output flag
    .ALMOSTFULL(ALMOSTFULL), // 1-bit almost full output flag
    .DBITERR(DBITERR), // 1-bit double bit error status output
    .DO(DO), // 64-bit data output
    .DOP(DOP), // 4-bit parity data output
    .ECCPARITY(ECCPARITY), // 8-bit generated error correction parity
    .EMPTY(EMPTY), // 1-bit empty output flag
    .FULL(FULL), // 1-bit full output flag
    .RDCOUNT(RDCOUNT), // 9-bit read count output
    .RDERR(RDERR), // 1-bit read error output
    .SBITERR(SBITERR), // 1-bit single bit error status output
    .WRCOUNT(WRCOUNT), // 9-bit write count output
    .WRERR(WRERR), // 1-bit write error
    .DI(DI), // 64-bit data input
    .DIP(DIP), // 4-bit parity input
    .RDCLK(RDCLK), // 1-bit read clock input
    .RDEN(RDEN), // 1-bit read enable input
    .RST(RST), // 1-bit reset input
    .WRCLK(WRCLK), // 1-bit write clock input
    .WREN(WREN) // 1-bit write enable input
);

// End of FIFO36_72_inst instantiation

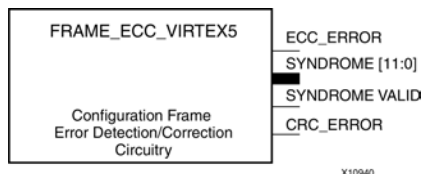
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

FRAME_ECC_VIRTEX5

Primitive: Virtex®-5 Configuration Frame Error Detection and Correction Circuitry



Introduction

This design element enables the dedicated, built-in ECC (Error Detection and Correction Circuitry) for the configuration memory of the FPGA. This element contains outputs that allow monitoring of the status of the ECC circuitry and the status of the readback CRC circuitry.

Port Descriptions

| Port | Direction | Width | Function |
|---------------|-----------|-------|---|
| ECCERROR | Output | 1 | Frame ECC error found. Value is a one when SYNDROME is non-zero and a zero when SYNDROME is all zeroes indicating no errors detected. |
| SYNDROME | Output | 12 | Frame ECC error where: <ul style="list-style-type: none"> No errors: All zeros. One bit error: SYNDROME[11]=0, SYNDROME[10:0]= location of error in FRAME. Two bit errors: SYNDROME[11]=1, SYNDROME[10:0]=don't care. More than two bit errors: Unknown output. |
| SYNDROMEVALID | Output | 1 | Frame ECC output indicating that the value on SYNDROME is valid. |
| CRCERROR | Output | 1 | Readback CRC error. |

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;

-- FRAME_ECC_VIRTEX5: Configuration Frame Error Correction Circuitry
--                               Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

FRAME_ECC_VIRTEX5_inst : FRAME_ECC_VIRTEX5
port map (
    CRCERROR => CRCERROR,    -- 1-bit output indicating a CRC error
    ECCERROR => ECCERROR,    -- 1-bit output indicating an ECC error
    SYNDROME => SYNDROME,    -- 12-bit output location of erroneous bit
    SYNDROMEVALID => SYNDROMEVALID -- 1-bit output indicating the
                                -- SYNDROME output is valid
);

-- End of FRAME_ECC_VIRTEX5_inst instantiation
```

Verilog Instantiation Template

```
// FRAME_ECC_VIRTEX5: Configuration Frame Error Correction Circuitry
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

FRAME_ECC_VIRTEX5 FRAME_ECC_VIRTEX5_inst (
    .CRCERROR(CRCERROR), // 1-bit output indicating a CRC error
    .ECCERROR(ECCERROR), // 1-bit output indicating an ECC error
    .SYNDROME(SYNDROME), // 12-bit output location of erroneous bit
    .SYNDROMEVALID(SYNDROMEVALID) // 1-bit output indicating the
                                // SYNDROME output is valid
);

// End of FRAME_ECC_VIRTEX5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

GTP_DUAL

Primitive: Dual Gigabit Transceiver

Introduction

This design element is a power-efficient transceiver for Virtex®-5 FPGAs. The GTP transceiver is highly configurable and tightly integrated with the programmable logic resources of the FPGA.

Design Entry Method

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

For More Information

- See the [Virtex-5 FPGA RocketIO GTP Transceivers User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).
- See the [Virtex-5 FPGA User Guide](#).

GTX_DUAL

Primitive: Dual Gigabit Transceiver

Introduction

This design element is a power-efficient transceiver for Virtex®-5 FPGAs. The GTX transceiver is highly configurable and tightly integrated with the programmable logic resources of the FPGA.

Design Entry Method

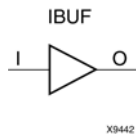
| | |
|-----------------------------|-------------|
| Instantiation | No |
| Inference | No |
| CORE Generator™ and wizards | Recommended |
| Macro support | No |

For More Information

- See the [Virtex-5 FPGA RocketIO GTX Transceivers User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).
- See the [Virtex-5 FPGA User Guide](#).

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/default values in order to configure the proper behavior of the buffer.

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;

-- IBUF: Single-ended Input Buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

IBUF_inst : IBUF
generic map (
    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
//      All devices
// Xilinx HDL Libraries Guide, version 12.3

IBUF #(
    .IOSTANDARD("DEFAULT")    // Specify the input I/O standard
)IBUF_inst (
    .O(O),                    // Buffer output
    .I(I)                     // Buffer input (connect directly to top-level port)
);

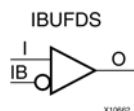
// End of IBUF_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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. |
| IOSTANDARD | String | See Data Sheet. | "DEFAULT" | Assigns an I/O standard to the element. |

VHDL Instantiation Template

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

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

-- IBUFDS: Differential Input Buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of IBUFDS_inst instantiation
```

Verilog Instantiation Template

```
// IBUFDS: Differential Input Buffer
//      Virtex-5, Spartan-3/3E/3A
// Xilinx HDL Libraries Guide, version 12.3

IBUFDS #(
    .DIFF_TERM("FALSE"),          // Differential Termination (Virtex-5, Spartan-3E/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)
);

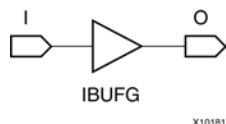
// End of IBUFDS_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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 |
|------------|--------|----------------|-----------|---|
| IOSTANDARD | String | See Data Sheet | "DEFAULT" | Assigns an I/O standard to the element. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- IBUFG: Single-ended global clock input buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

IBUFG_inst : IBUFG
generic map (
    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
//      All FPGA
// Xilinx HDL Libraries Guide, version 12.3

IBUFG #(
    .IOSTANDARD("DEFAULT") // Specify the input I/O standard
) IBUFG_inst (
    .O(O), // Clock buffer output
    .I(I) // Clock buffer input (connect directly to top-level port)
);

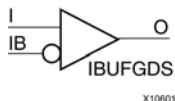
// End of IBUFG_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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. |
| IOSTANDARD | String | See Data Sheet | "DEFAULT" | Assigns an I/O standard to the element. |

VHDL Instantiation Template

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

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

-- IBUFGDS: Differential Global Clock Input Buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of IBUFGDS_inst instantiation
```

Verilog Instantiation Template

```
// IBUFGDS: Differential Global Clock Input Buffer
//      Virtex-5, Spartan-3/3E/3A
// Xilinx HDL Libraries Guide, version 12.3

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

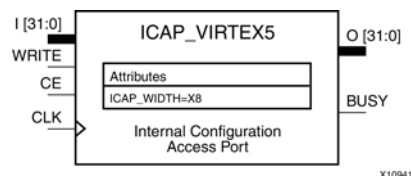
// End of IBUFGDS_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

ICAP_VIRTEX5

Primitive: Internal Configuration Access Port



Introduction

This design element gives you access to the configuration functions of the FPGA from the FPGA fabric. Using this component, commands and data can be written to and read from the configuration logic of the FPGA array. Since the improper use of this function can have a negative effect on the functionality and reliability of the FPGA, you shouldn't use this element unless you are very familiar with its capabilities.

Port Descriptions

| Port | Direction | Width | Function |
|-------|-----------|-------|-------------------------------|
| O | Output | 32 | Configuration data output bus |
| Busy | Output | 1 | Busy/Ready output |
| I | Input | 32 | Configuration data input bus |
| WRITE | Input | 1 | Active Low Write Input |
| CE | Input | 1 | Active Low Clock Enable Input |
| CLK | Input | 1 | Clock Input |

Design Entry Method

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

Refer to the Configuration User Guide for more details about the parallel bus bit order.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|------------|--------|--------------------|---------|---|
| ICAP_WIDTH | String | "X8", "X16", "X32" | "X8" | Specifies the input and output data width to be used with the ICAP_VIRTEX5. |

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_VIRTEX5: Internal Configuration Access Port
--                Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

ICAP_VIRTEX5_inst : ICAP_VIRTEX5
generic map (
    ICAP_WIDTH => "X8") -- "X8", "X16" or "X32"
port map (
    BUSY => BUSY,      -- Busy output
    O => O,            -- 32-bit data output
    CE => CE,          -- Clock enable input
    CLK => CLK,        -- Clock input
    I => I,            -- 32-bit data input
    WRITE => WRITE     -- Write input
);

-- End of ICAP_VIRTEX5_inst instantiation
```

Verilog Instantiation Template

```
// ICAP_VIRTEX5: Internal Configuration Access Port
//                Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

ICAP_VIRTEX5 #(
    .ICAP_WIDTH("X8") // "X8", "X16" or "X32"
) ICAP_VIRTEX5_inst (
    .BUSY(BUSY),      // Busy output
    .O(O),            // 32-bit data output
    .CE(CE),          // Clock enable input
    .CLK(CLK),        // Clock input
    .I(I),            // 32-bit data input
    .WRITE(WRITE)     // Write input
);

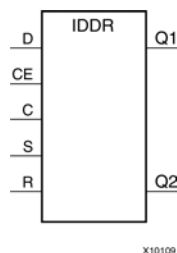
// End of ICAP_VIRTEX5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

IDDR

Primitive: Input Dual Data-Rate Register



Introduction

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

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

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

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

Port Descriptions

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

Design Entry Method

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

Available Attributes

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

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- IDDR: Double Data Rate Input Register with Set, Reset
--       and Clock Enable.
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of IDDR_inst instantiation

```

Verilog Instantiation Template

```
// IDDR: Input Double Data Rate Input Register with Set, Reset
//      and Clock Enable.
//      Virtex-5/6
// Xilinx HDL Libraries Guide, version 12.3

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

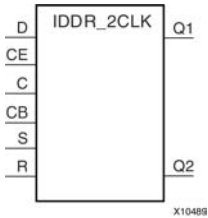
// End of IDDR_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

IDDR_2CLK

Primitive: Input Dual Data-Rate Register with Dual Clock Inputs



Introduction

This design element is a dedicated input register designed to receive external dual data rate (DDR) signals into Xilinx® FPGAs. In general, you should only use the IDDR_2CLK sparingly (it was intended for QDR II applications), since it requires more clocking resources, more power, and can imply certain placement restrictions that are not necessary when using the IDDR component. Alternatively, the IDDR component is easier to use, uses fewer resources, and has fewer restrictions. The IDDR_2CLK is available with modes that present the data to the FPGA fabric at the time and clock edge they are captured, or on the same clock edge. This feature allows designers to avoid additional timing complexities and resource usage.

- **OPPOSITE_EDGE mode** - Data is presented in the classic DDR methodology. Given a DDR data and clock at pin D and C respectively, Q1 changes after every positive edge of clock C, and Q2 changes after every positive edge of clock CB.
- **SAME_EDGE mode** - Data is still presented by positive edges of each clock. However, an extra register has been placed in front of the CB clocked data register. This extra register is clocked with positive clock edge of clock signal C. As a result, DDR data is now presented into the FPGA fabric at the positive edge of clock C. However, because of this feature, the data pair appears to be "separated." Q1 and Q2 no longer have pair 1 and 2. Instead, the first pair presented is Pair 1 and DON'T CARE, followed by Pair 2 and 3 at the next clock cycle.
- **SAME_EDGE_PIPELINED mode** - Presents data in a similar fashion as the SAME_EDGE mode. In order to avoid the "separated" effect of the SAME_EDGE mode, an extra register has been placed in front of the C clocked data register. A data pair now appears at the Q1 and Q2 pin at the same time during the positive edge of C. However, using this mode, costs you an additional cycle of latency for Q1 and Q2 signals to change.

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

Port Descriptions

| Port | Direction | Width | Function |
|---------|-----------|-------|---|
| Q1 : Q2 | Output | 1 | These pins are the IDDR output that connects to the FPGA fabric. Q1 is the first data pair and Q2 is the second data pair. |
| C | Input | 1 | Primary clock input pin used to capture the positive edge data. |
| CB | Input | 1 | Secondary clock input pin (typically 180 degrees out of phase with the primary clock) used to capture the negative edge data. |
| CE | Input | 1 | When asserted Low, this port disables the output clock at port O. |
| D | Input | 1 | This pin is where the DDR data is presented into the IDDR module. This pin connects to a top-level input or bi-directional port, and IODELAY configured for an input delay or to an appropriate input or bidirectional buffer. |

| Port | Direction | Width | Function |
|------|-----------|-------|--|
| R | Input | 1 | Active high reset forcing Q1 and Q2 to a logic zero. Can be synchronous or asynchronous based on the SRTYPE attribute. |
| S | Input | 1 | Active high reset forcing Q1 and Q2 to a logic one. Can be synchronous or asynchronous based on the SRTYPE attribute. |

Design Entry Method

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

- Connect the C pin to the appropriate clock source, representing the positive clock edge and CB to the clock source representing the negative clock edge.
- Connect the D pin to the top-level input, or bidirectional port, an IODELAY, or an instantiated input or bidirectional buffer.
- The Q1 and Q2 pins should be connected to the appropriate data sources.
- CE should be tied high when not used, or connected to the appropriate clock enable logic.
- R and S pins should be tied low, if not used, or to the appropriate set or reset generation logic.
- Set all attributes to the component to represent the desired behavior.
- Always instantiate this component in pairs with the same clocking, and to LOC those to the appropriate P and N I/O pair in order not to sacrifice possible I/O resources.
- Always instantiate this component in the top-level hierarchy of your design, along with any other instantiated I/O components for the design. This helps facilitate hierarchical design flows/practices.
- To minimize CLK skew, both CLK and CLKB should come from global routing (DCM / MMCM) and not from the local inversion. DCM / MMCM de-skews these clocks whereas the local inversion adds skew.

Available Attributes

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

VHDL Instantiation Template

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

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

-- IDDR_2CLK: Dual-Clock, Input Double Data Rate Input Register with
--           Set, Reset and Clock Enable.
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

IDDR_2CLK_inst : IDDR_2CLK
generic map (
    DDR_CLK_EDGE => "OPPOSITE_EDGE", -- "OPPOSITE_EDGE", "SAME_EDGE"
                                     -- or "SAME_EDGE_PIPELINED"
    INIT_Q1 => '0', -- Initial value of Q1: '0' or '1'
    INIT_Q2 => '0', -- Initial value of Q2: '0' or '1'
    SRTYPE => "SYNC") -- Set/Reset type: "SYNC" or "ASYN"
port map (
    Q1 => Q1, -- 1-bit output for positive edge of clock
    Q2 => Q2, -- 1-bit output for negative edge of clock
    C => C,   -- 1-bit primary clock input
    CB => CB, -- 1-bit secondary clock input
    CE => CE, -- 1-bit clock enable input
    D => D,   -- 1-bit DDR data input
    R => R,   -- 1-bit reset
    S => S    -- 1-bit set
);

-- End of IDDR_2CLK_inst instantiation
```

Verilog Instantiation Template

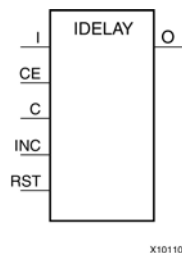
```
// IDDR_2CLK: Dual-Clock, Input Double Data Rate Input Register with
//           Set, Reset and Clock Enable.
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

IDDR_2CLK #(
    .DDR_CLK_EDGE("OPPOSITE_EDGE"), // "OPPOSITE_EDGE", "SAME_EDGE"
                                     // or "SAME_EDGE_PIPELINED"
    .INIT_Q1(1'b0), // Initial value of Q1: 1'b0 or 1'b1
    .INIT_Q2(1'b0), // Initial value of Q2: 1'b0 or 1'b1
    .SRTYPE("SYNC") // Set/Reset type: "SYNC" or "ASYN"
) IDDR_2CLK_inst (
    .Q1(Q1), // 1-bit output for positive edge of clock
    .Q2(Q2), // 1-bit output for negative edge of clock
    .C(C),   // 1-bit primary clock input
    .CB(CB), // 1-bit secondary clock input
    .CE(CE), // 1-bit clock enable input
    .D(D),   // 1-bit DDR data input
    .R(R),   // 1-bit reset
    .S(S)    // 1-bit set
);

// End of IDDR_2CLK_inst instantiation
```

IDELAY

Primitive: Input Delay Element



Introduction

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

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

Port Descriptions

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

Data Input and Output - I and O

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

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

The regional clock buffer, BUFIO, connects the incoming regional clock signal to the regional I/O clock tree, IOCLK. BUFIO also connects to the regional clock buffer, BUFR to connect to the regional clock tree, rclk. The input and output datapath is combinatorial and is not affected by the clock signal (C). However, you can choose to register the output signal (O) in the IOB.

- **Global clock-capable IOBs** - Global clock-capable IOBs are located in the center I/O column. The input of the IDELAY module in a global clock-capable IOB comes directly from the input global clock buffer, IBUFG. The output of the IDELAY module in a global clock-capable IOB can go to one of the following locations:
 - Directly to your logic
 - BUFG (in the case of a global clock signal)

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

Clock Input - C

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

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

Module Reset - RST

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

Increment/Decrement Signals - CE, INC

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

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

Note

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

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

Design Entry Method

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

Available Attributes

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

IOBDELAY_TYPE Attribute

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

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

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

IOBDELAY_VALUE Attribute

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

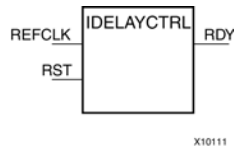
The value of the tap delay reverts to IOBDELAY_VALUE when the tap delay is reset (RST = 1), or the IOBDELAY_TYPE is set to FIXED.

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

IDELAYCTRL

Primitive: IDELAY Tap Delay Value Control



Introduction

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

Port Descriptions

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

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

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

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

Design Entry Method

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

VHDL Instantiation Template

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

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

-- IDELAYCTRL : Input Delay Element Control
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of IDELAYCTRL_inst instantiation
```

Verilog Instantiation Template

```
// IDELAYCTRL: Input Delay Control Element (Must be used in conjunction with the IDELAY
//           when used in FIXED or VARIABLE tap-delay mode)
//           Virtex-5/6
// Xilinx HDL Libraries Guide, version 12.3

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

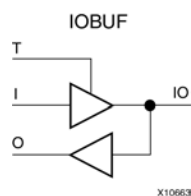
// End of IDELAYCTRL_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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. |
| IOSTANDARD | String | See Data Sheet | "DEFAULT" | Assigns an I/O standard to the element. |
| SLEW | String | "SLOW", "FAST", "QUIETIO" | "SLOW" | Sets the output rise and fall time. See the Data Sheet for recommendations of the best setting for this attribute. |

VHDL Instantiation Template

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

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

-- IOBUF: Single-ended Bi-directional Buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of IOBUF_inst instantiation
```

Verilog Instantiation Template

```
// IOBUF: Single-ended Bi-directional Buffer
//      All devices
// Xilinx HDL Libraries Guide, version 12.3

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

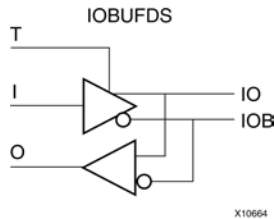
// End of IOBUF_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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 |
|------------|--------|----------------|-----------|---|
| IOSTANDARD | String | See Data Sheet | "DEFAULT" | Assigns an I/O standard to the element. |

VHDL Instantiation Template

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

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

-- IOBUFDS: Differential Bi-directional Buffer
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of IOBUFDS_inst instantiation
```

Verilog Instantiation Template

```
// IOBUFDS: Differential Bi-directional Buffer
//      Virtex-5, Spartan-3/3E/3A
// Xilinx HDL Libraries Guide, version 12.3

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

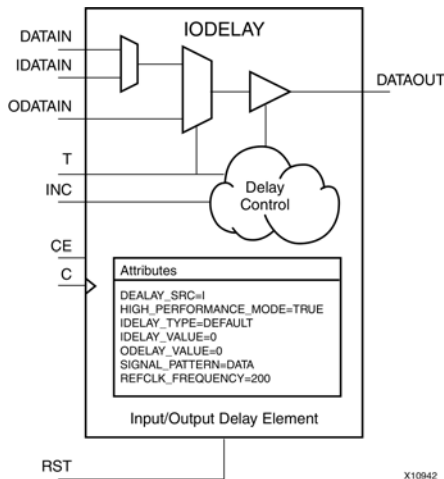
// End of IOBUFDS_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

IODELAY

Primitive: Input and Output Fixed or Variable Delay Element



Introduction

This design element can be used to provide a fixed delay or an adjustable delay to the input path and a fixed delay for the output path of the FPGA. This delay can be useful for the purpose of data alignment of incoming or outgoing data to/from the chip, as well as allowing for the tracking of data alignment over process, temperature, and voltage (PVT). The IODELAY is available on all FPGA I/Os and, when used in conjunction with the IDELAYCTRL component circuitry, can provide precise time increments of delay. When used in variable mode, the input path can be adjusted for increasing and decreasing amounts of delay. The output delay path is only available in a fixed delay. The IODELAY can also be used to add additional static or variable delay to an internal path (within the FPGA fabric). However, when IODELAY is used that way, this device is no longer available to the associated I/O for input or output path delays.

Port Descriptions

| Port | Direction | Width | Function |
|---------|-----------|-------|---|
| DATAOUT | Output | 1 | Delayed data output from input port (connect to input datapath logic) |
| IDATAIN | Input | 1 | Data input to device from the I/O (connect directly to port, I/O Buffer). When IDATAIN is used, DATAIN must be tied to a logic zero (ground). |
| ODATAIN | Input | 1 | Data input for the output datapath from the device (connect to output data source). When ODATAIN is used, DATAIN must be tied to a logic zero (ground). |
| DATAIN | Input | 1 | Data input for the internal datapath delay. When DATAIN is used, IDATAIN and ODATAIN must be tied to a logic zero (ground). |
| T | Input | 1 | 3-state input control. Tie high for input-only or internal delay or tie low for output only. |
| CE | Input | 1 | Active high enable increment/decrement function |
| INC | Input | 1 | Increment / Decrement tap delay |
| C | Input | 1 | Clock input (Must be connected for variable mode) |
| RST | Input | 1 | Active high, synchronous reset, resets delay chain to IDELAY_VALUE/ ODELAY_VALUE tap. If no value is specified, the default is 0. |

Design Entry Method

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

For input delay operation, connect the IDATAIN pin directly to either the top-level I/O port, input buffer, or I/O buffer. For output delay, connect the ODATAIN input to the logic sourcing the output data to be delayed. For internal path delays, connect the DATAIN pin to the proper source and destination logic within the FPGA. When you are using the IODELAY for internal signal delays, the IDATAIN and ODATAIN must be tied to a logic zero (ground).

In all cases, the DATAOUT should be connected to the I/Os or logic to be sourced from the delayed data. Connect the T pin to the control signal for the 3-state output operation when you are using the IODELAY. If you are using the IODELAY for output delays only, tie the T pin to a logic zero (ground). If you are using the IODELAY for input only, or for delaying an internal signal, tie the T pin to a logic one (Vcc). If the IODELAY is configured for VARIABLE delay, connect the CE, INC, C, and RST pins to the appropriate delay control signals. If only a FIXED delay mode is used, those pins should be tied to a logic zero (ground).

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------------------|---------|----------------------------------|-----------|---|
| HIGH_PERFORMANCE_MODE | Boolean | TRUE, FALSE | FALSE | When TRUE, this attribute reduces the output jitter. |
| DELAY_SRC | String | "I", "O", "IO" or "DATAIN" | "I" | Specifies the source to the IODELAY component. "I" means it will be connected directly to an input port or IBUF (input mode), "O" means it will be connected to an output port or OBUF (output mode), "IO" means it will be connected to a port, and "DATAIN" means it will not be connected to any port (internal mode). |
| IDELAY_TYPE | String | "DEFAULT", "FIXED" or "VARIABLE" | "DEFAULT" | Specifies a fixed, variable or default (eliminate hold time) input delay. |
| IDELAY_VALUE | Integer | 0 to 63 | 0 | Specifies the number of taps of delay for the input path when in fixed mode or the initial delay tap value for variable mode. |
| ODELAY_VALUE | Integer | 0 to 63 | 0 | Specifies the number of taps of delay for the output path. |
| REFCLK_FREQUENCY | Real | 190.00 to 210.00 | 200.00 | When using an associated IDELAYCTRL, specifies the input reference frequency to the component. |
| SIGNAL_PATTERN | String | "CLOCK", "DATA" | "DATA" | Used by the delay calculator to determine different propagation delays through the IODELAY block based on the setting. DATA will be the addition of per tap delay and per tap jitter. No jitter is introduced for clock-like signals. |

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;

-- IODELAY: Input and/or Output Fixed/Variable Delay Element
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

IODELAY_inst : IODELAY
generic map (
    DELAY_SRC => "I", -- Specify which input port to be used
                    -- "I"=IDATAIN, "O"=ODATAIN, "DATAIN"=DATAIN, "IO"=Bi-directional
    HIGH_PERFORMANCE_MODE => TRUE, -- TRUE specifies lower jitter
                    -- at expense of more power
    IDELAY_TYPE => "FIXED", -- "FIXED" or "VARIABLE"
    IDELAY_VALUE => 0, -- 0 to 63 tap values
    ODELAY_VALUE => 0, -- 0 to 63 tap values
    REFCLK_FREQUENCY => 200.0, -- Frequency used for IDELAYCTRL
                    -- 175.0 to 225.0
    SIGNAL_PATTERN => "DATA") -- Input signal type, "CLOCK" or "DATA"
port map (
    DATAOUT => DATAOUT, -- 1-bit delayed data output
    C => C, -- 1-bit clock input
    CE => CE, -- 1-bit clock enable input
    DATAIN => DATAIN, -- 1-bit internal data input
    IDATAIN => IDATAIN, -- 1-bit input data input (connect to port)
    INC => INC, -- 1-bit increment/decrement input
    ODATAIN => ODATAIN, -- 1-bit output data input
    RST => RST, -- 1-bit active high, synch reset input
    T => T -- 1-bit 3-state control input
);

-- End of IODELAY_inst instantiation
```

Verilog Instantiation Template

```
// IODELAY: Input and/or Output Fixed/variable Delay Element
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

(* IODELAY_GROUP = "<iodelay_group_name>" *) // Specifies group name for associated IODELAYs and IDELAYCTRL
IODELAY # (
    .DELAY_SRC("I"), // Specify which input port to be used, "I"=IDATAIN,
                    // "O"=ODATAIN, "DATAIN"=DATAIN, "IO"=Bi-directional
    .HIGH_PERFORMANCE_MODE("TRUE"), // "TRUE" specifies lower jitter
                    // at expense of more power
    .IDELAY_TYPE("FIXED"), // "FIXED" or "VARIABLE"
    .IDELAY_VALUE(0), // 0 to 63 tap values
    .ODELAY_VALUE(0), // 0 to 63 tap values
    .REFCLK_FREQUENCY(200.0), // Frequency used for IDELAYCTRL
                    // 175.0 to 225.0
    .SIGNAL_PATTERN("DATA") // Input signal type, "CLOCK" or "DATA"
) IODELAY_INST (
    .DATAOUT(DATAOUT), // 1-bit delayed data output
    .C(C), // 1-bit clock input
    .CE(CE), // 1-bit clock enable input
    .DATAIN(DATAIN), // 1-bit internal data input
    .IDATAIN(IDATAIN), // 1-bit input data input (connect to port)
    .INC(INC), // 1-bit increment/decrement input
    .ODATAIN(ODATAIN), // 1-bit output data input
    .RST(RST), // 1-bit active high, synch reset input
    .T(T) // 1-bit 3-state control input
);

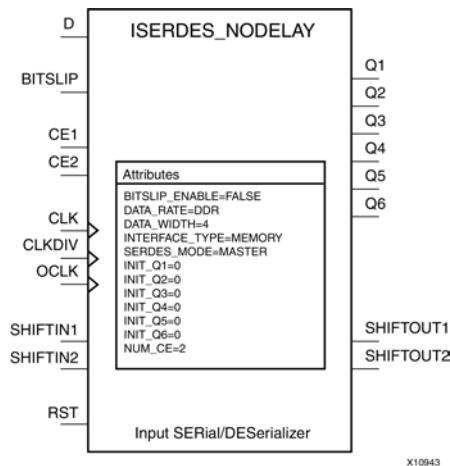
// End of IODELAY_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

ISERDES_NODELAY

Primitive: Input SERIAL/DESerializer



Introduction

The ISERDES_NODELAY is an input serial-to-parallel data converter that helps facilitate high-speed, source synchronous, serial data capturing. The ISERDES_NODELAY includes logic to assist in clocking and data alignment of either single data rate (SDR) or double data rate (DDR) data to/from 2- to 6-bit data widths for a single instance (MASTER) and 7- to 10-bit data widths for two cascaded ISERDES_NODELAY (MASTER/SLAVE). The ISERDES_NODELAY can be used in memory, networking or a number of different types of data interface applications. The ISERDES_NODELAY can be used in conjunction with an IODELAY component to assist in data alignment of the input serial data. In DDR mode, the ISERDES_NODELAY can be clocked by either a single clock or two clocks for capturing data. When you are using it in two clock mode, higher performance is possible. However, using it in this way might require more clocking resources, consume more power, and require certain placement restriction. Use single clock mode when the highest I/O performance is not needed.

Port Descriptions

| Port | Direction | Width | Function |
|-----------------------|-----------|-------|---|
| Q1 - Q6 | Output | 1 | Registered parallelized input data. |
| SHIFTOUT1 / SHIFTOUT2 | Output | 1 | If ISERDES_MODE="MASTER" and two ISERDES_NODELAY are to be cascaded, connect to the slave ISERDES_NODELAY IDATASHIFTIN1/2 inputs. |
| D | Input | 1 | Input data to be connected directly to the top-level input or I/O port of the design or to an IODELAY component if additional input delay control is desired. |
| BITSLLIP | Input | 1 | Input data BITSLLIP function enable. |
| CE1 / CE2 | Input | 1 | Input data register clock enables. |
| CLK | Input | 1 | Primary clock input pin used. |
| CLKB | Input | 1 | The bit ordering at the input of an OSERDES is the opposite of the bit ordering at the output of an ISERDES_NODELAY block. Please see the appropriate device user guide for detailed information. |

| Port | Direction | Width | Function |
|---------------------|-----------|-------|--|
| CLKDIV | Input | 1 | Divided clock to be used for parallelized data. |
| OCLK | Input | 1 | High speed output clock typically used for memory interfaces. |
| SHIFTIN1 / SHIFTIN2 | Input | 1 | If ISERDES_MODE="SLAVE" connect to the master ISERDES_NODELAY IDATASHIFTOUT1/2 outputs. This pin must be grounded. |
| RST | Input | 1 | Active high asynchronous reset signal for the registers of the SERDES. |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|----------------|---------|--|----------|--|
| BITSLIP_ENABLE | Boolean | TRUE or FALSE | FALSE | Enable the BITSLIP functionality. Only available in NETWORKING mode. |
| DATA_RATE | String | "SDR" or "DDR" | "DDR" | Single Data Rate or Double Data Rate operation |
| DATA_WIDTH | Integer | 4,6,8 or 10 if DATA_RATE="DDR", 2,3,4,5,6,7 or 8 if DATA_RATE="SDR" | 4 | Parallel data width selection |
| INTERFACE_TYPE | String | "MEMORY" or "NETWORKING" | "MEMORY" | Memory or Networking interface type |
| SERDES_MODE | String | "MASTER" or "SLAVE" | "MASTER" | Specify whether the ISERDES is operating in master or slave modes when cascaded width expansion. |
| NUM_CE | Integer | 1 or 2 | 2 | Specifies the number of clock enables used for the ISERDES_NODELAY. |

VHDL Instantiation Template

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

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

```
-- ISERDES_NODELAY: Input Serial / DESerializer
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
ISERDES_NODELAY_inst : ISERDES_NODELAY
generic map (
  BITSLIP_ENABLE => FALSE, -- TRUE/FALSE to enable bitflip controller
  -- Must be "FALSE" in interface type is "MEMORY"
```



```

DATA_RATE => "DDR", -- Specify data rate of "DDR" or "SDR"
DATA_WIDTH => 4, -- Specify data width -
                -- NETWORKING SDR: 2, 3, 4, 5, 6, 7, 8 : DDR 4, 6, 8, 10
                -- MEMORY SDR N/A : DDR 4
INTERFACE_TYPE => "MEMORY", -- Use model - "MEMORY" or "NETWORKING"
NUM_CE => 2, -- Define number of clock enables to an integer of 1 or 2
SERDES_MODE => "MASTER") --Set SERDES mode to "MASTER" or "SLAVE"
port map (
  Q1 => Q1, -- 1-bit registered SERDES output
  Q2 => Q2, -- 1-bit registered SERDES output
  Q3 => Q3, -- 1-bit registered SERDES output
  Q4 => Q4, -- 1-bit registered SERDES output
  Q5 => Q5, -- 1-bit registered SERDES output
  Q6 => Q6, -- 1-bit registered SERDES output
  SHIFTOUT1 => SHIFTOUT1, -- 1-bit cascade Master/Slave output
  SHIFTOUT2 => SHIFTOUT2, -- 1-bit cascade Master/Slave output
  BITSLIP => BITSLIP, -- 1-bit Bitslip enable input
  CE1 => CE1, -- 1-bit clock enable input
  CE2 => CE2, -- 1-bit clock enable input
  CLK => CLK, -- 1-bit master clock input
  CLKB => CLKB, -- 1-bit secondary clock input for DATA_RATE=DDR
  CLKDIV => CLKDIV, -- 1-bit divided clock input
  D => D, -- 1-bit data input, connects to IODELAY or input buffer
  OCLK => OCLK, -- 1-bit fast output clock input
  RST => RST, -- 1-bit asynchronous reset input
  SHIFTIN1 => SHIFTIN1, -- 1-bit cascade Master/Slave input
  SHIFTIN2 => SHIFTIN2 -- 1-bit cascade Master/Slave input
);

-- End of ISERDES_NODELAY_inst instantiation

```

Verilog Instantiation Template

```

// ISERDES_NODELAY: Input Serial / DESerializer
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

ISERDES_NODELAY #(
  .BITSLIP_ENABLE("FALSE"), // "TRUE"/"FALSE" to enable bitslip controller
                             // Must be "FALSE" if INTERFACE_TYPE set to "MEMORY"
  .DATA_RATE("DDR"),        // Specify data rate of "DDR" or "SDR"
  .DATA_WIDTH(4),           // Specify data width -
                             // NETWORKING SDR: 2, 3, 4, 5, 6, 7, 8 : DDR 4, 6, 8, 10
                             // MEMORY SDR N/A : DDR 4
  .INTERFACE_TYPE("MEMORY"), // Use model - "MEMORY" or "NETWORKING"
  .NUM_CE(2),               // Number of clock enables used, 1 or 2
  .SERDES_MODE("MASTER")    // Set SERDES mode to "MASTER" or "SLAVE"
) ISERDES_NODELAY_inst (
  .Q1(Q1), // 1-bit registered SERDES output
  .Q2(Q2), // 1-bit registered SERDES output
  .Q3(Q3), // 1-bit registered SERDES output
  .Q4(Q4), // 1-bit registered SERDES output
  .Q5(Q5), // 1-bit registered SERDES output
  .Q6(Q6), // 1-bit registered SERDES output
  .SHIFTOUT1(SHIFTOUT1), // 1-bit cascade Master/Slave output
  .SHIFTOUT2(SHIFTOUT2), // 1-bit cascade Master/Slave output
  .BITSLIP(BITSLIP),     // 1-bit Bitslip enable input
  .CE1(CE1),             // 1-bit clock enable input
  .CE2(CE2),             // 1-bit clock enable input
  .CLK(CLK),             // 1-bit master clock input
  .CLKB(CLKB),           // 1-bit secondary clock input for DATA_RATE=DDR
  .CLKDIV(CLKDIV),       // 1-bit divided clock input
  .D(D),                 // 1-bit data input, connects to IODELAY or input buffer
  .OCLK(OCLK),           // 1-bit fast output clock input
  .RST(RST),             // 1-bit asynchronous reset input
  .SHIFTIN1(SHIFTIN1),   // 1-bit cascade Master/Slave input
  .SHIFTIN2(SHIFTIN2)    // 1-bit cascade Master/Slave input
);

// End of ISERDES_NODELAY_inst instantiation

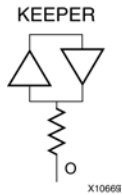
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

KEEPER

Primitive: KEEPER Symbol



Introduction

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

Port Descriptions

| Name | Direction | Width | Function |
|------|-----------|-------|---------------|
| O | Output | 1-Bit | Keeper output |

Design Entry Method

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

VHDL Instantiation Template

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

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

-- KEEPER: I/O Buffer Weak Keeper
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//           All FPGA, CoolRunner-II
// Xilinx HDL Libraries Guide, version 12.3

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

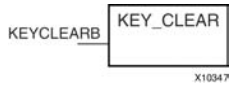
// End of KEEPER_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

KEY_CLEAR

Primitive: Virtex-5 Configuration Encryption Key Erase



Introduction

This design element allows you to erase the configuration encryption circuit key register from internal logic.

Port Descriptions

| Port | Direction | Width | Function |
|-----------|-----------|-------|---|
| KEYCLEARB | Input | 1 | Active low input, clears the configuration encryption key |

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;

-- KEY_CLEAR: Startup primitive for GSR, GTS or startup sequence control
--             Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

KEY_CLEAR_inst : KEY_CLEAR
  port map (
    KEYCLEARB => KEYCLEARB -- Active low key reset 1-bit input
  );

-- End of KEY_CLEAR_inst instantiation
```

Verilog Instantiation Template

```
// KEY_CLEAR: Startup primitive for GSR, GTS or startup sequence control
//             Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

KEY_CLEAR KEY_CLEAR_inst (
  .KEYCLEARB(KEYCLEARB) // Active low key reset 1-bit input
);

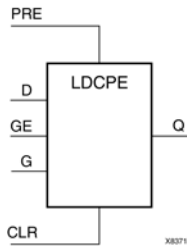
// End of KEY_CLEAR_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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.
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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

-- End of LDCPE_inst instantiation
```

Verilog Instantiation Template

```
// LDCPE: Transparent latch with Asynchronous Reset, Preset and
//       Gate Enable.
//       Virtex-5, Spartan-3/3E/3A/3A DSP
// Xilinx HDL Libraries Guide, version 12.3

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

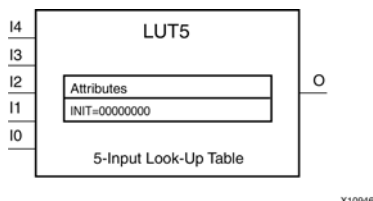
// End of LDCPE_inst instantiation
```


For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT5

Primitive: 5-Input Lookup Table with General Output



Introduction

This design element is a 5-input, 1-output look-up table (LUT) that can either act as an asynchronous 32-bit ROM (with 5-bit addressing) or implement any 5-input logic function. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. One LUT5 is packed into a LUT6 within a slice, or two LUT5s can be packed into a single LUT6 with some restrictions. The functionality of the LUT5, LUT5_L and LUT5_D is the same. However, the LUT5_L and LUT5_D allow the additional specification to connect the LUT5 output signal to an internal slice or CLB connection using the LO output. The LUT5_L specifies that the only connections from the LUT5 will be within a slice or CLB, while the LUT5_D allows the specification to connect the output of the LUT to both inter-slice/CLB logic and external logic as well. The LUT5 does not state any specific output connections and should be used in all cases except where internal slice or CLB signal connections must be implicitly specified.

An INIT attribute consisting of a 32-bit hexadecimal value must be specified to indicate the LUTs logical function. The INIT value is calculated by assigning a 1 to the corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 32'h80000000 (X"80000000" for VHDL) makes the output zero unless all of the inputs are one (a 5-input AND gate). A Verilog INIT value of 32'hfffffffe (X"FFFFFFFE" for VHDL) makes the output one unless all zeros are on the inputs (a 5-input OR gate).

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 |
|---|----|----|----|----|----------|
| I4 | I3 | I2 | I1 | I0 | LO |
| 0 | 0 | 0 | 0 | 0 | INIT[0] |
| 0 | 0 | 0 | 0 | 1 | INIT[1] |
| 0 | 0 | 0 | 1 | 0 | INIT[2] |
| 0 | 0 | 0 | 1 | 1 | INIT[3] |
| 0 | 0 | 1 | 0 | 0 | INIT[4] |
| 0 | 0 | 1 | 0 | 1 | INIT[5] |
| 0 | 0 | 1 | 1 | 0 | INIT[6] |
| 0 | 0 | 1 | 1 | 1 | INIT[7] |
| 0 | 1 | 0 | 0 | 0 | INIT[8] |
| 0 | 1 | 0 | 0 | 1 | INIT[9] |
| 0 | 1 | 0 | 1 | 0 | INIT[10] |
| 0 | 1 | 0 | 1 | 1 | INIT[11] |
| 0 | 1 | 1 | 0 | 0 | INIT[12] |
| 0 | 1 | 1 | 0 | 1 | INIT[13] |
| 0 | 1 | 1 | 1 | 0 | INIT[14] |
| 0 | 1 | 1 | 1 | 1 | INIT[15] |
| 1 | 0 | 0 | 0 | 0 | INIT[16] |
| 1 | 0 | 0 | 0 | 1 | INIT[17] |
| 1 | 0 | 0 | 1 | 0 | INIT[18] |
| 1 | 0 | 0 | 1 | 1 | INIT[19] |
| 1 | 0 | 1 | 0 | 0 | INIT[20] |
| 1 | 0 | 1 | 0 | 1 | INIT[21] |
| 1 | 0 | 1 | 1 | 0 | INIT[22] |
| 1 | 0 | 1 | 1 | 1 | INIT[23] |
| 1 | 1 | 0 | 0 | 0 | INIT[24] |
| 1 | 1 | 0 | 0 | 1 | INIT[25] |
| 1 | 1 | 0 | 1 | 0 | INIT[26] |
| 1 | 1 | 0 | 1 | 1 | INIT[27] |
| 1 | 1 | 1 | 0 | 0 | INIT[28] |
| 1 | 1 | 1 | 0 | 1 | INIT[29] |
| 1 | 1 | 1 | 1 | 0 | INIT[30] |
| 1 | 1 | 1 | 1 | 1 | INIT[31] |
| INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute | | | | | |

Port Description

| Name | Direction | Width | Function |
|--------------------|-----------|-------|--------------|
| O | Output | 1 | 5-LUT output |
| I0, I1, I2, I3, I4 | Input | 1 | LUT inputs |

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 logic value for the 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;

-- LUT5: 5-input Look-Up Table with general output
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT5_inst : LUT5
generic map (
  INIT => X"00000000") -- Specify LUT Contents
port map (
  O => O,  -- LUT general output
  I0 => I0, -- LUT input
  I1 => I1, -- LUT input
  I2 => I2, -- LUT input
  I3 => I3, -- LUT input
  I4 => I4  -- LUT input
);

-- End of LUT5_inst instantiation

```

Verilog Instantiation Template

```
// LUT5: 5-input Look-Up Table with general output
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT5 #(
    .INIT(32'h00000000) // Specify LUT Contents
) LUT5_inst (
    .O(O), // LUT general output
    .I0(I0), // LUT input
    .I1(I1), // LUT input
    .I2(I2), // LUT input
    .I3(I3), // LUT input
    .I4(I4) // LUT input
);

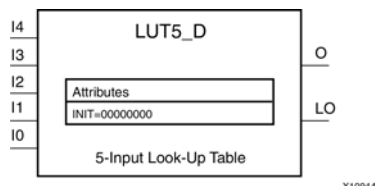
// End of LUT5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT5_D

Primitive: 5-Input Lookup Table with General and Local Outputs



Introduction

This design element is a 5-input, 1-output look-up table (LUT) that can either act as an asynchronous 32-bit ROM (with 5-bit addressing) or implement any 5-input logic function. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. One LUT5 will be packed into a LUT6 within a slice, or two LUT5s can be packed into a single LUT6 with some restrictions. The functionality of the LUT5, LUT5_L and LUT5_D is the same. However, the LUT5_L and LUT5_D allow the additional specification to connect the LUT5 output signal to an internal slice or CLB connection using the LO output. The LUT5_L specifies that the only connections from the LUT5 will be within a slice or CLB, while the LUT5_D allows the specification to connect the output of the LUT to both inter-slice/CLB logic and external logic as well. The LUT5 does not state any specific output connections and should be used in all cases except where internal slice or CLB signal connections must be implicitly specified.

An INIT attribute consisting of a 32-bit hexadecimal value must be specified to indicate the LUTs logical function. The INIT value is calculated by assigning a 1 to the corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 32'h80000000 (X"80000000" for VHDL) will make the output zero unless all of the inputs are one (a 5-input AND gate). A Verilog INIT value of 32'hfffffffe (X"FFFFFFFE" for VHDL) will make the output one unless all zeros are on the inputs (a 5-input OR gate).

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 | |
|---|----|----|----|----|----------|----------|
| I4 | I3 | I2 | I1 | I0 | O | LO |
| 0 | 0 | 0 | 0 | 0 | INIT[0] | INIT[0] |
| 0 | 0 | 0 | 0 | 1 | INIT[1] | INIT[1] |
| 0 | 0 | 0 | 1 | 0 | INIT[2] | INIT[2] |
| 0 | 0 | 0 | 1 | 1 | INIT[3] | INIT[3] |
| 0 | 0 | 1 | 0 | 0 | INIT[4] | INIT[4] |
| 0 | 0 | 1 | 0 | 1 | INIT[5] | INIT[5] |
| 0 | 0 | 1 | 1 | 0 | INIT[6] | INIT[6] |
| 0 | 0 | 1 | 1 | 1 | INIT[7] | INIT[7] |
| 0 | 1 | 0 | 0 | 0 | INIT[8] | INIT[8] |
| 0 | 1 | 0 | 0 | 1 | INIT[9] | INIT[9] |
| 0 | 1 | 0 | 1 | 0 | INIT[10] | INIT[10] |
| 0 | 1 | 0 | 1 | 1 | INIT[11] | INIT[11] |
| 0 | 1 | 1 | 0 | 0 | INIT[12] | INIT[12] |
| 0 | 1 | 1 | 0 | 1 | INIT[13] | INIT[13] |
| 0 | 1 | 1 | 1 | 0 | INIT[14] | INIT[14] |
| 0 | 1 | 1 | 1 | 1 | INIT[15] | INIT[15] |
| 1 | 0 | 0 | 0 | 0 | INIT[16] | INIT[16] |
| 1 | 0 | 0 | 0 | 1 | INIT[17] | INIT[17] |
| 1 | 0 | 0 | 1 | 0 | INIT[18] | INIT[18] |
| 1 | 0 | 0 | 1 | 1 | INIT[19] | INIT[19] |
| 1 | 0 | 1 | 0 | 0 | INIT[20] | INIT[20] |
| 1 | 0 | 1 | 0 | 1 | INIT[21] | INIT[21] |
| 1 | 0 | 1 | 1 | 0 | INIT[22] | INIT[22] |
| 1 | 0 | 1 | 1 | 1 | INIT[23] | INIT[23] |
| 1 | 1 | 0 | 0 | 0 | INIT[24] | INIT[24] |
| 1 | 1 | 0 | 0 | 1 | INIT[25] | INIT[25] |
| 1 | 1 | 0 | 1 | 0 | INIT[26] | INIT[26] |
| 1 | 1 | 0 | 1 | 1 | INIT[27] | INIT[27] |
| 1 | 1 | 1 | 0 | 0 | INIT[28] | INIT[28] |
| 1 | 1 | 1 | 0 | 1 | INIT[29] | INIT[29] |
| 1 | 1 | 1 | 1 | 0 | INIT[30] | INIT[30] |
| 1 | 1 | 1 | 1 | 1 | INIT[31] | INIT[31] |
| INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute | | | | | | |

Port Description

| Name | Direction | Width | Function |
|--------------------|-----------|-------|--|
| O | Output | 1 | 5-LUT output |
| L0 | Output | 1 | 5-LUT output for internal CLB connection |
| I0, I1, I2, I3, I4 | Input | 1 | LUT inputs |

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 logic value for the 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;

-- LUT5_D: 5-input Look-Up Table with general and local outputs
--          Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT5_D_inst : LUT5_D
generic map (
  INIT => X"00000000") -- Specify LUT contents
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
  I4 => I4  -- LUT input
);

-- End of LUT5_D_inst instantiation

```


Verilog Instantiation Template

```
// LUT5_D: 5-input Look-Up Table with general and local outputs
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT5_D #(
    .INIT(32'h00000000) // Specify LUT Contents
) LUT5_D_inst (
    .LO(LO), // LUT local output
    .O(0),   // LUT general output
    .I0(I0), // LUT input
    .I1(I1), // LUT input
    .I2(I2), // LUT input
    .I3(I3), // LUT input
    .I4(I4)  // LUT input
);

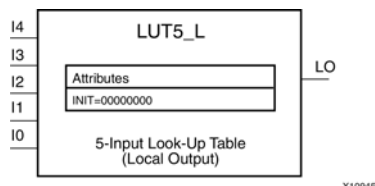
// End of LUT5_D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT5_L

Primitive: 5-Input Lookup Table with Local Output



Introduction

This design element is a 5-input, 1-output look-up table (LUT) that can either act as an asynchronous 32-bit ROM (with 5-bit addressing) or implement any 5-input logic function. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. One LUT5 will be packed into a LUT6 within a slice, or two LUT5s can be packed into a single LUT6 with some restrictions. The functionality of the LUT5, LUT5_L and LUT5_D is the same. However, the LUT5_L and LUT5_D allow the additional specification to connect the LUT5 output signal to an internal slice or CLB connection using the LO output. The LUT5_L specifies that the only connections from the LUT5 is within a slice or CLB, while the LUT5_D allows the specification to connect the output of the LUT to both inter-slice/CLB logic and external logic as well. The LUT5 does not state any specific output connections and should be used in all cases except where internal slice or CLB signal connections must be implicitly specified.

An INIT attribute consisting of a 32-bit hexadecimal value must be specified to indicate the LUTs logical function. The INIT value is calculated by assigning a 1 to the corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 32'h80000000 (X"80000000" for VHDL) makes the output zero unless all of the inputs are one (a 5-input AND gate). A Verilog INIT value of 32'hfffffffe (X"FFFFFFFE" for VHDL) makes the output one unless all zeros are on the inputs (a 5-input OR gate).

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 truth 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 logic 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 |
|---|----|----|----|----|----------|
| I4 | I3 | I2 | I1 | I0 | LO |
| 0 | 0 | 0 | 0 | 0 | INIT[0] |
| 0 | 0 | 0 | 0 | 1 | INIT[1] |
| 0 | 0 | 0 | 1 | 0 | INIT[2] |
| 0 | 0 | 0 | 1 | 1 | INIT[3] |
| 0 | 0 | 1 | 0 | 0 | INIT[4] |
| 0 | 0 | 1 | 0 | 1 | INIT[5] |
| 0 | 0 | 1 | 1 | 0 | INIT[6] |
| 0 | 0 | 1 | 1 | 1 | INIT[7] |
| 0 | 1 | 0 | 0 | 0 | INIT[8] |
| 0 | 1 | 0 | 0 | 1 | INIT[9] |
| 0 | 1 | 0 | 1 | 0 | INIT[10] |
| 0 | 1 | 0 | 1 | 1 | INIT[11] |
| 0 | 1 | 1 | 0 | 0 | INIT[12] |
| 0 | 1 | 1 | 0 | 1 | INIT[13] |
| 0 | 1 | 1 | 1 | 0 | INIT[14] |
| 0 | 1 | 1 | 1 | 1 | INIT[15] |
| 1 | 0 | 0 | 0 | 0 | INIT[16] |
| 1 | 0 | 0 | 0 | 1 | INIT[17] |
| 1 | 0 | 0 | 1 | 0 | INIT[18] |
| 1 | 0 | 0 | 1 | 1 | INIT[19] |
| 1 | 0 | 1 | 0 | 0 | INIT[20] |
| 1 | 0 | 1 | 0 | 1 | INIT[21] |
| 1 | 0 | 1 | 1 | 0 | INIT[22] |
| 1 | 0 | 1 | 1 | 1 | INIT[23] |
| 1 | 1 | 0 | 0 | 0 | INIT[24] |
| 1 | 1 | 0 | 0 | 1 | INIT[25] |
| 1 | 1 | 0 | 1 | 0 | INIT[26] |
| 1 | 1 | 0 | 1 | 1 | INIT[27] |
| 1 | 1 | 1 | 0 | 0 | INIT[28] |
| 1 | 1 | 1 | 0 | 1 | INIT[29] |
| 1 | 1 | 1 | 1 | 0 | INIT[30] |
| 1 | 1 | 1 | 1 | 1 | INIT[31] |
| INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute | | | | | |

Port Description

| Name | Direction | Width | Function |
|--------------------|-----------|-------|--|
| L0 | Output | 1 | 6/5-LUT output for internal CLB connection |
| I0, I1, I2, I3, I4 | Input | 1 | LUT inputs |

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 logic value for the 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;

-- LUT5_L: 5-input Look-Up Table with local output
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT5_L_inst : LUT5_L
generic map (
  INIT => X"00000000") -- Specify LUT Contents
port map (
  L0 => L0, -- LUT local output
  I0 => I0, -- LUT input
  I1 => I1, -- LUT input
  I2 => I2, -- LUT input
  I3 => I3, -- LUT input
  I4 => I4  -- LUT input
);

-- End of LUT5_L_inst instantiation

```

Verilog Instantiation Template

```
// LUT5_L: 5-input Look-Up Table with local output
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT5_L #(
    .INIT(32'h00000000) // Specify LUT Contents
) LUT5_L_inst (
    .LO(LO), // LUT local output
    .I0(I0), // LUT input
    .I1(I1), // LUT input
    .I2(I2), // LUT input
    .I3(I3), // LUT input
    .I4(I4)  // LUT input
);

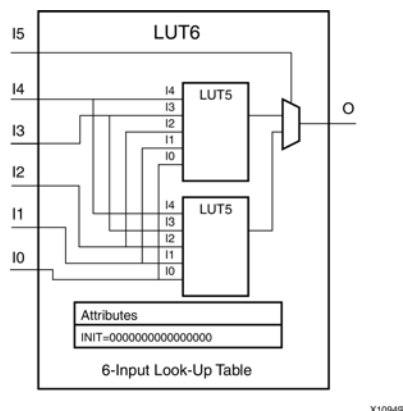
// End of LUT5_L_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT6

Primitive: 6-Input Lookup Table with General Output



Introduction

This design element is a 6-input, 1-output look-up table (LUT) that can either act as an asynchronous 64-bit ROM (with 6-bit addressing) or implement any 6-input logic function. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. A LUT6 is mapped to one of the four look-up tables in the slice. The functionality of the LUT6, LUT6_L and LUT6_D is the same. However, the LUT6_L and LUT6_D allow the additional specification to connect the LUT6 output signal to an internal slice, or CLB connection, using the LO output. The LUT6_L specifies that the only connections from the LUT6 will be within a slice, or CLB, while the LUT6_D allows the specification to connect the output of the LUT to both inter-slice/CLB logic and external logic as well. The LUT6 does not state any specific output connections and should be used in all cases except where internal slice or CLB signal connections must be implicitly specified.

An INIT attribute consisting of a 64-bit Hexadecimal value must be specified to indicate the LUTs logical function. The INIT value is calculated by assigning a 1 to corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 64'h8000000000000000 (X"8000000000000000" for VHDL) makes the output zero unless all of the inputs are one (a 6-input AND gate). A Verilog INIT value of 64'hffffffffffff (X"FFFFFFFFFFFFFFFF" for VHDL) makes the output one unless all zeros are on the inputs (a 6-input OR gate).

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 |
|--------|----|----|----|----|----|---------|
| I5 | I4 | I3 | I2 | I1 | I0 | O |
| 0 | 0 | 0 | 0 | 0 | 0 | INIT[0] |
| 0 | 0 | 0 | 0 | 0 | 1 | INIT[1] |
| 0 | 0 | 0 | 0 | 1 | 0 | INIT[2] |

| Inputs | | | | | | Outputs |
|--------|----|----|----|----|----|----------|
| I5 | I4 | I3 | I2 | I1 | I0 | O |
| 0 | 0 | 0 | 0 | 1 | 1 | INIT[3] |
| 0 | 0 | 0 | 1 | 0 | 0 | INIT[4] |
| 0 | 0 | 0 | 1 | 0 | 1 | INIT[5] |
| 0 | 0 | 0 | 1 | 1 | 0 | INIT[6] |
| 0 | 0 | 0 | 1 | 1 | 1 | INIT[7] |
| 0 | 0 | 1 | 0 | 0 | 0 | INIT[8] |
| 0 | 0 | 1 | 0 | 0 | 1 | INIT[9] |
| 0 | 0 | 1 | 0 | 1 | 0 | INIT[10] |
| 0 | 0 | 1 | 0 | 1 | 1 | INIT[11] |
| 0 | 0 | 1 | 1 | 0 | 0 | INIT[12] |
| 0 | 0 | 1 | 1 | 0 | 1 | INIT[13] |
| 0 | 0 | 1 | 1 | 1 | 0 | INIT[14] |
| 0 | 0 | 1 | 1 | 1 | 1 | INIT[15] |
| 0 | 1 | 0 | 0 | 0 | 0 | INIT[16] |
| 0 | 1 | 0 | 0 | 0 | 1 | INIT[17] |
| 0 | 1 | 0 | 0 | 1 | 0 | INIT[18] |
| 0 | 1 | 0 | 0 | 1 | 1 | INIT[19] |
| 0 | 1 | 0 | 1 | 0 | 0 | INIT[20] |
| 0 | 1 | 0 | 1 | 0 | 1 | INIT[21] |
| 0 | 1 | 0 | 1 | 1 | 0 | INIT[22] |
| 0 | 1 | 0 | 1 | 1 | 1 | INIT[23] |
| 0 | 1 | 1 | 0 | 0 | 0 | INIT[24] |
| 0 | 1 | 1 | 0 | 0 | 1 | INIT[25] |
| 0 | 1 | 1 | 0 | 1 | 0 | INIT[26] |
| 0 | 1 | 1 | 0 | 1 | 1 | INIT[27] |
| 0 | 1 | 1 | 1 | 0 | 0 | INIT[28] |
| 0 | 1 | 1 | 1 | 0 | 1 | INIT[29] |
| 0 | 1 | 1 | 1 | 1 | 0 | INIT[30] |
| 0 | 1 | 1 | 1 | 1 | 1 | INIT[31] |
| 1 | 0 | 0 | 0 | 0 | 0 | INIT[32] |
| 1 | 0 | 0 | 0 | 0 | 1 | INIT[33] |
| 1 | 0 | 0 | 0 | 1 | 0 | INIT[34] |
| 1 | 0 | 0 | 0 | 1 | 1 | INIT[35] |
| 1 | 0 | 0 | 1 | 0 | 0 | INIT[36] |
| 1 | 0 | 0 | 1 | 0 | 1 | INIT[37] |
| 1 | 0 | 0 | 1 | 1 | 0 | INIT[38] |
| 1 | 0 | 0 | 1 | 1 | 1 | INIT[39] |

| Inputs | | | | | | Outputs |
|---|----|----|----|----|----|----------|
| I5 | I4 | I3 | I2 | I1 | I0 | O |
| 1 | 0 | 1 | 0 | 0 | 0 | INIT[40] |
| 1 | 0 | 1 | 0 | 0 | 1 | INIT[41] |
| 1 | 0 | 1 | 0 | 1 | 0 | INIT[42] |
| 1 | 0 | 1 | 0 | 1 | 1 | INIT[43] |
| 1 | 0 | 1 | 1 | 0 | 0 | INIT[44] |
| 1 | 0 | 1 | 1 | 0 | 1 | INIT[45] |
| 1 | 0 | 1 | 1 | 1 | 0 | INIT[46] |
| 1 | 0 | 1 | 1 | 1 | 1 | INIT[47] |
| 1 | 1 | 0 | 0 | 0 | 0 | INIT[48] |
| 1 | 1 | 0 | 0 | 0 | 1 | INIT[49] |
| 1 | 1 | 0 | 0 | 1 | 0 | INIT[50] |
| 1 | 1 | 0 | 0 | 1 | 1 | INIT[51] |
| 1 | 1 | 0 | 1 | 0 | 0 | INIT[52] |
| 1 | 1 | 0 | 1 | 0 | 1 | INIT[53] |
| 1 | 1 | 0 | 1 | 1 | 0 | INIT[54] |
| 1 | 1 | 0 | 1 | 1 | 1 | INIT[55] |
| 1 | 1 | 1 | 0 | 0 | 0 | INIT[56] |
| 1 | 1 | 1 | 0 | 0 | 1 | INIT[57] |
| 1 | 1 | 1 | 0 | 1 | 0 | INIT[58] |
| 1 | 1 | 1 | 0 | 1 | 1 | INIT[59] |
| 1 | 1 | 1 | 1 | 0 | 0 | INIT[60] |
| 1 | 1 | 1 | 1 | 0 | 1 | INIT[61] |
| 1 | 1 | 1 | 1 | 1 | 0 | INIT[62] |
| 1 | 1 | 1 | 1 | 1 | 1 | INIT[63] |
| INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute | | | | | | |

Port Description

| Name | Direction | Width | Function |
|------------------------|-----------|-------|----------------|
| O | Output | 1 | 6/5-LUT output |
| I0, I1, I2, I3, I4, I5 | Input | 1 | LUT inputs |

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 logic value for the 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;

-- LUT6: 6-input Look-Up Table with general output
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT6_inst : LUT6
generic map (
  INIT => X"0000000000000000") -- Specify LUT Contents
port map (
  O => O, -- LUT general output
  I0 => I0, -- LUT input
  I1 => I1, -- LUT input
  I2 => I2, -- LUT input
  I3 => I3, -- LUT input
  I4 => I4, -- LUT input
  I5 => I5 -- LUT input
);

-- End of LUT6_inst instantiation
```

Verilog Instantiation Template

```
// LUT6: 6-input Look-Up Table with general output
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT6 #(
  .INIT(64'h0000000000000000) // Specify LUT Contents
) LUT6_inst (
  .O(O), // LUT general output
  .I0(I0), // LUT input
  .I1(I1), // LUT input
  .I2(I2), // LUT input
  .I3(I3), // LUT input
  .I4(I4), // LUT input
  .I5(I5) // LUT input
);

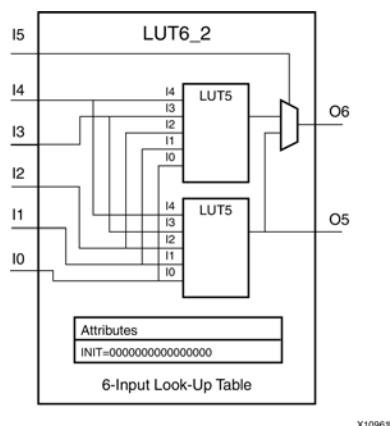
// End of LUT6_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT6_2

Primitive: Six-input, 2-output, Look-Up Table



Introduction

This design element is a 6-input, 2-output look-up table (LUT) that can either act as a dual asynchronous 32-bit ROM (with 5-bit addressing), implement any two 5-input logic functions with shared inputs, or implement a 6-input logic function and a 5-input logic function with shared inputs and shared logic values. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. A LUT6_2 will be mapped to one of the four look-up tables in the slice.

An INIT attribute consisting of a 64-bit hexadecimal value must be specified to indicate the LUTs logical function. The INIT value is calculated by assigning a 1 to corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 64'hfffffffffffffe (X"FFFFFFFFFFFFFFFE" for VHDL) makes the O6 output 1 unless all zeros are on the inputs and the O5 output a 1, or unless I[4:0] are all zeroes (a 5-input and 6-input OR gate). The lower half (bits 31:0) of the INIT values apply to the logic function of the O5 output.

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 | |
|--------|----|----|----|----|----|---------|---------|
| I5 | I4 | I3 | I2 | I1 | I0 | O5 | O6 |
| 0 | 0 | 0 | 0 | 0 | 0 | INIT[0] | INIT[0] |
| 0 | 0 | 0 | 0 | 0 | 1 | INIT[1] | INIT[1] |
| 0 | 0 | 0 | 0 | 1 | 0 | INIT[2] | INIT[2] |
| 0 | 0 | 0 | 0 | 1 | 1 | INIT[3] | INIT[3] |
| 0 | 0 | 0 | 1 | 0 | 0 | INIT[4] | INIT[4] |

| Inputs | | | | | | Outputs | |
|--------|---|---|---|---|---|----------|----------|
| 0 | 0 | 0 | 1 | 0 | 1 | INIT[5] | INIT[5] |
| 0 | 0 | 0 | 1 | 1 | 0 | INIT[6] | INIT[6] |
| 0 | 0 | 0 | 1 | 1 | 1 | INIT[7] | INIT[7] |
| 0 | 0 | 1 | 0 | 0 | 0 | INIT[8] | INIT[8] |
| 0 | 0 | 1 | 0 | 0 | 1 | INIT[9] | INIT[9] |
| 0 | 0 | 1 | 0 | 1 | 0 | INIT[10] | INIT[10] |
| 0 | 0 | 1 | 0 | 1 | 1 | INIT[11] | INIT[11] |
| 0 | 0 | 1 | 1 | 0 | 0 | INIT[12] | INIT[12] |
| 0 | 0 | 1 | 1 | 0 | 1 | INIT[13] | INIT[13] |
| 0 | 0 | 1 | 1 | 1 | 0 | INIT[14] | INIT[14] |
| 0 | 0 | 1 | 1 | 1 | 1 | INIT[15] | INIT[15] |
| 0 | 1 | 0 | 0 | 0 | 0 | INIT[16] | INIT[16] |
| 0 | 1 | 0 | 0 | 0 | 1 | INIT[17] | INIT[17] |
| 0 | 1 | 0 | 0 | 1 | 0 | INIT[18] | INIT[18] |
| 0 | 1 | 0 | 0 | 1 | 1 | INIT[19] | INIT[19] |
| 0 | 1 | 0 | 1 | 0 | 0 | INIT[20] | INIT[20] |
| 0 | 1 | 0 | 1 | 0 | 1 | INIT[21] | INIT[21] |
| 0 | 1 | 0 | 1 | 1 | 0 | INIT[22] | INIT[22] |
| 0 | 1 | 0 | 1 | 1 | 1 | INIT[23] | INIT[23] |
| 0 | 1 | 1 | 0 | 0 | 0 | INIT[24] | INIT[24] |
| 0 | 1 | 1 | 0 | 0 | 1 | INIT[25] | INIT[25] |
| 0 | 1 | 1 | 0 | 1 | 0 | INIT[26] | INIT[26] |
| 0 | 1 | 1 | 0 | 1 | 1 | INIT[27] | INIT[27] |
| 0 | 1 | 1 | 1 | 0 | 0 | INIT[28] | INIT[28] |
| 0 | 1 | 1 | 1 | 0 | 1 | INIT[29] | INIT[29] |
| 0 | 1 | 1 | 1 | 1 | 0 | INIT[30] | INIT[30] |
| 0 | 1 | 1 | 1 | 1 | 1 | INIT[31] | INIT[31] |
| 1 | 0 | 0 | 0 | 0 | 0 | INIT[0] | INIT[32] |
| 1 | 0 | 0 | 0 | 0 | 1 | INIT[1] | INIT[33] |
| 1 | 0 | 0 | 0 | 1 | 0 | INIT[2] | INIT[34] |
| 1 | 0 | 0 | 0 | 1 | 1 | INIT[3] | INIT[35] |
| 1 | 0 | 0 | 1 | 0 | 0 | INIT[4] | INIT[36] |
| 1 | 0 | 0 | 1 | 0 | 1 | INIT[5] | INIT[37] |
| 1 | 0 | 0 | 1 | 1 | 0 | INIT[6] | INIT[38] |
| 1 | 0 | 0 | 1 | 1 | 1 | INIT[7] | INIT[39] |
| 1 | 0 | 1 | 0 | 0 | 0 | INIT[8] | INIT[40] |
| 1 | 0 | 1 | 0 | 0 | 1 | INIT[9] | INIT[41] |
| 1 | 0 | 1 | 0 | 1 | 0 | INIT[10] | INIT[42] |
| 1 | 0 | 1 | 0 | 1 | 1 | INIT[11] | INIT[43] |

| Inputs | | | | | | Outputs | |
|--------|---|---|---|---|---|----------|----------|
| 1 | 0 | 1 | 1 | 0 | 0 | INIT[12] | INIT[44] |
| 1 | 0 | 1 | 1 | 0 | 1 | INIT[13] | INIT[45] |
| 1 | 0 | 1 | 1 | 1 | 0 | INIT[14] | INIT[46] |
| 1 | 0 | 1 | 1 | 1 | 1 | INIT[15] | INIT[47] |
| 1 | 1 | 0 | 0 | 0 | 0 | INIT[16] | INIT[48] |
| 1 | 1 | 0 | 0 | 0 | 1 | INIT[17] | INIT[49] |
| 1 | 1 | 0 | 0 | 1 | 0 | INIT[18] | INIT[50] |
| 1 | 1 | 0 | 0 | 1 | 1 | INIT[19] | INIT[51] |
| 1 | 1 | 0 | 1 | 0 | 0 | INIT[20] | INIT[52] |
| 1 | 1 | 0 | 1 | 0 | 1 | INIT[21] | INIT[53] |
| 1 | 1 | 0 | 1 | 1 | 0 | INIT[22] | INIT[54] |
| 1 | 1 | 0 | 1 | 1 | 1 | INIT[23] | INIT[55] |
| 1 | 1 | 1 | 0 | 0 | 0 | INIT[24] | INIT[56] |
| 1 | 1 | 1 | 0 | 0 | 1 | INIT[25] | INIT[57] |
| 1 | 1 | 1 | 0 | 1 | 0 | INIT[26] | INIT[58] |
| 1 | 1 | 1 | 0 | 1 | 1 | INIT[27] | INIT[59] |
| 1 | 1 | 1 | 1 | 0 | 0 | INIT[28] | INIT[60] |
| 1 | 1 | 1 | 1 | 0 | 1 | INIT[29] | INIT[61] |
| 1 | 1 | 1 | 1 | 1 | 0 | INIT[30] | INIT[62] |
| 1 | 1 | 1 | 1 | 1 | 1 | INIT[31] | INIT[63] |

INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute

Port Descriptions

| Port | Direction | Width | Function |
|------------------------|-----------|-------|----------------|
| O6 | Output | 1 | 6/5-LUT output |
| O5 | Output | 1 | 5-LUT output |
| I0, I1, I2, I3, I4, I5 | Input | 1 | LUT inputs |

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 LUT5/6 output function. |

VHDL Instantiation Template

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

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

-- LUT6_2: 6-input 2 output Look-Up Table
--          Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT6_2_inst : LUT6_2
generic map (
  INIT => X"0000000000000000" -- Specify LUT Contents
port map (
  O6 => O6, -- 6/5-LUT output (1-bit)
  O5 => O5, -- 5-LUT output (1-bit)
  I0 => I0, -- LUT input (1-bit)
  I1 => I1, -- LUT input (1-bit)
  I2 => I2, -- LUT input (1-bit)
  I3 => I3, -- LUT input (1-bit)
  I4 => I4, -- LUT input (1-bit)
  I5 => I5, -- LUT input (1-bit)
);

-- End of LUT6_2_inst instantiation
```

Verilog Instantiation Template

```
// LUT6_2: 6-input, 2 output Look-Up Table
//          Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT6_2 #(
  .INIT(64'h0000000000000000) // Specify LUT Contents
) LUT6_2_inst (
  .O6(O6), // 1-bit LUT6 output
  .O5(O5), // 1-bit lower LUT5 output
  .I0(I0), // 1-bit LUT input
  .I1(I1), // 1-bit LUT input
  .I2(I2), // 1-bit LUT input
  .I3(I3), // 1-bit LUT input
  .I4(I4), // 1-bit LUT input
  .I5(I5) // 1-bit LUT input (fast MUX select only available to O6 output)
);

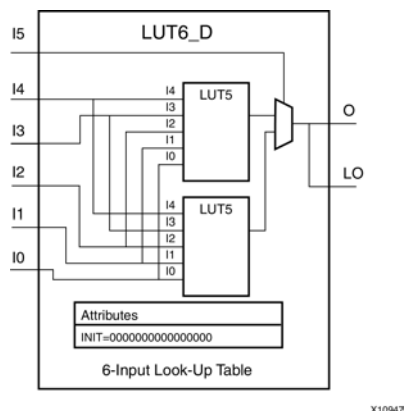
// End of LUT6_2_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT6_D

Primitive: 6-Input Lookup Table with General and Local Outputs



Introduction

This design element is a six-input, one-output look-up table (LUT) that can either act as an asynchronous 64-bit ROM (with 6-bit addressing) or implement any 6-input logic function. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. A LUT6 is mapped to one of the four look-up tables in the slice. The functionality of the LUT6, LUT6_L and LUT6_D is the same. However, the LUT6_L and LUT6_D allow the additional specification to connect the LUT6 output signal to an internal slice, or CLB connection, using the LO output. The LUT6_L specifies that the only connections from the LUT6 will be within a slice, or CLB, while the LUT6_D allows the specification to connect the output of the LUT to both inter-slice/CLB logic and external logic as well. The LUT6 does not state any specific output connections and should be used in all cases except where internal slice or CLB signal connections must be implicitly specified.

An INIT attribute consisting of a 64-bit Hexadecimal value must be specified to indicate the LUTs logical function. The INIT value is calculated by assigning a 1 to corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 64'h8000000000000000 (X"8000000000000000" for VHDL) makes the output zero unless all of the inputs are one (a 6-input AND gate). A Verilog INIT value of 64'hffffffffffff (X"FFFFFFFFFFFFFFFF" for VHDL) makes the output one unless all zeros are on the inputs (a 6-input OR gate).

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 is self-documenting than the above method. However, this method does require the code to first specify the appropriate parameters.

Logic Table

| Inputs | | | | | | Outputs | |
|--------|----|----|----|----|----|---------|---------|
| I5 | I4 | I3 | I2 | I1 | I0 | O | LO |
| 0 | 0 | 0 | 0 | 0 | 0 | INIT[0] | INIT[0] |
| 0 | 0 | 0 | 0 | 0 | 1 | INIT[1] | INIT[1] |
| 0 | 0 | 0 | 0 | 1 | 0 | INIT[2] | INIT[2] |

| Inputs | | | | | | Outputs | |
|--------|----|----|----|----|----|----------|----------|
| I5 | I4 | I3 | I2 | I1 | I0 | O | LO |
| 0 | 0 | 0 | 0 | 1 | 1 | INIT[3] | INIT[3] |
| 0 | 0 | 0 | 1 | 0 | 0 | INIT[4] | INIT[4] |
| 0 | 0 | 0 | 1 | 0 | 1 | INIT[5] | INIT[5] |
| 0 | 0 | 0 | 1 | 1 | 0 | INIT[6] | INIT[6] |
| 0 | 0 | 0 | 1 | 1 | 1 | INIT[7] | INIT[7] |
| 0 | 0 | 1 | 0 | 0 | 0 | INIT[8] | INIT[8] |
| 0 | 0 | 1 | 0 | 0 | 1 | INIT[9] | INIT[9] |
| 0 | 0 | 1 | 0 | 1 | 0 | INIT[10] | INIT[10] |
| 0 | 0 | 1 | 0 | 1 | 1 | INIT[11] | INIT[11] |
| 0 | 0 | 1 | 1 | 0 | 0 | INIT[12] | INIT[12] |
| 0 | 0 | 1 | 1 | 0 | 1 | INIT[13] | INIT[13] |
| 0 | 0 | 1 | 1 | 1 | 0 | INIT[14] | INIT[14] |
| 0 | 0 | 1 | 1 | 1 | 1 | INIT[15] | INIT[15] |
| 0 | 1 | 0 | 0 | 0 | 0 | INIT[16] | INIT[16] |
| 0 | 1 | 0 | 0 | 0 | 1 | INIT[17] | INIT[17] |
| 0 | 1 | 0 | 0 | 1 | 0 | INIT[18] | INIT[18] |
| 0 | 1 | 0 | 0 | 1 | 1 | INIT[19] | INIT[19] |
| 0 | 1 | 0 | 1 | 0 | 0 | INIT[20] | INIT[20] |
| 0 | 1 | 0 | 1 | 0 | 1 | INIT[21] | INIT[21] |
| 0 | 1 | 0 | 1 | 1 | 0 | INIT[22] | INIT[22] |
| 0 | 1 | 0 | 1 | 1 | 1 | INIT[23] | INIT[23] |
| 0 | 1 | 1 | 0 | 0 | 0 | INIT[24] | INIT[24] |
| 0 | 1 | 1 | 0 | 0 | 1 | INIT[25] | INIT[25] |
| 0 | 1 | 1 | 0 | 1 | 0 | INIT[26] | INIT[26] |
| 0 | 1 | 1 | 0 | 1 | 1 | INIT[27] | INIT[27] |
| 0 | 1 | 1 | 1 | 0 | 0 | INIT[28] | INIT[28] |
| 0 | 1 | 1 | 1 | 0 | 1 | INIT[29] | INIT[29] |
| 0 | 1 | 1 | 1 | 1 | 0 | INIT[30] | INIT[30] |
| 0 | 1 | 1 | 1 | 1 | 1 | INIT[31] | INIT[31] |
| 1 | 0 | 0 | 0 | 0 | 0 | INIT[32] | INIT[32] |
| 1 | 0 | 0 | 0 | 0 | 1 | INIT[33] | INIT[33] |
| 1 | 0 | 0 | 0 | 1 | 0 | INIT[34] | INIT[34] |
| 1 | 0 | 0 | 0 | 1 | 1 | INIT[35] | INIT[35] |
| 1 | 0 | 0 | 1 | 0 | 0 | INIT[36] | INIT[36] |
| 1 | 0 | 0 | 1 | 0 | 1 | INIT[37] | INIT[37] |
| 1 | 0 | 0 | 1 | 1 | 0 | INIT[38] | INIT[38] |
| 1 | 0 | 0 | 1 | 1 | 1 | INIT[39] | INIT[39] |

| Inputs | | | | | | Outputs | |
|---|----|----|----|----|----|----------|----------|
| I5 | I4 | I3 | I2 | I1 | I0 | O | LO |
| 1 | 0 | 1 | 0 | 0 | 0 | INIT[40] | INIT[40] |
| 1 | 0 | 1 | 0 | 0 | 1 | INIT[41] | INIT[41] |
| 1 | 0 | 1 | 0 | 1 | 0 | INIT[42] | INIT[42] |
| 1 | 0 | 1 | 0 | 1 | 1 | INIT[43] | INIT[43] |
| 1 | 0 | 1 | 1 | 0 | 0 | INIT[44] | INIT[44] |
| 1 | 0 | 1 | 1 | 0 | 1 | INIT[45] | INIT[45] |
| 1 | 0 | 1 | 1 | 1 | 0 | INIT[46] | INIT[46] |
| 1 | 0 | 1 | 1 | 1 | 1 | INIT[47] | INIT[47] |
| 1 | 1 | 0 | 0 | 0 | 0 | INIT[48] | INIT[48] |
| 1 | 1 | 0 | 0 | 0 | 1 | INIT[49] | INIT[49] |
| 1 | 1 | 0 | 0 | 1 | 0 | INIT[50] | INIT[50] |
| 1 | 1 | 0 | 0 | 1 | 1 | INIT[51] | INIT[51] |
| 1 | 1 | 0 | 1 | 0 | 0 | INIT[52] | INIT[52] |
| 1 | 1 | 0 | 1 | 0 | 1 | INIT[53] | INIT[53] |
| 1 | 1 | 0 | 1 | 1 | 0 | INIT[54] | INIT[54] |
| 1 | 1 | 0 | 1 | 1 | 1 | INIT[55] | INIT[55] |
| 1 | 1 | 1 | 0 | 0 | 0 | INIT[56] | INIT[56] |
| 1 | 1 | 1 | 0 | 0 | 1 | INIT[57] | INIT[57] |
| 1 | 1 | 1 | 0 | 1 | 0 | INIT[58] | INIT[58] |
| 1 | 1 | 1 | 0 | 1 | 1 | INIT[59] | INIT[59] |
| 1 | 1 | 1 | 1 | 0 | 0 | INIT[60] | INIT[60] |
| 1 | 1 | 1 | 1 | 0 | 1 | INIT[61] | INIT[61] |
| 1 | 1 | 1 | 1 | 1 | 0 | INIT[62] | INIT[62] |
| 1 | 1 | 1 | 1 | 1 | 1 | INIT[63] | INIT[63] |
| INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute | | | | | | | |

Port Description

| Name | Direction | Width | Function |
|------------------------|-----------|-------|----------------|
| O6 | Output | 1 | 6/5-LUT output |
| O5 | Output | 1 | 5-LUT output |
| I0, I1, I2, I3, I4, I5 | Input | 1 | LUT inputs |

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 logic value for the 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;

-- LUT6_D: 6-input Look-Up Table with general and local outputs
--          Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT6_D_inst : LUT6_D
generic map (
  INIT => X"0000000000000000") -- Specify LUT contents
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
  I4 => I4, -- LUT input
  I5 => I5  -- LUT input
);

-- End of LUT6_D_inst instantiation
```

Verilog Instantiation Template

```
// LUT6_D: 6-input Look-Up Table with general and local outputs
//          Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT6_D #(
  .INIT(64'h0000000000000000) // Specify LUT Contents
) LUT6_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
  .I4(I4), // LUT input
  .I5(I5)  // LUT input
);

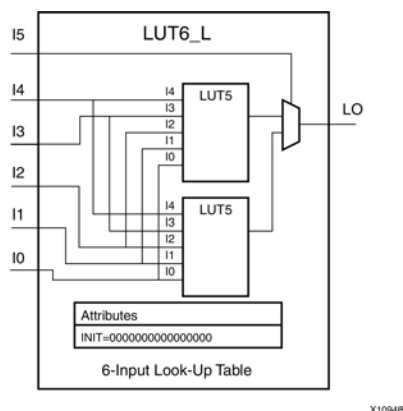
// End of LUT6_D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

LUT6_L

Primitive: 6-Input Lookup Table with Local Output



Introduction

This design element is a 6-input, 1-output look-up table (LUT) that can either act as an asynchronous 64-bit ROM (with 6-bit addressing) or implement any 6-input logic function. LUTs are the basic logic building blocks and are used to implement most logic functions of the design. A LUT6 is mapped to one of the four look-up tables in the slice. The functionality of the LUT6, LUT6_L and LUT6_D is the same. However, the LUT6_L and LUT6_D allow the additional specification to connect the LUT6 output signal to an internal slice, or CLB connection, using the LO output. The LUT6_L specifies that the only connections from the LUT6 are within a slice, or CLB, while the LUT6_D allows the specification to connect the output of the LUT to both inter-slice/CLB logic and external logic as well. The LUT6 does not state any specific output connections and should be used in all cases except where internal slice or CLB signal connections must be implicitly specified.

An INIT attribute consisting of a 64-bit hexadecimal value must be specified to indicate the LUT's logical function. The INIT value is calculated by assigning a 1 to the corresponding INIT bit value when the associated inputs are applied. For instance, a Verilog INIT value of 64'h8000000000000000 (X"8000000000000000" for VHDL) will make the output zero unless all of the inputs are one (a 6-input AND gate). A Verilog INIT value of 64'hffffffffffff (X"FFFFFFFFFFFFFFF" for VHDL) will make the output one unless all zeros are on the inputs (a 6-input OR gate).

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 truth 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 |
|--------|----|----|----|----|----|---------|
| I5 | I4 | I3 | I2 | I1 | I0 | LO |
| 0 | 0 | 0 | 0 | 0 | 0 | INIT[0] |
| 0 | 0 | 0 | 0 | 0 | 1 | INIT[1] |

| Inputs | | | | | | Outputs |
|--------|----|----|----|----|----|----------|
| I5 | I4 | I3 | I2 | I1 | I0 | LO |
| 0 | 0 | 0 | 0 | 1 | 0 | INIT[2] |
| 0 | 0 | 0 | 0 | 1 | 1 | INIT[3] |
| 0 | 0 | 0 | 1 | 0 | 0 | INIT[4] |
| 0 | 0 | 0 | 1 | 0 | 1 | INIT[5] |
| 0 | 0 | 0 | 1 | 1 | 0 | INIT[6] |
| 0 | 0 | 0 | 1 | 1 | 1 | INIT[7] |
| 0 | 0 | 1 | 0 | 0 | 0 | INIT[8] |
| 0 | 0 | 1 | 0 | 0 | 1 | INIT[9] |
| 0 | 0 | 1 | 0 | 1 | 0 | INIT[10] |
| 0 | 0 | 1 | 0 | 1 | 1 | INIT[11] |
| 0 | 0 | 1 | 1 | 0 | 0 | INIT[12] |
| 0 | 0 | 1 | 1 | 0 | 1 | INIT[13] |
| 0 | 0 | 1 | 1 | 1 | 0 | INIT[14] |
| 0 | 0 | 1 | 1 | 1 | 1 | INIT[15] |
| 0 | 1 | 0 | 0 | 0 | 0 | INIT[16] |
| 0 | 1 | 0 | 0 | 0 | 1 | INIT[17] |
| 0 | 1 | 0 | 0 | 1 | 0 | INIT[18] |
| 0 | 1 | 0 | 0 | 1 | 1 | INIT[19] |
| 0 | 1 | 0 | 1 | 0 | 0 | INIT[20] |
| 0 | 1 | 0 | 1 | 0 | 1 | INIT[21] |
| 0 | 1 | 0 | 1 | 1 | 0 | INIT[22] |
| 0 | 1 | 0 | 1 | 1 | 1 | INIT[23] |
| 0 | 1 | 1 | 0 | 0 | 0 | INIT[24] |
| 0 | 1 | 1 | 0 | 0 | 1 | INIT[25] |
| 0 | 1 | 1 | 0 | 1 | 0 | INIT[26] |
| 0 | 1 | 1 | 0 | 1 | 1 | INIT[27] |
| 0 | 1 | 1 | 1 | 0 | 0 | INIT[28] |
| 0 | 1 | 1 | 1 | 0 | 1 | INIT[29] |
| 0 | 1 | 1 | 1 | 1 | 0 | INIT[30] |
| 0 | 1 | 1 | 1 | 1 | 1 | INIT[31] |
| 1 | 0 | 0 | 0 | 0 | 0 | INIT[32] |
| 1 | 0 | 0 | 0 | 0 | 1 | INIT[33] |
| 1 | 0 | 0 | 0 | 1 | 0 | INIT[34] |
| 1 | 0 | 0 | 0 | 1 | 1 | INIT[35] |
| 1 | 0 | 0 | 1 | 0 | 0 | INIT[36] |
| 1 | 0 | 0 | 1 | 0 | 1 | INIT[37] |
| 1 | 0 | 0 | 1 | 1 | 0 | INIT[38] |

| Inputs | | | | | | Outputs |
|--------|----|----|----|----|----|----------|
| I5 | I4 | I3 | I2 | I1 | I0 | LO |
| 1 | 0 | 0 | 1 | 1 | 1 | INIT[39] |
| 1 | 0 | 1 | 0 | 0 | 0 | INIT[40] |
| 1 | 0 | 1 | 0 | 0 | 1 | INIT[41] |
| 1 | 0 | 1 | 0 | 1 | 0 | INIT[42] |
| 1 | 0 | 1 | 0 | 1 | 1 | INIT[43] |
| 1 | 0 | 1 | 1 | 0 | 0 | INIT[44] |
| 1 | 0 | 1 | 1 | 0 | 1 | INIT[45] |
| 1 | 0 | 1 | 1 | 1 | 0 | INIT[46] |
| 1 | 0 | 1 | 1 | 1 | 1 | INIT[47] |
| 1 | 1 | 0 | 0 | 0 | 0 | INIT[48] |
| 1 | 1 | 0 | 0 | 0 | 1 | INIT[49] |
| 1 | 1 | 0 | 0 | 1 | 0 | INIT[50] |
| 1 | 1 | 0 | 0 | 1 | 1 | INIT[51] |
| 1 | 1 | 0 | 1 | 0 | 0 | INIT[52] |
| 1 | 1 | 0 | 1 | 0 | 1 | INIT[53] |
| 1 | 1 | 0 | 1 | 1 | 0 | INIT[54] |
| 1 | 1 | 0 | 1 | 1 | 1 | INIT[55] |
| 1 | 1 | 1 | 0 | 0 | 0 | INIT[56] |
| 1 | 1 | 1 | 0 | 0 | 1 | INIT[57] |
| 1 | 1 | 1 | 0 | 1 | 0 | INIT[58] |
| 1 | 1 | 1 | 0 | 1 | 1 | INIT[59] |
| 1 | 1 | 1 | 1 | 0 | 0 | INIT[60] |
| 1 | 1 | 1 | 1 | 0 | 1 | INIT[61] |
| 1 | 1 | 1 | 1 | 1 | 0 | INIT[62] |
| 1 | 1 | 1 | 1 | 1 | 1 | INIT[63] |

INIT = Binary equivalent of the hexadecimal number assigned to the INIT attribute

Port Description

| Name | Direction | Width | Function |
|------------------------|-----------|-------|---|
| LO | Output | 1 | 6/5-LUT output or internal CLB connection |
| I0, I1, I2, I3, I4, I5 | Input | 1 | LUT inputs |

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 logic value for the 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;

-- LUT6_L: 6-input Look-Up Table with local output
--          Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

LUT6_L_inst : LUT6_L
generic map (
  INIT => X"0000000000000000") -- Specify LUT Contents
port map (
  LO => LO, -- LUT local output
  I0 => I0, -- LUT input
  I1 => I1, -- LUT input
  I2 => I2, -- LUT input
  I3 => I3, -- LUT input
  I4 => I4, -- LUT input
  I5 => I5  -- LUT input
);

-- End of LUT6_L_inst instantiation
```

Verilog Instantiation Template

```
// LUT6_L: 6-input Look-Up Table with local output
//          Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

LUT6_L #(
  .INIT(64'h0000000000000000) // Specify LUT Contents
) LUT6_L_inst (
  .LO(LO), // LUT local output
  .I0(I0), // LUT input
  .I1(I1), // LUT input
  .I2(I2), // LUT input
  .I3(I3), // LUT input
  .I4(I4), // LUT input
  .I5(I5)  // LUT input
);

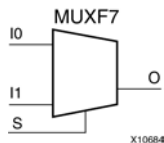
// End of LUT6_L_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

MUXF7 MUXF7_inst (
    .O(O),      // Output of MUX to general routing
    .I0(I0),    // Input (tie to MUXF6 LO out or LUT6 O6 pin)
    .I1(I1),    // Input (tie to MUXF6 LO out or LUT6 O6 pin)
    .S(S)       // Input select to MUX
);

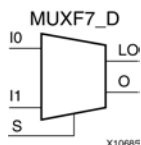
// End of MUXF7_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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
);

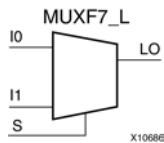
// End of MUXF7_D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--          Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//          Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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
);

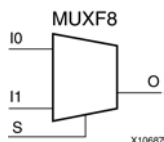
// End of MUXF7_L_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

MUXF8_inst : MUXF8
port map (
    O => O,      -- Output of MUX to general routing
    I0 => I0,    -- Input (tie to MUXF7 LO out)
    I1 => I1,    -- Input (tie to MUXF7 LO out)
    S => S      -- Input select to MUX
);

-- End of MUXF8_inst instantiation
```

Verilog Instantiation Template

```
// MUXF8: CLB MUX to tie two MUXF7's together with general output
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

MUXF8 MUXF8_inst (
    .O(O),      // Output of MUX to general routing
    .I0(I0),    // Input (tie to MUXF7 LO out)
    .I1(I1),    // Input (tie to MUXF7 LO out)
    .S(S)      // Input select to MUX
);

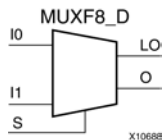
// End of MUXF8_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

MUXF8_D_inst : MUXF8_D
port map (
    LO => LO,  -- Output 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
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

MUXF8_D MUXF8_D_inst (
    .LO(LO), // Output 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
);

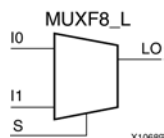
// End of MUXF8_D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

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
);

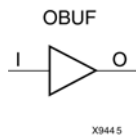
// End of MUXF8_L_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//      All devices
// Xilinx HDL Libraries Guide, version 12.3

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
);

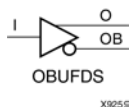
// End of OBUF_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//       Virtex-5, Spartan-3/3E/3A
// Xilinx HDL Libraries Guide, version 12.3

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
);

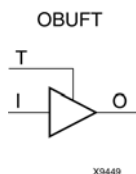
// End of OBUFDS_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//      All devices
// Xilinx HDL Libraries Guide, version 12.3

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
);

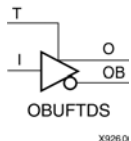
// End of OBUFT_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//           Virtex-5, Spartan-3/3E/3A
// Xilinx HDL Libraries Guide, version 12.3

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
);

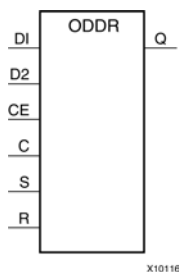
// End of OBUFTDS_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

ODDR

Primitive: Dedicated Dual Data Rate (DDR) Output Register



Introduction

This design element is a dedicated output register for use in transmitting dual data rate (DDR) signals from FPGA devices. The ODDR primitive's interface with the FPGA fabric are not limited to opposite edges. The ODDR is available with modes that allow data to be presented from the FPGA fabric at the same clock edge. This feature allows designers to avoid additional timing complexities and CLB usage. In addition, the ODDR works in conjunction with SelectIO™ features.

ODDR Modes

This element has two modes of operation. These modes are set by the DDR_CLK_EDGE attribute.

- **OPPOSITE_EDGE mode** - The data transmit interface uses the classic DDR methodology. Given a data and clock at pin D1-2 and C respectively, D1 is sampled at every positive edge of clock C, and D2 is sampled at every negative edge of clock C. Q changes every clock edge.
- **SAME_EDGE mode** - Data is still transmitted at the output of the ODDR by opposite edges of clock C. However, the two inputs to the ODDR are clocked with a positive clock edge of clock signal C and an extra register is clocked with a negative clock edge of clock signal C. Using this feature, DDR data can now be presented into the ODDR at the same clock edge.

Port Descriptions

| Port | Type | Width | Function |
|---------|--------|----------|---|
| Q | Output | 1 | Data Output (DDR) - The ODDR output that connects to the IOB pad. |
| C | Input | 1 | Clock Input - The C pin represents the clock input pin. |
| CE | Input | 1 | Clock Enable Input - When asserted High, this port enables the clock input on port C. |
| D1 : D2 | Input | 1 (each) | Data Input - This pin is where the DDR data is presented into the ODDR module. |
| R | Input | 1 | Reset - Depends on how SRTYPE is set. |
| S | Input | 1 | Set - Active High asynchronous set pin. This pin can also be Synchronous depending on the SRTYPE attribute. |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|--------------|---------|---------------------------------|-----------------|---|
| DDR_CLK_EDGE | String | "OPPOSITE_EDGE", "SAME_EDGE" | "OPPOSITE_EDGE" | DDR clock mode recovery mode selection. |
| INIT | Integer | 0, 1 | 1 | Q initialization value. |
| SRTYPE | String | "SYNC", "ASYN" | "SYNC" | Set/Reset type selection. |

VHDL Instantiation Template

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

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

-- ODDR: Output Double Data Rate Output Register with Set, Reset
--       and Clock Enable.
--       Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

ODDR_inst : ODDR
generic map(
  DDR_CLK_EDGE => "OPPOSITE_EDGE", -- "OPPOSITE_EDGE" or "SAME_EDGE"
  INIT => '0', -- Initial value for Q port ('1' or '0')
  SRTYPE => "SYNC") -- Reset Type ("ASYN" or "SYNC")
port map (
  Q => Q, -- 1-bit DDR output
  C => C, -- 1-bit clock input
  CE => CE, -- 1-bit clock enable input
  D1 => D1, -- 1-bit data input (positive edge)
  D2 => D2, -- 1-bit data input (negative edge)
  R => R, -- 1-bit reset input
  S => S -- 1-bit set input
);

-- End of ODDR_inst instantiation
```

Verilog Instantiation Template

```
// ODDR: Output Double Data Rate Output Register with Set, Reset
//       and Clock Enable.
//       Virtex-5/6
// Xilinx HDL Libraries Guide, version 12.3

ODDR #(
  .DDR_CLK_EDGE("OPPOSITE_EDGE"), // "OPPOSITE_EDGE" or "SAME_EDGE"
  .INIT(1'b0), // Initial value of Q: 1'b0 or 1'b1
  .SRTYPE("SYNC") // Set/Reset type: "SYNC" or "ASYN"
) ODDR_inst (
  .Q(Q), // 1-bit DDR output
  .C(C), // 1-bit clock input
  .CE(CE), // 1-bit clock enable input
  .D1(D1), // 1-bit data input (positive edge)
  .D2(D2), // 1-bit data input (negative edge)
  .R(R), // 1-bit reset
  .S(S) // 1-bit set
);

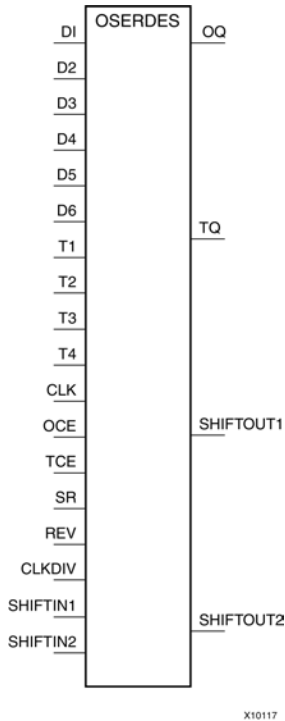
// End of ODDR_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

OSERDES

Primitive: Dedicated IOB Output Serializer



Introduction

Use the OSERDES primitive to easily implement a source synchronous interface. This device helps you by saving logic resources that would otherwise be implemented in the FPGA fabric. It also avoids additional timing complexities that you might encounter when you are designing circuitry in the FPGA fabric. This element contains multiple clock inputs to accommodate various applications, and will work in conjunction with SelectIO™ features.

Port Descriptions

| Port | Type | Width | Function |
|-------------|--------|----------|--|
| OQ | Output | 1 | Data Path Output - This port is the data output of the OSERDES module. This port connects the output of the data parallel-to-serial converter to the data input of the IOB pad. In addition, this output port can also be configured to bypass all the submodules within the OSERDES module. |
| SHIFTOUT1-2 | Output | 1 (each) | Carry Out for data input expansion. Connect to SHIFTIN1/2 of master. |
| TQ | Output | 1 | 3-State Path Output - This port is the 3-state output of the OSERDES module. This port connects the output of the 3-state parallel-to-serial converter to the control input of the IOB pad. |
| CLK | Input | 1 | High Speed Clock Input - This clock input is used to drive the parallel-to-serial converters. The possible source for the CLK port is from one of the following clock resources: <ul style="list-style-type: none"> Ten global clock lines in a clock region Four regional clock lines |

| Port | Type | Width | Function |
|------------|-------|----------|---|
| | | | <ul style="list-style-type: none"> Four clock capable I/Os (within adjacent clock region) Fabric (through bypass) |
| CLKDIV | Input | 1 | Divided High Speed Clock Input - This clock input is used to drive the parallel-to-serial converter. This clock must be a divided down version of the clock connected to the CLK port. One of the following clock resources can be used as a source for CLKDIV: <ul style="list-style-type: none"> Ten global clock lines in a clock region Four regional clock lines |
| D1-D6 | Input | 1 | Parallel Data Inputs - Ports D1 to D6 are the location in which all incoming parallel data enters the OSERDES module. This port is connected to the FPGA fabric, and can be configured from 2 to 6 bits. In the extended width mode, this port can be expanded up to 10 bits. |
| OCE | Input | 1 | Parallel to serial converter (data) clock enable - This port is used to enable the output of the data parallel-to-serial converter when asserted High. |
| SR | Input | 1 | Set/Reset Input - The set/reset (SR) pin forces the storage element into the state specified by the SRVAL attribute. SRVAL = "1" forces a logic 1. SRVAL = "0" forces a logic "0." The reset condition predominates over the set condition. |
| SHIFTIN1-2 | Input | 1 (each) | Carry Input for Data Input Expansion. Connect to SHIFTOUT1/2 of slave. |
| T1 - T4 | Input | 1 (each) | Parallel 3-State Inputs - Ports T1 to T4 are the location in which all parallel 3-state signals enters the OSERDES module. This port is connected to the FPGA fabric, and can be configured from 1 to 4 bits. This feature is not supported in the extended width mode. |
| TCE | Input | 1 | Parallel to serial converter (3-state) clock enable - This port is used to enable the output of the 3-state signal parallel-to-serial converter when asserted High. |

Design Entry Method

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

The data parallel-to-serial converter in the OSERDES module takes in 2 to 6 bits of parallel data and converts them into serial data. Data input widths larger than 6 (7, 8, and 10) are achievable by cascading two OSERDES modules for data width expansion. In order to do this, one OSERDES must be set into a MASTER mode, while another is set into SLAVE mode. You must connect the SHIFTOUT of "slave" and SHIFTIN of "master" ports together. The "slave" only uses D3 to D6 ports as its input. The parallel-to-serial converter is available for both SDR and DDR modes.

This module is designed such that the data input at D1 port is the first output bit. This module is controlled by CLK and CLKDIV clocks. The following table describes the relationship between CLK and CLKDIV for both SDR and DDR mode.

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

Output of this block is connected to the data input of an IOB pad of the FPGA. This IOB pad can be configured to a desired standard using SelectIO.

Parallel-to-Serial Converter (3-state)

The 3-state parallel-to-serial converter in the OSERDES module takes in up to 4 bits of parallel 3-state signals and converts them into serial 3-state signal. Unlike the data parallel-to-serial converter, the 3-state parallel-to-serial converter is not extendable to more than 4-bit, 3-state signals. This module is primarily controlled by CLK and CLKDIV clocks. In order to use this module, the following attributes must be declared: DATA_RATE_TQ and TRISTATE_WIDTH. In certain cases, you can also need to declare DATA_RATE_OQ and DATA_WIDTH. The following table lists the attributes needed for the desired functionality.

| Mode of Operation | DATA_RATE_TQ | TRISTATE_WIDTH |
|-------------------|--------------|----------------|
| 4-bit DDR* | DDR | 4 |
| 1-bit SDR | SDR | 1 |
| Buffer | BUF | 1 |

Output of this block is connected to the 3-state input of an IOB pad of the FPGA. This IOB pad can be configured to a desired standard using SelectIO.

Width Expansion

It is possible to use this element to transmit parallel data widths larger than six. However, the 3-state output is not expandable. In order to use this feature, *two* of these elements need to be instantiated, and the two must be an adjacent master and slave pair. The attribute MODE must be set to either "MASTER" or "SLAVE" in order to differentiate the modes of the OSERDES pair. In addition, you must connect the SHIFTIN ports of the MASTER to the SHIFTOUT ports of the SLAVE. This feature supports data widths of 7, 8, and 10 for SDR and DDR mode. The table below lists the data width availability for SDR and DDR mode.

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|--------------|--------|---------------------|---------|--|
| DATA_RATE_OQ | String | "SDR", "DDR" | "DDR" | Defines whether the data changes at every clock edge or every positive clock edge with respect to CLK. |
| DATA_RATE_TQ | String | "BUF", "SDR", "DDR" | "DDR" | Defines whether the 3-state changes at every clock edge, every positive clock edge, or buffer configuration with respect to CLK. |

| Attribute | Type | Allowed Values | Default | Description |
|----------------|---------|----------------------------|----------|---|
| DATA_WIDTH | Integer | 2, 3, 4, 5, 6, 7, 8, or 10 | 4 | If DATA_RATE_OQ = DDR, value is limited to 4, 6, 8, or 10. If DATA_RATE_OQ = SDR, value is limited to 2, 3, 4, 5, 6, 7, or 8. |
| INIT_OQ | Binary | 0, 1 | 0 | Defines the initial value of OQ output |
| INIT_TQ | Binary | 0, 1 | 0 | Defines the initial value of TQ output |
| SERDES_MODE | String | "MASTER", "SLAVE" | "MASTER" | Defines whether the OSERDES module is a master or slave when width expansion is used. |
| SRVAL_OQ | Binary | 0, 1 | 0 | Defines the value of OQ output when reset is invoked. |
| SRVAL_TQ | Binary | 0, 1 | 0 | Defines the value of TQ output when reset is invoked. |
| TRISTATE_WIDTH | Integer | 1, 2, 4 | 4 | If DATA_RATE_TQ = DDR, value is limited to 2 or 4. The value can only be set to 1 when DATA_RATE_TQ = SDR or BUF. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- OSERDES: Output SERDES
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

OSERDES_inst : OSERDES
generic map (
    DATA_RATE_OQ => "DDR", -- Specify data rate to "DDR" or "SDR"
    DATA_RATE_TQ => "DDR", -- Specify data rate to "DDR", "SDR", or "BUF"
    DATA_WIDTH => 4, -- Specify data width - For DDR: 4,6,8, or 10
                        -- For SDR or BUF: 2,3,4,5,6,7, or 8
    INIT_OQ => '0', -- INIT for Q1 register - '1' or '0'
    INIT_TQ => '0', -- INIT for Q2 register - '1' or '0'
    SERDES_MODE => "MASTER", --Set SERDES mode to "MASTER" or "SLAVE"
    SRVAL_OQ => '0', -- Define Q1 output value upon SR assertion - '1' or '0'
    SRVAL_TQ => '0', -- Define Q1 output value upon SR assertion - '1' or '0'
    TRISTATE_WIDTH => 4) -- Specify parallel to serial converter width
                        -- When DATA_RATE_TQ = DDR: 2 or 4
                        -- When DATA_RATE_TQ = SDR or BUF: 1 "
port map (
    OQ => OQ, -- 1-bit output
    SHIFTOUT1 => SHIFTOUT1, -- 1-bit data expansion output
    SHIFTOUT2 => SHIFTOUT2, -- 1-bit data expansion output
    TQ => TQ, -- 1-bit 3-state control output
    CLK => CLK, -- 1-bit clock input
    CLKDIV => CLKDIV, -- 1-bit divided clock input
    D1 => D1, -- 1-bit parallel data input
    D2 => D2, -- 1-bit parallel data input
    D3 => D3, -- 1-bit parallel data input
    D4 => D4, -- 1-bit parallel data input
    D5 => D5, -- 1-bit parallel data input
    D6 => D6, -- 1-bit parallel data input
    OCE => OCE, -- 1-bit clcok enable input
    REV => '0', -- Must be tied to logic zero
    SHIFTIN1 => SHIFTIN1, -- 1-bit data expansion input
    SHIFTIN2 => SHIFTIN2, -- 1-bit data expansion input
    SR => SR, -- 1-bit set/reset input
    T1 => T1, -- 1-bit parallel 3-state input

```



```

T2 => T2,    -- 1-bit parallel 3-state input
T3 => T3,    -- 1-bit parallel 3-state input
T4 => T4,    -- 1-bit parallel 3-state input
TCE => TCE   -- 1-bit 3-state signal clock enable input
);

-- End of OSERDES_inst instantiation

```

Verilog Instantiation Template

```

// OSERDES: Source Synchronous Output Serializer
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

OSERDES #(
    .DATA_RATE_OQ("DDR"), // Specify data rate to "DDR" or "SDR"
    .DATA_RATE_TQ("DDR"), // Specify data rate to "DDR", "SDR", or "BUF"
    .DATA_WIDTH(4), // Specify data width - For DDR: 4,6,8, or 10
                        //      For SDR or BUF: 2,3,4,5,6,7, or 8
    .INIT_OQ(1'b0), // INIT for OQ register - 1'b1 or 1'b0
    .INIT_TQ(1'b0), // INIT for TQ register - 1'b1 or 1'b0
    .SERDES_MODE("MASTER"), // Set SERDES mode to "MASTER" or "SLAVE"
    .SRVAL_OQ(1'b0), // Define OQ output value upon SR assertion - 1'b1 or 1'b0
    .SRVAL_TQ(1'b0), // Define TQ output value upon SR assertion - 1'b1 or 1'b0
    .TRISTATE_WIDTH(4) // Specify parallel to serial converter width
                        //      When DATA_RATE_TQ = DDR: 2 or 4
                        //      When DATA_RATE_TQ = SDR or BUF: 1
) OSERDES_inst (
    .OQ(OQ), // 1-bit data path output
    .SHIFTOUT1(SHIFTOUT1), // 1-bit data expansion output
    .SHIFTOUT2(SHIFTOUT2), // 1-bit data expansion output
    .TQ(TQ), // 1-bit 3-state control output
    .CLK(CLK), // 1-bit clock input
    .CLKDIV(CLKDIV), // 1-bit divided clock input
    .D1(D1), // 1-bit parallel data input
    .D2(D2), // 1-bit parallel data input
    .D3(D3), // 1-bit parallel data input
    .D4(D4), // 1-bit parallel data input
    .D5(D5), // 1-bit parallel data input
    .D6(D6), // 1-bit parallel data input
    .OCE(OCE), // 1-bit clock enable input
    .REV(1'b0), // Must be tied to logic zero
    .SHIFTIN1(SHIFTIN1), // 1-bit data expansion input
    .SHIFTIN2(SHIFTIN2), // 1-bit data expansion input
    .SR(SR), // 1-bit set/reset input
    .T1(T1), // 1-bit parallel 3-state input
    .T2(T2), // 1-bit parallel 3-state input
    .T3(T3), // 1-bit parallel 3-state input
    .T4(T4), // 1-bit parallel 3-state input
    .TCE(TCE) // 1-bit 3-state signal clock enable input
);

// End of OSERDES_inst instantiation

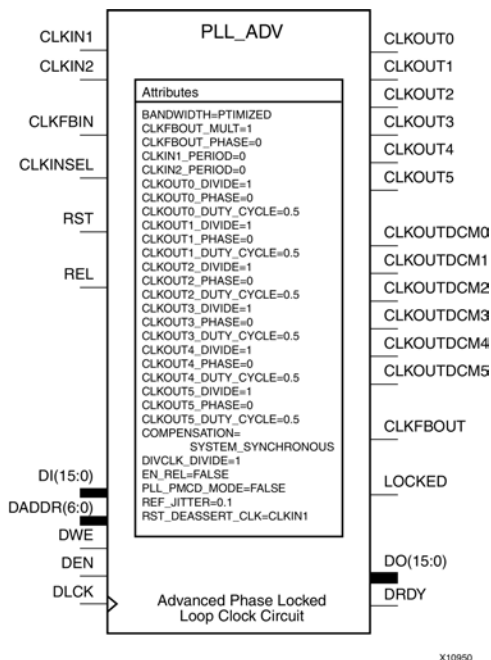
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

PLL_ADV

Primitive: Advanced Phase Locked Loop Clock Circuit



Introduction

The PLL_ADV primitive provides access to all PLL_BASE features. PLL_ADV is only provided in this document as a reference. It can be used for retargeting purposes. For most design situations, use the PLL_BASE primitive or the clocking wizard.

Port Descriptions

| Port | Type | Width | Function |
|-------------------------|--------|-------|--|
| CLKFBDCM | Output | 1 | PLL_ADV pin used for retargeting. PLL feedback used to compensate if the PLL is driving the DCM. If the CLKFBOUT pin is used for this purpose, the software will automatically map to the correct port. |
| CLKFBIN | Input | 1 | Feedback clock input. |
| CLKFBOUT | Output | 1 | Dedicated PLL feedback output. |
| CLKINSEL | Input | 1 | PLL_ADV pin used for retargeting. Connect to a static High or static Low to control the choice of clock input for PLL_ADV. <ul style="list-style-type: none"> High = CLKIN1 Low = CLKIN2 |
| CLKIN1 | Input | 1 | PLL_ADV pin used for retargeting. General clock input. |
| CLKIN2 | Input | 1 | PLL_ADV pin used for retargeting. Secondary clock input. |
| CLKOUTDCM0 - CLKOUTDCM5 | Output | 1 | PLL_ADV pin used for retargeting. User configurable clocks (0 through 5) that can only connect to the DCM within the same CMT as the PLL. |

| Port | Type | Width | Function |
|-------------------|--------|-------|--|
| CLKOUT0 - CLKOUT5 | Output | 1 | User configurable clock outputs (0 through 5) that can be divided versions of the VCO phase outputs (user controllable) from 1 (bypassed) to 128. The input clock and output clocks are phase aligned. |
| DADDR[4:0] | Input | 5 | Provides a reconfiguration address for the dynamic reconfiguration. When not used, all bits must be assigned zeros. |
| DCLK | Input | 1 | Reference clock for the dynamic reconfiguration port. |
| DEN | Input | 1 | Provides the enable control signal to access the dynamic reconfiguration feature. When the dynamic reconfiguration feature is not used, DEN must be tied low. When DEN is tied low, the DO outputs reflect the status signals. |
| DI[15:0] | Input | 16 | Provides reconfiguration data. When not used, all bits must be set to zeros. |
| DO[15:0] | Output | 16 | Provides PLL status or data output when using dynamic reconfiguration. For the DO bus to represent the PLL status, the following connections are required: <ul style="list-style-type: none"> DEN must be tied to GND. DWE must be tied to GND. DADDR bus must be all zeros. DI bus must be all zeros. |
| DRDY | Output | 1 | Provides the response to the DEN signal for the PLL dynamic reconfiguration feature. |
| DWE | Input | 1 | Provides the write enable control signal to write the DI data into the DADDR address. When not used, it must be tied low. |
| LOCKED | Output | 1 | Asynchronous output from the PLL that indicates when the PLL has achieved phase alignment within a predefined window and frequency matching within a predefined PPM range. The PLL automatically locks after power on, no extra reset is required. LOCKED is deasserted if the input clock stops or the phase alignment is violated (e.g., input clock phase shift). The PLL must be reset after LOCKED is deasserted. |
| REL | Input | 1 | Used in the retargeting of the Virtex®-4 PMCD component. Not suggested to be used in other circumstances. |
| RST | Input | 1 | The RST signal is an asynchronous reset for the PLL. The PLL will synchronously re-enable itself when this signal is released (i.e., PLL re-enabled). A reset is required when the input clock conditions change (e.g., frequency). |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed_Values | Default | Description |
|-----------|------|----------------|---------|-------------|
|-----------|------|----------------|---------|-------------|

| Attribute | Type | Allowed_Values | Default | Description |
|--------------------------------|---------------------------|---|-------------|---|
| BANDWIDTH | String | "OPTIMIZED", "HIGH", "LOW" | "OPTIMIZED" | Specifies the PLL programming algorithm affecting the jitter, phase margin and other characteristics of the PLL |
| CLKFBOUT_ DESKEW_ ADJUST | String | "NONE", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14", "15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28", "29", "30", "31" | "NONE" | Used by some IP cores to adjust for additional clock insertion delay for blocks like the PPC440. Unless otherwise instructed, Xilinx suggests leaving at "NONE" to get proper phase alignment. |
| CLKFBOUT_MULT | Integer | 1 to 64 | 1 | Specifies the amount to multiply all CLKOUT clock outputs if a different frequency is desired. This number, in combination with the associated CLKOUT#_DIVIDE value and DIVCLK_DIVIDE value, will determine the output frequency. |
| CLKFBOUT_PHASE | 1 significant digit Float | 0.0 to 360.0 | 0.0 | Specifies the phase offset in degrees of the clock feedback output. Shifting the feedback clock results in a negative phase shift of all output clocks to the PLL. |
| CLKIN1_PERIOD | 3 significant digit Float | Real value specified in units of ns with up to 3 decimal places of precision (ps precision). | 0.0 | Specifies the input period in ns to the PLL CLKIN1 input. Resolution is down to the ps. This information is mandatory and must be supplied when using the CLKIN1 clock input. |

| Attribute | Type | Allowed_Values | Default | Description |
|--|---------------------------|---|---------|--|
| CLKIN2_PERIOD | 3 significant digit Float | Real value specified in units of ns with up to 3 decimal places of precision (ps precision). | 0.0 | Specifies the input period in ns to the PLL CLKIN2 input. Resolution is down to the ps. This information is mandatory and must be supplied when using the CLKIN2 clock input. |
| CLKOUT0_DESKEW_ADJUST - CLKOUT5_DESKEW_ADJUST | String | "NONE", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13", "14", "15", "16", "17", "18", "19", "20", "21", "22", "23", "24", "25", "26", "27", "28", "29", "30", "31" | "NONE" | Parameter to be used in PPC440 designs only. For more information, see the section on clock insertion delays and PLL usage in the <i>Embedded Processor Block User Guide</i> . |
| CLKOUT0_DIVIDE - CLKOUT5_DIVIDE | Integer | 1 to 128 | 1 | Specifies the amount to divide the associated CLKOUT clock output if a different frequency is desired. This number in combination with the CLKFBOUT_MULT and DIVCLK_DIVIDE values will determine the output frequency. |
| CLKOUT0_DUTY_CYCLE - CLKOUT5_DUTY_CYCLE | 2 significant digit Float | 0.01 to 0.99 | 0.50 | Specifies the Duty Cycle of the associated CLKOUT clock output in percentage (i.e., 0.50 will generate a 50% duty cycle). |
| CLKOUT0_PHASE - CLKOUT5_PHASE | 1 significant digit Float | 0.0 to 360.0 | 0.0 | Allows specification of the output phase relationship of the associated CLKOUT clock output in number of degrees offset (i.e., 90 indicates a 90 or 1/4 cycle offset phase offset while 180 indicates a 180 offset or 1/2 cycle phase offset). |

| Attribute | Type | Allowed_Values | Default | Description |
|-----------------------|---------------------------|---|----------------------|---|
| COMPENSATION | String | "SYSTEM_SYNCHRONOUS", "SOURCE_SYNCHRONOUS", "INTERNAL", "EXTERNAL", "DCM2PLL", "PLL2DCM" | "SYSTEM_SYNCHRONOUS" | Specifies the PLL phase compensation for the incoming clock. SYSTEM_SYNCHRONOUS attempts to compensate all clock delay for 0 hold time. SOURCE_SYNCHRONOUS is used when a clock is provided with data and thus phased with the clock. Additional attributes automatically selected by the ISE software - INTERNAL EXTERNAL DCM2PLL PLL2DCM. |
| DIVCLK_DIVIDE | Integer | 1 to 52 | 1 | Specifies the division ratio for all output clocks with respect to the input clock. |
| EN_REL | Boolean | FALSE, TRUE | FALSE | When in PMCD mode (PLL_PMCD_MODE = TRUE), specifies release of divided clock CLKA outputs when the REL input pin is asserted. |
| PLL_PMCD_MODE | Boolean | FALSE, TRUE | FALSE | Enables PLL to act as PMCDs. |
| REF_JITTER | 3 significant digit Float | 0.000 to 1.000 | 0.100 | Allows specification of the expected jitter on the reference clock in order to better optimize PLL performance. A bandwidth setting of OPTIMIZED will attempt to choose the best parameter for input clocking when unknown. If known, then the value provided should be specified in terms of the UI percentage (the maximum peak to peak value) of the expected jitter on the input clock. |
| RESET_ON_LOSS_OF_LOCK | Boolean | FALSE, TRUE | FALSE | Must be set to FALSE, not supported in silicon. |
| RST_DEASSERT_CLK | String | "CLKIN1", "CLKFBIN", | "CLKIN1" | Specifies the deassertion of the RST signal to be synchronous to a selected PMCD input clock. |
| SIM_DEVICE | String | "VIRTEX5", "SPARTAN6" | "VIRTEX5" | Specifies target device in order to properly simulate this component When targeting Virtex®-5 must be set to VIRTEX5. |

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;

-- PLL_ADV: Phase-Lock Loop Clock Circuit
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

PLL_ADV_inst : PLL_ADV
generic map (
    BANDWIDTH => "OPTIMIZED", -- "HIGH", "LOW" or "OPTIMIZED"
    CLKFBOUT_MULT => 1,      -- Multiplication factor for all output clocks

```

```

CLKFBOUT_PHASE => 0.0,      -- Phase shift (degrees) of all output clocks
CLKIN1_PERIOD => 0.000,    -- Clock period (ns) of input clock on CLKIN1
CLKIN2_PERIOD => 0.000,    -- Clock period (ns) of input clock on CLKIN2
CLKOUT0_DIVIDE => 1,       -- Division factor for CLKOUT0 (1 to 128)
CLKOUT0_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT0 (0.01 to 0.99)
CLKOUT0_PHASE => 0.0,      -- Phase shift (degrees) for CLKOUT0 (0.0 to 360.0)
CLKOUT1_DIVIDE => 1,       -- Division factor for CLKOUT1 (1 to 128)
CLKOUT1_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT1 (0.01 to 0.99)
CLKOUT1_PHASE => 0.0,      -- Phase shift (degrees) for CLKOUT1 (0.0 to 360.0)
CLKOUT2_DIVIDE => 1,       -- Division factor for CLKOUT2 (1 to 128)
CLKOUT2_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT2 (0.01 to 0.99)
CLKOUT2_PHASE => 0.0,      -- Phase shift (degrees) for CLKOUT2 (0.0 to 360.0)
CLKOUT3_DIVIDE => 1,       -- Division factor for CLKOUT3 (1 to 128)
CLKOUT3_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT3 (0.01 to 0.99)
CLKOUT3_PHASE => 0.0,      -- Phase shift (degrees) for CLKOUT3 (0.0 to 360.0)
CLKOUT4_DIVIDE => 1,       -- Division factor for CLKOUT4 (1 to 128)
CLKOUT4_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT4 (0.01 to 0.99)
CLKOUT4_PHASE => 0.0,      -- Phase shift (degrees) for CLKOUT4 (0.0 to 360.0)
CLKOUT5_DIVIDE => 1,       -- Division factor for CLKOUT5 (1 to 128)
CLKOUT5_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT5 (0.01 to 0.99)
CLKOUT5_PHASE => 0.0,      -- Phase shift (degrees) for CLKOUT5 (0.0 to 360.0)
COMPENSATION => "SYSTEM_SYNCHRONOUS", -- "SYSTEM_SYNCHRONOUS",
                                         -- "SOURCE_SYNCHRONOUS", "INTERNAL",
                                         -- "EXTERNAL", "DCM2PLL", "PLL2DCM"

DIVCLK_DIVIDE => 1,        -- Division factor for all clocks (1 to 52)
EN_REL => FALSE,          -- Enable release (PMCD mode only)
PLL_PMCD_MODE => FALSE,   -- PMCD Mode, TRUE/FALSE
REF_JITTER => 0.100,      -- Input reference jitter (0.000 to 0.999 UI%)
RST_DEASSERT_CLK => "CLKIN1" -- In PMCD mode, clock to synchronize RST release
port map (
  CLKFBDCM => CLKFBDCM,    -- Output feedback signal used when PLL feeds a DCM
  CLKFBOUT => CLKFBOUT,    -- General output feedback signal
  CLKOUT0 => CLKOUT0,      -- One of six general clock output signals
  CLKOUT1 => CLKOUT1,      -- One of six general clock output signals
  CLKOUT2 => CLKOUT2,      -- One of six general clock output signals
  CLKOUT3 => CLKOUT3,      -- One of six general clock output signals
  CLKOUT4 => CLKOUT4,      -- One of six general clock output signals
  CLKOUT5 => CLKOUT5,      -- One of six general clock output signals
  CLKOUTDCM0 => CLKOUTDCM0, -- One of six clock outputs to connect to the DCM
  CLKOUTDCM1 => CLKOUTDCM1, -- One of six clock outputs to connect to the DCM
  CLKOUTDCM2 => CLKOUTDCM2, -- One of six clock outputs to connect to the DCM
  CLKOUTDCM3 => CLKOUTDCM3, -- One of six clock outputs to connect to the DCM
  CLKOUTDCM4 => CLKOUTDCM4, -- One of six clock outputs to connect to the DCM
  CLKOUTDCM5 => CLKOUTDCM5, -- One of six clock outputs to connect to the DCM
  DO => DO,                -- Dynamic reconfig data output (16-bits)
  DRDY => DRDY,            -- Dynamic reconfig ready output
  LOCKED => LOCKED,        -- Active high PLL lock signal
  CLKFBIN => CLKFBIN,      -- Clock feedback input
  CLKIN1 => CLKIN1,        -- Primary clock input
  CLKIN2 => CLKIN2,        -- Secondary clock input
  CLKINSEL => CLKINSEL,    -- Selects CLKIN1 or CLKIN2
  DADDR => DADDR,          -- Dynamic reconfig address input (5-bits)
  DCLK => DCLK,            -- Dynamic reconfig clock input
  DEN => DEN,              -- Dynamic reconfig enable input
  DI => DI,               -- Dynamic reconfig data input (16-bits)
  DWE => DWE,             -- Dynamic reconfig write enable input
  REL => REL,              -- Clock release input (PMCD mode only)
  RST => RST               -- Asynchronous PLL reset
);

-- End of PLL_ADV_inst instantiation

```

Verilog Instantiation Template

```
// PLL_ADV: Phase-Lock Loop Clock Circuit
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

PLL_ADV #(
    .BANDWIDTH("OPTIMIZED"), // "HIGH", "LOW" or "OPTIMIZED"
    .CLKFBOUT_MULT(1),       // Multiplication factor for all output clocks
    .CLKFBOUT_PHASE(0.0),    // Phase shift (degrees) of all output clocks
    .CLKIN1_PERIOD(0.000),   // Clock period (ns) of input clock on CLKIN1
    .CLKIN2_PERIOD(0.000),   // Clock period (ns) of input clock on CLKIN2
    .CLKOUT0_DIVIDE(1),      // Division factor for CLKOUT0 (1 to 128)
    .CLKOUT0_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT0 (0.01 to 0.99)
    .CLKOUT0_PHASE(0.0),     // Phase shift (degrees) for CLKOUT0 (0.0 to 360.0)
    .CLKOUT1_DIVIDE(1),      // Division factor for CLKOUT1 (1 to 128)
    .CLKOUT1_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT1 (0.01 to 0.99)
    .CLKOUT1_PHASE(0.0),     // Phase shift (degrees) for CLKOUT1 (0.0 to 360.0)
    .CLKOUT2_DIVIDE(1),      // Division factor for CLKOUT2 (1 to 128)
    .CLKOUT2_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT2 (0.01 to 0.99)
    .CLKOUT2_PHASE(0.0),     // Phase shift (degrees) for CLKOUT2 (0.0 to 360.0)
    .CLKOUT3_DIVIDE(1),      // Division factor for CLKOUT3 (1 to 128)
    .CLKOUT3_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT3 (0.01 to 0.99)
    .CLKOUT3_PHASE(0.0),     // Phase shift (degrees) for CLKOUT3 (0.0 to 360.0)
    .CLKOUT4_DIVIDE(1),      // Division factor for CLKOUT4 (1 to 128)
    .CLKOUT4_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT4 (0.01 to 0.99)
    .CLKOUT4_PHASE(0.0),     // Phase shift (degrees) for CLKOUT4 (0.0 to 360.0)
    .CLKOUT5_DIVIDE(1),      // Division factor for CLKOUT5 (1 to 128)
    .CLKOUT5_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT5 (0.01 to 0.99)
    .CLKOUT5_PHASE(0.0),     // Phase shift (degrees) for CLKOUT5 (0.0 to 360.0)
    .COMPENSATION("SYSTEM_SYNCHRONOUS"), // "SYSTEM_SYNCHRONOUS",
                                         // "SOURCE_SYNCHRONOUS", "INTERNAL", "EXTERNAL",
                                         // "DCM2PLL", "PLL2DCM"
    .DIVCLK_DIVIDE(1),       // Division factor for all clocks (1 to 52)
    .EN_REL("FALSE"),        // Enable release (PMCD mode only)
    .PLL_PMCD_MODE("FALSE"), // PMCD Mode, TRUE/FALSE
    .REF_JITTER(0.100),      // Input reference jitter (0.000 to 0.999 UI%)
    .RST_DEASSERT_CLK("CLKIN1") // In PMCD mode, clock to synchronize RST release
) PLL_ADV_inst (
    .CLKFBDCM(CLKFBDCM), // Output feedback signal used when PLL feeds a DCM
    .CLKFBOUT(CLKFBOUT), // General output feedback signal
    .CLKOUT0(CLKOUT0),    // One of six general clock output signals
    .CLKOUT1(CLKOUT1),    // One of six general clock output signals
    .CLKOUT2(CLKOUT2),    // One of six general clock output signals
    .CLKOUT3(CLKOUT3),    // One of six general clock output signals
    .CLKOUT4(CLKOUT4),    // One of six general clock output signals
    .CLKOUT5(CLKOUT5),    // One of six general clock output signals
    .CLKOUTDCM0(CLKOUTDCM0), // One of six clock outputs to connect to the DCM
    .CLKOUTDCM1(CLKOUTDCM1), // One of six clock outputs to connect to the DCM
    .CLKOUTDCM2(CLKOUTDCM2), // One of six clock outputs to connect to the DCM
    .CLKOUTDCM3(CLKOUTDCM3), // One of six clock outputs to connect to the DCM
    .CLKOUTDCM4(CLKOUTDCM4), // One of six clock outputs to connect to the DCM
    .CLKOUTDCM5(CLKOUTDCM5), // One of six clock outputs to connect to the DCM
    .DO(DO),              // Dynamic reconfig data output (16-bits)
    .DRDY(DRDY),          // Dynamic reconfig ready output
    .LOCKED(LOCKED),      // Active high PLL lock signal
    .CLKFBIN(CLKFBIN),    // Clock feedback input
    .CLKIN1(CLKIN1),      // Primary clock input
    .CLKIN2(CLKIN2),      // Secondary clock input
    .CLKINSEL(CLKINSEL),  // Selects '1' = CLKIN1, '0' = CLKIN2
    .DADDR(DADDR),        // Dynamic reconfig address input (5-bits)
    .DCLK(DCLK),          // Dynamic reconfig clock input
    .DEN(DEN),            // Dynamic reconfig enable input
    .DI(DI),              // Dynamic reconfig data input (16-bits)
    .DWE(DWE),            // Dynamic reconfig write enable input
    .REL(REL),            // Clock release input (PMCD mode only)
    .RST(RST)             // Asynchronous PLL reset
);

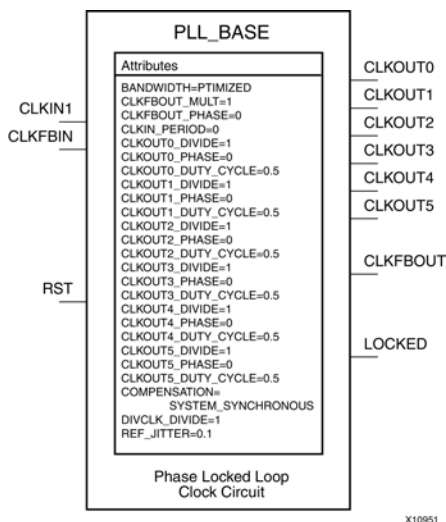
// End of PLL_ADV_inst instantiation
```


For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

PLL_BASE

Primitive: Basic Phase Locked Loop Clock Circuit



Introduction

This design element is a direct sub-set of the PLL_ADV design element, an embedded Phase Locked Loop clock circuit that provides added capabilities for clock synthesis and management both within the FPGA and in circuits external to the FPGA. The PLL_BASE is provided in order to ease the integration for most PLL clocking circuits. However, this primitive does not contain all of the functionality that the PLL can possibly provide. This component allows the input clock to be phase shifted, multiplied and divided, and supports other features, such as modification of the duty cycle and jitter filtering.

Port Descriptions

| Port | Direction | Width | Function |
|-------------------------------|-----------|-------|--|
| Clock Outputs/Inputs | | | |
| CLKOUT0-5 | Output | 1 | One of six phase controlled output clocks from the PLL. |
| CLKFBOUT | Output | 1 | Dedicated PLL feedback output used to determine how the PLL compensates clock network delay. Depending on the type of compensation desired, this output might or might not need to be connected. |
| CLKIN | Input | 1 | Clock source input to the PLL. This pin can be driven by a dedicated clock pin to the FPGA, a DCM output clock pin, or a BUFG output. |
| CLKFBIN | Input | 1 | Clock feedback input. This pin should only be sourced from the CLKFBOUT port. |
| Status Outputs/Control Inputs | | | |
| LOCKED | Output | 1 | Asynchronous output from the PLL that provides you with an indication the PLL has achieved phase alignment and is ready for operation. |
| RST | Input | 1 | Asynchronous reset of the PLL. |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|---|---------|---|----------------------|---|
| COMPENSATION | String | "SYSTEM_SYNCHRONOUS", "SOURCE_SYNCHRONOUS" | "SYSTEM_SYNCHRONOUS" | Specifies the PLL phase compensation for the incoming clock. SYSTEM_SYNCHRONOUS attempts to compensate all clock delay while SOURCE_SYNCHRONOUS is used when a clock is provided with data and thus phased with the clock. |
| BANDWIDTH | String | "HIGH", "LOW", "OPTIMIZED" | "OPTIMIZED" | Specifies the PLL programming algorithm affecting the jitter, phase margin and other characteristics of the PLL. |
| CLKOUT0_DIVIDE, CLKOUT1_DIVIDE, CLKOUT2_DIVIDE, CLKOUT3_DIVIDE, CLKOUT4_DIVIDE, CLKOUT5_DIVIDE | Integer | 1 to 128 | 1 | Specifies the amount to divide the associated CLKOUT clock output if a different frequency is desired. This number in combination with the FBCLKOUT_MULT value determines the output frequency. |
| CLKOUT0_PHASE, CLKOUT1_PHASE, CLKOUT2_PHASE, CLKOUT3_PHASE, CLKOUT4_PHASE, CLKOUT5_PHASE | Real | 0.01 to 360.0 | 0.0 | Allows specification of the output phase relationship of the associated CLKOUT clock output in number of degrees offset (i.e. 90 indicates a 90 degree or ¼ cycle offset phase offset while 180 indicates a 180 degree offset or ½ cycle phase offset). |
| CLKOUT0_DUTY_CYCLE, CLKOUT1_DUTY_CYCLE, CLKOUT2_DUTY_CYCLE, CLKOUT3_DUTY_CYCLE, CLKOUT4_DUTY_CYCLE, CLKOUT5_DUTY_CYCLE | Real | 0.01 to 0.99 | 0.50 | Specifies the Duty Cycle of the associated CLKOUT clock output in percentage (i.e. 0.50 generates a 50% duty cycle). |
| CLKFBOUT_MULT | Integer | 1 to 64 | 1 | Specifies the amount to multiply all CLKOUT clock outputs if a different frequency is desired. This number in combination with the associated CLKOUT#_DIVIDE value determines the output frequency. |
| DIVCLK_DIVIDE | Integer | 1 to 52 | 1 | Specifies the division ratio for all output clocks. |

| Attribute | Type | Allowed Values | Default | Description |
|----------------|------|-----------------|---------|---|
| CLKFBOUT_PHASE | Real | 0.0 to 360 | 0.0 | Specifies the phase offset in degrees of the clock feedback output. |
| REF_JITTER | Real | 0.000 to 0.999 | 0.100 | The reference clock jitter is specified in terms of the UI which is a percentage of the reference clock. The number provided should be the maximum peak to peak value on the input clock. |
| CLKIN_PERIOD | Real | 1.000 to 52.630 | 0.000 | Specified the input period in ns to the PLL CLKIN input. |

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;
```

```
-- PLL_BASE: Phase-Lock Loop Clock Circuit
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
PLL_BASE_inst : PLL_BASE
generic map (
    BANDWIDTH => "OPTIMIZED", -- "HIGH", "LOW" or "OPTIMIZED"
    CLKFBOUT_MULT => 1, -- Multiplication factor for all output clocks
    CLKFBOUT_PHASE => 0.0, -- Phase shift (degrees) of all output clocks
    CLKIN_PERIOD => 0.000, -- Clock period (ns) of input clock on CLKIN
    CLKOUT0_DIVIDE => 1, -- Division factor for CLKOUT0 (1 to 128)
    CLKOUT0_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT0 (0.01 to 0.99)
    CLKOUT0_PHASE => 0.0, -- Phase shift (degrees) for CLKOUT0 (0.0 to 360.0)
    CLKOUT1_DIVIDE => 1, -- Division factor for CLKOUT1 (1 to 128)
    CLKOUT1_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT1 (0.01 to 0.99)
    CLKOUT1_PHASE => 0.0, -- Phase shift (degrees) for CLKOUT1 (0.0 to 360.0)
    CLKOUT2_DIVIDE => 1, -- Division factor for CLKOUT2 (1 to 128)
    CLKOUT2_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT2 (0.01 to 0.99)
    CLKOUT2_PHASE => 0.0, -- Phase shift (degrees) for CLKOUT2 (0.0 to 360.0)
    CLKOUT3_DIVIDE => 1, -- Division factor for CLKOUT3 (1 to 128)
    CLKOUT3_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT3 (0.01 to 0.99)
    CLKOUT3_PHASE => 0.0, -- Phase shift (degrees) for CLKOUT3 (0.0 to 360.0)
    CLKOUT4_DIVIDE => 1, -- Division factor for CLKOUT4 (1 to 128)
    CLKOUT4_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT4 (0.01 to 0.99)
    CLKOUT4_PHASE => 0.0, -- Phase shift (degrees) for CLKOUT4 (0.0 to 360.0)
    CLKOUT5_DIVIDE => 1, -- Division factor for CLKOUT5 (1 to 128)
    CLKOUT5_DUTY_CYCLE => 0.5, -- Duty cycle for CLKOUT5 (0.01 to 0.99)
    CLKOUT5_PHASE => 0.0, -- Phase shift (degrees) for CLKOUT5 (0.0 to 360.0)
    COMPENSATION => "SYSTEM_SYNCHRONOUS", -- "SYSTEM_SYNCHRONOUS",
    -- "SOURCE_SYNCHRONOUS", "INTERNAL",
    -- "EXTERNAL", "DCM2PLL", "PLL2DCM"
    DIVCLK_DIVIDE => 1, -- Division factor for all clocks (1 to 52)
    REF_JITTER => 0.100) -- Input reference jitter (0.000 to 0.999 UI%)
port map (
    CLKFBOUT => CLKFBOUT, -- General output feedback signal
    CLKOUT0 => CLKOUT0, -- One of six general clock output signals
    CLKOUT1 => CLKOUT1, -- One of six general clock output signals
    CLKOUT2 => CLKOUT2, -- One of six general clock output signals
    CLKOUT3 => CLKOUT3, -- One of six general clock output signals
    CLKOUT4 => CLKOUT4, -- One of six general clock output signals
    CLKOUT5 => CLKOUT5, -- One of six general clock output signals
    LOCKED => LOCKED, -- Active high PLL lock signal
    CLKFBIN => CLKFBIN, -- Clock feedback input
    CLKIN => CLKIN, -- Clock input
    RST => RST -- Asynchronous PLL reset
);

-- End of PLL_BASE_inst instantiation
```

Verilog Instantiation Template

```
// PLL_BASE: Phase-Lock Loop Clock Circuit
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

PLL_BASE #(
    .BANDWIDTH("OPTIMIZED"), // "HIGH", "LOW" or "OPTIMIZED"
    .CLKFBOUT_MULT(1),       // Multiplication factor for all output clocks
    .CLKFBOUT_PHASE(0.0),    // Phase shift (degrees) of all output clocks
    .CLKIN_PERIOD(0.000),    // Clock period (ns) of input clock on CLKIN
    .CLKOUT0_DIVIDE(1),      // Division factor for CLKOUT0 (1 to 128)
    .CLKOUT0_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT0 (0.01 to 0.99)
    .CLKOUT0_PHASE(0.0),     // Phase shift (degrees) for CLKOUT0 (0.0 to 360.0)
    .CLKOUT1_DIVIDE(1),      // Division factor for CLKOUT1 (1 to 128)
    .CLKOUT1_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT1 (0.01 to 0.99)
    .CLKOUT1_PHASE(0.0),     // Phase shift (degrees) for CLKOUT1 (0.0 to 360.0)
    .CLKOUT2_DIVIDE(1),      // Division factor for CLKOUT2 (1 to 128)
    .CLKOUT2_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT2 (0.01 to 0.99)
    .CLKOUT2_PHASE(0.0),     // Phase shift (degrees) for CLKOUT2 (0.0 to 360.0)
    .CLKOUT3_DIVIDE(1),      // Division factor for CLKOUT3 (1 to 128)
    .CLKOUT3_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT3 (0.01 to 0.99)
    .CLKOUT3_PHASE(0.0),     // Phase shift (degrees) for CLKOUT3 (0.0 to 360.0)
    .CLKOUT4_DIVIDE(1),      // Division factor for CLKOUT4 (1 to 128)
    .CLKOUT4_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT4 (0.01 to 0.99)
    .CLKOUT4_PHASE(0.0),     // Phase shift (degrees) for CLKOUT4 (0.0 to 360.0)
    .CLKOUT5_DIVIDE(1),      // Division factor for CLKOUT5 (1 to 128)
    .CLKOUT5_DUTY_CYCLE(0.5), // Duty cycle for CLKOUT5 (0.01 to 0.99)
    .CLKOUT5_PHASE(0.0),     // Phase shift (degrees) for CLKOUT5 (0.0 to 360.0)
    .COMPENSATION("SYSTEM_SYNCHRONOUS"), // "SYSTEM_SYNCHRONOUS",
                                         // "SOURCE_SYNCHRONOUS", "INTERNAL", "EXTERNAL",
                                         // "DCM2PLL", "PLL2DCM"
    .DIVCLK_DIVIDE(1),       // Division factor for all clocks (1 to 52)
    .REF_JITTER(0.100)      // Input reference jitter (0.000 to 0.999 UI%)
) PLL_BASE_inst (
    .CLKFBOUT(CLKFBOUT),    // General output feedback signal
    .CLKOUT0(CLKOUT0),      // One of six general clock output signals
    .CLKOUT1(CLKOUT1),      // One of six general clock output signals
    .CLKOUT2(CLKOUT2),      // One of six general clock output signals
    .CLKOUT3(CLKOUT3),      // One of six general clock output signals
    .CLKOUT4(CLKOUT4),      // One of six general clock output signals
    .CLKOUT5(CLKOUT5),      // One of six general clock output signals
    .LOCKED(LOCKED),        // Active high PLL lock signal
    .CLKFBIN(CLKFBIN),      // Clock feedback input
    .CLKIN(CLKIN),          // Clock input
    .RST(RST)               // Asynchronous PLL reset
);

// End of PLL_BASE_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

PPC440

Primitive: Power PC 440 CPU Core

Introduction

This design element is a dual issue, superscalar processor that provides significant performance improvement over the older PowerPC® 405 while implementing the same instruction set architecture.

Design Entry Method

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

For More Information

- See the [IBM PPC440x5 CPU Core User's Manual](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).
- See the [Virtex-5 FPGA User Guide](#).

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
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//           All FPGA
// Xilinx HDL Libraries Guide, version 12.3

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

// End of PULLDOWN_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

PULLUP

Primitive: Resistor to VCC for Input PADS, Open-Drain, and 3-State Outputs



Introduction

This design element allows for an input, 3-state output or bi-directional port to be driven to a weak high value when not being driven by an internal or external source. This element establishes a High logic level for open-drain elements and macros when all the drivers are off.

Port Descriptions

| Port | Direction | Width | Function |
|------|-----------|-------|--|
| O | Output | 1 | Pullup output (connect directly to top level port) |

Design Entry Method

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

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- PULLUP: I/O Buffer Weak Pull-up
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//           All FPGA, CoolRunner-II
// Xilinx HDL Libraries Guide, version 12.3

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

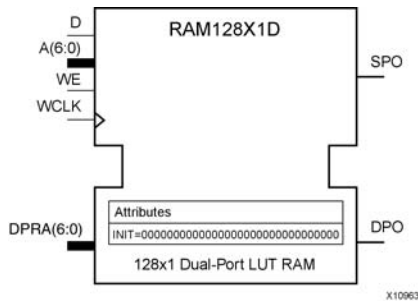
// End of PULLUP_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM128X1D

Primitive: 128-Deep by 1-Wide Dual Port Random Access Memory (Select RAM)



Introduction

This design element is a 128-bit deep by 1-bit wide random access memory and has a read/write port that writes the value on the D input data pin when the write enable (WE) is high to the location specified by the A address bus. This happens shortly after the rising edge of the WCLK and that same value is reflected in the data output SPO. When WE is low, an asynchronous read is initiated in which the contents of the memory location specified by the A address bus is output asynchronously to the SPO output. The read port can perform asynchronous read access of the memory by changing the value of the address bus DPRA, and by outputting that value to the DPO data output.

Port Descriptions

| Port | Direction | Width | Function |
|------|-----------|-------|--|
| SPO | Output | 1 | Read/Write port data output addressed by A |
| DPO | Output | 1 | Read port data output addressed by DPRA |
| D | Input | 1 | Write data input addressed by A |
| A | Input | 7 | Read/Write port address bus |
| DPRA | Input | 7 | Read port address bus |
| WE | Input | 1 | Write Enable |
| WCLK | Input | 1 | Write clock (reads are asynchronous) |

If instantiated, the following connections should be made to this component:

- Tie the WCLK input to the desired clock source, the D input to the data source to be stored and the DPO output to an FDCE D input or other appropriate data destination.
- Optionally, the SPO output can also be connected to the appropriate data destination or else left unconnected.
- The WE clock enable pin should be connected to the proper write enable source in the design.
- The 7-bit A bus should be connected to the source for the read/write addressing and the 7-bit DPRA bus should be connected to the appropriate read address connections.
- An optional INIT attribute consisting of a 128-bit Hexadecimal value can be specified to indicate the initial contents of the RAM.

If left unspecified, the initial contents default to all zeros.

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 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;

-- RAM128X1D: 128-deep by 1-wide positive edge write, asynchronous read
--             dual-port distributed LUT RAM
--             Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM128X1D_inst : RAM128X1D
generic map (
  INIT => X"00000000000000000000000000000000"
)
port map (
  DPO => DPO,      -- Read/Write port 1-bit output
  SPO => SPO,      -- Read port 1-bit output
  A => A,          -- Read/Write port 7-bit address input
  D => D,          -- RAM data input
  DPRA => DPRA,    -- Read port 7-bit address input
  WCLK => WCLK,    -- Write clock input
  WE => WE         -- RAM data input
);

-- End of RAM128X1D_inst instantiation
```

Verilog Instantiation Template

```
// RAM128X1D: 128-deep by 1-wide positive edge write, asynchronous read
//             dual-port distributed LUT RAM
//             Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAM128X1D #(
  .INIT(128'h00000000000000000000000000000000)
) RAM128X1D_inst (
  .DPO(DPO),      // Read port 1-bit output
  .SPO(SPO),      // Read/Write port 1-bit output
  .A(A),          // Read/Write port 7-bit address input
  .D(D),          // RAM data input
  .DPRA(DPRA),    // Read port 7-bit address input
  .WCLK(WCLK),    // Write clock input
  .WE(WE)         // Write enable input
);

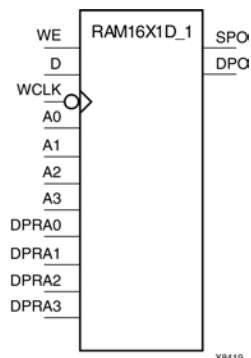
// End of RAM128X1D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM16X1D_1

Primitive: 16-Deep by 1-Wide Static Dual Port Synchronous RAM with Negative-Edge Clock



Introduction

This is a 16-word by 1-bit static dual port random access memory with synchronous write capability and negative-edge clock. The device has two separate address ports: the read address (DPRA3:DPRA0) and the write address (A3:A0). These two address ports are asynchronous. The read address controls the location of the data driven out of the output pin (DPO), and the write address controls the destination of a valid write transaction.

When the write enable (WE) is set to Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 4-bit write address. For predictable performance, write address and data inputs must be stable before a High-to-Low WCLK transition. This RAM block assumes an active-High (WCLK). (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

You can initialize RAM16X1D_1 during configuration using the INIT attribute.

The SPO output reflects the data in the memory cell addressed by A3:A0. The DPO output reflects the data in the memory cell addressed by DPRA3:DPRA0.

Note The write process is not affected by the address on the read address port.

Logic Table

Mode selection is shown in the following logic table:

| Inputs | | | Outputs | |
|---|------|---|---------|--------|
| WE (mode) | WCLK | D | SPO | DPO |
| 0 (read) | X | X | data_a | data_d |
| 1 (read) | 0 | X | data_a | data_d |
| 1 (read) | 1 | X | data_a | data_d |
| 1 (write) | ↓ | D | D | data_d |
| 1 (read) | ↑ | X | data_a | data_d |
| data_a = word addressed by bits A3:A0 | | | | |
| data_d = word addressed by bits DPRA3:DPRA0 | | | | |

Port Descriptions

| Port | Direction | Width | Function |
|-------|-----------|-------|-----------------------------|
| DPO | Output | 1 | Read-only 1-Bit data output |
| SPO | Output | 1 | R/W 1-Bit data output |
| A0 | Input | 1 | R/W address[0] input |
| A1 | Input | 1 | R/W address[1] input |
| A2 | Input | 1 | R/W address[2] input |
| A3 | Input | 1 | R/W address[3] input |
| D | Input | 1 | Write 1-Bit data input |
| DPRA0 | Input | 1 | Read-only address[0] input |
| DPRA1 | Input | 1 | Read-only address[1] input |
| DPRA2 | Input | 1 | Read-only address[2] input |
| DPRA3 | Input | 1 | Read-only address[3] input |
| WCLK | Input | 1 | Write clock input |
| WE | Input | 1 | Write enable input |

Design Entry Method

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

Available Attributes

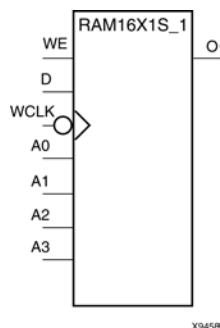
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|--|
| INIT | Hexadecimal | Any 16-Bit Value | All zeros | Initializes RAMs, registers, and look-up tables. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM16X1S_1

Primitive: 16-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock



Introduction

This element is a 16-word by 1-bit static random access memory with synchronous write capability and negative-edge clock. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 4-bit address (A3:A0). For predictable performance, address and data inputs must be stable before a High-to-Low WCLK transition. This RAM block assumes an active-Low (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can initialize this element during configuration using the INIT attribute.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|---|---------|
| WE(mode) | WCLK | D | O |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↓ | D | D |
| 1 (read) | ↑ | X | Data |
| Data = word addressed by bits A3:A0 | | | |

Design Entry Method

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

Available Attributes

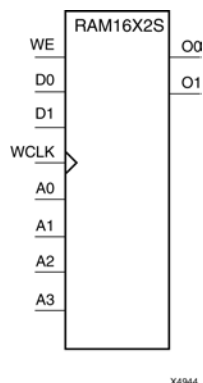
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|--|
| INIT | Hexadecimal | Any 16-Bit Value | All zeros | Specifies initial contents of the RAM. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM16X2S

Primitive: 16-Deep by 2-Wide Static Synchronous RAM



Introduction

This element is a 16-word by 2-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on the data input (D1:D0) into the word selected by the 4-bit address (A3:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O1:O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can use the INIT_xx properties to specify the initial contents of a wide RAM. INIT_00 initializes the RAM cells corresponding to the O0 output, INIT_01 initializes the cells corresponding to the O1 output, etc. For example, a RAM16X2S instance is initialized by INIT_00 and INIT_01 containing 4 hex characters each. A RAM16X8S instance is initialized by eight properties INIT_00 through INIT_07 containing 4 hex characters each. A RAM64x2S instance is completely initialized by two properties INIT_00 and INIT_01 containing 16 hex characters each.

Except for Virtex-4 devices, the initial contents of this element cannot be specified directly.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|-------|---------|
| WE (mode) | WCLK | D1:D0 | O1:O0 |
| 0 (read) | X | X | Data |
| 1(read) | 0 | X | Data |
| 1(read) | 1 | X | Data |
| 1(write) | ↑ | D1:D0 | D1:D0 |
| 1(read) | ↓ | X | Data |
| Data = word addressed by bits A3:A0 | | | |

Design Entry Method

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

Available Attributes

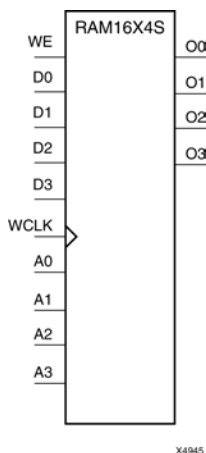
| Attribute | Type | Allowed Values | Default | Description |
|--------------------|-------------|------------------|-----------|--|
| INIT_00 to INIT_01 | Hexadecimal | Any 16-Bit Value | All zeros | Initializes RAMs, registers, and look-up tables. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM16X4S

Primitive: 16-Deep by 4-Wide Static Synchronous RAM



Introduction

This element is a 16-word by 4-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on the data input (D3:D0) into the word selected by the 4-bit address (A3:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O3:O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

Logic Table

| Inputs | | | Outputs |
|--------------------------------------|------|-------|---------|
| WE (mode) | WCLK | D3:D0 | O3:O0 |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↑ | D3:D0 | D3:D0 |
| 1 (read) | ↓ | X | Data |
| Data = word addressed by bits A3:A0. | | | |

Design Entry Method

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

Available Attributes

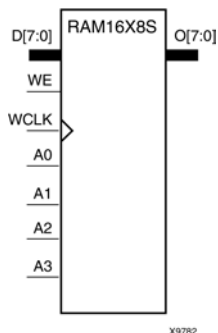
| Attribute | Type | Allowed Values | Default | Description |
|--------------------|-------------|------------------|-----------|-------------|
| INIT_00 to INIT_03 | Hexadecimal | Any 16-Bit Value | All zeros | INIT of RAM |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM16X8S

Primitive: 16-Deep by 8-Wide Static Synchronous RAM



Introduction

This element is a 16-word by 8-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on data inputs (D7:D0) into the word selected by the 4-bit address (A3:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O7:O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|-------|---------|
| WE (mode) | WCLK | D7:D0 | O7:O0 |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↑ | D7:D0 | D7:D0 |
| 1 (read) | ↓ | X | Data |
| Data = word addressed by bits A3–A0 | | | |

Design Entry Method

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

Available Attributes

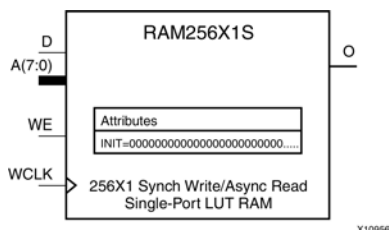
| Attribute | Type | Allowed Values | Default | Description |
|--------------------|-------------|------------------|-----------|--|
| INIT_00 to INIT_07 | Hexadecimal | Any 16-Bit Value | All zeros | Initializes RAMs, registers, and look-up tables. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM256X1S

Primitive: 256-Deep by 1-Wide Random Access Memory (Select RAM)



Introduction

This design element is a 256-bit deep by 1-bit wide random access memory with synchronous write and asynchronous read capability. This RAM is implemented using the LUT resources of the device (also known as Select RAM), and does not consume any of the block RAM resources of the device. If a synchronous read capability is preferred, a register can be attached to the output and placed in the same slice as long as the same clock is used for both the RAM and the register. The RAM256X1S has an active, High write enable, WE, so that when that signal is High, and a rising edge occurs on the WCLK pin, a write is performed recording the value of the D input data pin into the memory array. The output O displays the contents of the memory location addressed by A, regardless of the WE value. When a write is performed, the output is updated to the new value shortly after the write completes.

Port Descriptions

| Port | Direction | Width | Function |
|------|-----------|-------|--|
| O | Output | 1 | Read/Write port data output addressed by A |
| D | Input | 1 | Write data input addressed by A |
| A | Input | 8 | Read/Write port address bus |
| WE | Input | 1 | Write Enable |
| WCLK | Input | 1 | Write clock (reads are asynchronous) |

Design Entry Method

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

If instantiated, the following connections should be made to this component:

- Tie the WCLK input to the desired clock source, the D input to the data source to be stored, and the O output to an FDCE D input or other appropriate data destination.
- The WE clock enable pin should be connected to the proper write enable source in the design.
- The 8-bit A bus should be connected to the source for the read/write.
- An optional INIT attribute consisting of a 256-bit Hexadecimal value can be specified to indicate the initial contents of the RAM.

If left unspecified, the initial contents default to all zeros.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|-------------------|-----------|--|
| INIT | Hexadecimal | Any 256-Bit Value | All zeros | Specifies the 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;

-- RAM256X1S: 256-deep by 1-wide positive edge write, asynchronous read
--             single-port distributed LUT RAM
--             Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM256X1S_inst : RAM256X1S
generic map (
  INIT => X"0000000000000000000000000000000000000000000000000000000000000000"
)
port map (
  O => O,  -- Read/Write port 1-bit output
  A => A,  -- Read/Write port 8-bit address input
  D => D,  -- RAM data input
  WCLK => WCLK, -- Write clock input
  WE => WE -- Write enable input
);

-- End of RAM256X1S_inst instantiation
```

Verilog Instantiation Template

```
// RAM256X1S: 256-deep by 1-wide positive edge write, asynchronous read
//             single-port distributed LUT RAM
//             Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAM256X1S #(
  .INIT(256'h0000000000000000000000000000000000000000000000000000000000000000)
) RAM256X1S_inst (
  .O(O),      // Readw/rite port 1-bit output
  .A(A),      // Readw/rite port 8-bit address input
  .WE(WE),    // Write enable input
  .WCLK(WCLK), // Write clock input
  .D(D)       // RAM data input
);

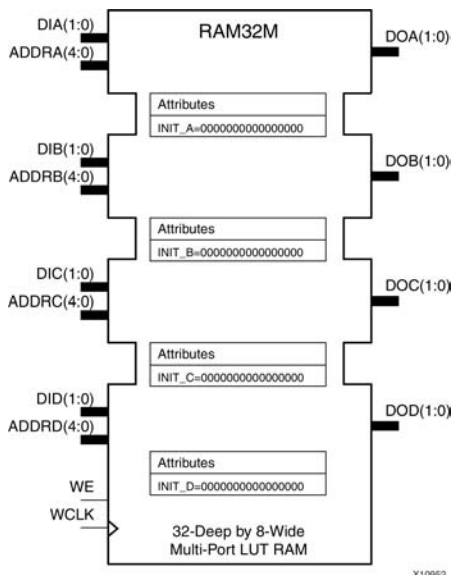
// End of RAM256X1S_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM32M

Primitive: 32-Deep by 8-bit Wide Multi Port Random Access Memory (Select RAM)



Introduction

This design element is a 32-bit deep by 8-bit wide, multi-port, random access memory with synchronous write and asynchronous independent, 2-bit, wide-read capability. This RAM is implemented using the LUT resources of the device known as SelectRAM™, and does not consume any of the Block RAM resources of the device. The RAM32M is implemented in a single slice and consists of one 8-bit write, 2-bit read port and three separate 2-bit read ports from the same memory. This configuration allows for byte-wide write and independent 2-bit read access RAM. If the DIA, DIB, DIC and DID inputs are all tied to the same data inputs, the RAM can become a 1 read/write port, 3 independent read port, 32x2 quad port memory. If DID is grounded, DOD is not used, while ADDRA, ADDRb and ADDRc are tied to the same address, the RAM becomes a 32x6 simple dual port RAM. If ADDRd is tied to ADDRA, ADDRb, and ADDRc, then the RAM is a 32x8 single port RAM. There are several other possible configurations for this RAM.

Port Descriptions

| Port | Direction | Width | Function |
|-------|-----------|-------|--|
| DOA | Output | 2 | Read port data outputs addressed by ADDRA |
| DOB | Output | 2 | Read port data outputs addressed by ADDRb |
| DOC | Output | 2 | Read port data outputs addressed by ADDRc |
| DOD | Output | 2 | Read/Write port data outputs addressed by ADDRd |
| DIA | Input | 2 | Write data inputs addressed by ADDRd (read output is addressed by ADDRA) |
| DIB | Input | 2 | Write data inputs addressed by ADDRd (read output is addressed by ADDRb) |
| DIC | Input | 2 | Write data inputs addressed by ADDRd (read output is addressed by ADDRc) |
| DID | Input | 2 | Write data inputs addressed by ADDRd |
| ADDRA | Input | 5 | Read address bus A |
| ADDRb | Input | 5 | Read address bus B |
| ADDRc | Input | 5 | Read address bus C |
| ADDRd | Input | 5 | 8-bit data write port, 2-bit data read port address bus D |
| WE | Input | 1 | Write Enable |
| WCLK | Input | 1 | Write clock (reads are asynchronous) |

Design Entry Method

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

This element can be inferred by some synthesis tools by describing a RAM with a synchronous write and asynchronous read capability. Consult your synthesis tool documentation for details on RAM inference capabilities and coding examples. Xilinx suggests that you instantiate RAM32Ms if you have a need to implicitly specify the RAM function, or if you need to manually place or relationally place the component. If a synchronous read capability is desired, the RAM32M outputs can be connected to an FDRSE (FDCPE is asynchronous set/reset is necessary) in order to improve the output timing of the function. However, this is not necessary for the proper operation of the RAM.

If you want to have the data clocked on the negative edge of a clock, an inverter can be described on the clock input to this component. This inverter will be absorbed into the block, giving you the ability to write to the RAM on falling clock edges.

If instantiated, the following connections should be made to this component. Tie the WCLK input to the desired clock source, the DIA, DIB, DIC and DID inputs to the data source to be stored and the DOA, DOB, DOC and DOD outputs to an FDCE D input or other appropriate data destination or left unconnected if not used. The WE clock enable pin should be connected to the proper write enable source in the design. The 5-bit ADDRd bus should be connected to the source for the read/write addressing and the 5-bit ADDRA, ADDRb and ADDRc buses should be connected to the appropriate read address connections. The optional INIT_A, INIT_B, INIT_C and INIT_D attributes consisting of a 64-bit hexadecimal values that specifies each port's initial memory contents can be specified. The INIT value correlates to the RAM addressing by the following equation: $ADDRy[z] = INIT_y[2*z+1:2*z]$. For instance, if the RAM ADDRc port is addressed to 00001, then the INIT_C[3:2] values would be the initial values shown on the DOC port before the first write occurs at that address. If left unspecified, the initial contents will default to all zeros.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|--|
| INIT_A | Hexadecimal | Any 64-Bit Value | All zeros | Specifies the initial contents of the RAM on the A port. |
| INIT_B | Hexadecimal | Any 64-Bit Value | All zeros | Specifies the initial contents of the RAM on the B port. |
| INIT_C | Hexadecimal | Any 64-Bit Value | All zeros | Specifies the initial contents of the RAM on the C port. |
| INIT_D | Hexadecimal | Any 64-Bit Value | All zeros | Specifies the initial contents of the RAM on the D port. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- RAM32M: 32-deep by 8-wide Multi Port LUT RAM
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM32M_inst : RAM32M
generic map (
  INIT_A => X"0000000000000000", -- Initial contents of A port
  INIT_B => X"0000000000000000", -- Initial contents of B port
  INIT_C => X"0000000000000000", -- Initial contents of C port
  INIT_D => X"0000000000000000") -- Initial contents of D port
port map (
  DOA => DOA, -- Read port A 2-bit output
  DOB => DOB, -- Read port B 2-bit output
  DOC => DOC, -- Read port C 2-bit output
  DOD => DOD, -- Read/Write port D 2-bit output
  ADDRA => ADDRA, -- Read port A 5-bit address input
  ADDR B => ADDR B, -- Read port B 5-bit address input
  ADDR C => ADDR C, -- Read port C 5-bit address input
  ADDR D => ADDR D, -- Read/Write port D 5-bit address input
  DIA => DIA, -- RAM 2-bit data write input addressed by ADDR D,
               -- read addressed by ADDRA
  DIB => DIB, -- RAM 2-bit data write input addressed by ADDR D,
               -- read addressed by ADDR B
  DIC => DIC, -- RAM 2-bit data write input addressed by ADDR D,
               -- read addressed by ADDR C
  DID => DID, -- RAM 2-bit data write input addressed by ADDR D,
               -- read addressed by ADDR D
  WCLK => WCLK, -- Write clock input
  WE => WE      -- Write enable input
);
-- End of RAM32M_inst instantiation

```

Verilog Instantiation Template

```
// RAM32M: 32-deep by 8-wide Multi Port LUT RAM
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAM32M #(
    .INIT_A(64'h0000000000000000), // Initial contents of A Port
    .INIT_B(64'h0000000000000000), // Initial contents of B Port
    .INIT_C(64'h0000000000000000), // Initial contents of C Port
    .INIT_D(64'h0000000000000000) // Initial contents of D Port
) RAM32M_inst (
    .DOA(DOA),      // Read port A 2-bit output
    .DOB(DOB),      // Read port B 2-bit output
    .DOC(DOC),      // Read port C 2-bit output
    .DOD(DOD),      // Readw/rite port D 2-bit output
    .ADDRA(ADDRA),  // Read port A 5-bit address input
    .ADDRB(ADDRB),  // Read port B 5-bit address input
    .ADDRC(ADDRC),  // Read port C 5-bit address input
    .ADDRD(ADDRD),  // Readw/rite port D 5-bit address input
    .DIA(DIA),      // RAM 2-bit data write input addressed by ADDRd,
                    //      read addressed by ADDRA
    .DIB(DIB),      // RAM 2-bit data write input addressed by ADDRd,
                    //      read addressed by ADDRb
    .DIC(DIC),      // RAM 2-bit data write input addressed by ADDRd,
                    //      read addressed by ADDRc
    .DID(DID),      // RAM 2-bit data write input addressed by ADDRd,
                    //      read addressed by ADDRd
    .WCLK(WCLK),    // Write clock input
    .WE(WE)         // Write enable input
);

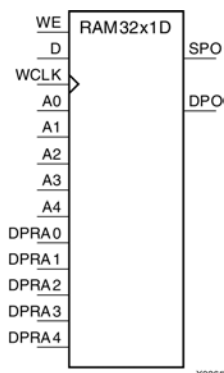
// End of RAM32M_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM32X1D

Primitive: 32-Deep by 1-Wide Static Dual Port Synchronous RAM



Introduction

The design element is a 32-word by 1-bit static dual port random access memory with synchronous write capability. The device has two separate address ports: the read address (DPRA4:DPRA0) and the write address (A4:A0). These two address ports are completely 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 5-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. You can initialize RAM32X1D during configuration using the INIT attribute. Mode selection is shown in the following logic table.

The SPO output reflects the data in the memory cell addressed by A4:A0. The DPO output reflects the data in the memory cell addressed by DPRA4:DPRA0. The write process is not affected by the address on the read address port.

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 |

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 | 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;

-- RAM32X1D: 32 x 1 positive edge write, asynchronous read
--           dual-port distributed RAM
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM32X1D_inst : RAM32X1D
generic map (
  INIT => X"00000000") -- Initial contents of RAM
port map (
  DPO => DPO,          -- Read-only 1-bit data output
  SPO => SPO,          -- R/W 1-bit data output
  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
  A4 => A4,            -- R/W address[4] 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
  DPRA4 => DPRA4,      -- Read-only address[4] input bit
  WCLK => WCLK,        -- Write clock input
  WE => WE             -- Write enable input
);

-- End of RAM32X1D_inst instantiation
```

Verilog Instantiation Template

```
// RAM32X1D: 32 x 1 positive edge write, asynchronous read dual-port distributed RAM
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAM32X1D #(
    .INIT(32'h00000000) // Initial contents of RAM
) RAM32X1D_inst (
    .DPO(DPO),          // Read-only 1-bit data output
    .SPO(SPO),          // Rw/ 1-bit data output
    .A0(A0),            // Rw/ address[0] input bit
    .A1(A1),            // Rw/ address[1] input bit
    .A2(A2),            // Rw/ address[2] input bit
    .A3(A3),            // Rw/ address[3] input bit
    .A4(A4),            // Rw/ address[4] 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
    .DPRA4(DPRA4),      // Read-only address[4] input bit
    .WCLK(WCLK),        // Write clock input
    .WE(WE)             // Write enable input
);

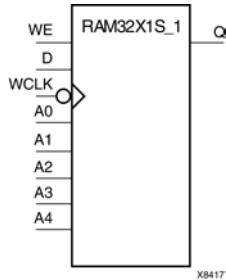
// End of RAM32X1D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM32X1S_1

Primitive: 32-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock



Introduction

The design element is a 32-word by 1-bit static random access memory with synchronous write capability. When the write enable is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 5-bit address (A4:A0). For predictable performance, address and data inputs must be stable before a High-to-Low (WCLK) transition. This RAM block assumes an active-Low (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins. You can initialize RAM32X1S_1 during configuration using the INIT attribute.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|---|---------|
| WE (Mode) | WCLK | D | O |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↓ | D | D |
| 1 (read) | ↑ | X | Data |
| Data = word addressed by bits A4:A0 | | | |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Descriptions |
|-----------|-------------|------------------|---------|--|
| INIT | Hexadecimal | Any 32-Bit Value | 0 | Initializes RAMs, registers, and look-up tables. |

VHDL Instantiation Template

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

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

-- RAM32X1S_1: 32 x 1 negedge write distributed (LUT) RAM
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM32X1S_1_inst : RAM32X1S_1
generic map (
    INIT => X"00000000")
port map (
    O => O,          -- RAM output
    A0 => A0,         -- RAM address[0] input
    A1 => A1,         -- RAM address[1] input
    A2 => A2,         -- RAM address[2] input
    A3 => A3,         -- RAM address[3] input
    A4 => A4,         -- RAM address[4] input
    D => D,           -- RAM data input
    WCLK => WCLK,     -- Write clock input
    WE => WE          -- Write enable input
);

-- End of RAM32X1S_1_inst instantiation
```

Verilog Instantiation Template

```
// RAM32X1S_1: 32 x 1 negedge write distributed (LUT) RAM
//      All FPGA
// Xilinx HDL Libraries Guide, version 12.3

RAM32X1S_1 #(
    .INIT(32'h00000000) // Initial contents of RAM
)RAM32X1S_1_inst (
    .O(O),              // RAM output
    .A0(A0),            // RAM address[0] input
    .A1(A1),            // RAM address[1] input
    .A2(A2),            // RAM address[2] input
    .A3(A3),            // RAM address[3] input
    .A4(A4),            // RAM address[4] input
    .D(D),              // RAM data input
    .WCLK(WCLK),        // Write clock input
    .WE(WE)             // Write enable input
);

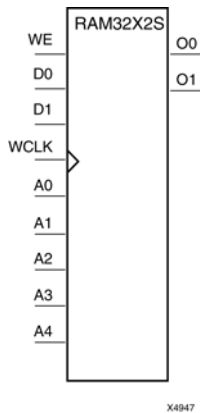
// End of RAM32X1S_1_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM32X2S

Primitive: 32-Deep by 2-Wide Static Synchronous RAM



Introduction

The design element is a 32-word by 2-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any positive transition on (WCLK) loads the data on the data input (D1-D0) into the word selected by the 5-bit address (A4-A0). For predictable performance, address and data inputs must be stable before a Low-to-High (WCLK) transition. This RAM block assumes an active-High (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block. The signal output on the data output pins (O1-O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can use the INIT_00 and INIT_01 properties to specify the initial contents of RAM32X2S.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|-------|---------|
| WE (Mode) | WCLK | D | O0-O1 |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↑ | D1:D0 | D1:D0 |
| 1 (read) | ↓ | X | Data |
| Data = word addressed by bits A4:A0 | | | |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Descriptions |
|-----------|-------------|------------------|-----------|------------------------|
| INIT_00 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 0 of RAM. |
| INIT_01 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 1 of RAM. |

VHDL Instantiation Template

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

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

-- RAM32X2S: 32 x 2 posedge write distributed (LUT) RAM
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM32X2S_inst : RAM32X2S
generic map (
    INIT_00 => X"00000000", -- INIT for bit 0 of RAM
    INIT_01 => X"00000000") -- INIT for bit 1 of RAM
port map (
    O0 => O0,      -- RAM data[0] output
    O1 => O1,      -- RAM data[1] output
    A0 => A0,      -- RAM address[0] input
    A1 => A1,      -- RAM address[1] input
    A2 => A2,      -- RAM address[2] input
    A3 => A3,      -- RAM address[3] input
    A4 => A4,      -- RAM address[4] input
    D0 => D0,      -- RAM data[0] input
    D1 => D1,      -- RAM data[1] input
    WCLK => WCLK,  -- Write clock input
    WE => WE       -- Write enable input
);

-- End of RAM32X2S_inst instantiation
```

Verilog Instantiation Template

```
// RAM32X2S: 32 x 2 posedge write distributed (LUT) RAM
//           All FPGA
// Xilinx HDL Libraries Guide, version 12.3

RAM32X2S #(
    .INIT_00(32'h00000000), // INIT for bit 0 of RAM
    .INIT_01(32'h00000000) // INIT for bit 1 of RAM
) RAM32X2S_inst (
    .O0(O0),      // RAM data[0] output
    .O1(O1),      // RAM data[1] output
    .A0(A0),      // RAM address[0] input
    .A1(A1),      // RAM address[1] input
    .A2(A2),      // RAM address[2] input
    .A3(A3),      // RAM address[3] input
    .A4(A4),      // RAM address[4] input
    .D0(D0),      // RAM data[0] input
    .D1(D1),      // RAM data[1] input
    .WCLK(WCLK),  // Write clock input
    .WE(WE)       // Write enable input
);

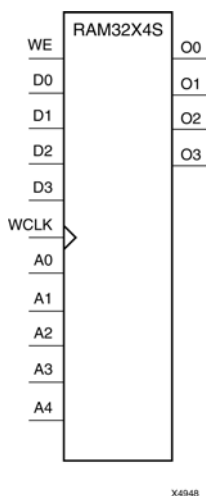
// End of RAM32X2S_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM32X4S

Primitive: 32-Deep by 4-Wide Static Synchronous RAM



Introduction

This design element is a 32-word by 4-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on the data inputs (D3-D0) into the word selected by the 5-bit address (A4:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O3-O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|-------|---------|
| WE | WCLK | D3-D0 | O3-O0 |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↑ | D3:D0 | D3:D0 |
| 1 (read) | ↓ | X | Data |
| Data = word addressed by bits A4:A0 | | | |

Design Entry Method

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

Available Attributes

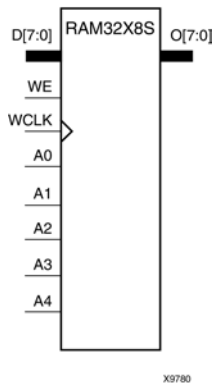
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|------------------------|
| INIT_00 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 0 of RAM. |
| INIT_01 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 1 of RAM. |
| INIT_02 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 2 of RAM. |
| INIT_03 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 3 of RAM. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM32X8S

Primitive: 32-Deep by 8-Wide Static Synchronous RAM



Introduction

This design element is a 32-word by 8-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on the data inputs (D7:D0) into the word selected by the 5-bit address (A4:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O7:O0) is the data that is stored in the RAM at the location defined by the values on the address pins.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|-------|---------|
| WE (mode) | WCLK | D7:D0 | O7:O0 |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↑ | D7:D0 | D7:D0 |
| 1 (read) | ↓ | X | Data |
| Data = word addressed by bits A4:A0 | | | |

Design Entry Method

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

Available Attributes

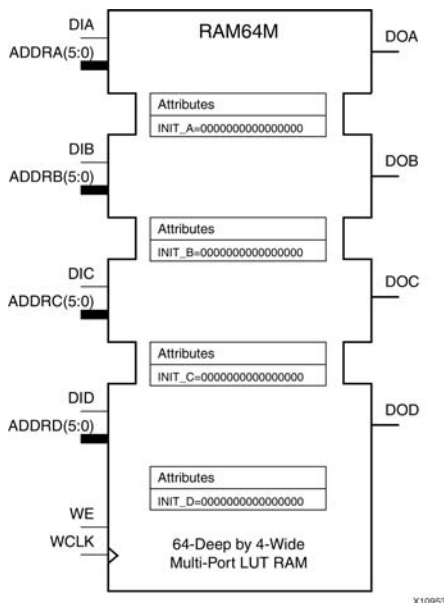
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|------------------------|
| INIT_00 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 0 of RAM. |
| INIT_01 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 1 of RAM. |
| INIT_02 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 2 of RAM. |
| INIT_03 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 3 of RAM. |
| INIT_04 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 4 of RAM. |
| INIT_05 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 5 of RAM. |
| INIT_06 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 6 of RAM. |
| INIT_07 | Hexadecimal | Any 32-Bit Value | All zeros | INIT for bit 7 of RAM. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM64M

Primitive: 64-Deep by 4-bit Wide Multi Port Random Access Memory (Select RAM)



Introduction

This design element is a 64-bit deep by 4-bit wide, multi-port, random access memory with synchronous write and asynchronous independent bit wide read capability. This RAM is implemented using the LUT resources of the device (also known as SelectRAM™) and does not consume any of the block RAM resources of the device. The RAM64M component is implemented in a single slice, and consists of one 4-bit write, 1-bit read port, and three separate 1-bit read ports from the same memory allowing for 4-bit write and independent bit read access RAM. If the DIA, DIB, DIC and DID inputs are all tied to the same data inputs, the RAM can become a 1 read/write port, 3 independent read port 64x1 quad port memory. If DID is grounded, DOD is not used. While ADDRA, ADDR B and ADDR C are tied to the same address the RAM becomes a 64x3 simple dual port RAM. If ADDR D is tied to ADDRA, ADDR B, and ADDR C; then the RAM is a 64x4 single port RAM. There are several other possible configurations for this RAM.

Port Descriptions

| Port | Direction | Width | Function |
|-------|-----------|-------|--|
| DOA | Output | 1 | Read port data outputs addressed by ADDRA |
| DOB | Output | 1 | Read port data outputs addressed by ADDRb |
| DOC | Output | 1 | Read port data outputs addressed by ADDRc |
| DOD | Output | 1 | Read/Write port data outputs addressed by ADDRd |
| DIA | Input | 1 | Write data inputs addressed by ADDRd (read output is addressed by ADDRA) |
| DIB | Input | 1 | Write data inputs addressed by ADDRd (read output is addressed by ADDRb) |
| DIC | Input | 1 | Write data inputs addressed by ADDRd (read output is addressed by ADDRc) |
| DID | Input | 1 | Write data inputs addressed by ADDRd |
| ADDRA | Input | 6 | Read address bus A |
| ADDRb | Input | 6 | Read address bus B |
| ADDRc | Input | 6 | Read address bus C |
| ADDRd | Input | 6 | 4-bit data write port, 1-bit data read port address bus D |
| WE | Input | 1 | Write Enable |
| WCLK | Input | 1 | Write clock (reads are asynchronous) |

Design Entry Method

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

This element can be inferred by some synthesis tools by describing a RAM with a synchronous write and asynchronous read capability. Consult your synthesis tool documentation for details on RAM inference capabilities and coding examples. Xilinx suggests that you instantiate RAM64Ms if you have a need to implicitly specify the RAM function, or if you need to manually place or relationally place the component. If a synchronous read capability is desired, the RAM64M outputs can be connected to an FDRSE (FDCPE is asynchronous set/reset is necessary) in order to improve the output timing of the function. However, this is not necessary for the proper operation of the RAM. If you want to have the data clocked on the negative edge of a clock, an inverter can be described on the clock input to this component. This inverter will be absorbed into the block giving the ability to write to the RAM on falling clock edges.

If instantiated, the following connections should be made to this component. Tie the WCLK input to the desired clock source, the DIA, DIB, DIC and DID inputs to the data source to be stored and the DOA, DOB, DOC and DOD outputs to an FDCE D input or other appropriate data destination or left unconnected if not used. The WE clock enable pin should be connected to the proper write enable source in the design. The 5-bit ADDRd bus should be connected to the source for the read/write addressing and the 5-bit ADDRA, ADDRb and ADDRc buses should be connected to the appropriate read address connections. The optional INIT_A, INIT_B, INIT_C and INIT_D attributes consisting of a 64-bit hexadecimal values that specifies each port's initial memory contents can be specified. The INIT value correlates to the RAM addressing by the following equation: $ADDRy[z] = INIT_y[z]$.

For instance, if the RAM ADDR_C port is addressed to 00001, then the INIT_C[1] values would be the initial values shown on the DOC port before the first write occurs at that address. If left unspecified, the initial contents will default to all zeros.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|----------|--|
| INIT_A | Hexadecimal | Any 64-Bit Value | All zero | Specifies the initial contents of the RAM on the A port. |
| INIT_B | Hexadecimal | Any 64-Bit Value | All zero | Specifies the initial contents of the RAM on the B port. |
| INIT_C | Hexadecimal | Any 64-Bit Value | All zero | Specifies the initial contents of the RAM on the C port. |
| INIT_D | Hexadecimal | Any 64-Bit Value | All zero | Specifies the initial contents of the RAM on the D port. |

VHDL Instantiation Template

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

```

Library UNISIM;
use UNISIM.vcomponents.all;

-- RAM64M: 64-deep by 4-wide Multi Port LUT RAM
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM64M_inst : RAM64M
generic map (
  INIT_A => X"0000000000000000",  -- Initial contents of A port
  INIT_B => X"0000000000000000",  -- Initial contents of B port
  INIT_C => X"0000000000000000",  -- Initial contents of C port
  INIT_D => X"0000000000000000")  -- Initial contents of D port
port map (
  DOA => DOA, -- Read port A 1-bit output
  DOB => DOB, -- Read port B 1-bit output
  DOC => DOC, -- Read port C 1-bit output
  DOD => DOD, -- Read/Write port D 1-bit output
  ADDRA => ADDRA,  -- Read port A 6-bit address input
  ADDR_B => ADDR_B,  -- Read port B 6-bit address input
  ADDR_C => ADDR_C,  -- Read port C 6-bit address input
  ADDR_D => ADDR_D,  -- Read/Write port D 6-bit address input
  DIA => DIA, -- RAM 1-bit data write input addressed by ADDR_D,
               -- read addressed by ADDRA
  DIB => DIB, -- RAM 1-bit data write input addressed by ADDR_D,
               -- read addressed by ADDR_B
  DIC => DIC, -- RAM 1-bit data write input addressed by ADDR_D,
               -- read addressed by ADDR_C
  DID => DID, -- RAM 1-bit data write input addressed by ADDR_D,
               -- read addressed by ADDR_D
  WCLK => WCLK, -- Write clock input
  WE => WE      -- Write enable input
);
-- End of RAM64M_inst instantiation

```

Verilog Instantiation Template

```
// RAM64M: 64-deep by 4-wide Multi Port LUT RAM
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAM64M #(
    .INIT_A(64'h0000000000000000), // Initial contents of A Port
    .INIT_B(64'h0000000000000000), // Initial contents of B Port
    .INIT_C(64'h0000000000000000), // Initial contents of C Port
    .INIT_D(64'h0000000000000000) // Initial contents of D Port
) RAM64M_inst (
    .DOA(DOA), // Read port A 1-bit output
    .DOB(DOB), // Read port B 1-bit output
    .DOC(DOC), // Read port C 1-bit output
    .DOD(DOD), // Readw/rite port D 1-bit output
    .DIA(DIA), // RAM 1-bit data write input addressed by ADDRd,
                // read addressed by ADDRA
    .DIB(DIB), // RAM 1-bit data write input addressed by ADDRd,
                // read addressed by ADDRb
    .DIC(DIC), // RAM 1-bit data write input addressed by ADDRd,
                // read addressed by ADDRc
    .DID(DID), // RAM 1-bit data write input addressed by ADDRd,
                // read addressed by ADDRd
    .ADDRA(ADDRA), // Read port A 6-bit address input
    .ADDRB(ADDRB), // Read port B 6-bit address input
    .ADDRC(ADDRC), // Read port C 6-bit address input
    .ADDRD(ADDRD), // Readw/rite port D 6-bit address input
    .WE(WE), // Write enable input
    .WCLK(WCLK) // Write clock input
);

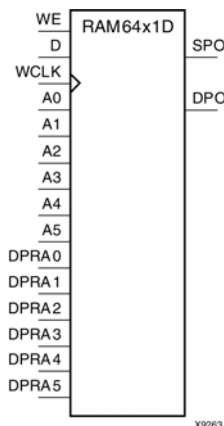
// End of RAM64M_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM64X1D

Primitive: 64-Deep by 1-Wide Dual Port Static Synchronous RAM



Introduction

This design element is a 64-word by 1-bit static dual port random access memory with synchronous write capability. The device has two separate address ports: the read address (DPRA5:DPRA0) and the write address (A5:A0). These two address ports are completely 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 6-bit (A0:A5) 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 A5:A0. The DPO output reflects the data in the memory cell addressed by DPRA5:DPRA0.

Note The write process is not affected by the address on the read address port.

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 A5:A0 | | | | |
| data_d = word addressed by bits DPRA5: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 64-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;

-- RAM64X1D: 64 x 1 negative edge write, asynchronous read
--           dual-port distributed RAM
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM64X1D_1_inst : RAM64X1D_1
generic map (
  INIT => X"0000000000000000") -- Initial contents of RAM
port map (
  DPO => DPO,      -- Read-only 1-bit data output
  SPO => SPO,      -- R/W 1-bit data output
  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
  A4 => A4,        -- R/W address[4] input bit
  A5 => A5,        -- R/W address[5] 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
  DPRA4 => DPRA4,  -- Read-only address[4] input bit
  DPRA5 => DPRA5,  -- Read-only address[5] input bit
  WCLK => WCLK,    -- Write clock input
  WE => WE         -- Write enable input
);

-- End of RAM64X1D_1_inst instantiation
```

Verilog Instantiation Template

```
// RAM64X1D: 64 x 1 positive edge write, asynchronous read dual-port distributed RAM
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAM64X1D #(
    .INIT(64'h0000000000000000) // Initial contents of RAM
) RAM64X1D_inst (
    .DPO(DPO),           // Read-only 1-bit data output
    .SPO(SPO),           // Rw/ 1-bit data output
    .A0(A0),             // Rw/ address[0] input bit
    .A1(A1),             // Rw/ address[1] input bit
    .A2(A2),             // Rw/ address[2] input bit
    .A3(A3),             // Rw/ address[3] input bit
    .A4(A4),             // Rw/ address[4] input bit
    .A5(A5),             // Rw/ address[5] 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
    .DPRA4(DPRA4),       // Read-only address[4] input bit
    .DPRA5(DPRA5),       // Read-only address[5] input bit
    .WCLK(WCLK),         // Write clock input
    .WE(WE)              // Write enable input
);

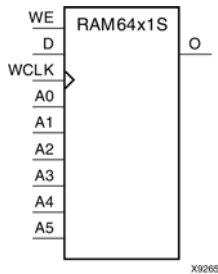
// End of RAM64X1D_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

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
//           All FPGA
// Xilinx HDL Libraries Guide, version 12.3

RAM64X1S #(
  .INIT(64'h0000000000000000) // Initial contents of RAM
) RAM64X1S_inst (
  .O(O),           // 1-bit data output
  .A0(A0),         // Address[0] input bit
  .A1(A1),         // Address[1] input bit
  .A2(A2),         // Address[2] input bit
  .A3(A3),         // Address[3] input bit
  .A4(A4),         // Address[4] input bit
  .A5(A5),         // Address[5] input bit
  .D(D),           // 1-bit data input
  .WCLK(WCLK),     // Write clock input
  .WE(WE)          // Write enable input
);

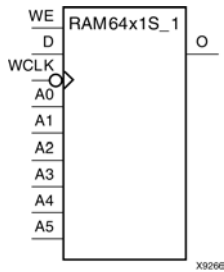
// End of RAM64X1S_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM64X1S_1

Primitive: 64-Deep by 1-Wide Static Synchronous RAM with Negative-Edge Clock



Introduction

This design element is a 64-word by 1-bit static random access memory with synchronous write capability. When the write enable is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When (WE) is High, any negative transition on (WCLK) loads the data on the data input (D) into the word selected by the 6-bit address (A5:A0). For predictable performance, address and data inputs must be stable before a High-to-Low (WCLK) transition. This RAM block assumes an active-Low (WCLK). However, (WCLK) can be active-High or active-Low. Any inverter placed on the (WCLK) input net is absorbed into the block.

The signal output on the data output pin (O) is the data that is stored in the RAM at the location defined by the values on the address pins.

You can initialize this element during configuration using the INIT attribute.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|---|---------|
| WE (mode) | WCLK | D | O |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↓ | D | D |
| 1 (read) | ↑ | X | Data |
| Data = word addressed by bits A5:A0 | | | |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|--|
| INIT | Hexadecimal | Any 64-Bit Value | All zeros | Initializes ROMs, RAMs, registers, and look-up tables. |

VHDL Instantiation Template

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

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

-- RAM64X1S_1: 64 x 1 negative edge write, asynchronous read single-port distributed RAM
--           Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAM64X1S_1_inst : RAM64X1S_1
generic map (
    INIT => X"0000000000000000")
port map (
    O => O,           -- 1-bit data output
    A0 => A0,          -- Address[0] input bit
    A1 => A1,          -- Address[1] input bit
    A2 => A2,          -- Address[2] input bit
    A3 => A3,          -- Address[3] input bit
    A4 => A4,          -- Address[4] input bit
    A5 => A5,          -- Address[5] input bit
    D => D,           -- 1-bit data input
    WCLK => WCLK,      -- Write clock input
    WE => WE           -- Write enable input
);

-- End of RAM64X1S_1_inst instantiation
```

Verilog Instantiation Template

```
// RAM64X1S_1: 64 x 1 negative edge write, asynchronous read single-port distributed RAM
//           All FPGA
// Xilinx HDL Libraries Guide, version 12.3

RAM64X1S_1 #(
    .INIT(64'h0000000000000000) // Initial contents of RAM
) RAM64X1S_1_inst (
    .O(O),           // 1-bit data output
    .A0(A0),         // Address[0] input bit
    .A1(A1),         // Address[1] input bit
    .A2(A2),         // Address[2] input bit
    .A3(A3),         // Address[3] input bit
    .A4(A4),         // Address[4] input bit
    .A5(A5),         // Address[5] input bit
    .D(D),           // 1-bit data input
    .WCLK(WCLK),     // Write clock input
    .WE(WE)          // Write enable input
);

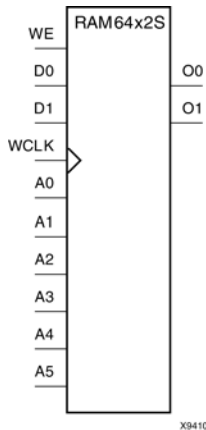
// End of RAM64X1S_1_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAM64X2S

Primitive: 64-Deep by 2-Wide Static Synchronous RAM



Introduction

This design element is a 64-word by 2-bit static random access memory with synchronous write capability. When the write enable (WE) is Low, transitions on the write clock (WCLK) are ignored and data stored in the RAM is not affected. When WE is High, any positive transition on WCLK loads the data on the data input (D1:D0) into the word selected by the 6-bit address (A5:A0). For predictable performance, address and data inputs must be stable before a Low-to-High WCLK transition. This RAM block assumes an active-High WCLK. However, WCLK can be active-High or active-Low. Any inverter placed on the WCLK input net is absorbed into the block.

The signal output on the data output pins (O1:O0) is the data that is stored in the RAM at the location defined by the values on the address pins. You can use the INIT_00 and INIT_01 properties to specify the initial contents of this design element.

Logic Table

| Inputs | | | Outputs |
|-------------------------------------|------|-------|---------|
| WE (mode) | WCLK | D0:D1 | O0:O1 |
| 0 (read) | X | X | Data |
| 1 (read) | 0 | X | Data |
| 1 (read) | 1 | X | Data |
| 1 (write) | ↑ | D1:D0 | D1:D0 |
| 1 (read) | ↓ | X | Data |
| Data = word addressed by bits A5:A0 | | | |

Design Entry Method

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

Available Attributes

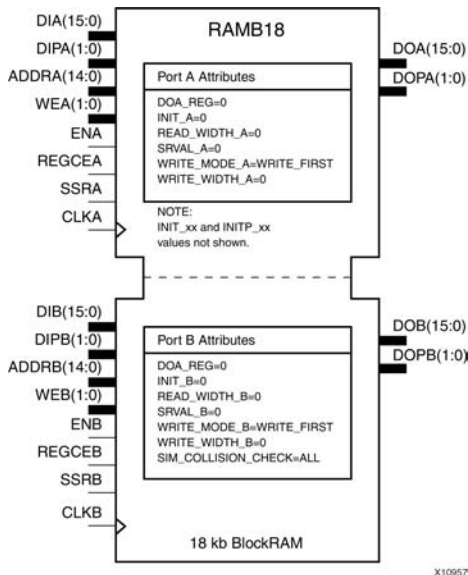
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|--|
| INIT_00 | Hexadecimal | Any 64-Bit Value | All zeros | Initializes RAMs, registers, and look-up tables. |
| INIT_01 | Hexadecimal | Any 64-Bit Value | All zeros | Initializes RAMs, registers, and look-up tables. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAMB18

Primitive: 18K-bit Configurable Synchronous True Dual Port Block RAM



Introduction

Virtex®-5 and above devices contain several block RAM memories which can be configured as FIFOs, automatic error correction RAM, or general-purpose 36kb or 18kb RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. The RAMB18 allows access to the block RAM in the 18kb configuration. This element can be cascaded to create a larger ram. This element can be configured and used as a 1-bit wide by 16K deep to an 18-bit wide by 1024-bit deep true dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, the READ and WRITE ports can operate fully independent and asynchronous to each other accessing the same memory array. When configured in the wider data width modes, byte-enable write operations are possible, and an optional output register can be used to reduce the clock-to-out times of the RAM.

Port Descriptions

| Port | Direction | Width | Function |
|----------------|-----------|----------------|---|
| DOA, DOB | Output | 1, 2, 4, 8, 16 | Port A/B data output bus. |
| DOPA, DOPB | Output | 0, 1, 2 | Port A/B parity data output bus. |
| DIA, DIB | Input | 1, 2, 4, 8, 16 | Port A/B data input bus. |
| DIPA, DIPB | Input | 0, 1, 2 | Port A/B parity data input bus. |
| ADDRA, ADDRb | Input | 14 | Port A/B address input bus. |
| WEA | Input | 2 | Port A byte-wide write enable. |
| WEB | Input | 2 | Port B byte-wide write enable. |
| ENA, ENB | Input | 1 | Port A/B enable |
| SSRA, SSRB | Input | 1 | Port A/B output registers synchronous set/reset. Active high will synchronous preset/reset to the associated port to the value specified for SRVAL_A/SRVAL_B. |
| REGCEA, REGCEB | Input | 1 | Port A/B output register clock enable input |
| CLKA, CLKB | Input | 1 | Port A/B clock input. |

Design Entry Method

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

The following table shows the necessary data, address and write enable connections for the variable width ports for each DATA_WIDTH value for either Port A or Port B. If a different width is used for the read and write on the same port, use the deeper of the two in order to select address connections.

All data and address ports not necessary for a particular configuration should either be left unconnected or grounded with the following exceptions.

| DATA_WIDTH Value | DI, DIP Connections | ADDR Connections | WE Connections | DO, DOP Connections |
|------------------|---------------------|------------------|--|---------------------|
| 1 | DI[0] | ADDR[14:0] | Connect WE[1:0] to single user WE signal | DO[0] |
| 2 | DI[1:0] | ADDR[14:1] | Connect WE[1:0] to single user WE signal | DO[1:0] |
| 4 | DI[3:0] | ADDR[14:2] | Connect WE[1:0] to single user WE signal | DO[3:0] |
| 9 | DI[7:0], DIP[0] | ADDR[14:3] | Connect WE[1:0] to single user WE signal | DO[7:0], DOP[0] |
| 18 | DI[15:0], DIP[1:0] | ADDR[14:4] | Connect WE[0] and WE[1] to user WE signal. | DO[15:0], DOP[1:0] |

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|------------------|-------------|----------------------|-----------|---|
| DOA_REG, DOB_REG | Integer | 0, 1 | 0 | A value of 1 enables the output registers the RAM enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle but will result in slower clock-to-out timing. |
| INIT_A | Hexadecimal | Any 18-Bit Value | All zeros | Specifies the initial value on the Port A output after configuration. |
| INIT_B | Hexadecimal | Any 18-Bit Value | All zeros | Specifies the initial value on the Port B output after configuration. |
| READ_WIDTH_A | Integer | 0, 1, 2, 4, 9, or 18 | 0 | Specifies the desired data width for a read on Port A including parity bits. The 0 signifies that the port is not used. |
| READ_WIDTH_B | Integer | 0, 1, 2, 4, 9, or 18 | 0 | Specifies the desired data width for a read on Port B including parity bits. The 0 signifies that the port is not used. |

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------------|-------------|---|-------------|---|
| 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> |
| SIM_MODE | String | "SAFE" or "FAST" . | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| SRVAL_A | Hexadecimal | Any 18-Bit Value | All zeros | Specifies the output value of Port A upon the assertion of the synchronous reset (SSRA) signal. |
| SRVAL_B | Hexadecimal | Any 18-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. |
| WRITE_WIDTH_A | Integer | 0,1, 2, 4, 9, or 18 | 0 | Specifies the desired data width for a write to Port A including parity bits. The 0 signifies that the port is not used. |
| WRITE_WIDTH_B | Integer | 0,1, 2, 4, 9, or 18 | 0 | Specifies the desired data width for a write to Port B including parity bits. The 0 signifies that the port is not used. |
| 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 zeroes | Allows specification of the initial contents of the 2kb parity data memory array. |

Mapping of INIT_A, INIT_B, SRVAL_A, SRVAL_B

The INIT_A, INIT_B, SRVAL_A and SRVAL_B attributes are all 18-bit attributes. However, if the READ_WIDTH is set to a value less than 18 for the particular port only a subset of the bits are used.

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;

-- RAMB18: 16k+2k Parity Paramatizable True Dual-Port BlockRAM
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAMB18_inst : RAMB18
generic map (
    DOA_REG => 0, -- Optional output register on A port (0 or 1)
    DOB_REG => 0, -- Optional output register on B port (0 or 1)
    INIT_A => X"00000", -- Initial values on A output port
    INIT_B => X"00000", -- Initial values on B output port
    READ_WIDTH_A => 0, -- Valid values are 1, 2, 4, 9, or 18
    READ_WIDTH_B => 0, -- Valid values are 1, 2, 4, 9, or 18
    SIM_COLLISION_CHECK => "ALL", -- Collision check enable "ALL", "WARNING_ONLY",
    -- "GENERATE_X_ONLY" or "NONE"
    SIM_MODE => "SAFE", -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
    -- Design Guide" for details
    SRVAL_A => X"00000", -- Set/Reset value for A port output
    SRVAL_B => X"00000", -- Set/Reset value for B port output
    WRITE_MODE_A => "WRITE_FIRST", -- "WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"
    WRITE_MODE_B => "WRITE_FIRST", -- "WRITE_FIRST", "READ_FIRST" or "NO_CHANGE"
    WRITE_WIDTH_A => 0, -- Valid values are 1, 2, 4, 9, or 18
    WRITE_WIDTH_B => 0, -- Valid values are 1, 2, 4, 9, or 18
    -- The following INIT_xx declarations specify the initial contents of the RAM
    INIT_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_07 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_0F => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_10 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_11 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_12 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_13 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_14 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_15 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_16 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_17 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_18 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_19 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1A => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1B => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1C => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1D => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1E => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_1F => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_20 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_21 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_22 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_23 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_24 => X"0000000000000000000000000000000000000000000000000000000000000000",
    INIT_25 => X"0000000000000000000000000000000000000000000000000000000000000000",
```

```

INIT_26 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_27 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_28 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_29 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2F => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_30 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_31 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_32 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_33 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_34 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_35 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_36 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_37 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_38 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_39 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3F => X"0000000000000000000000000000000000000000000000000000000000000000",
-- The next set of INITP_xx are for the parity bits
INITP_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_07 => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
  DOA => DOA, -- 16-bit A port data output
  DOB => DOB, -- 16-bit B port data output
  DOPA => DOPA, -- 2-bit A port parity data output
  DOPB => DOPB, -- 2-bit B port parity data output
  ADDR_A => ADDR_A, -- 14-bit A port address input
  ADDR_B => ADDR_B, -- 14-bit B port address input
  CLKA => CLKA, -- 1-bit A port clock input
  CLKB => CLKB, -- 1-bit B port clock input
  DIA => DIA, -- 16-bit A port data input
  DIB => DIB, -- 16-bit B port data input
  DIP_A => DIP_A, -- 2-bit A port parity data input
  DIP_B => DIP_B, -- 2-bit B port parity data input
  ENA => ENA, -- 1-bit A port enable input
  ENB => ENB, -- 1-bit B port enable input
  REGCEA => REGCEA, -- 1-bit A port register enable input
  REGCEB => REGCEB, -- 1-bit B port register enable input
  SSRA => SSRA, -- 1-bit A port set/reset input
  SSRB => SSRB, -- 1-bit B port set/reset input
  WEA => WEA, -- 2-bit A port write enable input
  WEB => WEB, -- 2-bit B port write enable input
);

-- End of RAMB18_inst instantiation

```

Verilog Instantiation Template

```

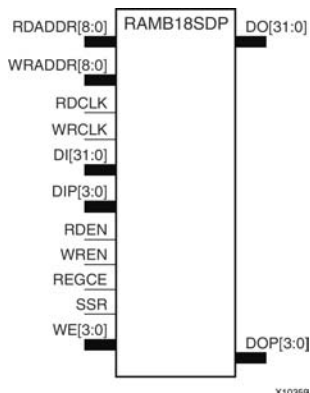
// RAMB18: 16k+2k Parity Paramatizable True Dual-Port BlockRAM
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAMB18 #(
  .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
  .DOA_REG(0), // Optional output registers on A port (0 or 1)
  .DOB_REG(0), // Optional output registers on B port (0 or 1)
  .INIT_A(18'h00000), // Initial values on A output port

```


RAMB18SDP

Primitive: 36-bit by 512 Deep, 18kb Synchronous Simple Dual Port Block RAM



Introduction

This design element is one of several block RAM memories that can be configured as FIFOs, automatic error correction RAM, or general-purpose, 36kb or 18kb RAM/ROM memories. These Block RAM memories offer fast and flexible storage of large amounts of on-chip data. The RAMB18SDP gives you access to the block RAM in the 18kb configuration. This component is set to a 36-bit wide by 512 deep simple dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, the READ and WRITE ports can operate fully independently and asynchronously to each other, accessing the same memory array. Byte-enable write operations are possible, and an optional output register can be used to reduce the clock-to-out times of the RAM.

Note This element, must be configured so that read and write ports have the same width.

Port Descriptions

| Port | Direction | Width | Function |
|----------------|-----------|-------|--|
| DO | Output | 32 | Data output bus addressed by RDADDR. |
| DOP | Output | 4 | Data parity output bus addressed by RDADDR. |
| DI | Input | 32 | Data input bus addressed by WRADDR. |
| DIP | Input | 4 | Data parity input bus addressed by WRADDR. |
| WRDDRA, RDDDRB | Input | 9 | Write/Read address input buses. |
| WE | Input | 4 | Write enable. |
| WREN, RDEN | Input | 1 | Write/Read enable |
| SSR | Input | 1 | Output registers synchronous reset. |
| REGCE | Input | 1 | Output register clock enable input (valid only when DO_REG=1). |
| WRCLK, RDCLK | Input | 1 | Write/Read clock input. |

Design Entry Method

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

Available Attributes

| Attribute(s) | Type | Allowed Values | Default | Description |
|----------------------|-------------|--|-----------|---|
| DO_REG | Integer | 0, 1 | 0 | A value of 1 enables the output registers to the RAM, enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle, but will have slower clock-to-out timing. |
| INIT | Hexadecimal | Any 36-Bit Value | All zeros | Specifies the initial value on the 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> |
| SIM_MODE | String | "SAFE" or "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| SRVAL | Hexadecimal | Any 36-Bit Value | All zeros | Specifies the output value of on the DO port upon the assertion of the synchronous reset (SSR) signal. |
| 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. |


```

INIT_33 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_34 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_35 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_36 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_37 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_38 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_39 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3F => X"0000000000000000000000000000000000000000000000000000000000000000",
-- The next set of INITP_xx are for the parity bits
INITP_00 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_01 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_02 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_03 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_04 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_05 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_06 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_07 => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
  DO => DO,          -- 32-bit Data Output
  DOP => DOP,         -- 4-bit Parity Output
  RDCLK => RDCLK,     -- 1-bit read port clock
  RDEN => RDEN,       -- 1-bit read port enable
  REGCE => REGCE,     -- 1-bit register enable input
  SSR => SSR,         -- 1-bit synchronous output set/reset input
  WRCLK => WRCLK,     -- 1-bit write port clock
  WREN => WREN,       -- 1-bit write port enable
  WRADDR => WRADDR,   -- 9-bit write port address input
  RDADDR => RDADDR,   -- 9-bit read port address input
  DI => DI,           -- 32-bit data input
  DIP => DIP,         -- 4-bit parity data input
  WE => WE            -- 4-bit write enable input
);

-- End of RAMB18SDP_inst instantiation

```

Verilog Instantiation Template

```

// RAMB18SDP: 36x512 Simple Dual-Port BlockRAM
// Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAMB18SDP #(
  .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
  .DO_REG(0), // Optional output register (0 or 1)
  .INIT(36'h00000000), // Initial values on output port
  .SIM_COLLISION_CHECK("ALL"), // Collision check enable "ALL", "WARNING_ONLY",
  // "GENERATE_X_ONLY" or "NONE"
  .SRVAL(36'h00000000), // Set/Reset value for port output

  // The following INIT_xx declarations specify the initial contents of the RAM
  .INIT_00(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_01(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_02(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_03(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_04(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_05(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_06(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_07(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_08(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_09(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_0A(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_0B(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_0C(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_0D(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_0E(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),
  .INIT_0F(256'h00000000_00000000_00000000_00000000_00000000_00000000_00000000_00000000),

```

Virtex-5 Libraries Guide for HDL Designs
UG621 (v 12.3) September 21, 2010

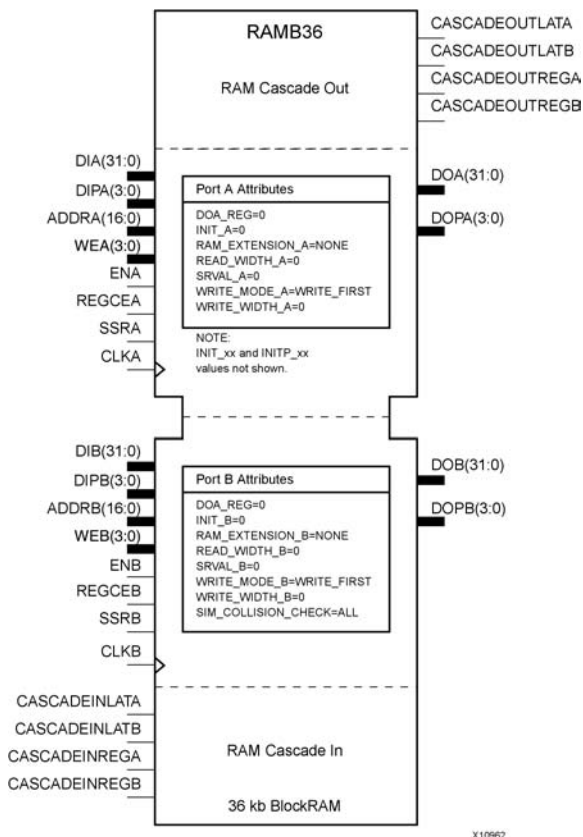
```
// End of RAMB18SDP_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAMB36

Primitive: 36kb Configurable Synchronous True Dual Port Block RAM



Introduction

This design element is one of several block RAM memories that can be configured as FIFOs, automatic error correction RAM, or general-purpose, 36kb or 18kb RAM/ROM memories. These block RAM memories offer fast and flexible storage of large amounts of on-chip data. This element allows access to the block RAM in the 36kb configuration. This component can be configured and used as a 1-bit wide by 32K deep to a 36-bit wide by 1K deep true 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 configured in the wider data width modes, byte-enable write operations are possible and an option output register can be used to reduce the clock-to-out times of the RAM at the expense of an extra clock cycle of latency.

This design element should be used for Simple Dual Port 72-bit wide, 512 deep, optional ECC scrub functionality. This element can be created using cascaded RAMB18s.

The following possible combination of elements can be placed in RAMB36:

- RAMB18/RAMB18
- RAMB18/FIFO18
- RAMB18SDP/RAMB18SDP
- RAMB18SDP/FIFO18_36

Port Descriptions

| Port | Direction | Width | Function |
|-----------------------------------|-----------|--------------------|--|
| DOA | Output | 1, 2, 4, 8, 16, 32 | Port A data output bus. |
| DOB | Output | 1, 2, 4, 8, 16, 32 | Port B data output bus. |
| DOPA, DOPB | Output | 0, 1, 2, 4 | Port A/B parity data output bus. |
| CASCADEOUTLATA, CASCADEOUTLATB | Output | 1 | Outputs for ports A and B used to cascade two BlockRAMs to create a 64K deep by 1 wide memory (connects to the lower CASCADEINLATA/B of another RAMB36, leave unconnected if not building a 64Kx1 RAM or if RAM_EXTENSION_A/B are not set to "LOWER"). |
| CASCADEOUTREGA, CASCADEOUTREGB | Output | 1 | Outputs for ports A and B used to cascade two BlockRAMs to create a 64K deep by 1 wide memory (connects to the lower CASCADEINREGA/B of another RAMB36, leave unconnected if not building a 64Kx1 RAM or if RAM_EXTENSION_A/B are not set to "LOWER"). |
| CASCADEINLATA, CASCADEINLATB | Input | 1 | Inputs for ports A and B used to cascade two BlockRAMs to create a 64K deep by 1 wide memory (connects to the upper CASCADEOUTLATA/B of another RAMB36, leave unconnected if not building a 64Kx1 RAM or if RAM_EXTENSION_A/B are not set to "UPPER"). |
| CASCADEINREGA, CASCADEINREGB | Input | 1 | Inputs for ports A and B used to cascade two BlockRAMs to create a 64K deep by 1 wide memory (connects to the upper CASCADEOUTREGA/B of another RAMB36, leave unconnected if not building a 64Kx1 RAM or if RAM_EXTENSION_A/B are not set to "UPPER"). |
| DIA | Input | 1, 2, 4, 8, 16, 32 | Port A data input bus. |
| DIB | Input | 1, 2, 4, 8, 16, 32 | Port B data input bus. |
| DIPA, DIPB | Input | 0, 1, 2, 4 | Port A/B parity data input bus. |
| ADDRA, ADDRb | Input | 16 | Port A/B address input bus; 16 for CASC mode. |
| WEA | Input | 4 | Port A byte-wide write enable |
| WEB | Input | 4 | Port B byte-wide write enable. |
| ENA, ENB | Input | 1 | Port A/B enable. Active high is enabled, while a low value will disable reads or writes to the associated port. |
| SSRA, SSRB | Input | 1 | Port A/B output registers synchronous set/reset. Active high will synchronous preset/reset to the associated port to the value specified for SRVAL_A/SRVAL_B. |
| REGCEA, REGCEB | Input | 1 | Port A/B output register clock enable input. Active high will clock enable the output registers to the associated port. |

Design Entry Method

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

The following table shows the necessary data, address and write enable connections for the variable width ports for each DATA_WIDTH value for either Port A or Port B. If a different width is used for the read and write on the same port, use the deeper of the two in order to select address connections.

All data and address ports not necessary for a particular configuration should either be left unconnected or grounded with the following exceptions.

- The address bit 15 is only used in cascadable block RAM. For noncascading block RAM, connect High.
- ADDR pins must be 16-bits wide. However, valid addresses for non-cascadable block RAM are only found on pin 14 to (15 - address width). The remaining pins, including pin 15, should be tied High.

| DATA_WIDTH Value | DI, DIP Connections | ADDR Connections | WE Connections | DO, DOP Connections |
|---------------------|---------------------|------------------|---|---------------------|
| 1 (with cascade) | DI[0] | ADDR[15:0] | Connect WE[3:0] to single user WE signal | DO[0] |
| 1 (without cascade) | DI[0] | ADDR[14:0] | Connect WE[3:0] to single user WE signal | DO[0] |
| 2 | DI[1:0] | ADDR[14:1] | Connect WE[3:0] to single user WE signal | DO[1:0] |
| 4 | DI[3:0] | ADDR[14:2] | Connect WE[3:0] to single user WE signal | DO[3:0] |
| 9 | DI[7:0], DIP[0] | ADDR[14:3] | Connect WE[3:0] to single user WE signal | DO[7:0], DOP[0] |
| 18 | DI[15:0], DIP[1:0] | ADDR[14:4] | 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[14:5] | Connect each WE[3:0] signal to the associated byte write enable/ | DO[31:0], DOP[3:0] |

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|------------------|-------------|------------------|-----------|--|
| DOA_REG, DOB_REG | Integer | 0, 1 | 0 | A value of 1 enables to the output registers to the RAM enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle, but will result in slower clock to out timing. |
| INIT_A | Hexadecimal | Any 36-Bit Value | All zeros | Specifies the initial value on the output of Port A of the RAMB36 after configuration. |
| INIT_B | Hexadecimal | Any 36-Bit Value | All zeros | Specifies the initial value on the output of Port B of the RAMB36 after configuration. |

| Attribute | Type | Allowed Values | Default | Description |
|---------------------|-------------|---|-----------|---|
| READ_WIDTH_A | Integer | 0, 1, 4, 9, 18 or 36 | 0 | Specifies the desired data width for a read on Port A, including parity bits. This value must be 0 if the Port B is not used. Otherwise, it should be set to the desired port width. |
| READ_WIDTH_B | Integer | 0, 1, 4, 9, 18 or 36 | 0 | Specifies the desired data width for a read on Port B including parity bits. This value must be 0 if the Port B is not used. Otherwise, it should be set to the desired port width. |
| 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> |
| SIM_MODE | String | "SAFE" or "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." Please see the <i>Synthesis and Simulation Design Guide</i> for more information. |
| SRVAL_A | Hexadecimal | Any 36-Bit Value | All zeros | Specifies the output value of the RAM upon the assertion of the Port A synchronous reset (SSRA) signal. |
| SRVAL_B | Hexadecimal | Any 36-Bit Value | All zeros | Specifies the output value of the RAM upon the assertion of the Port B synchronous reset (SSRB) signal. |

| Attribute | Type | Allowed Values | Default | Description |
|-------------------------------------|-------------|--|---------------|---|
| WRITE_MODE_A, WRITE_MODE_B | String | "WRITE_FIRST", "READ_FIRST" or "NO_CHANGE" | "WRITE_FIRST" | Specifies output behavior of the port being written to: <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_WIDTH_A | Integer | 0, 1, 2, 4, 9, 18 or 36 | 0 | Specifies the desired data width for a write to Port B including parity bits. This value must be 0 if the port is not used. Otherwise should be set to the desired write width. |
| WRITE_WIDTH_B | Integer | 0, 1, 2, 4, 9, 18 or 36 | 0 | Specifies the desired data width for a write to Port B including parity bits. This value must be 0 if the port is not used. Otherwise should be set to the desired write width. |
| RAM_EXTENTION_A, RAM_EXTENTION_B | String | "UPPER", "LOWER" or "NONE" | "NONE" | If not cascading two BlockRAMs to form a 72K x 1 RAM set to "NONE". If cascading RAMs, set to either "UPPER" or "LOWER" to indicate relative RAM location for proper configuration of the RAM. |
| INIT_00 to INIT_7F | Hexadecimal | Any 256-Bit Value | All zeros | Allows specification of the initial contents of the 72kb data memory array. |
| INITP_00 to INITP_0F | Hexadecimal | Any 256-Bit Value | All zeros | Allows specification of the initial contents of the 4kb parity data memory array. |

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;

-- RAMB36: 32k+4k Parity Paramatizable True Dual-Port BlockRAM
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

RAMB36_inst : RAMB36
generic map (
    DOA_REG => 0, -- Optional output register on A port (0 or 1)
    DOB_REG => 0, -- Optional output register on B port (0 or 1)
    INIT_A => X"000000000", -- Initial values on A output port
    INIT_B => X"000000000", -- Initial values on B output port
    RAM_EXTENSION_A => "NONE", -- "UPPER", "LOWER" or "NONE" when cascaded
    RAM_EXTENSION_B => "NONE", -- "UPPER", "LOWER" or "NONE" when cascaded
    READ_WIDTH_A => 0, -- Valid values are 1, 2, 4, 9, 18, or 36
    READ_WIDTH_B => 0, -- Valid values are 1, 2, 4, 9, 18, or 36
    SIM_COLLISION_CHECK => "ALL", -- Collision check enable "ALL", "WARNING_ONLY",
    -- "GENERATE_X_ONLY" or "NONE"
    SIM_MODE => "SAFE", -- Simulation: "SAFE" vs "FAST", see "Synthesis and Simulation
    -- Design Guide" for details

```


[illegible]

UG621 (v 12.3) September 21, 2010

```

INITP_OA => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_OB => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_OC => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_OD => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_OE => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_OF => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
  CASCADEOUTLATA => CASCADEOUTLATA, -- 1-bit cascade A latch output
  CASCADEOUTLATB => CASCADEOUTLATB, -- 1-bit cascade B latch output
  CASCADEOUTREGA => CASCADEOUTREGA, -- 1-bit cascade A register output
  CASCADEOUTREGB => CASCADEOUTREGB, -- 1-bit cascade B register output
  DOA => DOA, -- 32-bit A port data output
  DOB => DOB, -- 32-bit B port data output
  DOPA => DOPA, -- 4-bit A port parity data output
  DOPB => DOPB, -- 4-bit B port parity data output
  ADDRA => ADDRA, -- 16-bit A port address input
  ADDRb => ADDRb, -- 16-bit B port address input
  CASCADEINLATA => CASCADEINLATA, -- 1-bit cascade A latch input
  CASCADEINLATB => CASCADEINLATB, -- 1-bit cascade B latch input
  CASCADEINREGA => CASCADEINREGA, -- 1-bit cascade A register input
  CASCADEINREGB => CASCADEINREGB, -- 1-bit cascade B register input
  CLKA => CLKA, -- 1-bit A port clock input
  CLKB => CLKB, -- 1 bit B port clock input
  DIA => DIA, -- 32-bit A port data input
  DIB => DIB, -- 32-bit B port data input
  DIPA => DIPA, -- 4-bit A port parity data input
  DIPB => DIPB, -- 4-bit B port parity data input
  ENA => ENA, -- 1-bit A port enable input
  ENB => ENB, -- 1-bit B port enable input
  REGCEA => REGCEA, -- 1-bit A port register enable input
  REGCEB => REGCEB, -- 1-bit B port register enable input
  SSRA => SSRA, -- 1-bit A port set/reset input
  SSRB => SSRB, -- 1-bit B port set/reset input
  WEA => WEA, -- 4-bit A port write enable input
  WEB => WEB -- 4-bit B port write enable input
);

-- End of RAMB36_inst instantiation

```

Verilog Instantiation Template

```

// RAMB36: 32k+4k Parity Paramatizable True Dual-Port BlockRAM
//          Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAMB36 #(
  .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
  .DOA_REG(0), // Optional output registers on A port (0 or 1)
  .DOB_REG(0), // Optional output registers on B port (0 or 1)
  .INIT_A(36'h000000000), // Initial values on A output port
  .INIT_B(36'h000000000), // Initial values on B output port
  .RAM_EXTENSION_A("NONE"), // "UPPER", "LOWER" or "NONE" when cascaded
  .RAM_EXTENSION_B("NONE"), // "UPPER", "LOWER" or "NONE" when cascaded
  .READ_WIDTH_A(0), // Valid values are 1, 2, 4, 9, 18, or 36
  .READ_WIDTH_B(0), // Valid values are 1, 2, 4, 9, 18, or 36
  .SIM_COLLISION_CHECK("ALL"), // Collision check enable "ALL", "WARNING_ONLY",
                                // "GENERATE_X_ONLY" or "NONE"
  .SRVAL_A(36'h000000000), // Set/Reset value for A port output
  .SRVAL_B(36'h000000000), // Set/Reset value for B port output
  .WRITE_MODE_A("WRITE_FIRST"), // "WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"
  .WRITE_MODE_B("WRITE_FIRST"), // "WRITE_FIRST", "READ_FIRST", or "NO_CHANGE"
  .WRITE_WIDTH_A(0), // Valid values are 1, 2, 4, 9, 18, or 36
  .WRITE_WIDTH_B(0), // Valid values are 1, 2, 4, 9, 18, or 36

  // The following INIT_xx declarations specify the initial contents of the RAM
  .INIT_00(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_01(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_02(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_03(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_04(256'h0000000000000000000000000000000000000000000000000000000000000000),
  .INIT_05(256'h0000000000000000000000000000000000000000000000000000000000000000),

```

UG621 (v 12.3) September 21, 2010

[illegible]

```

.DOB(DOB),          // 32-bit B port data output
.DOPA(DOPA),        // 4-bit A port parity data output
.DOPB(DOPB),        // 4-bit B port parity data output
.ADDRA(ADDRA),      // 16-bit A port address input
.ADDRB(ADDRB),      // 16-bit B port address input
.CASCADEINLATA(CASCADEINLATA), // 1-bit cascade A latch input
.CASCADEINLATB(CASCADEINLATB), // 1-bit cascade B latch input
.CASCADEINREGA(CASCADEINREGA), // 1-bit cascade A register input
.CASCADEINREGB(CASCADEINREGB), // 1-bit cascade B register input
.CLKA(CLKA),        // 1-bit A port clock input
.CLKB(CLKB),        // 1-bit B port clock input
.DIA(DIA),          // 32-bit A port data input
.DIB(DIB),          // 32-bit B port data input
.DIPA(DIPA),        // 4-bit A port parity data input
.DIPB(DIPB),        // 4-bit B port parity data input
.ENA(ENA),          // 1-bit A port enable input
.ENB(ENB),          // 1-bit B port enable input
.REGCEA(REGCEA),    // 1-bit A port register enable input
.REGCEB(REGCEB),    // 1-bit B port register enable input
.SSRA(SSRA),        // 1-bit A port set/reset input
.SSRB(SSRB),        // 1-bit B port set/reset input
.WEA(WEA),          // 4-bit A port write enable input
.WEB(WEB),          // 4-bit B port write enable input
);

// End of RAMB36_inst instantiation

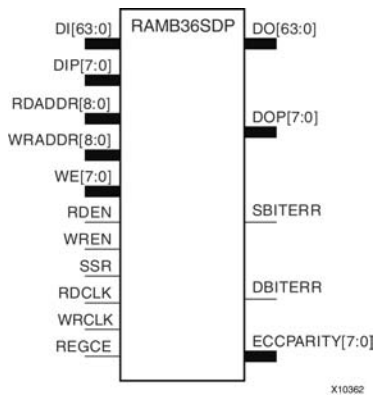
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

RAMB36SDP

Primitive: 72-bit by 512 Deep, 36kb Synchronous Simple Dual Port Block RAM with ECC (Error Correction Circuitry)



Introduction

This design element is one of several Block RAM memories that can be configured as FIFOs, automatic error correction RAM, or general-purpose, 36kb or 18kb RAM/ROM memories. These Block RAM memories offer fast and flexible storage of large amounts of on-chip data. The RAMB36SDP gives you access to the block RAM in the 36kb configuration. This component is set to a 72-bit wide by 512 deep simple dual port RAM. Both read and write operations are fully synchronous to the supplied clock(s) to the component. However, READ and WRITE ports can operate fully independently and asynchronously to each other accessing the same memory array. Byte-enable write operations are possible, and an optional output register can be used to reduce the clock-to-out times of the RAM. Error detection and correction circuitry can also be enabled to uncover and rectify possible memory corruptions.

Note This element, must be configured so that read and write ports have the same width.

Port Descriptions

| Port | Direction | Width | Function |
|----------------|-----------|-------|---|
| DO | Output | 64 | Data output bus addressed by RDADDR. |
| DOP | Output | 8 | Data parity output bus addressed by RDADDR. |
| SBITERR | Output | 1 | Status output from ECC function to indicate a single bit error was detected. EN_ECC_READ needs to be TRUE in order to use this functionality. |
| DBITERR | Output | 1 | Status output from ECC function to indicate a double bit error was detected. EN_ECC_READ needs to be TRUE in order to use this functionality. |
| ECCPARITY | Output | 8 | 8-bit data generated by the ECC encoder used by the ECC decoder for memory error detection and correction. |
| DI | Input | 64 | Data input bus addressed by WRADDR. |
| DIP | Input | 8 | Data parity input bus addressed by WRADDR. |
| WRADDR, RDADDR | Input | 9 | Write/Read address input buses. |
| WE | Input | 8 | Write enable |
| WREN, RDEN | Input | 1 | Write/Read enable |

| Port | Direction | Width | Function |
|--------------|-----------|-------|---|
| SSR | Input | 1 | Output registers synchronous reset. |
| REGCE | Input | 1 | Output register clock enable input (valid only when DO_REG=1) |
| WRCLK, RDCLK | Input | 1 | Write/Read clock input. |

Design Entry Method

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

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|---------------------|-------------|---|-----------|---|
| DO_REG | Integer | 0, 1 | 0 | A value of 1 enables to the output registers to the RAM enabling quicker clock-to-out from the RAM at the expense of an added clock cycle of read latency. A value of 0 allows a read in one clock cycle but will have slower clock to out timing. |
| INIT | Hexadecimal | Any 72-Bit Value | All zeros | Specifies the initial value on the output after configuration. |
| EN_ECC_READ | Boolean | TRUE or FALSE | FALSE | Enable the ECC decoder circuitry. |
| EN_ECC_WRITE | Boolean | TRUE or FALSE | FALSE | Enable the ECC encoder circuitry. |
| EN_ECC_SCRUB | Boolean | TRUE or FALSE | FALSE | Enable ECC scrubbing of RAM contents |
| 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> |
| SIM_MODE | String | "SAFE" or "FAST" | "SAFE" | This is a simulation only attribute. It will direct the simulation model to run in performance-oriented mode when set to "FAST." |

UG621 (v 12.3) September 21, 2010

```

INIT_68 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_69 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6F => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_70 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_71 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_72 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_73 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_74 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_75 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_76 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_77 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_78 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_79 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7F => 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",
INITP_08 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_09 => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0A => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0B => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0C => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0D => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0E => X"0000000000000000000000000000000000000000000000000000000000000000",
INITP_0F => X"0000000000000000000000000000000000000000000000000000000000000000")
port map (
  DBITERR => DBITERR, -- 1-bit double bit error status output
  SBITERR => SBITERR, -- 1-bit single bit error status output
  DO => DO,           -- 64-bit Data Output
  DOP => DOP,         -- 8-bit Parity Output
  ECCPARITY => ECCPARITY, -- 8-bit generated error correction parity
  RDCLK => RDCLK,     -- 1-bit read port clock
  RDEN => RDEN,       -- 1-bit read port enable
  REGCE => REGCE,     -- 1-bit register enable input
  SSR => SSR,         -- 1-bit synchronous output set/reset input
  WRCLK => WRCLK,     -- 1-bit write port clock
  WREN => WREN,       -- 1-bit write port enable
  WRADDR => WRADDR,   -- 9-bit write port address input
  RDADDR => RDADDR,   -- 9-bit read port address input
  DI => DI,           -- 64-bit data input
  DIP => DIP,         -- 8-bit parity data input
  WE => WE            -- 8-bit write enable input
);

-- End of RAMB36SDP_inst instantiation

```

Verilog Instantiation Template

```

// RAMB36SDP: 72x512 Simple Dual-Port BlockRAM w/ ECC
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

RAMB36SDP #(
  .SIM_MODE("SAFE"), // Simulation: "SAFE" vs. "FAST", see "Synthesis and Simulation Design Guide" for details
  .DO_REG(0), // Optional output register (0 or 1)

```

Virtex-5 Libraries Guide for HDL Designs
UG621 (v 12.3) September 21, 2010

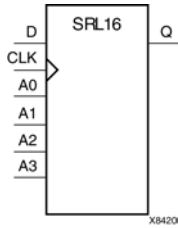
[illegible]

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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

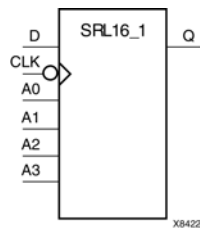
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|---|
| INIT | Hexadecimal | Any 16-Bit Value | All zeros | Sets the initial value of Q output after configuration. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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

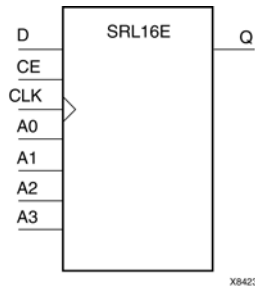
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|--|
| INIT | Hexadecimal | Any 16-Bit Value | All zeros | Sets the initial value of Q output after configuration |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3
```

```
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
//      All FPGAs
// Xilinx HDL Libraries Guide, version 12.3

SRL16E #(
    .INIT(16'h0000) // Initial Value of Shift Register
) SRL16E_inst (
    .Q(Q),           // SRL data output
    .A0(A0),         // Select[0] input
    .A1(A1),         // Select[1] input
    .A2(A2),         // Select[2] input
    .A3(A3),         // Select[3] input
    .CE(CE),         // Clock enable input
    .CLK(CLK),       // Clock input
    .D(D)            // SRL data input
);

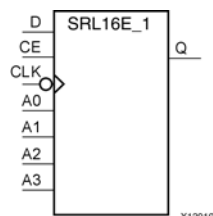
// End of SRL16E_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

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

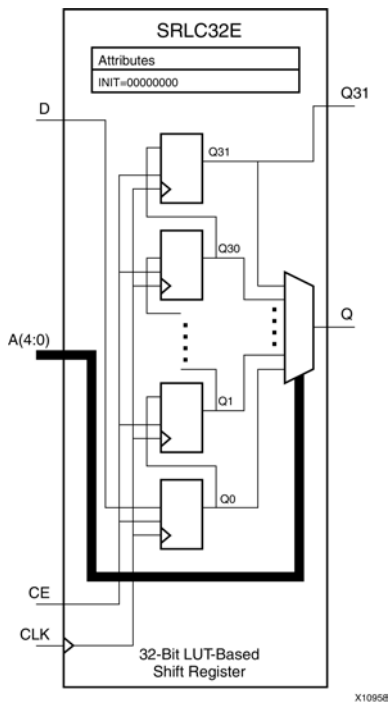
| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|---|
| INIT | Hexadecimal | Any 16-Bit Value | All zeros | Sets the initial value of content and output of shift register after configuration. |

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

SRLC32E

Primitive: 32 Clock Cycle, Variable Length Shift Register Look-Up Table (LUT) with Clock Enable



Introduction

This design element is a variable length, 1 to 32 clock cycle shift register implemented within a single look-up table (LUT). The shift register can be of a fixed length, static length, or it can be dynamically adjusted by changing the address lines to the component. This element also features an active, high-clock enable and a cascading feature in which multiple SRLC32Es can be cascaded in order to create greater shift lengths.

Port Descriptions

| Port | Direction | Width | Function |
|------|-----------|-------|---|
| Q | Output | 1 | Shift register data output |
| Q31 | Output | 1 | Shift register cascaded output (connect to the D input of a subsequent SRLC32E) |
| D | Input | 1 | Shift register data input |
| CLK | Input | 1 | Clock |
| CE | Input | 1 | Active high clock enable |
| A | Input | 5 | Dynamic depth selection of the SRL A=00000 ==> 1-bit shift length A=11111 ==> 32-bit shift length |

Design Entry Method

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

If instantiated, the following connections should be made to this component:

- Connect the CLK input to the desired clock source, the D input to the data source to be shifted/stored and the Q output to either an FDCPE or an FDRSE input or other appropriate data destination.
- The CE clock enable pin can be connected to a clock enable signal in the design or else tied to a logic one if not used.
- The 5-bit A bus can either be tied to a static value between 0 and 31 to signify a fixed 1 to 32 bit static shift length, or else it can be tied to the appropriate logic to enable a varying shift depth anywhere between 1 and 32 bits.
- If you want to create a longer shift length than 32, connect the Q31 output pin to the D input pin of a subsequent SRLC32E to cascade and create larger shift registers.
- It is not valid to connect the Q31 output to anything other than another SRLC32E.
- The selectable Q output is still available in the cascaded mode, if needed.
- An optional INIT attribute consisting of a 32-bit Hexadecimal value can be specified to indicate the initial shift pattern of the shift register.
- (INIT[0] will be the first value shifted out.)

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|-------------|------------------|-----------|---|
| INIT | Hexadecimal | Any 32-Bit Value | All zeros | Specifies the initial shift pattern of the SRLC32E. |

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;

-- SRLC32E: 32-bit variable length shift register LUT
--      with clock enable
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

SRLC32E_inst : SRLC32E
generic map (
  INIT => X"00000000")
port map (
  Q => Q,          -- SRL data output
  Q31 => Q31,      -- SRL cascade output pin
  A => A,          -- 5-bit shift depth select input
  CE => CE,        -- Clock enable input
  CLK => CLK,      -- Clock input
  D => D           -- SRL data input
);

-- End of SRLC32E_inst instantiation
```

Verilog Instantiation Template

```
// SRLC32E: 32-bit variable length cascadable shift register LUT
//           with clock enable
//           Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

SRLC32E #(
    .INIT(32'h00000000) // Initial Value of Shift Register
) SRLC32E_inst (
    .Q(Q),           // SRL data output
    .Q31(Q31),       // SRL cascade output pin
    .A(A),           // 5-bit shift depth select input
    .CE(CE),         // Clock enable input
    .CLK(CLK),       // Clock input
    .D(D)            // SRL data input
);

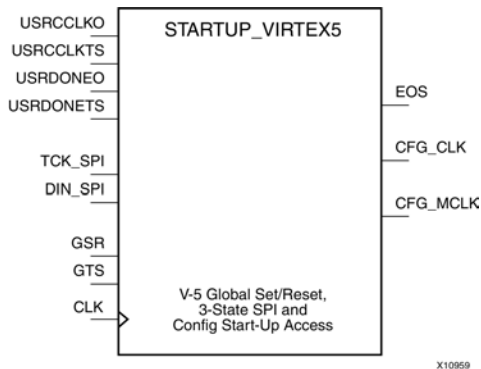
// End of SRLC32E_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

STARTUP_VIRTEX5

Primitive: Virtex®-5 Configuration Start-Up Sequence Interface



Introduction

This design element is used to interface device pins and logic to the Global Set/Reset (GSR) signal, the Global Tristate (GTS) dedicated routing, the internal configuration signals, or the input pins for the SPI PROM if an SPI PROM is used to configure the device. This primitive can also be used to specify a different clock for the device startup sequence at the end of configuring the device, and to access the configuration clock to the internal logic.

Port Descriptions

| Port | Direction | Width | Function |
|-----------|-----------|-------|--|
| EOS | Output | 1 | Active high signal indicates the End Of Configuration. |
| CFGCLK | Output | 1 | Configuration main clock output |
| CFGMCLK | Output | 1 | Configuration internal oscillator clock output |
| USRCCLKO | Input | 1 | Internal user CCLK |
| USRCCLKTS | Input | 1 | Internal user CCLK tristate enable |
| USRDONEO | Input | 1 | Internal user DONE pin output control |
| USRDONETS | Input | 1 | User DONE tristate enable |
| TCK_SPI | Output | 1 | Internal access to the TCK configuration pin when using SPI PROM configuration |
| DIN_SPI | Output | 1 | Internal access to the DIN configuration pin when using SPI PROM configuration |
| GSR | Input | 1 | Active high Global Set/Reset (GSR) signal |
| GTS | Input | 1 | Active high Global Tristate (GTS) signal |
| CLK | Input | 1 | User start-up clock |

Design Entry Method

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

If the dedicated Global Tristate (GTS) is to be 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. CFGMCLK and CFGCLK allow access to the internal configuration clocks, while EOS signals the end of the configuration startup sequence.

If you are configuring the device using a SPI PROM, and access to the SPI PROM is necessary after configuration, use the TCK_SPI and DIN_SPI pins of the component to gain access to the otherwise dedicated configuration input pins.

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_VIRTEX5: Startup primitive for GSR, GTS or startup sequence control,
--                   SPI PROM pins, configuration clock and start-up status
--                   Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

STARTUP_VIRTEX5_inst : STARTUP_VIRTEX5
port map (
    CFGCLK => CFGCLK, -- Config logic clock 1-bit output
    CFGMCLK => CFGMCLK, -- Config internal osc clock 1-bit output
    DINSPI => DINSPI, -- DIN SPI PROM access 1-bit output
    EOS => EOS, -- End of Startup 1-bit output
    TCKSPI => TCKSPI, -- TCK SPI PROM access 1-bit output
    CLK => CLK, -- Clock input for start-up sequence
    GSR => GSR_PORT, -- Global Set/Reset input (GSR cannot be used for the port name)
    GTS => GTS_PORT, -- Global 3-state input (GTS cannot be used for the port name)
    USRCCLKO => USRCCLKO, -- User CCLK 1-bit input
    USRCCLKTS => USRCCLKTS, -- User CCLK 3-state, 1-bit input
    USRDONEO => USRDONEO, -- User Done 1-bit input
    USRDONETS => USRDONETS -- User Done 3-state, 1-bit input
);

-- End of STARTUP_VIRTEX5_inst instantiation
```

Verilog Instantiation Template

```
// STARTUP_VIRTEX5: Startup primitive for accessing GSR, GTS, startup sequence
//                   control, SPI PROM pins, configuration clock and start-up status
//                   Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

STARTUP_VIRTEX5 STARTUP_VIRTEX5_inst (
    .CFGCLK(CFGCLK), // Config logic clock 1-bit output
    .CFGMCLK(CFGMCLK), // Config internal osc clock 1-bit output
    .DINSPI(DINSPI), // DIN SPI PROM access 1-bit output
    .EOS(EOS), // End Of Startup 1-bit output
    .TCKSPI(TCKSPI), // TCK SPI PROM access 1-bit output
    .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)
    .USRCCLKO(USRCCLKO), // User CCLK 1-bit input
    .USRCCLKTS(USRCCLKTS), // User CCLK 3-state 1-bit input
    .USRDONEO(USRDONEO), // User Done 1-bit input
    .USRDONETS(USRDONETS) // User Done 3-state 1-bit input
);

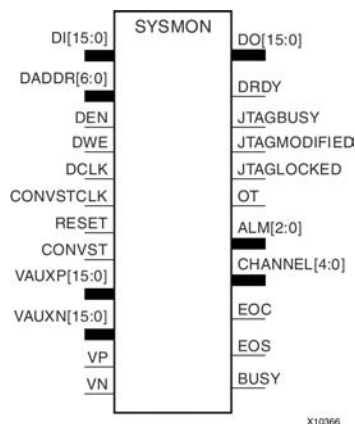
// End of STARTUP_VIRTEX5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA Configuration User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).
- See the [Virtex-5 FPGA User Guide](#).

SYSMON

Primitive: System Monitor



Introduction

This design element is built around a 10-bit, 200-kSPS (kilosamples per second) Analog-to-Digital Converter (ADC). When combined with a number of on-chip sensors, the ADC is used to measure FPGA physical operating parameters, including on-chip power supply voltages and die temperatures. Access to external voltages is provided through a dedicated analog-input pair (VP/VN) and 16 user-selectable analog inputs, known as auxiliary analog inputs (VAUXP[15:0], VAUXN[15:0]). The external analog inputs allow the ADC to monitor the physical environment of the board or enclosure.

Port Descriptions

| Port | Type | Width | Function |
|--------------|--------|-------|--|
| ALM[2:0] | Output | 3 | 3-bit output alarm for temp, Vccint and Vccaux |
| BUSY | Output | 1 | 1-bit output ADC busy signal |
| CHANNEL[4:0] | Output | 5 | 5-bit output channel selection |
| CONVST | Input | 1 | 1-bit input convert start |
| CONVSTCLK | Input | 1 | 1-bit input convert start clock |
| DADDR[6:0] | Input | 7 | 7-bit input address bus for dynamic reconfig |
| DCLK | Input | 1 | 1-bit input clock for dynamic reconfig |
| DEN | Input | 1 | 1-bit input enable for dynamic reconfig |
| DI[15:0] | Input | 16 | 16-bit input data bus for dynamic reconfig |
| DO[15:0] | Output | 16 | 16-bit output data bus for dynamic reconfig |
| DRDY | Output | 1 | 1-bit output data ready for dynamic reconfig |
| DWE | Input | 1 | 1-bit input write enable for dynamic reconfig |
| EOC | Output | 1 | 1-bit output end of conversion |
| EOS | Output | 1 | 1-bit output end of sequence |
| JTAGBUSY | Output | 1 | 1-bit output JTAG DRP busy |
| JTAGLOCKED | Output | 1 | 1-bit output DRP port lock |
| JTAGMODIFIED | Output | 1 | 1-bit output JTAG write to DRP |
| OT | Output | 1 | 1-bit output over temperature alarm |
| RESET | Input | 1 | 1-bit input active high reset |
| VAUXN[15:0] | Input | 16 | 16-bit input N-side auxiliary analog input |
| VAUXP[15:0] | Input | 16 | 16-bit input P-side auxiliary analog input |
| VN | Input | 1 | 1-bit input N-side analog input |
| VP | Input | 1 | 1-bit input P-side analog input |

Design Entry Method

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

Connect all desired input and output ports and set the appropriate attributes for the desired behavior of this component. For simulation, provide a text file to give the analog and temperature to the model. The format for this file is as follows:

```
// Must use valid headers on all columns
// Comments can be added to the stimulus file using '//'
TIME TEMP VCCAUX VCCINT VP VN VAUXP[0] VAUXN[0]
00000 45 2.5 1.0 0.5 0.0 0.7 0.0
05000 85 2.45 1.1 0.3 0.0 0.2 0.0
// Time stamp data is in nano seconds (ns)
// Temperature is recorded in C (degrees centigrade)
// All other channels are recorded as V (Volts)
// Valid column headers are:
// TIME, TEMP, VCCAUX, VCCINT, VP, VN,
// VAUXP[0], VAUXN[0],.....VAUXP[15], VAUXN[15]
// External analog inputs are differential so VP = 0.5 and VN = 0.0 the
// input on channel VP/VN is 0.5 - 0.0 = 0.5V
```

Note When compiling the included code, please do not add any extraneous spaces to the text as this could cause compilation to fail.

Available Attributes

| Attribute | Type | Allowed Values | Default | Description |
|-----------|--------------|----------------------|----------|--------------------------|
| INIT_40 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Configuration register 0 |
| INIT_41 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Configuration register 1 |
| INIT_42 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0800 | Configuration register 2 |
| INIT_43 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Test register 0 |
| INIT_44 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Test register 1 |
| INIT_45 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Test register 2 |
| INIT_46 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Test register 3 |
| INIT_47 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Test register 4 |
| INIT_48 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 0 |
| INIT_49 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 1 |
| INIT_4A | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 2 |
| INIT_4B | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 3 |
| INIT_4C | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 4 |
| INIT_4D | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 5 |
| INIT_4E | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 6 |

| Attribute | Type | Allowed Value | Default | Description |
|------------------|--------------|-------------------------|------------|---|
| INIT_4F | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Sequence register 7 |
| INIT_50 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 0 |
| INIT_51 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 1 |
| INIT_52 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 2 |
| INIT_53 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 3 |
| INIT_54 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 4 |
| INIT_55 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 5 |
| INIT_56 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 6 |
| INIT_57 | Hexa-decimal | 16'h0000 to 16'hffff | 16'h0000 | Alarm limit register 7 |
| SIM_DEVICE | String | "VIRTEX5", "VIRTEX6" | "VIRTEX5" | Specifes the target device family for simulation. |
| SIM_MONITOR_FILE | String | 0 bit String | design.txt | Simulation analog entry file |

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;

-- SYSMON: System Monitor
--      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

SYSMON_inst : SYSMON
generic map (
    INIT_40 => X"0000", -- Configuration register 0
    INIT_41 => X"0000", -- Configuration register 1
    INIT_42 => X"0000", -- Configuration register 2
    INIT_43 => X"0000", -- Test register 0
    INIT_44 => X"0000", -- Test register 1
    INIT_45 => X"0000", -- Test register 2
    INIT_46 => X"0000", -- Test register 3
    INIT_47 => X"0000", -- Test register 4
    INIT_48 => X"0000", -- Sequence register 0
    INIT_49 => X"0000", -- Sequence register 1
    INIT_4A => X"0000", -- Sequence register 2
    INIT_4B => X"0000", -- Sequence register 3
    INIT_4C => X"0000", -- Sequence register 4
    INIT_4D => X"0000", -- Sequence register 5
    INIT_4E => X"0000", -- Sequence register 6
    INIT_4F => X"0000", -- Sequence register 7
    INIT_50 => X"0000", -- Alarm limit register 0
    INIT_51 => X"0000", -- Alarm limit register 1
    INIT_52 => X"0000", -- Alarm limit register 2
    INIT_53 => X"0000", -- Alarm limit register 3
    INIT_54 => X"0000", -- Alarm limit register 4
    INIT_55 => X"0000", -- Alarm limit register 5
    INIT_56 => X"0000", -- Alarm limit register 6
    INIT_57 => X"0000", -- Alarm limit register 7
    SIM_MONITOR_FILE => "design.txt") -- Simulation analog entry file
port map (
    ALM => ALM,           -- 3-bit output for temp, Vccint and Vccaux
    BUSY => BUSY,         -- 1-bit output ADC busy signal
    CHANNEL => CHANNEL,   -- 5-bit output channel selection
    DO => DO,             -- 16-bit output data bus for dynamic reconfig
    DRDY => DRDY,         -- 1-bit output data ready for dynamic reconfig
    EOC => EOC,           -- 1-bit output end of conversion
    EOS => EOS,           -- 1-bit output end of sequence
    JTAGBUSY => JTAGBUSY, -- 1-bit output JTAG DRP busy
    JTAGLOCKED => JTAGLOCKED, -- 1-bit output DRP port lock
    JTAGMODIFIED => JTAGMODIFIED, -- 1-bit output JTAG write to DRP
    OT => OT,             -- 1-bit output over temperature alarm
    CONVST => CONVST,     -- 1-bit input convert start
    CONVSTCLK => CONVSTCLK, -- 1-bit input convert start clock
    DADDR => DADDR,       -- 7-bit input address bus for dynamic reconfig
    DCLK => DCLK,         -- 1-bit input clock for dynamic reconfig
    DEN => DEN,           -- 1-bit input enable for dynamic reconfig
    DI => DI,             -- 16-bit input data bus for dynamic reconfig
    DWE => DWE,           -- 1-bit input write enable for dynamic reconfig
    RESET => RESET,       -- 1-bit input active high reset
    VAUXN => VAUXN,       -- 16-bit input N-side auxiliary analog input
    VAUXP => VAUXP,       -- 16-bit input P-side auxiliary analog input
    VN => VN,             -- 1-bit input N-side analog input
    VP => VP)             -- 1-bit input P-side analog input
);

-- End of SYSMON_inst instantiation

```

Verilog Instantiation Template

```
// SYSMON: System Monitor
//      Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

SYSMON #(
    .INIT_40(16'h0), // Configuration register 0
    .INIT_41(16'h0), // Configuration register 1
    .INIT_42(16'h0), // Configuration register 2
    .INIT_43(16'h0), // Test register 0
    .INIT_44(16'h0), // Test register 1
    .INIT_45(16'h0), // Test register 2
    .INIT_46(16'h0), // Test register 3
    .INIT_47(16'h0), // Test register 4
    .INIT_48(16'h0), // Sequence register 0
    .INIT_49(16'h0), // Sequence register 1
    .INIT_4A(16'h0), // Sequence register 2
    .INIT_4B(16'h0), // Sequence register 3
    .INIT_4C(16'h0), // Sequence register 4
    .INIT_4D(16'h0), // Sequence register 5
    .INIT_4E(16'h0), // Sequence register 6
    .INIT_4F(16'h0), // Sequence register 7
    .INIT_50(16'h0), // Alarm limit register 0
    .INIT_51(16'h0), // Alarm limit register 1
    .INIT_52(16'h0), // Alarm limit register 2
    .INIT_53(16'h0), // Alarm limit register 3
    .INIT_54(16'h0), // Alarm limit register 4
    .INIT_55(16'h0), // Alarm limit register 5
    .INIT_56(16'h0), // Alarm limit register 6
    .INIT_57(16'h0), // Alarm limit register 7
    .SIM_MONITOR_FILE("design.txt") // Simulation analog entry file
) SYSMON_inst (
    .ALM(ALM),           // 3-bit output for temp, Vccint and Vccaux
    .BUSY(BUSY),         // 1-bit output ADC busy signal
    .CHANNEL(CHANNEL),   // 5-bit output channel selection
    .DO(DO),             // 16-bit output data bus for dynamic reconfig
    .DRDY(DRDY),         // 1-bit output data ready for dynamic reconfig
    .EOC(EOC),           // 1-bit output end of conversion
    .EOS(EOS),           // 1-bit output end of sequence
    .JTAGBUSY(JTAGBUSY), // 1-bit output JTAG DRP busy
    .JTAGLOCKED(JTAGLOCKED), // 1-bit output DRP port lock
    .JTAGMODIFIED(JTAGMODIFIED), // 1-bit output JTAG write to DRP
    .OT(OT),             // 1-bit output over temperature alarm
    .CONVST(CONVST),     // 1-bit input convert start
    .CONVSTCLK(CONVSTCLK), // 1-bit input convert start clock
    .DADDR(DADDR),       // 7-bit input address bus for dynamic reconfig
    .DCLK(DCLK),         // 1-bit input clock for dynamic reconfig
    .DEN(DEN),           // 1-bit input enable for dynamic reconfig
    .DI(DI),             // 16-bit input data bus for dynamic reconfig
    .DWE(DWE),           // 1-bit input write enable for dynamic reconfig
    .RESET(RESET),       // 1-bit input active high reset
    .VAUXN(VAUXN),       // 16-bit input N-side auxiliary analog input
    .VAUXP(VAUXP),       // 16-bit input P-side auxiliary analog input
    .VN(VN),             // 1-bit input N-side analog input
    .VP(VP)              // 1-bit input P-side analog input
);

// End of SYSMON_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).

TEMAC

Primitive: Tri-mode Ethernet Media Access Controller (MAC)

Introduction

This design element contains paired embedded Ethernet MACs that are independently configurable to meet all common Ethernet system connectivity needs.

Design Entry Method

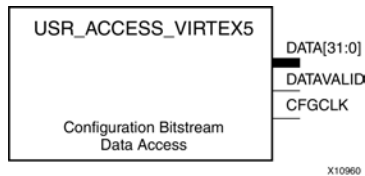
| | |
|-----------------------------|-------------|
| Instantiation | No |
| Inference | No |
| CORE Generator™ and wizards | Recommended |
| Macro support | No |

For More Information

- See the [Virtex-5 Embedded Tri-Mode Ethernet MAC User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).
- See the [Virtex-5 FPGA User Guide](#).

USR_ACCESS_VIRTEX5

Primitive: Virtex-5 User Access Register



Introduction

This design element enables you to access a 32-bit register within the configuration logic. You will thus be able to read the data from the bitstream. One use case for this component is to allow data stored in bitstream storage source to be accessed by the FPGA design after configuration.

Port Descriptions

| Port | Direction | Width | Function |
|-----------|-----------|-------|--|
| DATA | Output | 32 | Configuration Output Data |
| DATAVALID | Output | 1 | Active high DATA port contains valid data. |
| CFGCLK | Output | 1 | Configuration Clock |

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;

-- USR_ACCESS_VIRTEX5: Configuration Data Memory Access Port
--                      Virtex-5
-- Xilinx HDL Libraries Guide, version 12.3

USR_ACCESS_VIRTEX5_inst : USR_ACCESS_VIRTEX5
port map (
  CFGCLK => CFGCLK, -- 1-bit configuration clock output
  DATA => DATA,    -- 32-bit config data output
  DATAVALID => DATAVALID -- 1-bit data valid output
);

-- End of USR_ACCESS_VIRTEX5_inst instantiation
```

Verilog Instantiation Template

```
// USR_ACCESS_VIRTEX5: Configuration Data Memory Access Port
//                               Virtex-5
// Xilinx HDL Libraries Guide, version 12.3

USR_ACCESS_VIRTEX5 USR_ACCESS_VIRTEX5_inst (
    .CFGCLK(CFGCLK),           // 1-bit configuration clock output
    .DATA(DATA),              // 32-bit config data output
    .DATAVALID(DATAVALID)    // 1-bit data valid output
);

// End of USR_ACCESS_VIRTEX5_inst instantiation
```

For More Information

- See the [Virtex-5 FPGA User Guide](#).
- See the [Virtex-5 FPGA Data Sheet DC and Switching Characteristics](#).