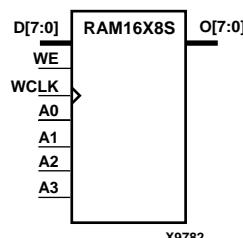


## RAM16X8S

### 16-Deep by 8-Wide Static Synchronous RAM

#### Architectures Supported

RAM16X8S	
Spartan-II, Spartan-IIIE	Macro
Spartan-3	No
Virtex, Virtex-E	Macro
Virtex-II, Virtex-II Pro, Virtex-II Pro X	Primitive
XC9500, XC9500XV, XC9500XL	No
CoolRunner XPLA3	No
CoolRunner-II	No
CoolRunner-IIS	No



RAM16X8S 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.

Except for Spartan-3, Virtex-II, Virtex-II Pro, and Virtex-II Pro X, the initial contents of RAM16X8S cannot be specified directly. See “[Specifying Initial Contents of a RAM](#)” in the RAM16X1D section.

For Spartan-3, Virtex-II, Virtex-II Pro, and Virtex-II Pro X, you can use INIT\_00 through INIT\_07 to specify the initial contents of RAM16X8S as described in the “[Specifying Initial Contents of a Spartan-3, Virtex-II, Virtex-II Pro, and Virtex-II Pro X Wide RAM](#)” section in the RAM16X2S section.

Mode selection is shown in the following truth 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

## Usage

For HDL, this design element can be inferred or instantiated. The instantiation code is shown below. For information on how to infer RAM, see the *XST User Guide*.

## VHDL Instantiation Template

```
-- Component Declaration for RAM16X8S should be placed
-- after architecture statement but before begin keyword

component RAM16X8S
    -- synthesis translate_off
    generic (INIT_00 : bit_vector := X"16";
             INIT_01 : bit_vector := X"16";
             INIT_02 : bit_vector := X"16";
             INIT_03 : bit_vector := X"16";
             INIT_04 : bit_vector := X"16";
             INIT_05 : bit_vector := X"16";
             INIT_06 : bit_vector := X"16";
             INIT_07 : bit_vector := X"16");
    -- synthesis translate_on
    port (O0    : out STD_ULOGIC;
          A0    : in STD_ULOGIC;
          A1    : in STD_ULOGIC;
          A2    : in STD_ULOGIC;
          A3    : in STD_ULOGIC;
          D     : in STD_ULOGIC;
          WCLK : in STD_ULOGIC;
          WE   : in STD_ULOGIC);
end component;

-- Component Attribute specification for RAM16X8S
-- should be placed after architecture declaration but
-- before the begin keyword

-- Enter attributes here

-- Component Instantiation for RAM16X8S should be placed
-- in architecture after the begin keyword

RAM16X8S_INSTANCE_NAME : RAM16X8S
    -- synthesis translate_off
    generic map (INIT_00 => hex_value,
                 INIT_01 => hex_value,
                 INIT_02 => hex_value,
                 INIT_03 => hex_value,
                 INIT_04 => hex_value,
                 INIT_05 => hex_value,
                 INIT_06 => hex_value,
                 INIT_07 => hex_value)
    -- synthesis translate_on
    port map (O0 => user_O0,
              A0 => user_A0,
              A1 => user_A1,
              A2 => user_A2,
              A3 => user_A3,
              D => user_D,
              WCLK => user_WCLK,
              WE => user_WE);
```

## Verilog Instantiation Template

```
RAM16X8S instance_name (.O0 (user_O0),
                        .A0 (user_A0),
                        .A1 (user_A1),
                        .A2 (user_A2),
                        .A3 (user_A3),
                        .D (user_D),
                        .WCLK (user_WCLK),
                        .WE (user_WE));

defparam user_instance_name.INIT_00 = hex_value;
defparam user_instance_name.INIT_01 = hex_value;
defparam user_instance_name.INIT_02 = hex_value;
defparam user_instance_name.INIT_03 = hex_value;
defparam user_instance_name.INIT_04 = hex_value;
defparam user_instance_name.INIT_05 = hex_value;
defparam user_instance_name.INIT_06 = hex_value;
defparam user_instance_name.INIT_07 = hex_value;
```

## Commonly Used Constraints

BEL

INIT\_xx

BLKNM

HBLKNM

HU\_SET

INIT

LOC

RLOC

U\_SET

XBLKNM