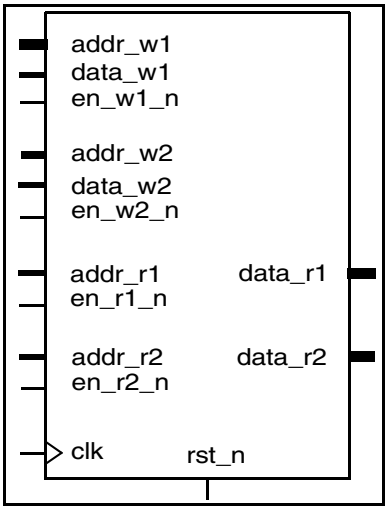
# DW_ram_2r_2w_s_dff

## Sync. Write, Async. Read, 4-port (2rd/2wr) RAM (FF-Based)

Version, STAR and Download Information: IP Directory

## Features and Benefits

- Parameter controlled data width
- Parameter controlled address width (controls memory size)
- Synchronous static memory
- Parameter controlled reset mode (synchronous or asynchronous)




## Description

DW_ram_2r_2w_s_dff implements a 4-port synchronous write, asynchronous read Flip-Flop based RAM with two write ports and two read ports. If both write ports attempt to write to the same RAM address at the same time, write port 1 is written and the data from write port 2 is ignored.

The inputs `en_w1_n` and `en_w2_n` are used to control when data is to be written to the RAM array. When `en_w1_n` is low (logic zero) at the rising edge of clk, the data on `data_w1` is written to RAM location `addr_w1`. When `en_w2_n` is low (logic zero) at the rising edge of `clk` and `addr_w2` not equal to `addr_w1`, the data on `data_w2` is written to RAM location `addr_w2`. This implies that wire port 1 has priority over write port 2 when both write port attempt to write to the same address in the same clock cycle (i.e. if `addr_w1 = addr_w2` and both `en_w1_n` and `addr_w1` are low, the memory location will be written with `data_w1` while `data_w2` will be ignored)

The inputs `en_r1_n` and `en_r2_n` are used to enable the read ports 1 and 2. When `en_r1_n` is inactive (logic one), `data_r1` is driven to all zeros. When `en_w1_n` is active (logic zero) `data_r1` is selected to contain the data in location `addr_r1` of the RAM. When `en_r2_n` is inactive (logic one), `data_r2` is driven to all zeros. When `en_w2_n` is active (logic zero) `data_r2` is selected to contain the data in location `addr_r2` of the RAM.

**Table 1-1 Pin Description**

| Pin Name | Width | Direction | Function |
|---|---|---|---|
| clk | 1 bit | Input | Clock |
| rst_n | 1 bit | Input | Reset, active low |
| en_w1_n | 1 bit | Input | Write port 1 enable, active low |
| addr_w1 | *addr_width* | Input | Write port 1 address |
| data_w1 | *width* | Input | Write port 1 data in |
| en_w2_n | 1 bit | Input | Write port 2 enable, active low |
| addr_w2 | *addr_width* | Input | Write port 2 address |

**Table 1-1 Pin Description (Continued)**

| Pin Name | Width | Direction | Function |
|---|---|---|---|
| data_w2 | *width* | Input | Write port 2 data in |
| en_r1_n | 1 bit | Input | Read port 1 enable, active low |
| addr_r1 | *addr_width* | Input | Read port 1 address |
| data_r1 | *width* | Output | Read port 1 data out |
| en_r2_n | 1 bit | Input | Read port 2 enable, active low |
| addr_r2 | *addr_width* | Input | Read port 2 address |
| data_r2 | *width* | Output | Read port 2 data out |

**Table 1-2 Parameter Description**

| Parameter | Values | Description |
|---|---|---|
| width | 1 to 8192<br>Default = 8 | Data width |
| addr_width | 1 to 12<br>Default = 3 | Address bus width - which controls memory depth. |
| rst_mode | 0 or 1<br>Default = 0 | Determines the reset methodology:<br>0 = `rst_n` asynchronously initializes the RAM,<br>1 = `rst_n` synchronously initializes the RAM |

**Table 1-3 Synthesis Implementations**

| Implementation Name | Function | License Feature Required |
|---|---|---|
| rtl | Synthesis model | DesignWare |

## Related Topics

- Memory – Synchronous RAMs Listing
- DesignWare Building Block IP Documentation Overview

## HDL Usage Through Component Instantiation - VHDL

```
library IEEE,WORK,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DWpackages.all;

entity DW_ram_2r_2w_s_dff_inst is
      generic (
        width : INTEGER := 8;
        addr_width : INTEGER := 3;
        rst_mode : INTEGER := 0
        );
      port (
        clk : in std_logic;
        rst_n : in std_logic;
        en_w1_n : in std_logic;
        addr_w1 : in std_logic_vector(addr_width-1 downto 0);
        data_w1 : in std_logic_vector(width-1 downto 0);
        en_w2_n : in std_logic;
        addr_w2 : in std_logic_vector(addr_width-1 downto 0);
        data_w2 : in std_logic_vector(width-1 downto 0);
        en_r1_n : in std_logic;
        addr_r1 : in std_logic_vector(addr_width-1 downto 0);
        data_r1 : out std_logic_vector(width-1 downto 0);
        en_r2_n : in std_logic;
        addr_r2 : in std_logic_vector(addr_width-1 downto 0);
        data_r2 : out std_logic_vector(width-1 downto 0)
        );
    end DW_ram_2r_2w_s_dff_inst;


architecture inst of DW_ram_2r_2w_s_dff_inst is

    component DW_ram_2r_2w_s_dff
      generic (
        width : INTEGER := 8;
        addr_width : INTEGER := 3;
        rst_mode : INTEGER := 0
        );
      port (
        clk : in std_logic;
        rst_n : in std_logic;

        en_w1_n : in std_logic;
        addr_w1 : in std_logic_vector(addr_width-1 downto 0);
        data_w1 : in std_logic_vector(width-1 downto 0);

        en_w2_n : in std_logic;
        addr_w2 : in std_logic_vector(addr_width-1 downto 0);
```

```
        data_w2 : in std_logic_vector(width-1 downto 0);

        en_r1_n : in std_logic;
        addr_r1 : in std_logic_vector(addr_width-1 downto 0);
        data_r1 : out std_logic_vector(width-1 downto 0);

        en_r2_n : in std_logic;
        addr_r2 : in std_logic_vector(addr_width-1 downto 0);
        data_r2 : out std_logic_vector(width-1 downto 0)
        );
    end component;

begin

    -- Instance of DW_ram_2r_2w_s_dff
    U1 : DW_ram_2r_2w_s_dff
    generic map ( width => width,
                  addr_width => addr_width,
                  rst_mode => rst_mode )

    port map (  clk => clk, rst_n => rst_n,
              en_w1_n => en_w1_n, addr_w1 => addr_w1, data_w1 => data_w1,
              en_w2_n => en_w2_n, addr_w2 => addr_w2, data_w2 => data_w2,
              en_r1_n => en_r1_n, addr_r1 => addr_r1, data_r1 => data_r1,
              en_r2_n => en_r2_n, addr_r2 => addr_r2, data_r2 => data_r2 );


end inst;
```

## HDL Usage Through Component Instantiation - Verilog

```
module DW_ram_2r_2w_s_dff_inst(
          clk, rst_n,
          en_w1_n, addr_w1, data_w1,
          en_w2_n, addr_w2, data_w2,
          en_r1_n, addr_r1, data_r1,
          en_r2_n, addr_r2, data_r2 );

parameter width = 8;
parameter addr_width = 3;
parameter rst_mode = 0;

input clk;
input rst_n;
input en_w1_n;
input [addr_width-1 : 0] addr_w1;
input [width-1 : 0] data_w1;
input en_w2_n;
input [addr_width-1 : 0] addr_w2;
input [width-1 : 0] data_w2;
input en_r1_n;
input [addr_width-1 : 0] addr_r1;
output [width-1 : 0] data_r1;
input en_r2_n;
input [addr_width-1 : 0] addr_r2;
output [width-1 : 0] data_r2;

    // Instance of DW_ram_2r_2w_s_dff
    DW_ram_2r_2w_s_dff #(width, addr_width, rst_mode)
      U1 (
          .clk(clk), .rst_n(rst_n),

          .en_w1_n(en_w1_n), .addr_w1(addr_w1), .data_w1(data_w1),
          .en_w2_n(en_w2_n), .addr_w2(addr_w2), .data_w2(data_w2),

          .en_r1_n(en_r1_n), .addr_r1(addr_r1), .data_r1(data_r1),
          .en_r2_n(en_r2_n), .addr_r2(addr_r2), .data_r2(data_r2)
         );

endmodule
```

# Copyright Notice and Proprietary Information



### Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

### Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

### Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

### Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
690 E. Middlefield Road
Mountain View, CA 94043

www.synopsys.com