100111001
RAM
001101001

DesignWare
Foundation
Building Blocks

DW_ram_2r_w_s_lat

Sync. Write-Port, Async. Dual Read-Port RAM (Latch-Based)

Version, STAR and Download Information: IP Directory

Features and Benefits

- Parameterized word depth
- Parameterized data width
- Synchronous static memory

Description

DW_ram_2r_w_s_lat implements a parameterized, synchronous, three-port static RAM consisting of one write port and two read ports. The RAM can perform simultaneous read and write operations.

```
rd1_addr
rd2_addr
wr_addr
data_in
              data_rd1_out
cs_n
wr_n          data_rd2_out
clk
```
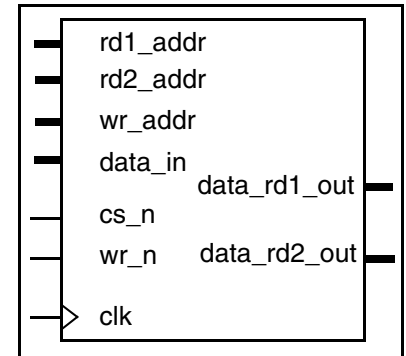
Table 1-1   Pin Description

Pin Name	Width	Direction	Function
clk	1 bit	Input	Clock
cs_n	1 bit	Input	Chip select, active low
wr_n	1 bit	Input	Write enable, active low
rd1_addr	ceil(log$_2$[depth]) bit	Input	Read1 address bus
rd2_addr	ceil(log$_2$[depth]) bit(s)	Input	Read2 address bus
wr_addr	ceil(log$_2$[depth]) bit(s)	Input	Write address bus
data_in	data_width bit(s)	Input	Input data bus
data_rd1_out	data_width bit(s)	Output	Output data bus for read1
data_rd2_out	data_width bit(s)	Output	Output data bus for read2

Table 1-2   Parameter Description

Parameter	Values	Description
data_width	1 to 256 Default = none	Width of `data_in` and `data_out` buses
depth	2 to 256 Default = none	Number of words in the memory array (address width)

DW_ram_2r_w_s_lat
Sync. Write-Port, Async. Dual Read-Port RAM (Latch-Based)

DesignWare Building Blocks

Table 1-3   Synthesis Implementations

Implementation Name	Function	License Feature Required
str	Synthesis model	DesignWare

Table 1-4   Simulation Models

Model	Function
DW06.DW_RAM_2R_W_S_LAT_CFG_SIM	VHDL simulation configuration
dw/dw06/src/DW_ram_2r_w_s_lat_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_ram_2r_w_s_lat.v	Verilog simulation model source code

Write data enters the RAM through the data_in input port, and is read through either the data_rd1_out port or the data_rd2_out port. The RAM is constantly reading regardless of the state of cs_n.

The rd1_addr, rd2_addr, and wr_addr ports are used to address the *depth* words in the memory. If rd1_addr or rd2_addr contains a value beyond the maximum *depth* value, that corresponding output port is driven LOW. For example, if rd1_addr = 7 and depth = 6, then the data_rd1_out bus is driven LOW. If rd2_addr is beyond the maximum depth, then data_rd2_out is driven LOW.

For wr_addr beyond the maximum depth (for example, wr_addr = 7 and depth = 6), nothing occurs and the data is lost. No warnings are given during simulations when an address beyond the scope of depth is used.

> ⚠️ **Attention** This component contains enable signals for internal latches that are derived from the wr_n port. To keep hold times to a minimum, you should consider instances of this component to be individual floorplanning elements.

Chip Selection, Reading and Writing

The cs_n input is the chip select, active low signal that enables data to be written to the RAM. The RAM is constantly reading, regardless of the state of cs_n.

When cs_n, wr_n (write enable, active low), and clk are LOW, data_in is transparent to the internal memory cell being accessed by wr_addr (data_in = output data of the memory cell). Therefore, during the period when clk is LOW, a change in data in is reflected on the output of the internal memory cell being accessed. If one of the read address buses is the same value as wr_addr and wr_n is LOW, data_in equals the output data when clk is LOW. Data is captured into the memory cell on the rising edge of clk.

When cs_n is HIGH, writing to the RAM is disabled.

Application Notes

DW_ram_2r_w_s_lat is intended to be used as small scratch-pad memory, programmable lookup tables, and writable control storage. Because DW_ram_2r_w_s_lat is built from the cells within the ASIC cell library, it should be kept small to obtain an efficient implementation. If a larger memory is required, you should consider using a hard macro RAM from the ASIC library in use.

DesignWare Building Blocks

DW_ram_2r_w_s_lat
Sync. Write-Port, Async. Dual Read-Port RAM (Latch-Based)

Timing Waveforms

Figure 1-1   Instantiated RAM Timing Waveforms

Write Timing, cs_n = 0, Normal Data Input and Output

* Write through is seen only when rd_addr = wr_addr.

Write Timing, cs_n = 0, Changing Data Input and Output

* Write through is seen only when rd_addr = wr_addr.

Read Port 1 Timing, address controlled, cs_n = don't care

Read Port 2 Timing, address controlled, cs_n = don't care

Related Topics

- Memory – Synchronous RAMs Listing

- DesignWare Building Block IP Documentation Overview

HDL Usage Through Component Instantiation - VHDL

```vhdl
library IEEE,DWARE,DWARE;
use IEEE.std_logic_1164.all;
use DWARE.DWpackages.all;
use DWARE.DW_foundation_comp.all;

entity DW_ram_2r_w_s_lat_inst is
  generic (inst_data_width : INTEGER := 8;   inst_depth : INTEGER := 8  );
  port (inst_clk     : in std_logic;
        inst_cs_n    : in std_logic;
        inst_wr_n    : in std_logic;
        inst_rd1_addr : in std_logic_vector(bit_width(inst_depth)-1 downto 0);
        inst_rd2_addr : in std_logic_vector(bit_width(inst_depth)-1 downto 0);
        inst_wr_addr : in std_logic_vector(bit_width(inst_depth)-1 downto 0);
        inst_data_in : in std_logic_vector(inst_data_width-1 downto 0);
        data_rd1_out_inst : out std_logic_vector(inst_data_width-1 downto 0);
        data_rd2_out_inst : out std_logic_vector(inst_data_width-1 downto 0)
        );
end DW_ram_2r_w_s_lat_inst;

architecture inst of DW_ram_2r_w_s_lat_inst is
begin

  -- Instance of DW_ram_2r_w_s_lat
  U1 : DW_ram_2r_w_s_lat
    generic map ( data_width => inst_data_width,  depth => inst_depth )
    port map (clk => inst_clk,  cs_n => inst_cs_n,  wr_n => inst_wr_n,
              rd1_addr => inst_rd1_addr,   rd2_addr => inst_rd2_addr,
              wr_addr => inst_wr_addr,   data_in => inst_data_in,
              data_rd1_out => data_rd1_out_inst,
              data_rd2_out => data_rd2_out_inst );
end inst;

-- pragma translate_off
configuration DW_ram_2r_w_s_lat_inst_cfg_inst of DW_ram_2r_w_s_lat_inst is
  for inst
  end for; -- inst
end DW_ram_2r_w_s_lat_inst_cfg_inst;
-- pragma translate_on
```

DesignWare Building Blocks

DW_ram_2r_w_s_lat
Sync. Write-Port, Async. Dual Read-Port RAM (Latch-Based)

HDL Usage Through Component Instantiation - Verilog

```verilog
module DW_ram_2r_w_s_lat_inst(inst_clk, inst_cs_n, inst_wr_n, inst_rd1_addr,
                             inst_rd2_addr, inst_wr_addr, inst_data_in,
                             data_rd1_out_inst, data_rd2_out_inst );
   parameter data_width = 8;
   parameter depth = 8;
   `define bit_width_depth 3 // ceil(log2(depth))

   input inst_clk;
   input inst_cs_n;
   input inst_wr_n;
   input [`bit_width_depth-1 : 0] inst_rd1_addr;
   input [`bit_width_depth-1 : 0] inst_rd2_addr;
   input [`bit_width_depth-1 : 0] inst_wr_addr;
   input [data_width-1 : 0] inst_data_in;
   output [data_width-1 : 0] data_rd1_out_inst;
   output [data_width-1 : 0] data_rd2_out_inst;

   // Instance of DW_ram_2r_w_s_lat
   DW_ram_2r_w_s_lat #(data_width, depth)
     U1 (.clk(inst_clk),    .cs_n(inst_cs_n),    .wr_n(inst_wr_n),
         .rd1_addr(inst_rd1_addr),    .rd2_addr(inst_rd2_addr),
         .wr_addr(inst_wr_addr),    .data_in(inst_data_in),
         .data_rd1_out(data_rd1_out_inst),
          .data_rd2_out(data_rd2_out_inst) );
endmodule
```

Copyright Notice and Proprietary Information