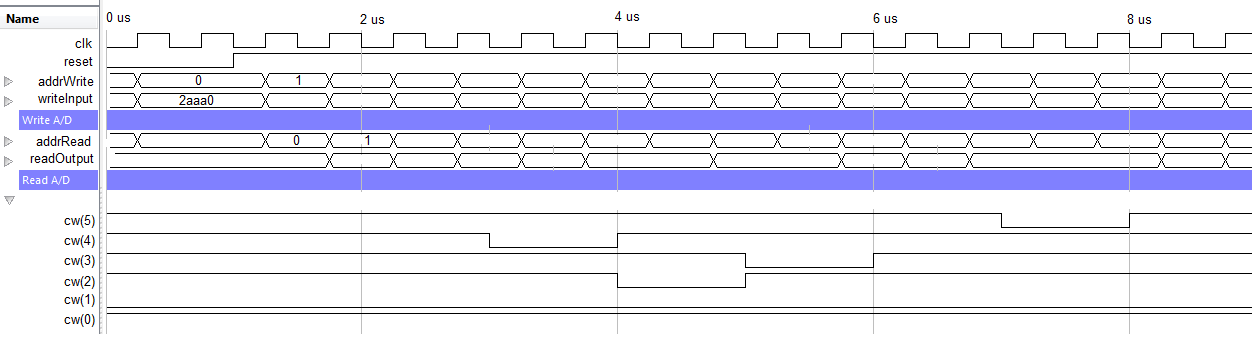
Complete the timing diagram for the BRAM defined on the opposing page.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Addr | writeInput | WREN=cw(4) | WE=cw(3,2) | RDEN=cw(5) | readOutput |
| 0x0 | 0x2AAA0 |  |  |  |  |
| 0x1 | 0x2AAA1 |  |  |  |  |
| 0x2 |  |  |  |  |  |
| 0x3 |  |  |  |  |  |
| 0x4 |  |  |  |  |  |
| 0x5 |  |  |  |  |  |
| 0x6 |  |  |  |  |  |
| 0x7 |  |  |  |  |  |
| 0x8 | 0x2AAA8 |  |  |  |  |
| 0x9 |  |  |  |  |  |
| 0xA |  |  |  |  |  |
| 0xB |  |  |  |  |  |
| 0xC |  |  |  |  |  |
| 0xD |  |  |  |  |  |
| 0xE | 0x2AAAE |  |  |  |  |
| 0xF |  |  |  |  |  |

-------------------------------------------------------------------------

-- Name: Chris Coulston

-- Date: Feb 5, 2015

-- File: lec12\_dp.vhdl

-------------------------------------------------------------------------

library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.NUMERIC\_STD.ALL; -- contains the unsigned data type

library UNIMACRO; -- This contains links to the Xilinx block RAM

use UNIMACRO.vcomponents.all;

entity lec12Dual\_dp is

Port( clk: in STD\_LOGIC;

reset : in STD\_LOGIC;

cw: std\_logic\_vector(5 downto 0));

end lec12Dual\_dp;

architecture behavior of lec12Dual\_dp is

signal writeInput, readOutput: std\_logic\_vector(17 downto 0);

signal addrWrite, addrRead: unsigned(9 downto 0);

signal vecAddrWrite, vecAddrRead : std\_logic\_vector(9 downto 0);

signal n\_reset: std\_logic;

begin

process(clk)

begin

if (rising\_edge(clk)) then

if (reset = '0') then

addrWrite <= (others => '0');

elsif (cw(1 downto 0) = "01") then

addrWrite <= addrWrite + 1;

elsif (cw(1 downto 0) = "11") then

addrWrite <= (others => '0');

end if;

end if;

end process;

addrRead <= addrWrite - 1; -- Have the read follow the writes

writeInput <= "10101010101010" & vecAddrWrite(3 downto 0);

n\_reset <= not reset; -- BRAM reset is active high

vecAddrWrite <= std\_logic\_vector(addrWrite); -- type conversion

vecAddrRead <= std\_logic\_vector(addrRead);

sampleMemory: BRAM\_SDP\_MACRO

generic map (

BRAM\_SIZE => "18Kb", -- Target BRAM, "9Kb" or "18Kb"

DEVICE => "SPARTAN6", -- Target device: "VIRTEX5", "VIRTEX6", "SPARTAN6"

DO\_REG => 0, -- Optional output register disabled

INIT => X"000000000000000000", -- Initial values on output port

INIT\_FILE => "NONE", -- Not sure how to initialize the RAM from a file

WRITE\_WIDTH => 18, -- Valid values are 1-36

READ\_WIDTH => 18, -- Valid values are 1-36

SIM\_COLLISION\_CHECK => "NONE", -- Simulation collision check

SRVAL => X"000000000000000000") -- Set/Reset value for port output

port map (

DO => readOutput, -- Output read data port width=READ\_WIDTH

RDADDR => vecAddrRead, -- Read address, width=FNC(BRAM\_SIZE)

RDCLK => clk, -- 1-bit input clock

RST => n\_reset, -- active high reset

RDEN => cw(5), -- read enable

REGCE => '1', -- read output register enable - ignored

DI => writeInput, -- Input data port, width=WRITE\_WIDTH

WE => cw(3 downto 2), -- RAM is byte read, high or low byte

WRADDR => vecAddrWrite, -- Write address, width=F(BRAM\_SIZE)

WRCLK => clk, -- 1-bit input write clock

WREN => cw(4)); -- 1-bit master write enable

end behavior;