California State University, Sacramento
The College of Engineering and Computer Science

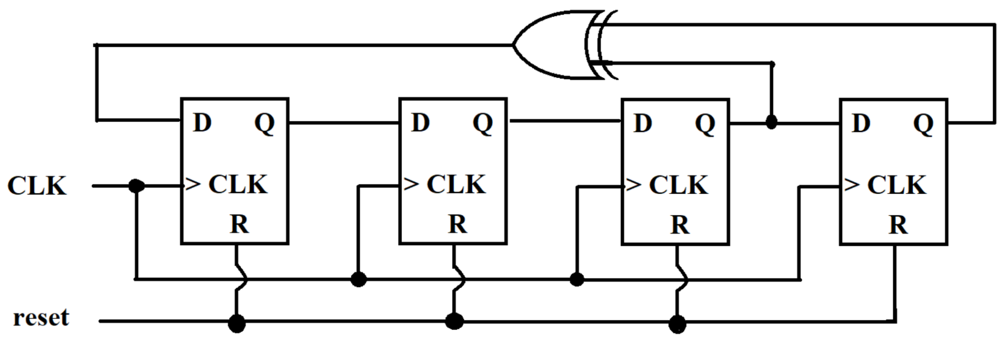
CPE 166 Advanced Logic Design

Midterm 2 – Part 2

Fall 2022

Student Name: VIGOMAR KIM ALGADOR

Part2.1 [20 points].
(1). Design the following LFSR circuit in VHDL and use "1111" as the seed value for the LFSR

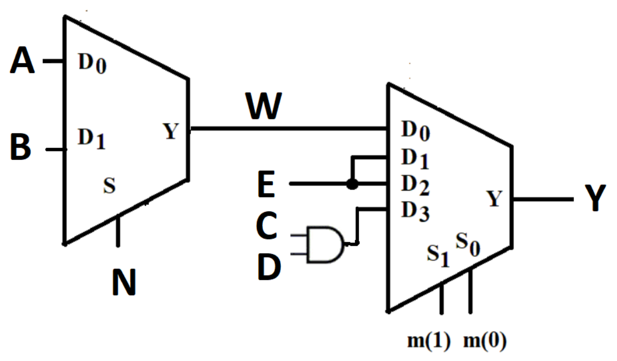


```
library ieee ;
use ieee.std_logic_1164.all ;

entity lfsr is
port (clk, reset: in std_logic;
        q: out std_logic_vector (3 downto 0));
end lfsr;

architecture arch of lfsr is
signal r_reg : std_logic_vector (3 downto 0) ;
signal fb : std_logic;
constant SEED: std_logic_vector (3 downto 0) :="1111";

begin
process (clk , reset)
begin
        if (reset='1') then
                r_reg <= SEED;
        elsif (rising_edge(clk)) then
                r_reg <= fb & r_reg(3 downto 1);
        end if;
end process;

fb <= r_reg(1) xor r_reg(0);
q <= r_reg;
end arch;
```

(2). Write a VHDL testbench for the above design.

```
library ieee ;
use ieee.std_logic_1164.all ;

entity lfsr_tb is
end lfsr_tb;

architecture testbench of lfsr_tb is
signal clk, reset: std_logic;
signal q: std_logic (3 downto 0);

component lfsr
port (clk, reset: in std_logic;
        q: out std_logic_vector (3 downto 0));
end component;

begin
uut: lfsr port map(clk, reset, q);

process begin
        clk <= '0'; wait for 5 ns;
        clk <= '1'; wait for 5 ns;
end process;

process begin
        reset <= '1'; wait for 2 ns;
        reset <= '0' wait for 200 ns;
        wait;
end process;
end testbench;
```

Part2.2 [10 points] Fill out blanks below by using VHDL.



W <= __B__ when __S = '1'__ else

__A__;

With m select
Y <= __W__ when __"00"__ ,

__E__ when __"01"__,

__E__ when __"10"__,

__C and D__ when others;

Part2.3 [20 Points] Design the following timer using VHDL.

This new timer system is specified below.
1 minute = 120 seconds instead of traditional 60 seconds.
1 hour = 120 minutes instead of traditional 60 minutes.
Suppose this new timer can only count from 0 to 5 hours and then it will repeat.
Use clk and reset as inputs.
Use S (seconds), M (minutes), and H (hours) as outputs.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity timer is
port( clk, reset: in std_logic;
        S: out std_logic_vector(6 downto 0);
        M: out std_logic_vector(6 downto 0);
        H: out std_logic_vector(2 downto 0) );
end timer;

architecture arch of timer is
signal S_reg : std_logic_vector(6 downto 0) ;
signal M_reg : std_logic_vector(6 downto 0) ;
signal H_reg : std_logic_vector(2 downto 0) ;
```

```
begin
process (clk,reset)
begin
        if (reset='1') then
                S_reg <= (others=>'0');
                M_reg <= (others=>'0');
                H_reg <= (others=>'0');
        elseif rising_edge(clk) then
                if(S_reg = 119) then            -- count for seconds
                        S_reg <= (others=>'0');
                else
                        S_reg <= S_reg +1 ;
                end if;
                if(S_reg = 119) then            -- count for minutes
                        if(M_reg = 119) then
                                M_reg <= (others=>'0');
                        else
                                M_reg <= M_reg +1 ;
                end if;
                if(S_reg = 119 and M_reg = 119) then          -- count for hours
                        if(H_reg = 4) then
                                H_reg <= (others=>'0');
                        else
                                H_reg <= H_reg +1 ;
                        end if;
                end if;
        end if;
end process;

S <= S_reg;
M <= M_reg;
H <= H_reg;
end arch;
```

Part2.4 [21 Points] Complete the following 128 x 8 RAM design in VHDL. cs is the high active chip select, we is the high active write enable, and oe is the high active output enable.

library __ieee__ ;

use ieee.__std-logic_1164__.all;

use ieee.__std_logic_unsigned__.all;

```
entity ram is
port (
 address : __in____    std_logic_vector ( ___3___ downto  0);

 data   :__out___    std_logic_vector ( ___7___ downto 0);
 cs     :in   std_logic;
 we     :in   std_logic;
 oe     :in   std_logic
 );
end ram;

architecture    beh_ram of ___ram___ is

type memory is array _(0 to 127)_  of std_logic_vector ( _7_ downto  0 );

signal mem : ___memory___ ;

begin

  MEM_WRITE:
  process (address, data, cs, we)
  begin

    if (cs = _'1'_   and we = _'1'_   ) then

      _mem ( conv_integer (address))  <= data ;_

    end if;
   end process;

   MEM_READ:
   process (address, cs, we, oe, mem)
   begin
    if (cs = _'1'_   and we = _'0'_   and oe = _'1'_  )  then

    _data_    <= mem(conv_integer (address));

    else

     _data_    <= ( _others => 'Z'_ );
    end if;
   end process;

end beh_ram;
```