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

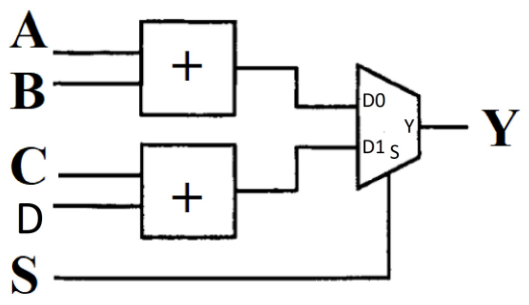
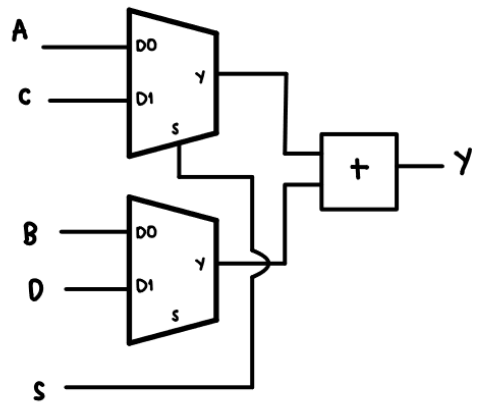
CPE 166 Advanced Logic Design

Final Exam Part 2

Fall 2022

Student Name: Vigomar Kim Algador

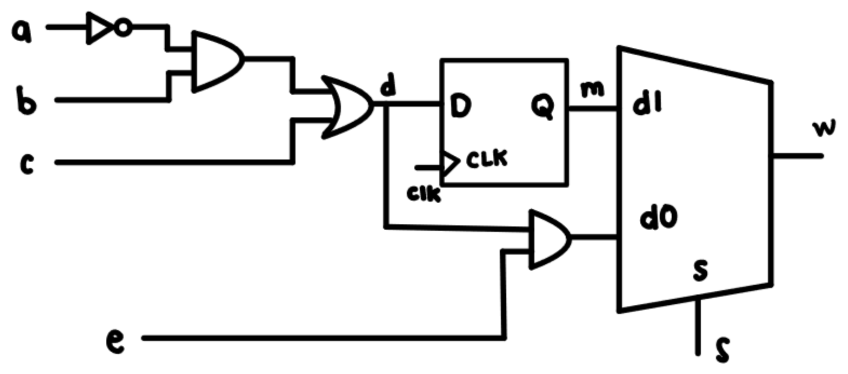
Part 2.1. [15 points] Fill out the 2 tables below.

	Redesign the circuit on the left of this table with resource sharing strategy using only 1 adder. Draw the equivalent circuit diagram below. 

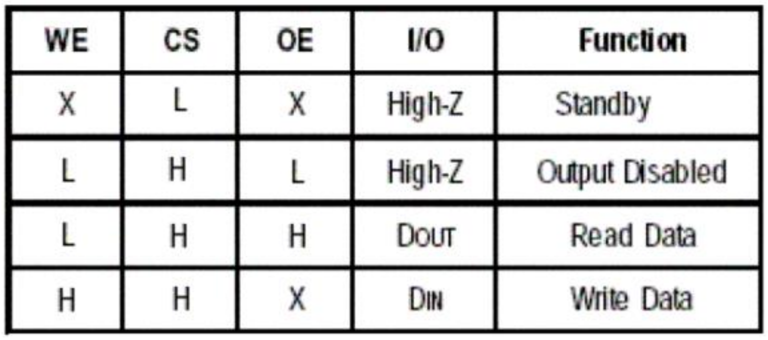
Draw the synthesized schematics for the following Verilog codes.

```
reg    m,  w;
wire  a,  b, c,   d,  e,  s;

assign  d =  ~ a &  b | c  ;

always@(posedge clk)
begin
       m  <= d;
end


always @( s or m or d or e )
begin
     if(s)
          w = m;
     else
          w = d  & e;
end
```

Your Schematic:



Part 2.2 [15 points] An SRAM has **8-bit** databus and **5-bit** address bus. The SRAM function table is shown below:



WE	CS	OE	I/O	Function
X	L	X	High-Z	Standby
L	H	L	High-Z	Output Disabled
L	H	H	DOUT	Read Data
H	H	X	DIN	Write Data

The CS, WE, and OE signals in the above function table are high active. Design the above SRAM in **<u>VHDL</u>**

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

entity ram is
port (      address : in std_logic_vector (4 downto 0) ;
             data    : inout 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 31) 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;
```

Part2.3. [25 points] Implement a **VHDL** design, which writes “11100111” to all of the address locations of the SRAM you designed in question 4. Use SRAM you designed in question 4 as one component in your question 5 design.

FSM DESIGN

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

entity mem_fsm is
port( clk, reset: in std_logic;
      address: out std_logic_vector (4 downto 0);
      data:   inout std_logic_vector (7 downto 0);
      cs, we, oe: out std_logic );
end mem_fsm;

architecture beh of mem_fsm is
type state_type is (s0, s1, s2, s3, s4);
signal state : state_type;
begin

process (clk, reset) begin
    if reset = '1' then
        state <= s0;
    elsif (rising_edge(clk)) then
        case state is
            when s0 =>
                state <= s1;
                address <= 0;
             when s1 =>
                 state <= s2;
                 address <= 0;
             when s2 =>
                  address <= address + 1;
                  if (address == 32)
                      state <= s3;
                  else
                      state <= s2;
                  end if;
               when s3 =>
                   state <= s4;
                   address <= 0;
                when s4 =>
                    address <= address + 1;
                    if (address == 32)
                        state <= s0;
                    else
                        state <= s4;
                    end if;
                when others =>
                    state <= s0;
                    address <= 0;
            end case;
        end if;
end process;

process (state) begin
    case state is
        when s0 =>
            cs <= 0; we <= 0; oe <= 0;
            data <= 4'bzzzzzzzz;
        when s2 =>
            cs <= 1; we <= 1; oe <= 0;
            data <= 4'b11100111;
        when s3 =>
            cs <= 1; we <= 0; oe <= 1;
            data <= 4'b11100111;
        when s4 =>
            cs <= 1; we <= 0; oe <= 1;
            data <= 4'b11100111;
        when others =>
            cs <= 0; we <= 0; oe <= 0;
             data <= 4'bzzzzzzzz;
    end case;
end process;
end beh;
```

TOP DESIGN

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

entity top is
port( clk, reset:  in std_logic;
        address:    out std_logic_vector (4 downto 0);
        data:          out std_logic_vector (7 downto 0);
         cs, we, oe: out std_logic );
end top;

architecture behavioral of top is

component ram
port ( address:       in std_logic_vector (4 downto 0) ;
            data:         inout std_logic_vector (7 downto 0) ;
            cs, we, oe: in std_logic );
end component;

component mem_fsm
port( clk, reset:    in std_logic;
         address:     out std_logic_vector (4 downto 0);
         data:           inout std_logic_vector (7 downto 0);
          cs, we, oe:  out std_logic );
end component;

begin
   g1: ram port map (address, data, cs, we, oe);
   g2: mem_fsm port map (clk, reset, address, data, cs, we, oe);
end behavioral;
```