Exercize 1

Design the combinational circuit with 4-bit input a, b, c, d, 1-bit output Z, and the following specifications:

- Z = 1 if ab > cd (ab is larger than cd)

- Z = 0 if ab < cd (ab is smaller than cd)

- Z is a “Don’t care” condition if ab = cd

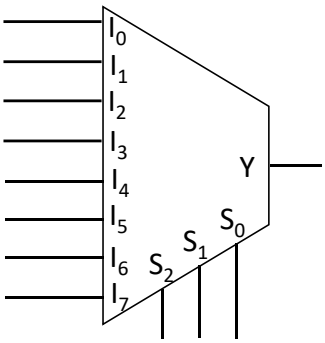
Highlight the following steps:

a) Find the truth table of the required logic function

b) Minimize the function in Sum of Products (SOP) form using Karnaugh map method

c) Draw the corresponding logic circuit

d) Realize the logic function through the 8-to-1 line-mux in the figure. Clearly indicate how a, b, c, d, and Z should be connected to the mux data inputs (I_0, I_1, I_2, I_3, I_4, I_5, I_6, I_7), selection inputs (S_2, S_1, S_0), and output (Y).

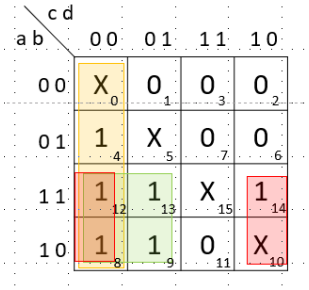


Solution Exercize 1:

a) Truth table

a	b	c	d	Z
0	0	0	0	X
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	X
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	X

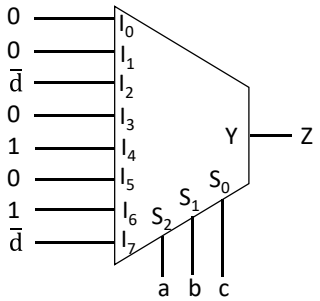
b) Minimization with Karnaugh-map



One possible minimized form is

$$Z = a\bar{c} + \bar{c}\bar{d} + a\bar{d}$$

d) Implementation with 8-to-1 line-mux (note: due to don't care conditions, there is more than one correct implementation)



Exercize 2

Using a Mealy-type machine, design a digital system with 1-bit input X and 1-bit output Z recognizing the input sequence "1110". The output Z must go to ‘1’ as soon as the system receives at the input the last bit of the correct sequence; output Z must be ‘0’ if the correct sequence is not recognized. Use positive edge triggered D-Flip-Flops.

Perform the following steps:

a) Draw the state diagram of the system

b) Find the state transition and output table

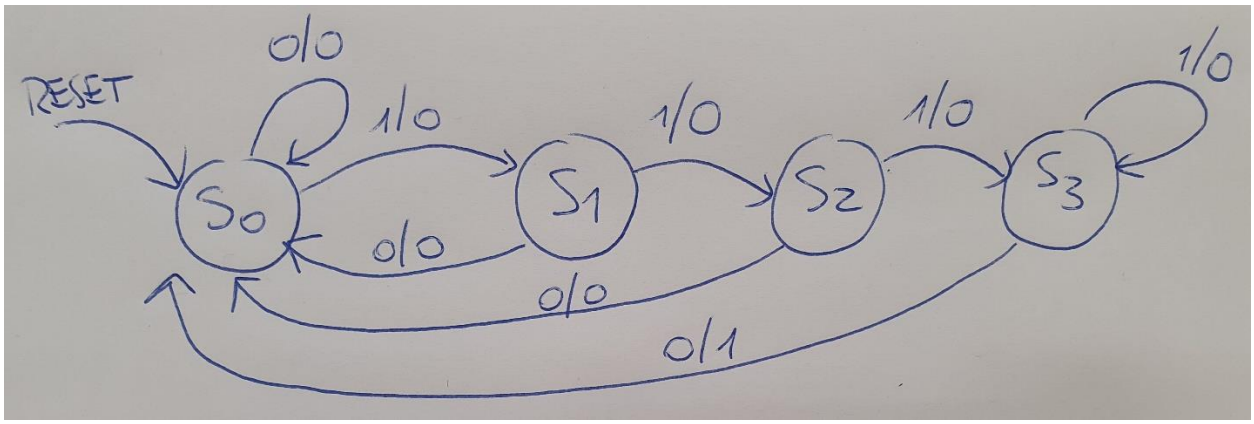
c) What is the minimum number of bits needed to represent the states of the system? How many flilp-flops are needed to implement the system using a 1-hot encoding representation for the states?

d) Write a VHDL code (at your choice) to describe the circuit

NOTE: synthesis of the sequential circuit is not required!

Solution Exercize 2:

a) State diagram



b) State transition and output table

Present state	Future state		Output Z	
	X = 0	X = 1	X = 0	X = 1
S_0	S_0	S_1	0	0
S_1	S_0	S_2	0	0
S_2	S_0	S_3	0	0
S_3	S_0	S_3	1	0

c) The minimum number of bits needed to represent the states of the system is 2.

4 filp-flops are needed to implement the system using a 1-hot encoding representation for the states.

d) A possible VHDL description is a behavioral one:

```
library IEEE;
use IEEE.std_logic_1164.all;
entity seq_rec is
      port (clk, rst, X: in std_logic;
                  Z: out std_logic);
end seq_rec;

architecture beh of seq_rec is
      type state_type is (S0, S1, S2, S3);
      signal state, next_state: state_type;
begin

-- Process 1: Updates state and implements asynchronous reset
      state_register: process (rst, clk)
      begin
            if (rst = '1') then
                  state <= S0;
            elsif (clk'event and clk = '1') then
                  state <= next_state;
            end if;
      end process;
```

```
-- Process 2: Computes the next state (function of input and present state)
      next_state_comb: process (X, state)
      begin

      case state is
            when S0 =>
                  if X = '0' then
                        next_state <= S0;
                  else
                        next_state <= S1;
                  end if;

            when S1 =>
                  if X = '0' then
                        next_state <= S0;
                  else
                        next_state <= S2;
                  end if;

            when S2 =>
                  if X='1' then
                        next_state <= S3;
                  else
                        next_state <= S0;
                  end if;

            when S3 =>
                  if X='0' then
                        next_state <= S0;
                  else
                        next_state <= S3;
                  end if;
      end case;
      end process;

-- Process 3: calculates the output (function of input and state)
      output_comb: process (X, state)
      begin

            if (state = S3 and X='0') then
                  Z <= '1'
            else
                  Z <= '0';
            end if;
      end process;

end beh;
```