An-Najah National University
Department of Computer Engineering
Digital design 2 LAB -10636391-
Second Semester 2020/2021

جامعة النجاح الوطنية
قسم هندسة الحاسوب
مختبر تصميم دوائر الكترونية 2 - 10636391**-**
الفصل الثاني 2021/2020

Instructor: Anas Toma

NAME	ID NUM	SEC
1- أحمد عبد الله جبر القرم	11819195	3/8:00-11:00
-	-	-

Experiment 4: Algorithmic State Machine (Car-Park)

Introduction :

In this lab , I will implement an algorithmic state machine (ASM) on the ZedBoard to implement a system that count the number of cars in a car-park.

Tools used in Lap :

1- Computer lap.

2- Vivado software .

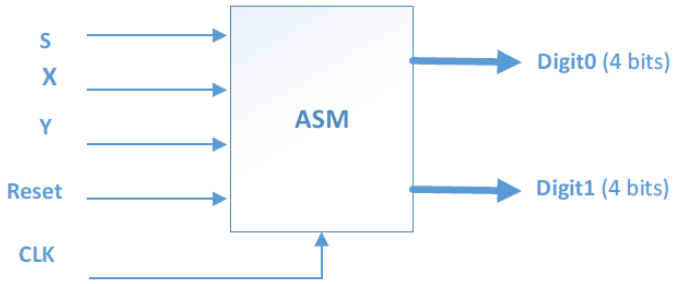
3- Zboard from Xilinx.

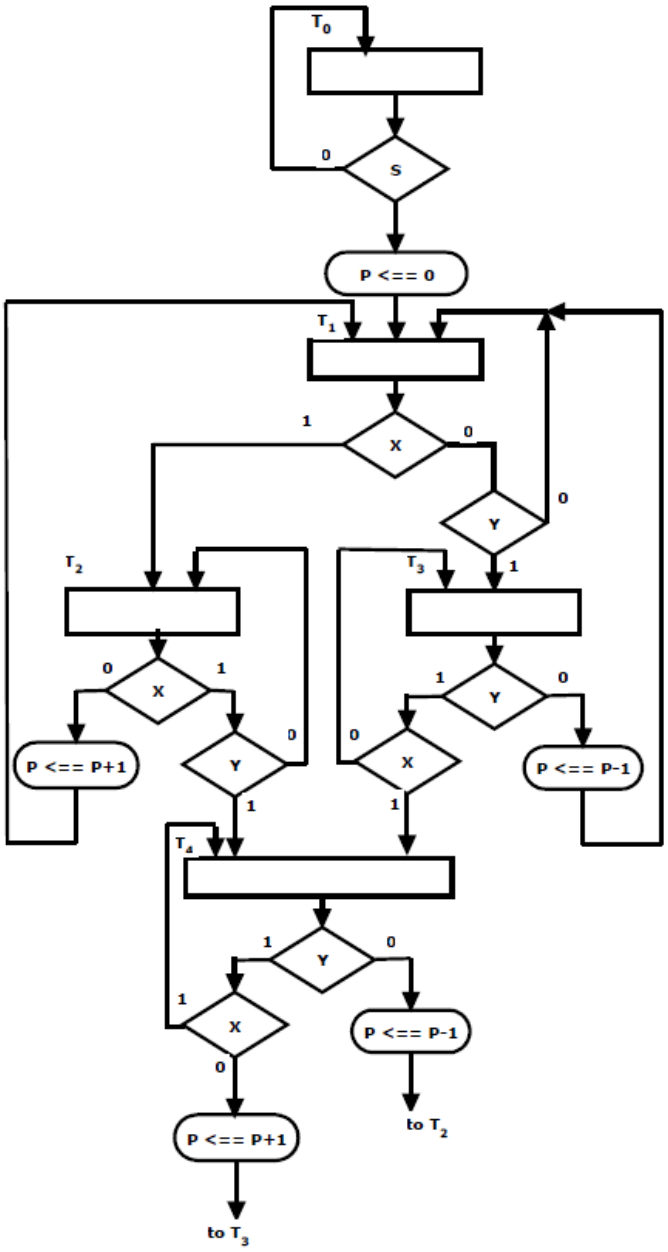
4- VHDL

Procedure :

Part 1 : ASM Simulation

I implemented the given asm in vhdl using 3 processe .





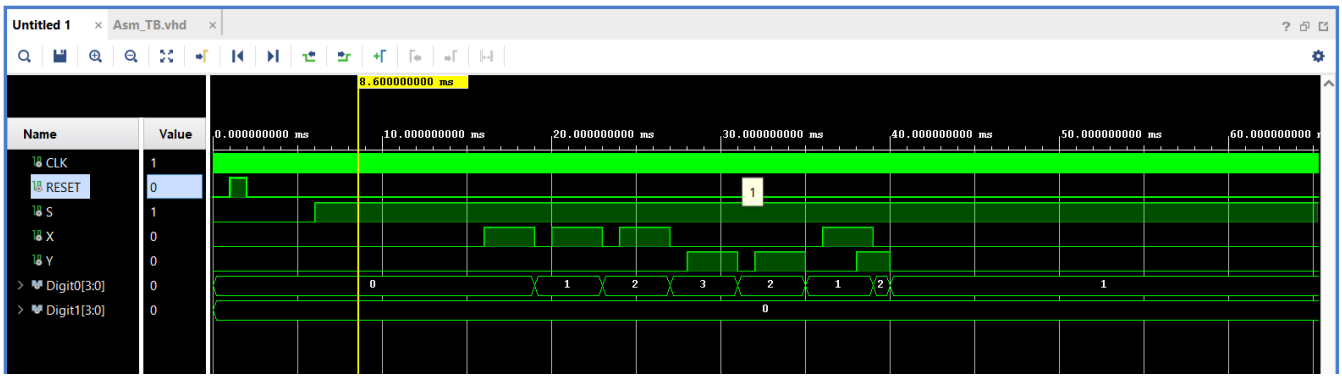
ASM Vhdl Code:

E:/projects/ASM_Car_Park/ASM_Car_Park.srcs/sources_1/new/C.vhd

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use ieee.numeric_std.all;
4  use ieee.std_logic_unsigned.all;
5  entity ASM is
6      port (
7          CLK, RESET, S, X , Y : in STD_LOGIC;
8          Digit0 , Digit1 : out STD_LOGIC_VECTOR (3 downto 0)
9      );
10 end ASM;
11 architecture Behavioral of ASM is
12     type state_type is (T0, T1,T2,T3,T4);
13     signal next_state :state_type ;
14     signal current_state:state_type:=T0;
15     signal temp0,temp1 :std_logic_vector (3 downto 0):="0000";
16 begin
17     state_register : process( clk, reset)
18     begin
19         if(reset = '1') then
20             current_State <= T0;
21
22         elsif (clk'event and clk = '1') then
23             current_State <= next_state;
24
25
26         end if;
27     end process;
28     next_state_process : process( s, X, Y, current_State)...
79     output_process : process(clk)...
146 end Behavioral;
147
```

```
    next_state_process : process( s, X, Y, current_State)
    begin
        case current_state is
            when T0 =>
                if ( s='1' )then
                   next_state <= T1;
                else
                    next_state <= T0;
                end if;
            when T1 =>
                if (x='0') then
                    if (y='0') then
                        next_state <= T1;
                    else
                        next_state <= T3;
                    end if;
                else
                    next_state <= T2;
                end if;
            when T2 =>
                if (x='1')then
                    if (y='0')then
                        next_state <= T2;
                    else
                        next_state <= T4;
                    end if;
                else
                    next_state <= T1;
                end if;
            when T3 =>
                if (y='0')then
                    next state <= T1;
                else
                    if (x='0')then
                        next_state <= T3;
                    else
                        next_state <= T4;
                    end if;
                end if;
              when T4 =>
                if (y='0')then
                    next_state <= T2;
                else
                    if (x='1')then
                        next_state <= T4;
                    else
                        next_state <= T3;
                    end if;
                end if;
        end case;
    end process;
    output_process : process(clk)
    begin
        if(clk'event and clk = '1') then
            case current_state is
                when T0 =>
                   Digit0<="0000";
                   Digit1<="0000";


                when T2 =>
                if (x='0' and clk='1')then
                    if (temp0 < "1001")then
                        temp0<=temp0+'1';
                        else
                        temp0<="0000";
                        if (temp1 < "1001")then
                            temp1<=temp1+'1';
                        else
                            Digit1<="0000";
                        end if;
                    end if;
                end if;

                when T3 =>
                if (y='0') then
                    if (temp0 > "0000")then
                        temp0<=temp0-'1';
                    else
                        temp0<="1001";
                        if (temp1>"0000")then
                            temp1<=temp1-'1';
                        else
                            Digit1<="0000";
                        end if;
                    end if;
                elsif (x='0')then
                    if (temp0 < "1001")then
                        temp0<=temp0+'1';
                        else
                        Digit0<="0000";
                        if (temp1<"1001")then
                            temp1<=temp1+'1';
                        else
                            Digit1<="1001";
                        end if;
                    end if;
                end if;
                when T1 =>null;
            end case;
        end if;
        Digit0<= temp0 ;
        Digit1<= temp1 ;
    end process ;
```

Asm Simulation :



Part 2 : Top Level (ASM Car-Park)

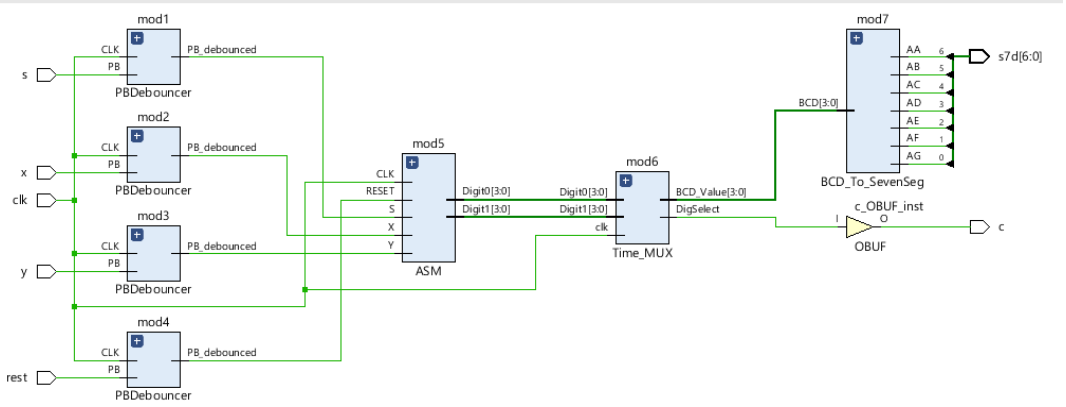
I had created the Top level entity with four component :

a. PBDebouncer: to debounce the signals from the push buttons.

b. Time_MUX: to select between the two digits (Tens and Ones) of the BCD counter.

c. BCD_To_SevenSeg:

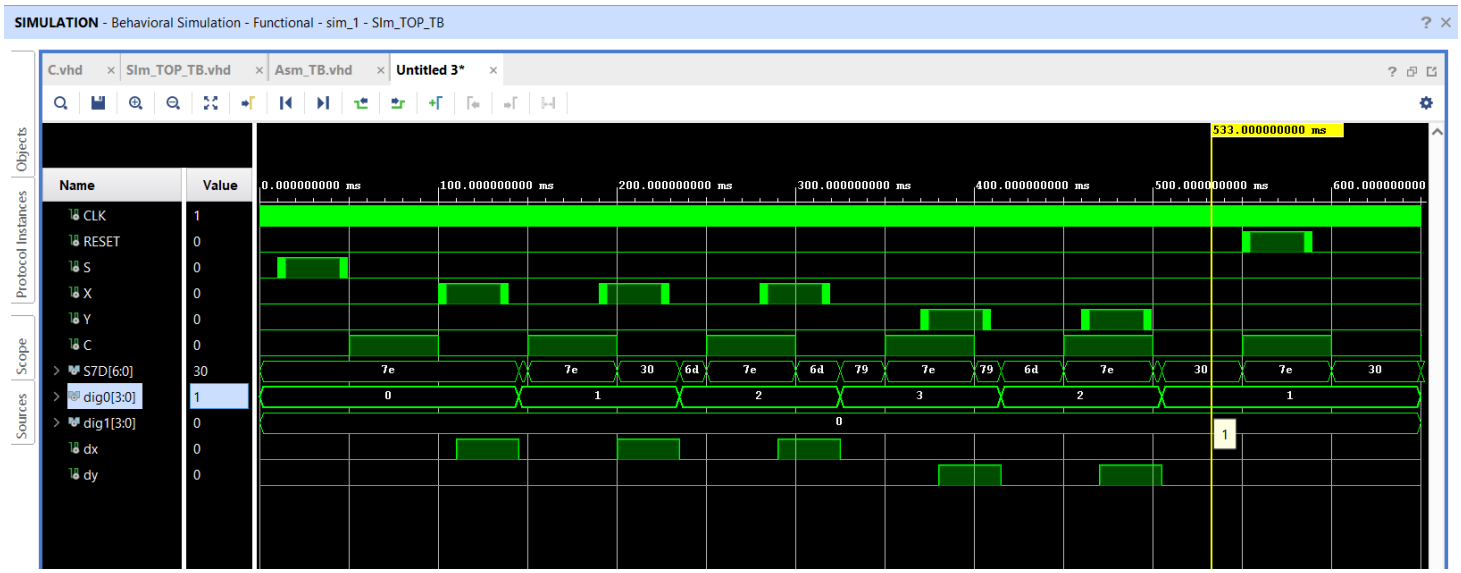
d. The previous Asm component.



Top level Vhdl Code :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Top_L_ASM is
    Port ( s : in STD_LOGIC;
           x : in STD_LOGIC;
           y : in STD_LOGIC;
           rest:in std_logic;
           clk : in STD_LOGIC;
           s7d : out STD_LOGIC_VECTOR (6 downto 0);
           c : out STD_LOGIC);
end Top_L_ASM;

architecture Behavioral of Top_L_ASM is

component ASM port (
        clk, RESET, S, X , Y : in STD_LOGIC;
        Digit0 , Digit1 : out STD_LOGIC_VECTOR (3 downto 0)
    );
end component;

component PBDebouncer Port (
        clk : in STD_LOGIC;
        PB : in STD_LOGIC;
        PB_debounced: out STD_LOGIC
        );
end component;

component Time_MUX Port (
        clk : in STD_LOGIC;
        Digit0 : in STD_LOGIC_VECTOR (3 downto 0);
        Digit1 : in STD_LOGIC_VECTOR (3 downto 0);
        BCD_Value : out STD_LOGIC_VECTOR (3 downto 0);
        DigSelect : out STD_LOGIC
        );
        end component;

component BCD_To_SevenSeg Port (
        BCD: in STD_LOGIC_VECTOR(3 downto 0);
        AA : out STD_LOGIC;
        AB : out STD_LOGIC;
        AC : out STD_LOGIC;
        AD : out STD_LOGIC;
        AE : out STD_LOGIC;
        AF : out STD_LOGIC;
        AG : out STD_LOGIC
        );
        end component;
signal ds,dx,dy,dRESET :std_logic;
signal dig0,dig1,mux_out:std_logic_vector (3 downto 0);
signal AA,AB,AC,AD,AE,AF,AG:std_logic;
begin
mod1 : PBDebouncer port map (clk,s,ds);
mod2 : PBDebouncer port map (clk,x,dx);
mod3 : PBDebouncer port map (clk,y,dy);
mod4 : PBDebouncer port map (clk,rest,dRESET);
mod5 : ASM port map (clk,dRESET,ds,dx,dy,dig0,dig1);
mod6: Time_MUX port map(clk,dig0,dig1,mux_out,c);
mod7: BCD_To_SevenSeg port map (mux_out,AA,AB,AC,AD,AE,AF,AG);
s7d(6)<= AA;
s7d(5)<= AB;
s7d(4)<= AC;
s7d(3)<= AD;
s7d(2)<= AE;
```

Top level simulation :



Synthesis, Implementation, and Bitstream Generation:

I had connected the Pins as shown :

Signal Name	Package Pin	I/O Std
JA1	Y11	LVCMOS33
JA2	AA11	LVCMOS33
JA3	Y10	LVCMOS33
JA4	AA9	LVCMOS33
JB1	W12	LVCMOS33
JB2	W11	LVCMOS33
JB3	V10	LVCMOS33
JB4	W8	LVCMOS33

	Signal Name	Package Pin	I/O Std
Up Button	S	T18	LVCMOS18
Down Button	Reset	R16	LVCMOS18
Right Button	X	R18	LVCMOS18
Left Button	Y	N15	LVCMOS18

Conclusion:

In this excrement I had learned how to create block that behave in a certain ASM chart (Algorithmic State Machine of Car-Park) .