CSE460

VLSI Design Lab

(Sec 08)

Assignment-2

Submitted By:

Rafsan Al Mamun

(18301033)

buX Username: Rafsan_Al_Mamun

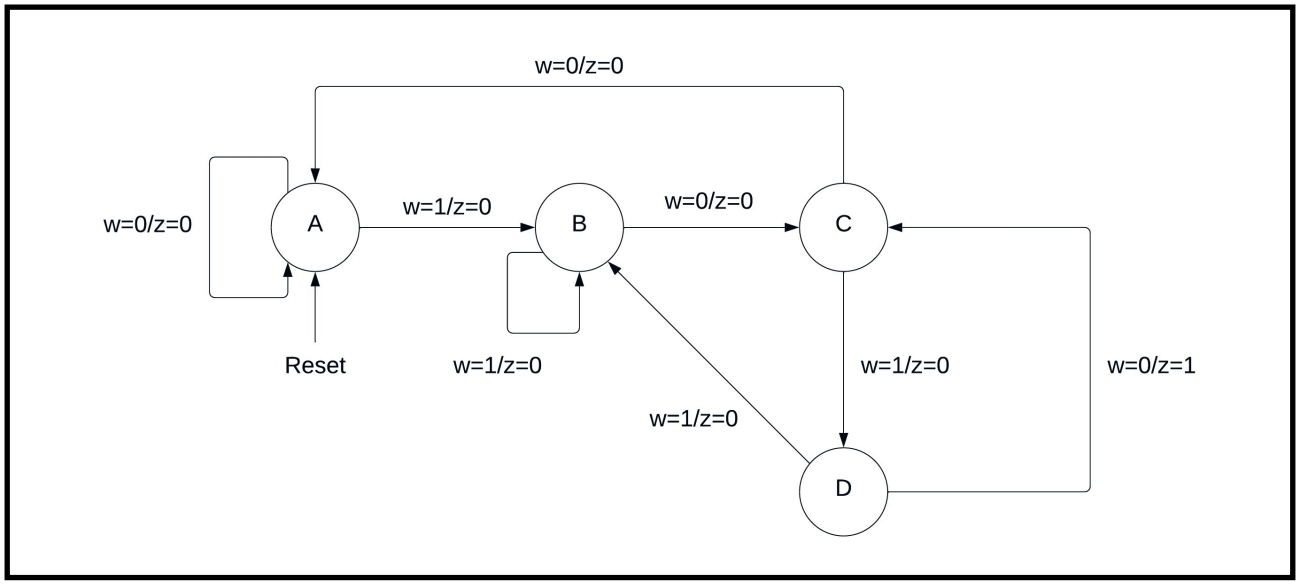
Submission Date: 30th March, 2022

Problem 1:

An FSM has an input w and an output z. The machine has to generate z = 1 when the following patterns in w are detected: 1010; otherwise, z = 0. The machine should reset when (Reset = 0).

A. We need to use a Mealy-Type FSM since the output is happening in the same clock cycle as when the pattern is detected instead of being a clock cycle later.

B. State Diagram:



C. State-Assigned Table:

Present State	Next State		z	
	w = 0	**w = 1**	**w = 0**	**w = 1**
00	00	01	0	0
01	10	01	0	0
10	00	11	0	0
11	10	01	1	0

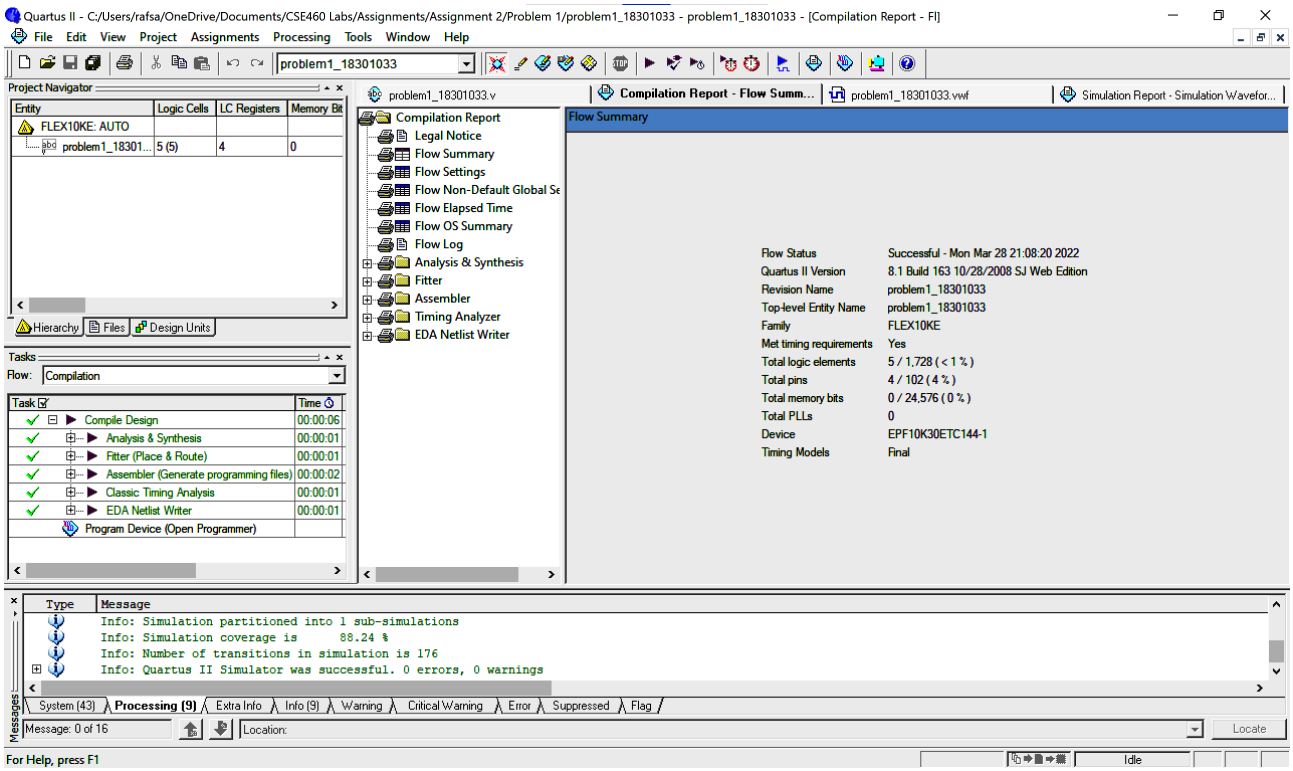
where, A = 00, B = 01, C = 10 and D = 11.

D. module problem1_18301033 (clk, rst, w, z);

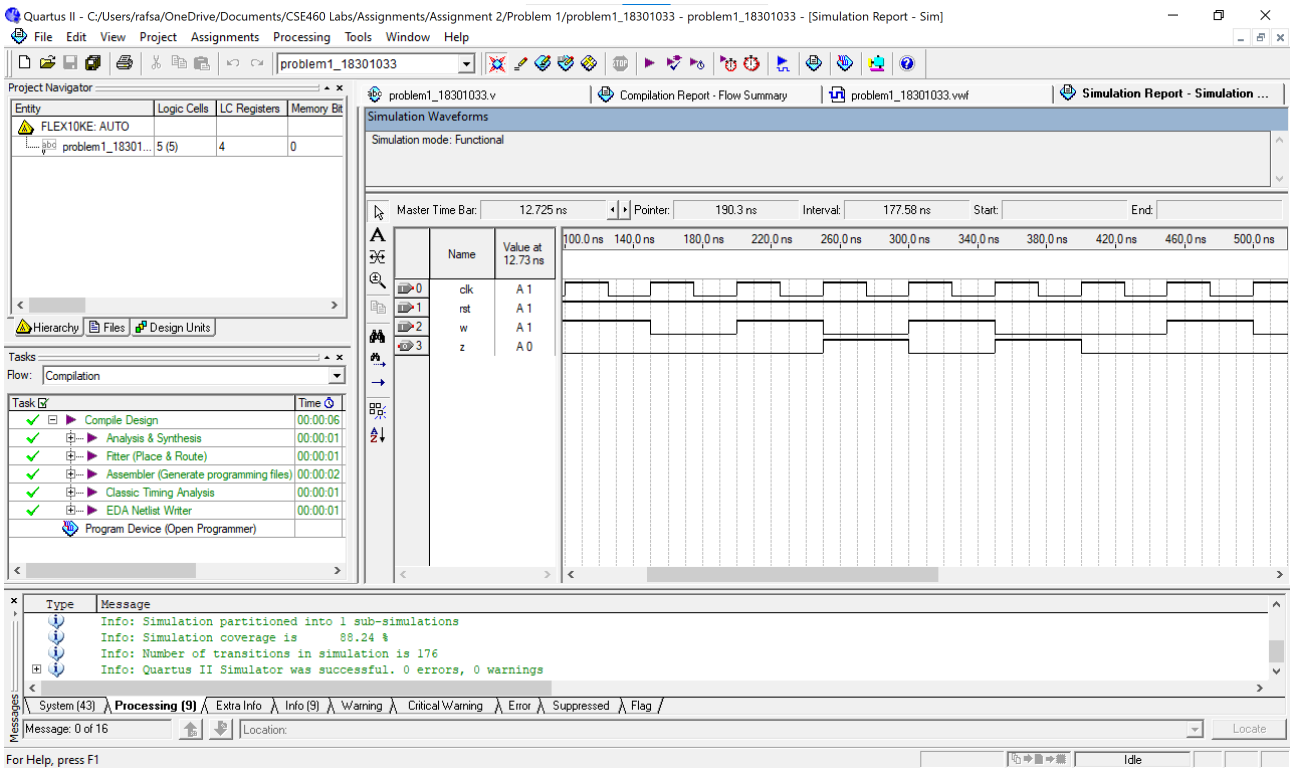
```
input clk, rst, w;
output reg z;
reg [1:0] present, next;
parameter [1:0] A=0, B=1, C=2, D=3;
```

```
always @(present, w)
    case (present)

    A:  if (w) begin
                    next = B;
                    z = 0;
            end
            else begin
                    next = A;
                    z = 0;
            end

    B:  if (w) begin
                    next = B;
                    z = 0;
            end
            else begin
                    next = C;
                    z = 0;
            end

    C:  if (w) begin
                    next = D;
                    z = 0;
            end
            else begin
                    next = A;
                    z = 0;
            end

    D:  if (w) begin
                    next = B;
                    z = 0;
            end
            else begin
                    next = C;
                    z = 1;
            end
    endcase

always @(posedge clk, negedge rst)
    if (rst == 0)
            present <= A;
    else
            present <= next;

endmodule
```

E. Compilation Report:



Simulation Report:



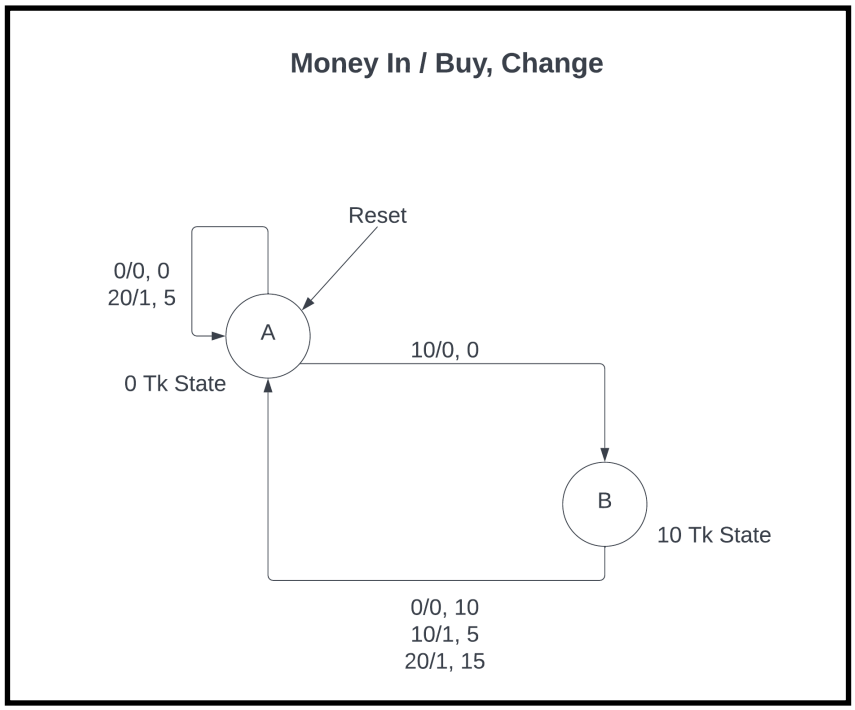
Explanation:

As can be seen in the simulation report, between 100 - 300 ns, there was a 1010 pattern, making the output equal to 1 during 250 - 300 ns (the same clock cycle the pattern was detected). Then again, between 200 - 400 ns, there was another overlapping 1010 pattern detected, making the output equal to 1 during 350 - 400 ns (the same clock cycle the pattern was detected). In all other times, the output was equal to 0.

Problem 2:

You have to design a vending machine for a 15 Tk product. User's money, returned money by the machine, and product bought condition is represented as mny (2-bit input), chg (output), and buy (1-bit output). The vending machine can only accept inputs: no money (mny = 00), Tk 10 (mny = 01), and Tk 20 (mny = 10). If no money is inserted into the machine, it will immediately return the user's money back. Once an acceptable input is more than or equal to 20 Tk, the machine immediately generates an output (buy=1), goes back to the initial state, and gives back the change (if required).

A. State Diagram:



B. It will produce 4 types of changes: 0Tk, 5Tk, 10Tk and 15Tk.

clock	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
mny	0	10	0	0	10	10	0	0	20	0	10	20
buy	0	0	0	0	0	1	0	0	1	0	0	1
chg	0	0	10	0	0	5	0	0	5	0	0	15

Hence, the chg output should be of 2bits to represent 4 changes.

C. State-Assigned Table:

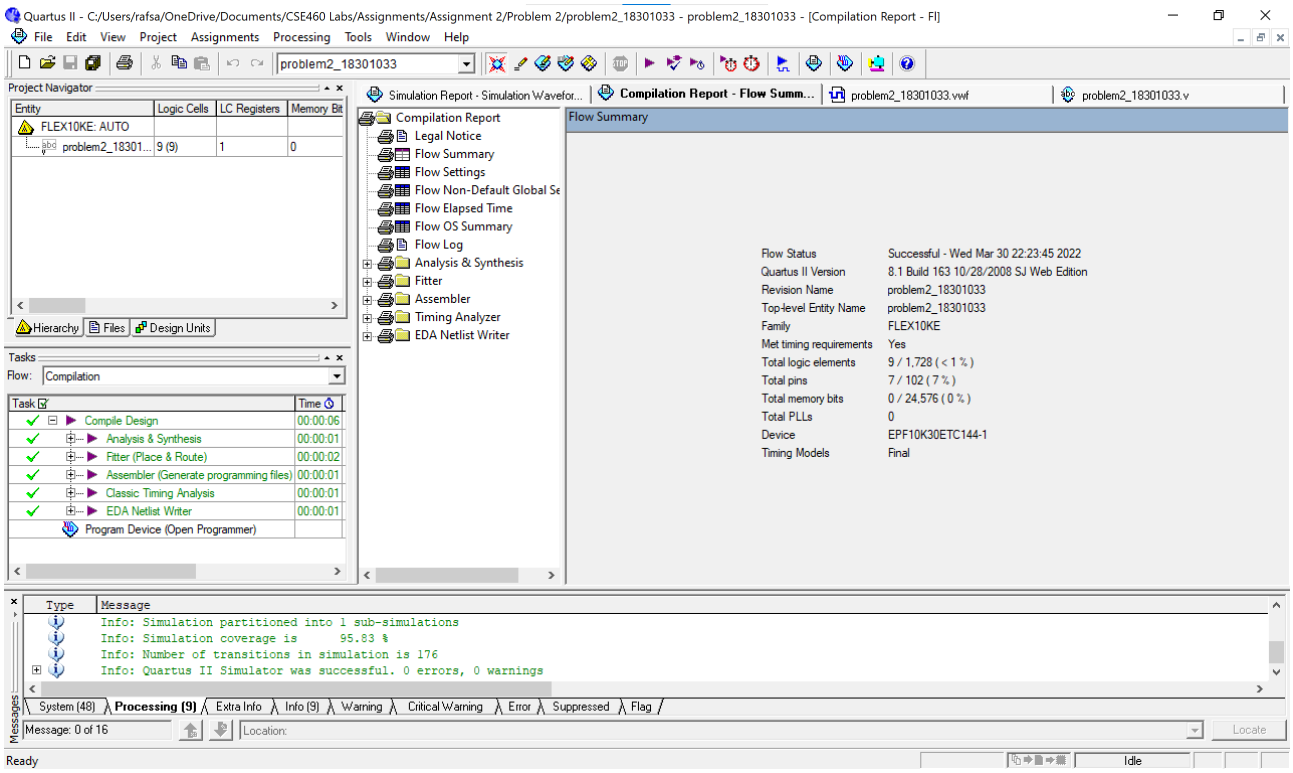
Present State	**Next State**			**buy**			**chg**		
	mny			**mny**			**mny**		
	00	**01**	**10**	**00**	**01**	**10**	**00**	**01**	**10**
0	0	1	0	0	0	1	00	00	01
1	0	0	0	0	1	1	10	01	11

where A = 0 and B = 1 for states, and 0Tk = 00, 5Tk = 01, 10Tk = 10 and 15Tk = 11 for change given.

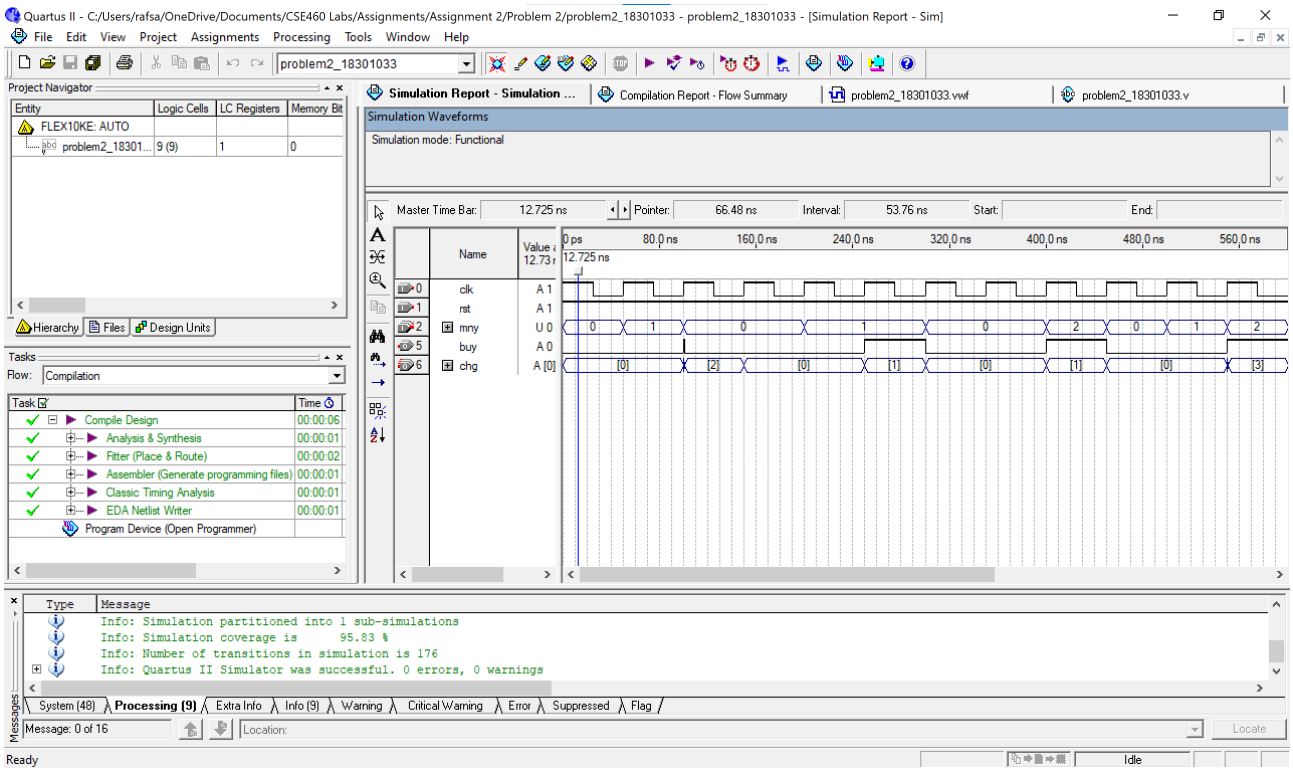
D.
```
module problem2_18301033 (clk, rst, mny, buy, chg);

    input clk, rst;
    input [1:0] mny;
    output reg buy;
    output reg [1:0] chg;
    reg present, next;
    parameter A=0, B=1;
    parameter [1:0] R0=0, R5=1, R10=2, R15=3;

    always @(present, mny)
        case(present)
                A:  if (mny == 2'b00)
                        begin
                                next = A;
                                buy = 0;
                                chg = R0;
                        end

                        else if (mny == 2'b01)
```

```
                begin
                        next = B;
                        buy = 0;
                        chg = R0;
                end

                else if (mny == 2'b10)
                begin
                        next = A;
                        buy = 1;
                        chg = R5;
                end

        B:  if (mny == 2'b00)
                begin
                        next = A;
                        buy = 0;
                        chg = R10;
                end

                else if (mny == 2'b01)
                begin
                        next = A;
                        buy = 1;
                        chg = R5;
                end

                else if (mny == 2'b10)
                begin
                        next = A;
                        buy = 1;
                        chg = R15;
                end
    endcase

always @(posedge clk, negedge rst)
    if (rst == 0)
            present <= A;

    else
            present <= next;

endmodule
```

E. Compilation Report:



Simulation Report:



Explanation:

As can be seen from the simulation report, whenever there is a 0Tk input only (mny = 00), buy = 0 and the change is 0Tk as well (chg = 00) [0-50ns]. When there is a single 10Tk input (mny = 01), buy = 0 and the change is 0Tk also (chg = 00) [50-100ns]. However, when this input is followed by:

- A 0Tk input (mny = 00), buy = 0 and the change is 10Tk (chg = 10) since 10Tk < 15Tk [100-150ns].
- A 10Tk input (mny = 01), buy = 1 and the change is 5Tk (chg = 01) since 20Tk > 15Tk [250-300ns].
- A 20Tk input (mny = 10), buy = 1 and the change is 15Tk (chg = 11) since 30Tk > 15Tk [550-600ns].

When there is a single 20Tk input (mny = 10), buy = 1 and change is 5Tk (chg = 01) since 20Tk > 15Tk [400-450ns]. Hence, the simulation follows the expected outputs of the problem statement.