BRAC UNIVERSITY

Inspiring Excellence

CSE 460

VLSI LAB

ASSIGNMENT 04

Submitted by

Md. Nasimuzzaman

Id: 19101051

CSE-460 Fall-2022

Submission Date: 16th November 2022

Lab Assignment 04

Problem statement

<u>Problem 1</u>:

You have to design a vending machine in Quartus for a **15 Tk product**. User's money, returned money by the machine, and product bought condition is represented as **cash_in (2-bit input), chg (output), and buy (1-bit output)** respectively.

The vending machine can only accept **three** inputs: no money (**cash_in** = 00), Tk 10 (**cash_in** = 01), and Tk 20 (**cash_in** = 10). 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).

<u>Requirements:</u>

A. Draw the state diagram.

B. How many types of changes (return) will the machine produce? How many bit/bits should **chg** output have to represent the returned money in the code?

C. Write the state-assigned table.

D. Write the Verilog code.

E. Run the simulation, and verify your answer. Add two ss (full screen and zoomed). Describe it briefly.

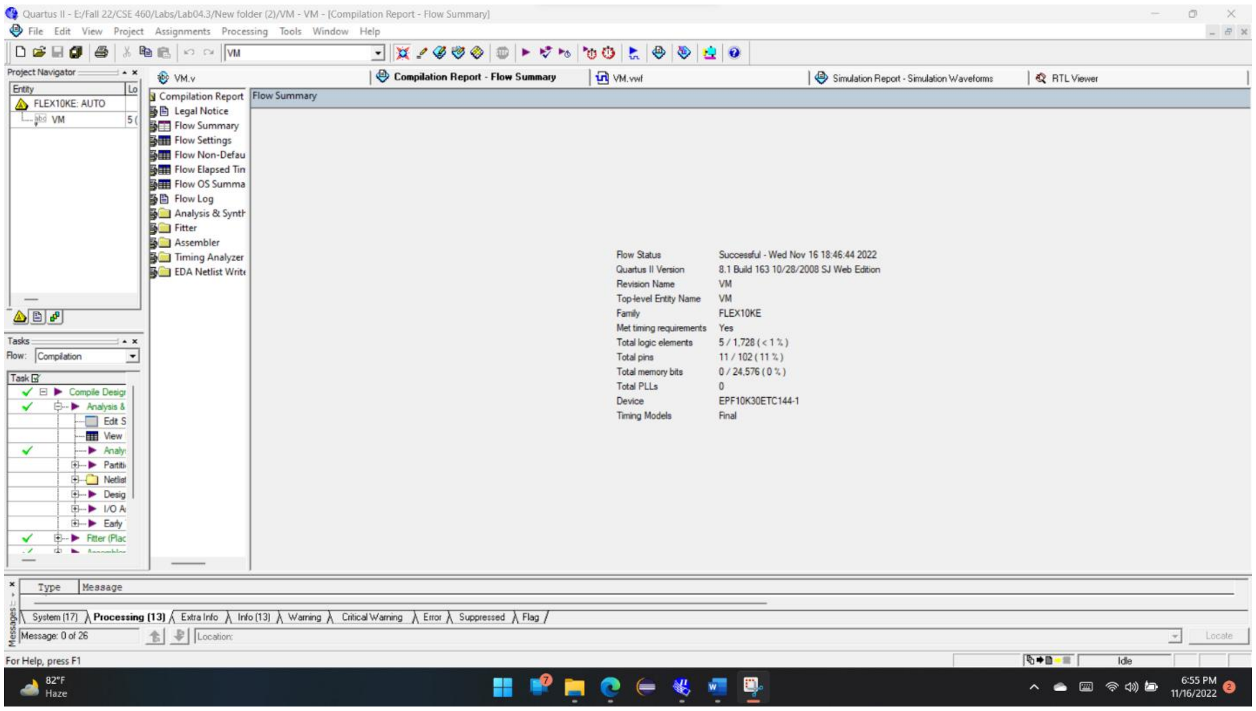
Code

```
module VM(clk, reset, cash_in, cash_return, purchase, current_state, next_state);


        input clk, reset;

        input [1:0] cash_in;


        output reg purchase;

        output reg [1:0] cash_return;

        output reg [1:0]current_state, next_state;


        parameter s0 = 2'b00, s1 = 2'b01;

        parameter [1:0] in_0tk = 2'b00,

                                        in_10tk = 2'b01,

                                        in_20tk = 2'b10;
```

```
parameter [1:0] ret_0tk = 2'b00,
                        ret_5tk = 2'b01,
                        ret_15tk = 2'b10;



always @(posedge clk, posedge reset)
begin
        if (reset == 1)
        begin
                next_state = s0;
                current_state = s0;
                purchase = 0;
                cash_return = ret_0tk;
        end
        else
        begin
                current_state = next_state;
                case(current_state)
                s0:
                begin

                        if(cash_in == in_0tk)
                        begin
                                next_state =s0;
                                purchase = 0;
                                cash_return = ret_0tk;
                        end
                        else if(cash_in == in_10tk)
                        begin
                                next_state =s1;
                                purchase = 0;
                                cash_return = ret_0tk;
                        end

                        else if (cash_in == in_20tk)
                        begin
                                next_state =s0;
                                purchase = 1;
                                cash_return = ret_5tk;
                        end

                end

                s1:
                begin

                        if(cash_in == in_0tk)
                        begin
                                next_state =s0;
                                purchase = 0;
                                cash_return = ret_0tk;
                        end

                        else if(cash_in == in_10tk)
```

```
                begin
                    next_state =s0;
                    purchase = 1;
                    cash_return = ret_5tk;
                end


                else if (cash_in == in_20tk)
                begin
                    next_state =s0;
                    purchase = 1;
                    cash_return = ret_15tk;
                end
            end

            endcase

        end
    end

endmodule
```

Compilation Report

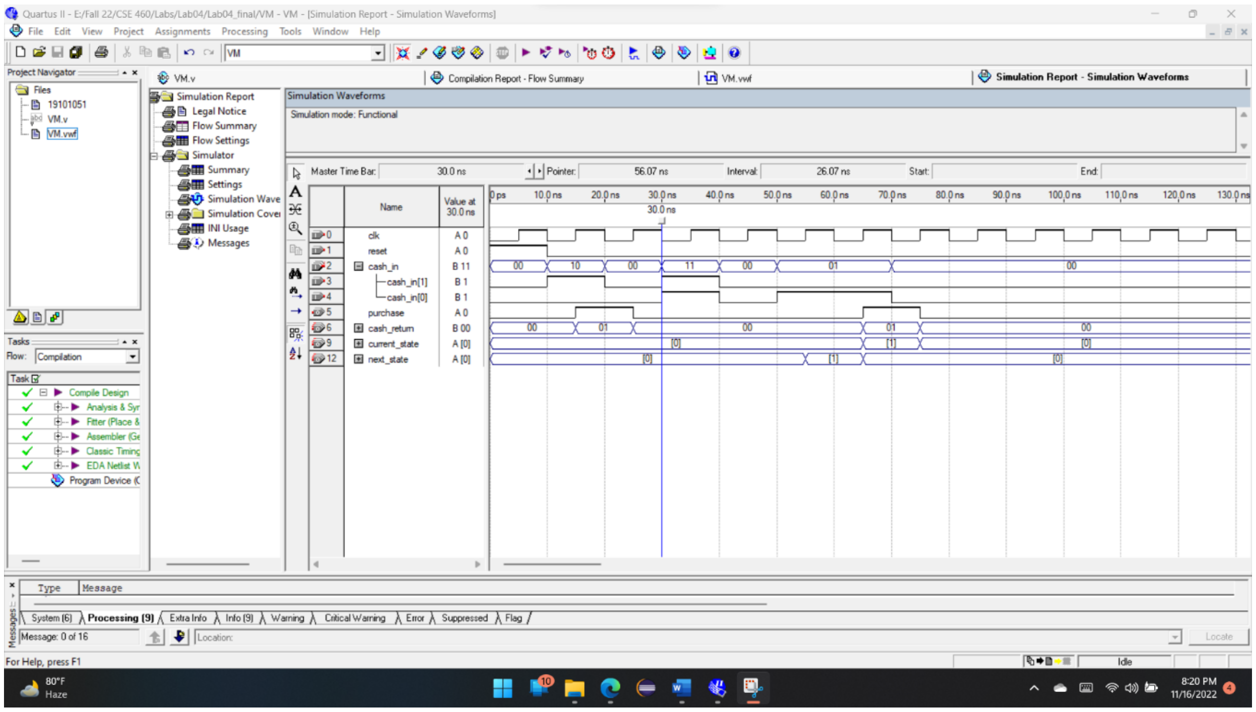


Quartus II - E:/Fall 22/CSE 460/Labs/Lab04.3/New folder (2)/VM - VM - [Compilation Report - Flow Summary]

Flow Summary

Flow Status	Successful - Wed Nov 16 18:46:44 2022
Quartus II Version	8.1 Build 163 10/28/2008 SJ Web Edition
Revision Name	VM
Top-level Entity Name	VM
Family	FLEX10KE
Met timing requirements	Yes
Total logic elements	5 / 1,728 (< 1 %)
Total pins	11 / 102 (11 %)
Total memory bits	0 / 24,576 (0 %)
Total PLLs	0
Device	EPF10K30ETC144-1
Timing Models	Final

Simulation Report



Quartus II - E:/Fall 22/CSE 460/Labs/Lab04/Lab04_final/VM - VM - [Simulation Report - Simulation Waveforms]

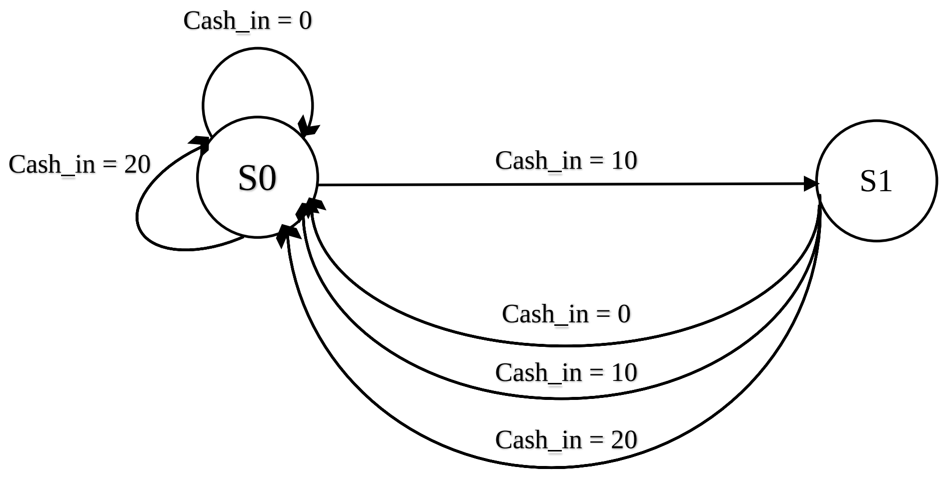
Simulation Waveforms

Simulation mode: Functional

Master Time Bar: 30.0 ns   Pointer: 56.07 ns   Interval: 26.07 ns

Name	Value at 30.0 ns
clk	A 0
reset	A 0
cash_in	B 11
cash_in[1]	B 1
cash_in[0]	B 1
purchase	A 0
cash_return	B 00
current_state	A [0]
next_state	A [0]

State Diagram



State Assign Table

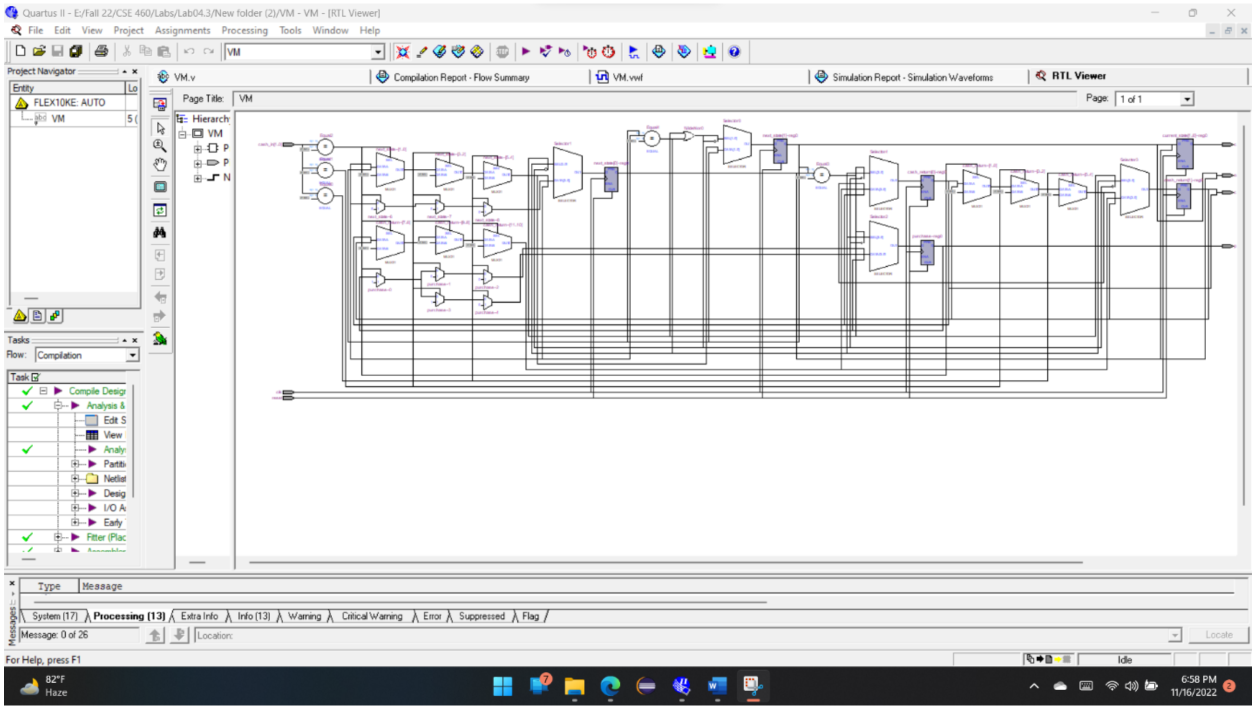
At state S0

Cash_in	Next_state	Purchase	Cash_returen
0	S0	0	0
10	S1	0	0
20	S0	1	5

At state S1

Cash_in	Next_state	Purchase	Cash_returen
0	S0	0	0
10	S0	1	5
20	S0	1	15

RTL View

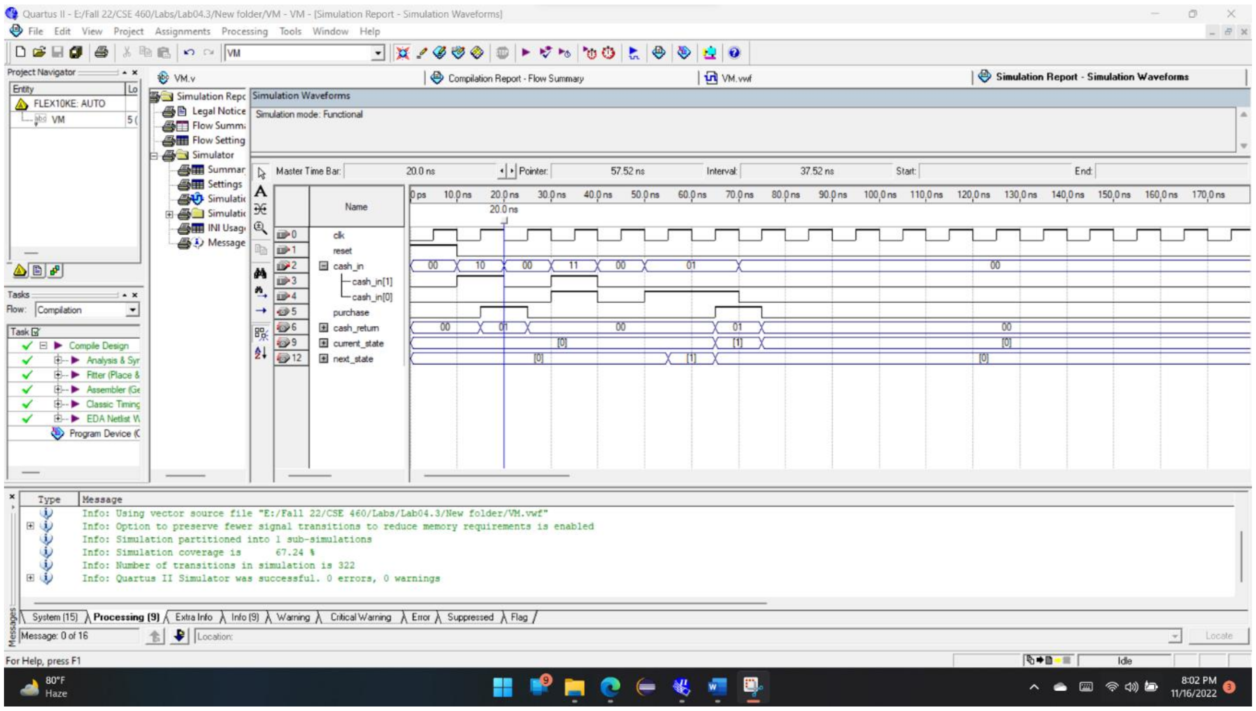

Quartus II - E:/Fall 22/CSE 460/Labs/Lab04.3/New folder (2)/VM - VM - [RTL Viewer]
File Edit View Project Assignments Processing Tools Window Help
Project Navigator
FLEX10KE: AUTO
VM
VM.v
Compilation Report - Flow Summary
VM.vwf
Simulation Report - Simulation Waveforms
RTL Viewer
Page Title: VM
Page: 1 of 1
Tasks
Flow: Compilation
System (17) Processing (13) Extra Info Info (13) Warning Critical Warning Error Suppressed Flag
Message: 0 of 26
For Help, press F1
Idle
82°F Haze
6:58 PM 11/16/2022

Explanation

In this task, I've simulated a Vending Machine, which accepts three inputs, no money (Cash_in = 00), Tk 10(Cash_in = 01), and Tk 20 (Cash_in = 10). The vending machine will purchase a product (cost 15Tk) only if the input is >= 20 Tk. Once this condition meets then the machine generates an output (buy/purchase = 1), then return back to the initial state and gives back the change if needed. In this case I've assigned the inputs, 0Tk as 00, 10 Tk as 01 and 20 Tk as 10. Also, the changes assigned 0Tk as 00, 5Tk as 01 and 15Tk as 10.

Now, we have two states here which are S0 and S1. If we analyze the whole scenario, when our input is 0tk in state S0, the machine remains in the initial state and the purchase is 0 also cash_back is 0. Then if we give 10tk input in state S0 it will go to the next state which is S1 but still purchase will be 0 and cash_back will be 0 as, our product value is 15tk and our input is still less than 20tk. After that if we give 20tk as an input in state S0, it will remain in the present state S0 and as our condition now meets, purchase will be 1 and there will be cash_back of 5tk.

Now, in the next state (S1) if we give 0tk as an input it will move back to the previous state(S0) also purchase will be 0 and there will be no cash_back. Again, if we give 10tk as an input in S1 there will be 1 purchase as we have 10tk previously which means now we have 20tk in S2. So, purchase will be 1 and also, we get cash_back of 5tk. Lastly when we give 20tk as an input in S1, we have previously 10tk in S2, now it becomes 30tk. So, purchase will happen and we will get 15tk as cash_back.



```
Info: Using vector source file "E:/Fall 22/CSE 460/Labs/Lab04.3/New folder/VM.vwf"
Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
Info: Simulation partitioned into 1 sub-simulations
Info: Simulation coverage is      67.24 %
Info: Number of transitions in simulation is 322
Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

If we look at the simulation report, 10ns to 20ns input is 10 which means 20tk and it meets our condition (input>=20) so we get the output(purchase) high in this part. Also, we get cash_bask 01 which was denoted for 5tk and it works perfectly.