CMPE 260

Laboratory Exercise 4: Vending Machine

Course section: CMPE26001L1.2165
Name: Hrishikesh Nitin Moholkar
Lecture instructor: Cliver, Richard
Teaching assistant: Grigoryev, Iosif
Date: 4/19/2017

Abstract

The lab deals with the use of finite state machine to develop a vending machine controller. Moore's machine is used in this lab. The vending machine comprises of coin controller and seven segment decoder. The coin controller deals with processes like accepting the coins, showing balance, dropping soda, showing error etc. The seven segment decoder shows the balance of the user. The working of the vending machine is verified by running the behavioral as well as post-route simulation. The VHDL program is then loaded into Nexys-3 board and tested. The results obtained are accurate and the lab is a success.

Design Methodology

The vending machine consists of two components:

1) Coin controller
2) Seven segment decoder

The state diagram for coin controller is shown below:

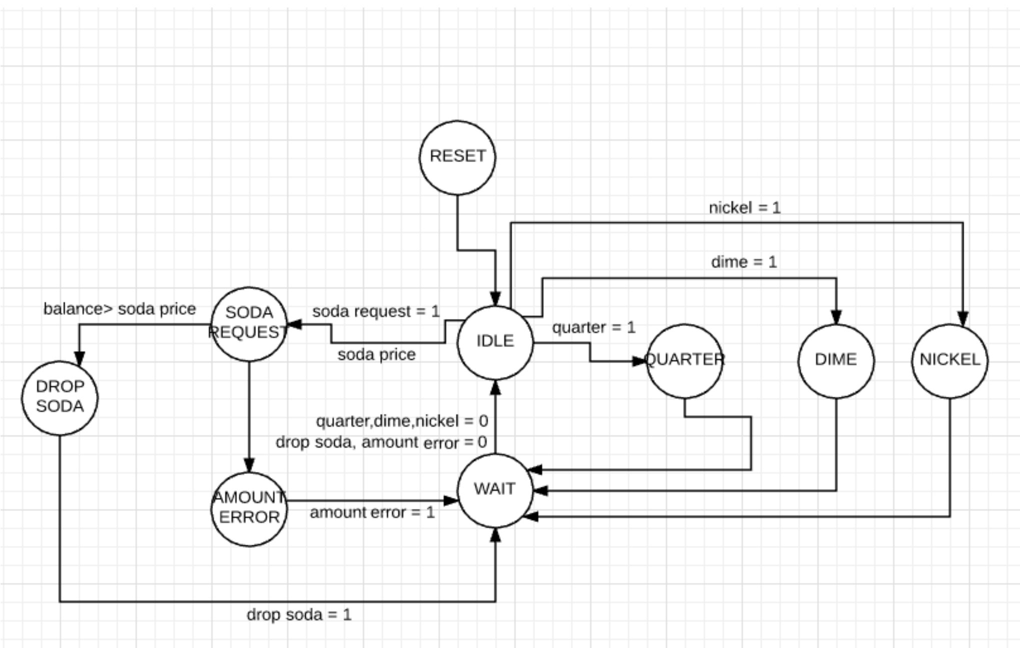

Figure 1.0: Coin controller state diagram

As per the state diagram, the coin controller consists of the RESET state, IDLE state, WAIT state, QUARTER state, DIME state, NICKEL state, SODA REQUEST state, DROP SODA state and AMOUNT ERROR state.
The vending machine has active low reset. When RESET state is called then every signal such as soda request signal, drop soda signal, quarter signal, dime signal, nickel signal, amount error signal and balance signal are all set to zero. Then, the vending machine goes into RESET state. When reset is one, the coin controller goes into IDLE state. If soda is requested, the machine gives an error as the price of soda is greater than the balance. Once quarter, dime or nickel is added, it goes into the QUARTER, DIME or NICKEL state. The machine doesn't accept quarter, dime or nickel at the same time. After that it goes into WAIT state. It remains in WAIT state if there is quarter, nickel or dime put into the machine otherwise it goes back to IDLE state. In order to request a soda, firstly soda price is entered then the soda is requested. If

the balance is sufficient then the vending machine goes into DROP SODA state and drops the soda. Then it goes to WAIT state and finally to IDLE state.

The coin controller has the following inputs:
Clock, reset, QP, DP, NP, soda request, soda price. Here, soda price is 4 bits long whereas the rest of them are 1 bit.
The coin controller has the following outputs:
Drop soda, amount deposited, amount error. Here all outputs are 1 bit long except amount deposit. Amount deposit is 12 bits long.

Seven segment decoder:
Seven segment decoder takes the output from coin controller and converts it into signals to light up the led display.
Seven segment decoder consists of the following inputs:
12 bits input which represents amount deposit.
The outputs are as follows:
Hundred display: 7 bits long.
Tens display: 7 bits long.
Ones display: 7 bits long.
Here the outputs are generated by splitting the inputs as follows:
Hundred display: 11 downto 8 of seven segment input
Tens display: 7 downto 4 of seven segment input
Ones display: 3 downto 0 of seven segment input

Temporary signals are used to store the split inputs. These temporary signals which are 4 bits long, are then converted into seven bits.

Result

The vending machine controller is tested for the following different cases:

1) Soda requested at Insufficient balance
2) Soda requested at the exact balance
3) Vending machine in reset state
4) Soda requested at reserved price.

Behavioral simulation of vending machine controller:

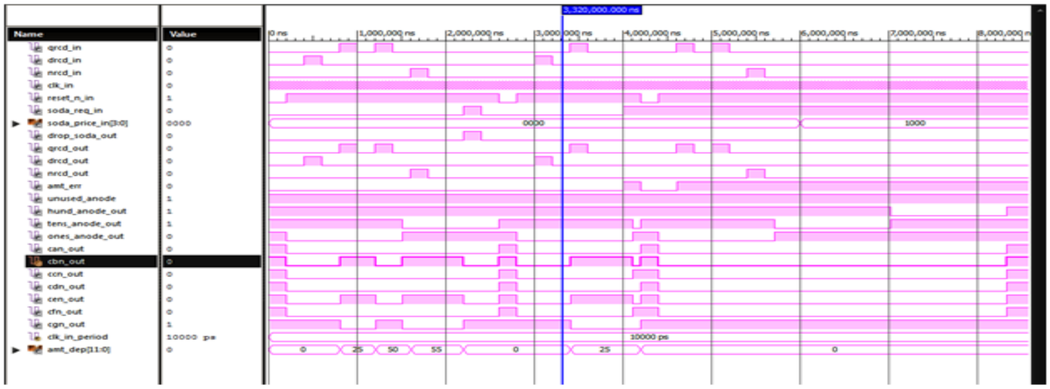

Figure 1.1: Behavioral Simulation of vending machine controller

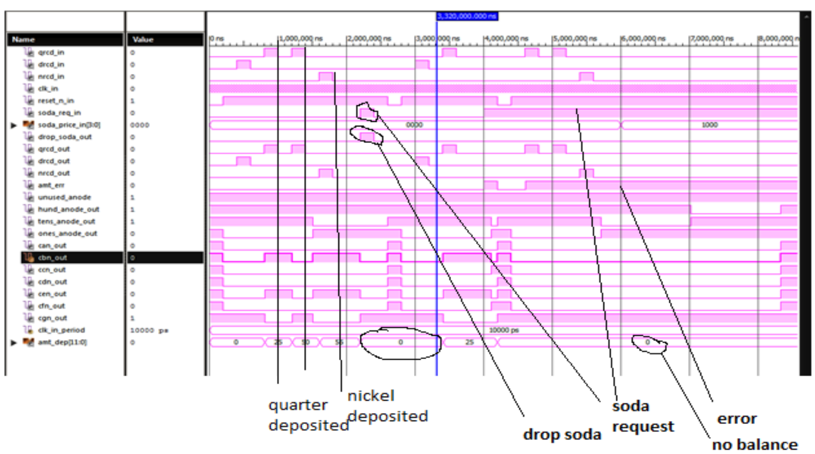

Figure 1.2: Marked Simulation of vending machine controller

The above behavioral simulation is run for 2 milliseconds. When a quarter is added, the amount deposited is shown in amount deposit signal. Similarly, another quarter and nickel is added. At this time, the balance is 55 and a soda is requested. The price entered has the code “0000”. The price corresponding to this code is 55 cents. So, drop soda turns 1 and there is no error. The changes are made to the balance. At this point there is no money in balance. So when soda is requested, it will show an error and won’t drop the soda.

Post-route simulation of vending machine controller:

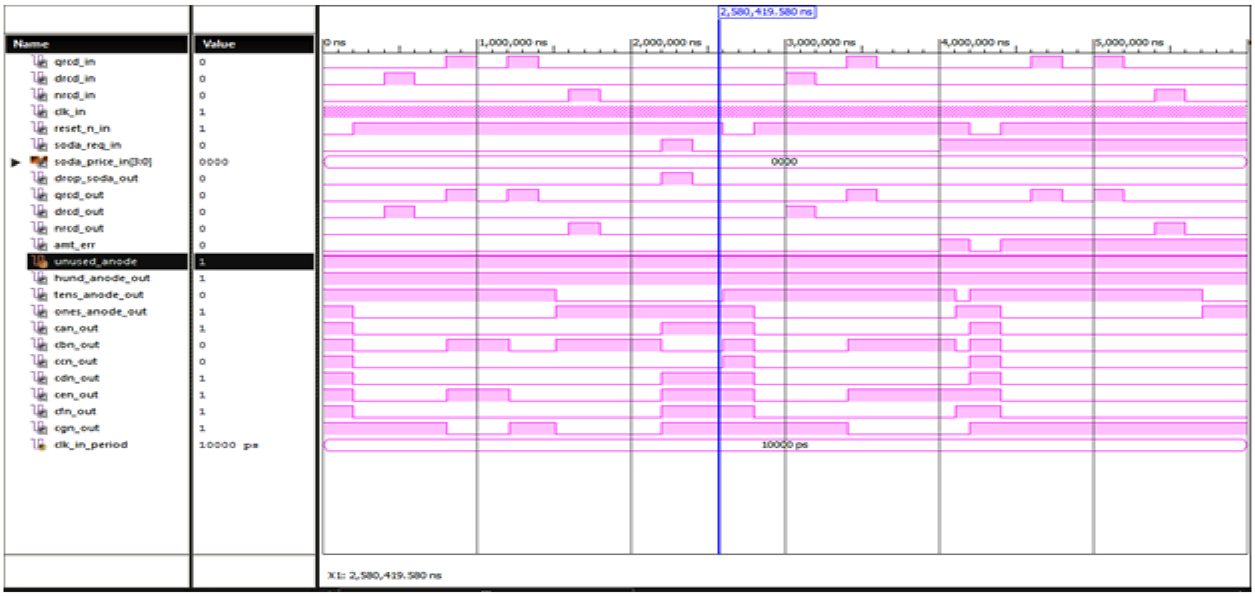

Figure 1.3: Post route for vending machine controller

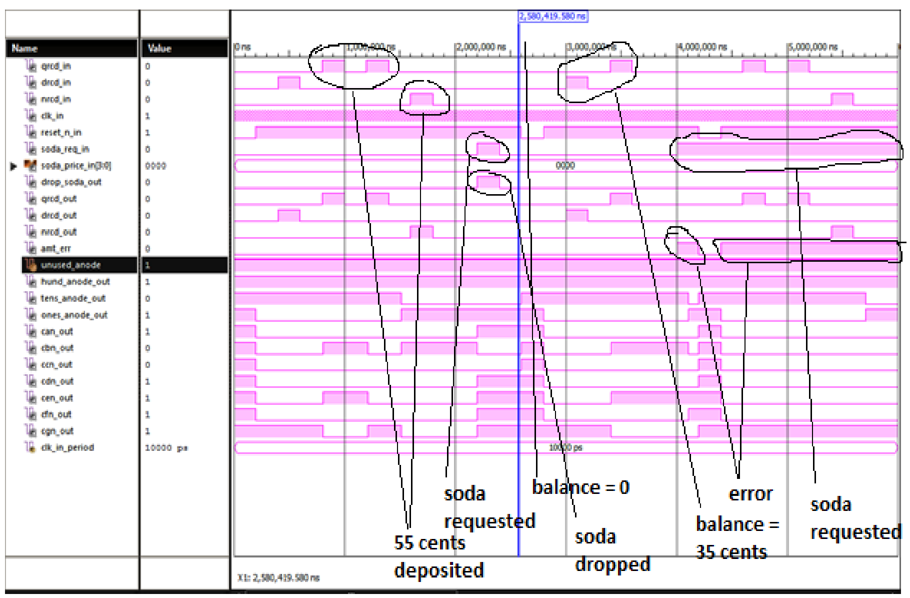

Figure 1.4: Post route marked simulation result

As per the above post-route simulation, 55 cents are deposited into the vending machine. Then soda of price 55 cents is requested and soda is dropped. Now the balance becomes zero. When 35 cents are added to the balance and soda is requested, it will give an error due to insufficient balance. When the vending machine is set to reset mode, the amount error as well as soda request are set to zero. When there is more money in the balance, then it drops soda as per the requested soda. For the soda price of code "1111" it gives an error.

Timing results:

As per the usr_interaction.par file, the timing results are as follows:

```
----------------------------------------------------------------------------------------------------------
  Constraint                                | Check        | Worst Case | Best Case | Timing |   Timing
                                            |              |   Slack    | Achievable | Errors |    Score
----------------------------------------------------------------------------------------------------------
  Autotimespec constraint for clock net Clk | SETUP        |        N/A|    6.717ns|     N/A|           0
  _in_BUFGP                                 | HOLD         |    0.385ns|           |       0|           0
----------------------------------------------------------------------------------------------------------
```

Figure 1.5: Timing result

The best case achievable for the usr_interaction is 6.717 ns.

The area used, number of occupied slices, flip-flops used and LUTS used are as follows:

```
Device Utilization Summary:

Slice Logic Utilization:
  Number of Slice Registers:                    93 out of  18,224    1%
    Number used as Flip Flops:                  93
    Number used as Latches:                      0
    Number used as Latch-thrus:                  0
    Number used as AND/OR logics:                0
  Number of Slice LUTs:                        166 out of   9,112    1%
    Number used as logic:                      160 out of   9,112    1%
      Number using O6 output only:             110
      Number using O5 output only:              18
      Number using O5 and O6:                   32
      Number used as ROM:                        0
    Number used as Memory:                       0 out of   2,176    0%
    Number used exclusively as route-thrus:      6
      Number with same-slice register load:      5
      Number with same-slice carry load:         1
      Number with other load:                    0

Slice Logic Distribution:
  Number of occupied Slices:                    59 out of   2,278    2%
  Number of MUXCYs used:                        52 out of   4,556    1%
  Number of LUT Flip Flop pairs used:          176
    Number with an unused Flip Flop:            91 out of     176   51%
    Number with an unused LUT:                  10 out of     176    5%
    Number of fully used LUT-FF pairs:          75 out of     176   42%
    Number of slice register sites lost
      to control set restrictions:               0 out of  18,224    0%
```

Figure 1.6: Device design summary

Slice registers occupied are 93 out of 18,224. Flip flop used are 93. LUT used are 166 out of 9,112.

Conclusion:

The behavioral simulation, post-route simulation and the Nexys-3 board simulation all had the correct expected results. In case of the Nexys-3 board simulation, the inputs and outputs of the vending machine were connected to different ports of the FPGA. Then the VHDL program was loaded on the Nexys-3 board. When tested for different cases, as specified in results section the board gave correct results. The overall lab is a success.

Sign off sheet: