Lab 3 Project Report

Ethan Keshishian

CS M152A Lab1

Introduction:

The goal of Lab 3 is to create a vending machine by designing a finite state machine. This is done with Verilog and the Xilinx software, which helps write the code as well as generate the waveforms. A finite state machine is a machine that is always in a particular state. There are two types of finite state machines: Moore machines, where the next state only depends on the current state, and Mealy machines, where the next state depends on the current state and the input. This design has a module called vending_machine at the top level, and multiple parameters to define states. These states are handled in an always block with a case. This allows logic for each state to remain separate. Some states include S_IDLE, S_RESET, S_RELOAD, S_FIRST_DIGIT, and more. In total, this implementation uses 21 states. This is in line with the example design given in the lab document. The included testbench, testbench_105422235.v, handles multiple potential use cases for the vending machine, preventing the implementation from being buggy, crashing, or otherwise incorrectly handling input. My final implementation includes waveforms from these test cases.

Module design:

1. States:
    a. S_RESET:
        - Resets all values to their defaults.
        - Refills the vending machine.
        - Remains in reset as long as RESET == 1.
    b. S_RELOAD:
        - Refills the vending machine.
        - Returns to S_IDLE if RELOAD is no longer 1.
    c. S_IDLE:
        - If RESET ==1, goes to S_RESET.
        - If RELOAD ==1, goes to S_RELOAD .
        - Otherwise, waits until CARD_IN is 1 to move to S_FIRST_DIGIT.
    d. S_FIRST_DIGIT:
        - Checks if KEY_PRESS ==1. If it is.
            - Reads from ITEM_CODE and check validity. If valid, move onto S_SECOND_DIGIT. Set internal cost accordingly.
            - If not valid, go to S_INVALID.
        - If KEY_PRESS isn’t 0, proceed to FD_0.
    e. FD_0, FD_2-4:

- 4 states for handling time between inputs.
- Implements the same checks as S_FIRST_DIGIT. Reads from ITEM_CODE and check validity.
- When KEY_PRESS is 1:
  - If valid, move to S_SECOND_DIGIT. Set internal cost accordingly.
  - If invalid, go to S_INVALID.
- Otherwise, proceed to the next state in the line.
- If at the last state, FD_4, and KEY_PRESS still isn't 1, go to S_INVALID.

f. S_SECOND_DIGIT:
- Checks if KEY_PRESS ==1. If it is:
  - Reads from ITEM_CODE and check validity. If valid, move onto S_CARD_VALIDATION and set COST.
  - If not valid, go to S_INVALID.
- If KEY_PRESS isn't 0, proceed to SD_0.

g. SD_1-4:
- 4 states for handling time between inputs
- Implements the same checks as S_SECOND _DIGIT. Reads from ITEM_CODE and check validity.
- If KEY_PRESS is 1:
  - If ITEM_CODE is valid, move to S_CARD_VALIDATION and set COST.
  - If invalid, go to S_INVALID.
- Otherwise, proceed to the next state in the line.
- If at the last state, SD_4, and KEY_PRESS still isn't 1, go to S_INVALID.

h. S_CARD_VALIDATION:
- Checks for the VALID_TRAN signal.
  - If VALID_TRAN is 1, decrements the appropriate counter if possible and moves to S_SUCCESS.
  - If the counter is already empty, go to S_INVALID.
- If VALID_TRAN isn't 1, proceed to CV_1.

i. CV_1-4:
- 4 states for handling time until card validation.
- Implements the same checks as S_CARD_VALIDATION. Checks for the VALID_TRAN signal.
  - If VALID_TRAN is 1, move to decrements the appropriate counter if possible and moves to S_SUCCESS.
    - If the counter is already 0, then proceed to S_INVALID.
  - Otherwise, proceed to the next state in the line.
- If at the last state, CV_4, and VALID still isn't 1, go to S_FAILURE.

j. S_SUCCESS:
- If here, the transaction succeeded.
- VEND is set to 1, then return to S_IDLE.

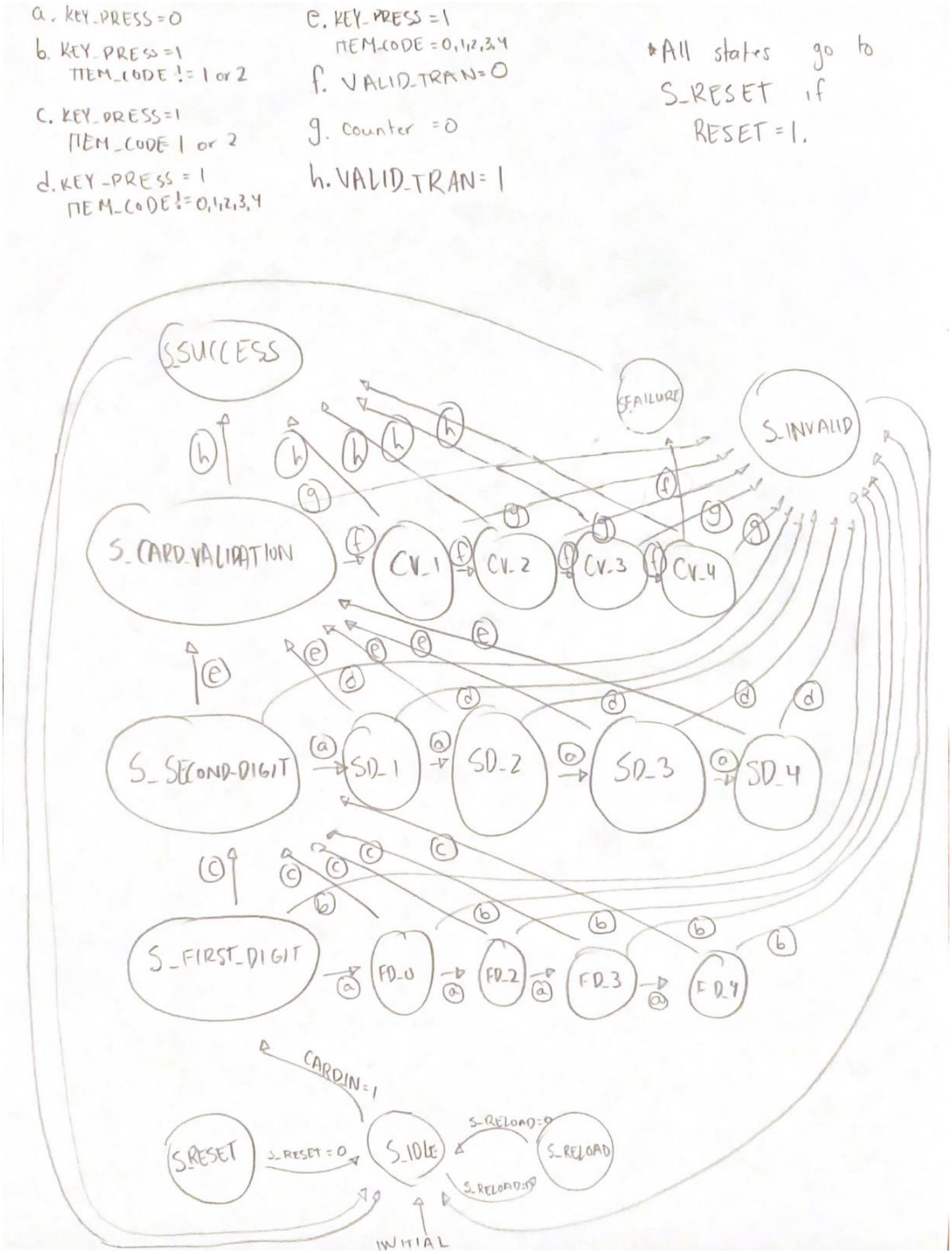
k. S_INVALID:

- If here, there was an error, according to the spec.
- Set INVALID_SEL to 1, then return to S_IDLE.

l. S_FAILURE:

- If here, card validation took too long, as described by the spec.
- Set FAILED_TRAN to 1, then return to S_IDLE.

2. FSM Diagram (States and transitions):

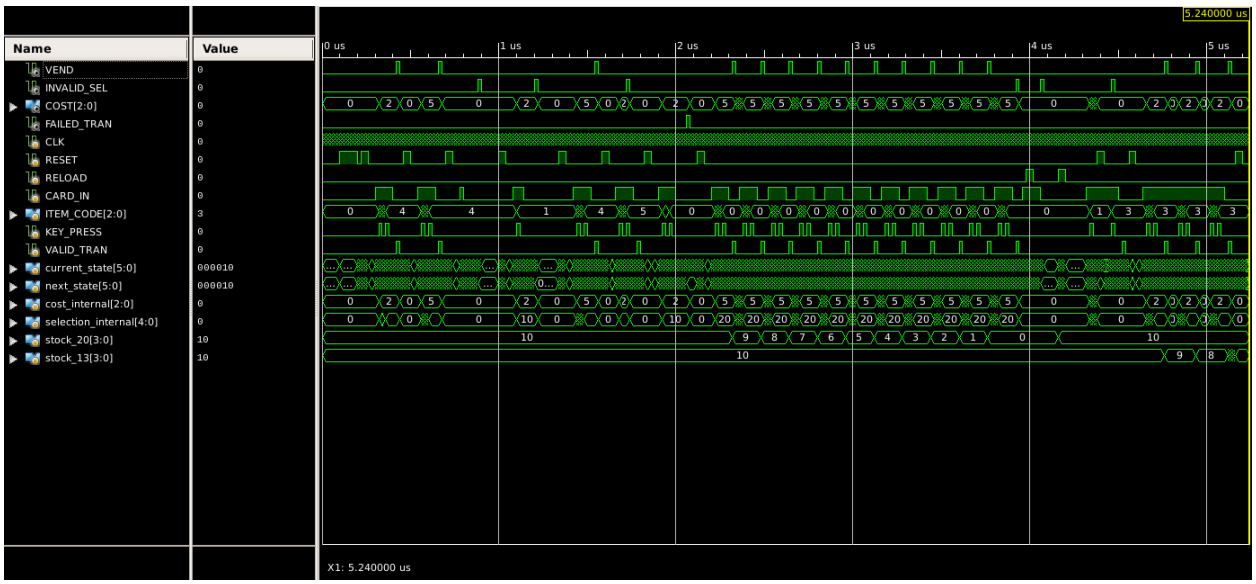


3. Testbench Design:

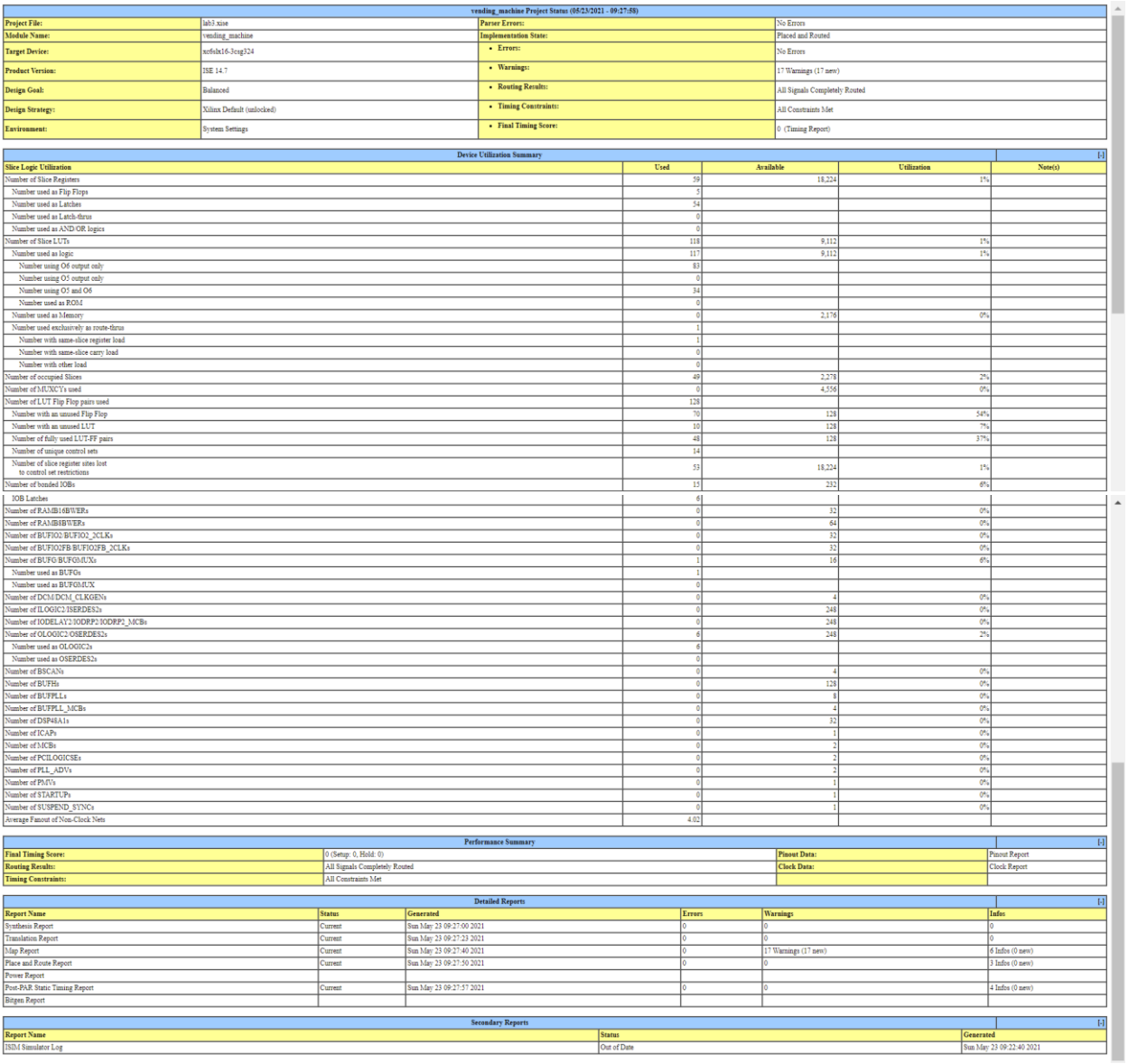
My testbench handles multiple cases. Here are the implemented cases:

a. Successful transaction that begins with the digit 1.
    - This is the case where a cheaper purchase is made, and COST is set to $2.
b. Successful transaction that begins with the digit 2.
    - This is the case where a pricier purchase is made, and COST is set to $5.
c. It took too long to press the first digit.
    - Occurs when the FD_4 state is reached without KEY_PRESS beings set to 1. The INVALID_SEL flag should be set.
d. It took too long to press the second digit.
    - Occurs when the SD_4 state is reached without KEY_PRESS beings set to 1. The INVALID_SEL flag should be set.
e. Bad input for first digit.
    - In this case, a digit is inputted that is not equal to 1 or 2. The INVALID_SEL output should be set to 1.
f. Bad input for second digit.
    - In this case, a digit is inputted that is not between 0 and 4. The INVALID_SEL output should be set to 1.
g. Card is declined.
    - In this case, the input is correct, but VALID_TRAN is never set. The timer should run out, and the FAILED_TRAN should be set to 1.
h. Finishing an item from a vending machine.
    - In this case, the same item is purchased 10 times. It accurately reflects this in the counter.
    - An eleventh purchase is made, but in this case, there is no inventory left, and the INVALID_SEL output should be set to 1. VEND should not be set to 1.
i. Attempting to reload when not in S_IDLE.
    - In this case, a reload is attempted when not in idle.
    - RELOAD is set to 1 while the current state is S_FIRST_DIGIT, and nothing should happen.
j. Successful reload.
    - In this case, RELOAD is set to 1 while the current state is S_IDLE. The reload should reset the counter for the empty item from the previous case.
k. Resetting halfway through a transaction.
    - In this case, the machine is reset in the middle of the transaction, and the transaction does not complete.
l. Multiple purchases without removing card.
    - In this case, after each transaction completes, S_IDLE is quickly exited and the same card is used for another transaction. Expected behavior is to have VEND be set to 1 for each transaction.

4. Waveforms:


Name
Value
VEND
INVALID_SEL
COST[2:0]
FAILED_TRAN
CLK
RESET
RELOAD
CARD_IN
ITEM_CODE[2:0]
KEY_PRESS
VALID_TRAN
current_state[5:0]
next_state[5:0]
cost_internal[2:0]
selection_internal[4:0]
stock_20[3:0]
stock_13[3:0]
000010
000010
0 us
1 us
2 us
3 us
4 us
5 us
5.240000 us
X1: 5.240000 us

5. Design Summary



vending_machine Project Status (05/23/2021 - 09:27:58)			
Project File:	lab3.xise	**Parser Errors:**	No Errors
Module Name:	vending_machine	**Implementation State:**	Placed and Routed
Target Device:	xc6slx16-3csg324	• **Errors:**	No Errors
Product Version:	ISE 14.7	• **Warnings:**	17 Warnings (17 new)
Design Goal:	Balanced	• **Routing Results:**	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	• **Timing Constraints:**	All Constraints Met
Environment:	System Settings	• **Final Timing Score:**	0 (Timing Report)

Device Utilization Summary				[-]
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	59	18,224	1%	
Number used as Flip Flops	5			
Number used as Latches	54			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	118	9,112	1%	
Number used as logic	117	9,112	1%	
Number using O6 output only	83			
Number using O5 output only	0			
Number using O5 and O6	34			
Number used as ROM	0			
Number used as Memory	0	2,176	0%	
Number used exclusively as route-thrus	1			
Number with same-slice register load	1			
Number with same-slice carry load	0			
Number with other load	0			
Number of occupied Slices	49	2,278	2%	
Number of MUXCYs used	0	4,556	0%	
Number of LUT Flip Flop pairs used	128			
Number with an unused Flip Flop	70	128	54%	
Number with an unused LUT	10	128	7%	
Number of fully used LUT-FF pairs	48	128	37%	
Number of unique control sets	14			
Number of slice register sites lost to control set restrictions	53	18,224	1%	
Number of bonded IOBs	15	232	6%	
IOB Latches	6			
Number of RAMB16BWERs	0	32	0%	
Number of RAMB8BWERs	0	64	0%	
Number of BUFIO2/BUFIO2_2CLKs	0	32	0%	
Number of BUFIO2FB/BUFIO2FB_2CLKs	0	32	0%	
Number of BUFG/BUFGMUXs	1	16	6%	
Number used as BUFGs	1			
Number used as BUFGMUX	0			
Number of DCM/DCM_CLKGENs	0	4	0%	
Number of ILOGIC2/ISERDES2s	0	248	0%	
Number of IODELAY2/IODRP2/IODRP2_MCBs	0	248	0%	
Number of OLOGIC2/OSERDES2s	6	248	2%	
Number used as OLOGIC2s	6			
Number used as OSERDES2s	0			
Number of BSCANs	0	4	0%	
Number of BUFHs	0	128	0%	
Number of BUFPLLs	0	8	0%	
Number of BUFPLL_MCBs	0	4	0%	
Number of DSP48A1s	0	32	0%	
Number of ICAPs	0	1	0%	
Number of MCBs	0	2	0%	
Number of PCILOGICSEs	0	2	0%	
Number of PLL_ADVs	0	2	0%	
Number of PMVs	0	1	0%	
Number of STARTUPs	0	1	0%	
Number of SUSPEND_SYNCs	0	1	0%	
Average Fanout of Non-Clock Nets	4.02			

Performance Summary				[-]
Final Timing Score:	0 (Setup: 0, Hold: 0)	**Pinout Data:**	Pinout Report	
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report	
Timing Constraints:	All Constraints Met			

Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Sun May 23 09:27:00 2021	0	0	0
Translation Report	Current	Sun May 23 09:27:23 2021	0	0	0
Map Report	Current	Sun May 23 09:27:40 2021	0	17 Warnings (17 new)	6 Infos (0 new)
Place and Route Report	Current	Sun May 23 09:27:50 2021	0	0	3 Infos (0 new)
Power Report					
Post-PAR Static Timing Report	Current	Sun May 23 09:27:57 2021	0	0	4 Infos (0 new)
Bitgen Report					

Secondary Reports			[-]
Report Name	**Status**	**Generated**	
ISIM Simulator Log	Out of Date	Sun May 23 09:22:40 2021	

6. Map Summary:

```
Map Report


Sun May 23 10:55:32 2021



Release 14.7 Map P.20131013 (lin64)

Xilinx Mapping Report File for Design 'vending_machine'
```

```
Design Information
------------------
Command Line   : map -intstyle ise -p xc6slx16-csg324-3 -w -logic_opt off
-ol
high -t 1 -xt 0 -register_duplication off -r 4 -global_opt off -mt off -ir
off
-pr off -lc off -power off -o vending_machine_map.ncd vending_machine.ngd
vending_machine.pcf
Target Device  : xc6slx16
Target Package : csg324
Target Speed   : -3
Mapper Version : spartan6 -- $Revision: 1.55 $
Mapped Date    : Sun May 23 10:53:43 2021

Design Summary
--------------
Number of errors:      0
Number of warnings:   17
Slice Logic Utilization:
  Number of Slice Registers:                    59 out of  18,224    1%
    Number used as Flip Flops:                   5
    Number used as Latches:                     54
    Number used as Latch-thrus:                  0
    Number used as AND/OR logics:                0
  Number of Slice LUTs:                        118 out of   9,112    1%
    Number used as logic:                      118 out of   9,112    1%
      Number using O6 output only:              84
      Number using O5 output only:               0
      Number using O5 and O6:                   34
      Number used as ROM:                        0
    Number used as Memory:                       0 out of   2,176    0%

Slice Logic Distribution:
```

Number of occupied Slices:	45	out of	2,278	1%
Number of MUXCYs used:	0	out of	4,556	0%
Number of LUT Flip Flop pairs used:	128			
Number with an unused Flip Flop:	69	out of	128	53%
Number with an unused LUT:	10	out of	128	7%
Number of fully used LUT-FF pairs:	49	out of	128	38%
Number of unique control sets:	15			
Number of slice register sites lost to control set restrictions:	61	out of	18,224	1%

A LUT Flip Flop pair for this architecture represents one LUT paired with one Flip Flop within a slice. A control set is a unique combination of clock, reset, set, and enable signals for a registered element.
The Slice Logic Distribution report is not meaningful if the design is over-mapped for a non-slice resource or if Placement fails.

IO Utilization:

Number of bonded IOBs:	15	out of	232	6%
IOB Latches:	6			

Specific Feature Utilization:

Number of RAMB16BWERs:	0	out of	32	0%
Number of RAMB8BWERs:	0	out of	64	0%
Number of BUFIO2/BUFIO2_2CLKs:	0	out of	32	0%
Number of BUFIO2FB/BUFIO2FB_2CLKs:	0	out of	32	0%
Number of BUFG/BUFGMUXs:	1	out of	16	6%
Number used as BUFGs:	1			
Number used as BUFGMUX:	0			
Number of DCM/DCM_CLKGENs:	0	out of	4	0%
Number of ILOGIC2/ISERDES2s:	0	out of	248	0%
Number of IODELAY2/IODRP2/IODRP2_MCBs:	0	out of	248	0%

Number of OLOGIC2/OSERDES2s:	6	out of	248	2%
Number used as OLOGIC2s:	6			
Number used as OSERDES2s:	0			
Number of BSCANs:	0	out of	4	0%
Number of BUFHs:	0	out of	128	0%
Number of BUFPLLs:	0	out of	8	0%
Number of BUFPLL_MCBs:	0	out of	4	0%
Number of DSP48A1s:	0	out of	32	0%
Number of ICAPs:	0	out of	1	0%
Number of MCBs:	0	out of	2	0%
Number of PCILOGICSEs:	0	out of	2	0%
Number of PLL_ADVs:	0	out of	2	0%
Number of PMVs:	0	out of	1	0%
Number of STARTUPs:	0	out of	1	0%
Number of SUSPEND_SYNCs:	0	out of	1	0%

Average Fanout of Non-Clock Nets: 4.02

Peak Memory Usage: 762 MB

Total REAL time to MAP completion: 5 secs

Total CPU time to MAP completion: 5 secs

Table of Contents

Section 9 - Area Group and Partition Summary

Section 10 - Timing Report

Section 11 - Configuration String Information

Section 12 - Control Set Information

Section 13 - Utilization by Hierarchy

Section 1 - Errors

Section 2 - Warnings

WARNING:PhysDesignRules:372 - Gated clock. Clock net

   current_state[5]_PWR_94_o_Select_712_o is sourced by a combinatorial pin.

   This is not good design practice. Use the CE pin to control the loading of

   data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

   current_state[5]_PWR_46_o_Select_616_o is sourced by a combinatorial pin.

   This is not good design practice. Use the CE pin to control the loading of

   data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net _n2384 is sourced by a

   combinatorial pin. This is not good design practice. Use the CE pin to

   control the loading of data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

   current_state[5]_PWR_22_o_Select_568_o is sourced by a combinatorial pin.

   This is not good design practice. Use the CE pin to control the loading of

   data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_142_o_Select_808_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_190_o_Select_904_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_118_o_Select_760_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_70_o_Select_664_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_6_o_Select_536_o is sourced by a combinatorial pin. This

is not good design practice. Use the CE pin to control the loading of data

into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net _n2385 is sourced by a

combinatorial pin. This is not good design practice. Use the CE pin to

control the loading of data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net _n2383 is sourced by a

combinatorial pin. This is not good design practice. Use the CE pin to

control the loading of data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_238_o_Select_1000_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_214_o_Select_952_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_166_o_Select_856_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_13_o_Select_550_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_3_o_Select_530_o is sourced by a combinatorial pin. This

is not good design practice. Use the CE pin to control the loading of data

into the flip-flop.

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_PWR_257_o_Select_1038_o is sourced by a combinatorial pin.

This is not good design practice. Use the CE pin to control the loading of

data into the flip-flop.

Section 3 - Informational

INFO:MapLib:562 - No environment variables are currently set.

INFO:LIT:244 - All of the single ended outputs in this design are using slew

rate limited output drivers. The delay on speed critical single ended outputs

can be dramatically reduced by designating them as fast outputs.

INFO:Pack:1716 - Initializing temperature to 85.000 Celsius. (default - Range:

0.000 to 85.000 Celsius)

INFO:Pack:1720 - Initializing voltage to 1.140 Volts. (default - Range: 1.140 to

1.260 Volts)

INFO:Map:215 - The Interim Design Summary has been generated in the MAP Report

(.mrp).

INFO:Pack:1650 - Map created a placed design.

Section 4 - Removed Logic Summary

Section 5 - Removed Logic

Section 6 - IOB Properties

```
+--------------------------------------------------------------------------------------------------------------------------------------------------------+
| IOB Name                          | Type             | Direction | IO Standard          | Diff  | Drive    | Slew | Reg (s)      | Resistor | IOB      |
|                                   |                  |           |                      | Term  | Strength | Rate |              |          | Delay    |
```

```
+-------------------------------------------------------------------------
-------------------------------------------------------------------------
------+
| CARD_IN                          | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| CLK                              | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| COST<0>                          | IOB              | OUTPUT    |
LVCMOS25             |       | 12       | SLOW | OLATCH       |          |
|
| COST<1>                          | IOB              | OUTPUT    |
LVCMOS25             |       | 12       | SLOW | OLATCH       |          |
|
| COST<2>                          | IOB              | OUTPUT    |
LVCMOS25             |       | 12       | SLOW | OLATCH       |          |
|
| FAILED_TRAN                      | IOB              | OUTPUT    |
LVCMOS25             |       | 12       | SLOW | OLATCH       |          |
|
| INVALID_SEL                      | IOB              | OUTPUT    |
LVCMOS25             |       | 12       | SLOW | OLATCH       |          |
|
| ITEM_CODE<0>                     | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| ITEM_CODE<1>                     | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| ITEM_CODE<2>                     | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| KEY_PRESS                        | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| RELOAD                           | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
| RESET                            | IOB              | INPUT     |
LVCMOS25             |       |          |      |              |          |
|
```

| VALID_TRAN | IOB | INPUT |
LVCMOS25 | | | | | | |
|
| VEND | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

+---

------+

Section 7 - RPMs

Section 8 - Guide Report

Guide not run on this design.

Section 9 - Area Group and Partition Summary

Partition Implementation Status

No Partitions were found in this design.

Area Group Information

No area groups were found in this design.

```
Section 10 - Timing Report

--------------------------

A logic-level (pre-route) timing report can be generated by using Xilinx
static

timing analysis tools, Timing Analyzer (GUI) or TRCE (command line), with
the

mapped NCD and PCF files. Please note that this timing report will be
generated

using estimated delay information. For accurate numbers, please generate a

timing report with the post Place and Route NCD file.


For more information about the Timing Analyzer, consult the Xilinx Timing

Analyzer Reference Manual; for more information about TRCE, consult the
Xilinx

Command Line Tools User Guide "TRACE" chapter.


Section 11 - Configuration String Details

----------------------------------------

Use the "-detail" map option to print out Configuration Strings


Section 12 - Control Set Information

-----------------------------------

Use the "-detail" map option to print out Control Set Information.


Section 13 - Utilization by Hierarchy

-------------------------------------

Use the "-detail" map option to print out the Utilization by Hierarchy
section.
```

Conclusion:

Overall, my implementation uses a top module and states to build the vending machine. It outputs four variables, as described above. The design details indicate that there isn't too many resources being used, but there are a couple warning that might warrant looking into, though they have no effect on the waveforms, which are displaying accurately. There were multiple difficulties I ran into with the project. I initially wanted to implement a counter instead of using multiple states, but this ran into undefined behavior. Using multiple states resulted in simpler and

more readable code. Another thing I learned to use were tasks, since they simplified the code tremendously. Maybe learning about tasks can be added to the course, since I enjoyed using them and found them extremely helpful. Overall, I really enjoyed the project, and found myself learning a lot from overcoming challenges like these.