Lab 4 Project Report

Ethan Keshishian

CS M152A Lab1

Introduction:

The goal of Lab 4 is to create a parking meter 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 parking_meter at the top level, and multiple parameters to define states. These states are handled in two always blocks, each with case statements. This allows logic for each state to remain separate. One always block is used to set the output, while the other is used to set the next state. The states used by the parking meter are S_RESET, S_RESET1, S_RESET2, S_ZERO, S_3MIN, and S_COUNTING. In total, this implementation uses six 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 parking meter, preventing the implementation from being buggy, crashing, or otherwise incorrectly handling input. It handles all edge cases. My final implementation includes waveforms from these test cases. My demonstration covers these test cases, though more investigation can still be done by zooming further in since there are too many to cover in detail.

Module design:

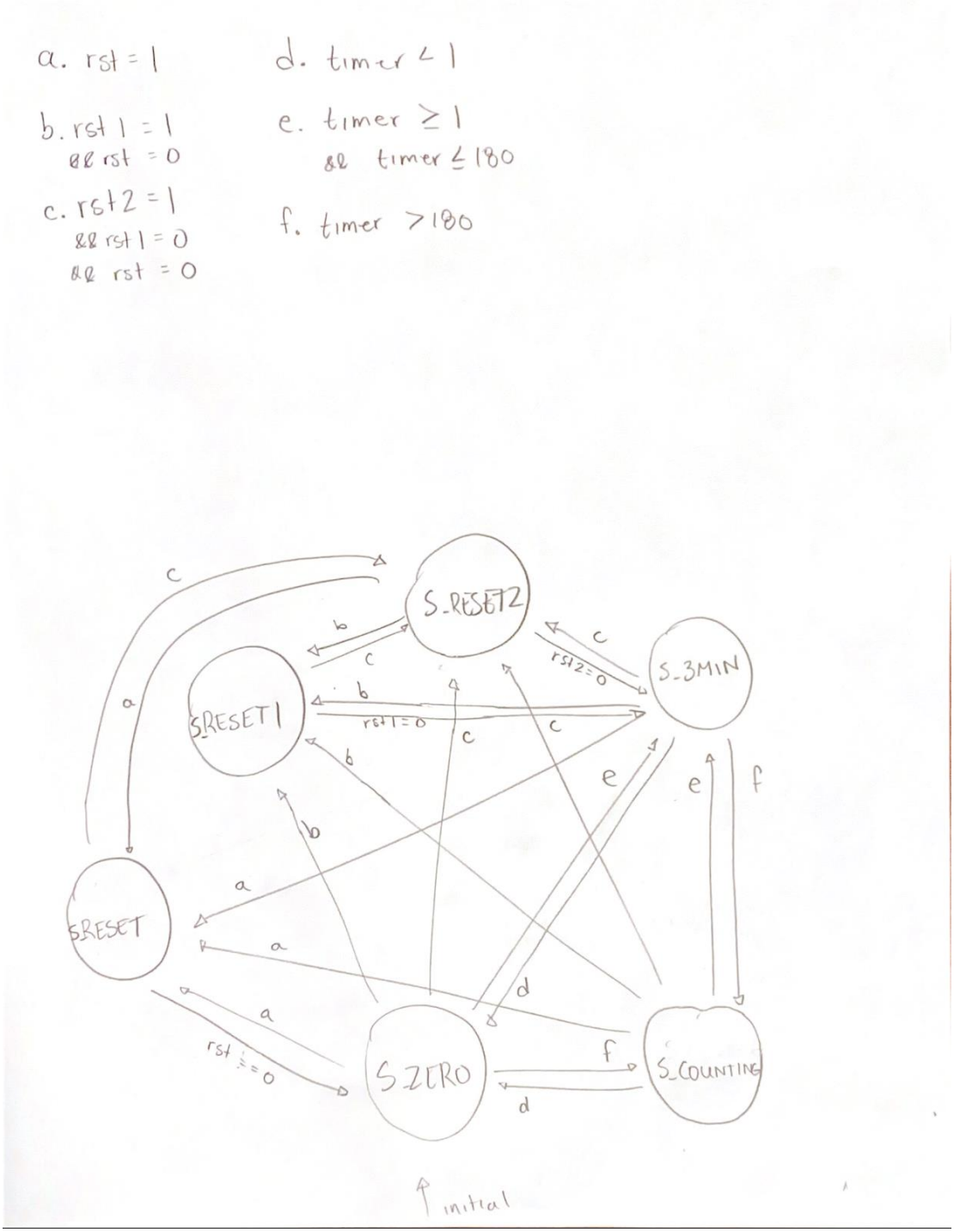
1. States:
    a. S_RESET:
        - Sets the next state to return to the initial state, S_ZERO.
        - Remains in reset as long as RESET == 1.
    b. S_RESET1:
        - Sets the timer to 16 and sets the next state to S_3MIN.
    c. S_RESET2:
        - Sets the timer to 150 and sets the next state to S_3MIN.
    d. S_ZERO:
        - The output is set to 0000. This flashes on for half a second and off for half a second.
        - NOTE: Internal clock should be set to 0 when leaving this state.
        - If add1 is high, then add 60 seconds to the timer. Otherwise, check if add2 is high. If it is, add 120 seconds to the timer. Otherwise, if add3 is high, then add 180 seconds to the timer. Finally, check if add4 is high. If it is, then add 300 seconds

to the timer. In all cases, there is a check to ensure the timer does not exceed 9999. If it does, the value is fixed to 9999.
    - If the timer is less than 1, set the next state equal to S_ZERO. If the timer is greater than or equal to 1 and less than or equal to 180, set the next state equal to S_3MIN. Otherwise, the timer must be greater than 180, so set the next state equal to S_COUNTING.

e. S_3MIN:
    - This is the state when the timer is greater than 1 and less than or equal to 180 seconds.
    - The timer flashes on and off, in such a way that even numbers are displayed to the clock, and odd numbers are blank.
    - If add1 is high, then add 60 seconds to the timer. Otherwise, check if add2 is high. If it is, add 120 seconds to the timer. Otherwise, if add3 is high, then add 180 seconds to the timer. Finally, check if add4 is high. If it is, then add 300 seconds to the timer. In all cases, there is a check to ensure the timer does not exceed 9999. If it does, the value is fixed to 9999.
    - If the timer is less than 1, set the next state equal to S_ZERO. If the timer is greater than or equal to 1 and less than or equal to 180, set the next state equal to S_3MIN. Otherwise, the timer must be greater than 180, so set the next state equal to S_COUNTING.

f. S_COUNTING:
    - This is the state when the timer is greater than 180 seconds.
    - The timer does not flash. Digits are displayed normally.
    - If add1 is high, then add 60 seconds to the timer. Otherwise, check if add2 is high. If it is, add 120 seconds to the timer. Otherwise, if add3 is high, then add 180 seconds to the timer. Finally, check if add4 is high. If it is, then add 300 seconds to the timer. In all cases, there is a check to ensure the timer does not exceed 9999. If it does, the value is fixed to 9999.
    - If the timer is less than 1, set the next state equal to S_ZERO. If the timer is greater than or equal to 1 and less than or equal to 180, set the next state equal to S_3MIN. Otherwise, the timer must be greater than 180, so set the next state equal to S_COUNTING.

2. FSM Diagram (States and transitions):



3. Testbench Design:
   My testbench handles multiple cases. Here are the implemented cases:
   a. Resetting.
      - This is the case where the reset button is pressed. It causes the parking meter to return to S_ZERO and display 0000.
   b. Idle on zero state.

- This is the case where the parking meter displays zero. It idles for a couple seconds to make sure that the flashing functionality is working.

c. Sending the add1 signal.
   - Occurs when the add1 signal is 1. Adds 60 seconds to the timer.

d. Sending the add2 signal.
   - Occurs when the add2 signal is 1. Adds 120 seconds to the timer.

e. Sending the add3 signal.
   - Occurs when the add3 signal is 1. Adds 180 seconds to the timer.

f. Sending the add4 signal.
   - Occurs when the add4 signal is 1. Adds 300 seconds to the timer.

g. Sending the reset1 signal.
   - Occurs when the reset1 signal is 1. Expected behavior is that the timer is set to 16 and the parking meter goes to the S_3MIN state.

h. Sending the reset2 signal.
   - Occurs when the reset2 signal is 1. Expected behavior is that the timer is set to 150 and the parking meter goes to the S_3MIN state.

i. Testing greater than 180 seconds remaining.
   - In this case, time is added to the parking meter so that it has more than 180 seconds remaining. It should go to the S_COUNTING state and count down without flashing.

j. Testing transition from 181 to 179 seconds.
   - In this case, the timer is allowed to run until it has less than 180 seconds remaining. Expected behavior is that the parking meter goes from the S_COUNTING state to the S_3MIN state and begins to flash such that odd numbers are blank.

k. Testing less than 180 seconds remaining.
   - In this case, the timer is set to less than 180.
   - The parking meter is made to idle for a couple seconds, so that the correct flashing behavior where odd numbers are blank can be verified.

l. Transitioning from 1 to 0 seconds.
   - In this case, the timer is reset to 16 seconds remaining, and allowed to count down to 0. Expected behavior is that the parking meter transitions from the S_3MIN state to the S_ZERO state, and the flashing behavior changes.

m. Testing overflow and consecutive add presses.
   - In this case, time is added to the timer consecutively. More than 9999 time is added, and the timer should cap the time at 9999, preventing overflow.

n. Testing reset1 from non-zero value.
   - First, the clock is set to a non-zero number. Then, rst1 is set to 1. Expected behavior is a transition to the S_RESET1 state without issue.

o. Testing reset2 from non-zero value.
   - First, the clock is set to a non-zero number. Then, rst2 is set to 1. Expected behavior is a transition to the S_RESET2 state without issue.

p. Testing adding time in S_3MIN state to flashing number.

- In this case, the timer is set to an odd time less than 180. Then, 60 seconds are added, such that the timer is still less than 180 seconds. The odd value should still be blank.

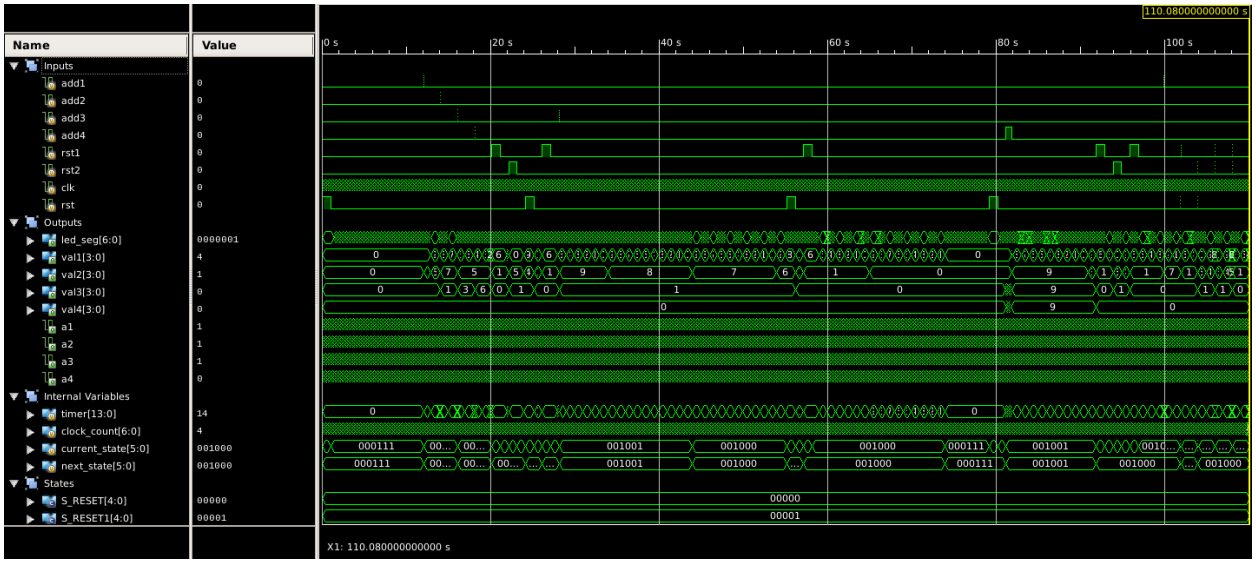
q. Testing rst1 from rst.
- In this case, rst is set to 1. Then, it is set to 0 and rst1 1 is set 1. It should transition between states correctly and indicate the correct value.

r. Testing rst2 from rst.
- In this case, rst is set to 1. Then, it is set to 0 and rs2 1 is set 1. It should transition between states correctly and indicate the correct value.

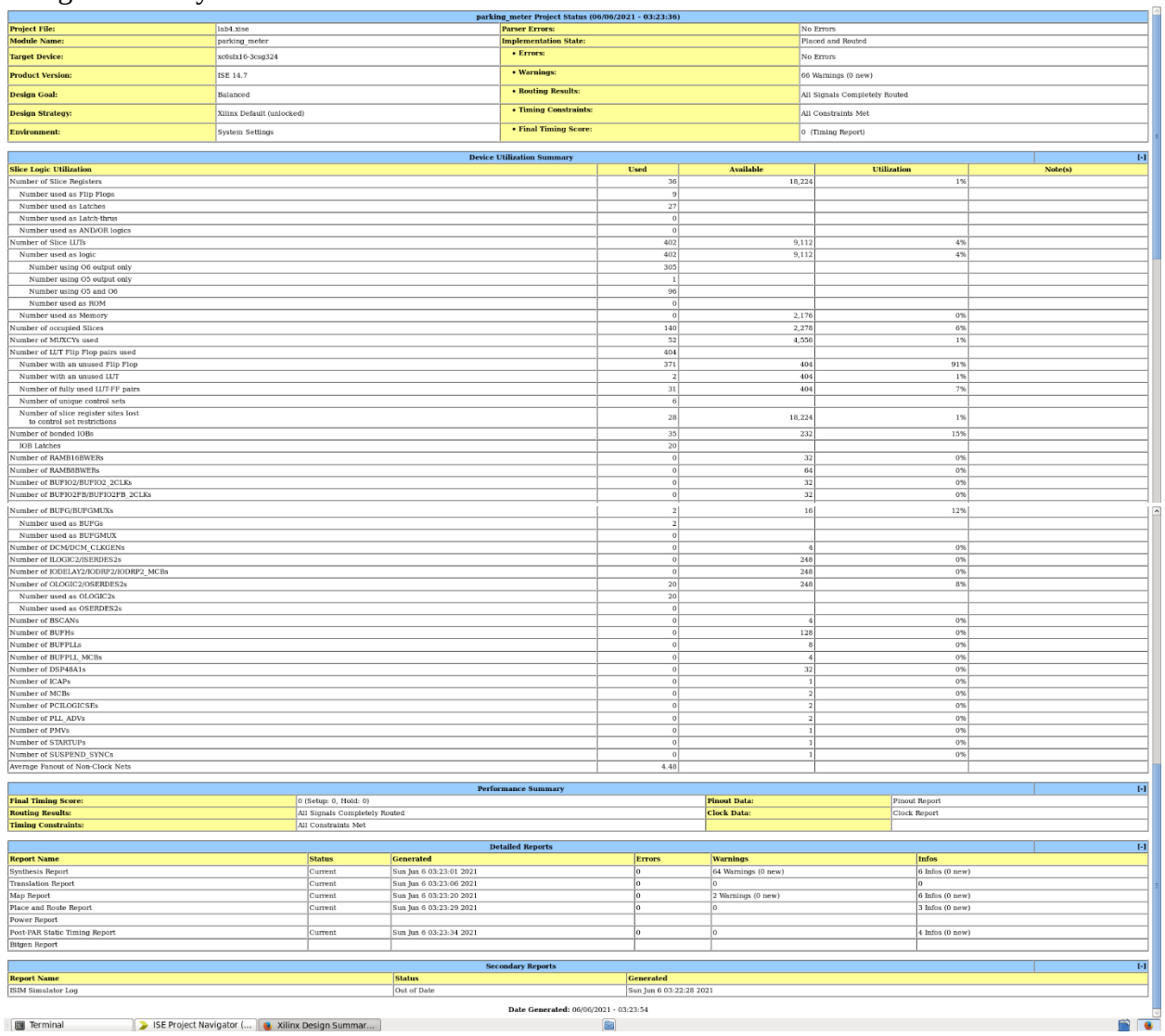
s. Testing rst2 from rst1.
- In this case, rst1 is set to 1. Then, it is set to 0 and rst2 1 is set 1. It should transition between states correctly and indicate the correct value.

t. Testing rst1 from rst2.
- In this case, rst2 is set to 1. Then, it is set to 0 and rst1 is set 1. It should transition between states correctly and indicate the correct value.

4. Waveforms:



5. Design Summary:



parking_meter Project Status (06/06/2021 - 03:23:36)

Project File:	lab4.xise	**Parser Errors:**	No Errors
Module Name:	parking_meter	**Implementation State:**	Placed and Routed
Target Device:	xc6slx16-3csg324	• **Errors:**	No Errors
Product Version:	ISE 14.7	• **Warnings:**	66 Warnings (0 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	36	18,224	1%	
Number used as Flip Flops	9			
Number used as Latches	27			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	402	9,112	4%	
Number used as logic	402	9,112	4%	
Number using O6 output only	305			
Number using O5 output only	1			
Number using O5 and O6	96			
Number used as ROM	0			
Number used as Memory	0	2,176	0%	
Number of occupied Slices	140	2,278	6%	
Number of MUXCYs used	52	4,556	1%	
Number of LUT Flip Flop pairs used	404			
Number with an unused Flip Flop	371	404	91%	
Number with an unused LUT	2	404	1%	
Number of fully used LUT-FF pairs	31	404	7%	
Number of unique control sets	6			
Number of slice register sites lost to control set restrictions	28	18,224	1%	
Number of bonded IOBs	35	232	15%	
IOB Latches	20			
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	2	16	12%	
Number used as BUFGs	2			
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	20	248	8%	
Number used as OLOGIC2s	20			
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	4.48			

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 Jun 6 03:23:01 2021	0	64 Warnings (0 new)	6 Infos (0 new)
Translation Report	Current	Sun Jun 6 03:23:06 2021	0	0	0
Map Report	Current	Sun Jun 6 03:23:20 2021	0	2 Warnings (0 new)	6 Infos (0 new)
Place and Route Report	Current	Sun Jun 6 03:23:29 2021	0	0	3 Infos (0 new)
Power Report					
Post-PAR Static Timing Report	Current	Sun Jun 6 03:23:34 2021	0	0	4 Infos (0 new)
Bitgen Report					

Secondary Reports

Report Name	Status	Generated
ISIM Simulator Log	Out of Date	Sun Jun 6 03:22:28 2021

Date Generated: 06/06/2021 - 03:23:54

Terminal | ISE Project Navigator (... | Xilinx Design Summar...

6. Map Summary:

```
Release 14.7 Map P.20131013 (lin64)

Xilinx Mapping Report File for Design 'parking_meter'


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 parking_meter_map.ncd parking_meter.ngd
```

parking_meter.pcf

Target Device : xc6slx16

Target Package : csg324

Target Speed   : -3

Mapper Version : spartan6 -- $Revision: 1.55 $

Mapped Date   : Sun Jun 6 03:23:09 2021

Design Summary

Number of errors: 0

Number of warnings: 2

Slice Logic Utilization:

Number of Slice Registers:	36	out of	18,224	1%
Number used as Flip Flops:	9			
Number used as Latches:	27			
Number used as Latch-thrus:	0			
Number used as AND/OR logics:	0			
Number of Slice LUTs:	402	out of	9,112	4%
Number used as logic:	402	out of	9,112	4%
Number using O6 output only:	305			
Number using O5 output only:	1			
Number using O5 and O6:	96			
Number used as ROM:	0			
Number used as Memory:	0	out of	2,176	0%

Slice Logic Distribution:

Number of occupied Slices:	140	out of	2,278	6%
Number of MUXCYs used:	52	out of	4,556	1%
Number of LUT Flip Flop pairs used:	404			
Number with an unused Flip Flop:	371	out of	404	91%
Number with an unused LUT:	2	out of	404	1%
Number of fully used LUT-FF pairs:	31	out of	404	7%

Number of unique control sets:	6			
Number of slice register sites lost to control set restrictions:	28	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:	35	out of	232	15%
IOB Latches:	20			

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:	2	out of	16	12%
Number used as BUFGs:	2			
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:	20	out of	248	8%
Number used as OLOGIC2s:	20			
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: 4.48

Peak Memory Usage: 769 MB
Total REAL time to MAP completion: 10 secs
Total CPU time to MAP completion: 9 secs

Table of Contents

Section 1 - Errors

Section 2 - Warnings

WARNING:PhysDesignRules:372 - Gated clock. Clock net

current_state[5]_GND_8_o_Select_85_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]_clock_count[6]_Select_359_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

---

2 block(s) optimized away

Section 5 - Removed Logic

Optimized Block(s):

TYPE	BLOCK
GND	XST_GND
VCC	XST_VCC

To enable printing of redundant blocks removed and signals merged, set the detailed map report option and rerun map.

Section 6 - IOB Properties

IOB Name	Type	Direction	IO Standard	Diff	Drive	Slew	Reg (s)	Resistor	IOB	Term	Strength	Rate		Delay
a1	IOB	OUTPUT	LVCMOS25		12	SLOW	OLATCH							
a2	IOB	OUTPUT	LVCMOS25		12	SLOW	OLATCH							

| a3 | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| a4 | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| add1 | | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| add2 | | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| add3 | | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| add4 | | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| clk | | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| led_seg<0> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| led_seg<1> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| led_seg<2> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| led_seg<3> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| led_seg<4> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| led_seg<5> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| led_seg<6> | | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | | | |
|

| rst | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| rst1 | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| rst2 | IOB | INPUT |
LVCMOS25 | | | | | | |
|

| val1<0> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val1<1> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val1<2> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val1<3> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val2<0> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val2<1> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val2<2> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val2<3> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val3<0> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val3<1> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val3<2> | IOB | OUTPUT |
LVCMOS25 | | 12 | SLOW | OLATCH | | |
|

| val3<3> | IOB | OUTPUT |
LVCMOS25 | 12 | SLOW | OLATCH | |
|

| val4<0> | IOB | OUTPUT |
LVCMOS25 | 12 | SLOW | OLATCH | |
|

| val4<1> | IOB | OUTPUT |
LVCMOS25 | 12 | SLOW | OLATCH | |
|

| val4<2> | IOB | OUTPUT |
LVCMOS25 | 12 | SLOW | OLATCH | |
|

| val4<3> | 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 parking meter. It outputs nine variables, as described above. The design details indicate that there aren'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 states for add1, add2, add3, and add4, but I wasn't satisfied with the behavior when time was added while flashing was supposed to occur. I added this as a test case to make sure it wasn't working, then changed my implementation until it began to work. I also noticed a lot of repeated code, so I used tasks again, and they simplified it tremendously. Maybe learning about tasks can be added to the course, since I enjoyed using them and found them extremely helpful. Another simplification I used was parameters, but instead of using them for states, I also used them for defining important variables. This helped me make quick changes in multiple parts of the project. Overall, I really enjoyed the project, and found myself learning a lot from overcoming challenges like these. Thanks for the great quarter!