CDA 4203L
Computer System Design Lab

Lab 4 Report
Vending Machine FSM

Today's Date:	2/20/2022
Team Members:	Ahmed El Maliki - U18019330
	Stella Kariuki – U19511653
	Ivan Gonzalez -
Work Distribution:	Briefly explain the tasks completed by each team member Ahmed – completed code and ensured proper simulation Stella – Board testing and video presentation Ivan – Report
No. of Hours Spent:	Way too many
Exercise Difficulty: (Easy, Average, Hard)	Hard – Debugger and clock divider came without instructions. Many warnings initially generated stating creation of latches during synthesis, combinatorial implementation of clock (gate clock). Debugging took a huge chunk of time. Link to video presentation: https://youtu.be/NenAHn2YNMs

Any Other Feedback:	Some more information on how to use the provided files should be included, and why they should be used. Websites or links for guidance should be enough.

Problem 1: Show the state diagram of your vending machine controller. Briefly explain how the design works. *Use as many pages as needed.*

Using the Moore model, States were decided upon using the following logic:

- Paying using nickels only would give us 10 total states, S0 through S9, where each state is as a result of an increment of 5c. So S0 would be 0c (base case, no money inserted), S1 would be 5c (1 nickel inserted), … S9 would be 45c (9 nickels inserted, no more needed)
- The max amount of money you can insert in the machine is 65c, given the case where you have already inserted 40c and only have a quarter in your pocket. Analyzing the intermediary states arrived at show that there are states between S9 (45c) and S13 (65c), and that we have a total of 14 states. The **state table** is shown in the snippet of the excel file below in purple. Note that the next state is as a result of the input in the current state, so each state will point to 4 different next states, due to the three different inputs (nickel, dime, quarter), and 1 next state is the default (0 input) which keeps the current state until a coin is inserted.
- The **Outputs** are zero for all states that are below 45c and are all shown below in **green**, described in **orange**. The **Inputs** are also specified in **blue** below. Note that the clock and reset inputs are not included here. They will be included in the simulation and code.

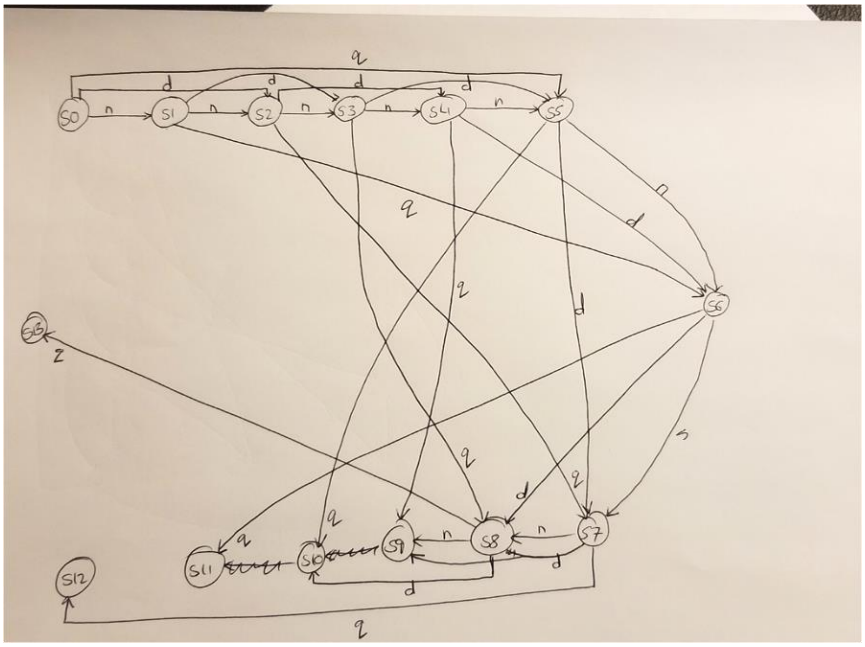
Curr_state	next 5	next 10	next 25	else
S0	S1	S2	S5	S0
S1	S2	S3	S6	S1
S2	S3	S4	S7	S2
S3	S4	S5	S8	S3
S4	S5	S6	S9	S4
S5	S6	S7	S10	S5
S6	S7	S8	S11	S6
S7	S8	S9	S12	S7
S8	S9	S10	S13	S8
S9	X	X	X	X
S10	X	X	X	X
S11	X	X	X	X
S12	X	X	X	X
S13	X	X	X	X

outputs	Description
change	for every 5, pulse
status	0 if accepting money, else 1
GiveSoda	1 if fully paid, else 0
GiveDiet	1 if fully paid, else 0

inputs:	
nickel	5
dime	10
quarter	25
diet	0/1
soda	0/1

States	Value	change	status	giveDiet/giveSoda
S0	0	0	0	0
S1	5	0	0	0
S2	10	0	0	0
S3	15	0	0	0
S4	20	0	0	0
S5	25	0	0	0
S6	30	0	0	0
S7	35	0	0	0
S8	40	0	0	0
S9	45	0	1	1
S10	50	5	1	1
S11	55	10	1	1
S12	60	15	1	1
S13	65	20	1	1

State diagram


S0
S1
S2
S3
S4
S5
S6
S7
S8
S9
S10
S11
S12
S13
n
d
q
z

Problem 1: Include the Verilog code of (a) FSM; (b) Testbench; and (c) Simulation waveforms. *Use as many pages as needed.*

a. FSM Verilog Code (Moore.v)

```
21  module Moore(clk, reset, nickel, dime, qtr, change, status, soda, diet, GiveDiet, GiveSoda);
22
23     input nickel, dime, qtr,soda,diet;
24     input clk;
25     input reset;
26     output reg GiveSoda;
27     output reg GiveDiet;
28     output reg change;
29     output reg status;
30
31
32      //states - 14 different states worst case
33     parameter s0 = 4'b0000; //0
34      parameter s1 = 4'b0001; //5
35      parameter s2 = 4'b0010; //10
36      parameter s3 = 4'b0011; //15
37      parameter s4 = 4'b0100; //20
38      parameter s5 = 4'b0101; //25
39      parameter s6 = 4'b0110;//30
40      parameter s7 = 4'b0111; //35
41      parameter s8 = 4'b1000;//40
42      parameter s9 = 4'b1001; //45
43      parameter s10 = 4'b1010;//50
44      parameter s11 = 4'b1011;//55
45      parameter s12 = 4'b1100;//60
46      parameter s13 = 4'b1101; //65
47
48
49      (*FSM_ENCODING = "SEQUENTIAL", SAFE_IMPLEMENTATION = "NO"*)
50      reg [3:0] state , next_state;
51
52      always @( posedge clk  or posedge reset) begin
53        if (reset==1)
54           state = s0;
55        else
56           state=next_state;
57        end
58
59
60
61
62
```

```
63
64          always@(nickel or dime or qtr or state) begin
65
66          (*FULL_CASE, PARALLEL_CASE*) case (state)
67             s0 : begin
68
69                if(nickel==1)
70                   next_state = s1;
71                else if (dime==1)
72                   next_state = s2;
73                else if (qtr==1)
74                   next_state = s5;
75                else
76                   next_state = s0;
77
78                end
79             s1 : begin
80
81                if(nickel==1)
82                   next_state = s2;
83                else if (dime==1)
84                   next_state = s3;
85                else if (qtr==1)
86                   next_state = s6;
87                else
88                   next_state = s1;
89
90                end
91             s2 : begin
92
93                if(nickel==1)
94                   next_state = s3;
95                else if (dime==1)
96                   next_state = s4;
97                else if (qtr==1)
98                   next_state = s7;
99                else
100                  next_state = s2;
101
102               end
103            s3 : begin
104
```

```
103             s3 : begin
104
105                if(nickel==1)
106                   next_state = s4;
107                else if (dime==1)
108                   next_state = s5;
109                else if (qtr==1)
110                   next_state = s8;
111                else
112                   next_state = s3;
113
114                end
115             s4 : begin
116
117                if(nickel==1)
118                   next_state = s5;
119                else if (dime==1)
120                   next_state = s6;
121                else if (qtr==1)
122                   next_state = s9;
123                else
124                   next_state = s4;
125
126                end
127             s5 : begin
128
129                if(nickel==1)
130                   next_state = s6;
131                else if (dime==1)
132                   next_state = s7;
133                else if (qtr==1)
134                   next_state = s10;
135                else
136                   next_state = s5;
137
138                end
139             s6 : begin
140
141                if(nickel==1)
142                   next_state = s7;
143                else if (dime==1)
144                   next_state = s8;
145                else if (qtr==1)
```

```
142                 next_state = s7;
143              else if (dime==1)
144                 next_state = s8;
145              else if (qtr==1)
146                 next_state = s11;
147              else
148                 next_state = s6;
149
150              end
151           s7 : begin
152
153              if(nickel==1)
154                 next_state = s8;
155              else if (dime==1)
156                 next_state = s9;
157              else if (qtr==1)
158                 next_state= s12;
159              else
160                 next_state =s7;
161
162              end
163           s8 : begin
164
165              if(nickel==1)
166                 next_state = s9;
167              else if (dime==1)
168                 next_state = s10;
169              else if (qtr==1)
170                 next_state = s13;
171              else
172                 next_state = s8;
173
174              end
175           s9 : begin
176
177
178              next_state=s0;
179              end
180           s10 : begin
181
182              next_state=s9;
183
```

```
182                next_state=s9;
183
184                end
185             s11 : begin
186
187                next_state=s10;
188                end
189             s12 : begin
190
191                next_state=s11;
192                end
193             s13 : begin
194
195                next_state=s12;
196                end
197
198             default :  begin
199                        next_state = s0;
200                        end
201
202
203
204
205          endcase
206          end
207
208
209          always@(state or soda or diet)
210          begin
211          case(state)
212          s0: begin GiveSoda = 0;
213                GiveDiet = 0;
214                change = 0;
215                status = 0;
216                end
217          s1:begin GiveSoda = 0;
218                GiveDiet = 0;
219                change = 0;
220                status = 0;
221                end
222          s2: begin GiveSoda = 0;
223                GiveDiet = 0;
224                change = 0;
```

```
222             s2: begin GiveSoda = 0;
223                   GiveDiet = 0;
224                   change = 0;
225                   status = 0;
226                   end
227             s3:begin GiveSoda = 0;
228                   GiveDiet = 0;
229                   change = 0;
230                   status = 0;
231                      end
232             s4: begin GiveSoda = 0;
233                   GiveDiet = 0;
234                   change = 0;
235                   status = 0;
236                   end
237             s5: begin GiveSoda = 0;
238                   GiveDiet = 0;
239                   change = 0;
240                   status = 0;
241                   end
242             s6: begin GiveSoda = 0;
243                   GiveDiet = 0;
244                   change = 0;
245                   status = 0;
246                   end
247             s7: begin GiveSoda = 0;
248                   GiveDiet = 0;
249                   change = 0;
250                   status = 0;
251                   end
252             s8: begin GiveSoda = 0;
253                   GiveDiet = 0;
254                   change = 0;
255                   status = 0;
256                   end
257             s9: begin
258                  if(soda==1)
259                  GiveSoda = 1;
260                  else GiveSoda = 0;
261                  if(diet==1)
262                  GiveDiet = 1;
263                  else GiveDiet = 0;
264                  change = 0;
```

```
262                 GiveDiet = 1;
263                 else GiveDiet = 0;
264                 change = 0;
265                 status = 1;
266                  end
267           s10: begin
268              if(soda==1)
269                 GiveSoda = 1;
270                 else GiveSoda = 0;
271                 if(diet==1)
272                 GiveDiet = 1;
273                 else GiveDiet = 0;
274                 change = 1;
275                 status = 1;
276                  end
277           s11: begin
278                 if(soda==1)
279                 GiveSoda = 1;
280                 else GiveSoda = 0;
281                 if(diet==1)
282                 GiveDiet = 1;
283                 else GiveDiet = 0;
284                 change = 1;
285                 status = 1;
286                  end
287           s12: begin
288              if(soda==1)
289                 GiveSoda = 1;
290                 else GiveSoda = 0;
291                 if(diet==1)
292                 GiveDiet = 1;
293                 else GiveDiet = 0;
294                 change = 1;
295                 status = 1;
296                  end
297           s13:begin
298                 if(soda==1)
299                 GiveSoda = 1;
300                 else GiveSoda = 0;
301                 if(diet==1)
302                 GiveDiet = 1;
303                 else GiveDiet = 0;
304                 change = 1;
```

```
286                end
287           s12: begin
288              if(soda==1)
289                GiveSoda = 1;
290                else GiveSoda = 0;
291                if(diet==1)
292                GiveDiet = 1;
293                else GiveDiet = 0;
294                change = 1;
295                status = 1;
296                 end
296           s13:begin
297                if(soda==1)
298                GiveSoda = 1;
299                else GiveSoda = 0;
300                if(diet==1)
301                GiveDiet = 1;
302                else GiveDiet = 0;
303                change = 1;
304                 status = 1;
305                 end
306           default: begin
307                 GiveSoda = 0;
308                 GiveDiet = 0;
309                 change = 0;
310                 status = 0;
311                 end
312           endcase
313           end
314
315
316
317
318
319   endmodule
320
321
```

b) Testbench – (FSMoore_tb.v)

```
25  module FSMoore_tb;
26
27      // Inputs
28      reg clk;
29      reg reset;
30      reg nickel;
31      reg dime;
32      reg qtr;
33      reg soda;
34      reg diet;
35
36      // Outputs
37      wire change;
38      wire status;
39      wire GiveDiet;
40      wire GiveSoda;
41
42      // Instantiate the Unit Under Test (UUT)
43      Moore uut (
44          .clk(clk),
45          .reset(reset),
46          .nickel(nickel),
47          .dime(dime),
48          .qtr(qtr),
49          .change(change),
50          .status(status),
51          .soda(soda),
52          .diet(diet),
53          .GiveDiet(GiveDiet),
54          .GiveSoda(GiveSoda)
55      );
56
57      always begin
58      clk = 1'b0;
59      #50;
60      clk = 1'b1;
61      #50;
62      end
63
64      initial begin
65  //Each Test case here is going to run 2 inputs below 45c
66  //and 2 inputs above or equal 45c, both for soda and diet inputs
67  //To run, just uncomment the test case you want to run in the 'initial begin' section
```

```
64     initial begin
65  //Each Test case here is going to run 2 inputs below 45c
66  //and 2 inputs above or equal 45c, both for soda and diet inputs
67  //To run, just uncomment the test case you want to run in the 'initial begin' section
68  //DO NOT COMMENT OUT THE INITIALIZATION SECTION
69
70  //INITIALIZATION
71          clk = 0;
72          reset = 0;
73          nickel = 0;
74          dime = 0;
75          qtr = 0;
76          soda = 0;
77          diet = 0;
78
79  //test case 1a => Soda, 35c --- okay
80
81
82          #50;
83          reset=1;
84          #50;
85          reset=0;
86          #50;
87
88          soda=1;
89          qtr=0;
90          dime=0;
91          nickel=0;
92
93          #100;
94          qtr=0;
95          dime=1;
96          nickel=0;
97
98          #100;
99          dime=1;
100         nickel=0;
101         qtr=0;
102
103         #100;
104         dime=1;
105         nickel=0;
106         qtr=0;
```

```
78
79  //test case 1a => Soda, 35c --- okay
80
81
82          #50;
83          reset=1;
84          #50;
85          reset=0;
86          #50;
87
88          soda=1;
89          qtr=0;
90          dime=0;
91          nickel=0;
92
93          #100;
94          qtr=0;
95          dime=1;
96          nickel=0;
97
98          #100;
99          dime=1;
100         nickel=0;
101         qtr=0;
102
103         #100;
104         dime=1;
105         nickel=0;
106         qtr=0;
107
108         #100;
109         dime=0;
110         nickel=1;
111         qtr=0;
112
113         #100;
114         dime=0;
115         nickel=0;
116         qtr=0;
117
```

```
118  //test case 1b -> Soda, 20c --okay
119
120          #50;
121          reset=1;
122          #50;
123          reset=0;
124          #50;
125
126          soda=1;
127          qtr=0;
128          dime=0;
129          nickel=0;
130
131          #100;
132          qtr=0;
133          dime=1;
134          nickel=0;
135
136          #100;
137          dime=1;
138          nickel=0;
139          qtr=0;
140
141          #100;
142          dime=0;
143          nickel=0;
144          qtr=0;
145
146  //test case 1c -> Soda, 45c ---okay
```

```
146  //test case 1c -> Soda, 45c ---okay
147
148          #50;
149          reset=1;
150          #50;
151          reset=0;
152          #50;
153
154          soda=1;
155          qtr=0;
156          dime=0;
157          nickel=0;
158
159          #100;
160          qtr=0;
161          dime=1;
162          nickel=0;
163
164          #100;
165          qtr=0;
166          dime=1;
167          nickel=0;
168
169          #100;
170          dime=1;
171          nickel=0;
172          qtr=0;
173
174          #100;
175          dime=1;
176          nickel=0;
177          qtr=0;
178
179          #100;
180          dime=0;
181          nickel=1;
182          qtr=0;
183
184          #100;
185          dime=0;
186          nickel=0;
187          qtr=0;
188
```

```
191  //test case 1d -> Soda, 65c ---okay
192
193          #50;
194          reset=1;
195          #50;
196          reset=0;
197          #50;
198
199          soda=1;
200          qtr=0;
201          dime=0;
202          nickel=0;
203
204          #100;
205          qtr=0;
206          dime=1;
207          nickel=0;
208
209          #100;
210          qtr=0;
211          dime=1;
212          nickel=0;
213
214          #100;
215          dime=1;
216          nickel=0;
217          qtr=0;
218
219          #100;
220          dime=1;
221          nickel=0;
222          qtr=0;
223
224          #100;
225          dime=0;
226          nickel=0;
227          qtr=1;
228
229          #100;
230          dime=0;
231          nickel=0;
232          qtr=0;
```

```
232         qtr=0;
233
234 //test case 2a -> Diet, 15c ---okay
235
236         #50;
237         reset=1;
238         #50;
239         reset=0;
240         #50;
241
242         diet=1;
243         qtr=0;
244         dime=0;
245         nickel=0;
246
247         #100;
248         qtr=0;
249         dime=0;
250         nickel=1;
251
252         #100;
253         qtr=0;
254         dime=1;
255         nickel=0;
256
257         #100;
258         dime=0;
259         nickel=0;
260         qtr=0;
261
262
263 //test case 2b -> Diet, 40c ---okay
264
265         #50;
266         reset=1;
```

```
262
263  //test case 2b -> Diet, 40c ---okay
264
265          #50;
266          reset=1;
267          #50;
268          reset=0;
269          #50;
270
271          diet=1;
272          qtr=0;
273          dime=0;
274          nickel=0;
275
276          #100;
277          qtr=0;
278          dime=1;
279          nickel=0;
280
281          #100;
282          qtr=0;
283          dime=1;
284          nickel=0;
285
286          #100;
287          dime=1;
288          nickel=0;
289          qtr=0;
290
291          #100;
292          dime=1;
293          nickel=0;
294          qtr=0;
295
296          #100;
297          dime=0;
298          nickel=0;
299          qtr=0;
300
301
```

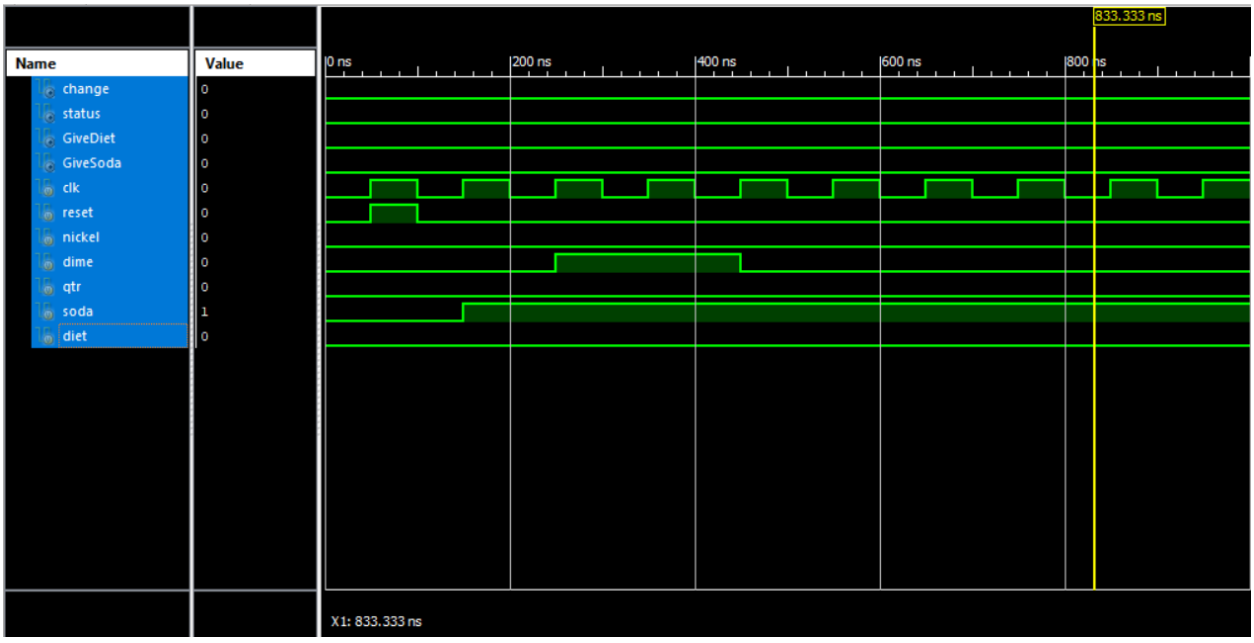
```
301
302  //test case 2c -> Diet, 60c ---okay
303
304          #50;
305          reset=1;
306          #50;
307          reset=0;
308          #50;
309
310          diet=1;
311          qtr=0;
312          dime=0;
313          nickel=0;
314
315          #100;
316          qtr=0;
317          dime=1;
318          nickel=0;
319
320          #100;
321          qtr=0;
322          dime=1;
323          nickel=0;
324
325          #100;
326          dime=1;
327          nickel=0;
328          qtr=0;
329
330          #100;
331          dime=0;
332          nickel=1;
333          qtr=0;
334
335          #100;
336          dime=0;
337          nickel=0;
338          qtr=1;
339
340          #100;
341          dime=0;
342          nickel=0;
343          qtr=0;
```

```
345  //test case 2d -> Diet, 55c -----okay
346
347          #50;
348          reset=1;
349          #50;
350          reset=0;
351          #50;
352
353          diet=1;
354          qtr=0;
355          dime=0;
356          nickel=0;
357
358          #100;
359          qtr=0;
360          dime=1;
361          nickel=0;
362
363          #100;
364          qtr=0;
365          dime=1;
366          nickel=0;
367
368          #100;
369          dime=1;
370          nickel=0;
371          qtr=0;
372
373          #100;
374          dime=0;
375          nickel=0;
376          qtr=1;
377
378          #100;
379          dime=0;
380          nickel=0;
381          qtr=0;
382
383      end
384
385  endmodule
386
387
```

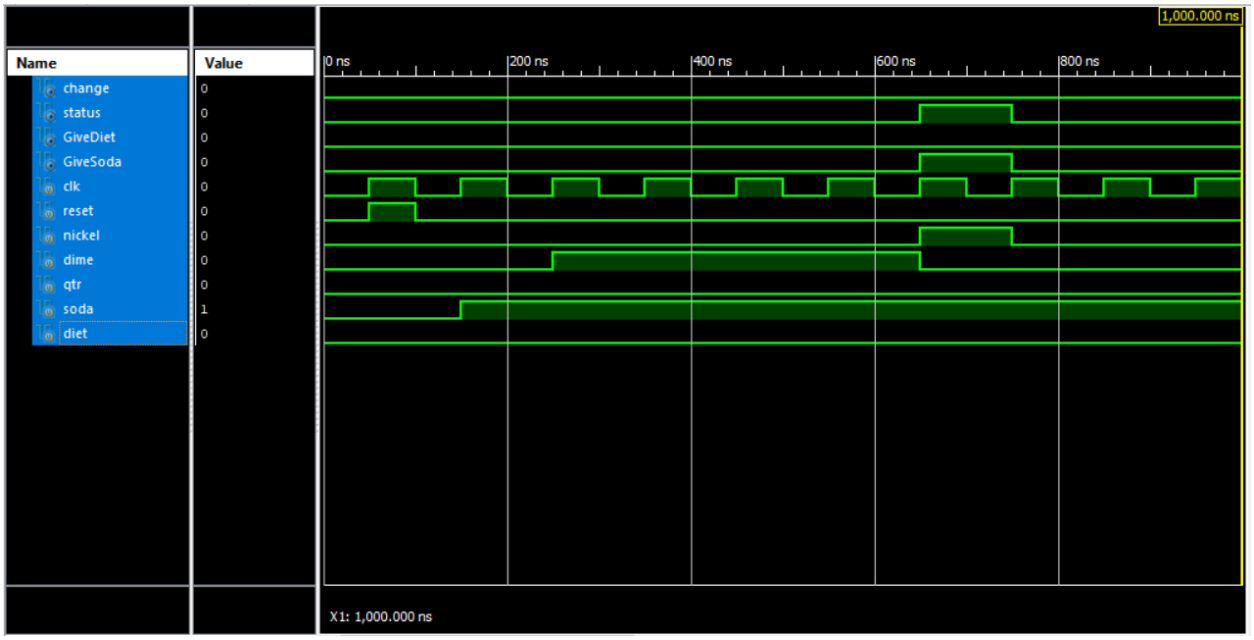
c) Test cases – simulations

I. Test case 1a : Soda, 35c

II. Case 1b Soda, 20c

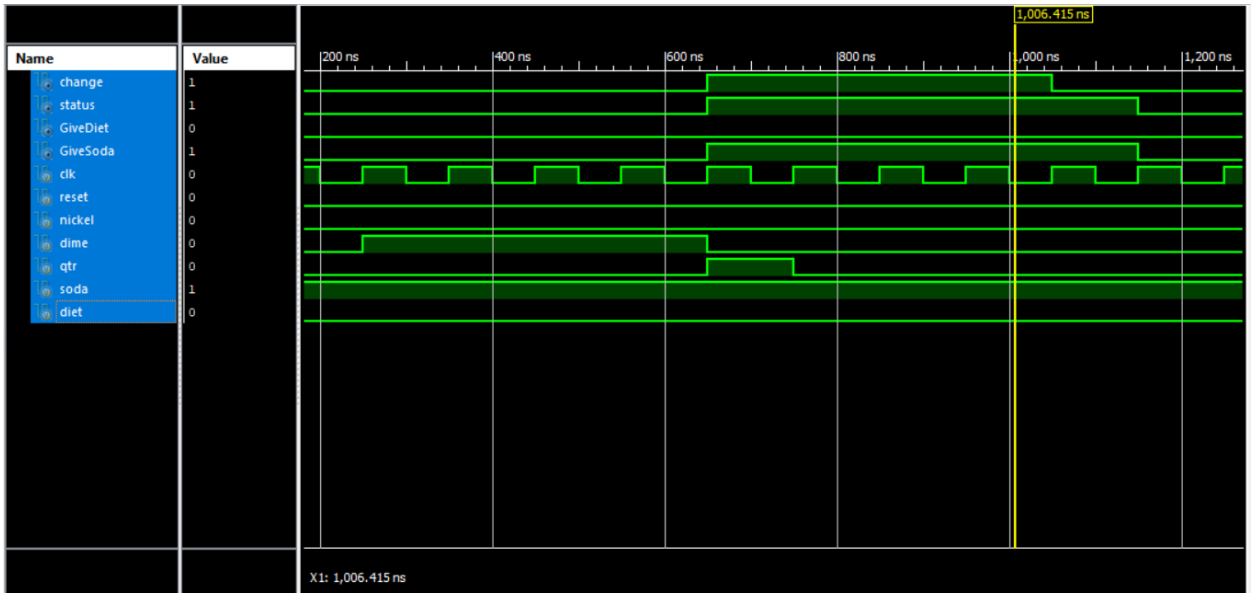


III. Case 1c Soda, 45c

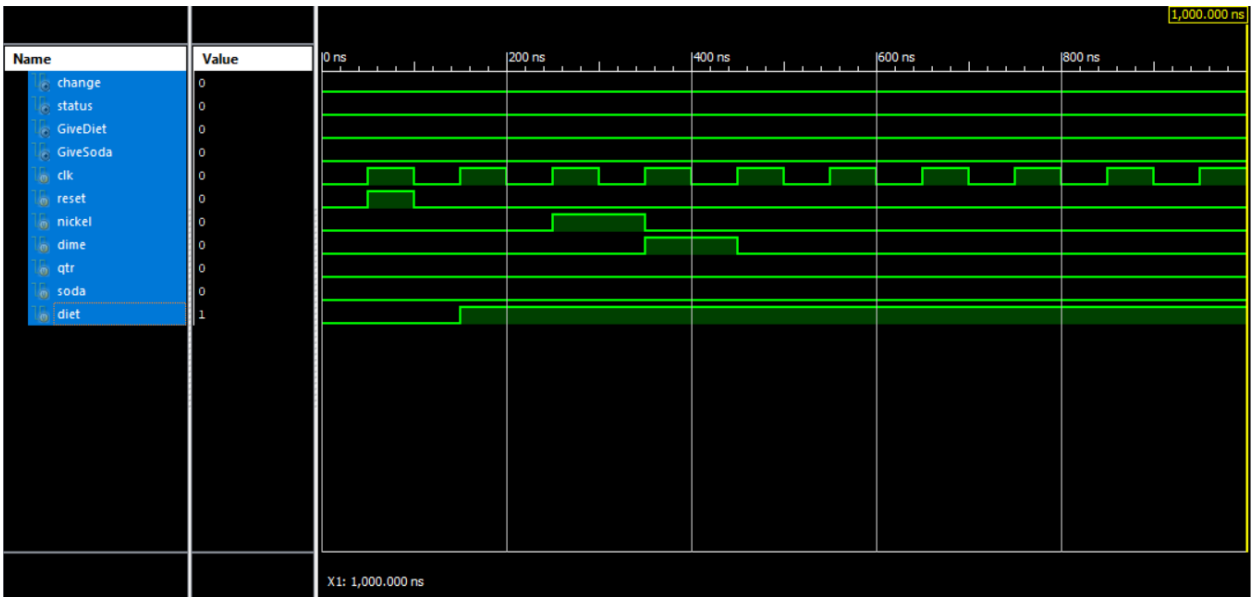


IV. Case 1d Soda, 65c

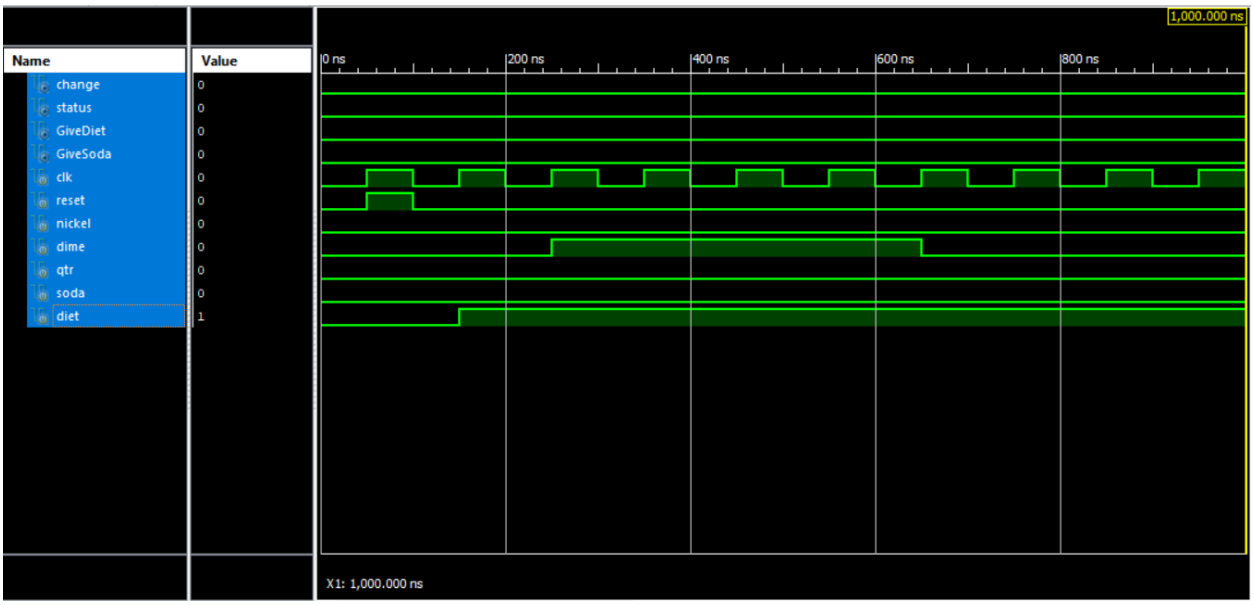
Note that change has a full clock cycle pulse for each nickel; Change of 20 is 4 cc's long.



V. Case 2a Diet, 15c



VI. Case 2b Diet, 40c

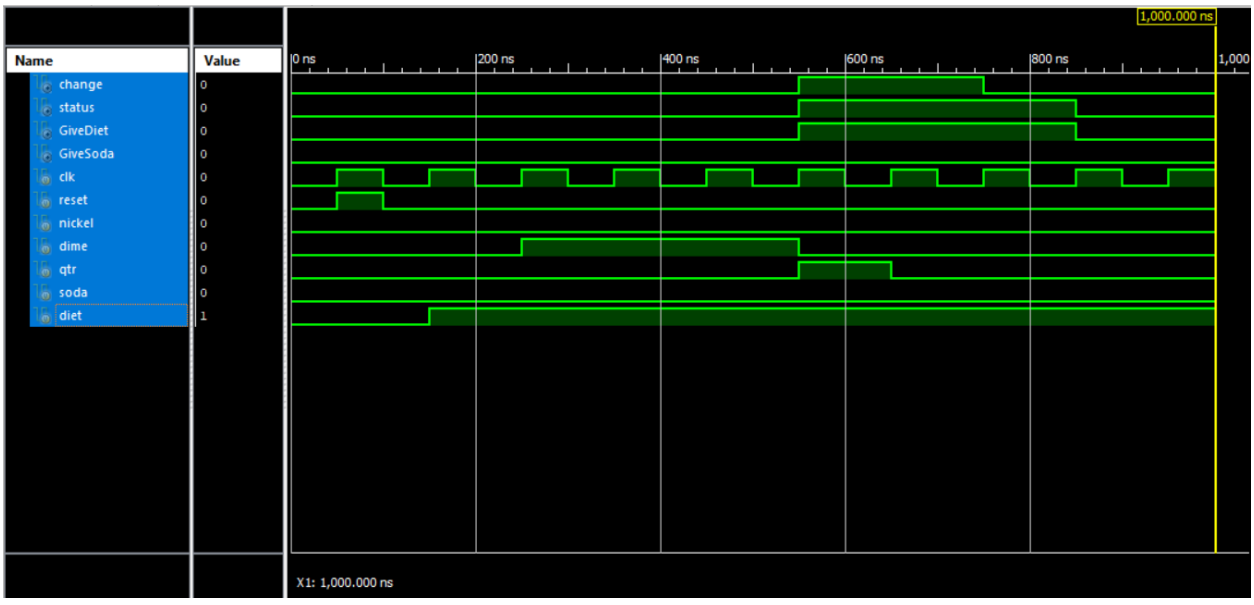


VII. Case 2c Diet, 60c

Note that change has a full clock cycle pulse for each nickel; Change of 15 is 3 cc's long

VIII. Case 2d Diet, 55c

Note that change has a full clock cycle pulse for each nickel; Change of 10 is 2 cc's long



Problem 2: Include (a) the **.ucf** file; and (b) synthesis report. *Use as many pages as needed.*

a. L4PINMAP.ucf

```
# PlanAhead Generated physical constraints
NET "SW<7>" CLOCK_DEDICATED_ROUTE = FALSE;
##Clock Signal 100MHz
NET "clk"   LOC= D11 ;

##Switches
   NET "SW[0]" LOC= V5;    //reset
   NET "SW[1]" LOC= U4  ;     //soda
   NET "SW[2]" LOC= V3  ;     //diet

## LEDs
   NET "LED[0]"   LOC=W3   ;     //RESET
   NET "LED[1]"   LOC=Y4   ;     //SODA
   NET "LED[2]"   LOC=Y1   ;     //DIET

   NET "LED[4]"   LOC=AB4  ;  //CHANGE
   NET "LED[5]"   LOC=W1   ;  //STATUS
   NET "LED[6]"   LOC=AB3  ;     //GIVESODA
   NET "LED[7]"   LOC=AA4  ;     //GIVEDIET


## Buttons

   NET "BTN[1]"   LOC = D5 ;  //NICKEL
   NET "BTN[2]"   LOC = A3 ;     //DIME
   NET "BTN[3]"   LOC = AB9 ;    //QTR
```

b. Synthesis Report

```
=========================================================================
*                    Synthesis Options Summary                          *
=========================================================================
---- Source Parameters
Input File Name                    : "L4_boardtest.prj"
Ignore Synthesis Constraint File   : NO

---- Target Parameters
Output File Name                   : "L4_boardtest"
Output Format                      : NGC
Target Device                      : xc6slx45-3-csg484

---- Source Options
Top Module Name                    : L4_boardtest
Automatic FSM Extraction           : YES
FSM Encoding Algorithm             : Auto
Safe Implementation                : No
FSM Style                          : LUT
RAM Extraction                     : Yes
RAM Style                          : Auto
ROM Extraction                     : Yes
Shift Register Extraction          : YES
ROM Style                          : Auto
Resource Sharing                   : YES
Asynchronous To Synchronous        : NO
Shift Register Minimum Size        : 2
Use DSP Block                      : Auto
Automatic Register Balancing       : No

---- Target Options
LUT Combining                      : Auto
Reduce Control Sets                : Auto
Add IO Buffers                     : YES
Global Maximum Fanout              : 100000
Add Generic Clock Buffer(BUFG)     : 16
Register Duplication               : YES
Optimize Instantiated Primitives   : NO
Use Clock Enable                   : Auto
Use Synchronous Set                : Auto
Use Synchronous Reset              : Auto
Pack IO Registers into IOBs        : Auto
```

```
Equivalent register Removal       : YES

---- General Options
Optimization Goal              : Speed
Optimization Effort            : 1
Power Reduction                : NO
Keep Hierarchy                 : No
Netlist Hierarchy              : As_Optimized
RTL Output                     : Yes
Global Optimization            : AllClockNets
Read Cores                     : YES
Write Timing Constraints       : NO
Cross Clock Analysis           : NO
Hierarchy Separator            : /
Bus Delimiter                  : <>
Case Specifier                 : Maintain
Slice Utilization Ratio        : 100
BRAM Utilization Ratio         : 100
DSP48 Utilization Ratio        : 100
Auto BRAM Packing              : NO
Slice Utilization Ratio Delta  : 5

=========================================================================
==========


=========================================================================
==========
*                     HDL Parsing                       *
=========================================================================
==========
Analyzing Verilog file "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\Moore.v" into library work
Parsing module <Moore>.
Analyzing Verilog file "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\clock_divider.v" into library work
Parsing module <clock_divider>.
Analyzing Verilog file "C:\Users\kariu\Documents\Spring 2022\CDA4203L\Labs\Lab4
Material\Lab_4___Finite_State_Machine_on_FPGA_export\Sample divider and
debouncer\debouncer.v" into library work
Parsing module <debouncer>.
Analyzing Verilog file "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\L4_boardtest.v" into library work
Parsing module <L4_boardtest>.
```

```
=========================================================================
*                     HDL Elaboration                     *
=========================================================================

Elaborating module <L4_boardtest>.

Elaborating module <clock_divider>.

Elaborating module <debouncer>.

Elaborating module <Moore>.

=========================================================================
*                     HDL Synthesis                       *
=========================================================================

Synthesizing Unit <L4_boardtest>.
    Related source file is "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\L4_boardtest.v".
WARNING:Xst:647 - Input <SW<7:3>> is never used. This port will be preserved and
left unconnected if it belongs to a top-level block or it belongs to a sub-block and the
hierarchy of this sub-block is preserved.
WARNING:Xst:647 - Input <BTN<0:0>> is never used. This port will be preserved and
left unconnected if it belongs to a top-level block or it belongs to a sub-block and the
hierarchy of this sub-block is preserved.
    Summary:
        no macro.
Unit <L4_boardtest> synthesized.

Synthesizing Unit <clock_divider>.
    Related source file is "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\clock_divider.v".
    Found 26-bit register for signal <count>.
    Found 1-bit register for signal <clk_div>.
    Found 26-bit adder for signal <count[25]_GND_2_o_add_2_OUT> created at line 26.
    Summary:
        inferred   1 Adder/Subtractor(s).
        inferred  27 D-type flip-flop(s).
        inferred   1 Multiplexer(s).
Unit <clock_divider> synthesized.

Synthesizing Unit <debouncer>.
```

Related source file is "C:\Users\kariu\Documents\Spring 2022\CDA4203L\Labs\Lab4 Material\Lab_4___Finite_State_Machine_on_FPGA_export\Sample divider and debouncer\debouncer.v".
Found 1-bit register for signal <sync2>.
Found 1-bit register for signal <sync3>.
Found 1-bit register for signal <clock_div_prev>.
Found 1-bit register for signal <out_prev>.
Found 1-bit register for signal <out>.
Found 1-bit register for signal <sync>.
Summary:
inferred 6 D-type flip-flop(s).
Unit <debouncer> synthesized.

Synthesizing Unit <Moore>.
Related source file is "C:\Users\kariu\Documents\Spring 2022\CDA4203L\Projects\Lab4\Moore.v".
s0 = 4'b0000
s1 = 4'b0001
s2 = 4'b0010
s3 = 4'b0011
s4 = 4'b0100
s5 = 4'b0101
s6 = 4'b0110
s7 = 4'b0111
s8 = 4'b1000
s9 = 4'b1001
s10 = 4'b1010
s11 = 4'b1011
s12 = 4'b1100
s13 = 4'b1101
Set property "FSM_ENCODING = SEQUENTIAL" for signal <state>.
Set property "SAFE_IMPLEMENTATION = NO" for signal <state>.
Set property "FSM_ENCODING = SEQUENTIAL" for signal <next_state>.
Set property "SAFE_IMPLEMENTATION = NO" for signal <next_state>.
Found 4-bit register for signal <state>.
Found finite state machine <FSM_0> for signal <state>.

States	14
Transitions	41
Inputs	3
Outputs	6
Clock	clk (rising_edge)
Reset	reset (positive)
Reset type	asynchronous
Reset State	0000
Encoding	SEQUENTIAL

```
   | Implementation     | LUT                                |
   ---------------------------------------------------------------------
   Summary:
         inferred   2 Multiplexer(s).
         inferred   1 Finite State Machine(s).
Unit <Moore> synthesized.

=========================================================================
HDL Synthesis Report

Macro Statistics
# Adders/Subtractors                   : 1
 26-bit adder                          : 1
# Registers                            : 20
 1-bit register                        : 19
 26-bit register                       : 1
# Multiplexers                         : 3
 1-bit 2-to-1 multiplexer              : 2
 26-bit 2-to-1 multiplexer             : 1
# FSMs                                 : 1

=========================================================================

=========================================================================
*                 Advanced HDL Synthesis                  *
=========================================================================


Synthesizing (advanced) Unit <clock_divider>.
The following registers are absorbed into counter <count>: 1 register on signal <count>.
Unit <clock_divider> synthesized (advanced).

=========================================================================
Advanced HDL Synthesis Report

Macro Statistics
# Counters                             : 1
 26-bit up counter                     : 1
# Registers                            : 19
 Flip-Flops                           : 19
# Multiplexers                         : 2
```

```
 1-bit 2-to-1 multiplexer                : 2
# FSMs                                   : 1

=========================================================================

=========================================================================
*                  Low Level Synthesis                  *
=========================================================================
Optimizing FSM <FSM_vending/FSM_0> on signal <state[1:4]> with SEQUENTIAL
encoding.
-------------------
 State | Encoding
-------------------
 0000  | 0000
 0101  | 0001
 0010  | 0010
 0001  | 0011
 0110  | 0100
 0011  | 0101
 0111  | 0110
 0100  | 0111
 1000  | 1000
 1001  | 1001
 1010  | 1010
 1011  | 1011
 1100  | 1100
 1101  | 1101
-------------------
INFO:Xst:2261 - The FF/Latch <d3/clock_div_prev> in Unit <L4_boardtest> is
equivalent to the following 2 FFs/Latches, which will be removed : <d2/clock_div_prev>
<d1/clock_div_prev>

Optimizing unit <L4_boardtest> ...

Optimizing unit <Moore> ...
WARNING:Xst:1293 - FF/Latch <cd/count_25> has a constant value of 0 in block
<L4_boardtest>. This FF/Latch will be trimmed during the optimization process.

Mapping all equations...
Building and optimizing final netlist ...
Found area constraint ratio of 100 (+ 5) on block L4_boardtest, actual ratio is 0.

Final Macro Processing ...
```

```
Processing Unit <L4_boardtest> :
        Found 2-bit shift register for signal <d3/sync2>.
        Found 2-bit shift register for signal <d2/sync2>.
        Found 2-bit shift register for signal <d1/sync2>.
Unit <L4_boardtest> processed.

=========================================================================
Final Register Report

Macro Statistics
# Registers                          : 40
 Flip-Flops                         : 40
# Shift Registers                     : 3
 2-bit shift register                : 3

=========================================================================

=========================================================================
*                     Partition Report                        *
=========================================================================

Partition Implementation Status
-------------------------------

  No Partitions were found in this design.

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

=========================================================================
*                     Design Summary                          *
=========================================================================

Top Level Output File Name         : L4_boardtest.ngc

Primitive and Black Box Usage:
------------------------------
# BELS                       : 126
#      GND                   : 1
#      INV                   : 1
```

```
#      LUT1                 : 24
#      LUT2                 : 26
#      LUT3                 : 2
#      LUT4                 : 1
#      LUT5                 : 4
#      LUT6                 : 14
#      MUXCY                : 24
#      MUXF7                : 3
#      VCC                  : 1
#      XORCY                : 25
# FlipFlops/Latches         : 43
#      FD                   : 10
#      FDC                  : 30
#      FDE                  : 3
# Shift Registers           : 3
#      SRLC16E              : 3
# Clock Buffers             : 1
#      BUFGP                : 1
# IO Buffers                : 13
#      IBUF                 : 6
#      OBUF                 : 7

Device utilization summary:
---------------------------

Selected Device : 6slx45csg484-3


Slice Logic Utilization:
 Number of Slice Registers:            43  out of  54576     0%
 Number of Slice LUTs:                 75  out of  27288     0%
    Number used as Logic:              72  out of  27288     0%
    Number used as Memory:              3  out of   6408     0%
      Number used as SRL:               3

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:     79
   Number with an unused Flip Flop:      36  out of     79    45%
   Number with an unused LUT:             4  out of     79     5%
   Number of fully used LUT-FF pairs:    39  out of     79    49%
   Number of unique control sets:         4

IO Utilization:
 Number of IOs:                        21
 Number of bonded IOBs:                14  out of    320     4%
```

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs: 1 out of 16 6%

Partition Resource Summary:

 No Partitions were found in this design.

==========
Timing Report

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
    FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE REPORT
    GENERATED AFTER PLACE-and-ROUTE.

Clock Information:

Clock Signal | Clock buffer(FF name) | Load |
---
clk | BUFGP | 42 |
cd/clk_div | NONE(FSM_vending/state_FSM_FFd2)| 4 |

INFO:Xst:2169 - HDL ADVISOR - Some clock signals were not automatically buffered by XST with BUFG/BUFR resources. Please use the buffer_type constraint in order to insert these buffers to the clock signals to help prevent skew problems.

Asynchronous Control Signals Information:

No asynchronous control signals found in this design

Timing Summary:

Speed Grade: -3

  Minimum period: 3.947ns (Maximum Frequency: 253.383MHz)
  Minimum input arrival time before clock: 2.929ns
  Maximum output required time after clock: 4.939ns
  Maximum combinational path delay: 5.536ns

Timing Details:

All values displayed in nanoseconds (ns)

===
==========
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 3.947ns (frequency: 253.383MHz)
  Total number of paths / destination ports: 1004 / 39

Delay: 3.947ns (Levels of Logic = 3)
  Source: cd/count_9 (FF)
  Destination: cd/count_0 (FF)
  Source Clock: clk rising
  Destination Clock: clk rising

  Data Path: cd/count_9 to cd/count_0
                            Gate   Net
    Cell:in->out   fanout   Delay   Delay Logical Name (Net Name)
    -- ------------
     FDC:C->Q   2   0.447   0.981 cd/count_9 (cd/count_9)
     LUT6:I0->O   2   0.203   0.845 cd/GND_2_o_GND_2_o_equal_7_o<25>4
(cd/GND_2_o_GND_2_o_equal_7_o<25>3)
     LUT6:I3->O   14   0.205   0.958 cd/GND_2_o_GND_2_o_equal_7_o<25>5
(cd/GND_2_o_GND_2_o_equal_7_o)
     LUT2:I1->O   1   0.205   0.000 cd/Mcount_count_eqn_01
(cd/Mcount_count_eqn_0)
     FDC:D   0.102   cd/count_0

    Total   3.947ns (1.162ns logic, 2.785ns route)
                               (29.4% logic, 70.6% route)

===
==========
Timing constraint: Default period analysis for Clock 'cd/clk_div'
  Clock period: 2.790ns (frequency: 358.468MHz)
  Total number of paths / destination ports: 29 / 4

Delay: 2.790ns (Levels of Logic = 2)
  Source: FSM_vending/state_FSM_FFd2 (FF)
  Destination: FSM_vending/state_FSM_FFd2 (FF)
  Source Clock: cd/clk_div rising
  Destination Clock: cd/clk_div rising

  Data Path: FSM_vending/state_FSM_FFd2 to FSM_vending/state_FSM_FFd2
                            Gate   Net

```
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDC:C->Q             12   0.447   1.253  FSM_vending/state_FSM_FFd2
(FSM_vending/state_FSM_FFd2)
     LUT6:I1->O            1   0.203   0.580  FSM_vending/state_FSM_FFd2-In1
(FSM_vending/state_FSM_FFd2-In1)
     LUT6:I5->O            1   0.205   0.000  FSM_vending/state_FSM_FFd2-In2
(FSM_vending/state_FSM_FFd2-In)
     FDC:D                 0.102          FSM_vending/state_FSM_FFd2
    ----------------------------------------
    Total                    2.790ns (0.957ns logic, 1.833ns route)
                                (34.3% logic, 65.7% route)

=========================================================================
==========
Timing constraint: Default OFFSET IN BEFORE for Clock 'clk'
  Total number of paths / destination ports: 29 / 29
-------------------------------------------------------------------------
Offset:              2.929ns (Levels of Logic = 1)
  Source:            SW<0> (PAD)
  Destination:       cd/count_0 (FF)
  Destination Clock: clk rising

  Data Path: SW<0> to cd/count_0
                            Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O            31   1.222   1.277  SW_0_IBUF (LED_0_OBUF)
     FDC:CLR                   0.430          cd/count_0
    ----------------------------------------
    Total                    2.929ns (1.652ns logic, 1.277ns route)
                                (56.4% logic, 43.6% route)

=========================================================================
==========
Timing constraint: Default OFFSET IN BEFORE for Clock 'cd/clk_div'
  Total number of paths / destination ports: 4 / 4
-------------------------------------------------------------------------
Offset:              2.929ns (Levels of Logic = 1)
  Source:            SW<0> (PAD)
  Destination:       FSM_vending/state_FSM_FFd2 (FF)
  Destination Clock: cd/clk_div rising

  Data Path: SW<0> to FSM_vending/state_FSM_FFd2
                            Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
```

```
    ----------------------------------------  ------------
    IBUF:I->O           31   1.222   1.277  SW_0_IBUF (LED_0_OBUF)
    FDC:CLR                   0.430          FSM_vending/state_FSM_FFd4
    ----------------------------------------
    Total                     2.929ns (1.652ns logic, 1.277ns route)
                              (56.4% logic, 43.6% route)

=========================================================================
==========
Timing constraint: Default OFFSET OUT AFTER for Clock 'cd/clk_div'
  Total number of paths / destination ports: 15 / 4
-------------------------------------------------------------------------
Offset:             4.939ns (Levels of Logic = 2)
  Source:           FSM_vending/state_FSM_FFd2 (FF)
  Destination:      LED<7> (PAD)
  Source Clock:     cd/clk_div rising

  Data Path: FSM_vending/state_FSM_FFd2 to LED<7>
                            Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
    FDC:C->Q            12   0.447   1.137  FSM_vending/state_FSM_FFd2
(FSM_vending/state_FSM_FFd2)
    LUT5:I2->O           1   0.205   0.579  FSM_vending/Mmux_GiveDiet11
(LED_7_OBUF)
    OBUF:I->O                 2.571          LED_7_OBUF (LED<7>)
    ----------------------------------------
    Total                     4.939ns (3.223ns logic, 1.716ns route)
                              (65.3% logic, 34.7% route)

=========================================================================
==========
Timing constraint: Default path analysis
  Total number of paths / destination ports: 5 / 5
-------------------------------------------------------------------------
Delay:              5.536ns (Levels of Logic = 3)
  Source:           SW<2> (PAD)
  Destination:      LED<7> (PAD)

  Data Path: SW<2> to LED<7>
                            Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
    IBUF:I->O            2   1.222   0.961  SW_2_IBUF (LED_2_OBUF)
    LUT5:I0->O           1   0.203   0.579  FSM_vending/Mmux_GiveDiet11
(LED_7_OBUF)
```

```
    OBUF:I->O             2.571         LED_7_OBUF (LED<7>)
    ----------------------------------------
    Total                 5.536ns (3.996ns logic, 1.540ns route)
                              (72.2% logic, 27.8% route)

==========================================================================
==========

Cross Clock Domains Report:
--------------------------

Clock to Setup on destination clock cd/clk_div
---------------+---------+---------+---------+---------+
               | Src:Rise| Src:Fall| Src:Rise| Src:Fall|
Source Clock   |Dest:Rise|Dest:Rise|Dest:Fall|Dest:Fall|
---------------+---------+---------+---------+---------+
cd/clk_div     |    2.790|         |         |         |
clk            |    2.569|         |         |         |
---------------+---------+---------+---------+---------+

Clock to Setup on destination clock clk
---------------+---------+---------+---------+---------+
               | Src:Rise| Src:Fall| Src:Rise| Src:Fall|
Source Clock   |Dest:Rise|Dest:Rise|Dest:Fall|Dest:Fall|
---------------+---------+---------+---------+---------+
clk            |    3.947|         |         |         |
---------------+---------+---------+---------+---------+

==========================================================================
==========


Total REAL time to Xst completion: 6.00 secs
Total CPU time to Xst completion: 5.93 secs

-->

Total memory usage is 4486176 kilobytes

Number of errors   :    0 (   0 filtered)
Number of warnings :    3 (   0 filtered)
Number of infos    :    2 (   0 filtered)
```