Michelle Wen
mw37583
EE316 (Telang)

Lab 6 Deliverables

Documentation of Design Process –

My first thought when implementing the design was that I needed four cases for the four different modes. For displaying the outputs onto the 7 segment display, I used some of my code from Lab 4. More specifically, I used the hex to 7 segment converter and the state machine code. When writing in the main module, I started out just having 4 cases and using if-else statements to complete the logic. However, after my code was not working for quite a while, I realized I did not take into account the multiple button presses and clock cycles that occurred. I then drew a simple finite state machine to capture the start/stop button and reset/load button while taking into account the holding down of the buttons (figure shown below).

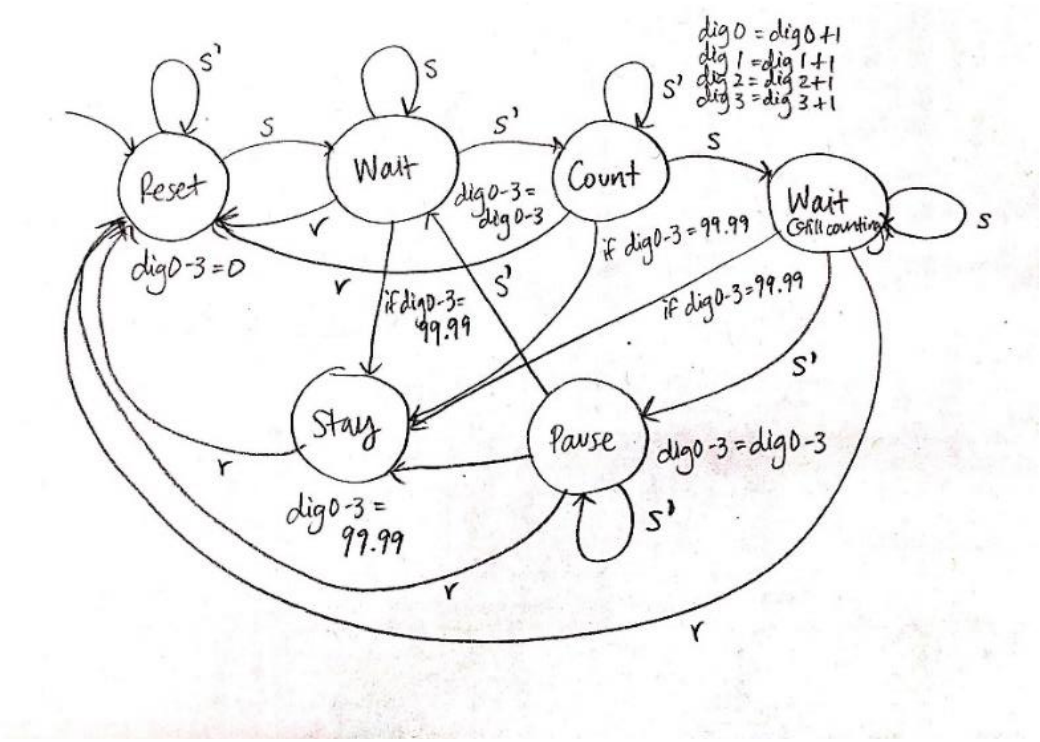
After drawing the finite state machine and understanding more clearly how the button presses have an effect on the design, I started to implement a new design which incorporated more cases for the states of the finite state machine. Originally, I only had 4 cases for the 4 different modes. Now, I had implemented 6 cases within each case for the modes. Each of the 6 cases represent the states, such as Reset, Wait, Count, Wait, Pause, and Stay. These states have a waiting feature that waits for the button press to be released before making any changes to the outputs. So, within each case I checked if the reset/load button was pressed and if the start/stop button was pressed. If neither were pressed, then I would execute some lines of code and return back to the same state. If the reset/load button was pressed, then I would go to the reset state where it clears or loads the digits depending on which mode it is in. If the start/stop button was pressed, I would go to a different state.

A difficulty I encountered while coding within the cases for the states was figuring out how to manipulate the digits. I implemented a slower clock using a clock divider for the least significant digit to run at 100Hz since I wanted the tenths digit to run at 10Hz, the ones digit to run at 1Hz, and the tens digit to run at 0.1Hz. For mode 1 and 2, I needed to keep into account that whenever the hundredths digit was at 9 I had to check if the tenths digit was at 9. If the tenths digit was at 9, I had to check the ones digit and similarly for the tens digit. If the digit to the left of the current digit was not at 9, then I would increment that digit by 1. For mode 3 and 4, a similar method was used except I checked if the digits were at 0 and I decremented the digits by 1 instead of incrementing.

After making sure my logic was correct and writing it all down for the cases, I tested it on the real board and found that my buttons produced no effect when pressed and my ones digit went through A-F instead of going back to 0. I later realized that I needed begin and end before and after a block of code when I'm executing more than 1 statement inside an if statement. I also found that I needed to check for the button presses at the very beginning of each case. After fixing those problems, the last problem I had was the reset button which I later fixed by removing the variable of that button from other submodules, since I only needed it in

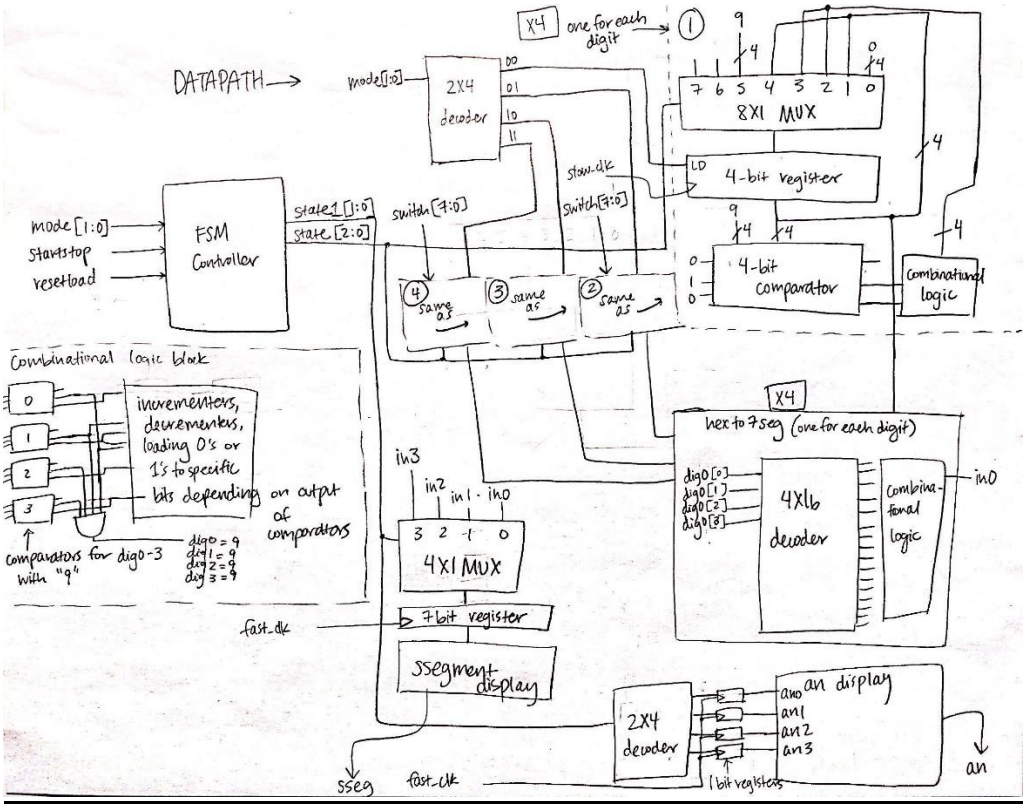
the main module. Now that my mode 1 worked, the other modes were easy to implement since the only differences were loading a value from the switches instead of loading all 0's, or decrementing by 1 and checking for 0's instead of incrementing by 1 and checking for 9's. I tested everything on the real board at this point and everything worked perfectly.

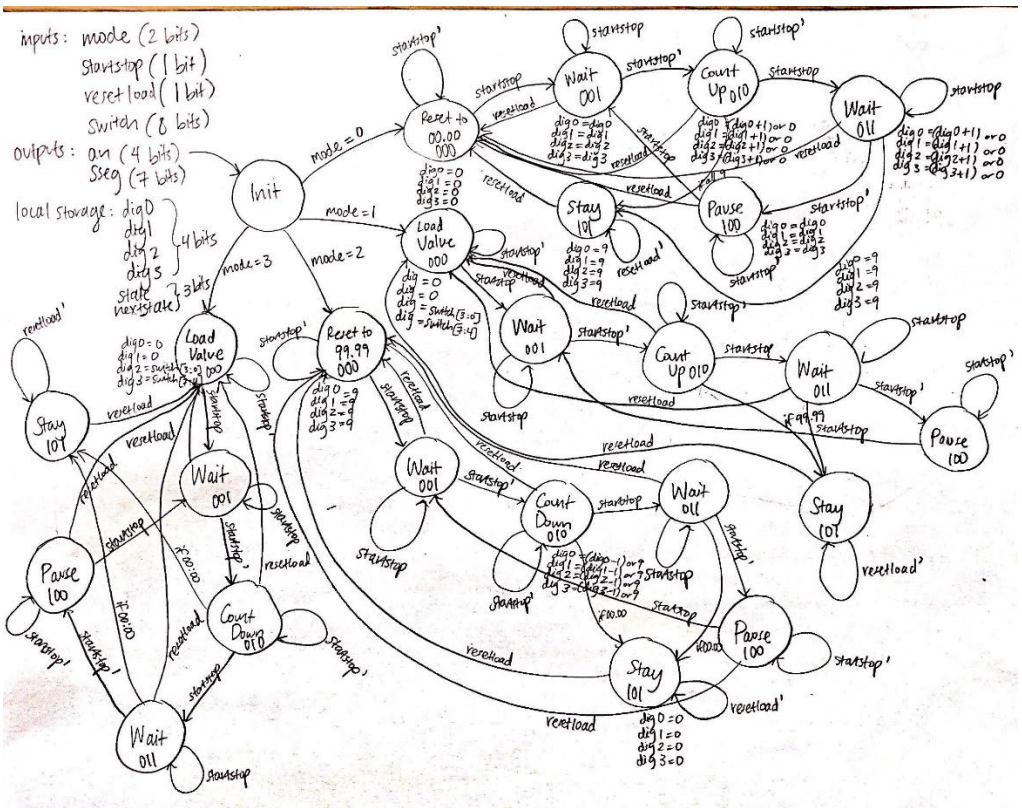
However, I still had one problem left, which was getting the testbench to work. I made a testbench for the deliverables, but the outputs shown on the simulation waveform gave all x's. I spent about 5 hours trying many methods such as changing the bits of the clock dividers to be 1, 2, or both 1 bits. I also added output wire slow_clk and fast_clk to my main module declarations. However, my testbench would still not work even when looking at my previous labs and writing the testbench based off of those.



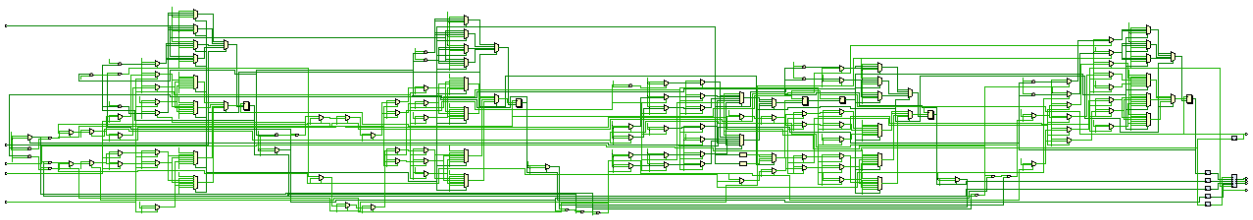
Michelle Wen
mw37583
EE316 (Telang)

Processor Architecture –





Michelle Wen
mw37583
EE316 (Telang)



Verilog Codes and Constraints –

(stopwatch_timer_main)

```
1  `timescale 1ns / 1ps
2
3  module stopwatch_timer_main(
4      input clk,  //clock for changing sseg values
5      input [1:0] mode,   //2 switches for mode select
6      input startstop,    //button for start and stop
7      input resetload,    //button for reset and load
8      input [7:0] switch, //8 switches for external load
9      output [3:0] an,
10     output [6:0] sseg,
11     output dp
12     //output wire slow_clk    //for simulation
13     //output wire fast_clk    //for simulation
14     );
15
16     wire fast_clk;    //for real board
17     wire slow_clk;    //for real board
18     wire [6:0] in0, in1, in2, in3; //the four sseg digits
19     //wire [6:0] swleft, swright;
20     reg [3:0] dig0; //hold individual digits
21     reg [3:0] dig1;
22     reg [3:0] dig2;
23     reg [3:0] dig3;
24
```

Michelle Wen
mw37583
EE316 (Telang)

```
25      reg [2:0] state;
26      reg [2:0] next_state;
27
28      //Module Instantiation of sseg display
29      hexto7seg c1 (.x(dig0), .r(in0));
30      hexto7seg c2 (.x(dig1), .r(in1));
31      hexto7seg c3 (.x(dig2), .r(in2));
32      hexto7seg c4 (.x(dig3), .r(in3));
33
34      //Module Instantiation of the clock divider
35      clk_div c5 (.clk(clk), .slow_clk(slow_clk));
36      clk_div_output c7 (.clk(clk), .fast_clk(fast_clk));
37
38      //Module Instantiation of the multiplexer
39      state_machine c6(
40          .clk(fast_clk),
41          .resetload(resetload),
42          .in0(in0),
43          .in1(in1),
44          .in2(in2),
45          .in3(in3),
46          .an(an),
47          .sseg(sseg),
48          .dp(dp)
```

```
49      );
50
51      initial begin
52        state = 3'b000;
53      end
54
55      always @ (posedge slow_clk) begin
56          case (mode)
57          2'b00: begin        //MODE 1 - Count up & Reset to 00.00
58              case(state)
59                  3'b000: begin   //Reset 00.00 state
60                    begin
61                      dig0 <= 0;
62                      dig1 <= 0;
63                      dig2 <= 0;
64                      dig3 <= 0; end
65                    if(startstop)
66                      next_state <= 3'b001;
67                    else
68                      next_state <= 3'b000; end
69
70                  3'b001: begin   //Wait state
71                    if(resetload)
72                      next_state <= 3'b000;
```

Michelle Wen
mw37583
EE316 (Telang)

```
73                  else if(startstop != 1)
74                    next_state <= 3'b010;
75                  else begin
76                    dig0 <= dig0;
77                    dig1 <= dig1;
78                    dig2 <= dig2;
79                    dig3 <= dig3;
80                    next_state <= 3'b001; end
81                  end
82
83                3'b010: begin   //Counting Up state
84                  if(resetload)
85                    next_state <= 3'b000;
86                  else if(startstop)
87                    next_state <= 3'b011;
88                  else begin
89                  //check if 99.99
90                  if((dig0==9)&&(dig1==9)&&(dig2==9)&&(dig3==9))
91                    next_state <= 3'b101;
92                  else if((dig0 == 9)&&(dig1 != 9))
93                    begin
94                      dig0 <= 0;
95                      dig1 <= dig1 + 1;
96                      next_state <= 3'b010;
```

```
97                    end
98                  else if((dig0 == 9)&&(dig1 == 9))
99                    begin
100                   if((dig2 == 9)&&(dig3 <= 9))
101                     begin
102                       dig0 <= 0;
103                       dig1 <= 0;
104                       dig2 <= 0;
105                       dig3 <= dig3 + 1;
106                       next_state <= 3'b010;
107                     end
108                   else    //dig2 != 9
109                     begin
110                     dig0 <= 0;
111                     dig1 <= 0;
112                     dig2 <= dig2 + 1;
113                     next_state <= 3'b010; end
114                   end
115                 else begin //dig0 != 9
116                   dig0 <= dig0 + 1;
117                   next_state <= 3'b010; end
118                 end
119                 end
120
```

Michelle Wen
mw37583
EE316 (Telang)

```
121                 3'b011: begin   //Wait state
122                   if(resetload)
123                     next_state <= 3'b000;
124                   else if(~startstop)
125                     next_state <= 3'b100;
126                   else begin
127                   //check if 99.99
128                   if((dig0==9)&&(dig1==9)&&(dig2==9)&&(dig3==9))
129                     next_state <= 3'b101;
130                   else if((dig0 == 9)&&(dig1 != 9))
131                     begin
132                       dig0 <= 0;
133                       dig1 <= dig1 + 1;
134                       next_state <= 3'b011;
135                     end
136                   else if((dig0 == 9)&&(dig1 == 9))
137                     begin
138                     if((dig2 == 9)&&(dig3 <= 9))
139                       begin
140                         dig0 <= 0;
141                         dig1 <= 0;
142                         dig2 <= 0;
143                         dig3 <= dig3 + 1;
144                         next_state <= 3'b011;
```

```
145                       end
146                     else begin   //dig2 != 9
147                       dig0 <= 0;
148                       dig1 <= 0;
149                       dig2 <= dig2 + 1;
150                       next_state <= 3'b011; end
151                     end
152                   else begin //dig0 != 9
153                     dig0 <= dig0 + 1;
154                     next_state <= 3'b011; end
155                   end
156                   end
157
158                 3'b100: begin   //Stop state
159                   if(resetload)
160                     next_state <= 3'b000;
161                   else if(startstop)
162                     next_state <= 3'b001;
163                   else begin
164                     dig0 <= dig0;
165                     dig1 <= dig1;
166                     dig2 <= dig2;
167                     dig3 <= dig3;
168                     next_state <= 3'b100; end
```

Michelle Wen
mw37583
EE316 (Telang)

```
169                 end
170
171               3'b101: begin   //Stay 99.99 state
172                 if(resetload)
173                   next_state <= 3'b000;
174                 else begin
175                   dig0 <= 9;
176                   dig1 <= 9;
177                   dig2 <= 9;
178                   dig3 <= 9;
179                   next_state <= 3'b101; end
180                 end
181           endcase
182           end
183         2'b01: begin        //MODE 2 - Load external value & Count Up from there
184            case(state)
185               3'b000: begin   //Load external value state
186                 begin
187                   dig0 <= 0;
188                   dig1 <= 0;
189                   dig2 <= switch[3:0];
190                   dig3 <= switch[7:4]; end
191                 if(startstop)
192                   next_state <= 3'b001;
```

```
193                  else
194                    next_state <= 3'b000; end
195
196               3'b001: begin   //Wait state
197                 if(resetload)
198                   next_state <= 3'b000;
199                 else if(startstop != 1)
200                   next_state <= 3'b010;
201                 else begin
202                   dig0 <= dig0;
203                   dig1 <= dig1;
204                   dig2 <= dig2;
205                   dig3 <= dig3;
206                   next_state <= 3'b001; end
207                 end
208
209               3'b010: begin   //Counting Up state
210                 if(resetload)
211                   next_state <= 3'b000;
212                 else if(startstop)
213                   next_state <= 3'b011;
214                 else begin
215                 //check if 99.99
216                 if((dig0==9)&&(dig1==9)&&(dig2==9)&&(dig3==9))
```

Michelle Wen
mw37583
EE316 (Telang)

```
217                   next_state <= 3'b101;
218                 else if((dig0 == 9)&&(dig1 != 9))
219                   begin
220                     dig0 <= 0;
221                     dig1 <= dig1 + 1;
222                     next_state <= 3'b010;
223                   end
224                 else if((dig0 == 9)&&(dig1 == 9))
225                   begin
226                   if((dig2 == 9)&&(dig3 <= 9))
227                     begin
228                       dig0 <= 0;
229                       dig1 <= 0;
230                       dig2 <= 0;
231                       dig3 <= dig3 + 1;
232                       next_state <= 3'b010;
233                     end
234                   else    //dig2 != 9
235                     begin
236                     dig0 <= 0;
237                     dig1 <= 0;
238                     dig2 <= dig2 + 1;
239                     next_state <= 3'b010; end
240                   end
```

```
241                   else begin //dig0 != 9
242                     dig0 <= dig0 + 1;
243                     next_state <= 3'b010; end
244                   end
245                 end
246
247                 3'b011: begin   //Wait state
248                   if(resetload)
249                     next_state <= 3'b000;
250                   else if(~startstop)
251                     next_state <= 3'b100;
252                   else begin
253                   //check if 99.99
254                   if((dig0==9)&&(dig1==9)&&(dig2==9)&&(dig3==9))
255                     next_state <= 3'b101;
256                   else if((dig0 == 9)&&(dig1 != 9))
257                     begin
258                       dig0 <= 0;
259                       dig1 <= dig1 + 1;
260                       next_state <= 3'b011;
261                     end
262                   else if((dig0 == 9)&&(dig1 == 9))
263                     begin
264                     if((dig2 == 9)&&(dig3 <= 9))
```

Michelle Wen
mw37583
EE316 (Telang)

```
265                     begin
266                       dig0 <= 0;
267                       dig1 <= 0;
268                       dig2 <= 0;
269                       dig3 <= dig3 + 1;
270                       next_state <= 3'b011;
271                     end
272                   else begin   //dig2 != 9
273                     dig0 <= 0;
274                     dig1 <= 0;
275                     dig2 <= dig2 + 1;
276                     next_state <= 3'b011; end
277                   end
278                 else begin //dig0 != 9
279                   dig0 <= dig0 + 1;
280                   next_state <= 3'b011; end
281                 end
282               end
283
284               3'b100: begin   //Stop state
285                 if(resetload)
286                   next_state <= 3'b000;
287                 else if(startstop)
288                   next_state <= 3'b001;
```

```
289                 else begin
290                   dig0 <= dig0;
291                   dig1 <= dig1;
292                   dig2 <= dig2;
293                   dig3 <= dig3;
294                   next_state <= 3'b100; end
295                 end
296
297               3'b101: begin   //Stay 99.99 state
298                 if(resetload)
299                   next_state <= 3'b000;
300                 else begin
301                   dig0 <= 9;
302                   dig1 <= 9;
303                   dig2 <= 9;
304                   dig3 <= 9;
305                   next_state <= 3'b101; end
306                 end
307           endcase
308       end
309       2'b10: begin        //MODE 3 - Count down to 00.00 & reset to 99.99
310           case(state)
311                 3'b000: begin //Reset to 99.99 state
312                   begin
```

Michelle Wen
mw37583
EE316 (Telang)

```
313                     dig0 <= 9;
314                     dig1 <= 9;
315                     dig2 <= 9;
316                     dig3 <= 9; end
317                     if(startstop)
318                       next_state <= 3'b001;
319                     else
320                       next_state <= 3'b000; end
321
322                   3'b001: begin     //Wait state
323                     if(resetload)
324                       next_state <= 3'b000;
325                     else if(startstop != 1)
326                       next_state <= 3'b010;
327                     else begin
328                       dig0 <= dig0;
329                       dig1 <= dig1;
330                       dig2 <= dig2;
331                       dig3 <= dig3;
332                       next_state <= 3'b001; end
333                   end
334
335
336                   3'b010: begin     //Counting Down state
```

```
337                     if(resetload)
338                       next_state <= 3'b000;
339                     else if(startstop)
340                       next_state <= 3'b011;
341                     else begin
342                     //check if 00.00
343                       if((dig3 == 0)&&(dig2 == 0)&&(dig1 == 0)&&(dig0 == 0))
344                         next_state <= 3'b101;
345                       else if((dig0 == 0)&&(dig1 != 0))
346                         begin
347                           dig0 <= 9;
348                           dig1 <= dig1 - 1;
349                           next_state <= 3'b010;
350                         end
351                       else if((dig0 == 0)&&(dig1 == 0))
352                         begin
353                         if((dig2 == 0)&&(dig3 != 0))         //!= sign
354                           begin
355                             dig0 <= 9;
356                             dig1 <= 9;
357                             dig2 <= 9;
358                             dig3 <= dig3 - 1;
359                             next_state <= 3'b010;
360                           end
```

Michelle Wen
mw37583
EE316 (Telang)

```
361                     else    //dig2 != 0
362                       begin
363                         dig0 <= 9;
364                         dig1 <= 9;
365                         dig2 <= dig2 - 1;
366                         next_state <= 3'b010;
367                       end
368                     end
369                   else begin    //dig0 != 0
370                     dig0 <= dig0 - 1;
371                     next_state <= 3'b010; end
372                 end
373               end
374
375               3'b011: begin     //Wait state
376                 if(resetload)
377                   next_state <= 3'b000;
378                 else if(~startstop)
379                   next_state <= 3'b100;
380                 else begin
381                 //check if 00.00
382                   if((dig3 == 0)&&(dig2 == 0)&&(dig1 == 0)&&(dig0 == 0))
383                     next_state <= 3'b101;
384                   else if((dig0 == 0)&&(dig1 != 0))
```

```
385                     begin
386                       dig0 <= 9;
387                       dig1 <= dig1 - 1;
388                       next_state <= 3'b011;
389                     end
390                   else if((dig0 == 0)&&(dig1 == 0))
391                     begin
392                     if((dig2 == 0)&&(dig3 != 0))          //!= sign
393                       begin
394                         dig0 <= 9;
395                         dig1 <= 9;
396                         dig2 <= 9;
397                         dig3 <= dig3 - 1;
398                         next_state <= 3'b011;
399                       end
400                     else    //dig2 != 0
401                       begin
402                         dig0 <= 9;
403                         dig1 <= 9;
404                         dig2 <= dig2 - 1;
405                         next_state <= 3'b011;
406                       end
407                     end
408                   else begin    //dig0 != 0
```

Michelle Wen
mw37583
EE316 (Telang)

```
409                         dig0 <= dig0 - 1;
410                         next_state <= 3'b011; end
411                   end
412                 end
413                 3'b100: begin
414                   if(resetload)
415                     next_state <= 3'b000;
416                   else if(startstop)
417                     next_state <= 3'b001;
418                   else begin
419                     dig0 <= dig0;
420                     dig1 <= dig1;
421                     dig2 <= dig2;
422                     dig3 <= dig3;
423                     next_state <= 3'b100; end
424                 end
425
426                 3'b101: begin     //Stay in 00.00 state
427                   if(resetload)
428                     next_state <= 3'b000;
429                   else begin
430                     dig0 <= 0;
431                     dig1 <= 0;
432                     dig2 <= 0;
```

```
433                     dig3 <= 0;
434                     next_state <= 3'b101; end
435                 end
436             endcase
437         end
438         2'b11: begin        //MODE 4 - Count down to 00.00 & load from value
439             case(state)
440                   3'b000: begin //Load external value
441                     begin
442                     dig0 <= 0;
443                     dig1 <= 0;
444                     dig2 <= switch[3:0];
445                     dig3 <= switch[7:4]; end
446                     if(startstop)
447                       next_state <= 3'b001;
448                     else
449                       next_state <= 3'b000; end
450
451                   3'b001: begin     //Wait state
452                     if(resetload)
453                       next_state <= 3'b000;
454                     else if(startstop != 1)
455                       next_state <= 3'b010;
456                     else begin
```

Michelle Wen
mw37583
EE316 (Telang)

```
457                     dig0 <= dig0;
458                     dig1 <= dig1;
459                     dig2 <= dig2;
460                     dig3 <= dig3;
461                     next_state <= 3'b001; end
462                 end
463
464
465                 3'b010: begin     //Counting Down state
466                   if(resetload)
467                     next_state <= 3'b000;
468                   else if(startstop)
469                     next_state <= 3'b011;
470                   else begin
471                   //check if 00.00
472                     if((dig3 == 0)&&(dig2 == 0)&&(dig1 == 0)&&(dig0 == 0))
473                       next_state <= 3'b101;
474                     else if((dig0 == 0)&&(dig1 != 0))
475                       begin
476                         dig0 <= 9;
477                         dig1 <= dig1 - 1;
478                         next_state <= 3'b010;
479                       end
480                     else if((dig0 == 0)&&(dig1 == 0))
```

```
481                       begin
482                       if((dig2 == 0)&&(dig3 != 0))          //!= sign
483                         begin
484                           dig0 <= 9;
485                           dig1 <= 9;
486                           dig2 <= 9;
487                           dig3 <= dig3 - 1;
488                           next_state <= 3'b010;
489                         end
490                       else    //dig2 != 0
491                         begin
492                           dig0 <= 9;
493                           dig1 <= 9;
494                           dig2 <= dig2 - 1;
495                           next_state <= 3'b010;
496                         end
497                       end
498                     else begin    //dig0 != 0
499                       dig0 <= dig0 - 1;
500                       next_state <= 3'b010; end
501                   end
502                 end
503
504                 3'b011: begin     //Wait state
```

Michelle Wen
mw37583
EE316 (Telang)

```
505                 if(resetload)
506                   next_state <= 3'b000;
507                 else if(~startstop)
508                   next_state <= 3'b100;
509                 else begin
510                 //check if 00.00
511                   if((dig3 == 0)&&(dig2 == 0)&&(dig1 == 0)&&(dig0 == 0))
512                     next_state <= 3'b101;
513                   else if((dig0 == 0)&&(dig1 != 0))
514                     begin
515                       dig0 <= 9;
516                       dig1 <= dig1 - 1;
517                       next_state <= 3'b011;
518                     end
519                   else if((dig0 == 0)&&(dig1 == 0))
520                     begin
521                     if((dig2 == 0)&&(dig3 != 0))          //!= sign
522                       begin
523                         dig0 <= 9;
524                         dig1 <= 9;
525                         dig2 <= 9;
526                         dig3 <= dig3 - 1;
527                         next_state <= 3'b011;
528                       end
```

```
529                       else    //dig2 != 0
530                         begin
531                           dig0 <= 9;
532                           dig1 <= 9;
533                           dig2 <= dig2 - 1;
534                           next_state <= 3'b011;
535                         end
536                       end
537                     else begin    //dig0 != 0
538                       dig0 <= dig0 - 1;
539                       next_state <= 3'b011; end
540                   end
541                 end
542                 3'b100: begin     //Hold values
543                   if(resetload)
544                     next_state <= 3'b000;
545                   else if(startstop)
546                     next_state <= 3'b001;
547                   else begin
548                     dig0 <= dig0;
549                     dig1 <= dig1;
550                     dig2 <= dig2;
551                     dig3 <= dig3;
552                     next_state <= 3'b100; end
```

Michelle Wen
mw37583
EE316 (Telang)

```
                    end

                    3'b101: begin     //Stay in 00.00 state
                      if(resetload)
                        next_state <= 3'b000;
                      else begin
                        dig0 <= 0;
                        dig1 <= 0;
                        dig2 <= 0;
                        dig3 <= 0;
                        next_state <= 3'b101; end
                    end
                endcase
            end
            endcase
        end

        always @ (posedge slow_clk) begin
            case (mode)
              2'b00: begin  //Mode 1
                  state <= next_state; end
              2'b01: begin  //Mode 2
                  state <= next_state; end
              2'b10: begin  //Mode 3
                  state <= next_state; end
              2'b11: begin  //Mode 4
                  state <= next_state; end
            endcase
        end
    endmodule
```

(hexto7seg)

```
`timescale 1ns / 1ps

module hexto7seg(
    input [3:0] x,  //input is the digit
    output reg [6:0] r
    );

    always @(*)
        case (x)
            4'b0000 : r = 7'b0000001;
            4'b0001 : r = 7'b1001111;
            4'b0010 : r = 7'b0010010;
            4'b0011 : r = 7'b0000110;
            4'b0100 : r = 7'b1001100;
            4'b0101 : r = 7'b0100100;
            4'b0110 : r = 7'b0100000;
            4'b0111 : r = 7'b0001111;
            4'b1000 : r = 7'b0000000;
            4'b1001 : r = 7'b0000100;   //9
            4'b1010 : r = 7'b0001000;
            4'b1011 : r = 7'b0000000;
            4'b1100 : r = 7'b0110001;
            4'b1101 : r = 7'b0000001;
            4'b1110 : r = 7'b0110000;
```

Michelle Wen
mw37583
EE316 (Telang)

```
            4'b1111 : r = 7'b0111000;
        endcase

endmodule
```

(clk_div)

```
`timescale 1ns / 1ps

module clk_div(
    input clk,
    output slow_clk
    );

    //26 bits is 1Hz (1s)
    //Want clock divider to run at 100Hz = 19-20 bits

    //reg [1:0] COUNT;  //for simulation
    reg [19:0] COUNT;

    //changing this changes the output to sseg (flashy)

    //assign slow_clk = COUNT[1];   //for simulation
    assign slow_clk = COUNT[19];

    always @ (posedge clk)
    begin
        COUNT = COUNT + 1;
    end

endmodule
```

(clk_div_output)

```
`timescale 1ns / 1ps

module clk_div_output(
    input clk,
    output fast_clk
    );

    //reg [1:0] COUNT;  //for simulation
    reg [11:0] COUNT;

    //changing this changes the output to sseg (flashy)

    //assign slow_clk = COUNT[1];   //for simulation
    assign fast_clk = COUNT[11];

    always @ (posedge clk)
    begin
        COUNT = COUNT + 1;
    end

endmodule
```

(state_machine)

Michelle Wen
mw37583
EE316 (Telang)

```
`timescale 1ns / 1ps

module state_machine(
    input clk,
    input resetload,
    input [6:0] in0,    //inputs are the 7 bit inputs for each digit
    input [6:0] in1,
    input [6:0] in2,
    input [6:0] in3,
    output reg [3:0] an,
    output reg [6:0] sseg,
    output reg dp
    );

    reg [1:0] state1;
    reg [1:0] nextstate;

    always @(*) begin
        case(state1)
          2'b00: nextstate = 2'b01;
          2'b01: nextstate = 2'b10;
          2'b10: nextstate = 2'b11;
          2'b11: nextstate = 2'b00;
        endcase
```

```
    end

    always @(*) begin
        case(state1)
          2'b00: sseg = in0;
          2'b01: sseg = in1;
          2'b10: sseg = in2;
          2'b11: sseg = in3;
        endcase
        case(state1)
          2'b00: begin an = 4'b1110; dp = 1; end //these lines only turn it on/off
          2'b01: begin an = 4'b1101; dp = 1; end
          2'b10: begin an = 4'b1011; dp = 0; end
          2'b11: begin an = 4'b0111; dp = 1; end
        endcase
    end

    always @(posedge clk) begin
        state1 <= nextstate;
    end
endmodule

```

(constraints file)

Michelle Wen
mw37583
EE316 (Telang)

```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports {clk}]
    set_property IOSTANDARD LVCMOS33 [get_ports {clk}]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk}]

## Switches
set_property PACKAGE_PIN V17 [get_ports {mode[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {mode[0]}]
set_property PACKAGE_PIN V16 [get_ports {mode[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {mode[1]}]
set_property PACKAGE_PIN W16 [get_ports {switch[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[0]}]
set_property PACKAGE_PIN W17 [get_ports {switch[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[1]}]
set_property PACKAGE_PIN W15 [get_ports {switch[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[2]}]
set_property PACKAGE_PIN V15 [get_ports {switch[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[3]}]
set_property PACKAGE_PIN W14 [get_ports {switch[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[4]}]
set_property PACKAGE_PIN W13 [get_ports {switch[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[5]}]
set_property PACKAGE_PIN V2 [get_ports {switch[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[6]}]
```

```
set_property PACKAGE_PIN T3 [get_ports {switch[7]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {switch[7]}]
#set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
#set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[11]}]
#set_property PACKAGE_PIN W2 [get_ports {sw[12]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[12]}]
#set_property PACKAGE_PIN U1 [get_ports {sw[13]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[13]}]
#set_property PACKAGE_PIN T1 [get_ports {sw[14]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[14]}]
#set_property PACKAGE_PIN R2 [get_ports {sw[15]}]
    #set_property IOSTANDARD LVCMOS33 [get_ports {sw[15]}]


##7 segment display
set_property PACKAGE_PIN W7 [get_ports {sseg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[6]}]
set_property PACKAGE_PIN W6 [get_ports {sseg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[5]}]
set_property PACKAGE_PIN U8 [get_ports {sseg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[4]}]
set_property PACKAGE_PIN V8 [get_ports {sseg[3]}]
```

Michelle Wen
mw37583
EE316 (Telang)

```
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[3]}]
set_property PACKAGE_PIN U5 [get_ports {sseg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[2]}]
set_property PACKAGE_PIN V5 [get_ports {sseg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[1]}]
set_property PACKAGE_PIN U7 [get_ports {sseg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sseg[0]}]

set_property PACKAGE_PIN V7 [get_ports {dp}]
    set_property IOSTANDARD LVCMOS33 [get_ports {dp}]

set_property PACKAGE_PIN U2 [get_ports {an[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[0]}]
set_property PACKAGE_PIN U4 [get_ports {an[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[1]}]
set_property PACKAGE_PIN V4 [get_ports {an[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[2]}]
set_property PACKAGE_PIN W4 [get_ports {an[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {an[3]}]


##Buttons
set_property PACKAGE_PIN U18 [get_ports {startstop}]
    set_property IOSTANDARD LVCMOS33 [get_ports {startstop}]
```

```
set_property PACKAGE_PIN T18 [get_ports {resetload}]
    set_property IOSTANDARD LVCMOS33 [get_ports {resetload}]
#set_property PACKAGE_PIN W19 [get_ports btnL]
    #set_property IOSTANDARD LVCMOS33 [get_ports btnL]
#set_property PACKAGE_PIN T17 [get_ports btnR]
    #set_property IOSTANDARD LVCMOS33 [get_ports btnR]
#set_property PACKAGE_PIN U17 [get_ports btnD]
    #set_property IOSTANDARD LVCMOS33 [get_ports btnD]
```

(tb_stopwatch_timer_main)

```
`timescale 1ns / 1ps

module tb_stopwatch_timer_main;
    reg clk;  //clock for changing sseg values
    reg [1:0] mode;   //2 switches for mode select
    reg startstop;    //button for start and stop
    reg resetload;    //button for reset and load
    reg [7:0] switch; //8 switches for external load
    wire [3:0] an;
    wire [6:0] sseg;
    wire dp;
    wire slow_clk;
    wire fast_clk;

    stopwatch_timer_main ul(
        .clk(clk),
        .mode(mode),
        .startstop(startstop),
        .resetload(resetload),
        .switch(switch),
        .an(an),
        .sseg(sseg),
        .dp(dp),
        .slow_clk(slow_clk),
```

Michelle Wen
mw37583
EE316 (Telang)

```
25          .fast_clk(fast_clk)
26      );
27
28      initial begin
29
30      clk = 0;
31      resetload = 0;
32      mode = 2'b00;
33      startstop = 0;
34      switch = 8'b00000000;
35
36      #50
37
38      resetload = 1;
39
40      #50
41      resetload = 0;
42      #50
43      resetload = 0;
44      #50
45      startstop = 1;
46
47      #50
```

```
48
49      startstop = 0;
50      #50
51      resetload = 0;
52      #50
53
54      startstop = 1;
55
56      #50
57
58      startstop = 0;
59      #50
60      resetload = 0;
61      #50
62
63      startstop = 1;
64
65      #50
66
67      startstop = 0;
68      #50
69      resetload = 0;
70      #50
```

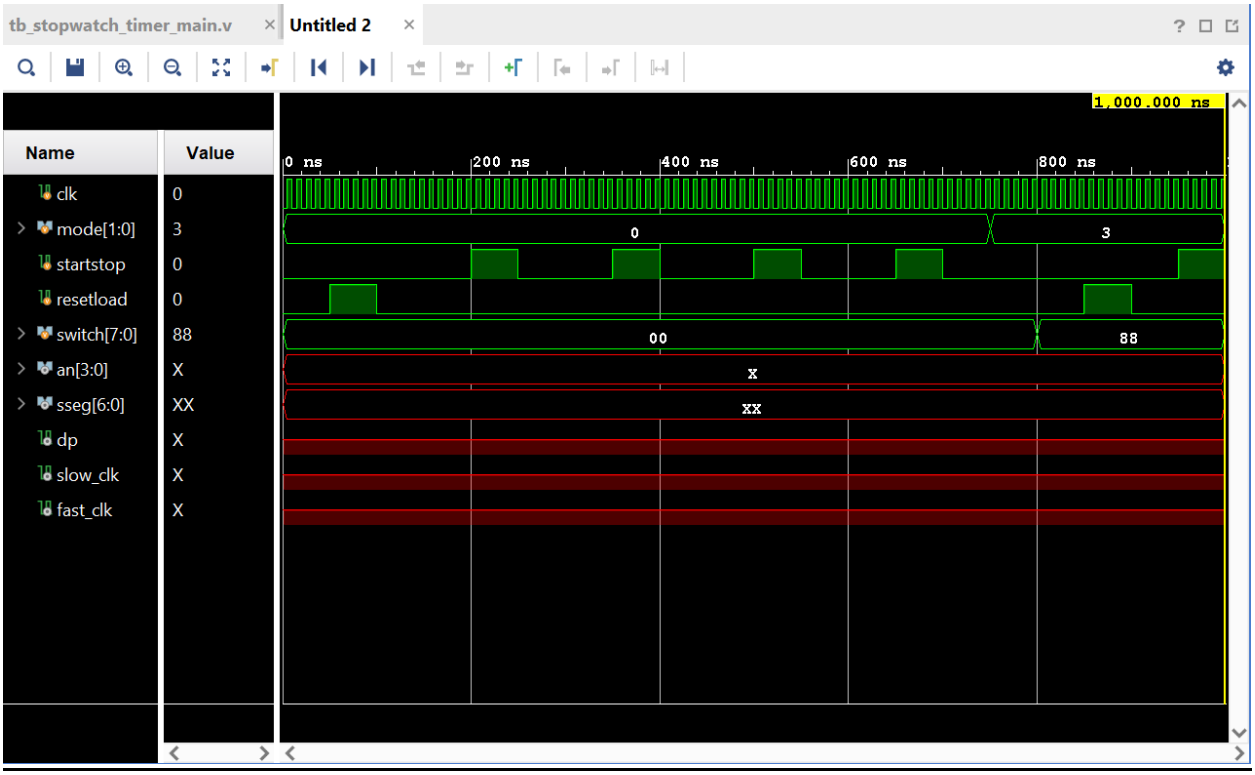
Michelle Wen
mw37583
EE316 (Telang)

```
71
72      startstop = 1;
73
74      #50
75
76      startstop = 0;
77
78      #50
79
80      mode = 2'b11;
81
82      #50
83
84      switch = 8'b10001000;
85
86      #50
87
88      resetload = 1;
89
90      #50
91
92      resetload = 0;
```

```
93
94      #50
95
96      startstop = 1;
97
98      #50
99
100     startstop = 0;
101
102     end
103
104     always
105     #5 clk = ~clk;
106
107 endmodule
108
```

Michelle Wen
mw37583
EE316 (Telang)

Simulation Waveforms of Testbench –



I was not able to get my simulation waveform to output the correct values. I based my testbench off previous testbenches from previous labs, including the parts about adding “output wire slow_clk” and “output wire fast_clk” to the declarations in the main module and adding wires in the testbench. I made sure to make my clock dividers faster (2 bits) so that it would be able to show up in simulation. I also made sure that I initialized every register in the testbench and tried using the scope and watch windows to debug my simulation. However, after trying to debug my testbench and simulation, I was still not able to figure out why my simulation was showing all x’s.