ELC 2137 Lab 10: 7-segment Display with Time-Division Multiplexing

Jake Simmons

April 9, 2020

Summary

Type the summary of your experiment and results here.

Q&A

Answer questions posed in the lab assignment here.

Results

Time (ns):	0	20	40	60	80	100	120	140	160	180	200	220	240	260	280	300	320
clk	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
en	0	1	1	1	0	0	1	0	1	0	1	1	0	1	1	1	1
rst	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W1	X	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	0
W2	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

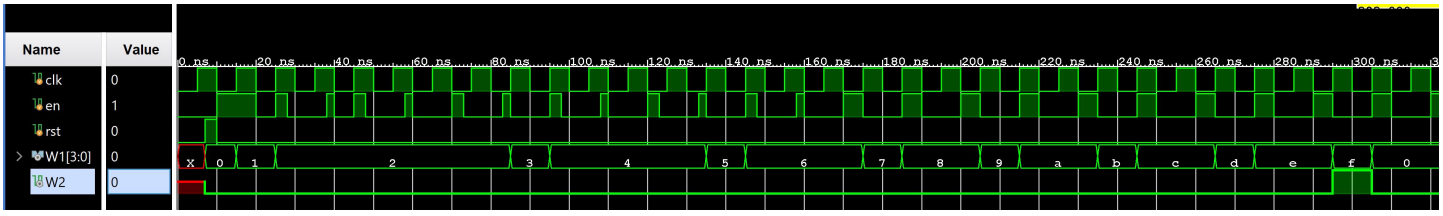

Figure 1: Counter simulation waveform and ERT

Time (ms):	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40
clk	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
rst	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
data	0	1	1	2	3	4	4	5	6	7	8	8	9	a	b	c	c	d	e	e	f
hexdec	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	0	0	0
seg	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0
dp	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
an	e	e	e	e	e	d	d	d	7	7	7	e	e	e	d	d	d	b	b	b	7

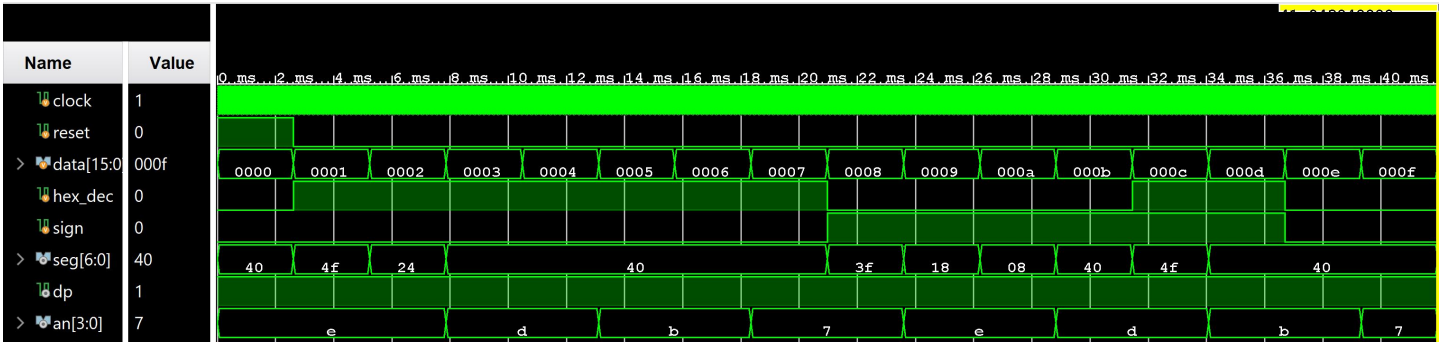

Figure 2: sseg4TDM simulation waveform and ERT

Code

Listing 1: Counter Module

```
‘timescale 1ns / 1ps
// ELC 2137 Jake Simmons 2020-7-4

module Counter #(parameter N=1) (
   input clk, rst, en,
   output [N-1:0] count ,
   output tick
   );
   // internal signals
   reg [N-1:0] Q_reg , Q_next;
   // register (state memory)
   always @(posedge clk, posedge rst)
      begin
         if (rst)
            Q_reg <= 0;
         else
            Q_reg <= Q_next;
      end
   // next -state logic
   always @*
      begin
         if (en)
            Q_next = Q_reg + 1;
         else
            Q_next = Q_reg; // no change
      end
   // output logic
   assign count = Q_reg;
   assign tick = (Q_reg=={N{1'b1}}) ? 1'b1 : 1'b0;
   endmodule // counter
```

Listing 2: Counter Test Bench

```
‘timescale 1ns / 1ps
//ELC 2137 Jake Simmons 2020-7-4

module counter_test();
   reg clk, en, rst;
   wire [3:0] W1;
   wire W2;

   Counter #(.N(4)) count1( .tick(W2), .clk(clk),

   .en(en), .rst(rst), .count(W1) );



   // clock runs continuously

   always begin
```

```
    clk = ~clk; #5;

end

// this block only runs once

initial begin

    clk = 0; en = 0; rst = 0; #7;
    rst = 1; #3; // reset
    en = 1; rst = 0; #10;
    en = 0; #5;
    en = 1; #3;
    en = 0; #10;
    en = 1; #2;
    en = 0; #5;
    en = 1; #3;
    en = 0; #10;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 0; #10;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 0; #10;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 0; #10;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 0; #10;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 1; #2;
    en = 0; #10;
    en = 1; #3;
    en = 1; #2;
```

```
      en = 0; #10;
      en = 1; #3;
      en = 1; #2;
      en = 0; #10;
      en = 1; #3;
      en = 1; #2;
      en = 0; #10;
      en = 1; #3;
      en = 1; #2;
      en = 0; #10;
      en = 1; #3;
      en = 1; #2;
      en = 0; #10;
      en = 1; #3;

   $finish;

   end
endmodule
```

Listing 3: sseg4TDM Module

```
`timescale 1ns / 1ps
//ELC 2137 2020-4-7

module sseg4_TDM(
   input clock,
   input reset,
   input [15:0] data,
   input hex_dec,
   input sign,
   output [6:0] seg,
   output dp,
   output [3:0] an
   );

   wire [1:0] digit_sel;
   wire [15:0] W1 ;
   wire [15:0] W2 ;
   wire [3:0] W3;
   wire [6:0] W4;
   wire [3:0] W5;
   wire W6;
   wire [1:0] W7;

   Counter #(.N(18)) timer( .clk(clock),.en(1'b1), .tick(W7), .rst(reset))
      ;

   Counter #(.N(2)) counter2( .clk(W7),.en(1'b1),.count(digit_sel), .rst(
      reset) );

   BCD11_2 B1( .in11(data[10:0]), .out11(W1));

   mux2 #(.N(16)) B2( .in0(data), .in1(W1), .sel(hex_dec), .out(W2));
```

```
   mux4 B3( .in0(W2[3:0]), .in1(W2[7:4]), .in2(W2[11:8]), .in3(W2[15:12]),
       .sel(digit_sel), .out(W3));

   sseg_decoder B5( .num(W3), .sseg(W4));

   mux2 #(.N(7)) B6( .in0(W4), .in1(7'b0111111), .out(seg) , .sel(W6));

   and G2( W6, sign, ~W5[3]);

   annode_decoder B7( .in(digit_sel), .out(W5));

   assign dp = 1;
   assign an = W5;
endmodule
```

Listing 4: sseg4TDM Test Bench

```
`timescale 1ns / 1ps
//ELC 2137 2020-7-4

module sseg4_TDM_test();
   reg clock;
   reg reset;
   reg [15:0] data;
   reg hex_dec;
   reg sign;
   wire [6:0] seg;
   wire dp;
   wire [3:0] an;

   sseg4_TDM sseg4( .clock(clock), .reset(reset), .data(data), .hex_dec(
      hex_dec),
   .sign(sign), .seg(seg), .dp(dp), .an(an));



   // clock runs continuously

   always begin

      clock = ~clock; #10;

   end

   // this block only runs once

   initial begin
      data = 0; hex_dec = 0; reset = 1;  clock = 0; sign = 0; #2621440;
      data = 1; hex_dec = 1; reset = 0;  sign = 0; #2621440;
      data = 2; hex_dec = 1; reset = 0;  sign = 0;  #2621440;
      data = 3; hex_dec = 1; reset = 0;  sign = 0;  #2621440;
      data = 4; hex_dec = 1; reset = 0;  sign = 0;  #2621440;
      data = 5; hex_dec = 1; reset = 0;  sign = 0;  #2621440;
```

```
      data = 6; hex_dec = 1; reset = 0;  sign = 0;  #2621440;
      data = 7; hex_dec = 1; reset = 0;  sign = 0;  #2621440;
      data = 8; hex_dec = 0; reset = 0;  sign = 1;  #2621440;
      data = 9; hex_dec = 0; reset = 0;  sign = 1;  #2621440;
      data = 10; hex_dec = 0; reset = 0;  sign = 1;  #2621440;
      data = 11; hex_dec = 0; reset = 0;  sign = 1;  #2621440;
      data = 12; hex_dec = 1; reset = 0;  sign = 1;  #2621440;
      data = 13; hex_dec = 1; reset = 0;  sign = 1;  #2621440;
      data = 14; hex_dec = 0; reset = 0;  sign = 0;  #2621440;
      data = 15; hex_dec = 0; reset = 0;  sign = 0;  #2621440;

   $finish;

   end
endmodule
```

Listing 5: calclab10 Module

```
‘timescale 1ns / 1ps
//ELC 2137 Jake Simmons 2020-4-8

module calc_lab10(
   input hex_dec,
   input [17:0] clock,
   input sign,
   input btnU,
   input btnD,
   input [11:0] sw,
   input clk,
   input btnC,
   output [15:0] led,
   output [6:0] seg,
   output dp,
   output [3:0] an
   );
   wire [7:0] W1;
   wire [7:0] W2;

   sseg4_TDM disp_unit( .data({W2, 8'b00000000}), .hex_dec(hex_dec),
   .reset(btnC), .clock(clock), .sign(sign),
   .seg(seg), .dp(dp), .an(an));

   top_lab9 calc_unit( .btnU(btnU), .btnD(btnD), .sw(sw),
   .clk(clk), .btnC(btnC), .led({W1, W2}) );

   assign led[7:0] = W1;
endmodule
```