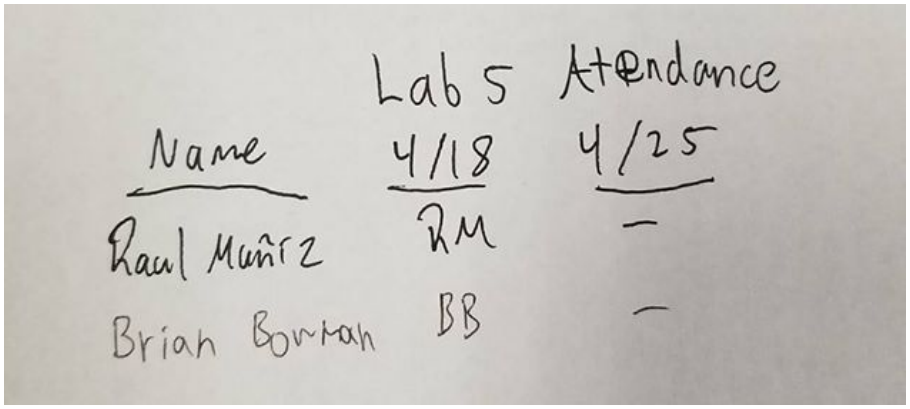
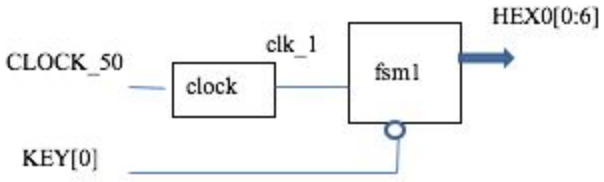


Part 1

I. **Problem:** Animate a 7-segment display to show a circular pattern.

II. **Conceptual Design:**



III. **Verilog Design:**

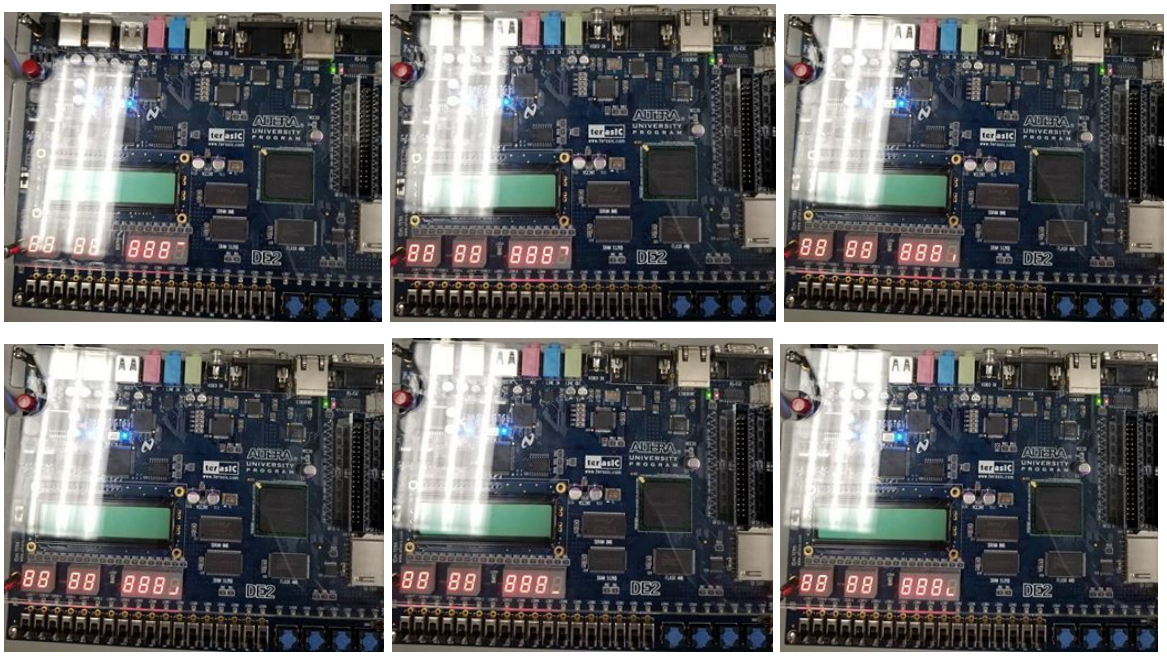
```
module lab5part1 (CLOCK_50, KEY, HEX0);
      input CLOCK_50;
      input [0:0] KEY;

      output [6:0] HEX0;

      wire clock_1Hz;

      reg [6:0] q, q_star;

      localparam [6:0]
      a  = 7'b1111110,
      ab = 7'b1111100,
      b  = 7'b1111101,
      bc = 7'b1111001,
      c  = 7'b1111011,
      cd = 7'b1110011,
      d  = 7'b1110111,
      de = 7'b1100111,
      e  = 7'b1101111,
```

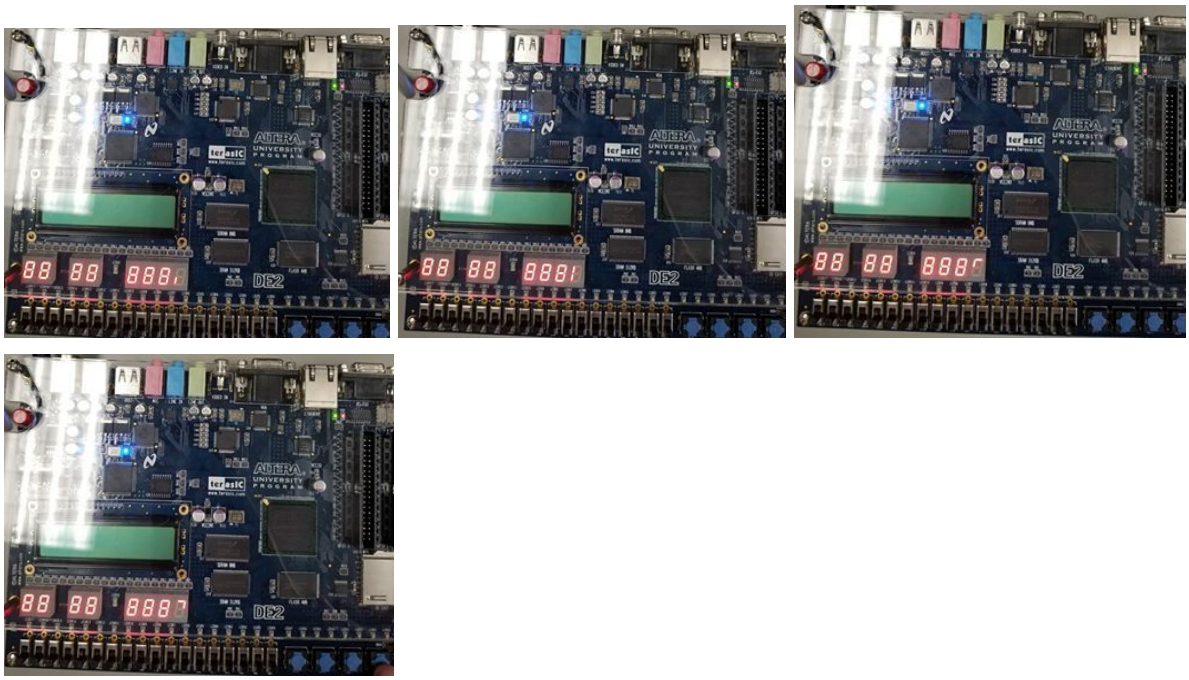
```
      ef = 7'b1001111,
      f  = 7'b1011111,
      fa = 7'b1011110;

      clock_div M0 (CLOCK_50, 25000000, clock_1Hz);

      assign HEX0 = q;

      always @ (*) begin
      case (q)
            a:  q_star = ab;
            ab: q_star = b;
            b:  q_star = bc;
            bc: q_star = c;
            c:  q_star = cd;
            cd: q_star = d;
            d:  q_star = de;
            de: q_star = e;
            e:  q_star = ef;
            ef: q_star = f;
            f:  q_star = fa;
            fa: q_star = a;
      endcase
      end

      always @ (posedge clock_1Hz, negedge KEY[0]) begin
      if (!KEY[0])
            q <= a;
      else
            q <= q_star;
      end
endmodule

// output frequency = clk_in frequency / (2 * clkscale)
module clock_div (clk_in, clkscale, clk_out);
      input clk_in;
      input [31:0] clkscale;
      output reg clk_out;

      reg [31:0] clkq = 0;

      always @ (posedge clk_in) begin
      clkq = clkq + 1;
```

```
        if (clkq == clkscale) begin
              clk_out = ~clk_out;
              clkq = 0;
        end
        end
endmodule
```

IV. Lab Procedure: After flashing the board, we verified the correct pattern was being produced on the 7-segment display. Then we verified the reset button worked properly.

Pictures





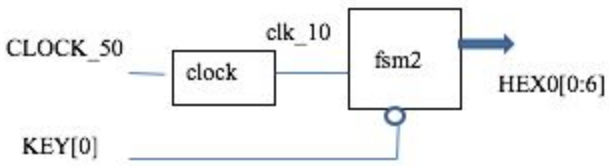
Troubleshooting: No issues.

Conclusion: We learned how to implement a finite-state machine on the DE2 board, and use it to produce an animation on a 7-segment display.

Part 2

I. **Problem:** Animate a 7-segment display to show a circular pattern, with some animation frames displaying for half as long as others.

II. **Conceptual Design:**



III. **Verilog Design:**

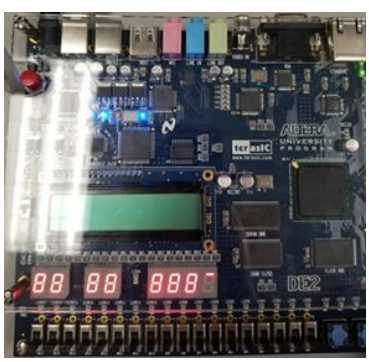
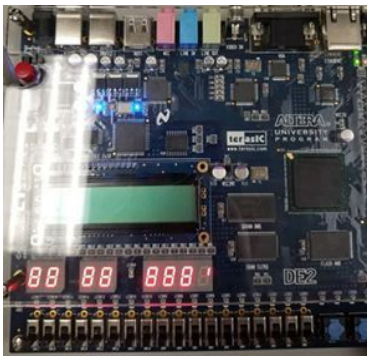
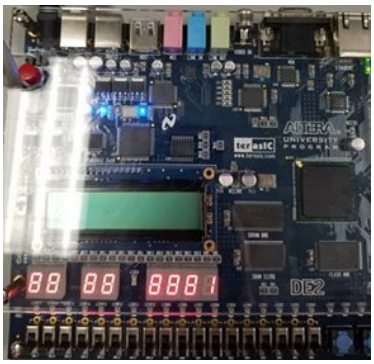
```
module lab5part2 (CLOCK_50, KEY, HEX0);
      input CLOCK_50;
      input [0:0] KEY;

      output [6:0] HEX0;
```

```
wire clock_10Hz;

reg [6:0] q, q_star;
reg [3:0] step;

localparam [6:0]
a  = 7'b1111110,
ab = 7'b1111100,
b  = 7'b1111101,
bc = 7'b1111001,
c  = 7'b1111011,
cd = 7'b1110011,
d  = 7'b1110111,
de = 7'b1100111,
e  = 7'b1101111,
ef = 7'b1001111,
f  = 7'b1011111,
fa = 7'b1011110;

clock_div M0 (CLOCK_50, 2500000, clock_10Hz);

assign HEX0 = q;

always @ (*) begin
case (q)
      a:  q_star = ab;
      ab: q_star = b;
      b:  q_star = bc;
      bc: q_star = c;
      c:  q_star = cd;
      cd: q_star = d;
      d:  q_star = de;
      de: q_star = e;
      e:  q_star = ef;
      ef: q_star = f;
      f:  q_star = fa;
      fa: q_star = a;
endcase
end

always @ (posedge clock_10Hz, negedge KEY[0]) begin
if (!KEY[0]) begin
```

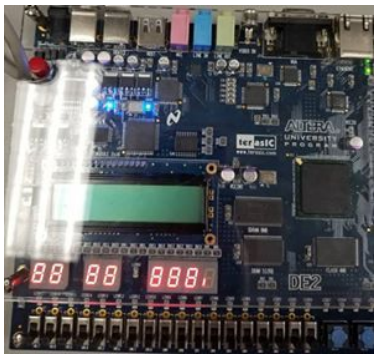
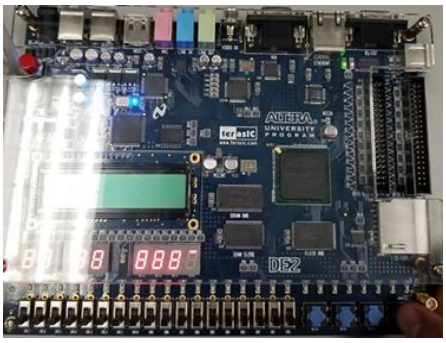
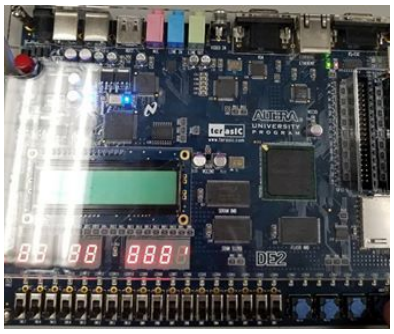
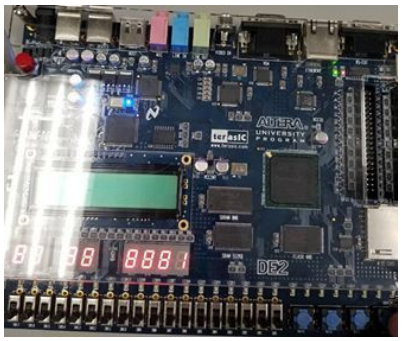
```
            q <= a;
            step <= 0;
      end else begin
            if (step == 9 || (step == 4 && (q == ab || q == bc || q ==
cd || q == de || q == ef || q == fa))) begin
                  q <= q_star;
                  step <= 0;
            end else begin
                  step <= step + 1;
            end
      end
      end
endmodule
```

IV. **Lab Procedure:** Our procedure was the same as in part 1, though this time we made sure the delay between transitions when two segments were lit was half as long as when one segment was lit.

Pictures


Lab5
Brian Bowman
Raul Muniz

Troubleshooting: No issues.

Conclusion: We learned how to modify our finite-state machine implementation so that different states have different durations, to produce a pattern on the 7-segment display with a different rhythm.