Taylor Cowley
EE 220 May 25 2016
Lab 09 Counters State Machine Design

Programmable timer calculations for 10Hz signal
50MHz / 10Hz = 5000000 (number to give timer)

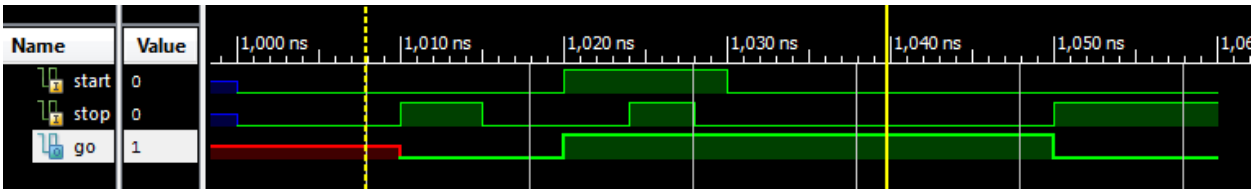
SR Latch Verilog Module

```
module latch(go, start, stop);
	output reg go;
	input start, stop;

	always@(go, start, stop)
	begin
		if(start)
			go = 1;
		else
			if(stop)
				go = 0;
			else
				go = go;
	end
endmodule
```

SR Latch TCL file

```
isim force add start 0 -time 0 -value 1 -time 20ns -value 0 -time 30ns
isim force add stop 0 -time 0 -value 1 -time 10ns -value 0 -time 15ns -value 1 -time 24ns -
value 0 -time 28ns -value 1 -time 50ns
run 60ns
```

SR Latch Simulation



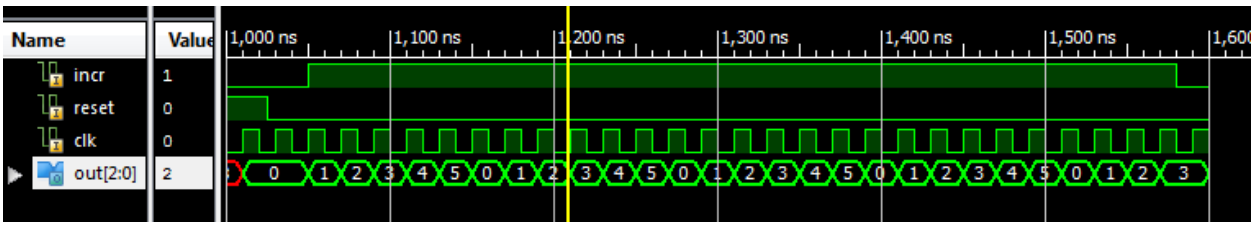
MOD6 Verilog Module

```
module count6(out, incr, reset, clk);
	output reg[2:0] out;
	input incr, reset, clk;

	always@(posedge clk)
	begin
		if(reset)
			out = 0;
		else if(incr)
			if(out == 3'd5)
				out = 0;
			else
				out = out + 1;
		else
			out = out;
	end
endmodule
```

MOD6 TCL file

```
isim force add clk 0 -time 0 -value 1 -time 10ns -repeat 20ns
isim force add incr 0 -time 0 -value 1 -time 50ns -value 0 -time 580ns
isim force add reset 1 -time 0 -value 0 -time 25ns
run 600ns
```

MOD6 Simulation Waveform



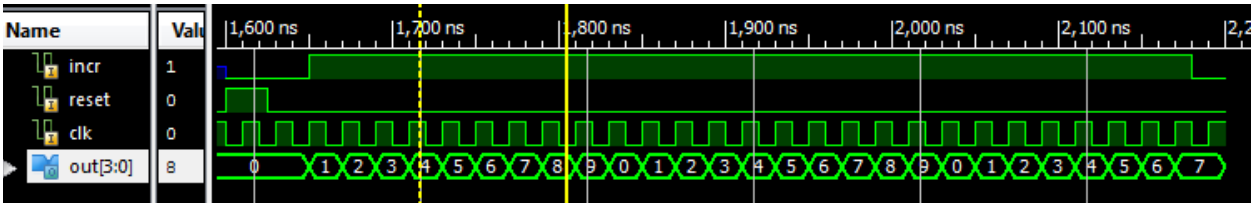
MOD10 Verilog module

```
module count10(out, incr, reset, clk);
	output reg[3:0] out;
	input incr, reset, clk;

	always@(posedge clk)
	begin
		if(reset)
			out = 0;
		else if(incr)
			if(out == 4'd9)
				out = 0;
			else
				out = out + 1;
			else
				out = out;
	end
endmodule
```

MOD10 TCL file

```
isim force add clk 0 -time 0 -value 1 -time 10ns -repeat 20ns
isim force add incr 0 -time 0 -value 1 -time 50ns -value 0 -time 580ns
isim force add reset 1 -time 0 -value 0 -time 25ns
run 600ns
```

MOD10 simulation waveform



Counter block Verilog Module

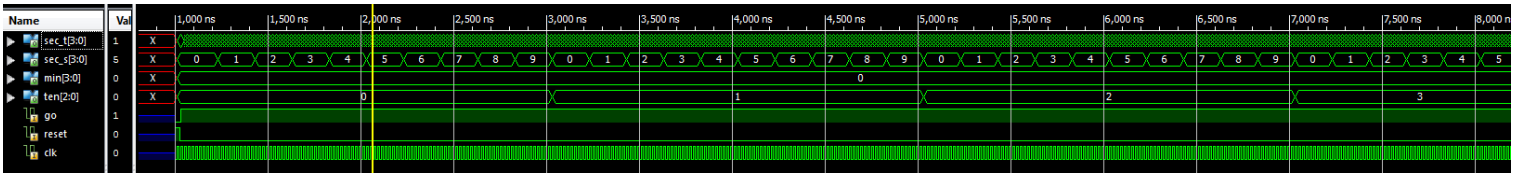
```
module counter_block(sec_t, sec_s, ten, min, go, reset, clk);
	output[3:0] sec_t, sec_s, min;
	output[2:0] ten;
	input go, reset, clk;

	reg incr_t, incr_s, incr_ten, incr_min;
	count10 tenths(sec_t, incr_t, reset, clk);
	count10 seconds(sec_s,incr_s, reset, clk);
	count6  tens(  ten, incr_ten,reset, clk);
	count10 minutes(min,incr_min,reset, clk);

	always@(sec_t, sec_s, ten, min, go)
	begin
		if(go)
		begin
			incr_t = 1;
			if(sec_t == 9)
				begin
				incr_s = 1;
				if(sec_s == 9)
					begin
					incr_ten = 1;
					if(ten == 5)
						begin
						incr_min = 1;
						end
					else
						begin
						incr_min = 0;
```

```
                                        end
                                end
                        else
                                begin
                                incr_ten = 0;
                                incr_min = 0;
                                end
                        end
                else
                        begin
                        incr_s = 0;
                        incr_ten = 0;
                        incr_min = 0;
                        end
                end
        else
        begin
                incr_t = 0;
                incr_s = 0;
                incr_ten = 0;
                incr_min = 0;
        end
    end
endmodule
```

Counter block TCL File

```
isim force add clk 0 -time 0 -value 1 -time 10ns -repeat 20ns
isim force add reset 1 -time 0 -value 0 -time 25ns
isim force add go 0 -time 0 -value 1 -time 30ns
run 6ms
```

Counter block Simulation waveform



TestBench Verilog Module

```
module stopwatch(seg, an, dp, btn, clk);
        output[6:0] seg;
        output[3:0] an;
        output dp;
        input [3:0] btn;
        input clk;

        wire reset;
        wire reset_timer;

        wire pulse;
        wire start, stop;
        prog_timer timer(clk, reset, 1, 24'd5000000, , pulse, );

        wire[3:0] secs, tenths, mins;
        wire[2:0] tens;
        wire go;
        counter_block counter(tenths, secs, tens, mins, go, reset_timer, pulse);


        assign start = btn[3];
        assign stop = btn[2];
        assign reset = btn[1];
        assign reset_timer = btn[0];
        latch l(go, start, stop);

        wire[15:0] data;
        assign data[15:12] = mins; // mins
        assign data[11:8]  = {1'b0, tens}; // tens of seconds
        assign data[7:4]   = secs; // seconds
        assign data[3:0]   = tenths; // tenths of seconds
        seg4x7 seg_controller(seg, an, dp, 4'b1010, data, clk, reset,);
```

```
endmodule
```

TestBench UCF File

```
## clock pin for Nexys 2 Board
NET "clk"   LOC = "B8";
## Buttons
NET "btn<0>" LOC = "B18";
NET "btn<1>" LOC = "D18";
NET "btn<2>" LOC = "E18";
NET "btn<2>" CLOCK_DEDICATED_ROUTE = FALSE;
NET "btn<3>" LOC = "H13";

## 7 segment display
NET "seg<0>" LOC = "L18";
NET "seg<1>" LOC = "F18";
NET "seg<2>" LOC = "D17";
NET "seg<3>" LOC = "D16";
NET "seg<4>" LOC = "G14";
NET "seg<5>" LOC = "J17";
NET "seg<6>" LOC = "H14";
NET "dp"     LOC = "C17";
NET "an<0>" LOC = "F17";
NET "an<1>" LOC = "H17";
NET "an<2>" LOC = "C18";
NET "an<3>" LOC = "F15";
```

Anomalies

This lab was much easier than the last lab. I really had no problems. There is a weird thing though- when I load the FPGA, it starts ticking the stopwatch immediately. I tried assigning an initial value to the go signal, but the synthesizer doesn't support that. I don't know why it does this.