Example 62: Traffic Lights

It is often useful to be able to sequence through an arbitrary number of states, staying in each state an arbitrary amount of time. For example, consider the set of traffic lights shown in Figure 8.13. The lights are assumed to be at a four-way intersection with one street going north-south and the other road going east-west.

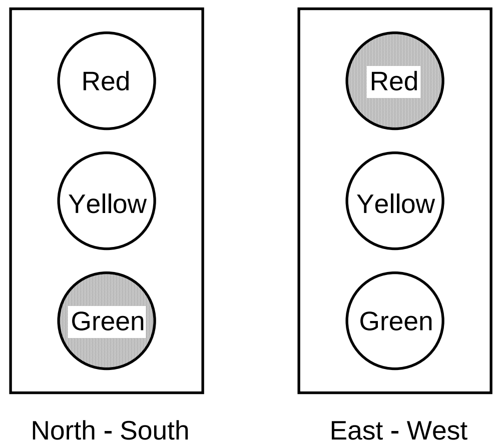

Figure 8.13 Six colored LEDs can represent a set of traffic lights

To simulate these traffic lights we will use the red, yellow, and green LEDs connected to *ld*[7:2] on the BASYS board and cycle through the six states shown in Table 8.2. A state diagram for controlling these traffic lights is shown in Fig. 8.14. If we use a 3 Hz clock to drive this state diagram then a delay of 1 second is achieved by staying in a state for three clock cycles. Similarly, a delay of 5 second is achieved by staying in a state for fifteen clock cycles. The *count* variable in Fig. 8.14 will be reset to zero when moving to the next state after a timeout.

Listing 8.6 is a Verilog program that implements the state diagram in Fig. 8.14 and its simulation is shown in Fig. 8.15. Because we need a counter for the delay count it is more convenient in this case to combine the state register and combinational modules C1 in the Moore machine in Fig. 8.3 into a single sequential *always* block as shown in Listing 8.6. Note in this case we use only a single *state* variable.

To generate the 3 Hz signal we will use the version of *clkdiv* shown in Listing 8.7. The top-level Verilog program is given in Listing 8.8.

Table 8.2 Traffic Light States

State	North - South	East - West	Delay (sec.)
0	Green	Red	5
1	Yellow	Red	1
2	Red	Red	1
3	Red	Green	5
4	Red	Yellow	1
5	Red	Red	1

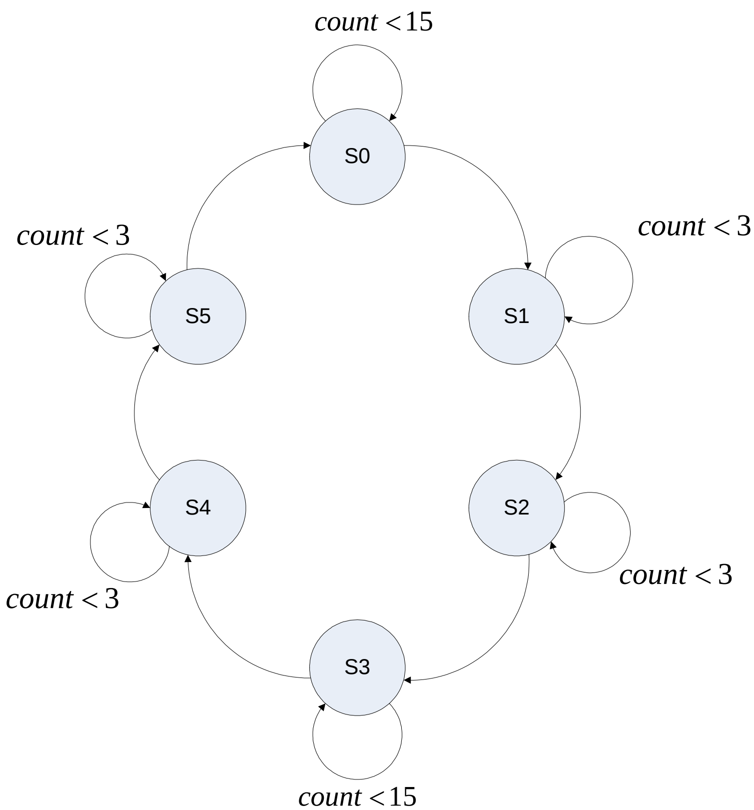

Figure 8.14 State diagram for controlling traffic lights

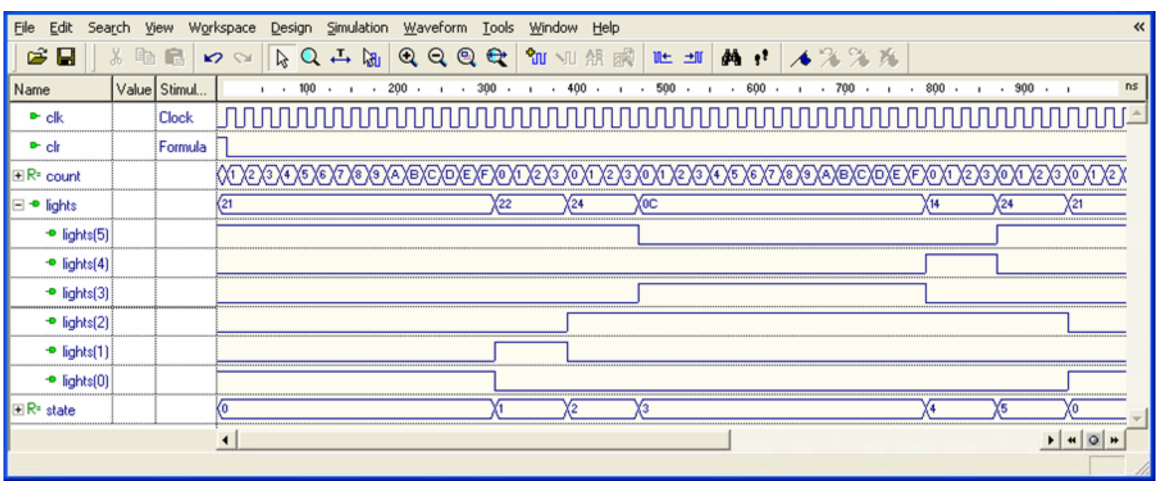

Figure 8.15 Simulation of the Verilog program in Listing 8.4

Listing 8.6 traffic.v

```
// Example 62a: traffic lights
module traffic (
input wire clk ,
input wire clr ,
output reg [5:0] lights
);
reg[2:0] state;
reg[3:0] count;

parameter  S0 = 3'b000, S1 =3'b001, S2 = 3'b010, // states
           S3 = 3'b011, S4 = 3'b100, S5 = 3'b101;
parameter  SEC5 = 4'b1111, SEC1 = 4'b0011;       // delays

always @(posedge clk or posedge clr)
  begin
      if (clr == 1)
        begin
            state <= S0;
            count <= 0;
        end
      else
        case(state)
            S0: if(count < SEC5)
                    begin
                      state <= S0;
                      count <= count + 1;
                    end
                  else
                    begin
                      state <= S1;
                      count <= 0;
                    end
            S1: if(count < SEC1)
                    begin
                      state <= S1;
                      count <= count + 1;
                    end
                  else
                    begin
                      state <= S2;
                      count <= 0;
                    end
            S2: if(count < SEC1)
                    begin
                      state <= S2;
                      count <= count + 1;
                    end
                  else
                    begin
                      state <= S3;
                      count <= 0;
                    end
```

Listing 8.6 (cont.) traffic.v

```
            S3: if(count < SEC5)
                    begin
                      state <= S3;
                      count <= count + 1;
                    end
                  else
                    begin
                      state <= S4;
                      count <= 0;
                    end
            S4: if(count < SEC1)
                    begin
                      state <= S4;
                      count <= count + 1;
                    end
                  else
                    begin
                      state <= S5;
                      count <= 0;
                    end
            S5: if(count < SEC1)
                    begin
                      state <= S5;
                      count <= count + 1;
                    end
                  else
                    begin
                      state <= S0;
                      count <= 0;
                    end
      default state <= S0;
   endcase
  end

 always @(*)
   begin
      case(state)
            S0: lights = 6'b100001;
            S1: lights = 6'b100010;
            S2: lights = 6'b100100;
            S3: lights = 6'b001100;
            S4: lights = 6'b010100;
            S5: lights = 6'b100100;
            default lights = 6'b100001;
     endcase
  end
endmodule
```

Listing 8.7 clkdiv.v

```
// Example 62b: clock divider
module clkdiv (
input wire clk ,
input wire clr ,
output wire clk3
);
reg [24:0] q;

//    25-bit counter
always @(posedge clk or posedge clr)
  begin
    if(clr == 1)
       q <= 0;
    else
       q <= q + 1;
  end

assign clk3 = q[24];      // 3 Hz

endmodule
```

Listing 8.8 traffic_lights_top.v

```
// Example 62: traffic_lights_top
module traffic_lights_top (
input wire clk ,
input wire [3:3] btn ,
output wire [7:2] ld
);
wire clk3;
wire clr;

assign clr = btn[3];

clkdiv U1 (.clk(clk),
      .clr(clr),
      .clk3(clk3)
);

traffic U2 (.clk(clk3),
      .clr(clr),
      .lights(ld)
);

endmodule
```