Jorge Avila

Dr. Bill Carroll

Advanced Digital Logic

06 February 2021

Knight Rider Flash Report

Jorge Avila

Top Module:

```
module KnightRiderFlasher(

    // I will need CLOCK_50 -> PIN_AF14
    // On/Off Toggle Module
    // Clock Divider
    // Up/Down Counter
    // Module Instantiation
    // Pin Assignment -> The 0:9 LEDR on the DE1-Soc board

    input OnOff, //key1
    input Clock50, //clock_50
    output [9:0] LEDRArray); //the 10 LEDR on the board

    wire clock_toggle;
    wire clock_final;
    reg direction = 0; //direction where it will start
    wire [3:0] ledNumber;

    ToggleLatch toggy (OnOff,Clock50,clock_toggle);

    divideX d(clock_toggle,clock_final);

    UpDownCounter udc(clock_final,direction,OnOff,ledNumber);

    always @ (ledNumber)
        begin
            if(ledNumber == 15)
              direction = 0;
            if(ledNumber == 10)
                direction = 1;
        end

    decoder2N d2n(ledNumber,clock_toggle,LEDRArray);

endmodule
```

On-Off Toggle Module:

```
//OnOff Toggle Latch. Assumes an normally-on push button switch for OnOff.
module ToggleLatch (
      input OnOff, IN,
      output OUT);

      reg state, nextstate;
      parameter ON= 1, OFF= 0;

            always @ (negedge OnOff)
                  state <= nextstate;
            always @ (state)
                        case(state)
                              OFF: nextstate = ON;
                              ON: nextstate = OFF;
                              //CLR turns the switch off.
                              //Pushing OnOff turns the switch on.
//Pushing OnOff turns the switch off.
                              endcase
      assign OUT = state&IN; //Out = In when switch in on. Otherwise, Out = 0.

endmodule
```

Clock-Divider Module:

```
module divideX (
      input CLK,
      output reg OUT);

      //parameter N = 5000000;
```

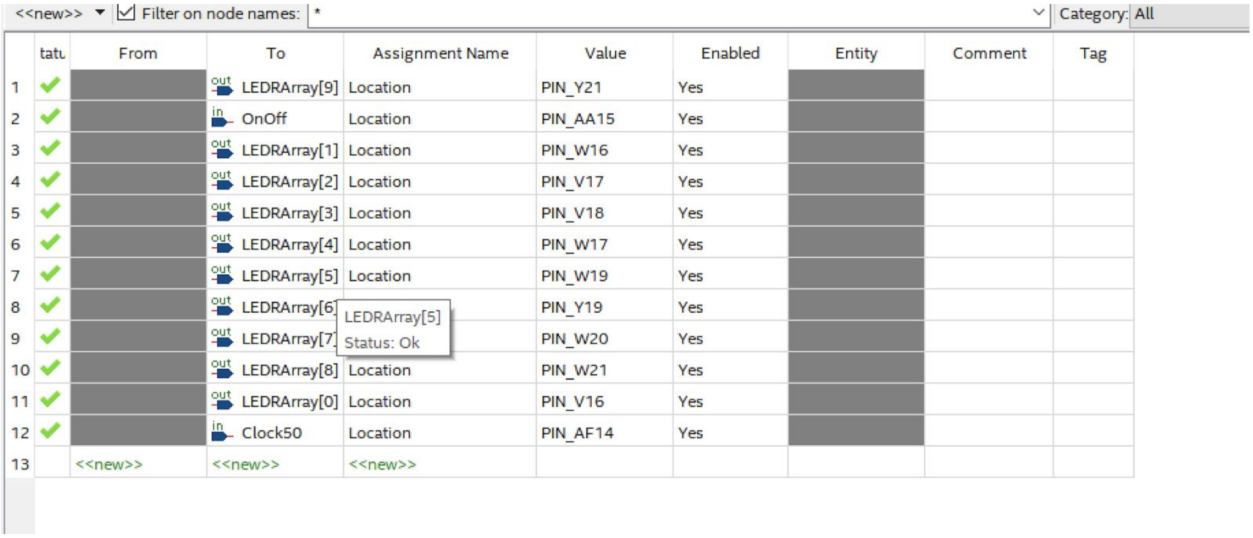
```
        //parameter N = 2500000;
        parameter N = 10000000;
        reg [31:0] count;

        always @ (negedge CLK)
        begin
                count = count + 1;
                if(count >= (N-1))
                        count = 0;
                if(count < (N/2))
                        OUT = 1;
                else
                        OUT=0;
        end
endmodule
```

In this piece of code I just uncommented and re commented the different rates I wanted the LEDR to light up.

Demonstration:

The demonstration is this LINK here. It shows the different rates that were required in the lab report.

Pin Assignments:



<<new>> ▼ ☑ Filter on node names: * Category: All

	tatu	From	To	Assignment Name	Value	Enabled	Entity	Comment	Tag
1	✓		LEDRArray[9]	Location	PIN_Y21	Yes			
2	✓		OnOff	Location	PIN_AA15	Yes			
3	✓		LEDRArray[1]	Location	PIN_W16	Yes			
4	✓		LEDRArray[2]	Location	PIN_V17	Yes			
5	✓		LEDRArray[3]	Location	PIN_V18	Yes			
6	✓		LEDRArray[4]	Location	PIN_W17	Yes			
7	✓		LEDRArray[5]	Location	PIN_W19	Yes			
8	✓		LEDRArray[6]		PIN_Y19	Yes			
9	✓		LEDRArray[7]		PIN_W20	Yes			
10	✓		LEDRArray[8]	Location	PIN_W21	Yes			
11	✓		LEDRArray[0]	Location	PIN_V16	Yes			
12	✓		Clock50	Location	PIN_AF14	Yes			
13		<<new>>	<<new>>	<<new>>					

LEDRArray[5]
Status: Ok

Full Code Below:

```
//Jorge Avila
//mavID: 1001543128
//Assignment #2

//This acts as the main code where everything will be called
module KnightRiderFlasher(

      // I will need CLOCK_50 -> PIN_AF14
      // On/Off Toggle Module
      // Clock Divider
      // Up/Down Counter
      // Module Instantiation
      // Pin Assignment -> The 0:9 LEDR on the DE1-Soc board

      input OnOff, //key1
      input Clock50, //clock_50
      output [9:0] LEDRArray); //the 10 LEDR

      wire clock_toggle;
      wire clock_final;
      reg direction = 0; //direction where it will start
```

```
    wire [3:0] ledNumber;

    ToggleLatch toggy (OnOff,Clock50,clock_toggle);

    divideX d(clock_toggle,clock_final);

    UpDownCounter udc(clock_final,direction,OnOff,ledNumber);

    decoder2N d2n(ledNumber,clock_toggle,LEDRArray);

endmodule


//OnOff Toggle Latch. Assumes an normally-on push button switch for
OnOff.
module ToggleLatch (
    input OnOff, IN,
    output OUT);

    reg state, nextstate;
    parameter ON= 1, OFF= 0;

        always @ (negedge OnOff)
            state <= nextstate;
        always @ (state)
                case(state)
                    OFF: nextstate = ON;
                    ON: nextstate = OFF;
                    //CLR turns the switch off.
                    //Pushing OnOff turns the switch on.
//Pushing OnOff turns the switch off.
                    endcase
    assign OUT = state&IN; //Out = In when switch in on. Otherwise,
Out = 0.

endmodule

module divideX (
    input CLK,
    output reg OUT);

    //parameter N = 5000000;
    //parameter N = 2500000;
    parameter N = 10000000;
    reg [31:0] count;

    always @ (negedge CLK)
    begin
        count = count + 1;
        if(count >= (N-1))
            count = 0;
        if(count < (N/2))
            OUT = 1;
        else
            OUT=0;
    end
endmodule

module UpDownCounter(

    input CLK, UP, clr,
    output reg [N-1:0] COUNT);
    parameter N = 4;
        always @ (posedge CLK, negedge clr)
                if(clr == 0)
                    COUNT <= 0; //clear this b
                else
                    if (UP == 0)
                        COUNT <= COUNT + 1;
                    else
                        COUNT <= COUNT - 1;

endmodule

//N to 2**N decoder - Bill Carroll's
module decoder2N #(parameter N = 4)
```

```
      (input[N-1:0] in,
      input enable,output[2**N-1:0] out);
      assign out = (enable) ? (1 << in) : 0;
endmodule
```