Digital Design Principles

Crosswalk Controller

Name: Rajkaran Singh Grewal
Student No: 8882386
Email: rgrewal2386@conestogac.on.ca

Objective:

- I use VHDL to implement a Crosswalk Controller.
- Introduction to Clock Frequency.

Verilog introduction:

For the program in Verilog, we will be initializing a main module which will have the following parameters: an input clock, four input push-buttons, 6 output arrays for the seven segment arrays, and finally an output led.

Within the module we will have a wire named clock Phase for the clock phase and output of the second module. We will also initialize two registers holding an array of 4 units for counter 1 and counter 2. Where counter 1 will have the initial value of 0000, and the initial value of counter 2 will be 1000 (8 in decimal). We will then initialize four registers for the push state, the state when the push buttons are pressed. We will initialize a module clock cycle for converting the 50 MHz in-built clock to a 1 Hz clock for our use. The module will take an input of our clock and will return the output of the wire clock Phase.

Within the always block where we are checking the posedge of clock Phase, we increment both the counters by one. Then we have an if statement which checks when the counter 1 is greater by the value of 15, and when it is we set the value of counter 1 to zero, we set the value of the first push state to zero and the third push state to zero. We have another if statement which checks when the value of counter 1 is greater than 8 and when it is it will set the value of the first and third push state to zero. The next if statement will check when the counter 2 is greater than 15 and when it is it will set the value of counter 2 to zero and set the value of the second and fourth push state to zero. The next if statement will check when the counter 2 will be greater than 8 and when it is it will set the value of second and fourth push set to zero. The next if statement will check if the value of push button 1 is zero it will set the value of push state 1 to 1 and the led will also be set to one. And in the condition of else we will set the led to zero. The same will continue with the rest of the push buttons. And finally, we will set the value of hex according to the output of the functions from display Seven Segment and

displaySevenSegment2. With these functions the value of the input are check through using a case and the output are set according to which of the input we received.

Finally in the second module we will register a counter holding a value of 26 units to zero. And when the counter value reaches 49,999,998 it will set the output clock to one while when the value reaches 49,999,999 the counter will be reset back to zero and the output, we be set to zero as well.

.

Verilog program screenshot:

```
1   module rgrewallab10Verilog(input CLK,
2                              input pushButton1,
3                              input pushButton2,
4                              input pushButton3,
5                              input pushButton4,
6                              output reg [6:0] hex1,
7                              output reg [6:0] hex2,
8                              output reg [6:0] hex3,
9                              output reg [6:0] hex4,
10                             output reg [6:0] hex5,
11                             output reg [6:0] hex6,
12                             output reg led
13                             );
14      wire clockPhase;
15      reg [3:0] counter = 4'b0000;
16      reg [3:0] counter2 = 4'b1000;
17      reg pushState1 = 0;
18      reg pushState2 = 0;
19      reg pushState3 = 0;
20      reg pushState4 = 0;
21      clockCycle colock(.CLK(CLK),
22                        .clockCycle(clockPhase));
23      always @(posedge clockPhase) begin
24          counter = counter + 1;
25          counter2 = counter2 + 1;
26          if(counter > 15) begin
27              counter = 0;
28              pushState1 = 0;
29              pushState3 = 0;
30          end
```

```
31          if(counter > 8) begin
32              pushState1 = 0;
33              pushState3 = 0;
34          end
35          if(counter2 > 15) begin
36              counter2 = 0;
37              pushState2 = 0;
38              pushState4 = 0;
39          end
40          if(counter2 > 8) begin
41              pushState2 = 0;
42              pushState4 = 0;
43          end
44
45          if(pushButton1 == 0) begin
46              pushState1 = 1;
47              led = 1;
48          end
49          else begin
50              led = 0;
51          end
52
53          if(pushButton2 == 0) begin
54              pushState2 = 1;
55              led = 1;
56          end
57          else begin
58              led = 0;
59          end
60
```

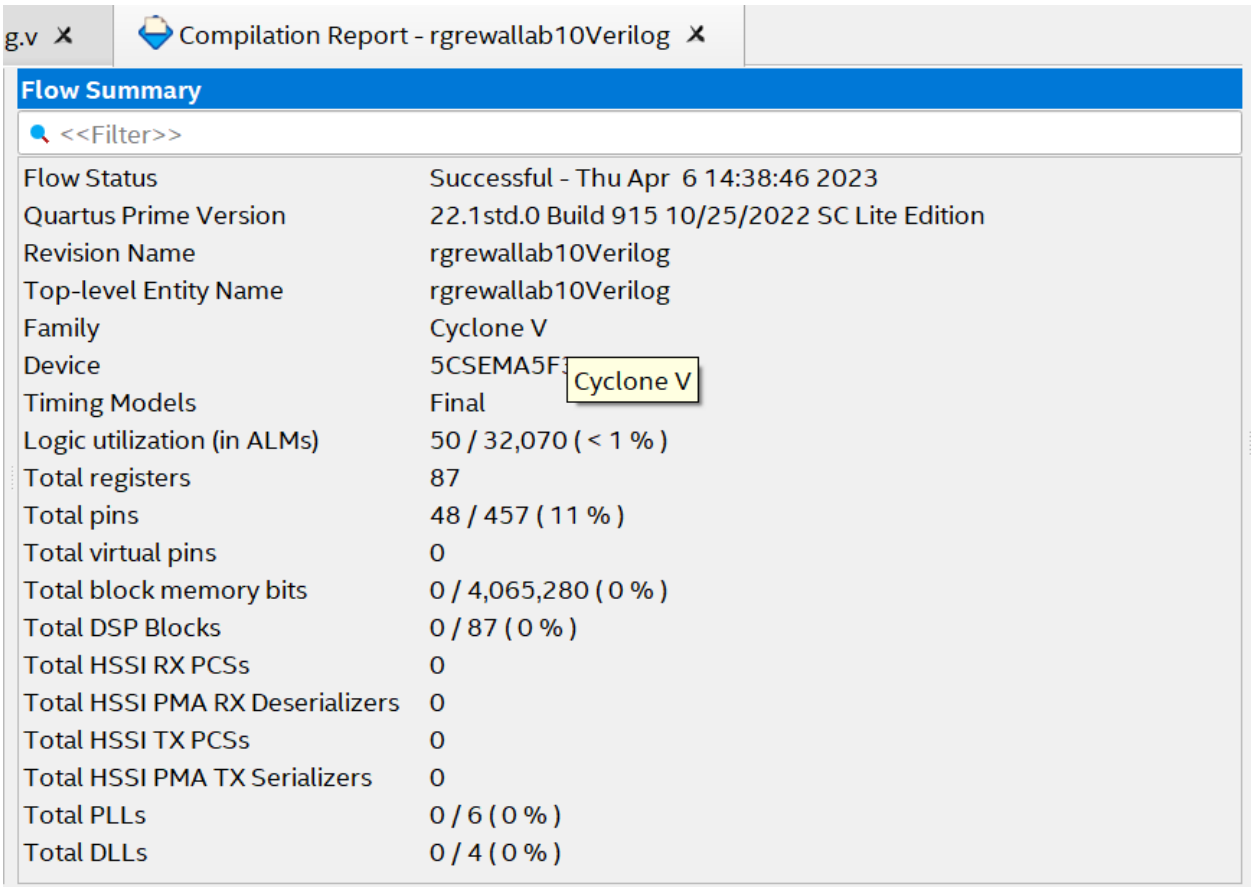
```
61          if(pushButton3 == 0) begin
62              pushState3 = 1;
63              led = 1;
64          end
65          else begin
66              led = 0;
67          end
68
69          if(pushButton4 == 0) begin
70              pushState4 = 1;
71              led = 1;
72          end
73          else begin
74              led = 0;
75          end
76
77          hex1 = displaySevenSegment(counter);
78          hex2 = displaySevenSegment(counter2);
79          hex3 = displaySevenSegment2(pushState1,counter);
80          hex4 = displaySevenSegment2(pushState2, counter2);
81          hex5 = displaySevenSegment2(pushState3, counter);
82          hex6 = displaySevenSegment2(pushState4, counter2);
83      end
84      function [6:0] displaySevenSegment(input [3:0] counter);
85      case(counter)
86          4'b0000: begin
87              displaySevenSegment = 7'b0100001;
88          end
89          4'b0001: begin
90              displaySevenSegment = 7'b0100001;
```

```
91          end
92          4'b0010: begin
93              displaySevenSegment = 7'b0100001;
94          end
95          4'b0011: begin
96              displaySevenSegment = 7'b0100001;
97          end
98          4'b0100: begin
99              displaySevenSegment = 7'b0100001;
100         end
101         4'b0101: begin
102             displaySevenSegment = 7'b1011000;
103         end
104         4'b0110: begin
105             displaySevenSegment = 7'b1011000;
106         end
107         4'b0111: begin
108             displaySevenSegment = 7'b1011000;
109         end
110         default: begin
111             displaySevenSegment = 7'b1111010;
112         end
113     endcase
114   endfunction
115    function [6:0] displaySevenSegment2(input pushState,input [4:0] counter);
116         if(pushState == 0) begin
117             displaySevenSegment2 = 7'b1111111;
118         end
119         else begin
120             case(counter)
```

```
121                 4'b0000:begin
122                     displaySevenSegment2 = 7'b0001111;
123                 end
124                 4'b0001: begin
125                     displaySevenSegment2 = 7'b0100000;
126                 end
127                 4'b0010: begin
128                     displaySevenSegment2 = 7'b0100100;
129                 end
130                 4'b0011: begin
131                     displaySevenSegment2 = 7'b1001100;
132                 end
133                 4'b0100: begin
134                     displaySevenSegment2 = 7'b0000110;
135                 end
136                 4'b0101: begin
137                     displaySevenSegment2 = 7'b0010010;
138                 end
139                 4'b0110: begin
140                     displaySevenSegment2 = 7'b1001111;
141                 end
142                 4'b0111: begin
143                     displaySevenSegment2 = 7'b0000001;
144                 end
145                 default: begin
146                     displaySevenSegment2 = 7'b0000000;
147                 end
148             endcase
149         end
150     endfunction
```

```
151  endmodule
152  module clockCycle(input CLK,
153                    output reg clockCycle);
154      reg [26:0] counter = 0;
155      always @(posedge CLK) begin
156          counter = counter + 1;
157          if(counter == 49_999_999) begin
158              counter = 0;
159              clockCycle = 0;
160          end
161          else if(counter == 49_999_998) begin
162              clockCycle = 1;
163          end
164      end
165  endmodule
```

Verilog compilation report:



Verilog pin planner screenshot:

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard
CLK	Input	PIN_AF14	3B	B3B_N0	PIN_AF14	2.5 V
hex1[6]	Output	PIN_AE26	5A	B5A_N0	PIN_AE26	2.5 V
hex1[5]	Output	PIN_AE27	5A	B5A_N0	PIN_AE27	2.5 V
hex1[4]	Output	PIN_AE28	5A	B5A_N0	PIN_AE28	2.5 V
hex1[3]	Output	PIN_AG27	5A	B5A_N0	PIN_AG27	2.5 V
hex1[2]	Output	PIN_AF28	5A	B5A_N0	PIN_AF28	2.5 V
hex1[1]	Output	PIN_AG28	5A	B5A_N0	PIN_AG28	2.5 V
hex1[0]	Output	PIN_AH28	5A	B5A_N0	PIN_AH28	2.5 V
hex2[6]	Output	PIN_AJ29	5A	B5A_N0	PIN_AJ29	2.5 V
hex2[5]	Output	PIN_AH29	5A	B5A_N0	PIN_AH29	2.5 V
hex2[4]	Output	PIN_AH30	5A	B5A_N0	PIN_AH30	2.5 V
hex2[3]	Output	PIN_AG30	5A	B5A_N0	PIN_AG30	2.5 V
hex2[2]	Output	PIN_AF29	5A	B5A_N0	PIN_AF29	2.5 V
hex2[1]	Output	PIN_AF30	5A	B5A_N0	PIN_AF30	2.5 V
hex2[0]	Output	PIN_AD27	5A	B5A_N0	PIN_AD27	2.5 V
hex3[6]	Output	PIN_AB23	5A	B5A_N0	PIN_AB23	2.5 V
hex3[5]	Output	PIN_AE29	5B	B5B_N0	PIN_AE29	2.5 V
hex3[4]	Output	PIN_AD29	5B	B5B_N0	PIN_AD29	2.5 V
hex3[3]	Output	PIN_AC28	5B	B5B_N0	PIN_AC28	2.5 V
hex3[2]	Output	PIN_AD30	5B	B5B_N0	PIN_AD30	2.5 V
hex3[1]	Output	PIN_AC29	5B	B5B_N0	PIN_AC29	2.5 V
hex3[0]	Output	PIN_AC30	5B	B5B_N0	PIN_AC30	2.5 V
hex4[6]	Output	PIN_AD26	5A	B5A_N0	PIN_AD26	2.5 V
hex4[5]	Output	PIN_AC27	5A	B5A_N0	PIN_AC27	2.5 V
hex4[4]	Output	PIN_AD25	5A	B5A_N0	PIN_AD25	2.5 V
hex4[3]	Output	PIN_AC25	5A	B5A_N0	PIN_AC25	2.5 V
hex4[2]	Output	PIN_AB28	5B	B5B_N0	PIN_AB28	2.5 V
hex4[1]	Output	PIN_AB25	5A	B5A_N0	PIN_AB25	2.5 V
hex4[0]	Output	PIN_AB22	5A	B5A_N0	PIN_AB22	2.5 V
hex5[6]	Output	PIN_AA24	5A	B5A_N0	PIN_AA24	2.5 V
hex5[5]	Output	PIN_Y23	5A	B5A_N0	PIN_Y23	2.5 V
hex5[4]	Output	PIN_Y24	5A	B5A_N0	PIN_Y24	2.5 V
hex5[3]	Output	PIN_W22	5A	B5A_N0	PIN_W22	2.5 V
hex5[2]	Output	PIN_W24	5A	B5A_N0	PIN_W24	2.5 V
hex5[1]	Output	PIN_V23	5A	B5A_N0	PIN_V23	2.5 V
hex5[0]	Output	PIN_W25	5B	B5B_N0	PIN_W25	2.5 V
hex6[6]	Output	PIN_V25	5B	B5B_N0	PIN_V25	2.5 V
hex6[5]	Output	PIN_AA28	5B	B5B_N0	PIN_AA28	2.5 V
hex6[4]	Output	PIN_Y27	5B	B5B_N0	PIN_Y27	2.5 V
hex6[3]	Output	PIN_AB27	5B	B5B_N0	PIN_AB27	2.5 V
hex6[2]	Output	PIN_AB26	5A	B5A_N0	PIN_AB26	2.5 V
hex6[1]	Output	PIN_AA26	5B	B5B_N0	PIN_AA26	2.5 V
hex6[0]	Output	PIN_AA25	5A	B5A_N0	PIN_AA25	2.5 V
led	Output	PIN_V16	4A	B4A_N0	PIN_V16	2.5 V
pushButton1	Input	PIN_AA14	3B	B3B_N0	PIN_AA14	2.5 V
pushButton2	Input	PIN_AA15	3B	B3B_N0	PIN_AA15	2.5 V
pushButton3	Input	PIN_W15	3B	B3B_N0	PIN_W15	2.5 V
pushButton4	Input	PIN_Y16	3B	B3B_N0	PIN_Y16	2.5 V

Conclusion:

Thus we can conclude that by using the programming language of verilog we will set a four set crossWalk using the simplest manner and using a secondary module.