Digital Design Principles

Seven Segment Display

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

Objective:

- Use VHDL to implement a binary to 7 Segment decoder.
- Introduction to the VHDL Case statement.

Truth Table:

Sw3	Sw2	Sw1	Sw0	LED
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	B

1	1	0	0	C
1	1	0	1	D
1	1	1	0	E
1	1	1	1	F

VHDL Introduction:

To make the program in VHDL we will need to initialize the library and entity. Within the entity we will state the 4 input switches and the seven segment display. Within the architecture of the program we will use case and when statement. Where for each different state the four inputs are the according segments of the seven segments display to light up the symbol to show the hexadecimal version of the input. The above truth table is the be followed to know which led is to be set accordingly.

VHDL program screenshot:

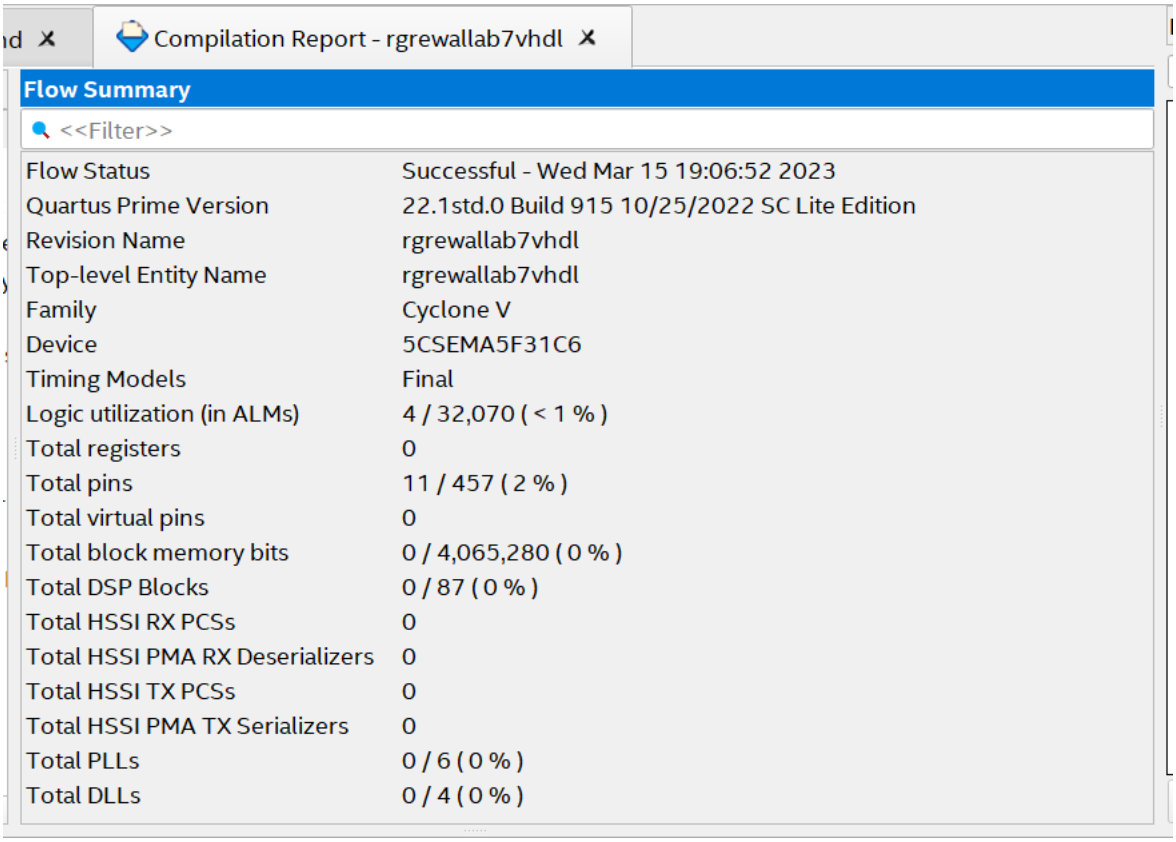
VHDL > rgrewallab7vhdl.vhd

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity rgrewallab7vhdl is
    port(
        Sw: in std_logic_vector(3 downto 0);
        led: out std_logic_vector(6 downto 0)
    );
end rgrewallab7vhdl;

architecture sevenSegmentDisplay of rgrewallab7vhdl is
    begin
        process(Sw) is
            begin
                case Sw is
                    when "0000" =>
                        led <= "0000001";

                    when "0001" =>
                        led <= "1001111";

                    when "0010" =>
                        led <= "0010010";

                    when "0011" =>
                        led <= "0000110";

                    when "0100" =>
                        led <= "1001100";

```

```
31                when "0101" =>
32                    led <= "0100100";
33
34                when "0110" =>
35                    led <= "0100000";
36
37                when "0111" =>
38                    led <= "0001111";
39
40                when "1000" =>
41                    led <= "0000000";
42
43                when "1001" =>
44                    led <= "0001100";
45
46                when "1010" =>
47                    led <= "0001000";
48
49                when "1011" =>
50                    led <= "1100000";
51
52                when "1100" =>
53                    led <= "0110001";
54
55                when "1101" =>
56                    led <= "1000010";
57
58                when "1110" =>
59                    led <= "0110000";
60
```

```
61                      when "1111" =>
62                          led <= "0111000";
63
64                      when others =>
65                          led <= "1111111";
66                  end case;
67          end process;
68  end sevenSegmentDisplay;
```

VHDL compilation report:



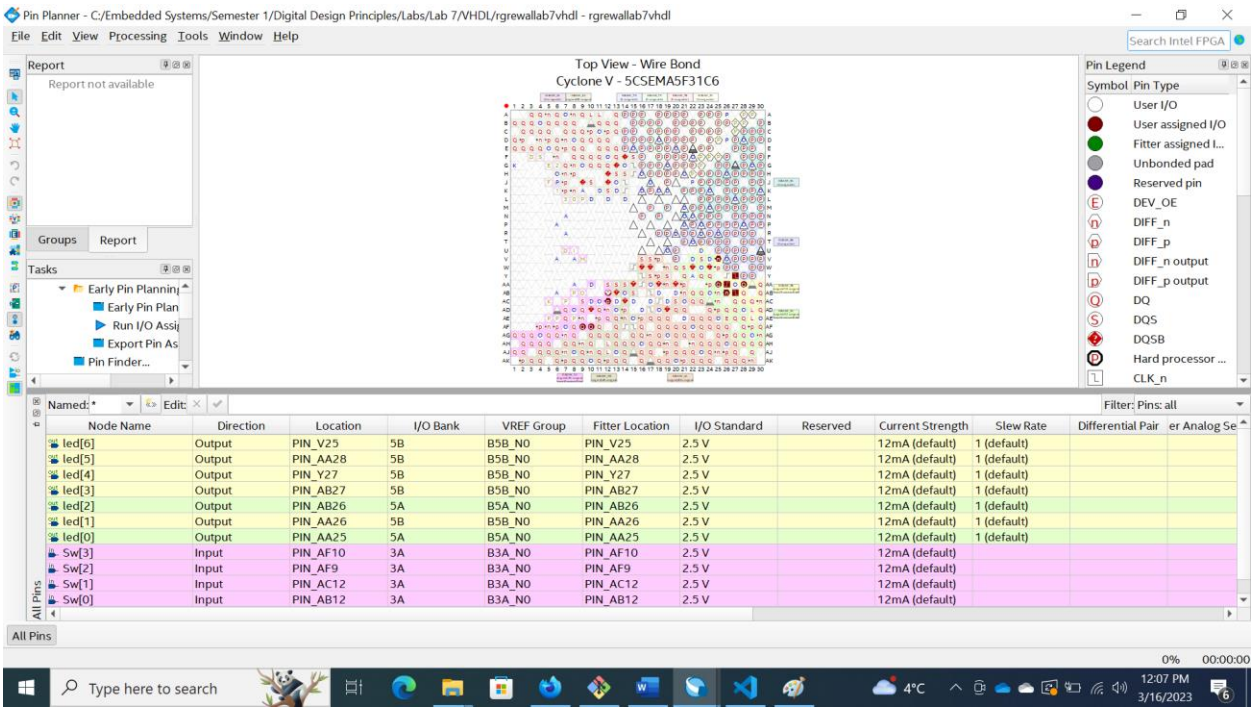
Compilation Report - rgrewallab7vhdl

Flow Summary

<<Filter>>

Flow Status	Successful - Wed Mar 15 19:06:52 2023
Quartus Prime Version	22.1std.0 Build 915 10/25/2022 SC Lite Edition
Revision Name	rgrewallab7vhdl
Top-level Entity Name	rgrewallab7vhdl
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	4 / 32,070 (< 1 %)
Total registers	0
Total pins	11 / 457 (2 %)
Total virtual pins	0
Total block memory bits	0 / 4,065,280 (0 %)
Total DSP Blocks	0 / 87 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0 / 6 (0 %)
Total DLLs	0 / 4 (0 %)

Pin Planner screenshot:



Verilog introduction:

For the program in Verilog we will need to initialize the module with the parameters having the four inputs and a seven segment display. Within the module block we use case where the output will assign the value according to the state of the input which matches to the truth table.

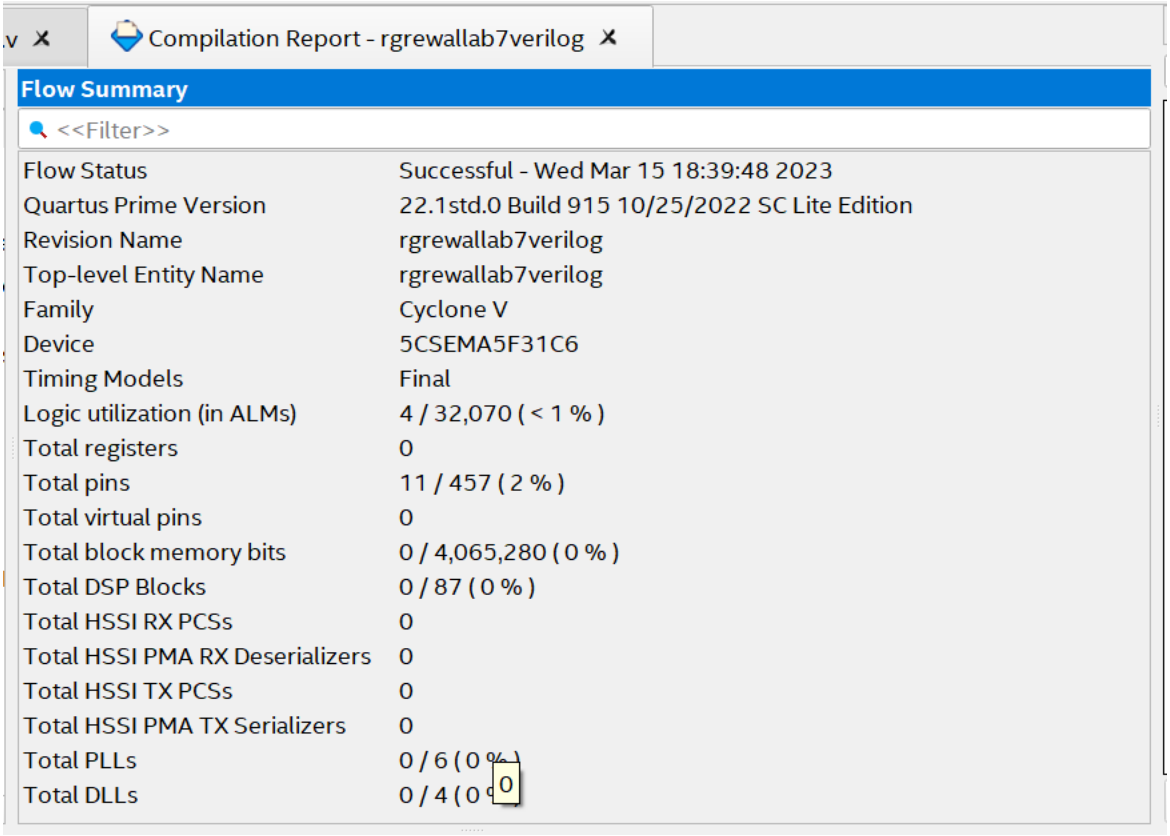
Verilog program screenshot:

Verilog > rgrewallab7verilog.v

```
1   module rgrewallab7verilog(input [3:0] Sw,output reg [6:0] led);
2   always @(Sw) begin
3       case(Sw)
4           4'b0000: begin
5               led = 7'b0000001;
6           end
7           4'b0001: begin
8               led = 7'b1001111;
9           end
10          4'b0010: begin
11              led = 7'b0010010;
12          end
13          4'b0011: begin
14              led = 7'b0000110;
15          end
16          4'b0100: begin
17              led = 7'b1001100;
18          end
19          4'b0101: begin
20              led = 7'b0100100;
21          end
22          4'b0110: begin
23              led = 7'b0100000;
24          end
25          4'b0111: begin
26              led = 7'b0001111;
27          end
28          4'b1000: begin
29              led = 7'b0000000;
30          end
```

```
        4'b1001: begin
            led = 7'b0001100;
        end
        4'b1010: begin
            led = 7'b0001000;
        end
        4'b1011: begin
            led = 7'b1100000;
        end
        4'b1100: begin
            led = 7'b0110001;
        end
        4'b1101: begin
            led = 7'b1000010;
        end
        4'b1110: begin
            led = 7'b0110000;
        end
        4'b1111: begin
            led = 7'b0111000;
        end
        default: begin
            led = 7'b1111111;
        end
    endcase
end
endmodule
```

Verilog compilation report:



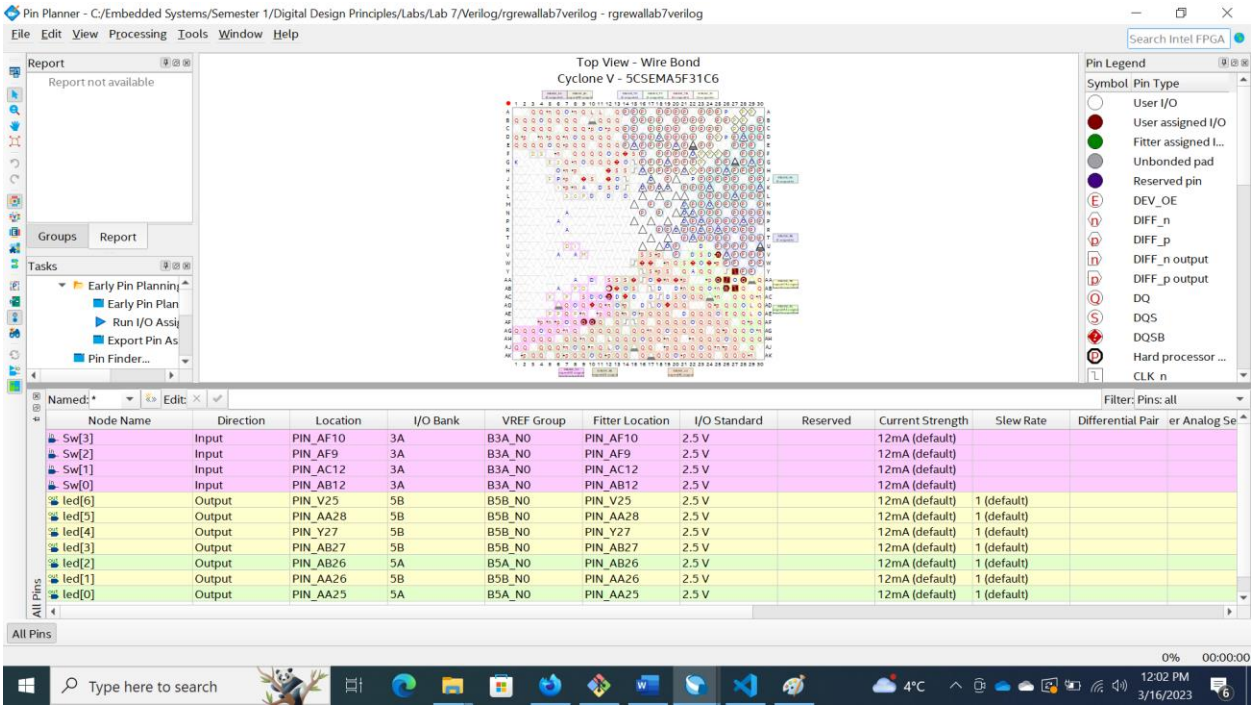
Compilation Report - rgrewallab7verilog

Flow Summary

<<Filter>>

Flow Status	Successful - Wed Mar 15 18:39:48 2023
Quartus Prime Version	22.1std.0 Build 915 10/25/2022 SC Lite Edition
Revision Name	rgrewallab7verilog
Top-level Entity Name	rgrewallab7verilog
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	4 / 32,070 (< 1 %)
Total registers	0
Total pins	11 / 457 (2 %)
Total virtual pins	0
Total block memory bits	0 / 4,065,280 (0 %)
Total DSP Blocks	0 / 87 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0 / 6 (0 %)
Total DLLs	0 / 4 (0 %)

Verilog pin planner screenshot:



Conclusion:

Thus we can conclude that by both the programming languages we will use case statement and the one seven segment display which corresponds to the input will light up. The above truth table will be followed to show a binary to hexacode convertor.