Seven Segment Display

Submitted by: Piyusha Jahagirdar

In this lab we designed, simulate and implemented a seven-segment display on the DE2 board using Quatrus for VHDL. We built a seven-segment display decoder that displayed numbers 0 to 9 and alphabets A to F. It took us 2 hours to complete the design.

Figure 1 shows the segments of the display

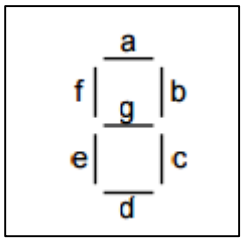

Figure 1

To design the decoder, we had to provide a segment with a 0 to make it glow as the out of the decoder was 'active low output'. Using this information, we designed our truth table for the decoder.

Table 1 shows the truth table for seven-segment display decoder

Hexadecimal Digit	Inputs				Outputs							In Hex
	D_3	D_2	D_1	D_0	S_g	S_f	S_e	S_d	S_c	S_b	S_a	
0	0	0	0	0	1	0	0	0	0	0	0	40
1	0	0	0	1	1	1	1	1	0	0	1	79
2	0	0	1	0	0	1	0	0	1	0	0	24
3	0	0	1	1	0	1	1	0	0	0	0	30
4	0	1	0	0	0	0	1	1	0	0	1	19
5	0	1	0	1	0	0	1	0	0	1	0	12
6	0	1	1	0	0	0	0	0	0	1	0	02
7	0	1	1	1	1	1	1	1	0	0	0	78
8	1	0	0	0	0	0	0	0	0	0	0	00
9	1	0	0	1	0	0	1	1	0	0	0	18
A	1	0	1	0	0	0	0	1	0	0	0	08
B	1	0	1	1	0	0	0	0	0	1	1	03
C	1	1	0	0	0	1	0	0	1	1	1	27
D	1	1	0	1	0	1	0	0	0	0	1	21
E	1	1	1	0	0	0	0	0	1	1	0	06
F	1	1	1	1	0	0	0	1	1	1	0	14

Table 1

Once the table was complete we started writing our code in VHDL using Quartus. Below is our code for Seven-Segment display.

```
--Quartus II VHDL Template

-- Seven Segment

--SevenSegment.vhd

library ieee;

use ieee.std_logic_1164.all;

use ieee.numeric_std.all;


entity SevenSegment is


	port

	(

		SW	: in STD_LOGIC_VECTOR(3 downto 0);

		HEX0 : out STD_LOGIC_VECTOR(6 downto 0)

	);


end entity;


architecture ss of Sevensegment is

begin


	process(SW)

	begin

		case(SW) is

			when "0000" => HEX0 <= "1000000" ;    --0

			when "0001" => HEX0 <= "1111001" ;    --1

			when "0010" => HEX0 <= "0100100" ;    --2

			when "0011" => HEX0 <= "0110000" ;    --3
```

```
                when "0100" => HEX0 <= "0011001" ;    --4
                when "0101" => HEX0 <= "0010010" ;    --5
                when "0110" => HEX0 <= "0000010" ;    --6
                when "0111" => HEX0 <= "1111000" ;    --7
                when "1000" => HEX0 <= "0000000" ;    --8
                when "1001" => HEX0 <= "0011000" ;    --9
                when "1010" => HEX0 <= "0001000" ;    --A
                when "1011" => HEX0 <= "0000011" ;    --b
                when "1100" => HEX0 <= "0100111" ;    --c
                when "1101" => HEX0 <= "0100001" ;    --d
                when "1110" => HEX0 <= "0000110" ;    --E
                when "1111" => HEX0 <= "0001110" ;    --F
                when others => HEX0 <= "0111111" ;
        end case;
    end process;
end ss;
```

Our code as divided in 3 parts:

1. Library and used declarations
2. Entity port declarations
3. Architecture description

The port SW was used for the input for the decoder and HEX0 was used for the output.

Once the code was complete and Compiled successfully we Simulated our code for all input combinations for 0 to F.

Figure 2 shows the simulation wave form.

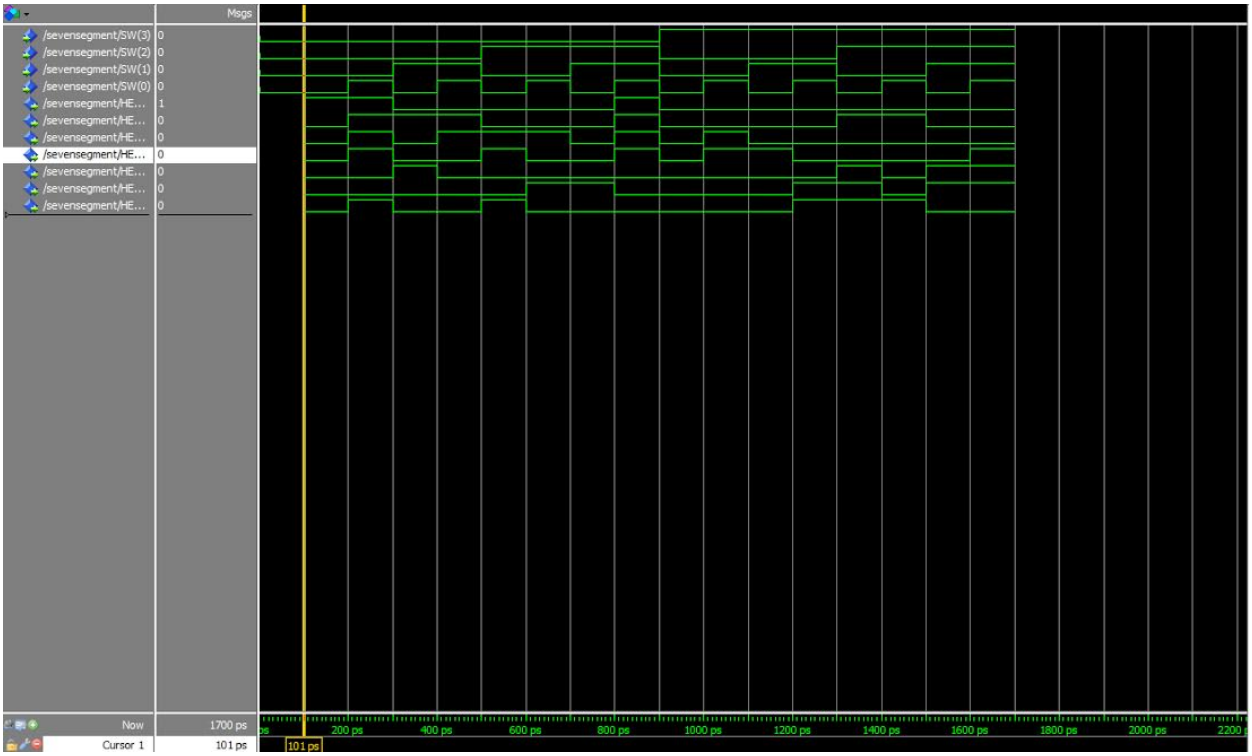

Figure 2

Our design worked successfully on the DE2 board. Below are the pictures of all the outputs from 0 to F

To complete the seven-segment display, it took us 2 hours.