

CS 288:  
BCD to 7 Segment Decoder  
&  
Pseudo Random Number Generator

Dheerendra Singh Rathor  
120050033  
`dheeru.rathor14@gmail.com`

March 4, 2014

## Aim:

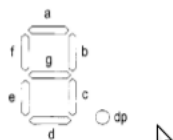
Using Xilinx ISE Tools to simulate BCD to 7 Segment decoder and Pseudo Random Number Generator using behavioral model.

## Procedure:

The procedure for 7 Segment decoder and Pseudo Random Number Generator is given in the following subsections respectively

### 1 BCD to 7 Segment Decoder

For making BCD to 7 Segment Decoder, I used to logic vectors viz. bcd and ouput respectively 3 downto 0 and 6 downto 0. Logic vector output is taken as "gfedcba".



(a) Diagram of a seven-segment LED display



(b) Hexadecimal digit patterns

Values to each case is given using the truth table given below.

S. No.	output value	bcd	output vector
1.	0	0000	0111111
2.	1	0001	0000110
3.	2	0010	1011011
4.	3	0011	1001111
5.	4	0100	1100110
6.	5	0101	1101101
7.	6	0110	1111101
8.	7	0111	0000111
9.	8	1000	1111111
10.	9	1001	1101111
11.	a	1010	1011111
12.	b	1011	1111100
13.	c	1100	0111001
14.	d	1101	1011110
15.	e	1110	1111001
16.	f	1111	1110001

This code is written in sequential behaviour using case when statements. Timing diagrams and code are included in coming sections.

## 2 Pseudo Random Number Generator

Pseudo Random Number Generator is written using 1 generator variable named "gen". 1 output vector named output which is the random number vector. and two incode variable are used as output\_temp and temp.

if gen is 1 then output will be a random value.

This code is written in sequential behaviour using case - when statement. Timing diagram and code are included in coming sections.

## Timing Diagrams

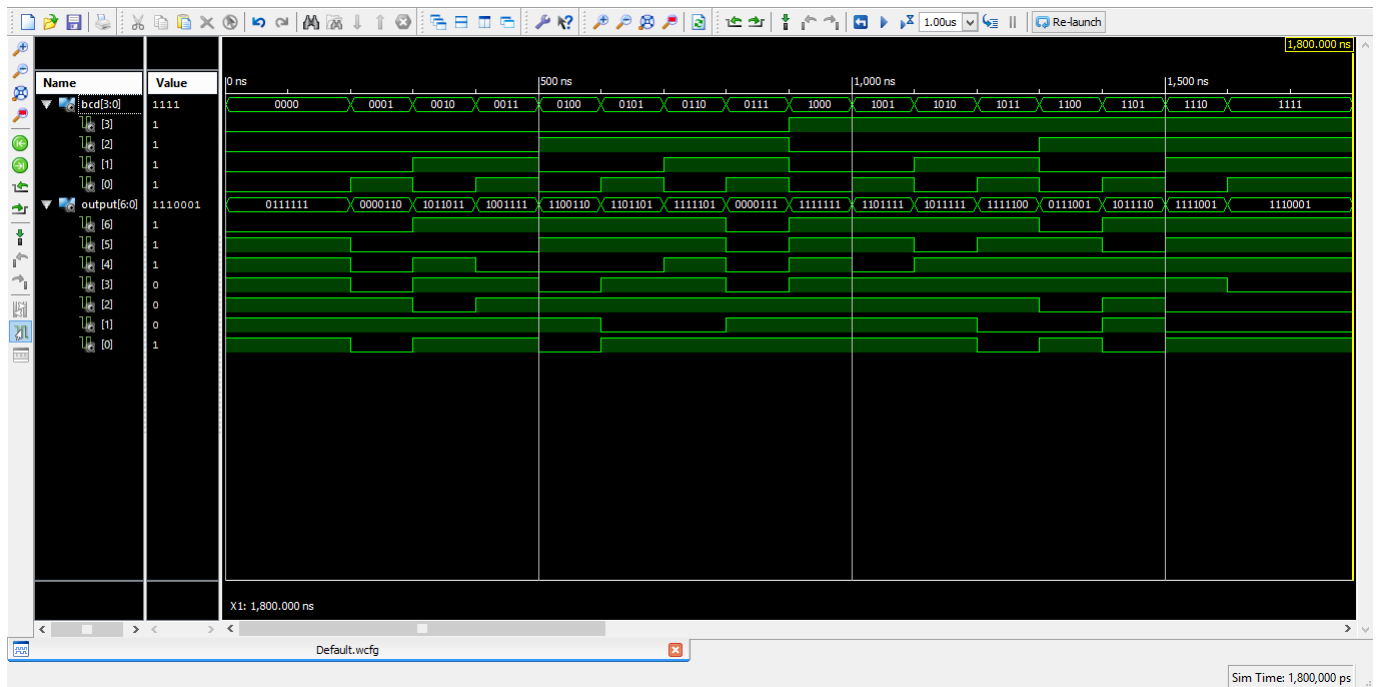
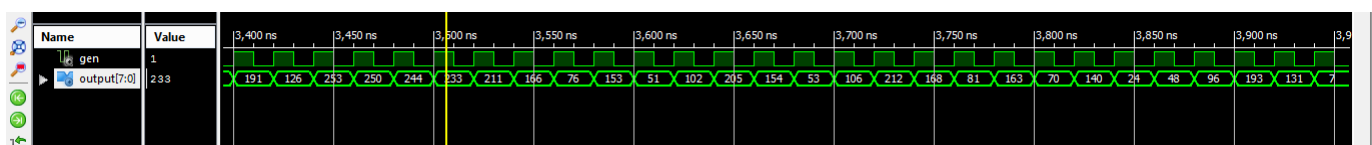
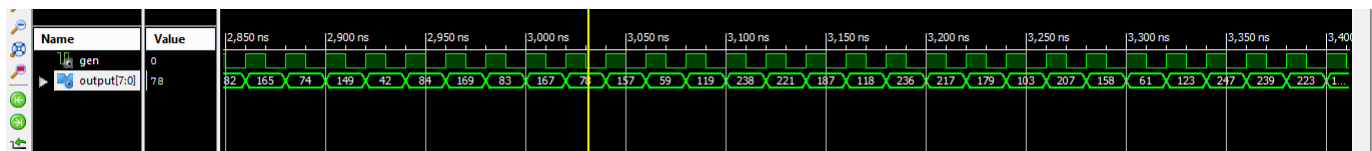
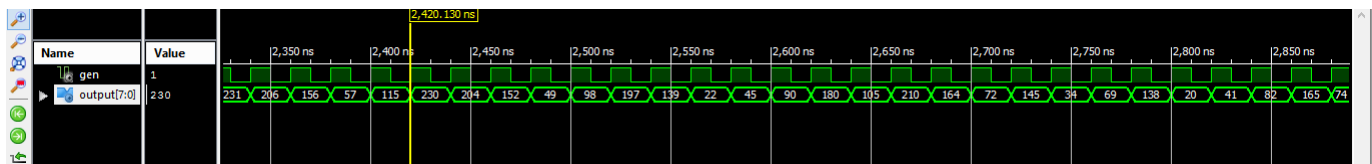
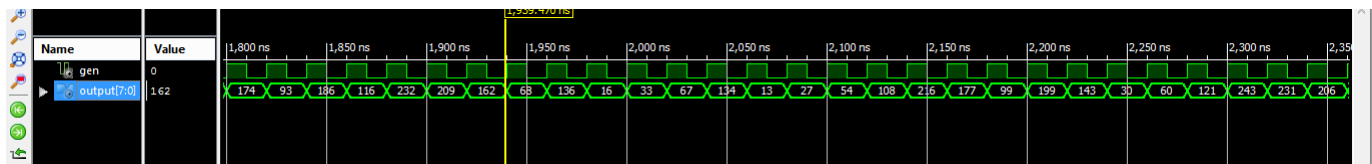
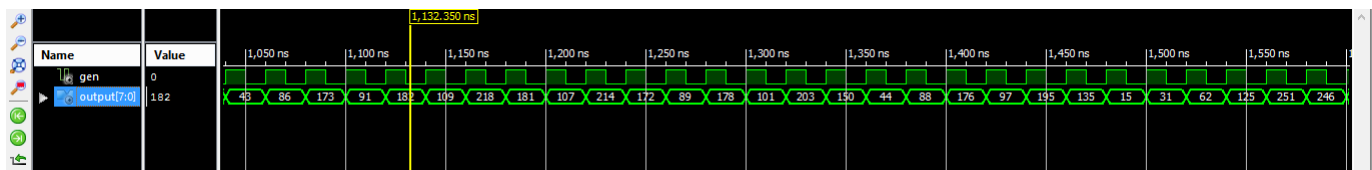
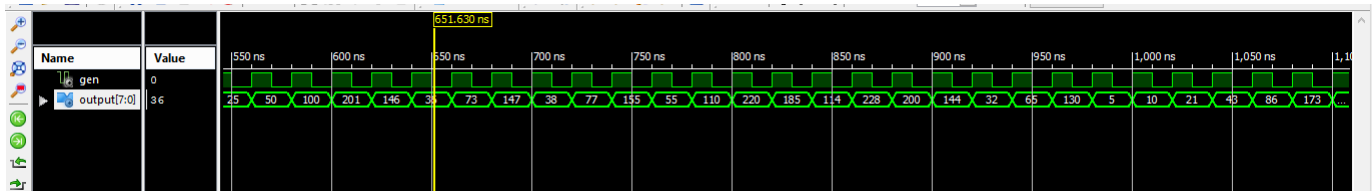
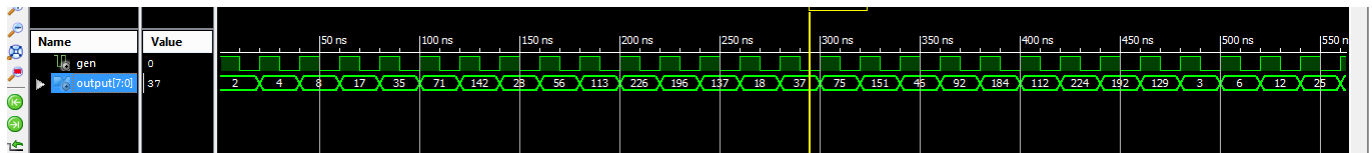


Figure 1: Timing Diagram for 7 segment decoder from 0ns to 1800 ns



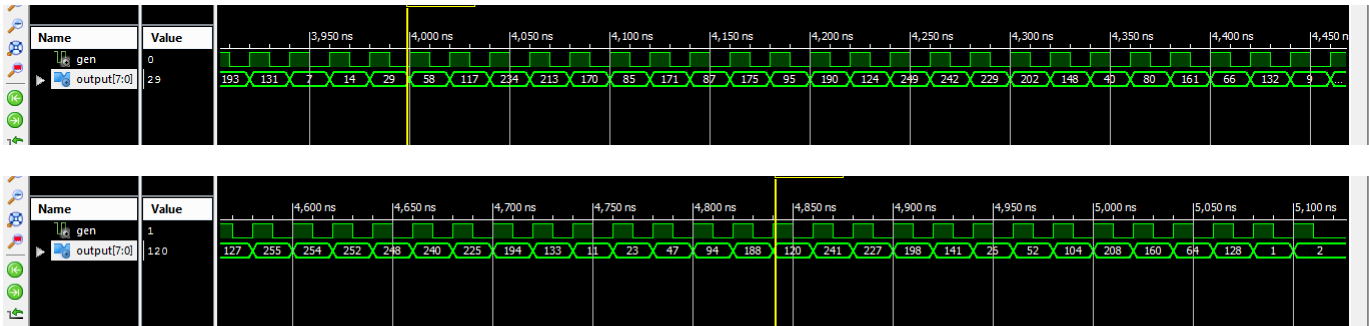


Figure 2: Timing Diagrams for pseudo random number generator

## Codes

Here I have include working code for both BCD to 7 segment Decoder and Pseudo Random Number Generator

### 1 code for BCD to 7 segment Decoder

```

1  -----
2  -- Company:
3  -- Engineer:
4  --
5  -- Create Date:      14:23:10 02/25/2014
6  -- Design Name:
7  -- Module Name:      bcd_7_segment - Behavioral
8  -- Project Name:
9  -- Target Devices:
10 -- Tool versions:
11 -- Description:
12 --
13 -- Dependencies:
14 --
15 -- Revision:
16 -- Revision 0.01 - File Created
17 -- Additional Comments:
18 --
19 -----
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22
23 -- Uncomment the following library declaration if using
24 -- arithmetic functions with Signed or Unsigned values
25 --use IEEE.NUMERIC_STD.ALL;
26

```

```

27 -- Uncomment the following library declaration if instantiating
28 -- any Xilinx primitives in this code.
29 --library UNISIM;
30 --use UNISIM.VComponents.all;
31
32 entity bcd_7_segment is
33     Port ( bcd : in  STD_LOGIC_VECTOR (3 downto 0);
34           output : out STD_LOGIC_VECTOR (6 downto 0));
35 end bcd_7_segment;
36
37 architecture Behavioral of bcd_7_segment is
38 begin process (bcd)
39 begin
40 output <= "0000000";
41 case bcd is
42     when "0000" =>
43         output <= "0111111";
44     when "0001" =>
45         output <= "0000110";
46     when "0010" =>
47         output <= "1011011";
48     when "0011" =>
49         output <= "1001111";
50     when "0100" =>
51         output <= "1100110";
52     when "0101" =>
53         output <= "1101101";
54     when "0110" =>
55         output <= "1111101";
56     when "0111" =>
57         output <= "0000111";
58     when "1000" =>
59         output <= "1111111";
60     when "1001" =>
61         output <= "1101111";
62     when "1010" =>
63         output <= "1011111";
64     when "1011" =>
65         output <= "1111100";
66     when "1100" =>
67         output <= "0111001";
68     when "1101" =>
69         output <= "1011110";
70     when "1110" =>
71         output <= "1111001";

```

```

72         when "1111" =>
73             output <= "1110001";
74         when others =>
75             null;
76
77 end case;
78 end process;
79
80 end Behavioral;

```

## 2 Code for Pseudo Random Number Generator

```

1  -----
2  -- Company:
3  -- Engineer:
4  --
5  -- Create Date:      15:15:18 02/25/2014
6  -- Design Name:
7  -- Module Name:      rand - Behavioral
8  -- Project Name:
9  -- Target Devices:
10 -- Tool versions:
11 -- Description:
12 --
13 -- Dependencies:
14 --
15 -- Revision:
16 -- Revision 0.01 - File Created
17 -- Additional Comments:
18 --
19 -----
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22
23 -- Uncomment the following library declaration if using
24 -- arithmetic functions with Signed or Unsigned values
25 --use IEEE.NUMERIC_STD.ALL;
26
27 -- Uncomment the following library declaration if instantiating
28 -- any Xilinx primitives in this code.
29 --library UNISIM;
30 --use UNISIM.VComponents.all;
31
32 entity rand is

```

```

33     Port ( gen : in  STD_LOGIC;
34           output : out STD_LOGIC_VECTOR (7 downto 0));
35 end rand;
36
37 architecture Behavioral of rand is
38 begin process(gen)
39 variable output_temp : std_logic_vector( 7 downto 0) := "00000001";
40 variable temp :std_logic := '0';
41 begin
42 case gen is
43     when '1' =>
44         temp := output_temp(7) xor output_temp(5)
45                xor output_temp(4) xor output_temp(3);
46         output_temp(7 downto 1) := output_temp(6 downto 0);
47         output_temp(0) := temp;
48         output <= output_temp;
49     when others =>
50         null;
51 end case;
52 end process;
53
54
55 end Behavioral;

```

## Inference

In this assignment I inferred the followings:

1. Working of BCD to 7 segment decoder
2. Truth table for bcd to 7 segment decoder
3. Using for loop in VHDL
4. Generating random numbers using hardware tools