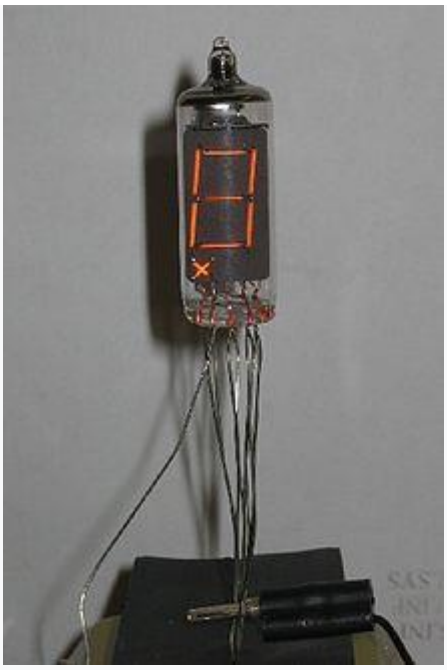
Verilog project for Seven-Segment Display

Created by : Rohit bagdi

M.tech IIT GUWAHATI POWER ENGINEERING (214102113)

Seven-segment representation of figures can be found in patents as early as 1903 (in U.S. Patent 1,126,641), when Carl Kinsley invented a method of telegraphically transmitting letters and numbers and having them printed on tape in a segmented format. In 1908, F. W. Wood invented an 8-segment display, which displayed the number 4 using a diagonal bar (U.S. Patent 974,943). In 1910, a seven-segment display illuminated by incandescent bulbs was used on a power-plant boiler room signal panel. They were also used to show the dialed telephone number to operators during the transition from manual to automatic telephone dialing. They did not achieve widespread use until the advent of LEDs in the 1970s.



Filament seven-segment display

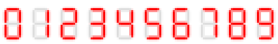
Some early seven-segment displays used incandescent filaments in an evacuated bulb; they are also known as numitrons. A variation (minitrons) made use of an evacuated potted box. Minitrons are filament segment displays that are housed in DIP packages like modern LED segment displays. They may have up to 16 segments There were also segment displays that used small incandescent light bulbs instead of LEDs or incandescent filaments. These worked similarly to modern LED segment displays.

Character:

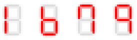
The seven segments are arranged as a rectangle of two vertical segments on each side with one horizontal segment on the top, middle, and bottom. Often the rectangle is *oblique* (slanted), which aids readability. In most applications, the segments are of nearly uniform shape and size (usually elongated hexagons, though trapezoids and rectangles can also be used), though in the case of adding machines, the vertical segments are longer and more oddly shaped at the ends in an effort to further enhance readability. The seven elements of the display can be lit in different combinations to represent the Arabic numerals.

Decimal:

The numerical digits 0 to 9 are the most common characters displayed on seven-segment displays. The most common patterns used for each of these is:



Alternate patterns: The numeral 1 may be represented with the left segments, the numerals 6 and 9 may be represented without a 'tail', and the numeral 7 represented *with* a 'tail':



In Unicode 13.0, 10 codepoints had been given for segmented digits 0–9 in the Symbols for Legacy Computing block, to replicate early computer fonts that included seven-segment versions of the digits. The official reference shows the less-common four-segment "7". The characters are simulated as:

	0	1	2	3	4	5	6	7	8	9
U+1FBFx	🯰	🯱	🯲	🯳	🯴	🯵	🯶	🯷	🯸	🯹

Hexadecimal

Four binary bits are needed to specify the numbers 0–9, but can also specify 10–15, so usually decoders with 4 bit inputs can also display Hexadecimal (Hex) digits. Today, a combination of uppercase and lowercase letters is commonly used for A–F; this is done to obtain a unique, unambiguous shape for each hexadecimal digit (otherwise, a capital 'D' would look identical to a '0' and a capital 'B' would look identical to an '8'). Also the digit '6' must be displayed with the top bar lit to avoid ambiguity with the letter 'b'.

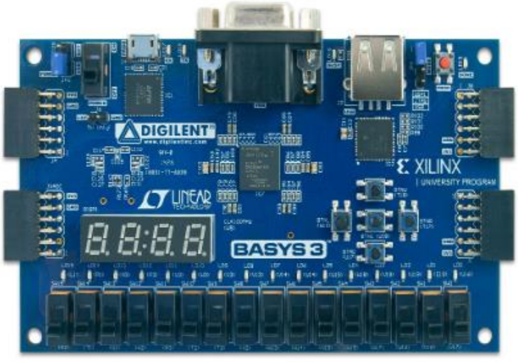
The following lookup table may be useful for writing code to drive a 7-segment display.

Hexadecimal encodings for displaying the digits 0 to F[22][23]

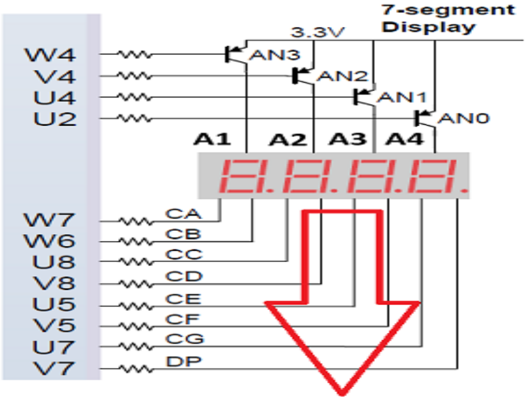
Digit	Display	p	a	b	c	d	e	f	g	pabcdefg	hex pabcdefg	pgfedcba	hex pgfedcba
0	🯰		on	on	on	on	on	on		01111110	0x7E	00111111	0x3F
1	🯱			on	on					00110000	0x30	00000110	0x06
2	🯲		on	on		on	on		on	01101101	0x6D	01011011	0x5B
3	🯳		on	on	on	on			on	01111001	0x79	01001111	0x4F
4	🯴			on	on			on	on	00110011	0x33	01100110	0x66
5	🯵		on		on	on		on	on	01011011	0x5B	01101101	0x6D
6	🯶		on		on	on	on	on	on	01011111	0x5F	01111101	0x7D

7	7		**on**	**on**	**on**					01110000	0x70	00000111	0x07
8	8		**on**	**on**	**on**	**on**	**on**	**on**	**on**	01111111	0x7F	01111111	0x7F
9	9		**on**	**on**	**on**	**on**		**on**	**on**	01111011	0x7B	01101111	0x6F
A	A		**on**	**on**	**on**		**on**	**on**	**on**	01110111	0x77	01110111	0x77
b	b				**on**	**on**	**on**	**on**	**on**	00011111	0x1F	01111100	0x7C
C	C		**on**			**on**	**on**	**on**		01001110	0x4E	00111001	0x39
d	d			**on**	**on**	**on**	**on**		**on**	00111101	0x3D	01011110	0x5E
E	E		**on**			**on**	**on**	**on**	**on**	01001111	0x4F	01111001	0x79
F	F		**on**				**on**	**on**	**on**	01000111	0x47	01110001	0x71

Using VHDL We are implementing Verilog code on FPGA Board



FPGA Board for seven segment display



Four 7-segment display will show the no.

Verilog code for 7-segment display :

Below is an example Verilog code for creating the refresh signal and LED-activating signals:

```
reg [19:0] refresh_counter;
// the first 18-bit for creating 2.6ms digit period
// the other 2-bit for creating 4 LED-activating signals
wire [1:0] LED_activating_counter;
// count       0    -> 1  -> 2  -> 3
// activates    LED1    LED2   LED3   LED4
// and repeat
always @(posedge clock_100Mhz or posedge reset)
begin
 if(reset==1)
  refresh_counter <= 0;
 else
  refresh_counter <= refresh_counter + 1;
end
assign LED_activating_counter = refresh_counter[19:18];
```

After that, we need to generate the anode signals (W4, V4, U4, U2) for the four-digit 7-segment LED display based on the LED-activating counter. The LED-activating counter will repeatedly count from zero to three for continuously activating and updating the four seven-segment LEDs.

Below is an example Verilog code for creating the anode signals and updating values of the four 7-segment LEDs on Basys 3 FPGA:

```
// anode activating signals for 4 LEDs
    // decoder to generate anode signals
    always @(*)
    begin
        case(LED_activating_counter)
        2'b00: begin
            Anode_Activate = 4'b0111;
            // activate LED1 and Deactivate LED2, LED3, LED4
            LED_BCD = displayed_number[15:11];
            // the first hex-digit of the 16-bit number
             end
        2'b01: begin
            Anode_Activate = 4'b1011;
            // activate LED2 and Deactivate LED1, LED3, LED4
            LED_BCD = displayed_number[10:8];
            // the second hex-digit of the 16-bit number
```

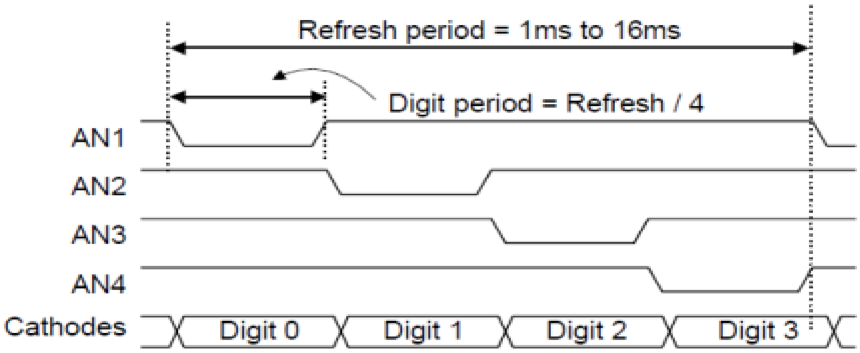
```
                end
        2'b10: begin
            Anode_Activate = 4'b1101;
            // activate LED3 and Deactivate LED2, LED1, LED4
            LED_BCD = displayed_number[7:4];
             // the third hex-digit of the 16-bit number
              end
        2'b11: begin
            Anode_Activate = 4'b1110;
            // activate LED4 and Deactivate LED2, LED3, LED1
             LED_BCD = displayed_number[3:0];
             // the fourth hex-digit of the 16-bit number
               end
        default:begin
             Anode_Activate = 4'b0111;
            // activate LED1 and Deactivate LED2, LED3, LED4
            LED_BCD = displayed_number[15:11];
            // the first hex-digit of the 16-bit number
            end
        endcase
    end
```

Last but not least, the example Verilog code for BCD to 7-segment decoder based on the decoder table above:

```
reg[6:0] LED_out;
// Cathode patterns of the 7-segment LED display
always @(*)
begin
 case(LED_BCD)
 4'b0000: LED_out = 7'b0000001; // "0"
 4'b0001: LED_out = 7'b1001111; // "1"
 4'b0010: LED_out = 7'b0010010; // "2"
 4'b0011: LED_out = 7'b0000110; // "3"
 4'b0100: LED_out = 7'b1001100; // "4"
 4'b0101: LED_out = 7'b0100100; // "5"
 4'b0110: LED_out = 7'b0100000; // "6"
 4'b0111: LED_out = 7'b0001111; // "7"
 4'b1000: LED_out = 7'b0000000; // "8"
 4'b1001: LED_out = 7'b0000100; // "9"
 default: LED_out = 7'b0000001; // "0"
 endcase
end
```

Pin constraint file for the four-digit seven-segment LED display on Basys 3 FPGA:

```
# Clock signal
set_property PACKAGE_PIN W5 [get_ports clock_100Mhz]
 set_property IOSTANDARD LVCMOS33 [get_ports clock_100Mhz]
set_property PACKAGE_PIN R2 [get_ports reset]
 set_property IOSTANDARD LVCMOS33 [get_ports reset]
#seven-segment LED display
set_property PACKAGE_PIN W7 [get_ports {LED_out[6]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[6]}]
set_property PACKAGE_PIN W6 [get_ports {LED_out[5]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[5]}]
set_property PACKAGE_PIN U8 [get_ports {LED_out[4]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[4]}]
set_property PACKAGE_PIN V8 [get_ports {LED_out[3]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[3]}]
set_property PACKAGE_PIN U5 [get_ports {LED_out[2]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[2]}]
set_property PACKAGE_PIN V5 [get_ports {LED_out[1]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[1]}]
set_property PACKAGE_PIN U7 [get_ports {LED_out[0]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[0]}]
set_property PACKAGE_PIN U2 [get_ports {Anode_Activate[0]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Activate[0]}]
set_property PACKAGE_PIN U4 [get_ports {Anode_Activate[1]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Activate[1]}]
set_property PACKAGE_PIN V4 [get_ports {Anode_Activate[2]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Activate[2]}]
set_property PACKAGE_PIN W4 [get_ports {Anode_Activate[3]}]
   set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Activate[3]}]
```



Seven-segment display timing diagram