Lab 8
Combinational Circuit

Digital System Fundamentals

Sorayut Glomglome

Objectives

- Contruct combinational circuit using VHDL.
- Use 7 segment display on Basys 3.
- See example of behavioral coding style

Leaning Outcome

- Construct priority encoder using VHDL.
- Use segment display on Basys 3.

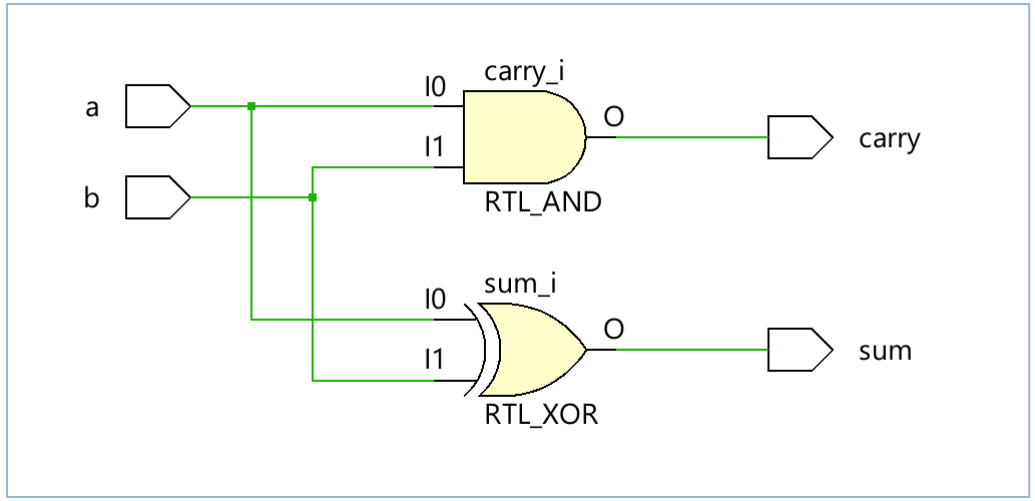
1. VHDL Coding styles

VHDL Coding styles

1) Dataflow
2) Behavioral
3) Structural

Dataflow Coding Style of Half Adder

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  entity half_adder is
26      Port ( a : in STD_LOGIC;
27             b : in STD_LOGIC;
28             sum : out STD_LOGIC;
29             carry : out STD_LOGIC);
30  end half_adder;
31
32  architecture dataflow of half_adder is
33
34  begin
35
36      sum <= a xor b;
37      carry <= a and b;
38
39  end dataflow;
40
```

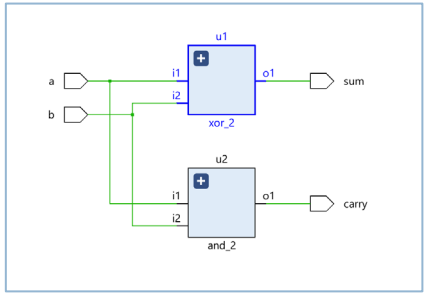


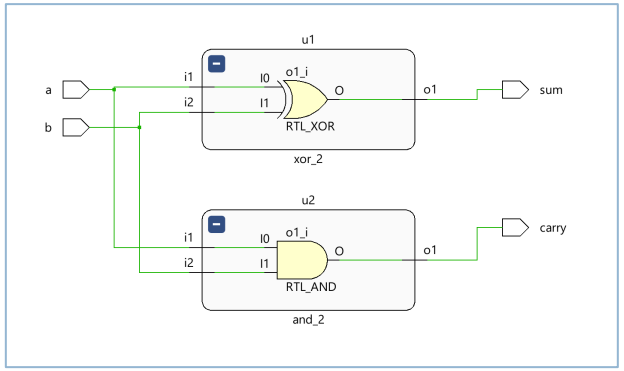
Structural Coding Style of Half Adder

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


entity xor_2 is
    Port ( i1 : in STD_LOGIC;
           i2 : in STD_LOGIC;
           o1 : out STD_LOGIC);
end xor_2;

architecture dataflow of xor_2 is

begin

    o1 <= i1 xor i2;

end dataflow;
```

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


entity and_2 is
    Port ( i1 : in STD_LOGIC;
           i2 : in STD_LOGIC;
           o1 : out STD_LOGIC);
end and_2;

architecture dataflow of and_2 is

begin

    o1 <= i1 and i2;

end dataflow;
```

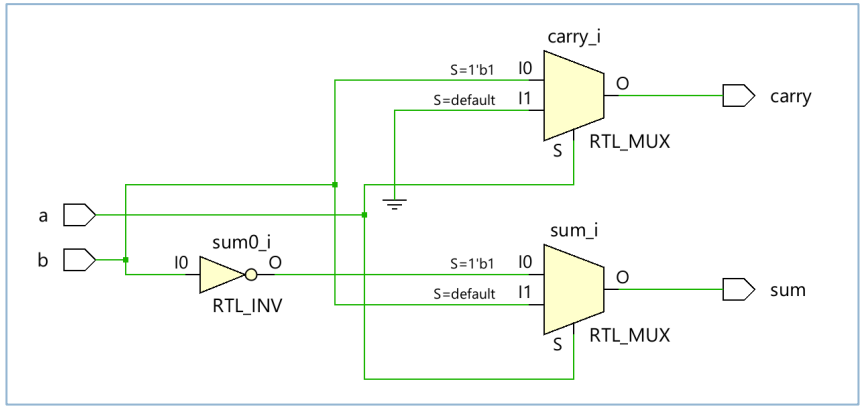
```vhdl

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity half_adder is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           sum : out STD_LOGIC;
           carry : out STD_LOGIC);
end half_adder;

architecture structural of half_adder is

    component xor_2 is
        port (  i1 : in STD_LOGIC;
                i2 : in STD_LOGIC;
                o1 : out STD_LOGIC);
    end component;

    component and_2 is
        port (  i1 : in STD_LOGIC;
                i2 : in STD_LOGIC;
                o1 : out STD_LOGIC);
    end component;

begin

    u1 : xor_2 port map (i1 => a, i2 => b, o1 => sum);
    u2 : and_2 port map (i1 => a, i2 => b, o1 => carry);


end structural;

```





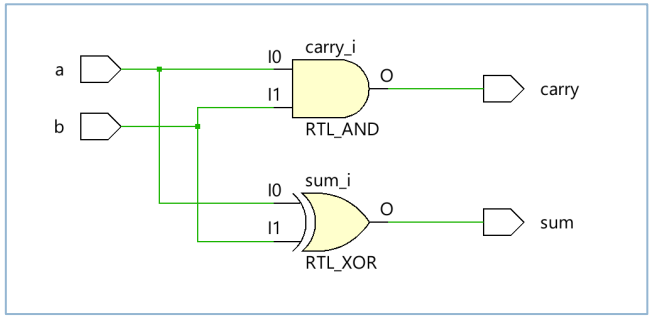
Behavioral Coding Style of Half Adder

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  entity half_adder is
26      Port ( a : in STD_LOGIC;
27             b : in STD_LOGIC;
28             sum : out STD_LOGIC;
29             carry : out STD_LOGIC);
30  end half_adder;
31
32  architecture behavioral of half_adder is
33
34  begin
35
36      ha : process (a, b)
37      begin
38          if a = '1' then
39              sum <= not b;
40              carry <= b;
41          else
42              sum <= b;
43              carry <= '0';
44          end if;
45      end process ha;
46
47  end behavioral;
```

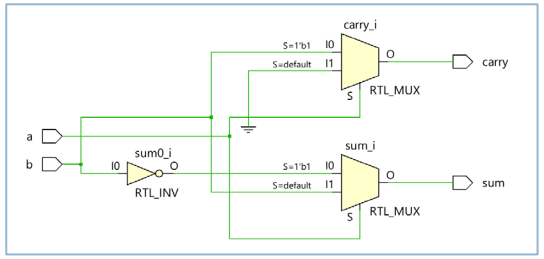


A	B	C_{out}	Σ
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

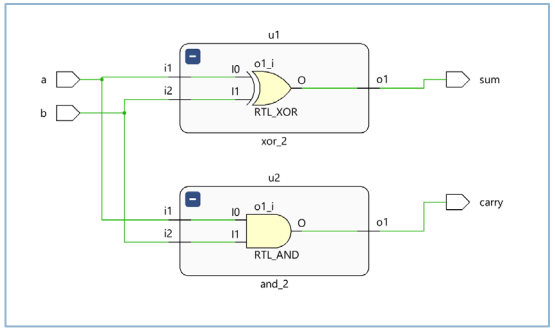
Three Coding Styles of Half Adder




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

entity half_adder is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           sum : out STD_LOGIC;
           carry : out STD_LOGIC);
end half_adder;

architecture dataflow of half_adder is

begin

    sum <= a xor b;
    carry <= a and b;

end dataflow;
```



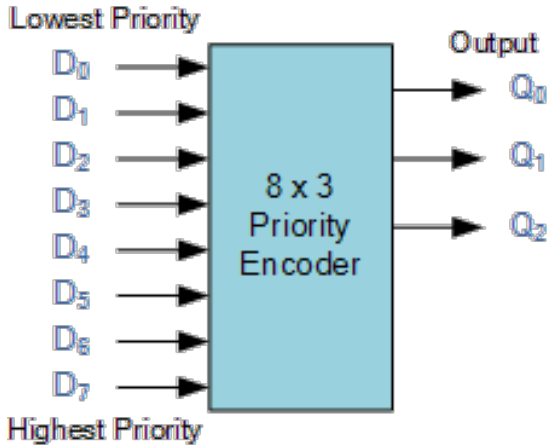

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity half_adder is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           sum : out STD_LOGIC;
           carry : out STD_LOGIC);
end half_adder;

architecture behavioral of half_adder is

begin

    ha : process (a, b)
    begin
        if a = '1' then
            sum <= not b;
            carry <= b;
        else
            sum <= b;
            carry <= '0';
        end if;
    end process ha;

end behavioral;
```




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


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity half_adder is
    Port ( a : in STD_LOGIC;
           b : in STD_LOGIC;
           sum : out STD_LOGIC;
           carry : out STD_LOGIC);
end half_adder;

architecture structural of half_adder is

    component xor_2 is
        port (  i1 : in STD_LOGIC;
                i2 : in STD_LOGIC;
                o1 : out STD_LOGIC);
    end component;

    component and_2 is
        port (  i1 : in STD_LOGIC;
                i2 : in STD_LOGIC;
                o1 : out STD_LOGIC);
    end component;

begin

    u1 : xor_2 port map (i1 => a, i2 => b, o1 => sum);
    u2 : and_2 port map (i1 => a, i2 => b, o1 => carry);


end structural;
```

2. Priority Encoder

8-to-3 Priority Encoder | Boolean Expression



Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Q_2	Q_1	Q_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	x	0	0	1
0	0	0	0	0	1	x	x	0	1	0
0	0	0	0	1	x	x	x	0	1	1
0	0	0	1	x	x	x	x	1	0	0
0	0	1	x	x	x	x	x	1	0	1
0	1	x	x	x	x	x	x	1	1	0
1	x	x	x	x	x	x	x	1	1	1

X = dont care

Priority Encoder Output Expression

$$Q_0 = \sum\left(\bar{D}_6\left(\bar{D}_4\bar{D}_2D_1 + \bar{D}_4D_3 + D_5\right) + D_7\right)$$

$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4(D_2 + D_3) + D_6 + D_7\right)$$

$$Q_2 = \sum(D_4 + D_5 + D_6 + D_7)$$

Output Q_0

$$Q_0 = \sum(1, 3, 5, 7)$$

$$Q_0 = \sum\left(\bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4\bar{D}_3\bar{D}_2D_1 + \bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4D_3 + \bar{D}_7\bar{D}_6D_5 + D_7\right)$$

$$Q_0 = \sum\left(\bar{D}_6\bar{D}_4\bar{D}_2D_1 + \bar{D}_6\bar{D}_4D_3 + \bar{D}_6D_5 + D_7\right)$$

$$Q_0 = \sum\left(\bar{D}_6\left(\bar{D}_4\bar{D}_2D_1 + \bar{D}_4D_3 + D_5\right) + D_7\right)$$

Output Q_1

$$Q_1 = \sum(2, 3, 6, 7)$$

$$Q_1 = \sum\left(\bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4\bar{D}_3D_2 + \bar{D}_7\bar{D}_6\bar{D}_5\bar{D}_4D_3 + \bar{D}_7D_6 + D_7\right)$$

$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4D_2 + \bar{D}_5\bar{D}_4D_3 + D_6 + D_7\right)$$

$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4(D_2 + D_3) + D_6 + D_7\right)$$

Output Q_2

$$Q_2 = \sum(4, 5, 6, 7)$$

$$Q_2 = \sum\left(\bar{D}_7\bar{D}_6\bar{D}_5D_4 + \bar{D}_7\bar{D}_6D_5 + \bar{D}_7D_6 + D_7\right)$$

$$Q_2 = \sum(D_4 + D_5 + D_6 + D_7)$$

Priority Encoder and Digital Encoder Tutorial (electronics-tutorials.ws)

Use signal as Temp

Priority Encoder Output Expression

$$Q_0 = \sum\left(\bar{D}_6\left(\bar{D}_4\bar{D}_2D_1 + \bar{D}_4D_3 + D_5\right) + D_7\right)$$

$$Q_1 = \sum\left(\bar{D}_5\bar{D}_4(D_2 + D_3) + D_6 + D_7\right)$$

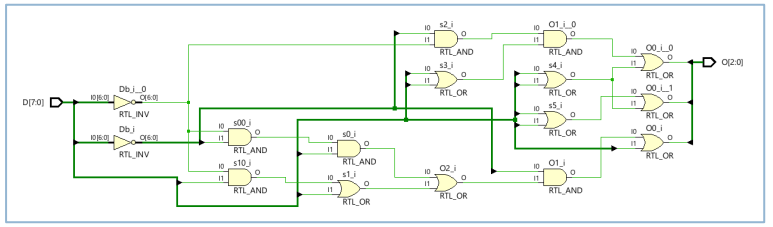
$$Q_2 = \sum(D_4 + D_5 + D_6 + D_7)$$

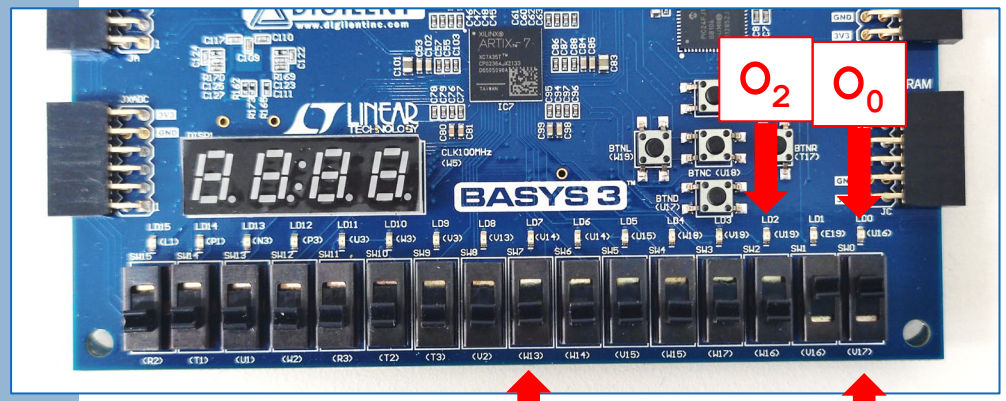
```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
33
34  entity encoder8to3 is
35      Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
36             O : out STD_LOGIC_VECTOR (2 downto 0));
37  end encoder8to3;
38
39  architecture dataflow of encoder8to3 is
40
41  signal Db, min : STD_LOGIC_VECTOR (7 downto 0);
42  signal s0, s1, s2, s3, s4, s5, s6 : std_logic;
43
44  begin
45
46      Db   <= not D;
47
48      s0   <= (Db(4) and Db(2)) and D(1);
49      s1   <= (Db(4) and D(3)) or D(5);
50      O(0) <= ((Db(6) and (s0 or s1))) or D(7);
51
52      s2   <= Db(5) and Db(4);
53      s3   <= D(2) or D(3);
54      s4   <= D(6) or D(7);
55      O(1) <= (s2 and s3) or s4;
56
57      s5   <= D(4) or D(5);
58      s6   <= D(6) or D(7);
59      O(2) <= s5 or s6;
60
61  end dataflow;
62
```

Checkpoint 1

Checkpoint 1

- Construct 8-to-3 priority encoder according to previous slides.
- Do the testbench with at least 11 input cases.
  - 8 input cases for each input switch.
  - 3 input cases with at least 2 activated input switches.
  - increase the duration of wait for statement to 50 ns.
- Test on Basys 3 using the following constraints.

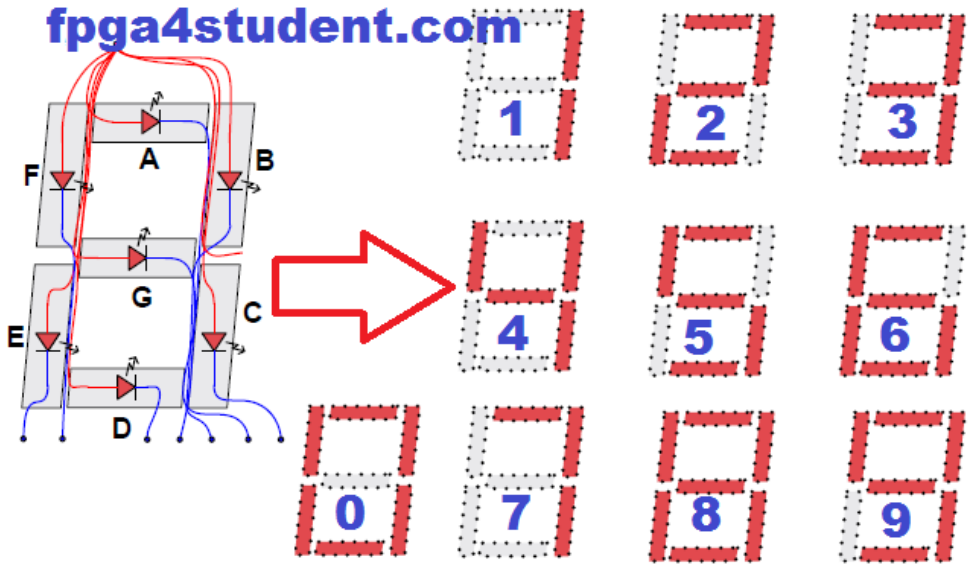


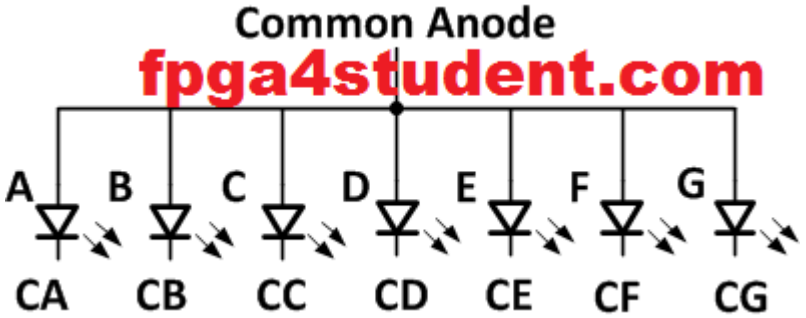


Name	Direction	Board Part Pin	Board Part Interface	Interface	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std
D[7]	IN					W13	✓	14	LVCMOS33*
D[6]	IN					W14	✓	14	LVCMOS33*
D[5]	IN					V15	✓	14	LVCMOS33*
D[4]	IN					W15	✓	14	LVCMOS33*
D[3]	IN					W17	✓	14	LVCMOS33*
D[2]	IN					W16	✓	14	LVCMOS33*
D[1]	IN					V16	✓	14	LVCMOS33*
D[0]	IN					V17	✓	14	LVCMOS33*
O[2]	OUT					U19	✓	14	LVCMOS33*
O[1]	OUT					E19	✓	14	LVCMOS33*
O[0]	OUT					U16	✓	14	LVCMOS33*

3. Seven Segment Display on Basys 3

Common Anode 7Seg Used on Basys 3

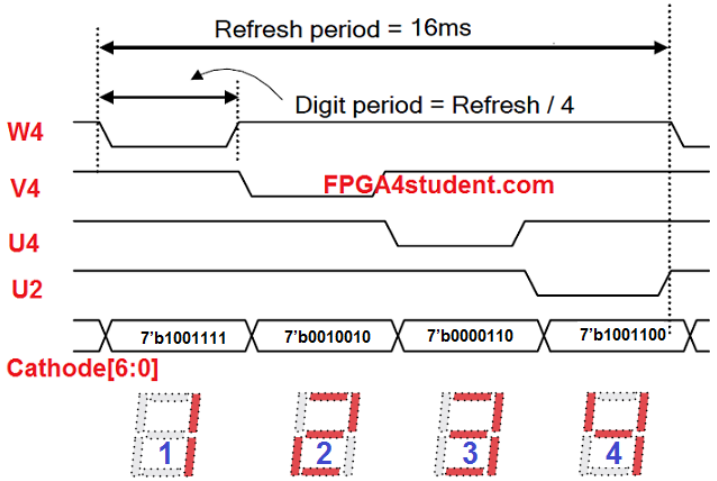


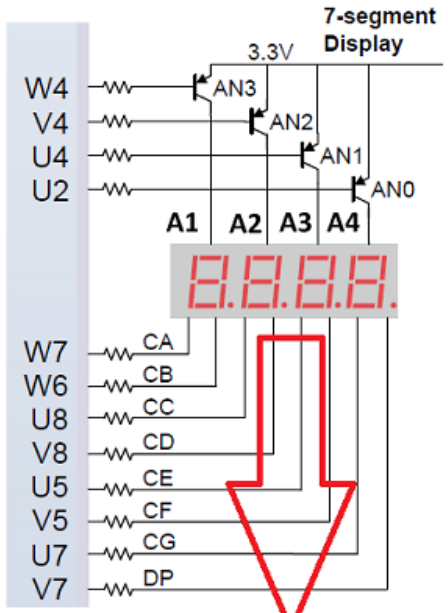


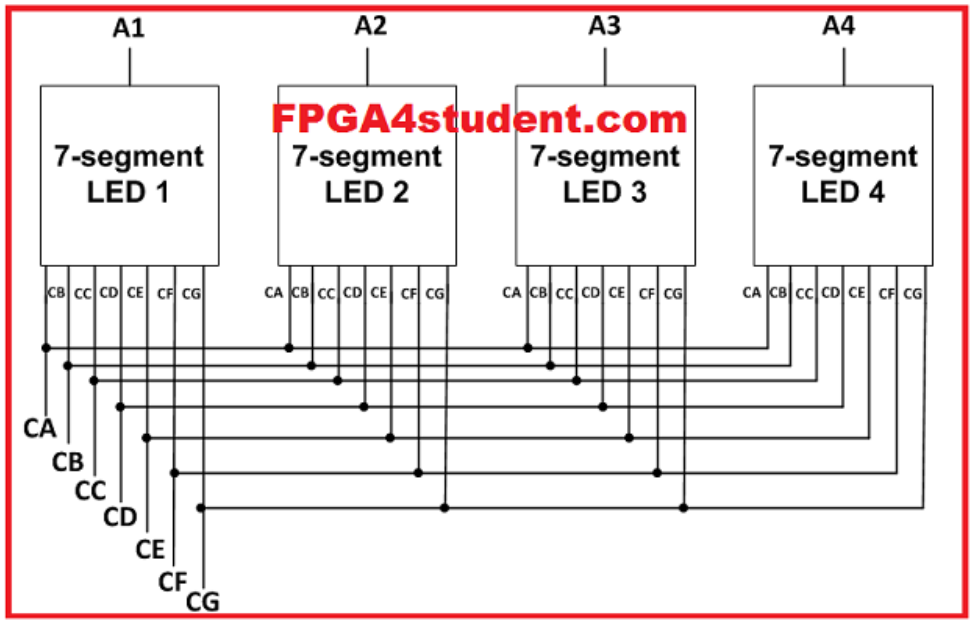
[FPGA Tutorial] Seven-Segment LED Display on Basys 3 FPGA - FPGA4student.com

VHDL code for Seven-Segment Display on Basys 3 FPGA - FPGA4student.com

Constraints





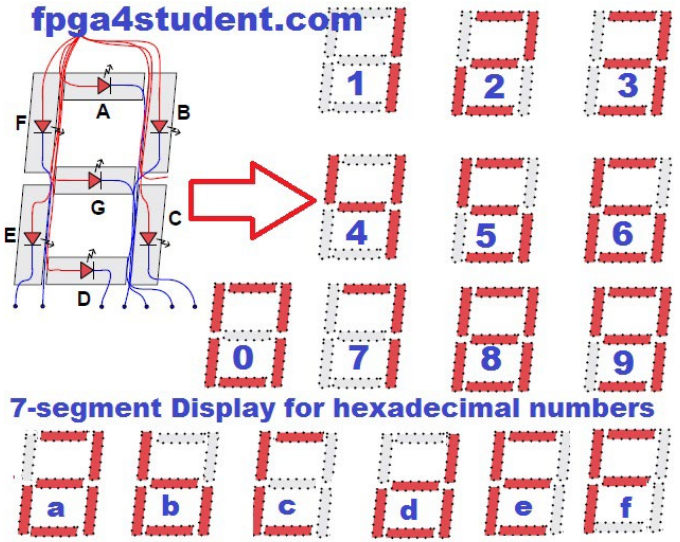


To avoid the displaying discontinuity perceived by the human eye, the four seven-segment LEDs should be continuously refreshed at about 1KHz to 60Hz or it should be refreshed at every 1ms to 16ms.

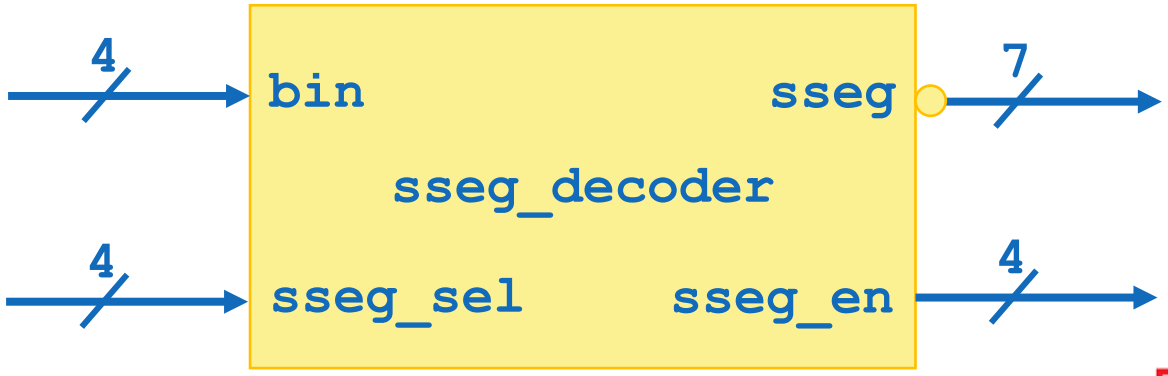
Persistence of vision

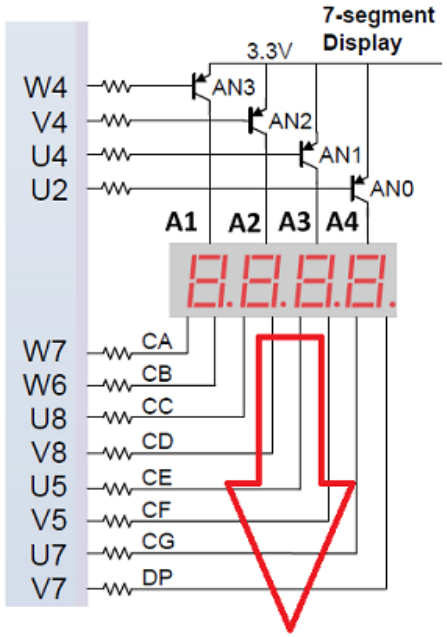
Binary to 7 Segment | Truth Table

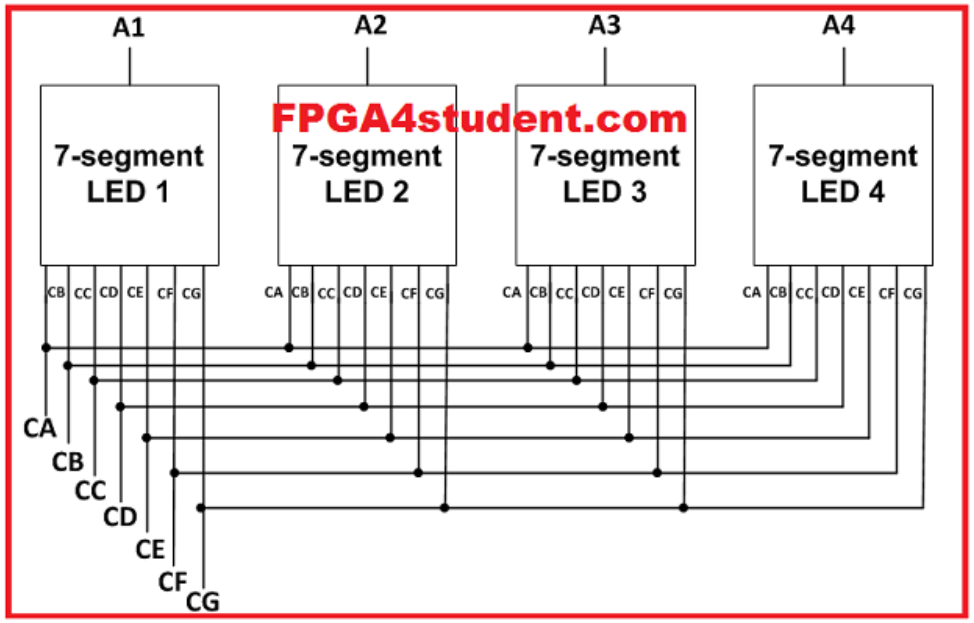
Hex Number	Common-Anode	CA	CB	CC	CD	CE	CF	CG	Cathode[6:0]
0	high	low	low	low	low	low	low	high	0000001
1	high	high	low	low	high	high	high	high	1001111
2	high	low	low	high	low	low	high	low	0010010
3	high	low	low	low	low	high	high	low	0000110
4	high	high	low	low	high	high	low	low	1001100
5	high	low	high	low	low	high	low	low	0100100
6	high	low	high	low	low	low	low	low	0100000
7	high	low	low	low	high	high	high	high	0001111
8	high	low	low	low	low	low	low	low	0000000
9	high	low	low	low	low	high	low	low	0000100
a	high	low	low	low	low	low	high	low	0000010
b	high	high	high	low	low	low	low	low	1100000
c	high	low	high	high	low	low	low	high	0110001
d	high	high	low	low	low	low	high	low	1000010
e	high	low	high	high	low	low	low	low	0110000
f	high	low	high	high	high	low	low	low	0111000



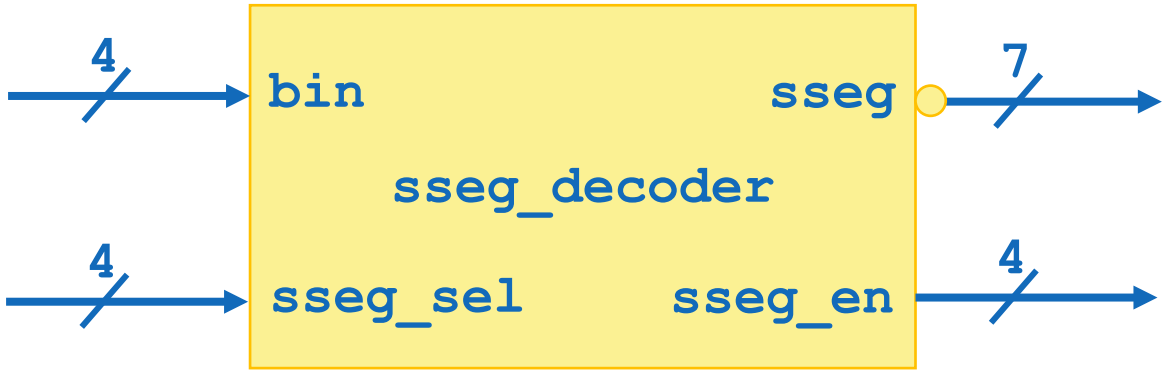
sseg_decoder | Entity







Behavioral Coding Style

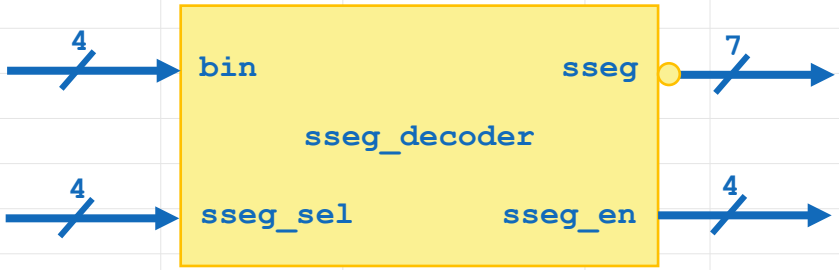


Hex Number	Common-Anode	CA	CB	CC	CD	CE	CF	CG	Cathode[6:0]
			fpga4student.com						
0	high	low	low	low	low	low	low	high	0000001
1	high	high	low	low	high	high	high	high	1001111
2	high	low	low	high	low	low	high	low	0010010
3	high	low	low	low	low	high	high	low	0000110
4	high	high	low	low	high	high	low	low	1001100
5	high	low	high	low	low	high	low	low	0100100
6	high	low	high	low	low	low	low	low	0100000
7	high	low	low	low	high	high	high	high	0001111
8	high	low	low	low	low	low	low	low	0000000
9	high	low	low	low	low	high	low	low	0000100
a	high	low	low	low	low	low	high	low	0000010
b	high	high	high	low	low	low	low	low	1100000
c	high	low	high	high	low	low	low	high	0110001
d	high	high	low	low	low	low	high	low	1000010
e	high	low	high	high	low	low	low	low	0110000
f	high	low	high	high	high	low	low	low	0111000

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using...
33
34  entity sseg_decoder is
35      Port ( bin : in STD_LOGIC_VECTOR (3 downto 0);
36             sseg_sel : in STD_LOGIC_VECTOR (3 downto 0);
37             sseg : out STD_LOGIC_VECTOR (6 downto 0);
38             sseg_en : out STD_LOGIC_VECTOR (3 downto 0));
39  end sseg_decoder;
40
41  architecture Behavioral of sseg_decoder is
42
43  begin
44
45      sseg_en <= sseg_sel;
46
47      sseg_display : process (bin)
48
49      begin
50          case bin is
52              when "0000" => sseg <= "0000001"; -- "0"
53              when "0001" => sseg <= "1001111"; -- "1"
54              when "0010" => sseg <= "0010010"; -- "2"
55              when "0011" => sseg <= "0000110"; -- "3"
56              when "0100" => sseg <= "1001100"; -- "4"
57              when "0101" => sseg <= "0100100"; -- "5"
58              when "0110" => sseg <= "0100000"; -- "6"
59              when "0111" => sseg <= "0001111"; -- "7"
60              when "1000" => sseg <= "0000000"; -- "8"
61              when "1001" => sseg <= "0000100"; -- "9"
62              when "1010" => sseg <= "0000010"; -- a
63              when "1011" => sseg <= "1100000"; -- b
64              when "1100" => sseg <= "0110001"; -- C
65              when "1101" => sseg <= "1000010"; -- d
66              when "1110" => sseg <= "0110000"; -- E
67              when "1111" => sseg <= "0111000"; -- F
68          end case;
69      end process sseg_display;
70  end Behavioral;
71
```

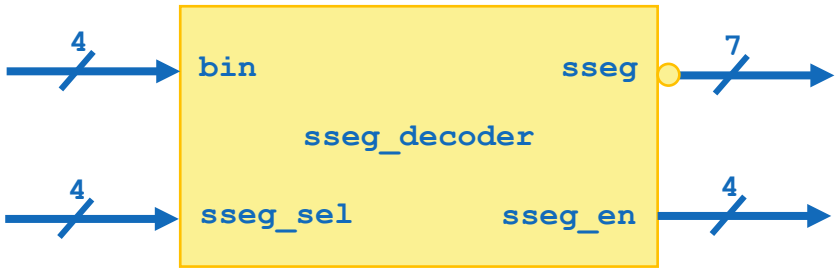
Constraints

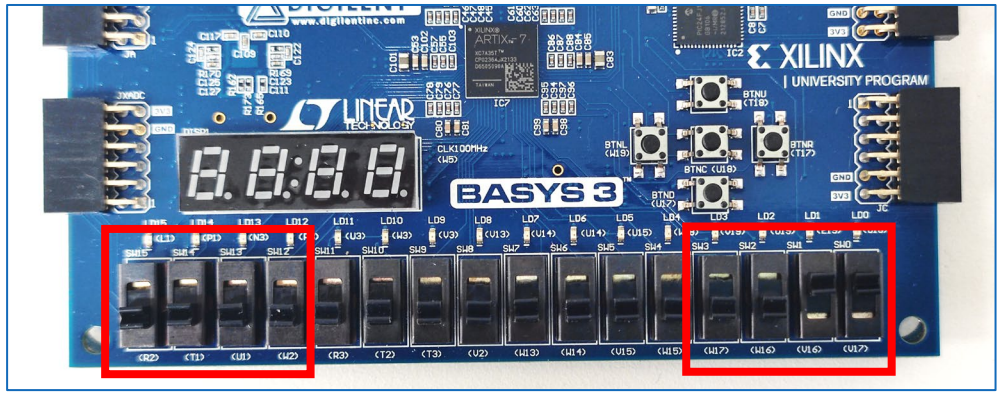
Name	Direction	Board Part Pin	Board Part Interface	Interface	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco
bin[3]	IN					W17	✓	14	LVCMOS33*	3.300
bin[2]	IN					W16	✓	14	LVCMOS33*	3.300
bin[1]	IN					V16	✓	14	LVCMOS33*	3.300
bin[0]	IN					V17	✓	14	LVCMOS33*	3.300
sseg[6]	OUT					W7	✓	34	LVCMOS33*	3.300
sseg[5]	OUT					W6	✓	34	LVCMOS33*	3.300
sseg[4]	OUT					U8	✓	34	LVCMOS33*	3.300
sseg[3]	OUT					V8	✓	34	LVCMOS33*	3.300
sseg[2]	OUT					U5	✓	34	LVCMOS33*	3.300
sseg[1]	OUT					V5	✓	34	LVCMOS33*	3.300
sseg[0]	OUT					U7	✓	34	LVCMOS33*	3.300
sseg_en[3]	OUT					W4	✓	34	LVCMOS33*	3.300
sseg_en[2]	OUT					V4	✓	34	LVCMOS33*	3.300
sseg_en[1]	OUT					U4	✓	34	LVCMOS33*	3.300
sseg_en[0]	OUT					U2	✓	34	LVCMOS33*	3.300
sseg_sel[3]	IN					R2	✓	34	LVCMOS33*	3.300
sseg_sel[2]	IN					T1	✓	34	LVCMOS33*	3.300
sseg_sel[1]	IN					U1	✓	34	LVCMOS33*	3.300
sseg_sel[0]	IN					W2	✓	34	LVCMOS33*	3.300



Testing

- Create bin-to-seg project using code from previous slides.
- Config constraints.
- Generate .bit file and Program Device.
- Test input signal : bin and sseg_sel.



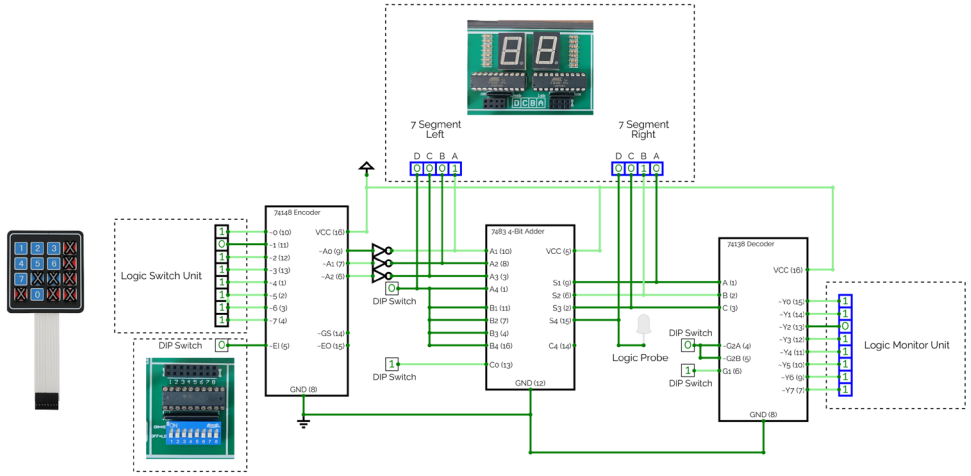


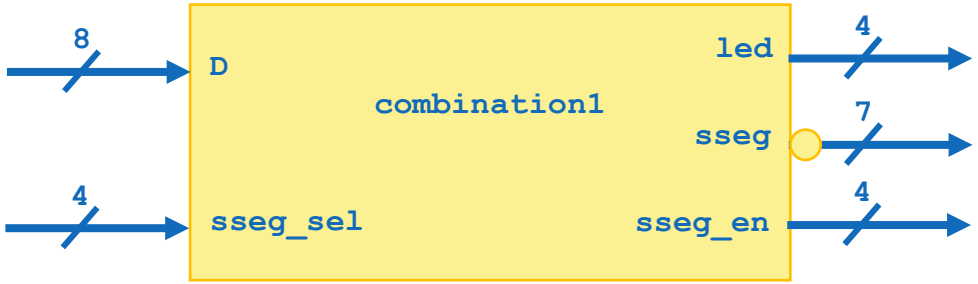
sseg_sel bin

Checkpoint 2

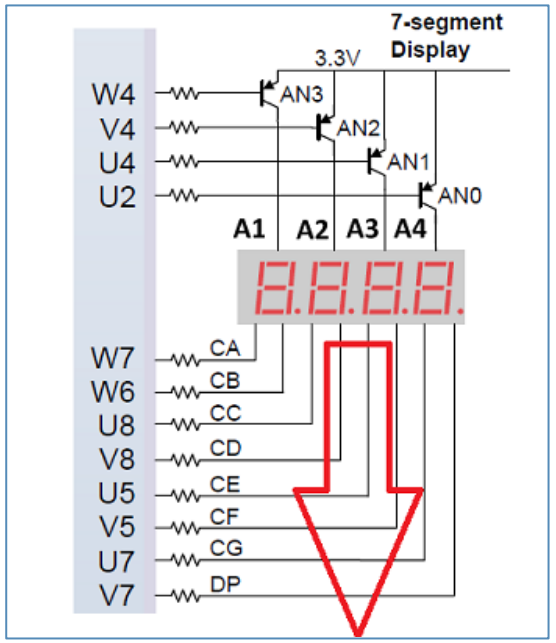
Combination1 | Entity

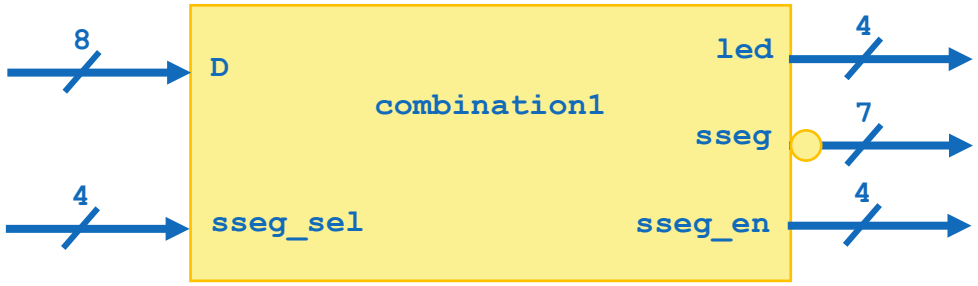
- Combination1 circuit combines 8-to-3 priority encoder and seven segment decoder into one circuit.
- Simplifiled version from Lab 5 with active high inputs.
- 7 Segment and 4 LEDs will show the result according to active input switches (D & sseg_sel).

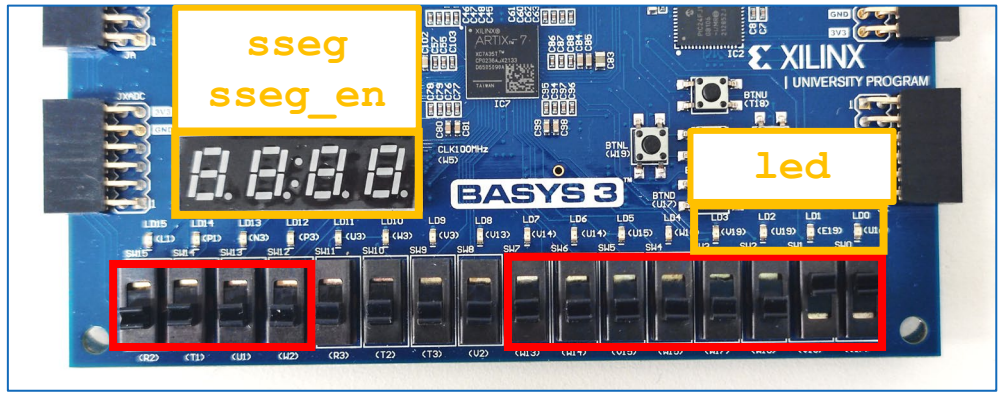




Constraints







Checkpoint 2

- Construct combination1 circuit.
- Config constraints.
- Generate .bit file and Program Device.
- Test on Basys 3.

THANK YOU