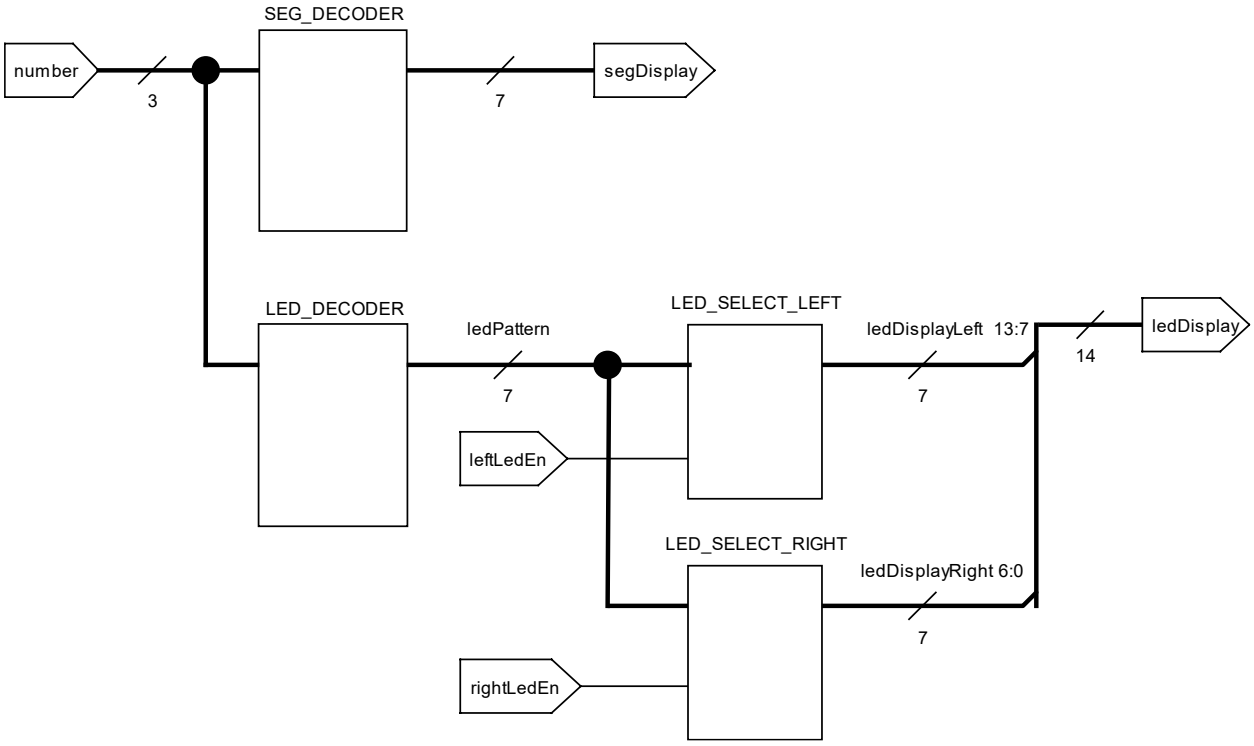
Humza Rana and Ayman Saad

Lab 02

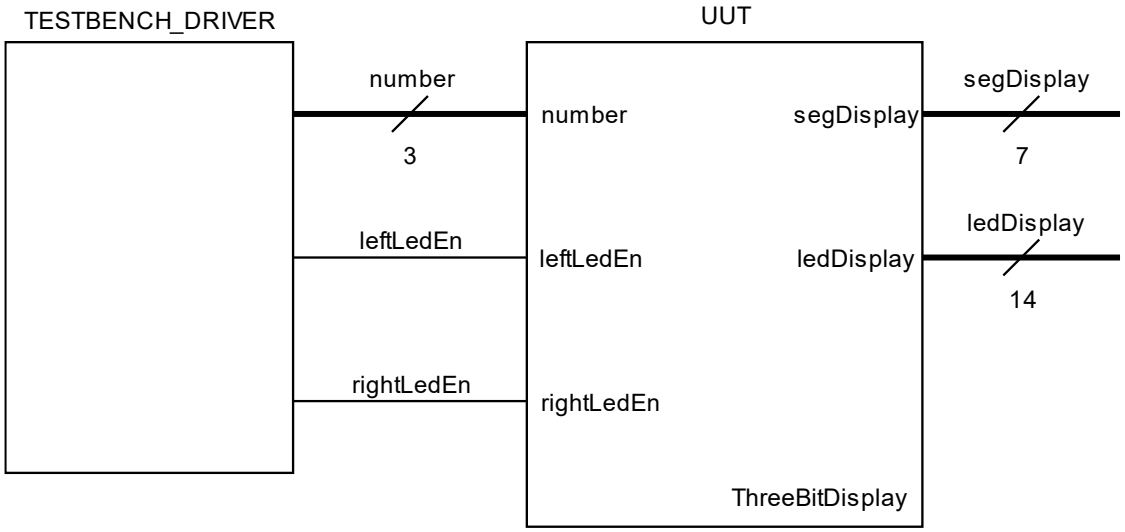
CPE 3020

Spring 2025

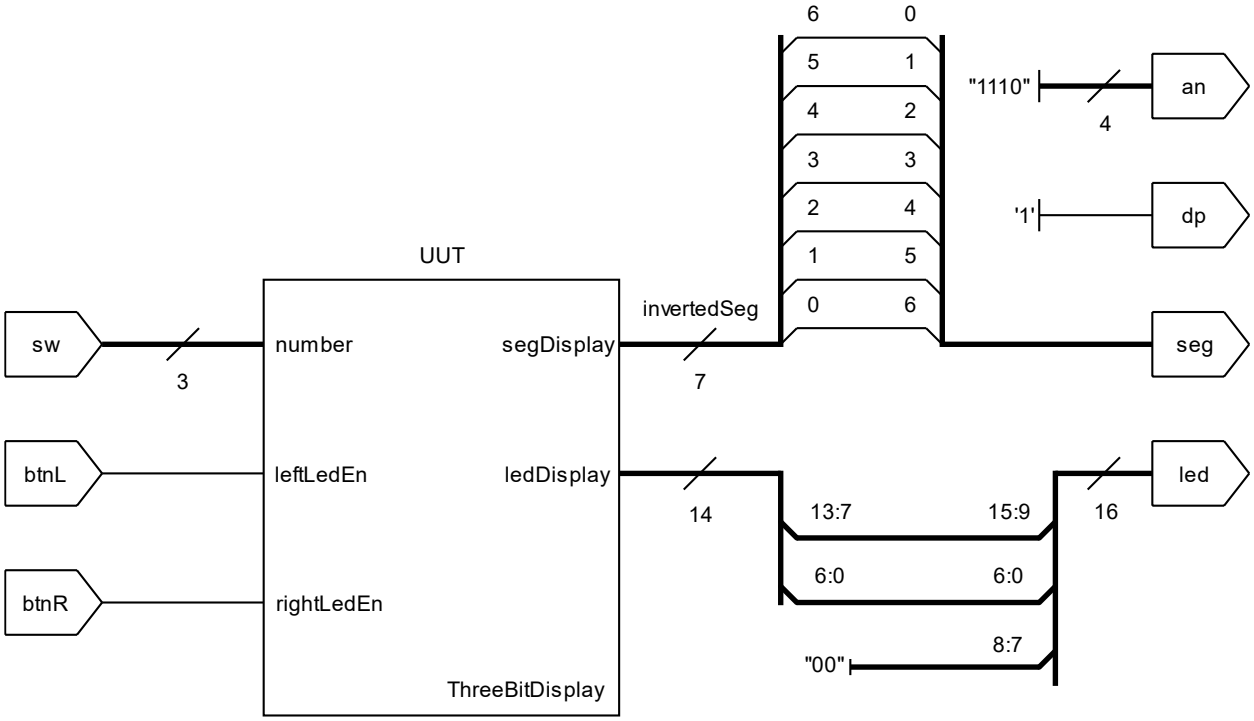
Design Block Diagram:



Test Bench Block Diagram:



Basys3 Wrapper Block Diagram:



VHDL Code:

```
--**********************************************************************
--*
--* Name: ThreeBitDisplay
--* Designer: Ayman Saad
--*
--*     This component takes in a 3 bit number and displays it as a digit
--*     on a seven-segment display and as a 2 rows of LEDs. Both rows of
--*     LEDs are enabled indivisuly with seperate inputs while the
--*     seven-segment display always remains enabled.
--*
--**********************************************************************

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity ThreeBitDisplay is
    Port (
        number:     in  STD_LOGIC_VECTOR (2 downto 0);
        leftLedEn:  in  std_logic;
        rightLedEn: in  std_logic;
        segDisplay: out STD_LOGIC_VECTOR (6 downto 0);  --active low; MSB =
a, LSB = g
        ledDisplay: out STD_LOGIC_VECTOR (13 downto 0)  --active high
    );
end ThreeBitDisplay;

architecture ThreeBitDisplay_ARCH of ThreeBitDisplay is

    constant ACTIVE: std_logic := '1';

    --7 segment display constants
    constant BLANK_SEG: std_logic_vector(6 downto 0) := "1111111";
    constant ZERO_SEG:  std_logic_vector(6 downto 0) := "0000001";
    constant ONE_SEG:   std_logic_vector(6 downto 0) := "1001111";
    constant TWO_SEG:   std_logic_vector(6 downto 0) := "0010010";
    constant THREE_SEG: std_logic_vector(6 downto 0) := "0000110";
    constant FOUR_SEG:  std_logic_vector(6 downto 0) := "1001100";
    constant FIVE_SEG:  std_logic_vector(6 downto 0) := "0100100";
    constant SIX_SEG:   std_logic_vector(6 downto 0) := "0100000";
    constant SEVEN_SEG: std_logic_vector(6 downto 0) := "0001111";

    --signals
    signal ledPattern:      std_logic_vector (6 downto 0);
    signal ledDisplayLeft:  std_logic_vector (6 downto 0);
    signal ledDisplayRight: std_logic_vector (6 downto 0);




begin
    LED_DECODER: with number select
```

```
      ledPattern <= "0000000" when "000",
                    "0000001" when "001",
                    "0000011" when "010",
                    "0000111" when "011",
                    "0001111" when "100",
                    "0011111" when "101",
                    "0111111" when "110",
                    "1111111" when others;

   LED_SELECT_LEFT: with leftLedEn select
      ledDisplayLeft <= ledPattern        when ACTIVE,
                        (others => '0')   when others;

   LED_SELECT_RIGHT: with rightLedEn select
      ledDisplayRight <= ledPattern        when ACTIVE,
                         (others => '0')   when others;

   SEG_DECODER: with number select
      segDisplay <= ZERO_SEG  when "000",
                    ONE_SEG   when "001",
                    TWO_SEG   when "010",
                    THREE_SEG when "011",
                    FOUR_SEG  when "100",
                    FIVE_SEG  when "101",
                    SIX_SEG   when "110",
                    SEVEN_SEG when others;

    ledDisplay <= (ledDisplayLeft & ledDisplayRight);

end ThreeBitDisplay_ARCH;
```

Test Bench Code:

```
--**********************************************************************
--*
--* Name: ThreeBitDisplay_TB
--* Designer: Ayman Saad
--*
--*     This component tests the functionality of the ThreeBitDisplay
--*     component by testing all possible inputs and generating a
--*     waveform of all inputs and outputs.
--*
--**********************************************************************

library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;

entity ThreeBitDisplay_TB is
end ThreeBitDisplay_TB;

architecture Behavioral of ThreeBitDisplay_TB is
    component ThreeBitDisplay
        port (
            number:     in  std_logic_vector (2 downto 0);
            leftLedEn:  in  std_logic;
            rightLedEn: in  std_logic;
            segDisplay: out std_logic_vector (6 downto 0);
            ledDisplay: out std_logic_vector (13 downto 0)
        );
    end component;

    --UUT signals
    signal number:      std_logic_vector (2 downto 0);
    signal leftLedEn:   std_logic;
    signal rightLedEn:  std_logic;
    signal segDisplay:  std_logic_vector (6 downto 0);
    signal ledDisplay:  std_logic_vector (13 downto 0);

begin
    UUT: ThreeBitDisplay port map(
        number     => number,
        leftLedEn  => leftLedEn,
        rightLedEn => rightLedEn,
        segDisplay => segDisplay,
        ledDisplay => ledDisplay
    );




    TESTBENCH_DRIVER : process
        variable counter: std_logic_vector (4 downto 0);
    begin
        --iterate over all possible inputs to find all possible outputs
        for i in 0 to 31 loop
            counter := std_logic_vector(TO_UNSIGNED(i, counter'length));
```

```
            --map counter number to UUT inputs
            number      <= counter(2 downto 0);
            rightLedEn  <= counter(3);
            leftLedEn   <= counter(4);
            wait for 20 ns;
        end loop;
        wait;
    end process;

end Behavioral;
```

Basys3 Wrapper Code:

```
--
**********************************************************************
--*
--* Name: Basys3_ThreeBitDisplay
--* Designer: Ayman Saad
--*
--*     This component serves as a wrapper for the ThreeBitDisplay
--*     component to the Basys3 board. The number displayed is selected
--*     with switches 0 to 2, while the digit is displayed on the rightmost
--*     seven-segment display and the LEDs are displayed with the boards
--*     LEDs. LED7 and LED8 are unused since only numbers 0 to 7 are
--*     available as inputs.
--*
--
**********************************************************************

library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity Basys3_ThreeBitDisplay is
    port (
        sw:   in  std_logic_vector (2 downto 0);
        btnL: in  std_logic;
        btnR: in  std_logic;
        led:  out std_logic_vector (15 downto 0); --active high
        seg:  out std_logic_vector (6 downto 0);  --active low; MSB = g, LSB
= a
        dp:   out std_logic;                      --active low
        an:   out std_logic_vector (3 downto 0)   --active low
    );
end Basys3_ThreeBitDisplay;

architecture Basys3_ThreeBitDisplay_ARCH of Basys3_ThreeBitDisplay is
    component ThreeBitDisplay
        port (
            number:      in  std_logic_vector (2 downto 0);
            leftLedEn:   in  std_logic;
            rightLedEn:  in  std_logic;
            segDisplay:  out std_logic_vector (6 downto 0); --active low; MSB
= a, LSB = g
            ledDisplay:  out std_logic_vector (13 downto 0) --active high
        );
    end component;

    --holds the direct segDisplay output before it is sent to seg
    signal invertedSeg: std_logic_vector (6 downto 0);
```

```
begin
    UUT: ThreeBitDisplay port map(
        number                  => sw,
        leftLedEn               => btnL,
        rightLedEn              => btnR,
        ledDisplay(13 downto 7) => led(15 downto 9), --left side LEDs
        ledDisplay(6 downto 0)  => led(6 downto 0),  --right side LEDs
        segDisplay              => invertedSeg
    );

    --reverse bit order of 7 segment display
    seg(0) <= invertedSeg(6);
    seg(1) <= invertedSeg(5);
    seg(2) <= invertedSeg(4);
    seg(3) <= invertedSeg(3);
    seg(4) <= invertedSeg(2);
    seg(5) <= invertedSeg(1);
    seg(6) <= invertedSeg(0);

    an <= "1110";               --enable only rightmost seven-segment display
    dp <= '1';                  --disable decimal point on display
    led(8 downto 7) <= "00";    --disable center LEDs

end Basys3_ThreeBitDisplay_ARCH;
```