LAB 3 – 3300L

Fall 2024 / Dr. Van Blerkom

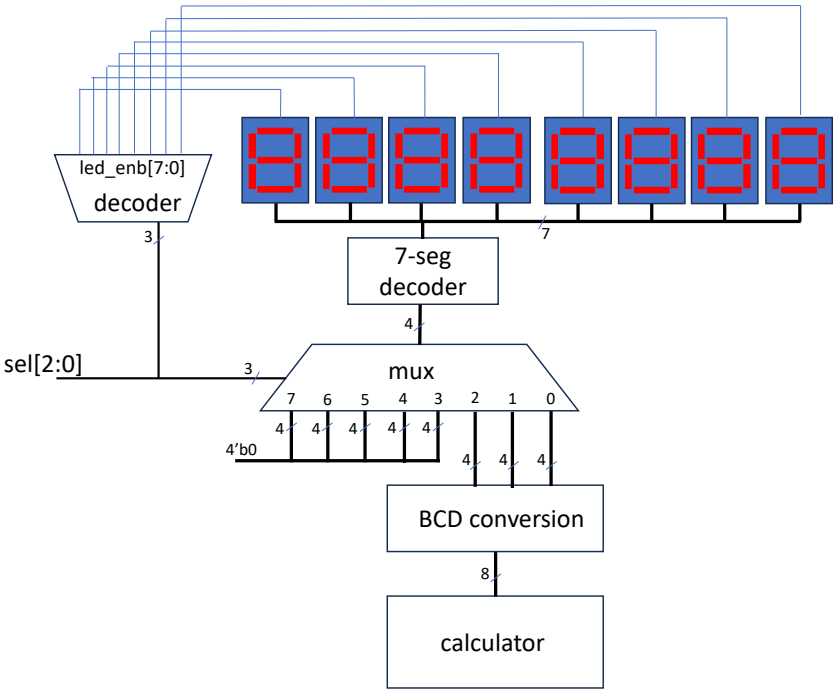
For this lab, we will add the BCD "double-dabble" converter to the calculator from lab 2, and then we will display the output on the seven-segment displays, using the verilog from lab 1. You should also output the carry_out and overflow signals to 2 individual LEDs.

Follow the block diagram below. The 8 bit value from the calculator will enter the BCD "double-dabble" block, which will convert it to a 12 bit output, where each 4-bit nibble represents one decimal digit.

Then, take the 12 bit output and put it through a multiplexor, which will select one of the three 4-bit nibbles at a time, controlled by a three-bit select signal. The 4-bit output from the multiplexor will then go to the seven-segment display decoder from lab 1, which will create the 7 outputs to produce the correct digit on the display. There will be three valid decimal digits from the BCD block – tie the fourth through eighth 4-bit inputs of the multiplexor to 0, so that the top digits display 0.

The code will also send the three-bit select signal to a decoder to enable the correct seven-segment block, when that digit's output is selected. Note that to enable a seven-segment block, the enable output needs to be set to 0, while the unselected seven-segment block enables should be set to 1. The truth table for the decoder is shown in the block diagram.

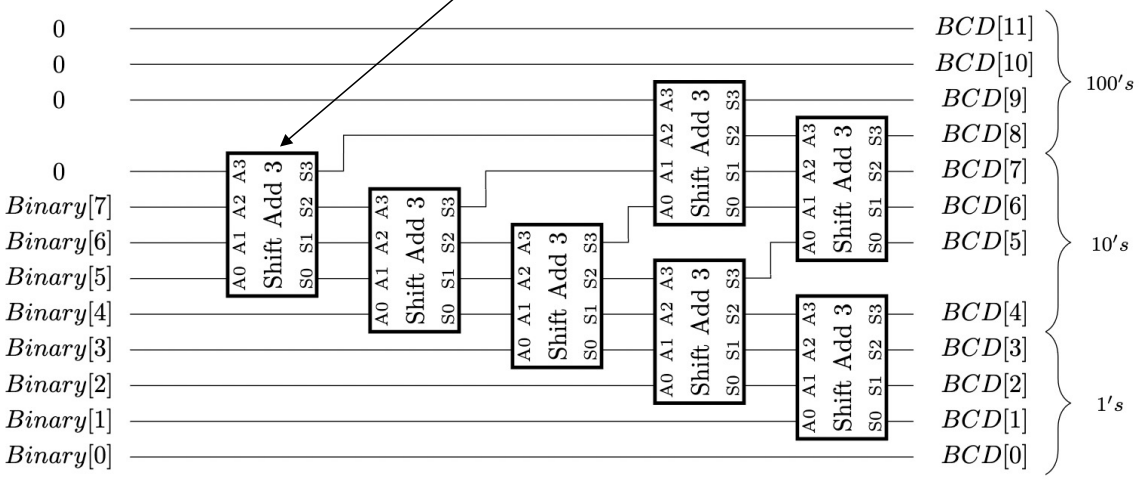
sel[2:0]	led_enb[7:0]
000	11111110
001	11111101
010	11111011
011	11110111
100	11101111
101	11011111
110	10111111
111	01111111



Block diagram of implementation

“Double Dabble” combinatorial implementation

$A[3]$	$A[2]$	$A[1]$	$A[0]$	$S[3]$	$S[2]$	$S[1]$	$S[0]$
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X



The three-bit select signal will be created by a counter that will cycle through each digit repeatedly. The code to generate this select signal is shown below. You will need to instantiate this module, and connect the clk signal up to your top level module so that it can be connected to the clock pin on the FPGA. Uncomment the clock related items in the XDC constraint file as shown, and change to port name to match the name you used on your top module for the clock.

```
module count_3bit_select(
    input clk,
    output [2:0] sel
    );

    reg [16:0] cntbig;

    always @(posedge clk)
    begin
        cntbig <= cntbig + 1;
    end

    assign sel = cntbig[16:14];
endmodule
```

Verilog to generate select signal

```
## Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { clk }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk}];
```

XDC file changes to provide a clock to your design

Extra Credit:

Modify your code so that you can produce a minus sign on the top 7-segment block, when the output is negative (note, this only happens when we have the adder/subtractor selected).

To make the minus sign, you can add a line to the case statement for the 7-segment decoder to use one of the unused values to produce a minus sign, i.e. if you give the decoder an input of 4'b1010 = 10, you could produce a minus sign.

Then you will need to add a multiplexor to choose whether to output the minus sign or a zero for the top digit, depending on if the output is negative. In addition, you will need to change the 2's complement negative value from the calculator back to a positive value for display.