ECEN 240 Lab 8 - ALU with Flip Flops

(Vivado and SystemVerilog Instructions)

1. Create a Lab8_ALU_FF project from the Lab6_ALU project. To do this:

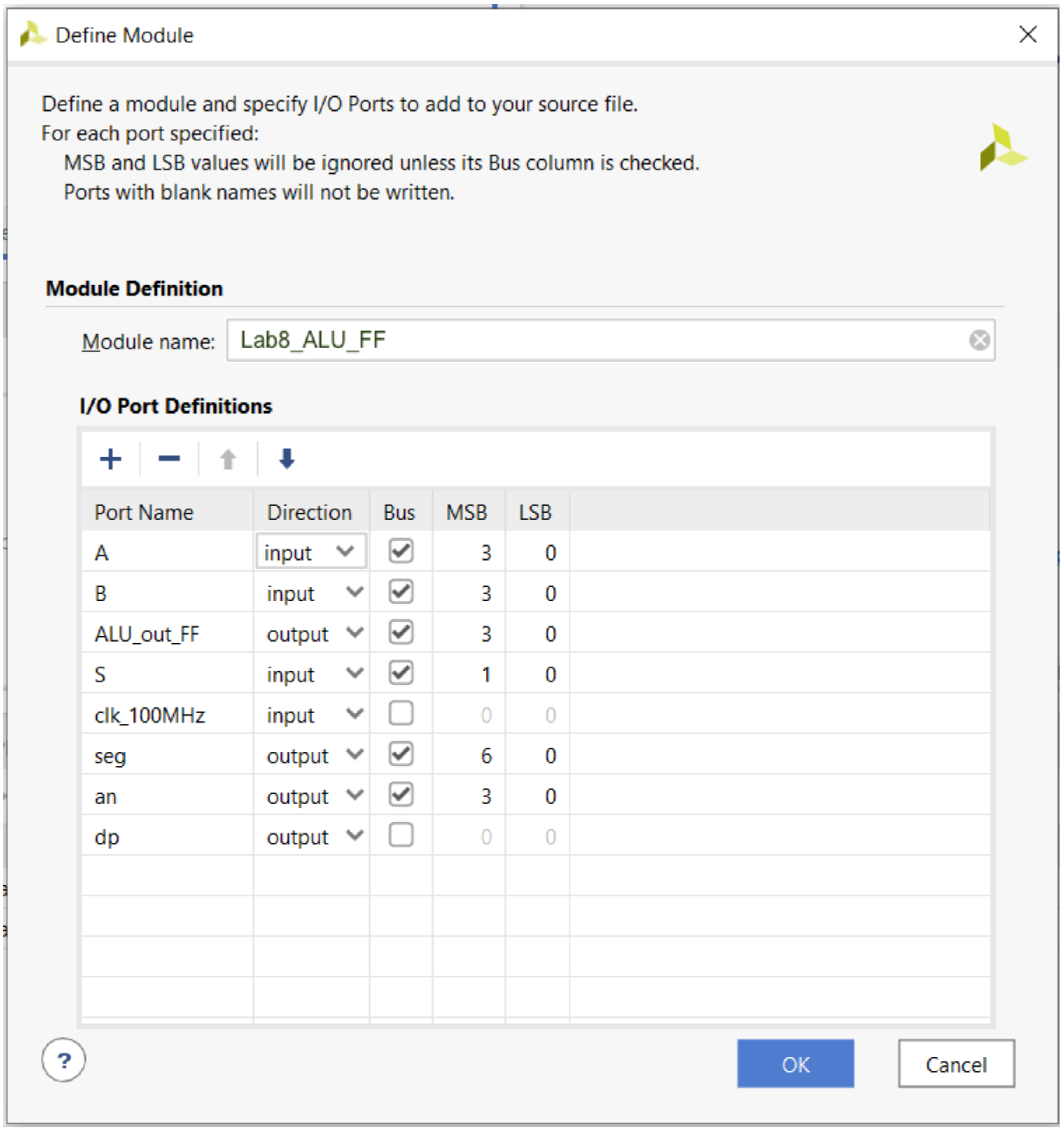
- Open the Lab6_ALU project
- From the "File" menu, select "Project > Save As" and enter the new project name as "Lab8_ALU_FF".

2. Download the "clk_div.v" module file from canvas and add it to your project as a source. To do this:

- Save the "clk_div.v" module to your desktop
- Select "Add Sources" sources from the Vivado menu (on the left).
- Select "Add or create design sources" and click "next".
- Click on the "Add Files" button
- Navigate to the "clk_div.v" file on your desktop and click on the "Finish" button.

3. Create a new module file called "Lab8_ALU_FF.sv" by doing the following:

- Select "Add Sources" from the left Vivado menu.
- Select "Add or create design sources" and click "next".
- Click on the "Create File" button.
- Select the File type as "SystemVerilog.
- Enter the file name "Lab8_ALU_FF.sv".
- The File location should be "Local to Project".
- When you click on the "Finish" button a window opens for you to describe the ports. Type the following and then "OK":



Define Module

Define a module and specify I/O Ports to add to your source file.
For each port specified:
MSB and LSB values will be ignored unless its Bus column is checked.
Ports with blank names will not be written.

Module Definition

Module name: Lab8_ALU_FF

I/O Port Definitions

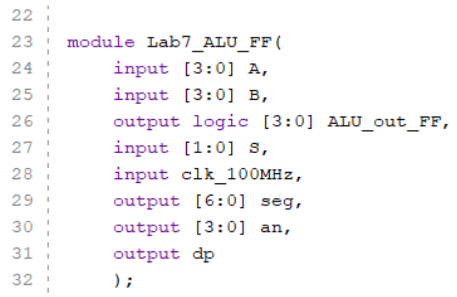
Port Name	Direction	Bus	MSB	LSB
A	input	✓	3	0
B	input	✓	3	0
ALU_out_FF	output	✓	3	0
S	input	✓	1	0
clk_100MHz	input	☐	0	0
seg	output	✓	6	0
an	output	✓	3	0
dp	output	☐	0	0

4. The “seg” signal is 7 bits wide and controls the seven segments in the seven segment displays. The “an” signal is four bits wide and selects which of the four seven-segment displays will be active. The “dp” signal is the decimal point of the seven-segment display.

The seven-segment displays will be used to display the clock signal. When the lower signals are on, that represents the low level of the clock. When the upper segments are on, that represents the high level of the clock. This way, you will be able to "see" the clock operate, and you will be able to see when the flip flop should be capturing new data.

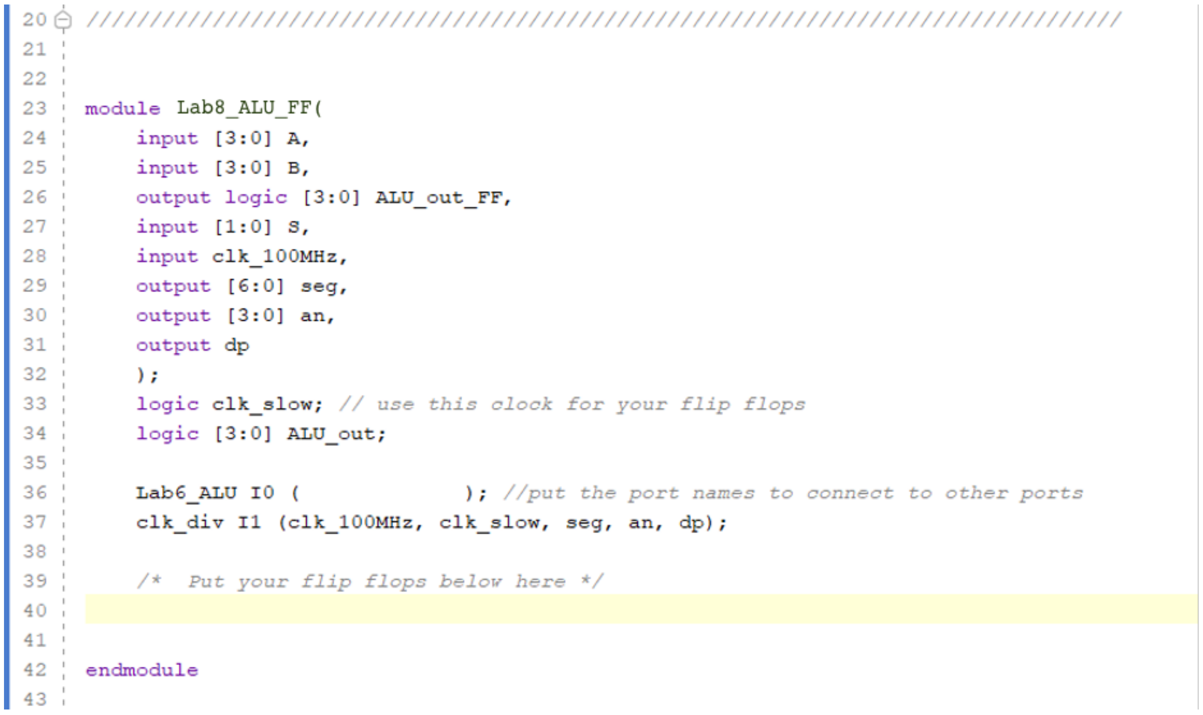
If you make a mistake, that is fine. Open the "Lab8_ALU_FF" text file and make the change within the file port list code. A few things to check:

- The "A" and "B" ports are input busses that are 4-bits wide (3 to 0).
- The "ALU_out_FF" port is an output bus that is declared as a 4-bits wide (3 to 0).
- "seg", "an", and "dp" should be the "output" direction.
- Insert "logic" after the "output" on the "ALU_out_FF" signal.



```
module Lab7_ALU_FF(
    input [3:0] A,
    input [3:0] B,
    output logic [3:0] ALU_out_FF,
    input [1:0] S,
    input clk_100MHz,
    output [6:0] seg,
    output [3:0] an,
    output dp
    );
```

5. Within the of the "Lab8_ALU_FF" module, instantiate the "Lab6_ALU" module and the "clk_div" module as shown:



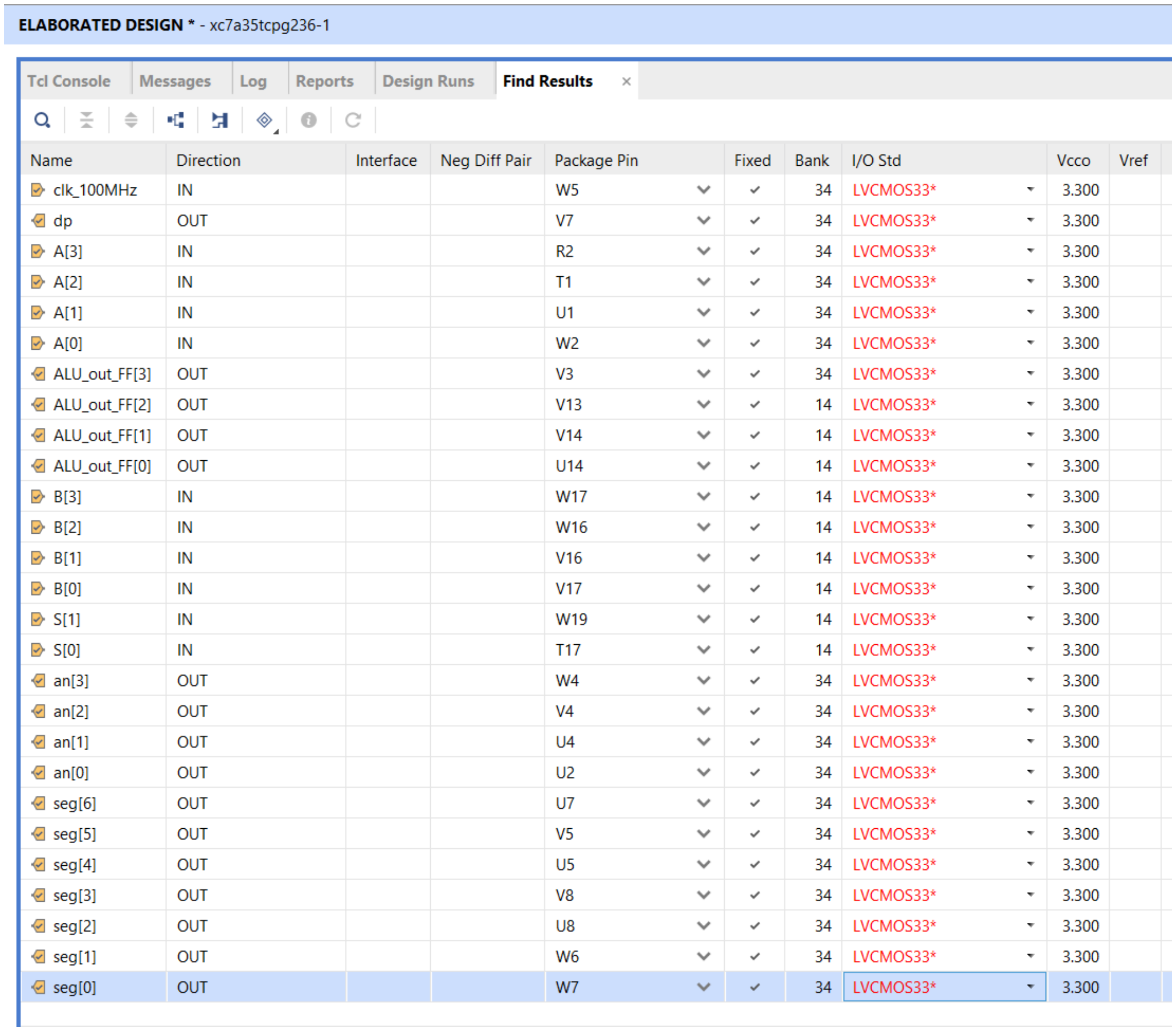
```
//////////////////////////////////////////////////////////////////////////////////


module Lab8_ALU_FF(
    input [3:0] A,
    input [3:0] B,
    output logic [3:0] ALU_out_FF,
    input [1:0] S,
    input clk_100MHz,
    output [6:0] seg,
    output [3:0] an,
    output dp
    );
    logic clk_slow; // use this clock for your flip flops
    logic [3:0] ALU_out;

    Lab6_ALU I0 (              ); //put the port names to connect to other ports
    clk_div I1 (clk_100MHz, clk_slow, seg, an, dp);

    /*  Put your flip flops below here */


endmodule
```

6. Add flip flops to store the output of the ALU

- Your Lab6_ALU will give you a 4-bit output signal. Use this as the input to the flip flop, and the output is the 4-bit "ALU_out_FF" signal.

7. Open the "Elaborated" design schematic.

8. Tell Vivado which pins to use on the FPGA chip. Click on the blue "27 I/O Ports" at the top of the schematic menu. Type the following information into the I/O Ports list. This tells Vivado how to map the signal to the switches, seven segment display, and the clocks. You also need to tell Vivado to use 3.3V by selecting the "LVCMOS33" for the I/O Std of each of these signals (to see how these pin assignments were made, look at the last figure in this document).



ELABORATED DESIGN * - xc7a35tcpg236-1

Tcl Console | Messages | Log | Reports | Design Runs | **Find Results**

Name	Direction	Interface	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref
clk_100MHz	IN			W5	✓	34	LVCMOS33*	3.300	
dp	OUT			V7	✓	34	LVCMOS33*	3.300	
A[3]	IN			R2	✓	34	LVCMOS33*	3.300	
A[2]	IN			T1	✓	34	LVCMOS33*	3.300	
A[1]	IN			U1	✓	34	LVCMOS33*	3.300	
A[0]	IN			W2	✓	34	LVCMOS33*	3.300	
ALU_out_FF[3]	OUT			V3	✓	34	LVCMOS33*	3.300	
ALU_out_FF[2]	OUT			V13	✓	14	LVCMOS33*	3.300	
ALU_out_FF[1]	OUT			V14	✓	14	LVCMOS33*	3.300	
ALU_out_FF[0]	OUT			U14	✓	14	LVCMOS33*	3.300	
B[3]	IN			W17	✓	14	LVCMOS33*	3.300	
B[2]	IN			W16	✓	14	LVCMOS33*	3.300	
B[1]	IN			V16	✓	14	LVCMOS33*	3.300	
B[0]	IN			V17	✓	14	LVCMOS33*	3.300	
S[1]	IN			W19	✓	14	LVCMOS33*	3.300	
S[0]	IN			T17	✓	14	LVCMOS33*	3.300	
an[3]	OUT			W4	✓	34	LVCMOS33*	3.300	
an[2]	OUT			V4	✓	34	LVCMOS33*	3.300	
an[1]	OUT			U4	✓	34	LVCMOS33*	3.300	
an[0]	OUT			U2	✓	34	LVCMOS33*	3.300	
seg[6]	OUT			U7	✓	34	LVCMOS33*	3.300	
seg[5]	OUT			V5	✓	34	LVCMOS33*	3.300	
seg[4]	OUT			U5	✓	34	LVCMOS33*	3.300	
seg[3]	OUT			V8	✓	34	LVCMOS33*	3.300	
seg[2]	OUT			U8	✓	34	LVCMOS33*	3.300	
seg[1]	OUT			W6	✓	34	LVCMOS33*	3.300	
seg[0]	OUT			W7	✓	34	LVCMOS33*	3.300	

9. Now run the synthesis step (side menu). If you are prompted to save the constraint file, call it "Lab8_ALU_FF".

10. After successfully running Synthesis, select "Run Implementation" and open the implemented design. Open the schematic of the implemented design. You will see that the implemented design no longer shows basic function of each module. Instead, it shows that Vivado implemented your design with several small ROMs or LUTs (Look UP Tables).

11. Select "Generate Bitstream" from the bottom of the left menu. This is turning your design into a file that can be dumped into the FPGA. It can take a while to generate the bitstream. You can follow the progress by watching top right of the Vivado window.
12. Connect the Basys3 to the computer using the USB cable, and turn on the Basys3 power switch.
13. Select "Open Hardware Manager" from the bottom of the left menu, and select "Open Target", then "Auto Connect". Once the computer has connected to the Basys3, you are ready to dump the configuration data into the FPGA. Select "Program Device".

You are now running the ALU with flip flops on the Basys3!

- Use the four switches on the far left for your "A" input number
- Use the four switches on the far right for your "B" input number
- Use the left pushbutton as the "S[1]" input
- Use the right pushbutton as the "S[0]" input
- View the output on the middle four LEDs (above the switches). This output will not update until the rising edge of clk_slow.

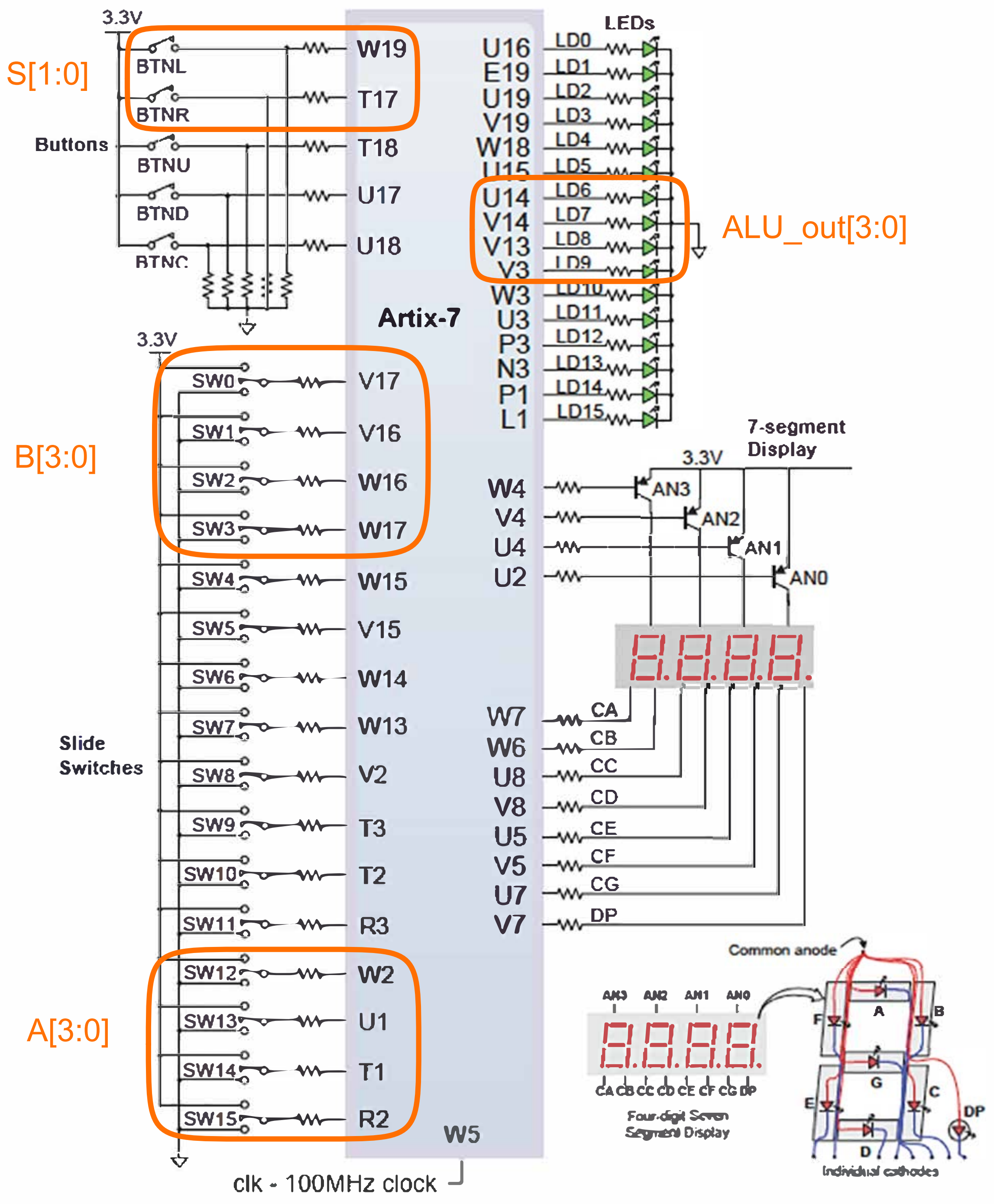


Figure 16. General purpose I/O devices on the Basys 3.