Function	ALUFN[21:16]
ADD	00XX00*
SUB	00XX01
MUL	00XX10
OR	010000
AND	010001
XOR	010010
“A”	010011
“B”	010100
NOR	011000
NAND	011001
XNOR	011010
NOT “A”	011011
NOT “B”	011100
SL	10XX00
SR	10XX01
SLA	10XX10
SRA	10XX11
CMPEQ	11X01X
CMPLT	11X10X
CMPLE	11X11X

*An ALUFN bit labelled “X” means that those input bits do not matter with regards to the function, and will be ignored rather than being treated as invalid ALUFN code.

Any other input (mostly for BOOLE where ALUFN[21:20] = 01) will be treated as invalid and just return 0, for example 0111111 (which does not correspond to a valid function) will just return 0.

Manual mode

Flip the leftmost DIP switch to “1”. The leftmost LED should light up to indicate that you are in manual mode, and the 7 segment display should light up [off 0 0 0 respectively] for to display values of Z, V and N for adder / comparison modules. When in manual mode, the instructions are as follows.

- Assign a 16-bit binary value with the DIP switches [15:0] (middle and right sets), from MSB to LSB. This value is in 2’s complement.
- Press the top button to store the value on the DIP switches as “A” into the FPGA, or the middle button to store the value as “B” instead.
- Use the DIP switches [21:16] (the 6 rightmost switches on the left set of switches) to set the ALUFN code as shown above.
- Press the bottom button to run the computation. The 16-bit output will be displayed on the LEDs above the middle and bottom set of switches. The ALUFN input will also be lit up on the corresponding LEDs, and the 7 segment display may change to show the corresponding Z V and N values where applicable.
- The values of A, B and ALUFN can be changed at any time during manual mode, i.e. you do not have to store A before B, and you can change the value of A after running one ALUFN computation.
- Reset everything back to initial values (0) by pressing the left button.

To end manual mode, flip the leftmost DIP switch down to 0.

Automatic (test case) mode

When the leftmost DIP switch is not “1”, pressing the right button instead begins automatic mode, where the FPGA will run various test cases. For each test case, you should see the following:

1. The test case number on the 7-segment display, alongside the ALUFN code on DIP[21:16].
2. “A” on the 7 segment display, followed by the 16-bit value of “A” on the LEDs.
3. “B” on the 7 segment display, followed by the 16-bit value of “B” on the LEDs.
4. An indication of “PASS” or “FAIL” on the 7-segment display.*

*During this indication, LEDs [21:16] may show 000ZVN instead of the ALUFN code, depending on what is being tested in the test case. Refer to the code for each test case to see which functions are tested in automatic mode, as well as how to interpret these LEDs. All test cases should result in a “PASS”.

Quick summary of test cases (owing to time constraints for checkoff, not all functions have test cases implemented, but each module has their proof of concept. The remaining functionalities can be confirmed using manual mode):

1. Adder (without overflow)
2. Adder (with overflow) [Proof of concept for Adder module]
3. Compare equals
4. Compare less than [Proof of concept for Compare module]
5. Bitwise AND
6. Bitwise XNOR [Proof of concept for Boolean module + extra function]
7. Shift Left unsigned
8. Shift Right signed [Proof of concept for Shifter module]
9. “B”
10. NOT “A”
11. Multiply [Showing some extra functions]