Project Presentation:

-We made an ALU that can perform 4 operations on up to two 3-bit inputs.

-Unsigned addition, unsigned subtraction, bit equality test, division by 2

-unsigned addition: 3-bit + 3-bit → 4-bit output

-unsigned subtraction: input a > input b only as per specifications, 3-bit – 3-bit → 3-bit output, 7 is max value and 0 is min value, the carry out is ignored

-bit equality test: compare input a to input b and light up LED0 if equal

-division by 2: shift input a 1-bit to the right and output on LED1LED0, since output is at most 3 and at least 0

-General approach involved creating the 4 separate data paths with a mux that selects the final output of the operations according to the user input

-2 select switches to pick operation, output is displayed on 4 LED’s.

Individual Defense:

Design choices:

-Picked the multiple data path and mux approach because it is conceptually easier

-Reused the adder for unsigned subtraction because it saved on creating a new module specifically for unsigned addition.

Problems encountered:

-Unsigned subtraction wasn’t working because there was an overflow bit from doing two’s complement and the 3-bit adder was not narrow enough to ignore the carry out bit, so we applied a bit mask to the first bit.

-Problems with the test bench – we copied over the verilog modules and ucf into an empty project and it just worked.

-General verilog code issues, such as assigning a wire vs register, instantiating instances of modules, etc.

Improvements:

-Taking the ALU extender and adder approach would improve the scalability of the design if we were to add more components.

-Using wider inputs/outputs, more operations like gt/lt, n-bit shift, wider adder, signed addition and subtraction, etc.