1 – Modify the assembler to accept hex input when the string 0x is encounter (do not force input as decimal). Alternatively create an addu operation which reads the value as an unsigned number.

2 – As implemented the assembler does not check if the memory address being accessed is in the correct segment. Since the addresses in the memory and data segment are the same values, this can lead to a program accessing invalid memory as an instruction or to retrieve a data value. Implement a check in the assembler program that ensures label is in the correct memory segment, e.g. if the instruction expects a data value, the label must refer to an address must be in the data segment; if the instruction expects a value for a branch to another instruction, the label must refer to an address in the text segment.

3 – Implement the br instruction, which is an unconditional branch statement. This branch should be taken regardless of the value of the $ac register.

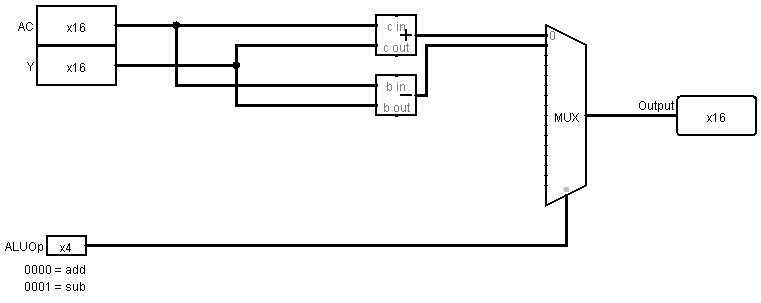
4 – Implement exception handling in CPU is an overflow condition is encountered. For now, the exception should just branch to address 0xdf, and the instructions at 0xef are

clac

beqz 0xef

If the br instruction has been implemented, just branch to 0xff, and the instruction at 0xff should be a br 0xff.

5 – Implement the following for the ALU. Note that this ALU can be substituted directly for the current ALU, and is in the Logisim CPU circuit. The name of this ALU is ALU1.



To this ALU add the following operations. Update the assembler, and write programs to tests your results.

1. Multiply (mul, ALUOpt=0010)
2. Divide (div, ALUOpt=0011)
3. Remainder (rem, ALUOpt=0100)
4. Shift Left Logical (sll, ALUOpt=0101)
5. Shift Right Logical (srl, ALUOpt=0110)
6. Shift Right Arithmetic (sra, ALUOpt=0111)
7. Or operation (or, ALUopt=1000)
8. AND operation (and, ALUopt=1001)
9. NOT operation (not, ALUopt=1010)
10. XOR operation (xor, ALUopt=1011)

* Why wasn’t a Shift Left Arithmetic operation included?
* Without changing the ALUOpt code, how many total operations can be specified in this ALU?

6 – Using an 8 bit register named $temp, create a mechanism to load 16 bit immediate values into the $ac. To do this, add 2 operators, the lti and movei operators. They will work as follows:

1. lti immediate will load the 8-bit immediate value into the $temp register. This is the high 8 bits for the value (bits 8...15)
2. movei immediate will move the 8-bits from temp to the high bits in the $ac, and move the 8-bit immediate value into the low 8-bit of the $ac register (bits 0…7).

The following would load the value 1,440,826 (0x15FC3A) into memory (note: this is easier if problem 1 is completed. If not the lower 2 bytes have to be entered as a negative number).

lti 21

movei -966

To complete this problem, the assembler must be changed, the control unit must be changed, an 8-bit temp register must be added to the CPU, a splitter must be added to combine the $temp and immediate values, and a mux must be added to save this value if it is selected.