Casey McGuire
CS281
November 3rd, 2014

Ainv = 0

1)

a)Ainvert would be 1, Binvert would be 1, CarryIn would be 1, and Operation would be 10.

b)Ainvert would be 1, Binvert would be 0, CarryIn would be 1 or 0 (doesn’t matter), and Operation would be 00.

c)Ainvert would be 1, Binvert would be 0, CarryIn would be 1 or 0 (doesn’t matter), and Operation would be 01.

2)

a) Ainvert would be 0, Binvert would be 1 and Operation would be 11. This is because we are trying to determine if a < b. The way to do this is to subtract b from a and to put the sign bit (the most significant bit) into the least significant bit’s place.

b)Ainvert would be 0, Binvert would be 1 and Operation would be 10. This is because we are testing whether a and b are equal. In order to do this, we subtract b from a and then OR every output. Since the output of this OR gate is negated, the only way ’zero’ can contain a valid output (that is, a 1) is if every output is a zero.

2b) There is ALWAYS an output on the Zero line, regardless of the operation we've asked the ALU to perform.

c) This would set all the bits to zero except for the least significant bit, which would be the result of adding the most significant bits of the input. That is, if the most significant bits of the inputs are both 1s or both 0s, the least significant bit of the output will be zero. On the other hand, if one of the most significant bits of the input is 0 and the other is 1, then the least significant bit of the output will be 1. This is possibly testing whether the inputs have the same sign bit. That is, it returns a 1 if they don’t have the same sign bits and a zero if they do.

2c) This will return a 1 in the output if the SUM of the two integers is negative, and a 0 otherwise.