Bit accurate data types exercise

Anh Huy Bui - 293257

Questions #1: What value did you choose for WIDTH and why? How did you calculate RESULT\_WIDTH during compile time? What is its value using your value for WIDTH and the given N = 8?

For a range of [-9, 18], I choose WIDTH = 6, which has a range from -32 to 31.

I assume that choosing WIDTH = 6 means range of [-32, 31].

One case that return highest value of output is that all elements of the input matrixes are -32 or 31. For that, the highest value of output (with N = 8) can be:

(-32)\* (-32)\*8 = 8192 or 31\*31\*8 = 7688.

In that case, the range of output should higher than 8192, and also a (power of 2) -1, which is 16383

Another case that return lower value of output is that all elements of one input matrix are -32, all elements of the other input matrix are 31. For that, the lowest value of output (with N = 8) can be:

(-32)\* (31)\*8 = -7936

In that case, the range of output should lower than -7936, and also a -(power of 2) , which is -16384.

Therefore, RESULT\_WIDTH = 15 and range [-16384, 16383]

Questions #2: What is the post-assignment total area score for both designs? Is the difference between them surprisingly large/small? If so, speculate why.

The area when Algorithmic C data types are used is significantly smaller, about a half compare to original C++ datatypes and also simulation time also lower.

This is because Algorithmic C data types are optimized with the choices of bit widths, so that we can choose a suitable range for our data without having to execute extra bit widths.

Questions #3: What values did you choose for WIDTH and INT\_WIDTH and why? How did you calculate MULT\_WIDTH now?

For range [-10, 19] and we want to be able to represent the numbers accurately to the precision of the third decimal, I choose WIDTH = 10 and INT\_WIDTH = 6.

So that we can have a range of [-32, 31.9375].

Same as question 1, the range have to higher than maximum value for output which can be (-32)\* (-32)\*8 = 8192 and we want to be able to represent the numbers accurately to the precision of the third decimal. So I choose RESULT\_WIDTH = 19 and MULT\_WIDTH = 15, I will have range for the result [-16384, 16383.9375]

Question #4: What is the total post-assignment area of the fixed-point design?

Total post-assignment area is 159101.292

Questions #5: What is the total post-assignment area score for each case? Analyze the difference between the scores.

Total post-assignment area

+ Use “+”: 347.524

+ Use “-”: 364.465

+ Use “\*“: 2723.738

+ Use “/“: 6789.600

+ Divide by 7: 1524.673

+ Divide by 8: 273.586

Adder and subtractor are one of basic units of operators so it cost little amount of area. Meanwhile, multiplier and divider are combined of many adder, which result in cost of a large amount of area

Divide by 8 costs much less area than divide by 7 because 8 is a power of 2, costs nothing in additional area.