1. Review textbook supplement material 3.11, located on Blackboard. Write 4 – 5 sentences about what you learned.

Answer: First thing I learned was just how much work it took to get floating point number correctly calculated and represented in computer. Computers used to cut accuracy when doing floating points, and some couldn’t do it at all without errors. It took quite awhile for it to be fixed and calculated correctly.

1. Explain why in multiplication parallel processing can be used, but in division, it cannot. Answer must include how the computer circuitry is used in each. (4 – 5 sentences)

Answer: With multiplication, parallel processing is possible. This is due to the fact that we can partition carry chains, then one can work on operations at the same time as the others. With division, it isn’t possible. When dividing, we must first know the size of the difference before we can preform the next operation.

1. Why must the designer of a floating-point representation find a compromise between the size of the fraction and the size of the exponent? Answer must include information about precision and range. (4 – 5 sentences)

Answer: The main reason there must be a compromise is space. Space for floating-point number are limited to 32-bit and 64-bit for single and double precision respectively. 1 bit is reserved for the sign of the number, so that cuts the space to 31 and 63. With that limited space, you must still store a floating-point numbers fraction and exponent. When you add a bit to one, you take one away from the other. When designing floating-point representation, you must decide if they need more precision by giving more bits to the fraction while taking bits from the exponent, or a wider range of numbers which will give more bits to the exponent and take from the fraction.

1. Explain “Good design demands good compromises” and provide an example. (4 – 5 sentences)

Answer: When designing a system, you can’t do all the things you want to with the system. Therefore, long the way you must design a system must be simple and quicker. So you must make compromises to reach that design. One example is the MIPS design. The designers make all the instructions the same size at the expense of one instruction format. With doing that though it brought about the need for multiple instruction formats as different cases needed more size for the instructions.

1. Explain the difference between a 32-bit unsigned and a 32-bit signed binary word. Answer must include size considerations. (4 – 5 sentences)

Answer: The same with a single-point precision number, a 32-bit binary word only has 32-bits to represent its decimal number. The 32-bit word can use all of it’s bits for a whole number from 0 to 232 – 1. Now, if you want to represent a negative number you must have a bit that says which sign your number is. Taking one bit for the sign decreases the range of numbers you can represent with the remaining one. So a 32-bit *signed* binary number would go from −(231)  to 231 – 1. Your range is the same, but the size of the numbers are about half of your unsigned number.