JHU Computer Organization Module 2 Hint: IEEE 754

Joseph G. Kovba

September 9, 2020

1 Converting an IEEE 754 Decimal Number to Binary

In this example we're going to take a decimal number and convert it into IEEE 754 single-precision floating point format. Let's use 32.5_{10} as an example. The IEEE 754 format gives us 1 sign bit, 8 exponent bits, and 23 fraction/mantissa bits. We'll start with the sign bit which is straightforward. Since our decimal number is positive, our sign bit $s = 0$.

Next we will convert the whole number and fractional portions into binary separately. $32_{10} = 100000_2$ and $0.5_{10} = 0.1_2$. Combining these two parts together we get: $32.5_{10} = 100000.1_2$. To get this number in "normalized" format, we must move the "radix" (decimal point) 5 spots to the left. Just like we multiply by 10 or -10 in decimal to move the decimal decimal point right and left, in binary, we can accomplish this by multiplying by positive or negative powers of 2. So, $100000.1_2 = 1.000001_2 \times 2^5$. Notice we now have an exponent $e = 5$. This is not quite our final exponent, though. When converting from decimal into binary IEEE 754 format, we must add 127 to our exponent to account for the "bias" (which serves primarily to represent the exponent as an unsigned number rather than as a two's complement number). Adding a bias of 127 to our exponent of 5 gives us an exponent of 132. Since the IEEE 754 format gives us 8 bits to represent our exponent our final exponent is encoded in binary as: $e = 10000100_2$.

Finally we will convert our fraction/mantissa portion to 23-bit binary. From above, recall $32.5_{10} = 100000.1_2 = 1.000001_2 \times 2^5$. In IEEE 754 format, we do NOT need to represent the 1 explicitly since every number will have a 1 in this position, it's always there implicitly, so we will focus only on the fractional portion: 0.000001_2. We have the first 6 bits from converting the number to normalized format, so all we have to do is pad our number with 17 zeroes to make it the full 23-bits long: f = 000 0010 0000 0000 0000 0000.

We now have our sign bit $s = 0$, our exponent $e = 10000100_2$, and our fraction f = 000 0010 0000 0000 0000 0000. Appending these together we get the final 32-bit IEEE 754 binary string: 0100 0010 0000 0010 0000 0000 0000 0000. We can equivalently convert this to hex by converting each group of 4 bits to a single hex digit: 0x42020000. To verify this, we can plug-in to the following formula: $FP = (-1)^s \times (1 + f) \times 2^e = (-1)^0 \times (1 + 000\ 0010\ 0000\ 0000\ 0000\ 0000)_2 \times 2^{10000100_2 - 127_{10}} = 1 \times (1 + 0.000001_2) \times 2^{132-127} = 1 \times (1 + 0.000001_2) \times 2^5 = 1 \times 100000.1_2 = 1 \times 32.5_{10} = 32.5_{10}$. Boom, done!