Solutions to Homework 3

3.2

hex		binary			
`0x5ED4`		0101	1110	1101	0100
`-0x07A4`		-0000	0111	1010	0100
`0x5730`		0101	0111	0011	0000

Note that these are in sign-magnitude format, so you shouldn't be converting to two's complement.

3.4

octal		binary			
`4365`		100	011	110	101
`-3412`		-011	100	001	010
`0753`		000	111	101	011

3.7

185 = 10111001. But this is in sign-magnitude, so that really means $= -0111001$ in 7 bits. 122 = 01111010 = +1111010 in 7 bits . What we have, then, is actually a 7-bit subtraction:

$$\begin{array}{rl} 1111010 & \\ -0111001 & \\ \hline 1000001 & = 65_{10} \end{array}$$

This fits into 8 bits: 01000001. No overflow.

3.20

Two's complement: 201,326,592

Unsigned: Same as above

3.21

000011	00000000000000000000000000

This turns out to be the instruction: `jal 0`

3.22

0	00011000	00000000000000000000000

Use formula: $(-1)^s \times (1 + M) \times 2^{exp-bias} = 1.0 \times 2^{-103}$

BUT, this is **not** in decimal! This is in base 2!! Remember that the formula just plugs in the mantissa, which is in base 2, plus we're multiplying by 2, which *is* the base. Thus, multiply it out and the **decimal** answer is $9.860761315 \times 10^{-32}$.

As an example, suppose an answer came out to 1.1×2^1. This is really $1.1_2 \times 2 = 1.5_{10} \times 2 = 3_{10}$, not 2.2_{10}.

3.23

$63.25 = 111111.01 = 1.1111101 \times 2^{5+127=132=10000100}$

0	10000100	11111010000000000000000

3.29

Oh, boy.

$2.6125 \times 10^1 = 26.125 = 11010.001_2$

$4.150390625 \times 10^{-1} = 0.4150390625 = 0.0110101001_2$

Now normalize the base 2 values:

1.1010001×2^4

1.10101001×2^{-2}

To add, align the decimals by right-shifting the smaller number to get the same exponents. This means we have to right-shift the bottom value by 6. This gives us:

$$\begin{array}{rl} & 1.1010001000\ 0000 \times 2^4 \\ + & 0.0000011010\ 1001 \times 2^4 \\ \hline & 1.1010100010\ 1001 \times 2^4 \end{array}$$

But we have a machine with 16 bits, so the mantissa is only 10 bits. That means we have the extra bit pattern 1001 hanging off the right side. Since we have a guard, round, and sticky bit, that means we can only handle three of those bits while adding. Because there's one bit to the right, the sticky bit becomes 1, and we have:

$$\begin{array}{rl} & 1.1010001000\ 000 \times 2^4 \\ + & 0.0000011010\ 101 \times 2^4 \\ \hline & 1.1010100010\ 101 \times 2^4 \end{array}$$

Finally, to store the final number, the pattern 101 indicates **round up**, so the last of the 10 bits we can store will change to 1:

1.1010100011×2^4

This works out to 26.546875.

Additional problems:

1. `9FFE FFB2`$_{16}$ = 1001 1111 1111 1110 1111 1111 1011 0010 $= 2,684,288,946_{10}$
2. 1111 1111 1111 1111 1111 1110 1111 1110 $= -258_{10}$
3. 0000 0000 0001 1111 0111 1111 1111 0111 $= 2,064,375_{10}$

4.

M	A	Q	Q_{-1}	Count	Operation
10011	00000	01000	0	5	initial
	00000	00100	0	4	arithmetic shift right
	00000	00010	0	3	arithmetic shift right
	00000	00001	0	2	arithmetic shift right
	01101	00001	0		A = A - M
	00110	10000	1	1	arithmetic shift right
	11001	10000	1		A = A + M
	11100	11000	0	0	arithmetic shift right

Answer is 1110011000_2, which is $-0001101000_2 = -104_{10}$

5. Similar to the above; the quotient is in $Q = 00010$ and the remainder is in $A = 00101$. But now you must figure out the signs, as the original division was $\frac{17}{-6}$. Use the equation:

$$\text{Dividend} = \text{Divisor} \times \text{Quotient} + \text{Remainder}$$

To make this work, the quotient is -2 and the remainder is +5:

$$17 = -6 \times -2 + 5$$