COMP1521 WK07

number representations

Course Updates

- Assignment 1 was due this Monday! Congrats for making it through the first half of the course 🥳
    - 5 days to submit with the late penalty (before 0 is awarded)
- Assignment 2 releasing some time soon
    - C Assignment

Contents

- Negative numbers - Two’s Complement
    - Q1, Q2, Q3
- Floating point IEEE-754 representation
    - Q4, Q5

Two's Complement

- Negative numbers are determined by whether or not the m.s.b (most significant bit) is set to 1

Example.

-15 = 0b10001

Representing Two's Complement Numbers in Binary

1. Represent the number in binary
2. Flip all the bits (set 0s to 1s and 1s to 0s)
3. Add one to the number

Example. num = -127

```
1.  127 = -0b0111 1111
2.      =  0b1000 0000
3.      =  0b1000 0001
```

Tutorial Q2

Assume that the following hexadecimal values are 16-bit twos-complement. Convert each to the corresponding decimal value.

a. 0x0013   0000 0000 0001 0011 → 19

b. 0x1234   → 0001 0010 0011 0100 →

c. 0xffff

d. 0x8000

1000 0000 0000 0000 → -1

-1

1000 0000 0000 0001 → $-(2^{15}+1)$

Tutorial Q3

Give a representation for each of the following decimal values in 16-bit two's-complement bit-strings. Show the value in binary, octal and hexadecimal.

a. 1000

b. 100000

c. -5

d. -100

$40 - 32 = 8$

64 32

2^{15} 2^{14} 2^{13} 2^{12} 2^{11} 2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0}

0 0 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0

0x03E8

001750

Tutorial Q3

max pos → $2^{15}-1$

Give a representation for each of the following decimal values in 16-bit two's-complement bit-strings. Show the value in binary, octal and hexadecimal.

a. 1000

b. 100000 = 2^{16} = 65536

c. -5

d. -100

Tutorial Q3

Give a representation for each of the following decimal values in 16-bit two's-complement bit-strings. Show the value in binary, octal and hexadecimal.

a. 1000

b. 100000

c. -5

d. -100

1. 0000 0000 0000 0101

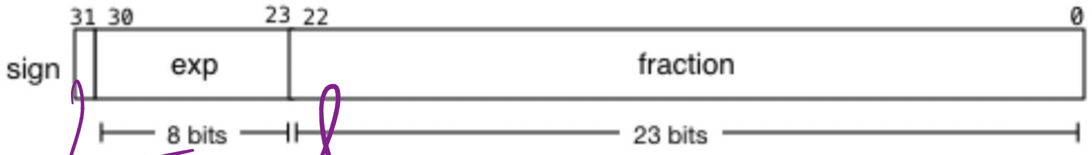
2. 1111 1111 1111 1011

0xFFFB 0177773

Floating Point Numbers

- We need a way to represent base 10 Real numbers in Binary
- Easy to represent Integers in base2 but Fractional numbers can be tough to get accurate (can cause floating point errors)
- Base10 has scientific notation
  - 6.022 * 10^23
- Base2 also has scientific notation
- IEEE-754 is the standard for floating point arithmetic

Representing Floating Point Numbers in Binary



$2^7 - 1$

$2^{(\#\ \text{exp bits})} - 1$

- **sign bit** → indicates if number is positive or negative
- **exponent bits** → unsigned 8 bit value which is interpreted from -127 to 128 by subtracting the “bias” 127
- **fraction (mantissa) bits** → 23 bits whose values are in the range 0 to 1. We add 1 to the fractional component to convert the values to be between 1 to 2

```
sign * (1 + fraction) * 2^(exponent - bias)
```

8 bit Floating Point Example

- **sign bit** → 1 bit
- **exponent bits** → 3 bits (bias = 3)
- **fraction (mantissa) bits** → 4 bits

$2^{3-1} - 1 = 3$

Example: **Convert `01000110`**

	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	exp			frac			
				1/2	1/4	1/8	1/16
0	1	0	0	0	1	1	0

sign = 0 → pos

exp = 4

$\text{frac} = \frac{1}{4} + \frac{1}{8} = \frac{3}{8}$

$$\left(1+\frac{3}{8}\right) \times 2^{4-3} = \left(1+\frac{3}{8}\right) \times 2$$

$$= 2.75$$

Tutorial Q4

What decimal numbers do the following single-precision IEEE 754-encoded bit-strings represent?

a. 0 00000000 00000000000000000000000 → 0

b. 1 00000000 00000000000000000000000

c. 0 01111111 10000000000000000000000

d. 0 01111110 00000000000000000000000

e. 0 01111110 11111111111111111111111

f. 0 10000000 01100000000000000000000

g. 0 10010100 10000000000000000000000

h. 0 01101110 10100000101000001010000

exp = 0

frac = 0

$(1+0) \times 2^{0-127}$

$\frac{1}{2^{127}} \approx 0$

b. −0

Each of the above is a single 32-bit bit-string, but partitioned to show the sign, exponent and fraction parts.

Tutorial Q4

What decimal numbers do the following single-precision IEEE 754-encoded bit-strings represent?

a. 0 00000000 00000000000000000000000
b. 1 00000000 00000000000000000000000
c. 0 01111111 10000000000000000000000
d. 0 01111110 00000000000000000000000
e. 0 01111110 11111111111111111111111
f. 0 10000000 01100000000000000000000
g. 0 10010100 10000000000000000000000
h. 0 01101110 10100000101000001010000

sign = 0 → pos

exp = 127

frac = $\frac{1}{2}$

$(1+\frac{1}{2}) \times 2^{127-127}$

$1.5 \times 2^0 = 1.5$

Each of the above is a single 32-bit bit-string, but partitioned to show the sign, exponent and fraction parts.

Tutorial Q4

What decimal numbers do the following single-precision IEEE 754-encoded bit-strings represent?

a. 0 00000000 00000000000000000000000
b. 1 00000000 00000000000000000000000
c. 0 01111111 10000000000000000000000
d. 0 01111110 00000000000000000000000
e. 0 01111110 11111111111111111111111
f. 0 10000000 01100000000000000000000
g. 0 10010100 10000000000000000000000
h. 0 01101110 10100001010000001010000

Each of the above is a single 32-bit bit-string, but partitioned to show the sign, exponent and fraction parts.

sign = 0 → pos

exp = 126

$$frac = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \dots + \frac{1}{2^{23}}$$

$$\approx 0.9999..$$

$$(1 + 0.999..) \times 2^{126-127}$$

$$= 1.999 \times 2^{-1} = \frac{1.999...}{2} \approx 1$$

Tutorial Q5

$(1 + frac) \times 2^{exp - bias}$

Convert the following decimal numbers into IEEE 754-encoded bit-strings:

a. 2.5 $= (1 + 0.25) \times 2^{1}$

$exp - 127 = 1$

$exp = 128$

$frac = \frac{1}{4} \rightarrow 010000000000\ldots$

b. 0.375

c. 27.0

d. 100.0

0 1000 0000 0100000000000...

Tutorial Q5

$$(1+\text{frac}) \times 2^{\text{exp}-127}$$

Convert the following decimal numbers into IEEE 754-encoded bit-strings:

a. 2.5

b. 0.375

c. 27.0

d. 100.0

$$(1+0.5) \times 2^{-2}$$

$\text{exp} = -2 + 127$
$= 125$

frac = 100000....

exp = 01111101

0 01111101 100000......

Special Floating Point Numbers

000000000 11 00110.

- **0** → exponent bits are 0
- **Infinity**→ exponent bits are all 1's, fraction is 0

0000000

- **NaN** → exponent bits are all 1's, fraction is not 0