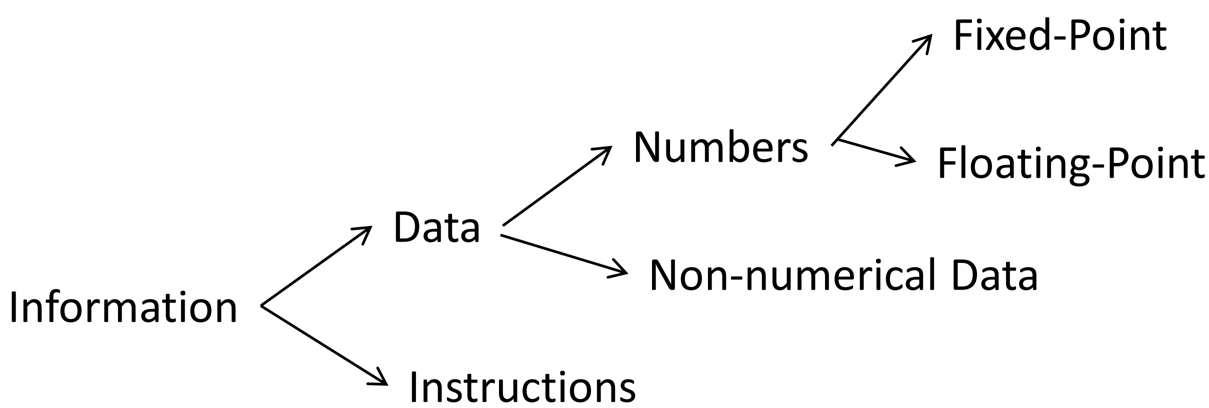
Number System

Modern Computing

- Information is made up of binary digit sequence organized in words.
- Length of the digit sequence is very important for representation.
  - Usually in the multiples of 8(called "byte")



Factors for choosing proper number representation

- The specification of the type of number to be represented.
  - Codes for integers differ from real numbers.
- The range of values that can be covered in the representation
- The precision of the representation - the maximum accuracy that has to be assured by the format
- The estimation of the hardware complexity required by the representation.

Major Number Representation In Computing Systems

- Two major approaches to represent Numbers:
  - Fixed Point
    - Generally allows representation of integers, sub-unitary fractional numbers
    - covers a limited range of values.
    - Precision depends on the number of word bits.
    - Requires a moderate hardware circuitry investment.
  - Floating Point
    - Representation of real number ranges
    - Covers a large range of values
    - Precision depends on the number of bits of one part of the representation (mantissa)
    - Hardware requirement is increased.

Fixed-Point Number Representation

- It is the representation of a number with fixed number of digits before and after the radix point.
- A fixed-point representation of a number may be thought to consist of 3 parts:
  - the sign field,
  - integer field,
  - fractional field.
- One way to store a number using a 32-bit format:
  - reserve 1 bit for the sign,
  - 15 bits for the integer part
  - 16 bits for the fractional part.
  - *A number whose representation exceeds 32 bits would have to be stored inexactly.*
- On a computer,
  - 0 is used to represent +
  - 1 is used to represent −

Fixed-Point Number Representation

- Eg:
  - String: 1 000000000101011 1010000000000000
    - It represents: $(-101011.101)_2 = -43.625$

Floating-Point Number Representation

- The floating-point notation is by far more flexible.
- In this method, a binary floating point number is represented by

$$(\text{sign}) \times \text{mantissa} \times 2^{\pm\,\text{exponent}}$$

  - Sign is one bit,
  - the mantissa is a binary fraction with a non-zero leading bit,
  - the exponent is a binary integer.
- To store a normalized number in 32-bit format:
  - 1 bit for the sign,
  - 8 bits for the signed exponent,
  - 23 bits for the portion b1b2b3...b23 of the fractional part of the number.
  - The leading bit 1 is not stored (as it is always 1 for a normalized number) and is referred to as a “hidden bit”

Floating-Point Number Representation

- Any $x \neq 0$ may be written in the form:

  $$\pm(1.b_1b_2b_3\ldots)_2 \times 2^n ,$$

  – called the normalized representation of x.
- The normalized representation is achieved by choosing the exponent n so that the binary point “floats” to the position after the first nonzero digit.
- This is the binary version of scientific notation.
- Exponent is usually represented in excess representation or bias representation:
  – excess representation: actual exponent + Bias
  – It ensures exponent is unsigned
  – Eg: If exponent is 8 bit bias is 127

Floating-Point Number Representation

Usually done by other ways other than '00000101' like bias representation

- Eg:
  - String: 1 (8-bit to represent 5) 10101100000000000000000
    - It represents: $(-1.101011)_2 \times 2^5 = (-110101.1)_2 = -53.5$

IEEE 754 standard

- Modern computers adopt IEEE 754 standard for representing floating-point numbers.
- There are two representation schemes:
  - 32-bit single-precision
  - 64-bit double-precision.
- Representation:

1 bit	Single: 8 bits	Single: 23 bits
	Double: 11 bits	Double: 52 bits
Sign	Exponent	Fraction

•Exponent Bias:
  - Single precision - 127
  - Double precision - 1203

•Fraction:
  - Normalized-Always has a leading pre-binary-point bit of 1 (hidden bit)

- Number :

$x = (-1)^{Sign} \times (1 + Fraction) \times 2^{(Exponent - Bias)}$

IEEE-754 32-bit Single-Precision Floating-Point Numbers

- In 32-bit single-precision floating-point representation:
  - The most significant bit is the *sign bit* (S),
    - 0 for positive numbers
    - 1 for negative numbers.
  - The following 8 bits represent *exponent* (E).
    - Bias: 127
  - The remaining 23 bits represents *fraction* (F).
    - Normalized and has a hidden bit of 1

IEEE-754 32-bit Single-Precision Floating-Point Numbers

- Eg: 1 1000 0001 011 0000 0000 0000 0000 0000
    - S = 1
    - E = 1000 0001
    - F = 011 0000 0000 0000 0000 0000
  - In the *normalized form*, the actual fraction is normalized with an implicit leading 1 in the form of 1.F.
    - Hence Actual fraction: 1.011 0000 0000 0000 0000 0000
    - 1.011 0000 0000 0000 0000 0000 = $1x2^{0}+1x2^{-2}+1x2^{-3}$ = 1.375
  - S=1, this is a negative number
  - With excess representation the actual exponent is E-127
    - Hence Actual exponent: 129-127=2
  - The number represented is -1.375x 2^{2} = -5.5

IEEE-754 32-bit Single-Precision Floating-Point Numbers

- Representing Zero
  - Normalized form has a serious problem,
    - with an implicit leading 1 for the fraction, it cannot represent the number zero
    - De-normalized form was devised to represent zero and other numbers.
  - For E=0, the numbers are in the de-normalized form.
    - An implicit leading 0 (instead of 1) is used for the fraction; and the actual exponent is always -126.
    - Hence, the number zero can be represented with E=0 and F=0 (because $0.0\times2^{-126}=0$).
  - can also represent very small positive and negative numbers in de-normalized form with E=0.
    - For example, if S=1, E=0, and F=011 0000 0000 0000 0000 0000.
      - The actual fraction is $0.011=1\times2^{-2}+1\times2^{-3}=0.375$.
      - Since S=1, it is a negative number.
      - With E=0, the actual exponent is -126.
      - Hence the number is $-0.375\times2^{-126}=-4.4\times10^{-39}$, which is an extremely small negative number (close to zero).

IEEE-754 32-bit Single-Precision Floating-Point Numbers

- For $1 \le E \le 254$,
  - N = (-1)^S × 1.F × 2^(E-127).
  - These numbers are in the so-called *normalized* form.
  - The sign-bit represents the sign of the number.
  - Fractional part (1.F) are normalized with an implicit leading 1.
  - The exponent is bias (or in excess) of 127, so as to represent both positive and negative exponent.
  - The range of exponent is -126 to +127.
- For E = 0,
  - N = (-1)^S × 0.F × 2^(-126).
  - These numbers are in the so-called *denormalized* form.
  - The exponent of 2^-126 evaluates to a very small number.
  - Denormalized form is needed to represent zero (with F=0 and E=0).
  - It can also represents very small positive and negative number close to zero.
- For E = 255,
  - it represents special values, such as ±INF (positive and negative infinity) and NaN (not a number)

IEEE-754 32-bit Single-Precision Floating-Point Numbers

- Represent –0.75
  - $-0.75 = (-1)1 \times 1.1_2 \times 2^{-1}$
  - S = 1
  - Fraction = $1000\ldots00_2$
  - Exponent = –1 + Bias
    - Single: $-1 + 127 = 126 = 01111110_2$
    - Double: $-1 + 1023 = 1022 = 01111111110_2$
  - Single: 1 01111110 1000…00
  - Double: 1 01111111110 1000…00

$(0.75)_{10} = (0.11)_2$