Fixed-Point and Floating-Point Representation

Computer System Architecture

By

M. Morris Mano

Fixed-Point Representation

- Computers must represent everything with 1’s and 0’s, including the sign of a number and fixed/floating point number
- Number may have a Binary/Decimal Point
  - Position of point is needed to represent ***fractions, integers,* or *mixed integer-fraction number***
    - **Ex: 32.25**

      1) 0.25 2) 32.0 3) 32.25
  - Two ways of specifying the position of the binary point in a register by giving it a
    1. Fixed Point
    2. Floating Point

Fixed-Point Representation cont..

- **Fixed Point :**
  - The binary point is always fixed in one position
  - 2 positions mostly used
    1. A binary point in the ***extreme left*** of the register to make a store no. a fraction.(***Fraction*** : 0.xxxxx)
    2. A binary point in the ***extreme right*** of the register (***Integer*** : xxxxx.0)
       - *Binary point is not actually present, but the number stored in the register is treated as a fraction or as an integer*
- **Floating Point :**
  - Second register is used to designate the position of the binary point in the first register.

Fixed Point Representation

Binary Fixed-Point Representation

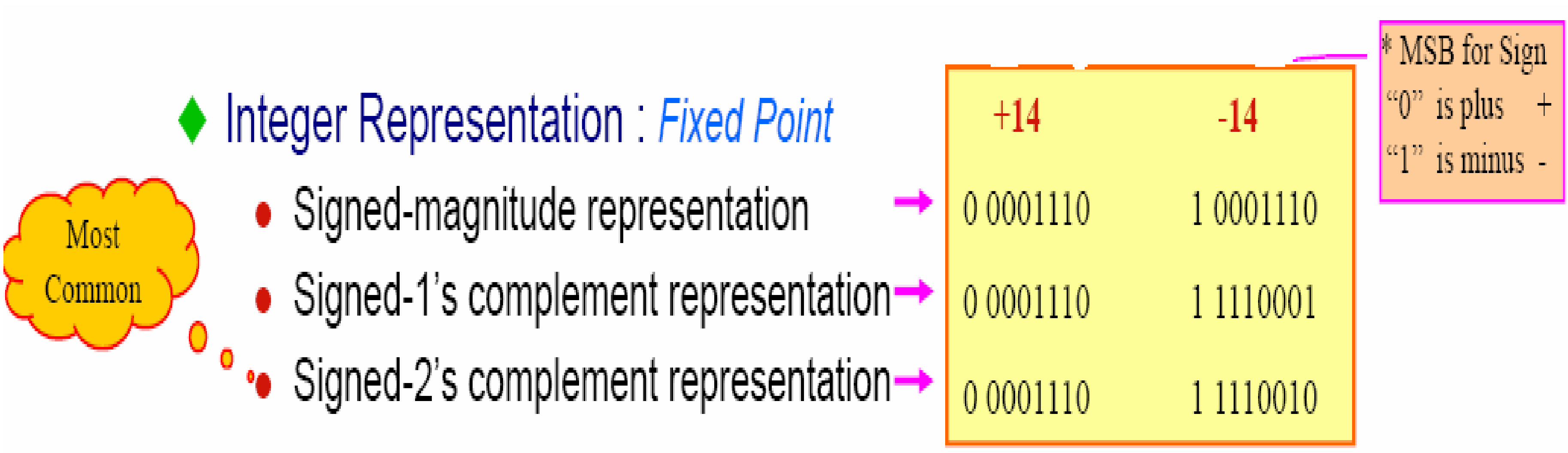
$X = x_n\, x_{n-1}\, x_{n-2} \ldots x_1\, x_0 \,.\, x_{-1}\, x_{-2} \ldots x_{-m}$

Sign Bit (x_n): 0 for positive, 1 for negative

Remaining Bits($x_{n-1}x_{n-2} \ldots x_1x_0.\ x_{-1}x_{-2} \ldots x_{-m}$)

Signed Numbers

- Need to be able to represent both *positive* and *negative* numbers
  - Following 3 representations are used.



Signed Numbers cont..

- **Example:**
- Represent +9 and -9 in 7 bit-binary number
- Only one way to represent +9 ==> 0 001001
- Three different ways to represent -9:
- In signed-magnitude : 1 001001
- In signed-1's complement : 1 110110
- In signed-2's complement : 1 110111
- In general, in computers, fixed point numbers are represented either integer part only or fractional part only.

Pros and cons of integer representation

- **Signed magnitude representation:**
  - 2 representations for 0
  - Simple
  - 255 different numbers can be represented.
  - Need to consider both sign and magnitude in arithmetic
  - Different logic for addition and subtraction
- **1’s complement representation:**
  - 2 representations for 0
  - Complexity in performing addition and subtraction
  - 255 different numbers can be represented.
- **2’s complement representation:**
  - Only one representation for 0
  - 256 different numbers can be represented.
  - Arithmetic works easily
  - Negating is fairly easy

Arithmetic Addition

- Addition Rules of Ordinary Arithmetic
  - The signs are ***same*** :
    - sign= ***common*** sign,
    - result= ***add***
    - Ex:

      (-12) + (-13) = -25

      (+12) + (+13) = +25
  - The signs are ***different*** :
    - sign= ***larger*** sign,
    - result= ***larger-smaller***
    - Ex:

    (+25) + (-37)= 37 - 25 = -12

Arithmetic Addition cont…

- Addition Rules of the signed 2's complement
  - Add the two numbers including their sign bits
  - Discard any carry out of the sign bit position

***Addition Exam)**

+ 6	00000110	- 6	11111010
+ 13	00001101	+ 13	00001101
+ 19	00010011	+ 7	00000111
+ 6	00000110	- 6	11111010
- 13	11110011	- 13	11110011
- 7	11111001	- 19	11101101

Arithmetic Subtraction

- Subtraction is changed to an Addition

  $(\pm A) - (+B) = (\pm A) + (-B)$

  $(\pm A) - (-B) = (\pm A) + (+B)$

- **Example:**

  (- 6) - (- 13) = +7

  1111010 - 11110011 = 11111010 + 2's comp of 11110011

  = 11111010 + 00001101

  = 1 00000111

  = +7

  – Discard End Carry

Overflow

- Two numbers of n digits each are added and the sum occupies n+1 digits.
- It is not a problem in manual calculation.
- It is problem in digital computers because width of the register is finite.
- n+1 bit cannot be accommodated in a register with a standard length of n bits
- Many computer detect the occurrence of an overflow by a corresponding Flip-Flop (It is set).
- It cannot occur if one number is +ve and other is –ve.

Overflow cont…

- An overflow may occur if the two numbers added are both positive or both negative
  - When two unsigned numbers are added
    - an overflow is detected from the end carry out of the MSB position
  - When two signed numbers are added
    - MSB always represents the sign
    - Negative numbers are in 2's complement form.
      - *sign bit is treated as part of the number*
      - *end carry does not indicate an overflow*

Overflow Detection

- Detected by observing the ***carry into*** the sign bit position and the ***carry out*** of the sign bit position
- If these two carries are not equal, an overflow condition is produced (***Exclusive-OR gate = 1***)

Overflow Detection

- +70 and +80 – stored in two 8-bit register.
- Each register can accommodate a binary number range from +127 to -128.
- Sum= +150, it exceeds the capacity of the register.
- If carry out is taken as sign bit of result then 9-bit resul
- Answ , we say tl

* Overflow Exam)

	out in		out in
carries	0 1	carries	1 0
+ 70	0 1000110	- 70	1 0111010
+ 80	0 1010000	- 80	1 0110000
+ 150	1 0010110	- 150	0 1101010

Decimal Fixed-Point Representation

- A 4 bit decimal code requires four F/Fs for each decimal digit
- The representation of 4385 in BCD requires 16 F/Fs (0100 0011 1000 0101)
- The representation in decimal is *wasting a considerable amount of storage* space and the circuits required to perform decimal arithmetic are *more complex*

***Decimal Exam) (+375) + (-240)**

375 + (10's comp of 240)= 375 + 760

0 375	(0000 0011 0111 0101)
+9 760	(1001 0111 0110 0000)
0 135	(0000 0001 0011 0101)