Computer Organization

COMP2120

Qi Zhao

January 21, 2024

Number Representation Part I

Number system in computer

- We understand decimal number, base-10
- Computer: Binary, not decimal
  Easy to be represented, stored, and transmitted.
  E.g. hole in a punched card, high or low voltage
- There are 10 types of people in this world, those who understand binary and those who don't
- NOT the same as in mathematics.
  E.g., the product of two positive integers may be negtive
  Associative law

$$(3.14 + 1e20) - 1e20 \neq 3.14 + (1e20 - 1e20)$$

Number system in computer

Finite Precision Numbers

- the size of memory available for storing a number is fixed. e.g. 32-bit integers
- The number of possible representation is fixed, i.e. 2^{32} for 32-bit
- Overflow (number too large to be represented) and Underflow (number too small to be represented)
- Negative numbers
- Fractions
- Irrational Number*
- Complex Numbers*

Radix Number Systems

- Positional number system: a string of digits

$$(\dots a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3} \dots)_r$$

- position i, an associated weight r^i, r is radix, or base
- $0 \leq a_i < r$. Decimal: $r = 10$, $a_i \in \{0, 1, \dots, 9\}$; Binary: $r = 2$, $a_i \in \{0, 1\}$
- Dot: radix point
- Value:

$$\begin{aligned} &\dots + a_3 r^3 + a_2 r^2 + a_1 r + a_0 r^0 + a_{-1} r^{-1} + a_{-2} r^{-2} + a_{-3} r^{-3} + \dots \\ &= \sum_i (a_i \times r^i) \end{aligned}$$

Radix Number Systems

- The formula applies to all radices, binary (base 2), octal (base 8), hexadecimal (base 16)
- Hexadecimal: $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15)$
- $0\mathrm{x}$ or $0\mathrm{X}$ also mean Hexadecimal
  $\mathrm{FA1D3}_{16} = 0\mathrm{xFA1D3}$
- What is 10 (hexadecimal) ?
- The conversion between octal/hexadecimal and binary is very easy.
- Octal and Hexadecimal representation is to facilitate human only. Computer still use binary representation.

Radix Number Systems

Example:

$$1984_{10} = 1 \times 10^3 + 9 \times 10^2 + 8 \times 10 + 4$$
$$1B84_{16} = 1 \times 16^3 + 11 \times 16^2 + 8 \times 16 + 4$$

Note that the principle behind decimal representation and other radix representation is the same.
We may use decimal numbers as analogy for explanation.

Conversion between different radices-to radix 10

- $$\begin{aligned}(\dots a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3} \dots)_r \\ = \dots + a_3 r^3 + a_2 r^2 + a_1 r + a_0 r^0 + a_{-1} r^{-1} + a_{-2} r^{-2} + a_{-3} r^{-3} + \dots \\ = \sum_i (a_i \times r^i)\end{aligned}$$

- Radix 2

$$10010011_2 = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 147$$

- Radix 8

$$223_8 = 2 \times 8^2 + 2 \times 8^1 + 3 \times 2^0 = 147$$

- Radix 16

$$93_{16} = 9 \times 16^1 + 3 \times 16^0 = 147$$

Conversion between different radices-to radix 10

- Fractional numbers $1A6.BE_{16}$

$$\begin{aligned} 1A6.BE_{16} &= 1 \times 16^2 + 10 \times 16^1 + 6 \times 16^0 + 11 \times 16^{-1} + 14 \times 16^{-2} \\ &= 422.7421875 \end{aligned}$$

- How many operations ? $D = \sum_{i=0}^{n} (a_i \times r^i)$

$$Oper \leq n + \sum_{i=0}^{n} i = \mathcal{O}(n^2)$$

- A faster way

$$\begin{aligned} D = \sum_{i=0}^{n} a_i \times r^i &= r \left(\sum_{i=1}^{n} a_i \times r^{i-1} \right) + a_0 \\ &= r \left(r \left(\sum_{i=2}^{n} a_i \times r^{i-2} \right) + a_1 \right) + a_0 = \dots \end{aligned}$$

Conversion between different radices-to radix 10

- Example

$$\begin{aligned} 1A6_{16} &= 16 \times (16 \times 1 + 10) + 6 \\ &= 16 \times (16 + 10) + 6 = 422_{10}. \end{aligned}$$

- $Oper = \mathcal{O}(n)$

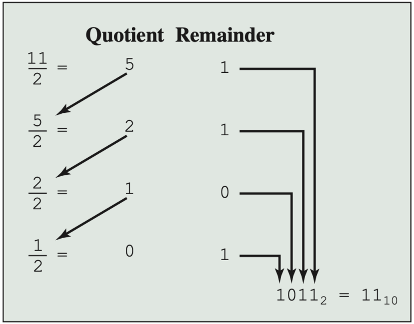
Conversion between different radices-From radix 10 to radix 2

- Example:

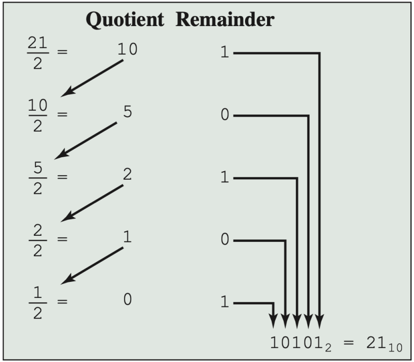
$$
\begin{aligned}
1A6_{16} &= 16 \times (16 \times 1 + 10) + 6 \\
&= 16 \times (16 + 10) + 6 = 422_{10}.
\end{aligned}
$$

- Find $a_0, a_1, \cdots$ by repeatedly divided by r and find the quotient and remainder.
- $\frac{422}{16}$, Quotient: 26, Remainder: 6. $a_0 = 6$.
- Divide quotient by r,
- $\frac{26}{16}$, Quotient: 1, Remainder: 10. $a_1 = A$, $a_2 = 1$.
- Fina all digits by repeatedly dividing the quotient by r.

Conversion between different radices-From radix 10 to radix 2



(a) 11_{10}

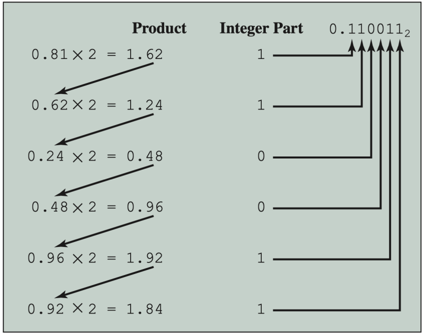


(b) 21_{10}

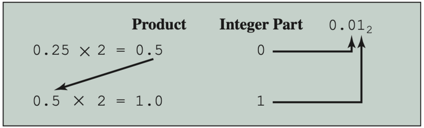
Figure 9.1 Examples of Converting from Decimal Notation to Binary Notation for Integers

Conversion between different radices-From radix 10 to radix 2

- Fractional numbers: multiply by r and get the integer part and repeat this process



(a) $0.81_{10} = 0.110011_2$ (approximately)



(b) $0.25_{10} = 0.01_2$ (exactly)

Figure 9.2 Examples of Converting from Decimal Notation to Binary Notation for Fractions

Example

Convert 19.7421875 to octal:

- Integer part: $\frac{19}{8} = 2 + \frac{3}{8}$, $19 = 23_8$.
- Fractional part:
  - $0.7421875 \times 8 = 5.9375$, First digit $a_{-1} = 5$
  - $0.9375 \times 8 = 7.5$, Second digit $a_{-2} = 7$
  - $0.5 \times 8 = 4$, Third digit $a_{-3} = 4$

  Thus, $0.7421875 = 0.574_8$
- $19.7421875 = 23.574_8$

Between radix 2, 8 and 16

- there exist convenient ways to convert between radix 2 and octal (hexadecimal).
- Binary digits are grouped into sets of four bits.

$0000 = 0$	$0100 = 4$	$1000 = 8$	$1100 = C$
$0001 = 1$	$0101 = 5$	$1001 = 9$	$1101 = D$
$0010 = 2$	$0110 = 6$	$1010 = A$	$1110 = E$
$0011 = 3$	$0111 = 7$	$1011 = B$	$1111 = F$

- Starting from the decimal point, group 4 binary digits to the left (to the right, after decimal point), and convert the 4 digits to hex digit
- Example

$$0001\ 1001\ 1000\ 0100\ .\ 1011\ 0110 = 1984.B6_{16}$$

- Similar to octal

$$001\ 100\ 110\ 000\ 100\ .\ 101\ 101\ 100 = 14604.554_8$$

To radix 2

- Example:

$$7BA3.BC5_{16} = 0111\ 1011\ 1010\ 0011.\ 1011\ 1100\ 0101$$
$$56743.5704_8 = 101\ 110\ 111\ 100\ 011.\ 101\ 111\ 000\ 100$$

$0000 = 0$	$0100 = 4$	$1000 = 8$	$1100 = C$
$0001 = 1$	$0101 = 5$	$1001 = 9$	$1101 = D$
$0010 = 2$	$0110 = 6$	$1010 = A$	$1110 = E$
$0011 = 3$	$0111 = 7$	$1011 = B$	$1111 = F$

Sign and Magnitude Representation

- Radix number system: unsigned encoding, representing positive numbers.
- How to deal with negative numbers
- One solution: represent the sign using the leftmost bit, and the rest of the bits for magnitude.
- Example, 1-bit sign, 7-bit magnitude

$$
\begin{aligned}
+17 &= 00010001 \\
-17 &= 10010001
\end{aligned}
$$

- How to represent 0, we have two zeros 00000000 or 10000000 ?
- Difficult to do arithmetic.

Excess K

- Also referred to as offset binary, excess code or biased representation
- n-bit Excess K notation: range from $0 - K$ to $2^n - 1 - K$.
- 3 bit is from 0 to 7. Middle is 3 or 4

	$K = 4$	$K = 3$
111	3	4
110	2	3
101	1	2
100	0	1
011	-1	0
010	-2	-1
001	-3	-2
000	-4	-3

- Excess-2^{n-1}, Excess-$2^{n-1} - 1$, usually the second one
- The lower half is negative, the upper half positive.
- useful in floating point exponent represenation

Representation of Signed Numbers (Positive and Negative)

- The representation scheme is a mapping between the bit pattern (unsigned integer value) and the actual value it represents,

$$f(\mathit{bit\ pattern}) = \mathit{actual\ value}$$

- The bit pattern can be treated as a positive number. The mapping is from a positive number to a number (data) to be represented.
- E.g., excess K,

$$f(\mathit{bit\ pattern}) = \mathit{value} = \mathit{bit\ pattern} - K$$

One's complement

- Positive numbers: same as an unsigned bit pattern
- Negative numbers: representing $-N$ by inverting all bits of binary representation of N
- inverting a bit x is $1 - x$

$$
\begin{aligned}
7 &= 0111(4-bit) \\
-7 &= inverting(0111) \\
&= 1000
\end{aligned}
$$

$$
\begin{aligned}
7 &= 0000\,0111(8-bit) \\
-7 &= inverting(0000\,0111) \\
&= 1111\,1000
\end{aligned}
$$

- What is mapping $f(bit\ pattern)$

One's complement

- The bit pattern of -7 is

$$1111\ 1111 - 0000\ 0111 = 1111\ 1000$$
$$1111\ 1111 - N = (2^n - 1) - N$$

- bit pattern value $= (2^n - 1) - N$, represented value $= -N$

$$f(bit\ pattern) = bit\ pattern - (2^n - 1)$$
$$bit\ pattern = 2^n - 1 - N$$

- Negative number: Most significant bit (MSB, the leftmost bit)=1
- What about positive numbers?

$$f(bit\ pattern) = bit\ pattern$$

One's complement

- What $1101\,1010$ represents?

$$f = 1101\,1010 - (2^8 - 1) = 2^7 + 2^6 + 2^4 + 2^3 + 2 - (2^8 - 1) = -37$$

- How to represent 0? Two zeros, $000\ldots, 0$, $111\ldots, 1$
- Problem with arithmetic: cannot just add them together as if they are simple binary numbers,

Two's complement

- Positive numbers, same as an unsigned bit pattern
- Negative numbers, representing by One's complement $+1$,

$$-7(8\ bit) = 1111\ 1000\ One's$$
$$-7(8\ bit) = 1111\ 1001\ Two's$$

- the bit pattern of $-N$ has a value of $2^n - N$.
- Negative number: Most significant bit (MSB, the leftmost bit)=1
- Only one zero, $000\ldots0$.
- Simple arithmetic

Example: (8-bit numbers)

Scheme	Bit pattern of -7	Bit pattern of 9	$9 + (-7) = 2$
One's complement	1111 1000	0000 1001	0000 0010
Two's complement	1111 1001	0000 1001	0000 0010
Sign and magnitude	1000 0111	0000 1001	0000 0010
Excess 127($2^{8-1} - 1$)	0111 1000	1000 1000	1000 0001

- Two's complement: just add them together as if they are simple binary numbers, and discard any carry from the MSB, $9 + (-7)$

$$9 + 2^n - 7 = 2^n(\text{carry out from MSB}) + 2$$

-

$$\begin{array}{rr} & 1111\ 1001 \\ + & 0000\ 1001 \\ = & \mathbf{1}\,0000\ 0010 \end{array}$$

Range Extension

- Take an n-bit integer and store it in m bits, $m > n$
- Example: Sign magnitude, just move the sign bit and add zeros

$$
\begin{aligned}
+18 &= 00010010 && \text{(sign magnitude, 8 bits)} \\
+18 &= 0000000000010010 && \text{(sign magnitude, 16 bits)} \\
-18 &= 10010010 && \text{(sign magnitude, 8 bits)} \\
-18 &= 1000000000010010 && \text{(sign magnitude, 16 bits)}
\end{aligned}
$$

- Two's complement: adding zeros does not work

$$
\begin{aligned}
+18 &= 00010010 && \text{(twos complement, 8 bits)} \\
+18 &= 0000000000010010 && \text{(twos complement, 16 bits)} \\
-18 &= 11101110 && \text{(twos complement, 8 bits)} \\
-32{,}658 &= 1000000001101110 && \text{(twos complement, 16 bits)}
\end{aligned}
$$

Range Extension

- Move the sign bit to the new leftmost position and fill with copies of the sign bit

$$
\begin{aligned}
-18 &= 11101110 \quad \text{(twos complement, 8 bits)} \\
-18 &= 1111111111101110 \quad \text{(twos complement, 16 bits)}
\end{aligned}
$$

- Two's complement: the value of n bit representation of $-|A|$ is $2^n - |A|$
- After extension, the value of m bit representation is

$$2^n - |A| + 2^{m-1} + 2^{m-2} + \cdots + 2^n = 2^n - |A| + 2^m - 2^n = 2^m - |A|$$

- After range extension, the value represented is still $|A|$.

Two's complement

Table 10.1 Characteristics of Twos Complement Representation and Arithmetic

Range	-2^{n-1} through $2^{n-1} - 1$
Number of Representations of Zero	One
Negation	Take the Boolean complement of each bit of the corresponding positive number, then add 1 to the resulting bit pattern viewed as an unsigned integer.
Expansion of Bit Length	Add additional bit positions to the left and fill in with the value of the original sign bit.
Overflow Rule	If two numbers with the same sign (both positive or both negative) are added, then overflow occurs if and only if the result has the opposite sign.
Subtraction Rule	To subtract B from A, take the twos complement of B and add it to A.