Digital Representation of Information

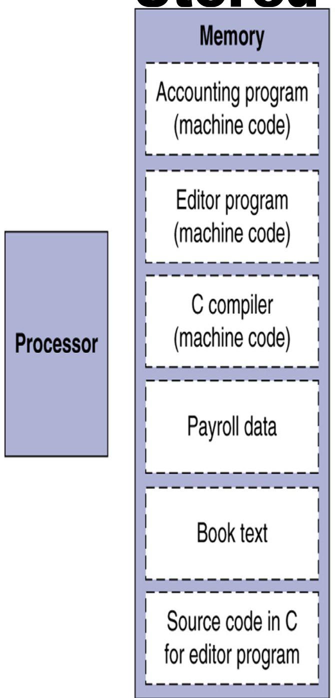
VOLKAN KURSUN

Some material from McGraw Hill

Outline

- Prelude: Digital Representation of Information
- Positional Number Systems
  - Decimal Number System
  - Binary Number System
  - Octal Number System
  - Hexadecimal Number System
  - General base-r system

Stored Program Computers



- Modern computers are built on two key principles (stored program concept):
  1) Instructions and data are represented as binary numbers
  2) These binary numbers are stored in memory to be read or written as needed by a program
- Stored programs allow a computer to perform various tasks:
  To switch from one task to another simply load the memory with the corresponding program and data and tell the computer to begin executing at a specific location in memory
- Representing instructions and data with binary numbers and letting them share the same memory **simplifies both the memory hardware and the software**: same memory technology used for both data and instructions

Digital Representation of Information

- Information is represented in logic circuits as electronic signals
- Different signal levels represent different digits of information
- To make the design of logic circuits easier (more compatible with how transistors operate), each digit is allowed to take only two possible values, denoted as 0 and 1
- These logic levels are implemented as voltage levels in a circuit:
  - 0 is typically represented with 0V (ground)
  - 1 is typically represented with the power supply voltage (V_{DD})
- All information in logic circuits is represented as combinations of 0s and 1s (binary digits)

Data Representation in Computers

- Everything in computer is a string of bits.
  - Represent 1000_{10} as 1111101111_2.
  - Represent 1.708984375_{10} as 1.101101011_2.
  - Represent $9.5*10^2$ as 1.11011011_2*2^9.
- Issues of binary number representation:
  - How to represent –ve number,
  - How to represent fractional and real number
  - How to handle number that go beyond the representation range.

Outline

- Prelude: Digital Representation of Information
- Positional Number Systems
  - Decimal Number System
  - Binary Number System
  - Octal Number System
  - Hexadecimal Number System
  - General base-r system

Number Systems

- Human beings typically use **decimal** (base 10) numbers for counting
  - Natural choice: **we have 10 fingers**
- Computers use **binary** (base 2) number system
  - Natural choice: **transistors have binary state: on or off**, voltage high or low
- In computing, **hexadecimal** (base 16) and octal (base 8) number systems are also commonly used for compact representation of binary numbers with many digits (bits)

Decimal Numbers (Base 10)

- Decimal number system has 10 symbols that are called digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9

A number written without any prefix or suffix is interpreted as decimal. Sometimes a suffix of **D** may be used to represent decimal numbers explicitly

Positional number system: every digit, depending on its position, has a corresponding weight and value

$(a_{n-1}a_{n-2}a_{n-3}\ldots a_1a_0)_{10} = a_{n-1}*10^{n-1} + a_{n-2}*10^{n-2} + a_{n-3}*10^{n-3} + \ldots + a_1*10^1 + a_0*10^0$

- Example: 2019 = 2019**D** = $(2019)_{10}$

  $2019 = 2*10^3 + 0*10^2 + 1*10^1 + 9*10^0$

Decimal Numbers with Fractions

$D = 342.12 = 3\times100 + 4\times10 + 2\times1 + 1\times0.1 + 2\times0.01$

$D = d_2 d_1 d_0 . d_{-1} d_{-2}$ $\quad d_i \in \underbrace{\{0, 1, \ldots, 9\}}_{\text{Arabic numerals}}$ (digit)

radix point

base (radix): 10

Positional number system

$V(D)$ = value represented by D

$= d_2\times10^2 + d_1\times10^1 + d_0\times10^0 + d_{-1}\times10^{-1} + d_{-2}\times10^{-2}$

* power of the digit depends on the position

Outline

- Prelude: Digital Representation of Information
- Positional Number Systems
  - Decimal Number System
  - Binary Number System
  - Octal Number System
  - Hexadecimal Number System
  - General base-r system

<u>Unsigned</u> Number Representation

- In computers, unsigned number refers to non-negative integers
  - Such as 0, 1, 2, 3 …
- Example: binary number 0b 101 = 5 (base 10)

Unsigned Binary Numbers (Base 2)

- Computers use **<u>binary</u>** (base 2) number system
  - Natural choice: **<u>transistor on or off</u>**, voltage high or low, binary state
- Binary number system has 2 symbols that are called binary digits (bits): 0 and 1

A binary number is identified with either a prefix of **0b** or **0B** or a prefix of **b'bits inside apostrophe'** or a suffix of **B** or **b**

$(a_{n-1}a_{n-2}a_{n-3}\ldots a_1a_0)_2 = a_{n-1}*2^{n-1} + a_{n-2}*2^{n-2} + a_{n-3}*2^{n-3} + \ldots + a_1*2^1 + a_0*2^0$

- Example: **0b**101011 = **b'**101011**'** = 101011**B** = $(101011)_2 = 1*2^5 + 0*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 = 43$

Unsigned Binary ↔ Decimal Conversion

Unsigned Binary → Decimal

0b1001010 = $?_{ten}$

Binary Digit	Decimal Value
0	$0 \times 2^0 = 0$
1	$1 \times 2^1 = 2$
0	$0 \times 2^2 = 0$
1	$1 \times 2^3 = 8$
0	$0 \times 2^4 = 0$
0	$0 \times 2^5 = 0$
1	$1 \times 2^6 = 64$
	$\Sigma = 74_{ten}$

Decimal → Binary

74 = 0b?

Decimal	Binary Digit
74	
74/2 = 37	0
37/2 = 18	1
18/2 = 9	0
9/2 = 4	1
4/2 = 2	0
2/2 = 1	0
1/2 = 0	1
Collect → remainder bits in reverse order	0b1001010

Unsigned Binary ↔ Decimal Conversion

Unsigned Binary → Decimal

0b11011 = $?_{ten}$

Binary Digit	Decimal Value
1	$1 \times 2^0 = 1$
1	$1 \times 2^1 = 2$
0	$0 \times 2^2 = 0$
1	$1 \times 2^3 = 8$
1	$1 \times 2^4 = 16$
	$\Sigma = 27$

- Same symbols can mean different things
- Be explicit unless you mean decimal

Decimal → Binary

11011 = 0b?

Decimal	Binary
11011	
/2 = 5505	1
/2 = 2752	1
/2 = 1376	0
/2 = 688	0
/2 = 344	0
/2 = 172	0
/2 = 86	0
/2 = 43	0
/2 = 21	1
/2 = 10	1
/2 = 5	0
/2 = 2	1
/2 = 1	0
/2 = 0	1
Collect →	0b10101100000011

Unsigned Binary Integers

Decimal equivalent of an n-bit unsigned binary number:

$$x = x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \cdots + x_1 2^1 + x_0 2^0$$

Range: 0 to $+2^n - 1$

- 32-Bit Binary Number Example:

0b 0000 0000 0000 0000 0000 0000 0000 1011
$= 0 + \ldots + 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0$
$= 0 + \ldots + 8 + 0 + 2 + 1 = (11)_{10}$

Range with 32 bits: 0 to $+2^{32} - 1$

0 to +4,294,967,295

Unsigned Binary Integers

Decimal equivalent of an n-bit unsigned binary number:

$$x = x_{n-1}2^{n-1} + x_{n-2}2^{n-2} + \cdots + x_1 2^1 + x_0 2^0$$

Range: 0 to $+2^n - 1$

Range with 32 bits: 0 to $+2^{32} - 1$

0 to +4,294,967,295

Range with 64 bits: 0 to $+2^{64} - 1$

0 to +18,446,744,073,709,551,615

18 quintillion 446 quadrillion 744 trillion 73 billion 709 million 551 thousand 615

Binary with Fraction to Decimal

$d_i \in \{0,1\}$ (base 2)

Example: $D = 10011_2$ ← least significant bit (LSB)

most significant bit (MSB)

$V(D) = 1\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 = 19$

$= 1\times10^1 + 9\times10^0 = 19_{10}$

Decimal equivalent of an unsigned binary number with fraction:

Exercise: 1010.011_2 to base 10 (decimal)

Answer:

$1\times2^3 + 1\times2^1 + 1\times2^{-2} + 1\times2^{-3} = 8 + 2 + 0.25 + 0.125$

$= 10.375_{10}$

Decimal Fraction to Base-2 Conversion

Successively multiply the fractions with 2 and record and collect the integer parts **until the fraction becomes zero**

Convert 0.6875_{10} to binary

d_{-1}: $0.6875 \times 2 = 1.3750$

d_{-2}: $0.375 \times 2 = 0.75$

d_{-3}: $0.75 \times 2 = 1.5$

d_{-4}: $0.5 \times 2 = 1.0$

$0.6875_{10} = 0.1011_2$

Decimal Fraction to Base-2 Conversion

There is **no guarantee** that the fraction would become zero after a certain number of steps: **repeating binary sequences may be encountered in the fraction**

- Successively multiply the fraction by 2 **until either the fraction becomes 0 or the binary sequence starts to repeat**
- Example: Convert 3.47 to binary

0.47	x 2	0.94	0.32	x 2	0.64	0.92	x 2	1.84
0.94	x 2	1.88	0.64	x 2	1.28	0.84	x 2	1.68
0.88	x 2	1.76	0.28	x 2	0.56	0.68	x 2	1.36
0.76	x 2	1.52	0.56	x 2	1.12	0.36	x 2	0.72
0.52	x 2	1.04	0.12	x 2	0.24	0.72	x 2	1.44
0.04	x 2	0.08	0.24	x 2	0.48	0.44	x 2	0.88
0.08	x 2	0.16	0.48	x 2	0.96	0.88	x 2	1.76
0.16	x 2	0.32	0.96	x 2	1.92	0.76	x 2	1.52

Repeating 20-digit binary sequence

Decimal Fraction to Base-2 Conversion

- Successively multiply the fraction by 2 **until either the fraction becomes 0 or the binary sequence starts to repeat**
- Example: Convert 3.47 to binary

0.47	x 2	0.94	0.32	x 2	0.64	0.92	x 2	1.84
0.94	x 2	1.88	0.64	x 2	1.28	0.84	x 2	1.68
0.88	x 2	1.76	0.28	x 2	0.56	0.68	x 2	1.36
0.76	x 2	1.52	0.56	x 2	1.12	0.36	x 2	0.72
0.52	x 2	1.04	0.12	x 2	0.24	0.72	x 2	1.44
0.04	x 2	0.08	0.24	x 2	0.48	0.44	x 2	0.88
0.08	x 2	0.16	0.48	x 2	0.96	0.88	x 2	1.76
0.16	x 2	0.32	0.96	x 2	1.92	0.76	x 2	1.52

Repeating 20-digit binary sequence

- 3.47 = 0b11.01$\underbrace{11100001010001111010}_{\text{Repeating 20-digit binary sequence}}$…

Outline

- Prelude: Digital Representation of Information
- Positional Number Systems
  - Decimal Number System
  - Binary Number System
  - Octal Number System
  - Hexadecimal Number System
  - General base-r system

Octal Numbers

$d_i \in \{0,1,2,3,4,5,6,7\}$ (base 8)

Use **0o** or **0O** prefixes or **Q** or **q** suffixes to distinguish octal numbers

* Each octal digit is equivalent to a 3-bit binary string

octal digit	equivalent 3-bit binary string
0	000
1	001
2	010
3	011
⋮	⋮
7	111

Used by native Americans. Count spaces between fingers.

Binary to Octal Conversion

$$\underbrace{010}\,\underbrace{001}\,\underbrace{100}\,\underbrace{111}_2 = 2147_8$$

$d_3 = 2$ $d_2 = 1$ $d_1 = 4$ $d_0 = 7$

Octal equivalent of an unsigned binary number with fraction:

$$\underbrace{010}.\underbrace{101}\,\underbrace{100}\,\underbrace{101}\,\underbrace{1_2\,00} = 2.5454_8$$

$d_0 = 2$ $d_{-1} = 5$ $d_{-2} = 4$ $d_{-3} = 5$ $d_{-4} = 4$

Octal to Binary Conversion

Binary equivalent of an octal number with fraction:

$$542.7_8 = 101100010.111_2$$

Octal with Fraction to Decimal

Decimal equivalent of an octal number with fraction:

$$436.5_8 = 4 \times 8^2 + 3 \times 8^1 + 6 \times 8^0 + 5 \times 8^{-1} = 286.625_{10}$$

Outline

- Prelude: Digital Representation of Information
- Positional Number Systems
  - Decimal Number System
  - Binary Number System
  - Octal Number System
  - Hexadecimal Number System
  - General base-r system

Hexadecimal Numbers (Base 16)

- Hexadecimal number system has 16 symbols called hexadecimal digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Hexadecimal numbers are identified with either a prefix of **0x** or **0X** or a suffix of **H** or **h**
- **Compact representation** of bit strings
  - 4 bits per hexadecimal digit

Nibble: group of 4 bits

0	0000	4	0100	8	1000	c	1100
1	0001	5	0101	9	1001	d	1101
2	0010	6	0110	a	1010	e	1110
3	0011	7	0111	b	1011	f	1111

Hexadecimal Numbers (Base 16)

- **Compact representation** of bit strings
  - 4 bits per hexadecimal digit

0	0000	4	0100	8	1000	c	1100
1	0001	5	0101	9	1001	d	1101
2	0010	6	0110	a	1010	e	1110
3	0011	7	0111	b	1011	f	1111

Example: **0x**eca86420 = eca86420**H** = $(eca86420)_{16}$ in binary

Replace each hexadecimal digit with 4 equivalent bits:

0b1110 1100 1010 1000 0110 0100 0010 0000

Binary to Hexadecimal Conversion

$$\underbrace{0100}_{d_2=4}\ \underbrace{0110}_{d_1=6}\ \underbrace{0111}_{d_0=7}{}_2 = 467_{16}$$

Hexadecimal equivalent of an unsigned binary number with fraction:

$$\underbrace{0010}_{2}.\underbrace{1011}_{B}\ \underbrace{0010}_{2}\ \underbrace{11_2 00}_{C} = 2.B2C_{16}$$

Hexadecimal ↔ Decimal Conversion

Hexadecimal → Decimal

0x11011 = $?_{ten}$

Hexadecimal Digit	Decimal Value	
1	1×16^0	= 1
1	1×16^1	= 16
0	0×16^2	= 0
1	1×16^3	= 4096
1	1×16^4	= 65536
	Σ	= 69649

Decimal → Hexadecimal

11011 = 0x?

Decimal	Hexadecimal
11011	
/16 = 688	3
/16 = 43	0
/16 = 2	B
/16 = 0	2
Collect → remainders in reverse order	0x2B03

- Same symbols can mean different things
- **<u>Be explicit unless you mean decimal</u>**

Conversion Between Radix

Nibble: group of 4 bits

- Byte = 8-bit data
  - Binary 00000000_2 to 11111111_2
  - Decimal 0_{10} to 255_{10}
  - Octal 000_8 to 377_8 $(11,111,111_2)$
  - Hexadecimal 00_{16} to FF_{16}

Byte, halfword (2 bytes), word (4 bytes), and doubleword (8 bytes): data sizes used in computers (largest data size depends on the width of the processor datapath and registers). Data are processed in chunks called bytes, halfwords, words, and doublewords in a computer

32-bit words in 32-bit ISA (Example: RV32, ARM-32)

64-bit doublewords in 64-bit ISA (Example: RV64, ARM-64)

Hex	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Outline

- Prelude: Digital Representation of Information
- Positional Number Systems
  - Decimal Number System
  - Binary Number System
  - Octal Number System
  - Hexadecimal Number System
  - General base-r system

General Base-r System

$D = (d_{p-1} d_{p-2} \cdots d_1 d_0 . d_{-1} d_{-2} \cdots d_{-n})_r \leftarrow$ base/radix (integer)

most significant digit (MSD)

least significant digit (LSD)

Base-r point

$d_i \in \{0, 1, \ldots, r-1\}$

$$V(D) = \sum_{i=-n}^{p-1} d_i r^i$$

Note: even in numbers with fractions, the **least significant digit is the rightmost digit**

Decimal to Base-r Conversion

- Take decimal representation, keep dividing by r until quotient is zero
- Record the remainder at each step
- Last remainder is the MSD. First remainder is the LSD

Example : Convert 179_{10} to base 2

	quotient	remainder	
179/2	89	1	(LSB)
89/2	44	1	
44/2	22	0	
22/2	11	0	
11/2	5	1	
5/2	2	1	
2/2	1	0	
1/2	0	1	(MSB)

$179_{10} = 10110011_2$

Decimal to Base-2 Conversion Example-2

- Take decimal representation, keep dividing by r until quotient is zero
- Record the remainder at each step
- Last remainder is the MSD. First remainder is the LSD

Convert $(857)_{10}$ to binary

		Remainder	
857 ÷ 2	= 428	1	LSB
428 ÷ 2	= 214	0	
214 ÷ 2	= 107	0	
107 ÷ 2	= 53	1	
53 ÷ 2	= 26	1	
26 ÷ 2	= 13	0	
13 ÷ 2	= 6	1	
6 ÷ 2	= 3	0	
3 ÷ 2	= 1	1	
1 ÷ 2	= 0	1	MSB

Result is $(1101011001)_2$

Base-r to Base-k Conversion

- Convert base r to decimal
- Convert decimal to base k

231_4 to base 5

$231_4 = 2 \times 4^2 + 3 \times 4^1 + 1 \times 4^0 = 32 + 12 + 1 = 45_{10}$

	quotient	remainder	
45/5	9	0	(LSD)
9/5	1	4	
1/5	0	1	(MSD)

$231_4 = 140_5$