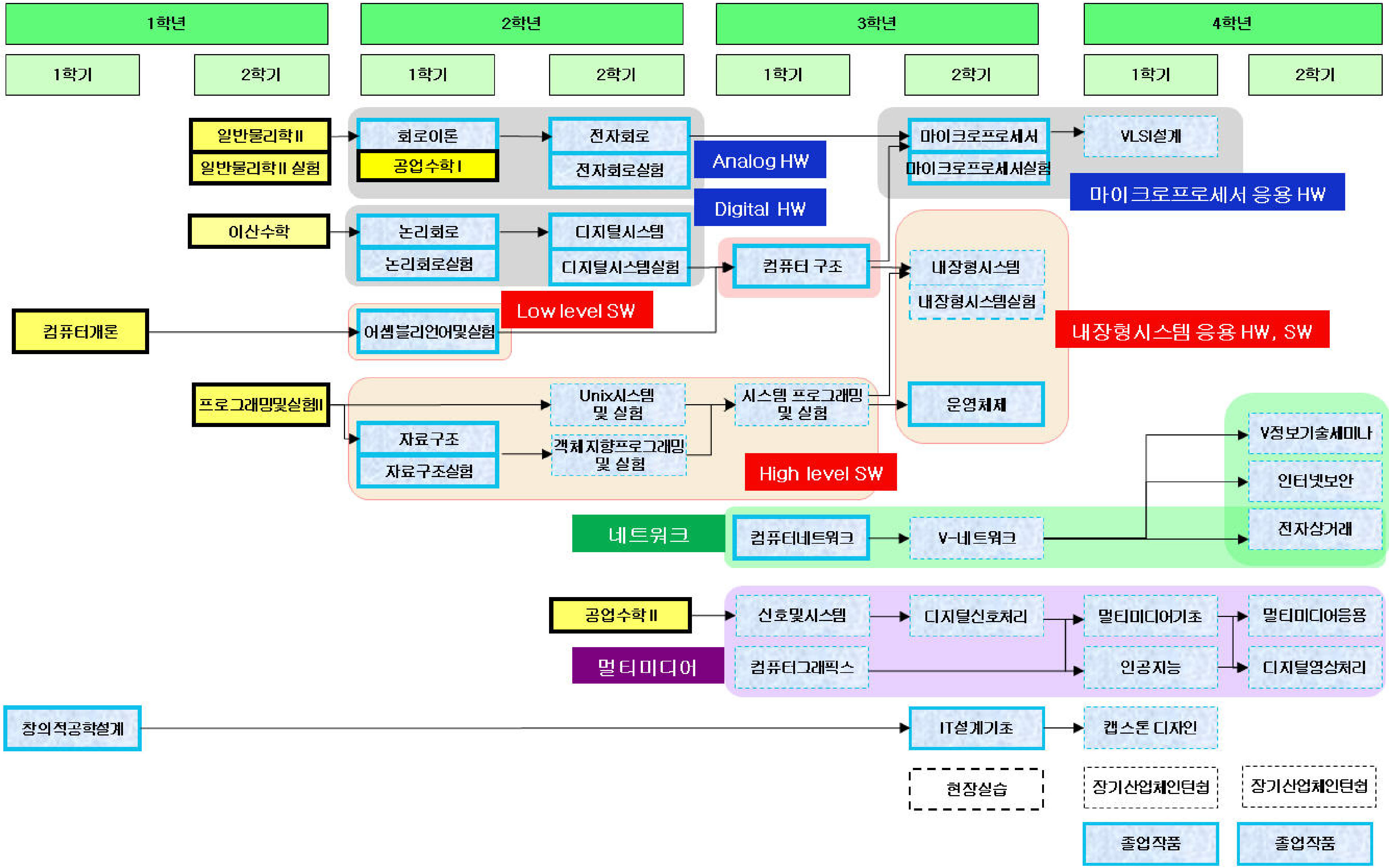
Introduction
Unit 1. Number Systems and Conversion

Spring 2015

School of Electrical Engineering

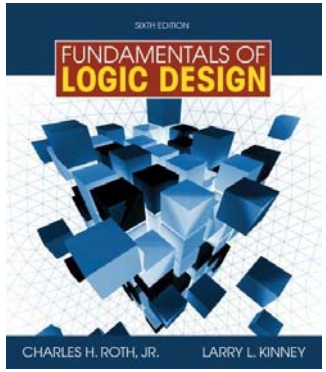
Prof. Jong-Myon Kim

이수체계도


1학년
2학년
3학년
4학년
1학기
2학기
1학기
2학기
1학기
2학기
1학기
2학기
일반물리학II
일반물리학II 실험
회로이론
공업수학I
전자회로
전자회로실험
Analog HW
마이크로프로세서
마이크로프로세서실험
VLSI설계
마이크로프로세서 응용 HW
Digital HW
이산수학
논리회로
논리회로실험
디지털시스템
디지털시스템실험
컴퓨터 구조
내장형시스템
내장형시스템실험
Low level SW
컴퓨터개론
어셈블리언어및실험
내장형시스템 응용 HW, SW
프로그래밍및실험II
Unix시스템
및 실험
시스템 프로그래밍
및 실험
운영체제
자료구조
자료구조실험
객체지향프로그래밍
및 실험
High level SW
IT정보기술세미나
인터넷보안
전자상거래
네트워크
컴퓨터네트워크
V-네트워크
공업수학II
신호및시스템
디지털신호처리
멀티미디어기초
멀티미디어응용
멀티미디어
컴퓨터그래픽스
인공지능
디지털영상처리
창의적공학설계
IT설계기초
캡스톤 디자인
현장실습
장기산업체인턴쉽
장기산업체인턴쉽
졸업작품
졸업작품

Course Information

- **Textbook**
  - Fundamentals of Logic Design (6^{th} Ed.), Charles H. Roth, Jr, Thomson Brooks/Cole.



- **Class Website**
  - http://uclass.ulsan.ac.kr/

Grading Policy

- 4 In-class Tests : 80%
- Term Project : 10%
  - Individual work, no collaboration
  - No late turn-in will ever be accepted
- Class Attend : 10%
  - 결석: -1, 지각: -0.5
- Final Grade is relative to your peer in class

Objectives

Topics introduced in this chapter:

- Difference between Analog and Digital System
- Difference between Combinational and Sequential Circuits
- Binary number and digital systems
- Number systems and Conversion
- Add, Subtract, Multiply, Divide Positive Binary Numbers
- 1’s Complement, 2’s Complement for Negative binary number
- BCD code, 6-3-1-1 code, excess-3 code

Digital Systems and Switching

Digital Systems

- ⇨ computation, data processing, control, communication, measurement
- ⇨ reliable, integration

Differences

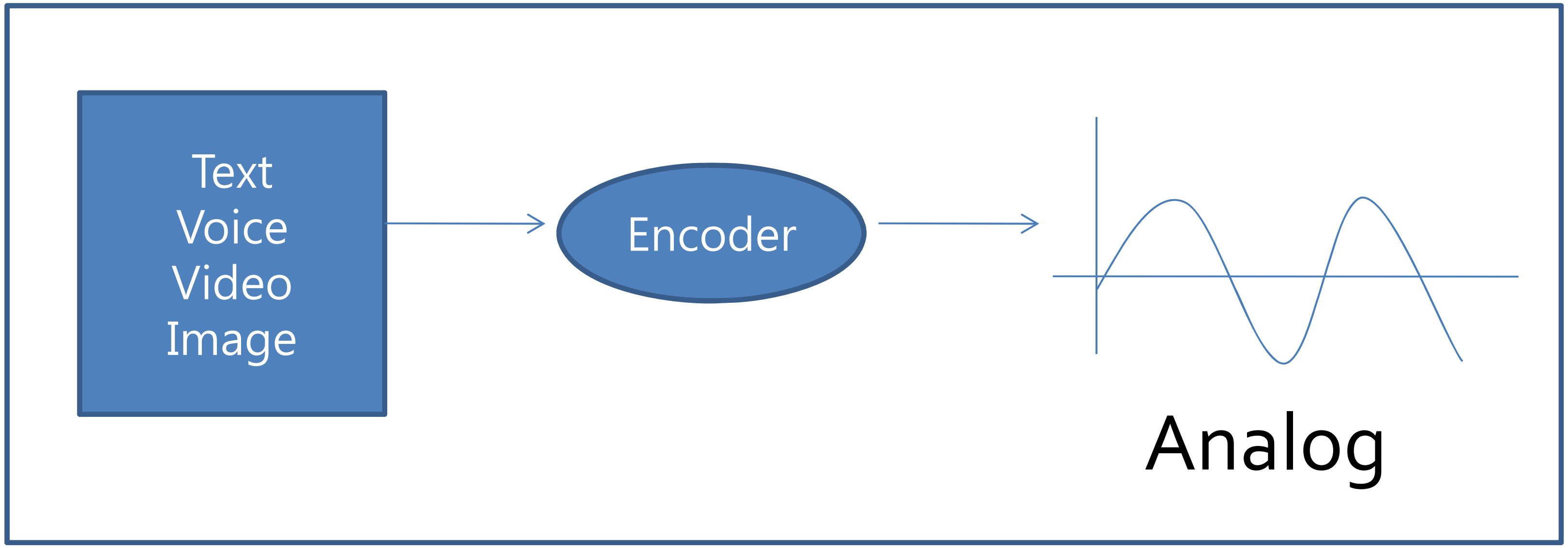
Analog – Continuous

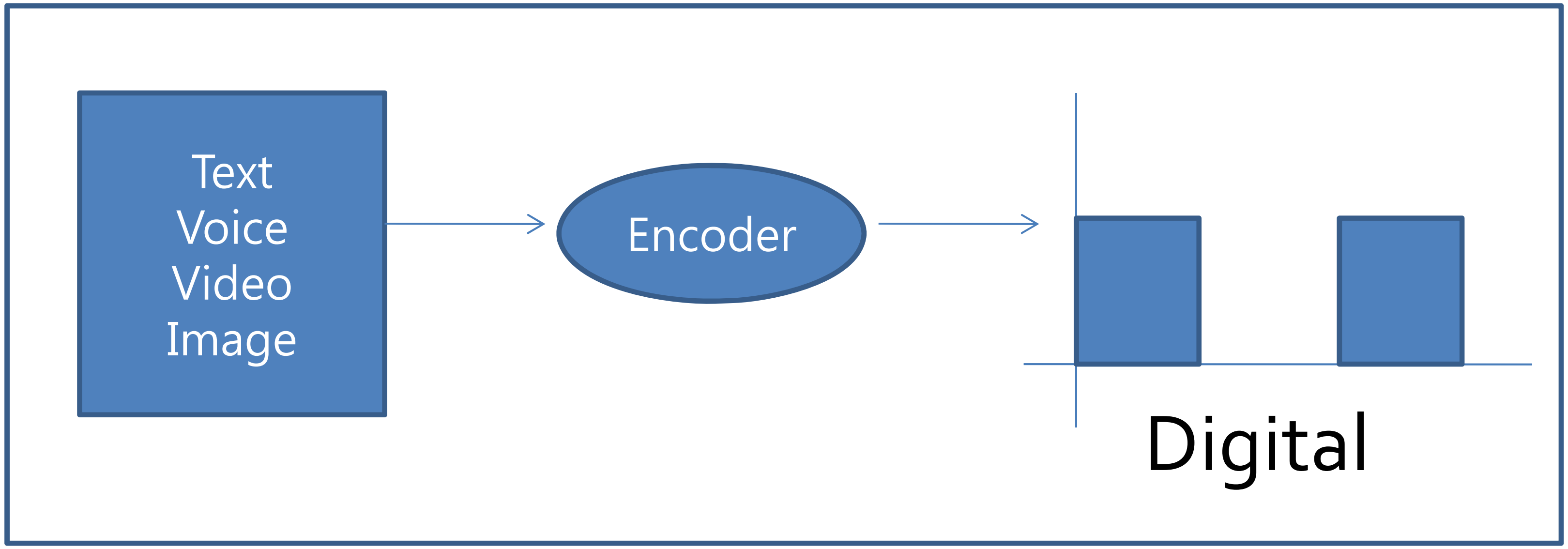
- ⇨ Natural Phenomena (Pressure, Temperature, Speed…)
- ⇨ Difficulty in realizing, processing using electronics

Digital – Discrete

- ⇨ Binary Digit → Signal processing as bit unit
- ⇨ Easy in realizing, processing using electronics
- ⇨ High performance due to integrated circuit technology

Analog versus Digital





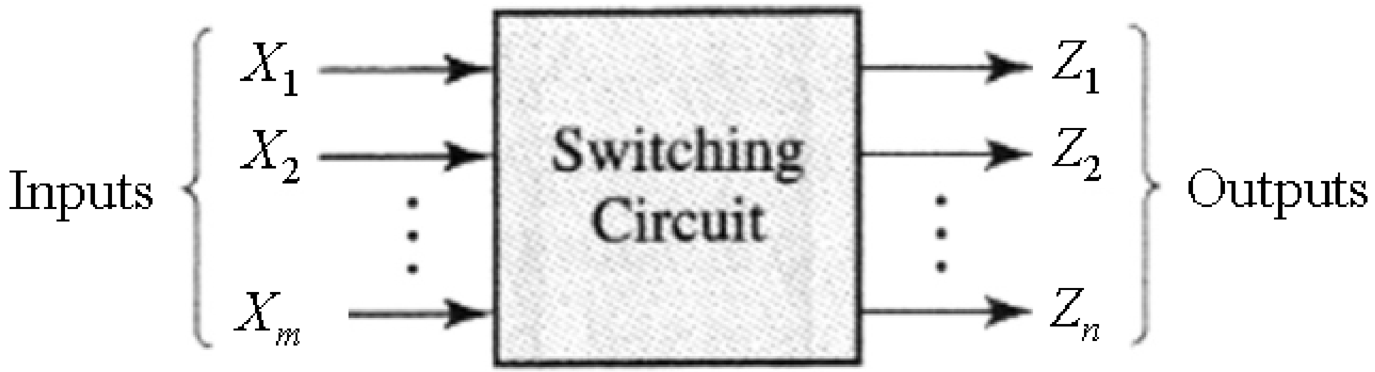
Binary Digit?

Binary

- ⇨ Two values (0,1)
- ⇨ Each digit is called as a “bit”
- ⇨ Thus, good things in binary number
- ⇨ Number representation with only two values (0,1)
- ⇨ Can be implemented with simple electronics devices
  - ⇨ Ex. Voltage high (1), low (0)
  - ⇨ Ex. Switch on (1), off (0) etc.

Switching Circuit

- **Combinational Circuit**
  - Outputs depend on only present inputs, not on past inputs
  - Have no "memory" function
- **Sequential Circuit**
  - Outputs depend on both present inputs and past inputs
  - Have "memory" function



What is Logic Design?

- Given the function, implement logic hardware for that function
  - Representation of the function
    - Sentence, speak, pseudo code, program
    - Truth table
    - Karnaugh maps
    - Minterm and Maxterm expansions
    - FSM
    - …
  - How to implement
    - You can Implement logic circuits by connecting logic gates
    - There are many logic circuits for only one function, but it is important to implement optimal one

Design Steps

- Board-level
- Chip-level
- Module-level
- Gate-level
- Circuit-level
- Transistor-level

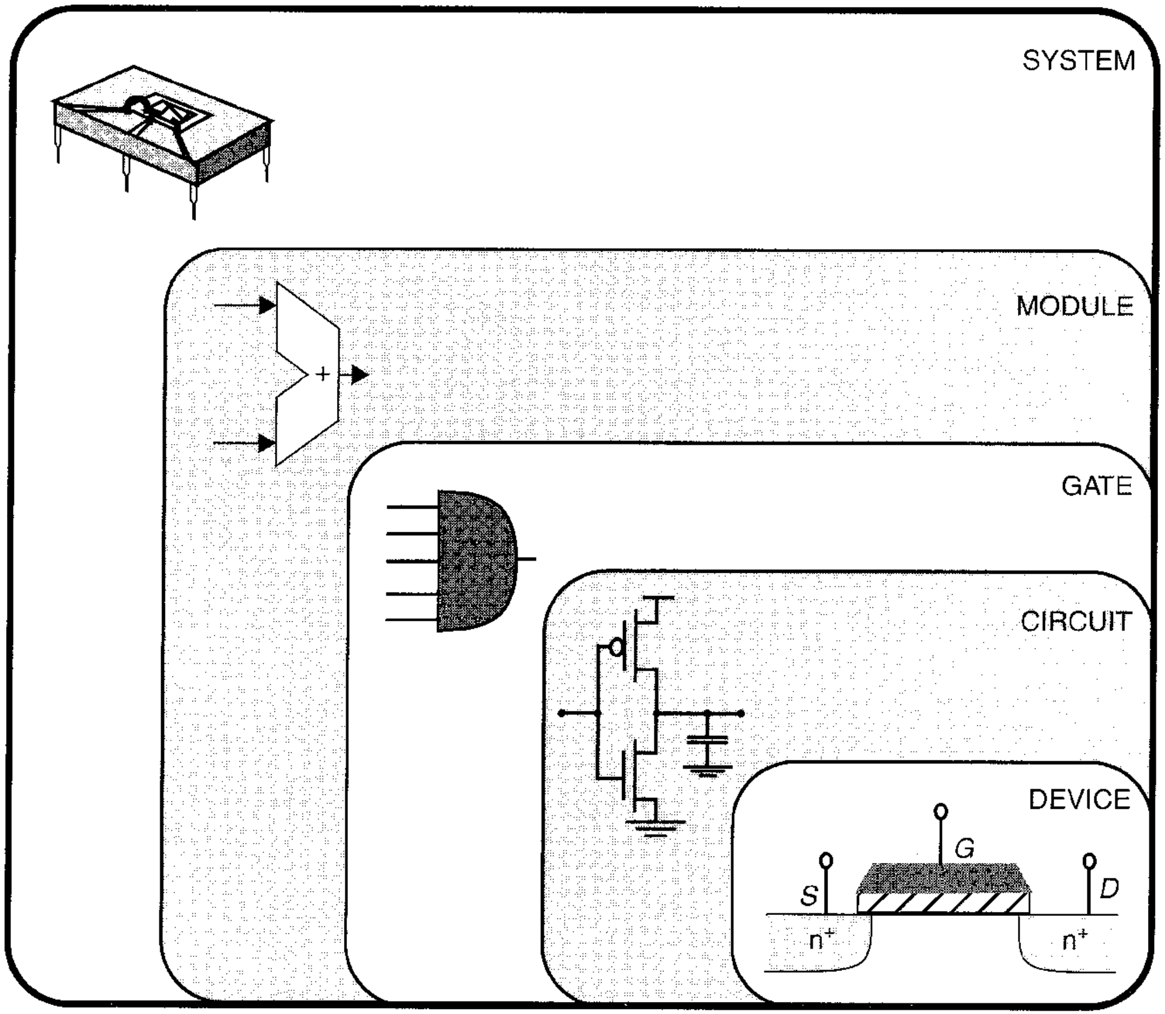
Computer Architecture
Microprocessor

Logic Design

Electronic Circuit

VLSI/CAD
SoC Design

Design Steps



[Jan Rabaey's Digital Circuit Design]

Number Systems and Conversion

- **Decimal :** $953.78_{10} = 9\times10^2 + 5\times10^1 + 3\times10^0 + 7\times10^{-1} + 8\times10^{-2}$

- **Binary :** $1011.11_2 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0 + 1\times2^{-1} + 1\times2^{-2}$

$$= 8 + 0 + 2 + 1 + \frac{1}{2} + \frac{1}{4} = 11\frac{3}{4} = 11.75_{10}$$

- **Radix(Base) :**

$$N = (a_4 a_3 a_2 a_1 a_0 . a_{-1} a_{-2} a_{-3})_R$$
$$= a_4\times R^4 + a_3\times R^3 + a_2\times R^2 + a_1\times R^1 + a_0\times R^0 + a_{-1}\times R^{-1} + a_{-2}\times R^{-2} + a_{-3}\times R^{-3}$$

- **Example :**

$$147.3_8 = 1\times8^2 + 4\times8^1 + 7\times8^0 + 3\times8^{-1} = 64 + 32 + 7 + \frac{3}{8}$$
$$= 103.375_{10}$$

- **Hexa-Decimal :**

$$A2F_{16} = 10\times16^2 + 2\times16^1 + 15\times16^0 = 2560 + 32 + 15 = 2607_{10}$$

Number Systems and Conversion

Example : Decimal to Binary Conversion

2	$\lfloor 53$	
2	$\lfloor 26$	rem. = 1 = a_0
2	$\lfloor 13$	rem. = 0 = a_1
2	$\lfloor 6$	rem. = 1 = a_2
2	$\lfloor 3$	rem. = 0 = a_3
2	$\lfloor 1$	rem. = 1 = a_4
	0	rem. = 1 = a_5

$$53_{10} = 110101_2$$

Number Systems and Conversion

Example : Convert 0.7 to Binary

$$
\begin{array}{r}
.7 \\
\underline{\quad 2} \\
(1).4 \\
\underline{\quad 2} \\
(0).8 \\
\underline{\quad 2} \\
(1).6 \\
\underline{\quad 2} \\
(1).2 \\
\underline{\quad 2} \\
(0).4 \\
\underline{\quad 2} \\
(0).8
\end{array}
$$

$\leftarrow$ **Process starts repeating here because .4 was previously obtained**

$$0.7_{10} = 0.1\,\underline{0110}\,\underline{0110}\,\underline{0110}\cdots_2$$

Number Systems and Conversion

Example : Convert 231.3_4 to Base-7

1. Convert to Decimal $231.3_4 = 2\times16+3\times4+1+\frac{3}{4} = 45.75_{10}$

2-1. Convert of a decimal integer to base 7

2-2. Convert of a decimal fraction to base 7

$$
\begin{array}{rrl}
7 & \underline{\angle\; 45} & \\
7 & \underline{\angle\;\;\, 6} & \text{rem.}3 \\
 & 0 & \text{rem.}6
\end{array}
$$

$$
\begin{array}{r}
.75 \\
\underline{\quad 7} \\
(5).25 \\
\underline{\quad 7} \\
(1).75 \\
\underline{\quad 7} \\
(5).25 \\
\underline{\quad 7} \\
(1).75
\end{array}
$$

$$45.75_{10} = 63.5151\cdots_7$$

Number Systems and Conversion

Conversion of Binary to Octal, Hexa-Decinal

- $(101011010111)_2$
  $= (\qquad\qquad\qquad\qquad)_8$, octal

- $(10111011)_2$
  $= (\qquad\qquad\qquad\qquad)_8$, octal

- $(1010111100100101)_2$
  $= (\qquad\qquad\qquad\qquad)_{16}$, Hexadecimal

- $(1101101000)_2$
  $= (\qquad\qquad\qquad\qquad)_{16}$, Hexadecimal

Binary Arithmetic

Addition

$$0+0=0$$

$$0+1=1$$

$$1+0=1$$

$$1+1=0$$ **and carry 1 to the next column**

Example

$$
\begin{array}{rcl}
 & & 1111 \leftarrow \textbf{carries} \\
13_{10} & = & 1101 \\
11_{10} & = & \underline{1011} \\
 & & 11000 = 24_{10}
\end{array}
$$

Binary Arithmetic

Subtraction

$$0 - 0 = 0$$
$$0 - 1 = 1 \quad \textbf{and borrow 1 from the next column}$$
$$1 - 0 = 1$$
$$1 - 1 = 0$$

Example

$$\begin{array}{r} 1 \\ 11101 \\ -10011 \\ \hline 1010 \end{array} \longleftarrow \text{(indicates a borrow From the } 3^{rd} \text{ column)}$$

$$\begin{array}{r} 1111 \\ 10000 \\ -\quad 11 \\ \hline 1101 \end{array} \longleftarrow \text{borrows}$$

$$\begin{array}{r} 111 \\ 111001 \\ -\quad 1011 \\ \hline 101110 \end{array} \longleftarrow \text{borrows}$$

Binary Arithmetic

Subtraction Example with Decimal

column 2 column 1

$$\begin{array}{r} 205 \\ -\ \ 18 \\ \hline 187 \end{array}$$

$$\begin{aligned}
205-18 &= [2\times10^2 + 0\times10^1 + 5\times10^0] \\
&\quad -[\qquad\qquad 1\times10^1 + 8\times10^0]
\end{aligned}$$

note borrow from column 1

$$\begin{aligned}
&= [2\times10^2 + (0-1)\times10^1 + (10+5)\times10^0] \\
&\quad -[\qquad\qquad\qquad 1\times10^1 + \qquad 8\times10^0]
\end{aligned}$$

note borrow from column 2

$$\begin{aligned}
&= [(2-1)\times10^2 + (10+0-1)\times10^1 + 15\times10^0] \\
&\quad -[\qquad\qquad\qquad\qquad\qquad 1)\times10^1 + 8\times10^0] \\
&= [1\times10^2 \qquad + \quad 8\times10^1 \qquad\quad + 7\times10^0] = 187
\end{aligned}$$

Binary Arithmetic

Multiplication

$0 \times 0 = 0$

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

Multiply: 13 x 11 (10)

1101
1011
1101
1101
0000
1101
$10001111 = 143_{10}$

1111	multiplicand
1101	multiplier
1111	first partial product
0000	second partial product
(01111)	sum of first two partial products
1111	third partial product
(1001011)	sum after adding third partial product
1111	fourth partial product
11000011	final product (sum after adding fourth partial product)

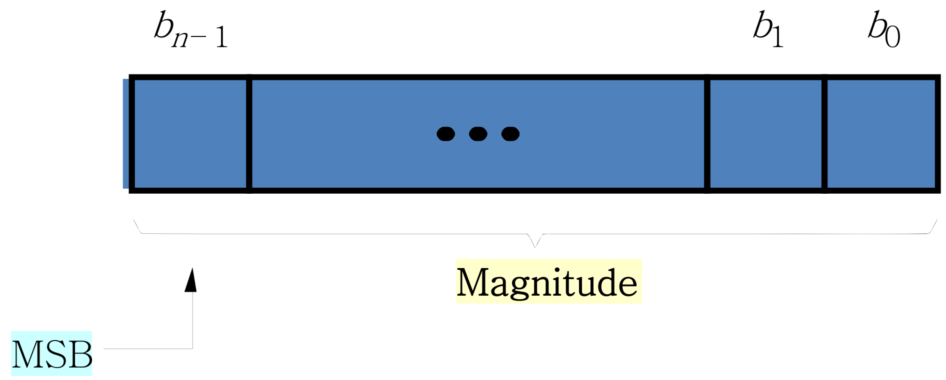
Binary Arithmetic

Division

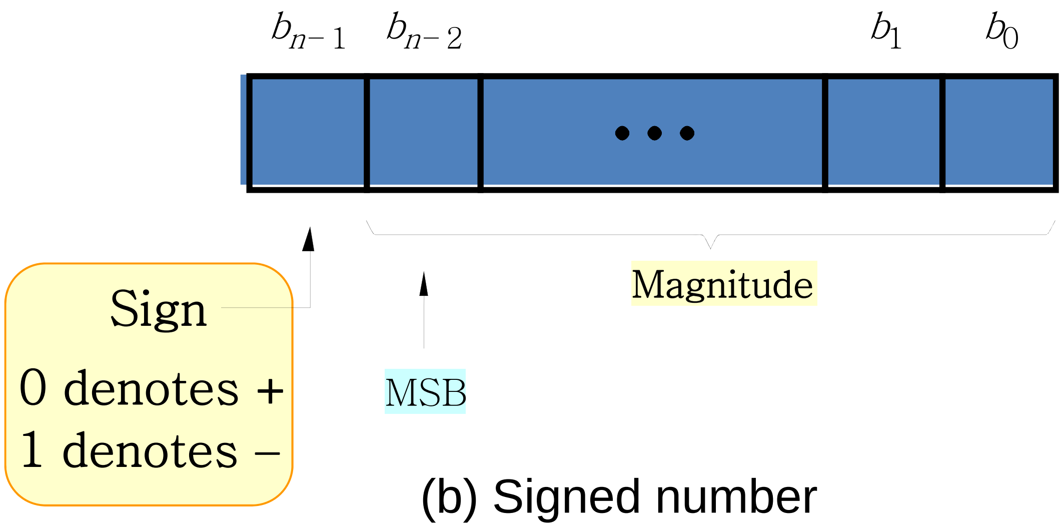
```
            1101
      ___________
  1011 | 10010001
         1011
         ----
          1110
          1011
          ----
            1101
            1011
            ----
              10
```

The quotient is 1101 with a remainder of 10.

Representation of Negative Numbers



(a) Unsigned number



(b) Signed number

Representation of Negative Numbers

2's Complement Representation for Negative Numbers

$$N^* = 2^n - N$$

+N	Positive integers (all systems)	-N	Negative integers: Sign and magnitude	Negative integers: 2's complement N^*	Negative integers: 1's complement $\overline{N}$
+0	0000	-0	1000	-	1111
+1	0001	-1	1001	1111	1110
+2	0010	-2	1010	1110	1101
+3	0011	-3	1011	1101	1100
+4	0100	-4	1100	1100	1011
+5	0101	-5	1101	1011	1010
+6	0110	-6	1110	1010	1001
+7	0111	-7	1111	1001	1000
		-8	-	1000	-

Representation of Negative Numbers

- **1's Complement Representation for Negative Numbers**

$$\overline{N} = (2^n - 1) - N$$

- **Example :**

$$\begin{array}{rl} 2^n - 1 = & 111111 \\ N = & 010101 \\ \hline \overline{N} = & 101010 \end{array}$$

$$N^* = 2^n - N = (2^n - 1 - N) + 1 = \overline{N} + 1$$

$\rightarrow$ 2's complement: 1's complement + '1'

Representation of Negative Numbers

Addition of 2's Complement Numbers

Case 1

+3	0011	
+4	0100	
+7	0111	(correct answer)

Addition of two positive numbers, sum<2^{n-1}

Case 2

+5	0101	
+6	0110	
	1011	← wrong answer because of **overflow** (+11 requires 5 bits including sign)

Addition of two positive numbers, sum≥2^{n-1}

Case 3

+5	0101	
−6	1010	
	1111	(correct answer)

Addition of positive and negative numbers

Case 4

−5	1011	
+6	0110	
	(1)0001	← correct answer when the carry from the sign bit is ignored (this is *not* an overflow)

Same as case 3 except positive number has greater magnitude

Representation of Negative Numbers

Addition of 2's Complement Numbers

Case 5

-3	1101	Addition of two negative numbers, $\lvert sum \rvert \le 2^{n-1}$
$\underline{-4}$	$\underline{1100}$	
-7	(1)1001	← correct answer when the last carry is ignored (this is *not* an overflow)

Case 6

-5	1011	Addition of two negative numbers, $\lvert sum \rvert > 2^{n-1}$
$\underline{-6}$	$\underline{1010}$	
	(1)0101	← wrong answer because of overflow (-11 requires 5 bits including sign)

Representation of Negative Numbers

Addition of 1's Complement Numbers

Case 3

```
 + 5     0101
 − 6     1001
 − 1     1110      (correct answer)
```

Case 4

```
 − 5      1010
 + 6      0110
      (1) 0000
       └───→ 1    (end-around carry)
                  (correct answer, no overflow)
          0001
```

Case 5

```
 − 3     1100
 − 4     1011
     (1) 0111
      └───→ 1    (end-around carry)
         1000    (correct answer, no overflow)
```

Representation of Negative Numbers

Addition of 1's Complement Numbers

Case 6

$$
\begin{array}{rcr}
 & & 1010 \\
-5 & & \underline{1001} \\
\underline{-6} & (1) & 0011 \\
 & & \underline{\quad 1} \\
 & & 0100
\end{array}
$$

(end-around carry)

(wrong answer because of overflow)

$$Case\,4: \quad -A+B \;\; (\text{where } B > A)$$

$$\overline{A}+B=(2^n-1-A)+B=2^n+(B-A)-1$$

$$Case\,5: \quad -A-B \;\; (A+B<2^{n-1})$$

$$\overline{A}+\overline{B}=(2^n-1-A)+(2^n-1-B)=2^n+[2^n-1-(A+B)]-1$$

Representation of Negative Numbers

- **Addition of 1's Complement Numbers using 8-bit storage**

$$
\begin{array}{rrl}
 & 11110100 & \quad (-11) \\
 & \underline{11101011} & \quad \underline{+(-20)} \\
(1) & 11011111 & \\
 & \underline{1} & \quad \text{(end-around carry)} \\
 & 11100000 = (-31) &
\end{array}
$$

- **Addition of 2's Complement Numbers using 8-bit storage**

$$
\begin{array}{rl}
11111000 & \quad (-8) \\
\underline{00010011} & \quad \underline{+19} \\
\cancel{(1)}00001011 & \quad = +11
\end{array}
$$

(discard last carry)

Binary Codes

$$\overbrace{1001}^{9}\ \overbrace{0011}^{3}\ \overbrace{0111}^{7}\ .\ \overbrace{0010}^{2}\ \overbrace{0101}^{5}$$

Decimal Digit	8-4-2-1 Code (BCD)	6-3-1-1 Code	Excees-3 Code
0	0000	0000	0011
1	0001	0001	0100
2	0010	0011	0101
3	0011	0100	0110
4	0100	0101	0111
5	0101	0111	1000
6	0110	1000	1001
7	0111	1001	1010
8	1000	1011	1011
9	1001	1100	1100

Binary Codes

6-3-1-1 Code

$$N = w_3 a_3 + w_2 a_2 + w_1 a_1 + w_0 a_0$$

$$N = 6 \cdot 1 + 3 \cdot 0 + 1 \cdot 1 + 1 \cdot 1 = 8$$

ASCII Code : 7-bit code

1010011	1110100	1100001	1110010	1110100
S	t	a	r	t

Examples

- **Convert to hexadecimal and then to binary**

  (a) 1305.375_{10} (b) 1644.875_{10}

- **Add the following numbers in binary using 2's complement to represent negative numbers. Use a word length of 7 bits (including sign) and indicate if an overflow occurs.**

  (a) $(21)_{10} + (43)_{10}$
  (b) $(-10)_{10} + (-11)_{10}$
  (c) $(-12)_{10} + (13)_{10}$