SoC Design : Lecture 2
Review of Logic Circuit

Spring 2024

School of IT Convergence
Prof. Jong-Myon Kim

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

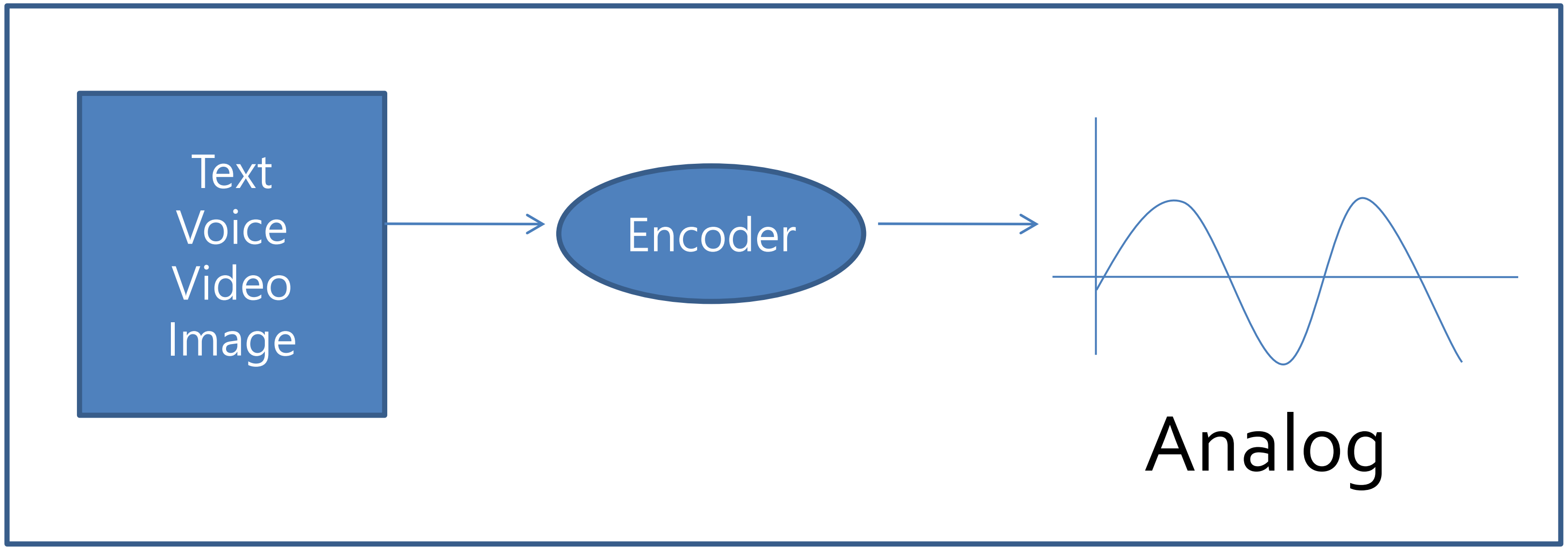
Digital – Discrete

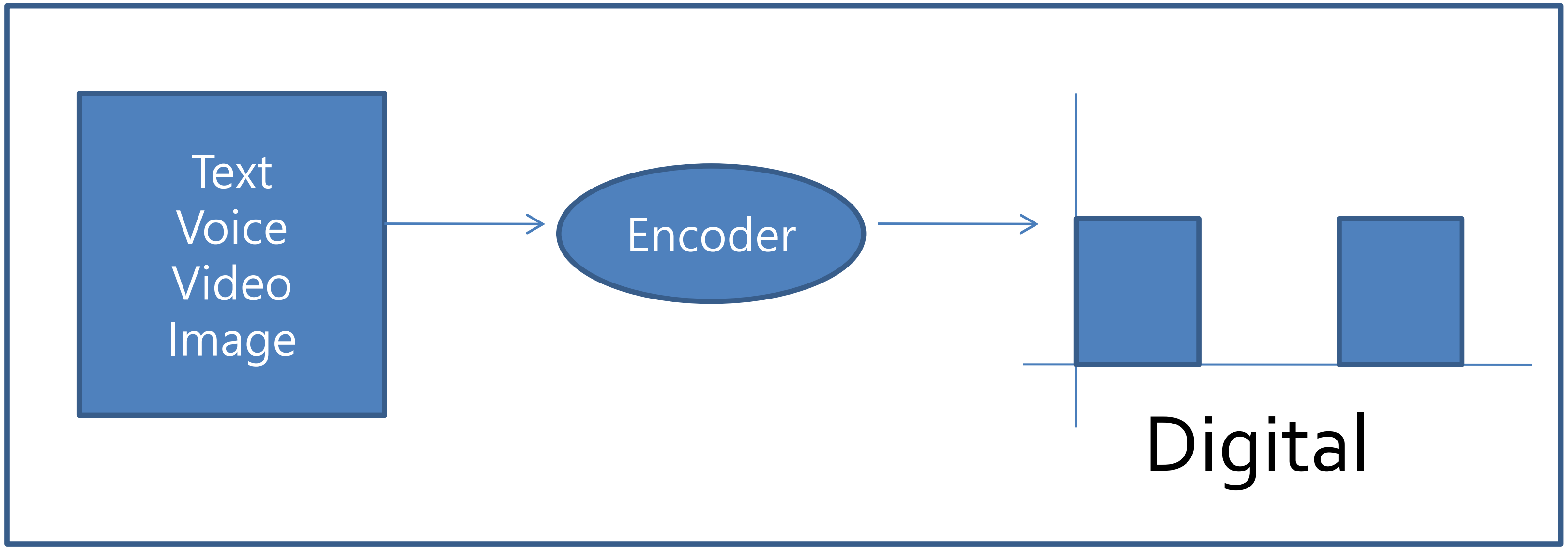
⇨ Binary Digit → Signal processing as bit unit

⇨ Easy in realizing, processing using electronics

⇨ High performance due to integrated circuit technology

Analog versus Digital





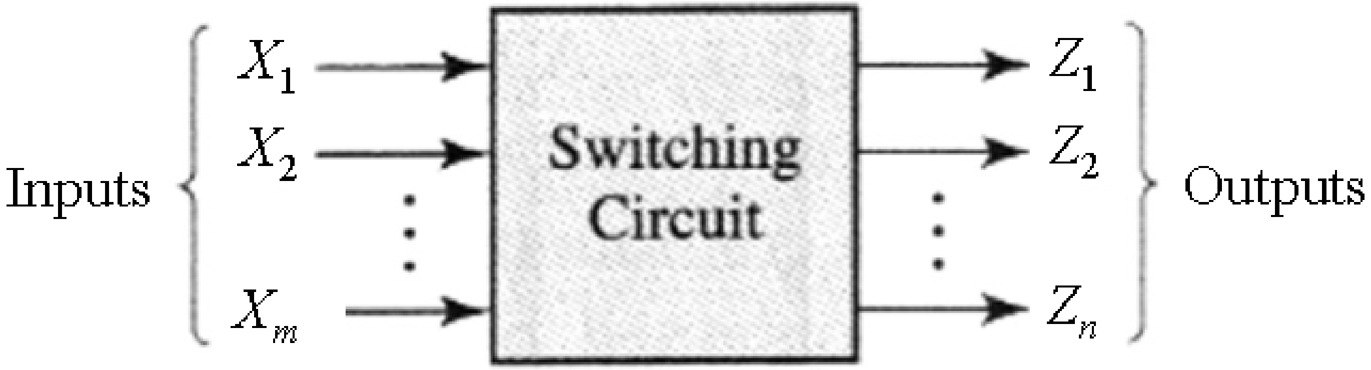
Binary Digit?

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

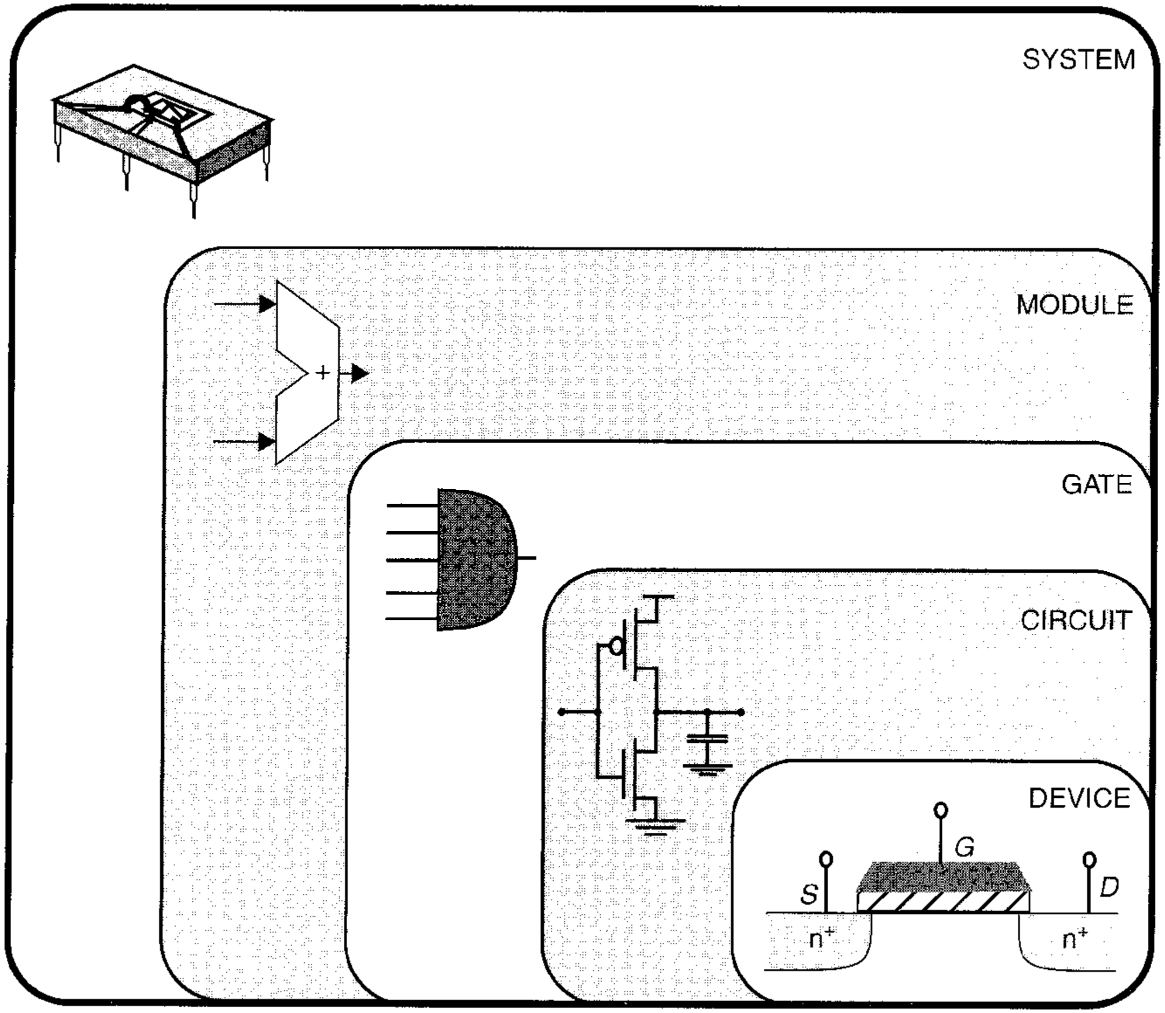
- **Combinational Circuit**
  - Outputs depend on only present inputs, not on past inputs
  - Have no “memory” function
- **Sequential Circuit**
  - Outputs depend on both present inputs and past inputs
  - Have “memory” function



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



[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) :**

$$N = (a_4a_3a_2a_1a_0.a_{-1}a_{-2}a_{-3})_R$$
$$= a_4\times R^4 + a_3\times R^3 + a_2\times R^2 + a_1\times R^1 + a_0\times R^0 + a_{-1}\times R^{-1} + a_{-2}\times R^{-2} + a_{-3}\times R^{-3}$$

- **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

$$
\begin{array}{rll}
2 & \underline{\big/\,53} & \\
2 & \underline{\big/\,26} & \text{rem.} = 1 = a_0 \\
2 & \underline{\big/\,13} & \text{rem.} = 0 = a_1 \\
2 & \underline{\big/\,6} & \text{rem.} = 1 = a_2 \\
2 & \underline{\big/\,3} & \text{rem.} = 0 = a_3 \\
2 & \underline{\big/\,1} & \text{rem.} = 1 = a_4 \\
  & \ \ 0 & \text{rem.} = 1 = a_5
\end{array}
$$

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

Number Systems and Conversion

Example : Convert 0.7 to Binary

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

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

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

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

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

Number Systems and Conversion

Conversion of Binary to Octal, Hexa-Decinal

- $(101011010111)_2$
  $= (\qquad\qquad)_8$, octal
- $(10111011)_2$
  $= (\qquad\qquad)_8$, octal
- $(1010111100100101)_2$
  $= (\qquad\qquad)_{16}$, Hexadecimal
- $(1101101000)_2$
  $= (\qquad\qquad)_{16}$, Hexadecimal

Binary Arithmetic

Addition

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

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

Example

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

Binary Arithmetic

Subtraction

$$0 - 0 = 0$$

$$0 - 1 = 1$$ **and borrow 1 from the next column**

$$1 - 0 = 1$$

$$1 - 1 = 0$$

Example

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

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

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

Binary Arithmetic

Subtraction Example with Decimal

column 2 column 1

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

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

note borrow from column 1

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

note borrow from column 2

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

Binary Arithmetic

Multiplication

$0 \times 0 = 0$

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

```
     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)
```

Multiply: 13 x 11 (10)

```
     1101
     1011
  -------
     1101
    1101
   0000
  1101
  -------
 10001111
```

$10001111 = 143_{10}$

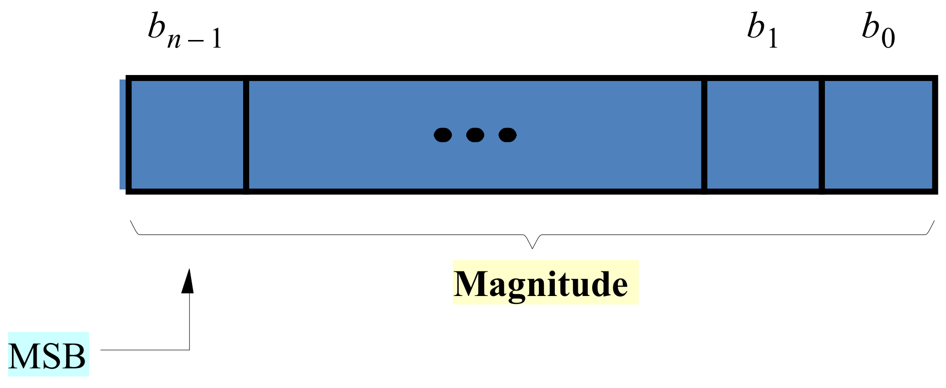
Binary Arithmetic

Division

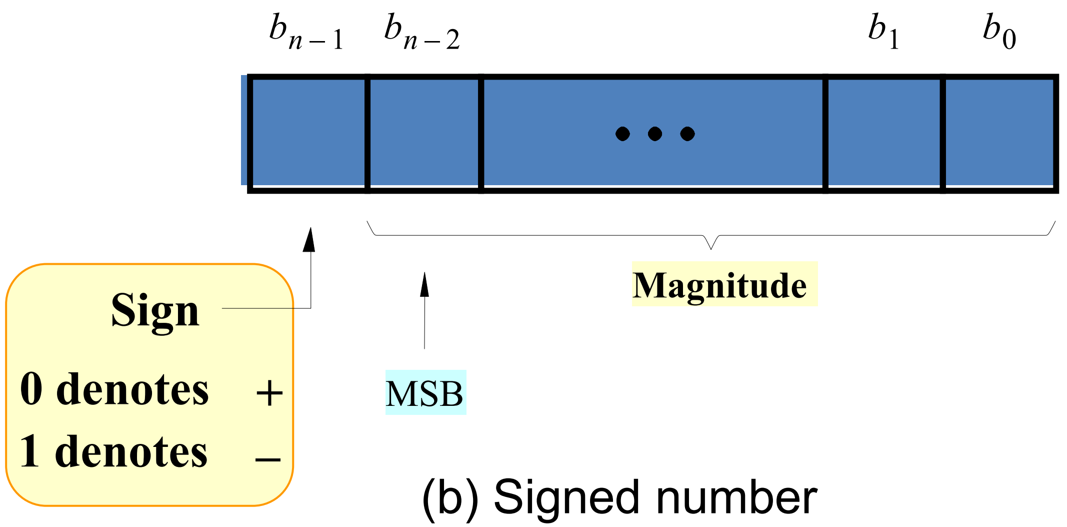
$$
\begin{array}{r}
1101 \\
1011\overline{\big)\,10010001} \\
\underline{1011} \\
1110 \\
\underline{1011} \\
1101 \\
\underline{1011} \\
10
\end{array}
$$

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

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

			Negative integers		
+N	Positive integers (all systems)	-N	Sign and magnitude	2’s complement N^*	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 :**

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

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

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

Representation of Negative Numbers

Addition of 2's Complement Numbers

Case 1

$$\begin{array}{rr} +3 & 0011 \\ \underline{+4} & \underline{0100} \\ +7 & 0111 \end{array}$$

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

(correct answer)

Case 2

$$\begin{array}{rr} +5 & 0101 \\ \underline{+6} & \underline{0110} \\ & 1011 \end{array}$$

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

$\longleftarrow$ wrong answer because of **overflow** (+11 requires 5 bits including sign)

Case 3

$$\begin{array}{rr} +5 & 0101 \\ \underline{-6} & \underline{1010} \\ & 1111 \end{array}$$

Addition of positive and negative numbers

(correct answer)

Case 4

$$\begin{array}{rr} -5 & 1011 \\ \underline{+6} & \underline{0110} \\ & (1)0001 \end{array}$$

Same as case 3 except positive number has greater magnitude

$\longleftarrow$ correct answer when the carry from the sign bit is ignored (this is *not* an overflow)

Representation of Negative Numbers

Addition of 2's Complement Numbers

Case 5

Addition of two negative numbers, $|sum| \le 2^{n-1}$

$$\begin{array}{rr} -3 & 1101 \\ \underline{-4} & \underline{1100} \\ -7 & (1)1001 \end{array}$$

$\longleftarrow$ correct answer when the last carry is ignored (this is *not* an overflow)

Case 6

Addition of two negative numbers, $|sum| > 2^{n-1}$

$$\begin{array}{rr} -5 & 1011 \\ \underline{-6} & \underline{1010} \\ & (1)0101 \end{array}$$

$\longleftarrow$ wrong answer because of overflow (-11 requires 5 bits including sign)

Representation of Negative Numbers

- **Addition of 1’s Complement Numbers**

Case 3

$$\begin{array}{rr l} +5 & 0101 & \\ \underline{-6} & \underline{1001} & \\ -1 & 1110 & \text{(correct answer)} \end{array}$$

Case 4

$$\begin{array}{rr l} -5 & 1010 & \\ \underline{+6} & \underline{0110} & \\ (1) & 0000 & \\ \llcorner\!\!\longrightarrow & \underline{1} & \text{(end-around carry)} \\ & 0001 & \text{(correct answer, } \textit{no} \text{ overflow)} \end{array}$$

Case 5

$$\begin{array}{rr l} -3 & 1100 & \\ \underline{-4} & \underline{1011} & \\ (1) & 0111 & \\ \llcorner\!\!\longrightarrow & \underline{1} & \text{(end-around carry)} \\ & 1000 & \text{(correct answer, } \textit{no} \text{ overflow)} \end{array}$$

Representation of Negative Numbers

- **Addition of 1's Complement Numbers**

Case 6

$$\begin{array}{rrrl} & & 1010 & \\ -5 & & \underline{1001} & \\ \underline{-6} & (1) & 0011 & \\ & \llcorner\!\!\longrightarrow & \underline{1} & \text{(end-around carry)} \\ & & 0100 & \text{(wrong answer because of overflow)} \end{array}$$

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

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

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

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

Representation of Negative Numbers

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

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

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

$$
\begin{array}{rl}
11111000 & \quad (-8) \\
\underline{00010011} & \quad \underline{+19} \\
\cancel{(1)}00001011 & \quad = +11 \\
\uparrow\!\llcorner\, \text{(discard last carry)} &
\end{array}
$$

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**

$$N = w_3a_3 + w_2a_2 + w_1a_1 + w_0a_0$$

$$N = 6\cdot1 + 3\cdot0 + 1\cdot1 + 1\cdot1 = 8$$

- **ASCII Code : 7-bit code**

1010011	1110100	1100001	1110010	1110100
S	t	a	r	t

Objectives

Topics introduced in this chapter

- ⇨ Understand the basic operations and laws of Boolean algebra
- ⇨ Relate these operations and laws to AND, OR, NOT gates and switches
- ⇨ Prove these laws using a truth table
- ⇨ Manipulation of algebraic expression using
    - ⇨ Multiplying out
    - ⇨ Factoring
    - ⇨ Simplifying
    - ⇨ Finding the complement of an expression

Introduction

- Basic mathematics for logic design: Boolean algebra
- Restrict to switching circuits (Two state values 0, 1) – Switching algebra
- Boolean Variable : X, Y, … can only have two state values (0, 1)
  - representing True(1) False (0)

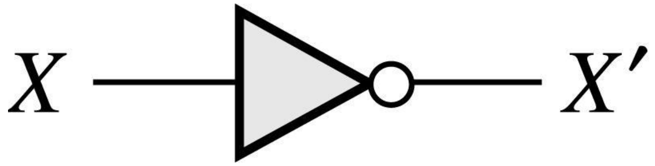
Basic Operations

- **Not (Inverter)**

$$0' = 1 \text{ and } 1' = 0$$

$$X' = 1 \text{ if } X = 0 \text{ and } X' = 0 \text{ if } X = 1$$

- **Gate Symbol**



Basic Operations

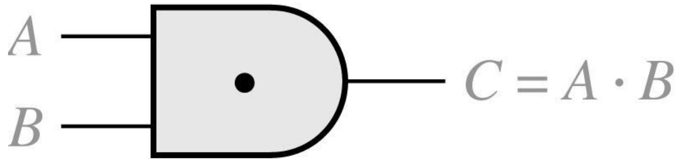
- **AND**

$$0 \cdot 0 = 0,\ \ 0 \cdot 1 = 0,\ \ 1 \cdot 0 = 0,\ \ 1 \cdot 1 = 1$$

- **Truth Table**

A B	$C = A \cdot B$
0 0	0
0 1	0
1 0	0
1 1	1

- **Gate Symbol**



Basic Operations

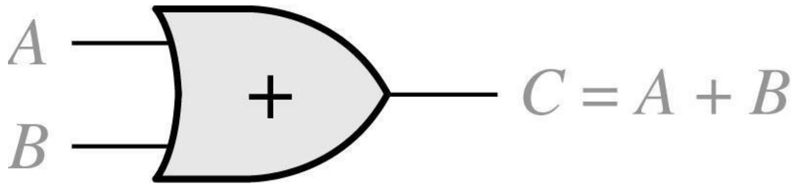
- **OR**

$$0+0=0,\ 0+1=1,\ 1+0=1,\ 1+1=1$$

- **Truth Table**

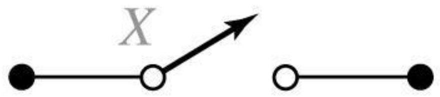
A B	$C = A \cdot B$
0 0	0
0 1	1
1 0	1
1 1	1

- **Gate Symbol**



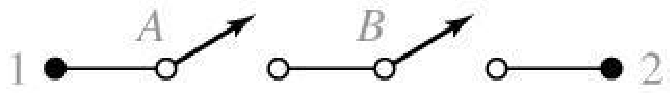
Basic Operations

- **Apply to Switch**



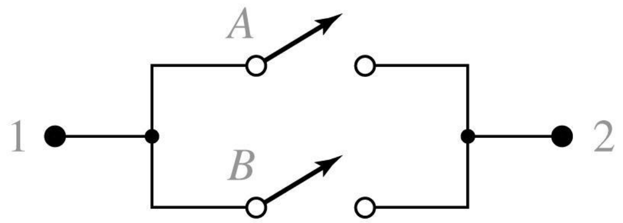
$X = 0 \rightarrow$ switch open
$X = 1 \rightarrow$ switch closed

- **AND T=A·B**



$T = 0 \rightarrow$ open circuit between terminals 1 and 2
$T = 1 \rightarrow$ closed circuit between terminals 1 and 2

- **OR T=A+B**

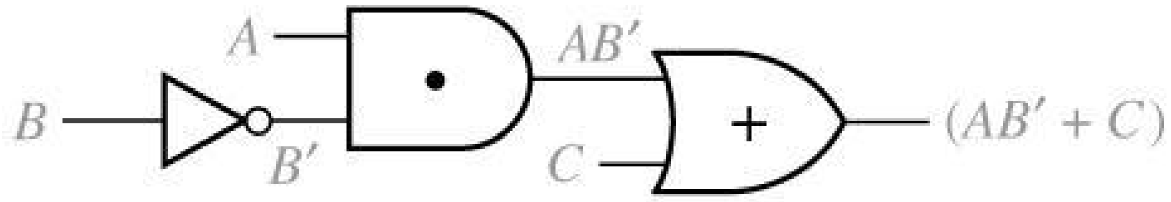


Boolean Expressions and Truth Tables

- **Logic Expression :**

$$(AB'+C)$$

- **Circuit of Logic**

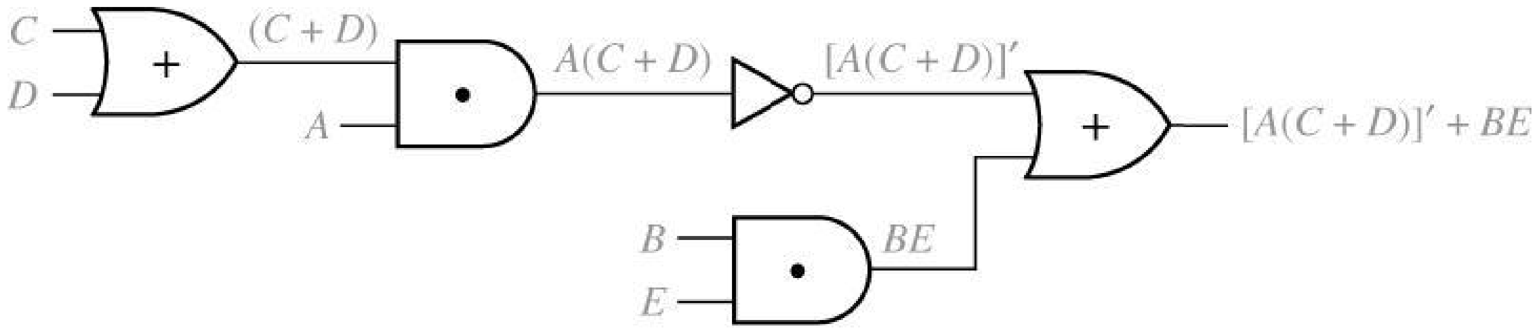


Boolean Expressions and Truth Tables

- **Logic Expression :**

$$[A(C+D)]'+BE$$

- **Circuit of Logic**

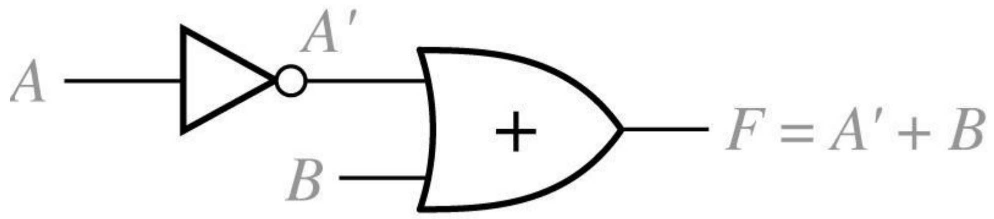


- **Logic Evaluation : A=B=C=1, D=E=0**

$$[A(C+D)]'+BE=[1(1+0)]'+1\cdot 0=[1(1)]'+0=0+0=0$$

Boolean Expressions and Truth Tables

- **2-Input Circuit and Truth Table**



A B		A'	F = A' + B
0 0		1	1
0 1		1	1
1 0		0	0
1 1		0	1

Boolean Expressions and Truth Tables

Proof using Truth Table

$$AB'+C=(A+C)(B'+C)$$

n variable needs $\underbrace{2x2x2x}_{\text{n times}}\cdots = 2^n$ rows

A B C		B'	AB'	AB' + C		A + C	B' + C	(A + C) (B' + C)
0 0 0		1	0	0		0	1	0
0 0 1		1	0	1		1	1	1
0 1 0		0	0	0		0	0	0
0 1 1		0	0	1		1	1	1
1 0 0		1	1	1		1	1	1
1 0 1		1	1	1		1	1	1
1 1 0		0	0	0		1	0	0
1 1 1		0	0	1		1	1	1

Basic Theorems

- **Operations with 0, 1** $X+0=X$ $X\cdot 1=X$

  $X+1=1$ $X\cdot 0=0$
- **Idempotent Laws** $X+X=X$ $X\cdot X=X$
- **Involution Laws** $(X')'=X$
- **Complementary Laws** $X+X'=1$ $X\cdot X'=0$

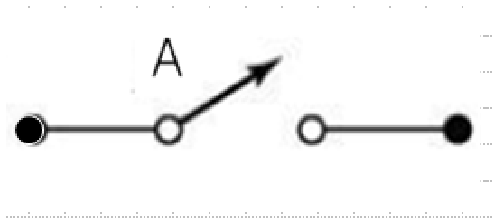
- **Proof** $X=0,\quad 0+0'=0+1,\quad \text{and if } X=1,\quad 1+1'=1+0=1$

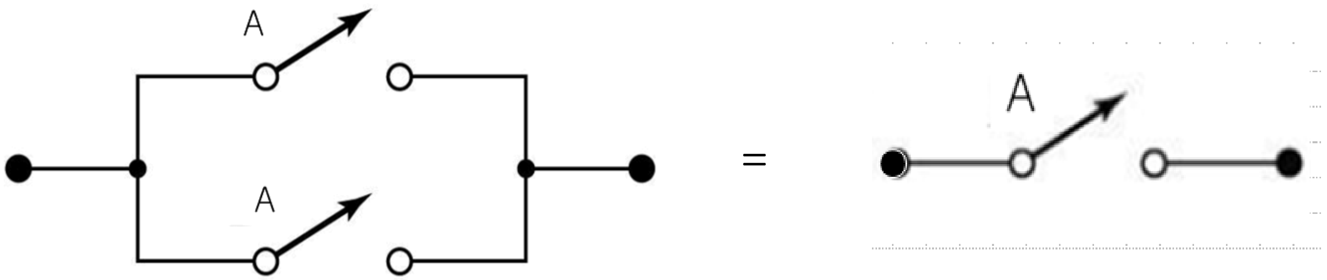
- **Examples**

$$(AB'+D)E+1=1$$

$$(AB'+D)(AB'+D)'=0$$

Basic Theorems with Switch Circuits





Basic Theorems with Switch Circuits

=

Commutative, Associative, Distributive

- **Commutative Laws :** $XY = YX, \quad X + Y = Y + X$
- **Associative Laws :** $(XY)Z = X(YZ) = XYZ$

  $(X+Y)+Z = X+(Y+Z) = X+Y+Z$

- **Proof of Associative Law for AND**

X Y Z	XY	YZ	(XY)Z	X(YZ)
0 0 0	0	0	0	0
0 0 1	0	0	0	0
0 1 0	0	0	0	0
0 1 1	0	1	0	0
1 0 0	0	0	0	0
1 0 1	0	0	0	0
1 1 0	1	0	0	0
1 1 1	1	1	1	1

Associative Laws for AND and OR

Commutative, Associative, Distributive

- **AND :** $XYZ = 1 \text{ iff } X = Y = Z = 1$
- **OR :** $X + Y + Z = 0 \text{ iff } X = Y = Z = 0$
- **Distribute Laws :** $X(Y+Z) = XY + XZ$

$$X + YZ = (X+Y)(X+Z)$$

Valid only Boolean algebra not for ordinary algebra

Proof

$$(X+Y)(X+Z) = X(X+Z) + Y(X+Z) = XX + XZ + YX + YZ$$
$$= X + XZ + XY + YZ = X \cdot 1 + XZ + XY + YZ$$
$$= X(1 + Z + Y) + YZ = X \cdot 1 + YZ = X + YZ$$

Simplification Theorems

Useful Theorems for Simplification

$XY + XY' = X$	$(X+Y)(X+Y') = X$
$X + XY = X$	$X(X+Y) = X$
$(X+Y')Y = XY$	$XY' + Y = X + Y$

Proof

$$X + XY = X \cdot 1 + XY = X(1+Y) = X \cdot 1 = X$$
$$X(X+Y) = XX + XY = X + XY = X$$
$$Y + XY' = (Y+X)(Y+Y') = (Y+X)1 = Y + X$$

Proof with Switch

$$=$$

Simplification Theorems

- **Equivalent Gate Circuits**

$$F = A(A'+B) = AB$$

$=$

Multiplying Out and Factoring

To obtain a sum-of-product form ➔ Multiplying out using distributive laws

- **Sum of product form :** $AB' + CD'E + AC'E$
- **Not in sum of product form :** $(A+B)CD + EF$
- **Multiplying out and eliminating redundant terms :**

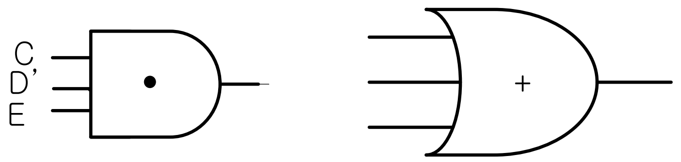
$$\begin{aligned}(A+BC)(A+D+E) &= A + AD + AE + ABC + BCD + BCE \\ &= A(1+D+E+BC) + BCD + BCE \\ &= A + BCD + BCE\end{aligned}$$

Multiplying Out and Factoring

To obtain a product of sum form → all sums are the sum of single variable

- **Product of sum form :** $(A+B')(C+D'+E)(A+C'+E')$

Circuits of SOP and POS Forms



Sum of product form

Product of sum form

DeMorgan’ s Laws

DeMorgan’s Laws

$$(X+Y)' = X'Y'$$

$$(XY)' = X'+Y'$$

Proof

X Y		X’ Y’	X + Y	(X + Y)’	X’ Y’		XY	(XY)’	X’ + Y’
0 0		1 1	0	1	1		0	1	1
0 1		1 0	1	0	0		0	1	1
1 0		0 1	1	0	0		0	1	1
1 1		0 0	1	0	0		1	0	0

DeMorgan’s Laws for n variables

$$(X_1 + X_2 + X_3 + \ldots + X_n)' = X_1'X_2'X_3'\ldots X_n'$$

$$(X_1X_2X_3\ldots X_n)' = X_1'+X_2'+X_3'+\ldots+X_n'$$

Example

$$(X_1 + X_2 + X_3)' = (X_1 + X_2)'X_3' = X_1'X_2'X_3'$$

DeMorgan' s Laws

- **Inverse of A'B + AB'**

$$F' = (A'B + AB')' = (A'B)'(AB')' = (A+B')(A'+B)$$
$$= AA' + AB + B'A' + BB' = A'B' + AB$$

A B		A' B	A B'	F = A'B+AB'		A' B'	A B	F' = A'B' + AB
0 0		0	0	0		1	0	1
0 1		1	0	1		0	0	0
1 0		0	1	1		0	0	0
1 1		0	0	0		0	1	1

- **Dual: 'dual' is formed by replacing AND with OR, OR with AND, 0 with 1, 1 with 0**

$$(XYZ\ldots)^D = X + Y + Z + \ldots \qquad (X + Y + Z + \ldots)^D = XYZ\ldots$$

$$(AB'+C)' = (AB')'C' = (A'B)C', \qquad \text{so} \qquad (AB'+C)^D = (A+B')C$$

Objectives

Topics introduced in this chapter

- ⇨ Apply Boolean laws and theorems to manipulation of expression
- ⇨ Simplifying
- ⇨ Finding the complement
- ⇨ Multiplying out and factoring
- ⇨ Exclusive-OR and Equivalence operation (Exclusive-NOR)
- ⇨ Consensus theorem

Multiplying Out and Factoring Expressions

To obtain a sum-of-product form ➔ Multiplying out using distributive laws

$$X(Y+Z) = XY + XZ$$

$$(X+Y)(X+Z) = X + YZ$$

- **Theorems for multiplying out**

$$(\underbrace{X+\overbrace{Y)(X'}+Z}) = XZ + X'Y \qquad (3\text{-}3)$$

$$\text{If } X = 0, (3\text{-}3) \text{ reduces to } Y(1+Z) = 0 + 1{*}Y \text{ or } Y = Y.$$

$$\text{If } X = 1, (3\text{-}3) \text{ reduces to } (1+Y)Z = Z + 0{*}Y \text{ or } Z = Z.$$

because the equation is valid for both $X = 0$ and $X = 1$, it is always valid.

The following example illustrates the use of Theorem (3-3) for factoring:

- **Theorems for factoring**

$$\underbrace{A\overbrace{B+A'}C} = (A+C)(A'+B)$$

Multiplying Out and Factoring Expressions

- **Theorems for multiplying out**

$$(Q+AB')(C'D+Q')=QC'D+Q'AB'$$

- **Multiplying out using distributed laws**

$$(Q+AB')(C'D+Q')=QC'D+\boxed{QQ'+AB'C'D}+AB'Q'$$

Redundant terms

- **Multiplying out : (1) distributed laws, (2) theorem (3-3)**

$$(A+B+C')(A+B+D)(A+B+E)(A+D'+E)(A'+C)$$

$$=(A+B+C'D)(A+B+E)[AC+A'(D'+E)]$$

$$=(A+B+C'DE)(AC+A'D'+A'E)$$

$$=AC+\cancel{ABC}+A'BD'+A'BE+A'C'DE$$

What theorem was applied to eliminate ABC?

Multiplying Out and Factoring Expressions

To obtain a product-of-sum form ➔ Factoring using distributive laws

- **Theorems using factoring**

$$\underbrace{A\overbrace{B+A'}C}_{} = (A+C)(A'+B)$$

- **Example of factoring**

$$AC + A'BD' + A'BE + A'C'DE$$

$$= \underbrace{AC}_{XZ} + \underset{X'}{A'}(\underbrace{BD' + BE + C'DE}_{Y})$$

$$= (A + BD' + BE + C'DE)(A' + C)$$

$$= [\underbrace{A + C'DE}_{X} + \underset{Y}{B}(\underbrace{D' + E}_{Z})](A' + C)$$

$$= (A + B + C'DE)(A + \cancel{C'DE} + D' + E)(A' + C)$$

$$= (A + B + C')(A + B + D)(A + B + E)(A + D' + E)(A' + C)$$

Exclusive-OR and Equivalence Operations

- **Exclusive-OR**

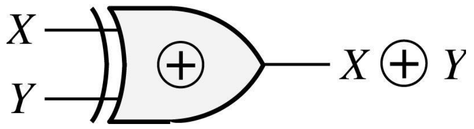
$0 \oplus 0 = 0 \qquad 0 \oplus 1 = 1$

$1 \oplus 0 = 1 \qquad 1 \oplus 1 = 0$

- **Truth Table**

XY	$X \oplus Y$
0 0	0
0 1	1
1 0	1
1 1	0

- **Symbol**



Exclusive-OR and Equivalence Operations

Theorems for Exclusive-OR:

$$X \oplus Y = X'Y + XY'$$

Because $X \oplus Y = 1$ iff X is 0 and Y is 1 or X is 1 and Y is 0

$$X \oplus 0 = X$$

$$X \oplus 1 = X'$$

$$X \oplus X = 0$$

$$X \oplus X' = 1$$

$$X \oplus Y = Y \oplus X \text{ (commutative law)}$$

$$(X \oplus Y) \oplus Z = X \oplus (Y \oplus Z) = X \oplus Y \oplus Z \text{ (associative law)}$$

$$X(Y \oplus Z) = XY \oplus XZ \text{ (distributive law)}$$

$$(X \oplus Y)' = X \oplus Y' = X' \oplus Y = XY + X'Y'$$

Exclusive-OR and Equivalence Operations

Equivalence operation (Exclusive-NOR)

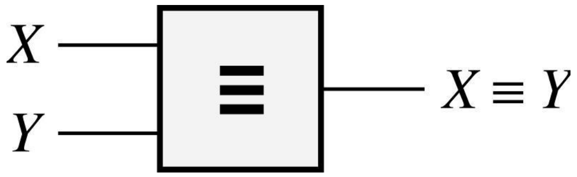
$$(0 \equiv 0) = 1 \quad (0 \equiv 1) = 0$$

$$(1 \equiv 0) = 0 \quad (1 \equiv 1) = 1$$

Truth Table

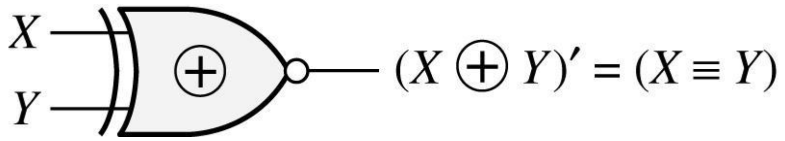
XY	$X \equiv Y$
0 0	1
0 1	0
1 0	0
1 1	1

Symbol



Exclusive-OR and Equivalence Operations

Exclusive-NOR



Example of EXOR and Equivalence

$$F = (A'B \equiv C) + (B \oplus AC')$$

$$\begin{aligned} F &= [(A'B)C + (A'B)'C'] + [B'(AC') + B(AC')'] \\ &= A'BC + (A+B')C' + AB'C' + B(A'+C) \\ &= B(A'C + A' + C) + C'(A + B' + AB') = B(A'+C) + C'(A+B') \end{aligned}$$

Useful theorem

$$(XY' + X'Y)' = XY + X'Y' \qquad (3\text{-}19)$$

$$\begin{aligned} A' \oplus B \oplus C &= [A'B' + (A')'B] \oplus C \\ &= (A'B' + AB)C' + (A'B' + AB)'C && \text{(by (3-6))} \\ &= (A'B' + AB)C' + (A'B + AB')C && \text{(by (3-19))} \\ &= A'B'C' + ABC' + A'BC + AB'C \end{aligned}$$

The Consensus Theorem

- **Consensus Theorem :** $XY + X'Z + YZ = XY + X'Z$

- **Proof :**
$$\begin{aligned} XY + X'Z + YZ &= XY + X'Z + (X + X')YZ \\ &= (XY + XYZ) + (X'Z + X'YZ) \\ &= XY(1+Z) + X'Z(1+Y) = XY + X'Z \end{aligned}$$

- **Example :**
$$a'b' + ac + bc' + \underset{\uparrow\ \text{consensus}}{b'c} + \overset{\text{consensus}\ \downarrow}{ab} = a'b' + ac + bc'$$

- **Dual form of consensus theorem :**
$$(X+Y)(X'+Z)(Y+Z) = (X+Y)(X'+Z)$$

- **Example :**
$$(a+b+c')\overset{\downarrow}{(a+b+d')}(b+c+d') = (a+b+c')(b+c+d')$$

The Consensus Theorem (Cont'd)

- **Example: eliminate BCD**

$$A'C'D + A'BD + \cancel{BCD} + ABC + ACD'$$

- **Example: eliminate A'BD, ABC**

$$A'C'D + \cancel{A'BD} + BCD + \cancel{ABC} + ACD'$$

- **Example: Reducing an expression by adding a term**

$$F = ABCD + B'CDE + A'B' + BCE'$$

$$F = ABCD + B'CDE + A'B' + BCE' + \boxed{ACDE}$$

Consensus term added

- **Final expression:**

$$F = A'B' + BCE' + ACDE$$

Algebraic Simplification of Switching Expressions

- **Combining terms:** $XY + XY' = X$

  Example: $abc'd' + abcd' = abd' \quad [X = abd', Y = c]$

- **Adding terms using X + X = X**

  $$ab'c + abc + a'bc = ab'c + abc + abc + a'bc = ac + bc$$

  Example: $(a+bc)(d+e') + a'(b'+c')(d+e') = d + e'$

  $[X = d + e', Y = a + bc, Y' = a'(b'+c')]$

- **Eliminating terms:** $X + XY = X$

  Example:

  $a'b + a'bc = a'b \quad [X = a'b]$

  $a'bc' + bcd + a'bd = a'bc' + bcd \quad [X = c, Y = bd, Z = a'b]$

Algebraic Simplification of Switching Expressions

- **Eliminating literals:** $X + X'Y = X + Y$

Example:

$$\begin{aligned} A'B + A'B'C'D' + ABCD' &= A'(B + B'C'D') + ABCD' \\ &= A'(B + C'D') + ABCD' \\ &= B(A' + ACD') + A'C'D' \\ &= B(A' + CD') + A'C'D' \\ &= A'B + BCD' + A'C'D' \end{aligned}$$

- **Adding redundant terms:**

Example:

$WX + XY + X'Z' + WY'Z'$ (add *WZ'* by consensus theorem)

$= WX + XY + X'Z' + WY'Z' + WZ'$ (eliminate *WY'Z'*)

$= WX + XY + X'Z' + WZ'$ (eliminate *WZ'*)

$= WX + XY + X'Z'$

Proving Validity of an Equation

Proving an equation valid

⇨ Construct a truth table and evaluate both sides
  - ⇨Tedious, not elegant method

⇨ Manipulate one side by applying theorems until it is the same as the other side

⇨ Reduce both sides of the equation independently

⇨ Apply same operation in both sides if the operation is reversible
  - ⇨Complement both sides etc
  - ⇨ not permissible: add terms, multiply terms

Proving Validity of an Equation

Strategy to prove equation valid

1. First reduce both sides to SOP (or POS)
2. Compare the two sides of the equation to see how they differ
3. Then try to add terms to one side of the equation that are present on the other side
4. Finally, try to eliminate terms form one side that are not present on the other

Proving Validity of an Equation

- **Prove:**

$$A'BD' + BCD + ABC' + AB'D = BC'D' + AD + A'BC$$

$$= A'BD' + BCD + ABC' + AB'D + BC'D' + A'BC + ABD$$

(add consensus of *A'BD'* and *ABC'*)

(add consensus of *A'BD'* and *BCD*)

(add consensus of *BCD* and *ABC'*)

$$= AD + A'BD' + BCD + ABC' + BC'D' + A'BC = BC'D' + AD + A'BC$$

(eliminate consensus of *BC'D'* and *AD*)

(eliminate consensus of *AD* and *A'BC*)

(eliminate consensus of *BC'D'* and *A'BC*)

Proving Validity of an Equation

- **Some of Boolean Algebra are not true for ordinary algebra**

Example: If $x + y = x + z$, then $y = z$ **True in ordinary algebra**

$1 + 0 = 1 + 1$ but $0 \neq 1$ **Not True in Boolean algebra**

Example: If $xy = xz$, then $y = z$ **True in ordinary algebra**

Not True in Boolean algebra

Example: If $y = z$, then $x + \mathrm{y} = \mathrm{x} + \mathrm{z}$ **True in ordinary algebra**

If $y = z$, then $xy = \mathrm{xz}$ **True in Boolean algebra**