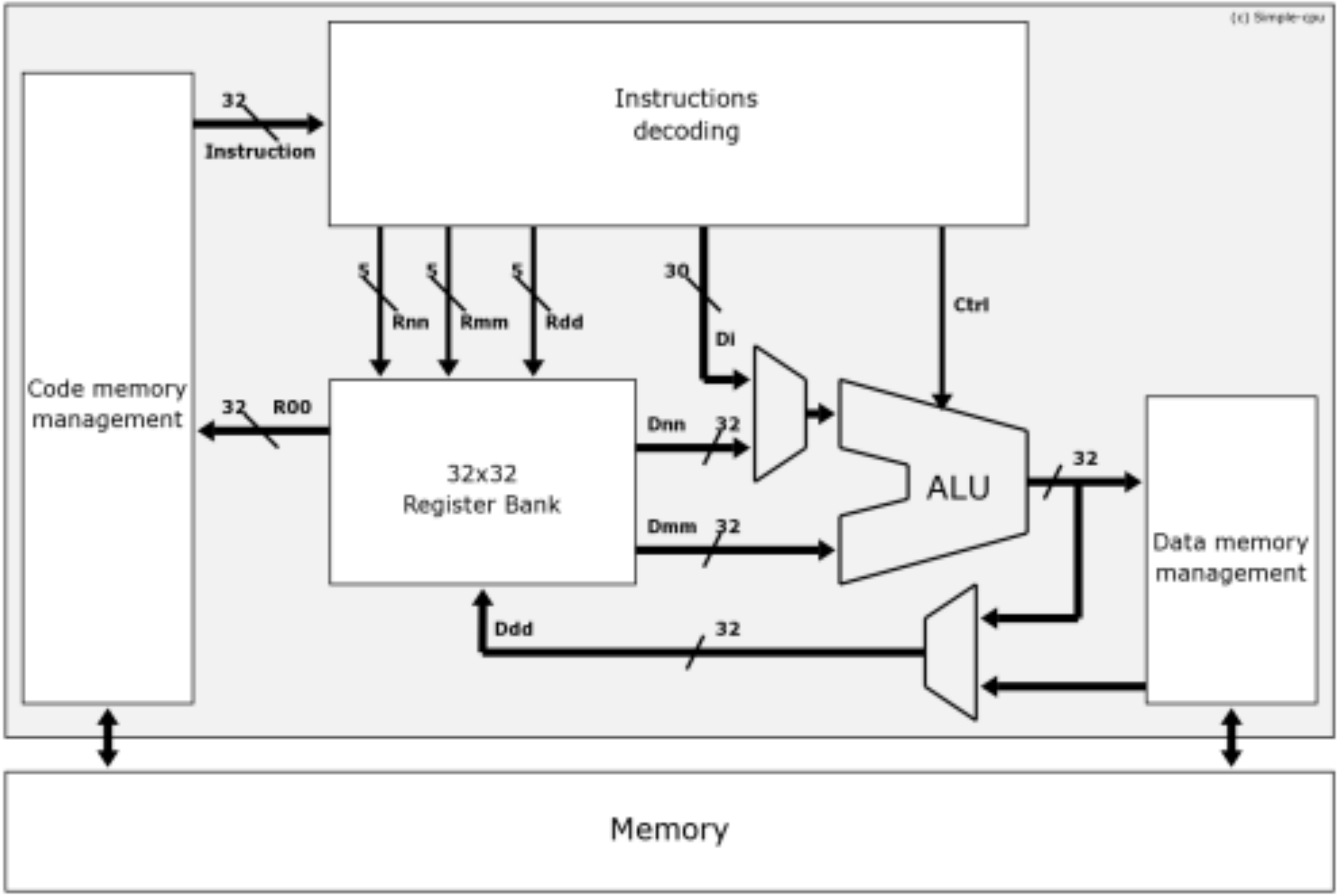
Digital Logic Design

Sung-Soo Lim

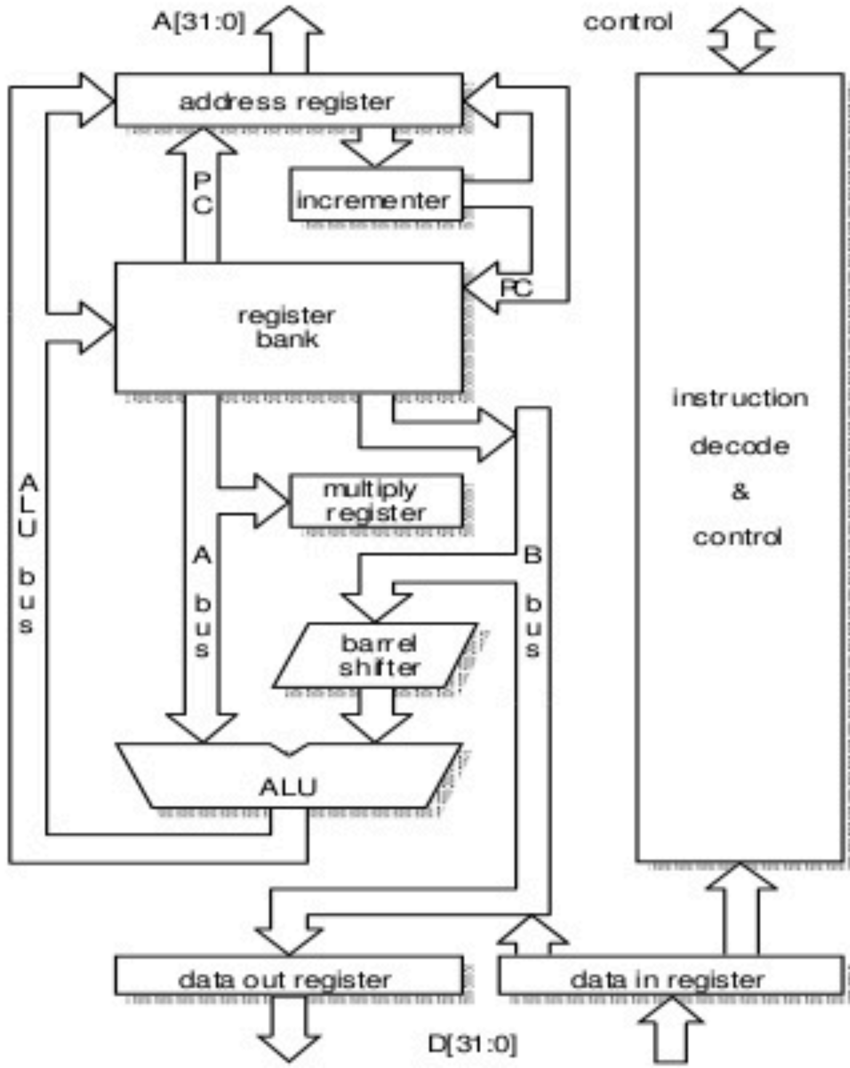
Kookmin Univ. Embedded System Lab

https://youtu.be/cNN_tTXABUA

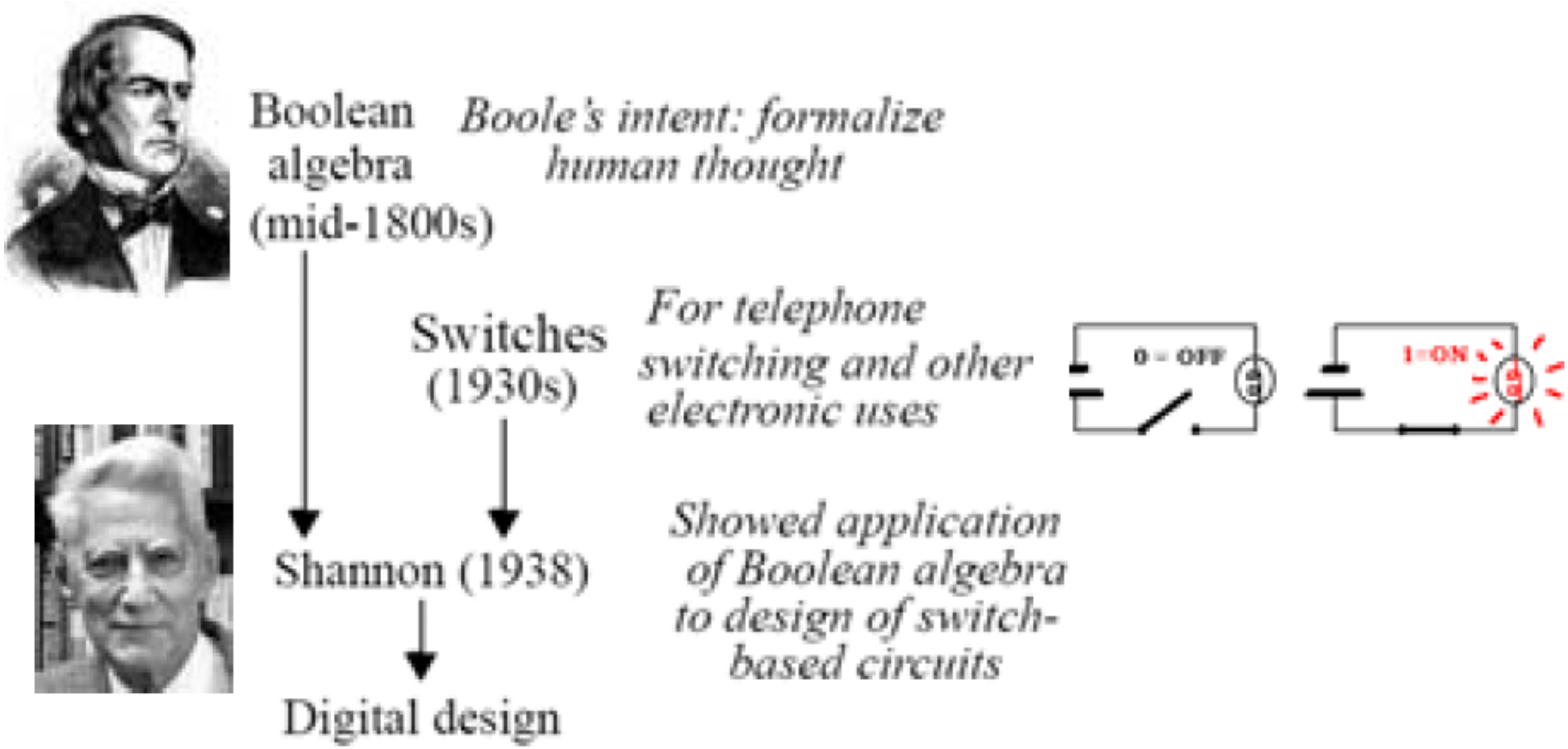


KESL
Kookmin Univ. Embedded Systems Lab

The ARM Architecture



Circuit Logic: Historical Perspective

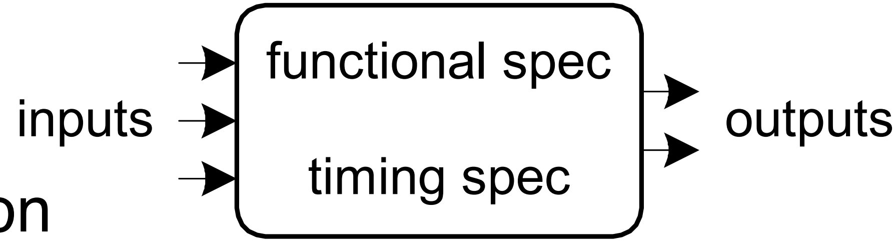


Boole 에 의해 디지털 논리의 기초가 만들어짐
Shannon이 실제 회로에 Boole의 이론을 적용함

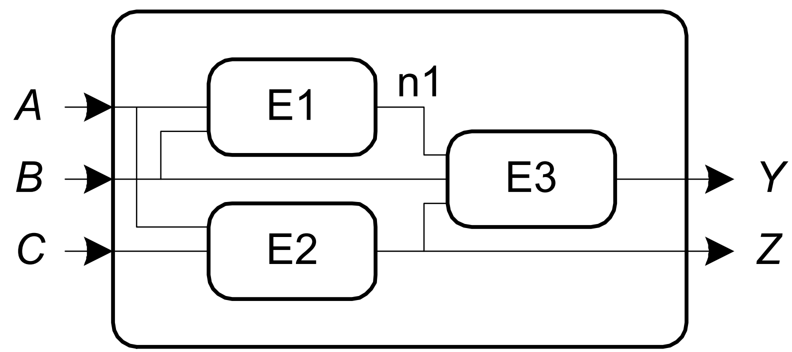
Logic Circuit Introduction

Logic Circuit의 구성

- Inputs
- Outputs
- Functional specification
- Timing specification



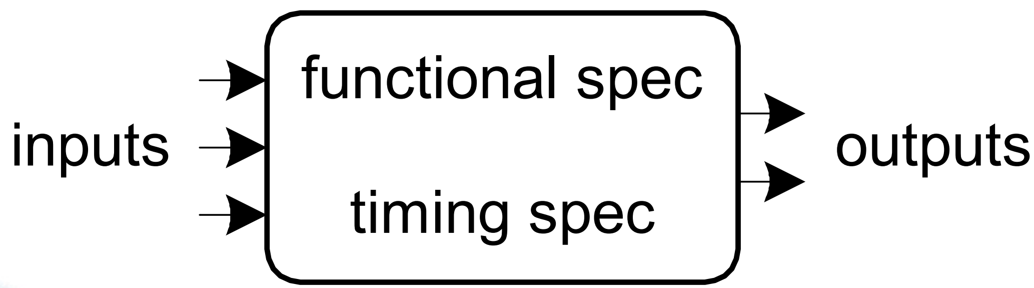
- Nodes
  - Inputs: *A*, *B*, *C*
  - Outputs: *Y*, *Z*
  - Internal: n1
- Circuit elements
  - E1, E2, E3
  - Each a circuit



KESL
Kookmin Univ. Embedded Systems Lab

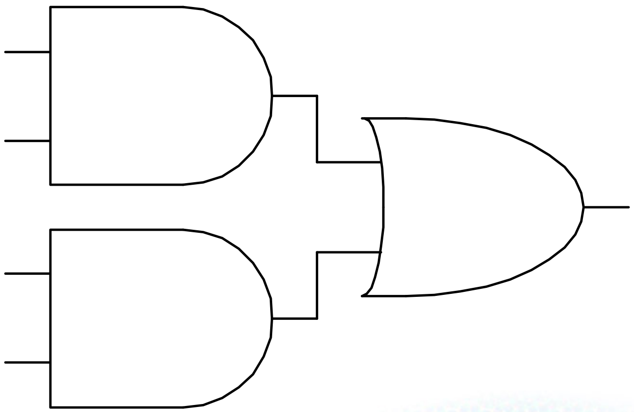
Types of Logic Circuits

- 조합논리회로 (**Combinational Logic)**
  - Memoryless
  - 출력은 현재 입력값에 의해서 정해짐
- 순차논리회로 (**Sequential Logic)**
  - Has memory
  - 출력은 현재 입력과 이전 입력(상태)에 의해 정해짐



조합논리회로의 특징

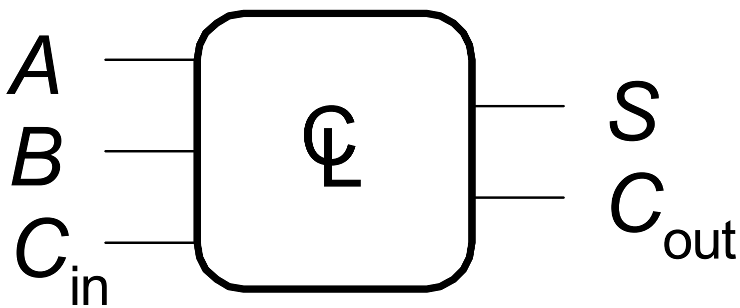
- 모든 구성요소가 조합논리회로이다
- 각 노드는 하나의 입력이 되거나 오직 하나의 출력에 연결된다
- 순환(Cycle)이 존재하지 않는다
- **Example:**



Boolean Logic

- 논리 함수의 기능을 입력 변수의 식으로 정의
- **Example:** $S = F(A, B, C_{in})$

$$C_{out} = F(A, B, C_{in})$$



$$S = A \oplus B \oplus C_{in}$$
$$C_{out} = AB + AC_{in} + BC_{in}$$

Some Definitions

- Complement: 변수의 NOT을 적용한 형태
  $\bar{A}, \bar{B}, \bar{C}$
- Literal: 변수 혹은 변수의 NOT 형태
  $A, \bar{A}, B, \bar{B}, C, \bar{C}$
- Implicant: literal의 곱 형태
  $ABC, AC, B\bar{C}$
- Minterm: 모든 변수가 한번씩 쓰인 곱의 형태 (변수 혹은 변수의 NOT)
  $ABC, A\bar{B}C, \bar{A}B\bar{C}$
- Maxterm: 모든 변수가 한번씩 쓰인 합의 형태
  $(A+B+C), (A+\bar{B}+C), (\bar{A}+B+\bar{C})$

Combinational Logic Design

- 일상 행동을 논리식으로 변환:
  - I'll (i) go to lunch if Mary (m) goes OR John (j) goes, AND Sally (s) does not go.

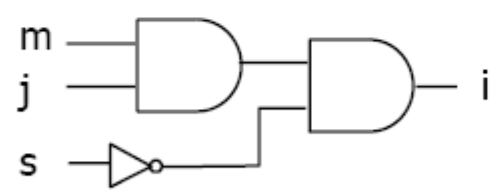
Which answer correctly represents the statement above:

A) $j = (i + m) \cdot (s)$

B) $i = (j) \cdot (m + !s)$

C) $i = (m + j) \cdot (!s)$

looks a lot like the circuits we were previously building

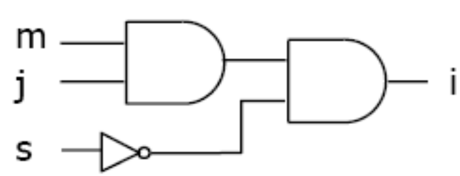


Combinational Logic Design

- 일상 행동을 논리식으로 변환:
  - I'll (i) go to lunch if Mary (m) goes OR John (j) goes, AND Sally (s) does not go.

> Will you go to lunch if:
> $m = 1, j = 0$, and $s = 1$?
> A) No ($i = 0$)
> B) Yes ($i = 1$)

evaluation process is also the same



m	j	s	i
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Boolean Equations Example

- You are going to the cafeteria for lunch
  – You won't eat lunch (E)
  – If it's not open (O) or
  – If they only serve corndogs (C)
- (E)에 대한 진리표.

O	C	E
0	0	0
0	1	0
1	0	1
1	1	0

KESL
Kookmin Univ. Embedded Systems Lab

Sum-of-Products (SOP) Form

- 모든 식은 SOP form 으로 표현 가능하다
- 각 행은 **minterm이다**
- 각 minterm은 literal의 곱 (AND) 으로 표현된다
- 각 열에 표현된 minterm은 TRUE이다
- 각 행에 표현된 minterm을 모두 OR하면 TRUE이다
- 그래서, a sum (OR) of products (AND terms) 이라 부른다

A	*B*	*Y*	minterm	minterm name
0	0	0	$\overline{A}\ \overline{B}$	m_0
0	1	1	$\overline{A}\ B$	m_1
1	0	0	$A\ \overline{B}$	m_2
1	1	1	$A\ B$	m_3

$$Y = \mathrm{F}(A, B) =$$

KESL
Kookmin Univ. Embedded Systems Lab

Sum-of-Products (SOP) Form

- 모든 식은 SOP form 으로 표현 가능하다
- 각 행은 **minterm이다**
- 각 minterm은 literal의 곱 (AND) 으로 표현된다
- 각 열에 표현된 minterm은 TRUE이다
- 각 행에 표현된 minterm을 모두 OR하면 TRUE이다
- 그래서, a sum (OR) of products (AND terms) 이라 부른다

A	*B*	*Y*	minterm	minterm name
0	0	0	$\overline{A}\ \overline{B}$	m_0
0	1	1	$\overline{A}\ B$	m_1
1	0	0	$A\ \overline{B}$	m_2
1	1	1	$A\ B$	m_3

$$Y = \mathrm{F}(A, B) =$$

Sum-of-Products (SOP) Form

- 모든 식은 SOP form 으로 표현 가능하다
- 각 행은 **minterm이다**
- 각 minterm은 literal의 곱 (AND) 으로 표현된다
- 각 열에 표현된 minterm은 TRUE이다
- 각 행에 표현된 minterm을 모두 OR하면 TRUE이다
- 그래서, a sum (OR) of products (AND terms) 이라 부른다

A	B	Y	minterm	minterm name
0	0	0	$\overline{A}\ \overline{B}$	m_0
0	1	1	$\overline{A}\ B$	m_1
1	0	0	$A\ \overline{B}$	m_2
1	1	1	$A\ B$	m_3

$$Y = \mathrm{F}(A, B) = \overline{\mathrm{A}}\mathrm{B} + \mathrm{AB} = \Sigma(1, 3)$$

KESL
Kookmin Univ. Embedded Systems Lab

Product-of-Sums (POS) Form

- 모든 Boolean 논리식은 POS form으로 표현할 수 있다
- 각 행은 **maxterm으로 이루어진다**
- maxterm은 literal의 합(OR)으로 구성된다
- 각 행에 표현된 maxterm은 그 행에서 FALSE 이다
- 모든 행의 maxterm을 곱(AND)한 식 역시 FALSE이다
- 그래서, a product (AND) of sums (OR terms) 이라 부른다

A	B	Y	maxterm	maxterm name
0	0	0	$A + B$	M_0
0	1	1	$A + \overline{B}$	M_1
1	0	0	$\overline{A} + B$	M_2
1	1	1	$\overline{A} + \overline{B}$	M_3

$$Y = \mathrm{F}(A, B) = (A + B)(\overline{A} + B) = \Pi(0, 2)$$

KESL
Kookmin Univ. Embedded Systems Lab

SOP & POS Form

- SOP – sum-of-products

O	C	E	minterm
0	0	0	$\overline{O}\ \overline{C}$
0	1	0	$\overline{O}\ C$
1	0	1	$O\ \overline{C}$
1	1	0	$O\ C$

$$E = O\overline{C}$$
$$= \Sigma(2)$$

- POS – product-of-sums

O	C	E	maxterm
0	0	0	$O + C$
0	1	0	$O + \overline{C}$
1	0	1	$\overline{O} + C$
1	1	0	$\overline{O} + \overline{C}$

$$E = (O + C)(O + \overline{C})(\overline{O} + \overline{C})$$
$$= \Pi(0, 1, 3)$$

How to do Logic Mathematically

Axioms

Theorems

Properties

Mathematical Induction

Algebraic Manipulations

KESL
Kookmin Univ. Embedded Systems Lab

Boolean Algebra

- Axioms과 theorems은 Boolean 논리식을 간단화(**simplify)**하기 위해 사용한다
- 변수는 오직 0과 1 중 하나의 값을 가진다
- **Duality** in axioms and theorems:
  - ANDs 와 ORs, 0's 과 1's 을 서로 바꿀 수 있다

Boolean Axioms

	Axiom		Dual	Name
A1	$B = 0$ if $B \neq 1$	A1′	$B = 1$ if $B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2′	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	A3′	$1 + 1 = 1$	AND/OR
A4	$1 \bullet 1 = 1$	A4′	$0 + 0 = 0$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	A5′	$1 + 0 = 0 + 1 = 1$	AND/OR

	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1′	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	T2′	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	T3′	$B + B = B$	Idempotency
T4		$\overline{\overline{B}} = B$		Involution
T5	$B \bullet \overline{B} = 0$	T5′	$B + \overline{B} = 1$	Complements

Boolean Theorems

	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1′	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	T2′	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	T3′	$B + B = B$	Idempotency
T4		$\bar{\bar{B}} = B$		Involution
T5	$B \bullet \bar{B} = 0$	T5′	$B + \bar{B} = 1$	Complements

Boolean Theorems of Several Vars

	Theorem		Dual	Name
T6	$B \bullet C = C \bullet B$	T6′	$B + C = C + B$	Commutativity
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	T7′	$(B + C) + D = B + (C + D)$	Associativity
T8	$(B \bullet C) + (B \bullet D) = B \bullet (C + D)$	T8′	$(B + C) \bullet (B + D) = B + (C \bullet D)$	Distributivity
T9	$B \bullet (B + C) = B$	T9′	$B + (B \bullet C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	T10′	$(B + C) \bullet (B + \overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D)$ $= B \bullet C + \overline{B} \bullet D$	T11′	$(B + C) \bullet (\overline{B} + D) \bullet (C + D)$ $= (B + C) \bullet (\overline{B} + D)$	Consensus
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots}$ $= (\overline{B_0} + \overline{B_1} + \overline{B_2} \ldots)$	T12′	$\overline{B_0 + B_1 + B_2 \ldots}$ $= (\overline{B_0} \bullet \overline{B_1} \bullet \overline{B_2})$	De Morgan's Theorem

KESL
Kookmin Univ. Embedded Systems Lab

Boolean Algebra: Axioms

- Boolean Algebra의 가장 기초적인 원칙

1a. $0 \cdot 0 = 0$
1b. $1 + 1 = 1$
2a. $1 \cdot 1 = 1$
2b. $0 + 0 = 0$
3a. $0 \cdot 1 = 1 \cdot 0 = 0$
3b. $0 + 1 = 1 + 0 = 1$
4a. If $x = 0$, then $x' = 1$
4b. If $x = 1$, then $x' = 0$

AND

a	b	F
0	0	0
0	1	0
1	0	0
1	1	1

OR

a	b	F
0	0	0
0	1	1
1	0	1
1	1	1

NOT

a	F
0	1
1	0

Boolean Algebra: Theorems

AND

a	x	F
0	0	0
0	1	0
1	0	0
1	1	1

OR

a	x	F
0	0	0
0	1	1
1	0	1
1	1	1

AND

a	x	F
0	0	0
0	1	0
1	0	0
1	1	1

AND

a	x	F
0	0	0
0	1	0
1	0	0
1	1	1

5a. $x \cdot 0 = 0$ *(Null Elements)*
5b. $x + 1 = 1$
6a. $x \cdot 1 = x$ *(Identity)*
6b. $x + 0 = x$
7a. $x \cdot x = x$ *(Idempotent)*
7b. $x + x = x$
8a. $x \cdot x' = 0$ *(Complement)*
8b. $x + x' = 1$
9. $x'' = x$ *(Involution)*

x is a variable

Boolean Algebra: Duality

- Principle of duality
  - 주어진 논리식에 대해 모든 AND를 OR연산으로(혹은 그 반대), 모든 1을 0으로(혹은 그 반대) 바꾼 논리식을 Dual이라고 한다
  - TRUE인 논리식의 Dual식은 역시 TRUE이다

1a. $0 \cdot 0 = 0$
1b. $1 + 1 = 1$
2a. $1 \cdot 1 = 1$
2b. $0 + 0 = 0$
3a. $0 \cdot 1 = 1 \cdot 0 = 0$
3b. $0 + 1 = 1 + 0 = 1$
4a. If $x = 0$, then $x' = 1$
4b. If $x = 1$, then $x' = 0$

5a. $x \cdot 0 = 0$
5b. $x + 1 = 1$
6a. $x \cdot 1 = x$
6b. $x + 0 = x$
7a. $x \cdot x = x$
7b. $x + x = x$
8a. $x \cdot x' = 0$
8b. $x + x' = 1$
9. $x'' = x$

Boolean Algebra: Properties

- 주요 Properties

10a.	$x \cdot y = y \cdot x$	*(Commutative)*
10b.	$x + y = y + x$	
11a.	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$	*(Associative)*
11b.	$x + (y + z) = (x + y) + z$	
12a.	$x \cdot (y + z) = x \cdot y + x \cdot z$	*(Distributive)*
12b.	$x + (y \cdot z) = (x + y) \cdot (x + z)$	*this one is tricky!*
13a.	$x + x \cdot y = x$	*(Absorption)*
13b.	$x \cdot (x + y) = x$	
14a.	$x \cdot y + x \cdot y' = x$	*(Combining)*
14b.	$(x + y) \cdot (x + y') = x$	
15a.	$(x \cdot y)' = x' + y'$	*(DeMorgan's Theorem)*
15b.	$(x + y)' = x' \cdot y'$	
16a.	$x + x' \cdot y = x + y$	
16b.	$x \cdot (x' + y) = x \cdot y$	

Boolean Algebra: Induction

- Property들을 증명하는 방법
  - Truth table
  - Algebraic Manipulation
- DeMorgan's Theorem 을 진리표로 증명하기
  - $(x \cdot y)' = x' + y'$

		LHS		RHS		
x	y	$(x \cdot y)$	$(x \cdot y)'$	x'	y'	$x' + y'$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

Truth tables produce same result – functions are equivalent!

Algebraic Manipulation

- $(x1 + x3) \cdot (x1' + x3') = x1 \cdot x3' + x1' \cdot x3$ 을 algebraic manipulation으로 증명하기

LHS = $(x1 + x3) \cdot (x1' + x3')$

Use Distribuitve property 12a - $x \cdot (y + z) = x \cdot y + x \cdot z$

LHS = $(x1 + x3) \cdot x1' + (x1 + x3) \cdot x3'$

Use Distribuitve property again

LHS = $(x1 \cdot x1') + (x3 \cdot x1') + (x1 \cdot x3') + (x3 \cdot x3')$

Use complement property 8a - $x \cdot x' = 0$

LHS = $0 + (x3 \cdot x1') + (x1 \cdot x3') + 0$

Use Identity 6b - $x + 0 = x$

LHS = $(x3 \cdot x1') + (x1 \cdot x3')$

Use commutative property 10a - $x \cdot y = y \cdot x$ and 10b - $x + y = y + x$

LHS = $(x1 \cdot x3') + (x1' \cdot x3)$

LHS matches RHS of the inital equation

Algebraic Manipulation

- $(a \cdot b \cdot c) + (a \cdot b \cdot c') = a \cdot b$?

$$LHS = (a \cdot b \cdot c) + (a \cdot b \cdot c')$$

Distribuitve property 12a - $x \cdot (y + z) = x \cdot y + x \cdot z$

$$(\underbrace{a \cdot b}_{x} \cdot \underbrace{c}_{y}) + (\underbrace{a \cdot b}_{x} \cdot \underbrace{c'}_{z}) = (\underbrace{a \cdot b}_{x}) \cdot \underbrace{(c + c')}_{(y+z)}$$

$$LHS = (a \cdot b) \cdot (c + c')$$

Complement proprety 8b - $x + x' = 1$

$$LHS = (a \cdot b) \cdot (1)$$

Identity property 6a - $x \cdot 1 = x$

$$LHS = a \cdot b$$

Algebraic Manipulation

- $x + (x' \cdot z)$ is equivalent to $x + z$?

LHS = $x + (x' \cdot z)$

Distribuitve property 12b - $x + (y \cdot z) = (x + y) \cdot (x + z)$

LHS = $(x + x') \cdot (x + z)$

Complement proprety 8b - $x + x' = 1$

LHS = $(1) \cdot (x + z)$

Identity property 6a - $x \cdot 1 = x$

LHS = $(x + z)$

Algebraic Manipulation

- $(x \cdot x') + (x \cdot y) \cdot (x' + y')$ 는 항상 1일까요?

LHS = $(x \cdot x') + (x \cdot y) \cdot (x' + y')$

Complement proprety 8a - $x \cdot x' = 0$

Identity property 6a - $x + 0 = x$

LHS = $(x \cdot y) \cdot (x' + y')$

Distribuitve property 12a – $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$

LHS = $(x \cdot y \cdot x') + (x \cdot y \cdot y')$

Use complement property 8a - $x \cdot x' = 0$

LHS = $(y \cdot 0) + (x \cdot 0)$

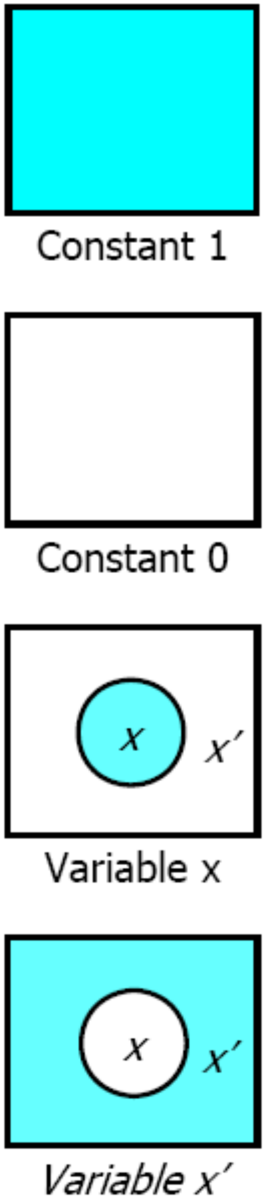
Use Null Elements 5a - $x \cdot 0 = 0$

LHS = $0 + 0$

Axiom 2b – $0 + 0 = 0$

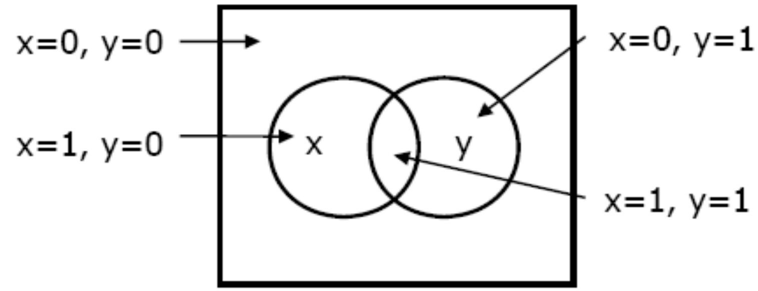
Venn Diagram

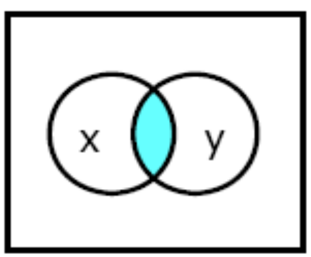
- 논리식을 증명하기 위해 사용 가능한 것들
  - Truth tables (perfect induction)
  - Algebraic manipulation
- 그래픽 표현 방법을 이용한 방법: Venn Diagram
  - Universe represented by a square
    - Boolean algebra has only two values
    - Universe B = {0, 1}
  - Elements of a set are enclosed by a contour
    - Square, circle, ellipse, …



Venn Diagram: 두 변수 표현

- 두 변수 x와 y의 표현
  - 겹치는 영역은 x = y = 1

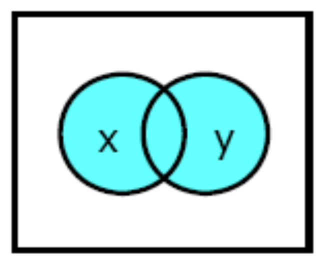




x · y

also referred to as ***intersection*** *of x and y*

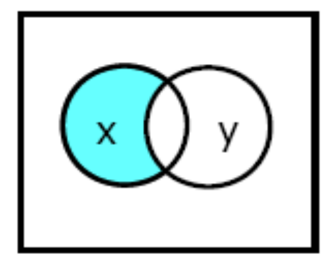
x	y	F
0	0	0
0	1	0
1	0	0
1	1	1



x + y

also referred to as ***union*** *of x and y*

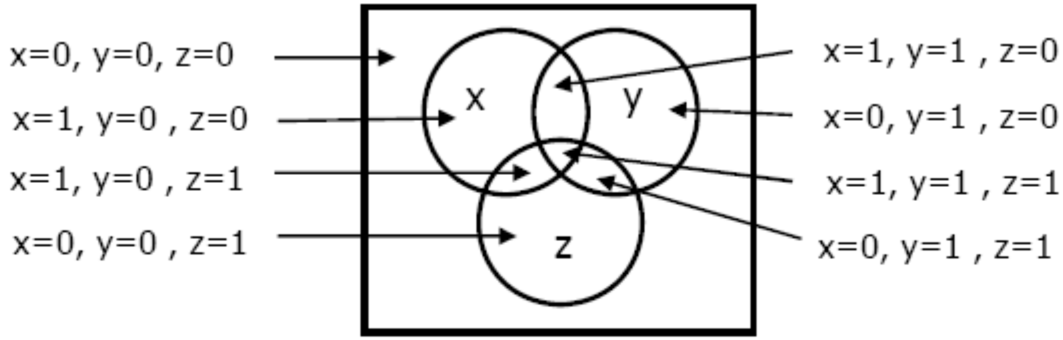
x	y	F
0	0	0
0	1	1
1	0	1
1	1	1

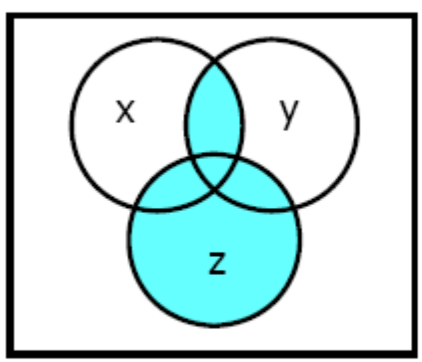


x · y′

x	y	F
0	0	0
0	1	0
1	0	1
1	1	0

Venn Diagram: 세 변수 표현





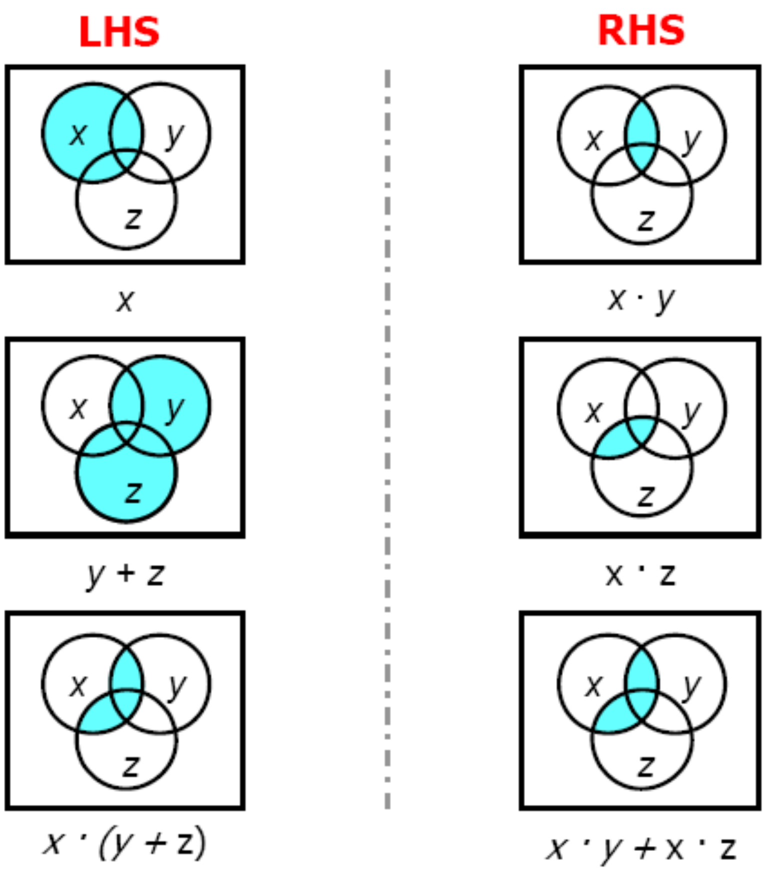
x · y + z

intersection *of x and y,*

then ***union*** *of that intersection with z*

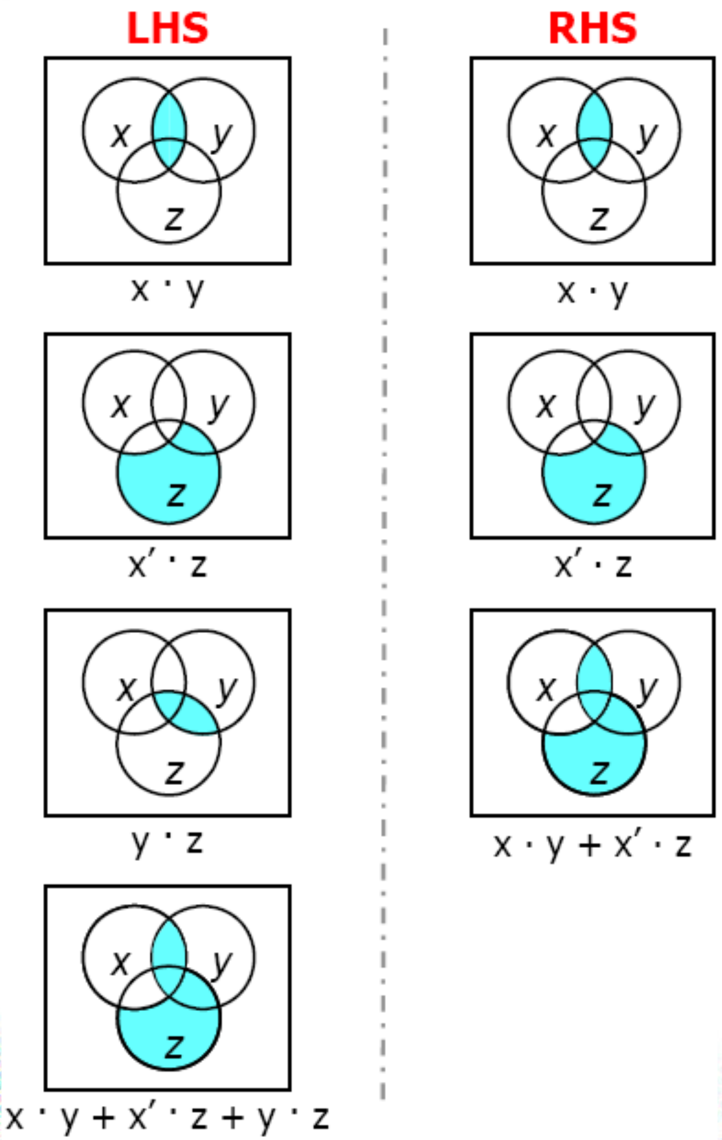
Venn Diagram: Equivalence of Logic Expressions

- Distribution property $x \cdot (y + z) = x \cdot y + x \cdot z$



Venn Diagram: Equivalence of Logic Expressions

- $x \cdot y + x' \cdot z + y \cdot z = x \cdot y + x' \cdot z$



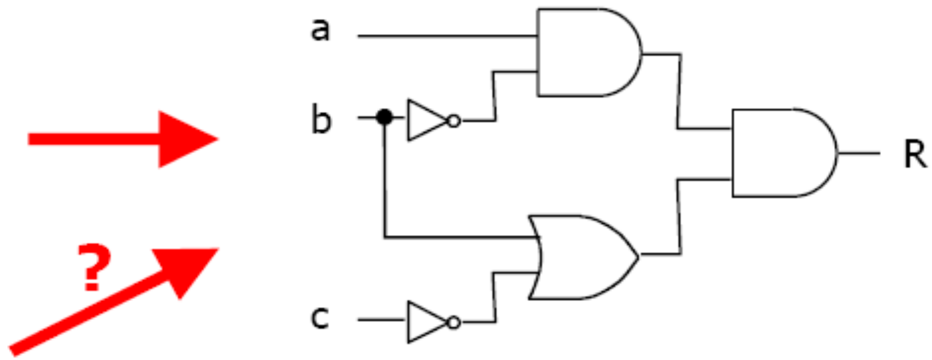
Precedence of Operations

Symbol	Name	Description
( )	Parentheses	Evaluate expression nested in parentheses first
′	NOT	Evaluate from left to right
·	AND	Evaluate from left to right
+	OR	Evaluate from left to right

Synthesis Using AND, OR, and NOT Gates

- 논리 표현식의 예
  – R = (a · b’) · (b + c’)
- 진리표에서 논리회로를 도출하는 방법?

$R = (a \cdot b') \cdot (b + c')$



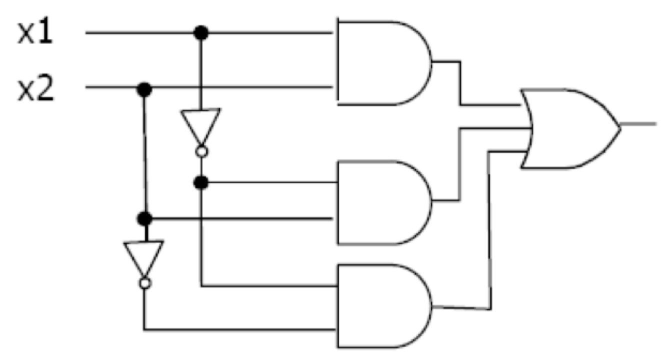
a	b	c	R
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Synthesis Using AND, OR, and NOT Gates

- 다음 기능을 수행하는 논리식을 논리회로로 구성
  - Two inputs x1 and x2
    - Produce output =1 if x1 = 0/x2 = 0, or x1 = 0/x2 = 1, or x1 = 1/x2 = 1
    - Produce output = 0 if x1 = 1/x2 = 0

x1	x2	F
0	0	1
0	1	1
1	0	0
1	1	1

$$F = x1'x2' + x1'x2 + x1x2$$



Synthesis Using AND, OR, and NOT Gates

- 논리 회로 도출의 대원칙
  - 더 최적화 가능한 논리식 도출이 가능한지 확인한다
- 더 간단한 논리회로는 더 작은 비용으로 구현 가능하다 (대부분의 경우 더 빠르다)