



# 1. 디지털 논리회로

## 목표

---

- 논리 회로 기본 구성 요소의 기능을 이해하고 활용함.
- 특정 기능 수행을 위한 논리 회로를 설계함.



---

# 디지털 컴퓨터



---

## 디지털 컴퓨터

- 소프트웨어
  - 응용 소프트웨어
  - 시스템 소프트웨어
- 하드웨어
  - 전자적 구성 요소 및 그 연결



# 디지털

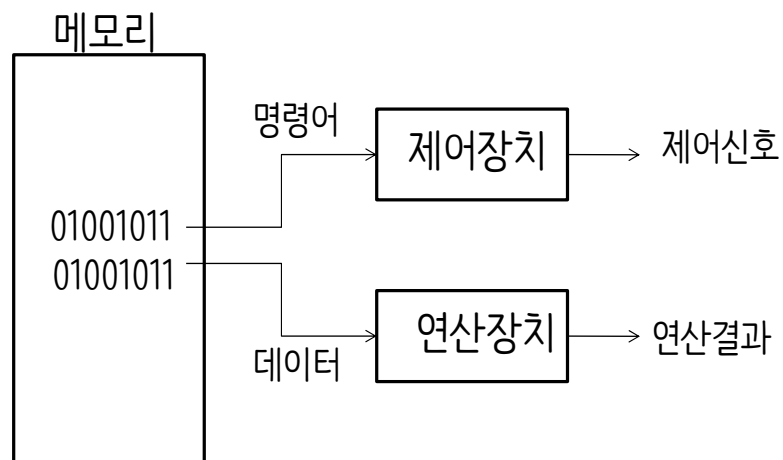
---

- 제한 수의 불연속적인 값으로 표현
  - 명령어
  - 데이터
- 이진수, binary number
  - 0, 1의 불연속적인 값 이용
  - Bit, binary digit



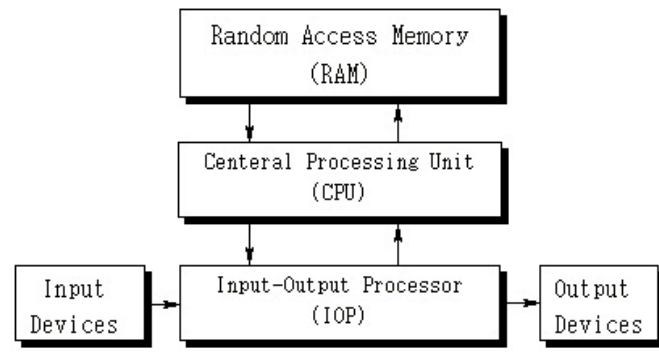
## 디지털 데이터 예

---



## 디지털 컴퓨터 구성도

---



## 논리 게이트











# 게이트, gate

- 이진 정보 처리
  - 전압 신호 이용
  - 예 :
    - 3v : 논리 1
    - 0v : 논리 0
- 입력 신호에 따라 출력 신호 결정
- 진리(치)표, truth table



## 디지털 논리 게이트

명칭	그래픽 기호	함수식	진리치표	명칭	그래픽 기호	함수식	진리치표																														
AND		$X = AB$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1	NAND		$X = (AB)'$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	1	0	1	1	1	0	1	1	1	0
A	B	X																																			
0	0	0																																			
0	1	0																																			
1	0	0																																			
1	1	1																																			
A	B	X																																			
0	0	1																																			
0	1	1																																			
1	0	1																																			
1	1	0																																			
OR		$X = A+B$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1	NOR		$X = (A+B)'$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	0
A	B	X																																			
0	0	0																																			
0	1	1																																			
1	0	1																																			
1	1	1																																			
A	B	X																																			
0	0	1																																			
0	1	0																																			
1	0	0																																			
1	1	0																																			
Inverter		$X = A'$	<table><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0	XOR		$X = (A \oplus B)$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	0									
A	X																																				
0	1																																				
1	0																																				
A	B	X																																			
0	0	0																																			
0	1	1																																			
1	0	1																																			
1	1	0																																			
Buffer		$X = A$	<table><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	A	X	0	0	1	1	XNOR		$X = (A \odot B)$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	1									
A	X																																				
0	0																																				
1	1																																				
A	B	X																																			
0	0	1																																			
0	1	0																																			
1	0	0																																			
1	1	1																																			



---

## 부울 대수



---

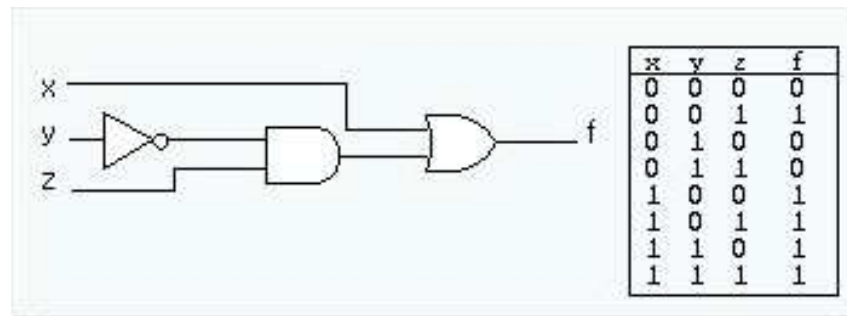
## 부울대수

- 이진 변수 및 논리적인 연산 표현
- 이진 변수
  - 논리-0 또는 논리-1 값을 갖는 변수
  - 예 : A, B, x, y 등
- 논리적인 연산
  - AND, OR, NOT, XOR, XNOR



## 부울식 = 진리표 = 논리도

- $F = x + y'z$
- 진리표 및 논리도



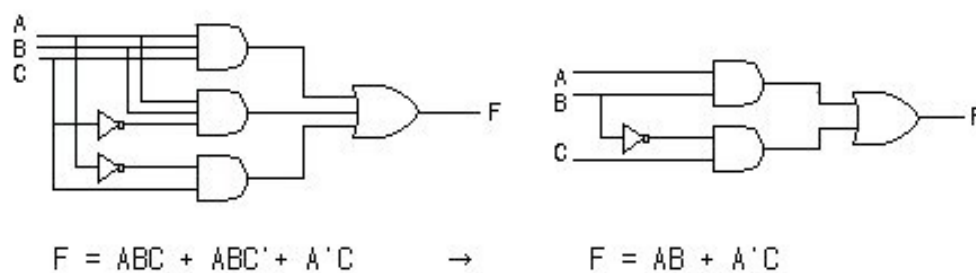
## 부울대수

<표 1-1> 부울 대수의 기본적 관계

(1) $x + 0 = x$	(2) $x \cdot 0 = 0$
(3) $x + 1 = 1$	(4) $x \cdot 1 = x$
(5) $x + x = x$	(6) $x \cdot x = x$
(7) $x + x' = 1$	(8) $x \cdot x' = 0$
(9) $x + y = y + x$	(10) $xy = yx$
(11) $x + (y + z) = (x + y) + z$	(12) $x(yz) = (xy)z$
(13) $x(y + z) = xy + xz$	(14) $x + yz = (x + y)(x + z)$
(15) $(x + y)' = x' y'$	(16) $(xy)' = x' + y'$
(17) $(x')' = x$	

## 같은 기능 다른 부울식/블록도

---



---

맵의 간소화, SIMPLIFICATION





## 같은 기능

---

- 동일 입력 조합 → 동일 출력값
- 진리표가 동일



## 간략화, Simplify

---

- 간소화
- 동일 기능의 더 간단한 부울식/블록도
- 방법
  - 부울 대수의 기본 관계 이용
  - 맵 이용



## 부울 대수의 기본 관계 이용 간략화

$$\begin{aligned}F &= ABC + ABC' + A'C \\&= AB(C + C') + A'C \\&= AB + A'C\end{aligned}$$

$$\begin{aligned}F &= ABC + AB\bar{C} + \bar{A}C \\&= AB(C + \bar{C}) + \bar{A}C \\&= AB + \bar{A}C\end{aligned}$$



## 민텀항

- 진리표 상의 각 입력 조합
- N개의 이진 변수 :  $2^n$ 개의 민텀항

x	y	z	f
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

→  $x'y'z'$

→  $x'y'z$

→  $xy'z$

→  $xyz$



## 민텀항에 대한 맵

		B	
		0	1
A	0	0	1
	1	2	3

(a) Two-variable map

		BC			
		00	01	11	10
A	0	0	1	3	2
	1	4	5	7	6

(b) Three-variable map

		CD			
		00	01	11	10
AB	00	0	1	3	2
	01	4	5	7	6
	11	12	13	15	14
	10	8	9	11	10

(c) Four-variable map

## 맵 이용 간략화

- Karnaugh 맵 이용
- 진리표 민텀(minterm)항에 대응하는 맵을 작성
- 출력이 1이 되는 민텀 표시

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

→

A	BC			
	00	01	11	10
0		1		
1	1	1	1	1

$$f(x, y, z) = \sum (1, 4, 5, 6, 7)$$

$$= x'y'z + xy'z' + xy'z + xyz' + xyz$$

## 맵이용 간략화

- 값이 1인 인접한 민텀 그룹화
  - 가능한 한 크게 2, 4, 8, ... 개씩
  - 중복 가능
- 각 민텀 그룹의 부울식 표시
- 부울식을 논리합(OR)로 묶음.



## 맵이용 간략화 예

$$F(A, B, C) = \sum (0, 2, 4, 5, 6)$$

		BC			
		00	01	11	10
A	0	1			1
	1	1	1		1

Red lines and arrows indicate groupings: a horizontal line for  $AB'$  (covering cells 00 and 10 in row 0) and a vertical line for  $C'$  (covering cells 00 and 01 in column 1). A black circle highlights the cells 01 and 11 in row 1.

$$F = AB' + C'$$

$$\begin{aligned} & AB'C' + AB'C \\ &= AB'(C' + C) \\ &= AB'(1) \\ &= AB' \end{aligned}$$

$$\begin{aligned} & A'B'C' + AB'C' + A'BC' + ABC' \\ &= (A' + A)B'C' + (A' + A)BC' \\ &= B'C' + BC' \\ &= (B' + B)C' \\ &= C' \end{aligned}$$



## 논리곱의 논리합

---

- Sum of Product
  - Sum 항 : OR 항
  - Product 항 : AND 항
- NAND 게이트 이용 구현이 용이



## 논리합의 논리곱

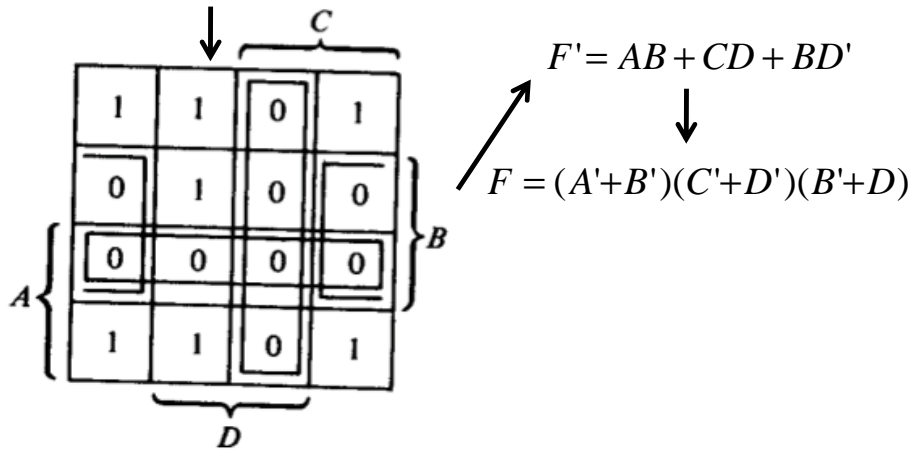
---

- Product of Sum
- NOR 게이트 이용한 구현이 용이
- 맵에서 민텀의 값이 0이 되는 부분이용 간략화 : F'구함
- 위의 간략화된 결과를 이용 F를 구함



## 논리합의 논리곱

$$F(A, B, C, D) = \sum (0, 1, 2, 5, 8, 9, 10)$$



## Don't Care

- 민텀의 값이 결과에 영향을 주지 않음
- X로 표시
- 간략화 예 :

$$F(A, B, C) = \sum (0, 2, 6)$$

$$d(A, B, C) = \sum (1, 3, 5)$$

	<b>BC</b>		
<b>A</b>	1	X	X
	X	X	1



---

## 조합논리회로



---

## 논리회로

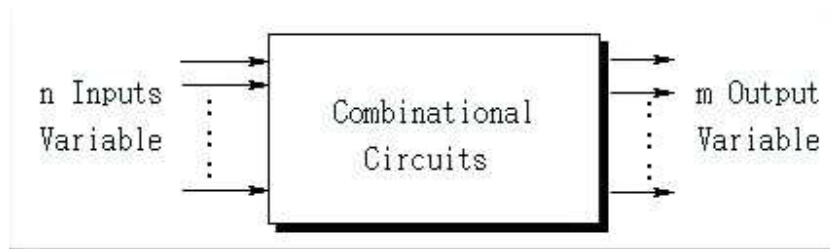
- 부울 대수를 물리적 장치로 구현
  - 입력에 대해 논리 연산 수행
- 종류
  - 조합논리회로, Combination Logic Circuit
  - 순차(서)논리회로, Sequential Logic Circuit



## 조합논리회로

---

- 출력은 **입력 조합의 의해 결정**
- 시간, 상태 개념 없음
- $n$ 개의 입력/  $m$ 개의 출력
  - $m$ 개의 부울 함수
  - 각 출력은 상호 독립적
  - 각 출력은 각각 간략화



## 조합논리회로 설계

---

- 기능 정의/제시
- 입력/출력의 수 및 문자 기호 결정
- 진리표 작성
- 간략화 과정 수행
- 블록도 작성





## 조합논리회로설계 - 반가산기

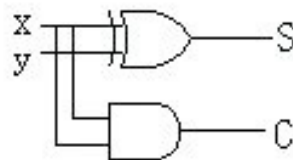
- Half Adder, HA
- 두 비트 입력을 덧셈하여 두 비트(캐리, 합) 출력
- 입력 수 및 문자기호 결정
  - 세 비트 입력 :  $x, y$
  - 두 비트 출력 :  $C, S$



## 조합논리회로설계 - 반가산기

(a) <Truth table>    (b) Block Diagram    논리함수식

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



$$S = x'y + xy' = x \oplus y$$
$$C = xy$$



## 조합논리회로설계 - 전가산기

- Full Adder, FA
- 세 비트 입력을 덧셈하여 두 비트(캐리, 합) 출력
- 입력 수 및 문자기호 결정
  - 세 비트 입력 :  $x, y, z$
  - 두 비트 출력 :  $S, C$

### 진리표 작성

입력			출력	
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



## 조합논리회로설계 - 전가산기

- 간략화를 위한 맵 작성

**S**

	1		1
1		1	

$$S = x'y'z + x'yz' + xy'z' + xyz$$

$$= x \oplus y \oplus z$$

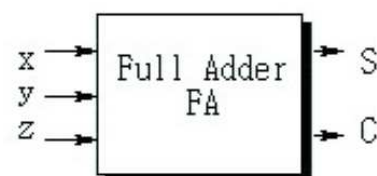
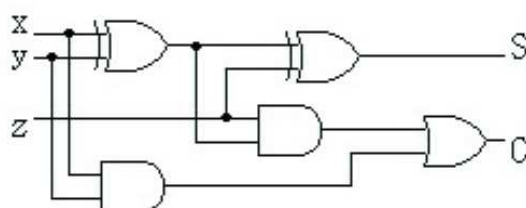
**C**

		1	
	1	1	1

$$C = xy + xz + yz$$

$$= xy + (x'y + xy')z$$

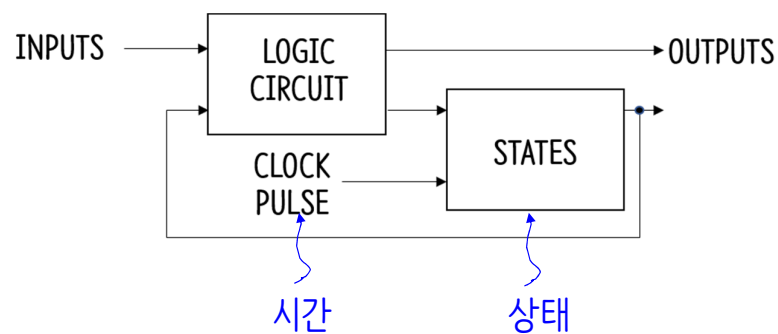
- 논리회로 작성



# 플립플롭

## 순차논리회로

- 시간/상태 개념 포함
- 출력과 상태 정보
- 입력 조합 + 현재 상태 → 출력
- 입력조합 + 현재 상태 → 다음 상태



## 상태, state

---

- 논리-0, 논리-1 상태
- 다음 상태 ← 현재상태, 외부입력
- 상태 저장 - 플립플롭



## 플립플롭

---

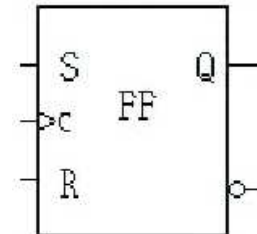
- 순차 논리 회로의 저장 요소
- 논리-0/논리-1 상태 저장
- 입력 - 1 또는 2 비트
- 출력 - 정상 출력/보수화된 출력
- 종류
  - SR 플립플롭
  - D 플립플로
  - T 플립플롭
  - JK 플립플롭



## SR 플립플롭

- Set/Reset 플립플롭
- 특성표

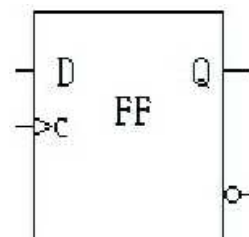
S	R	Q(t+1)	
0	0	Q(t)	No change
0	1	0	Clear to 0
1	0	1	Set to 1
1	1	?	-



## D 플립플롭

- Data 플립플롭
- 특성표

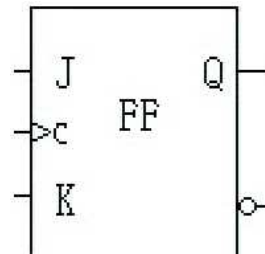
D	Q(t+1)	
0	0	Clear to 0
1	1	Set to 1



## JK 플립플롭

- Set(j)/Reset(K) 플립플롭
- 특성표

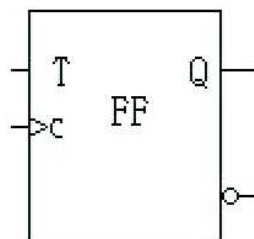
J	K	Q(t+1)	
0	0	Q(t)	No change
0	1	0	Clear to 0
1	0	1	Set to 1
1	1	Q'(t)	Complement



## T 플립플롭

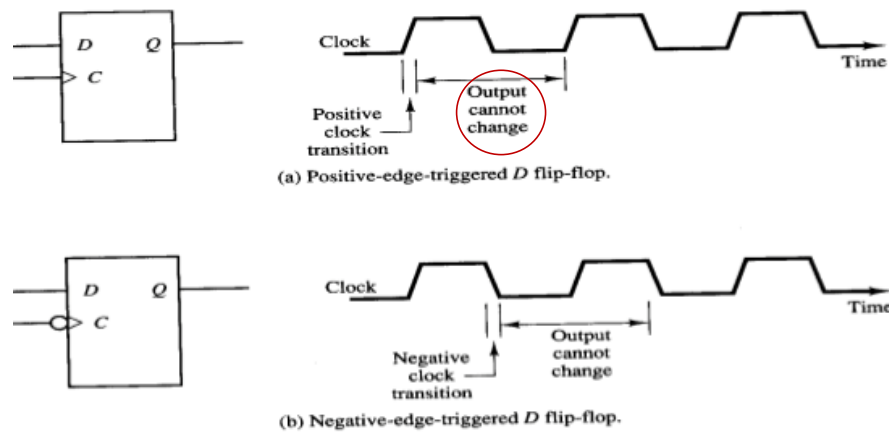
- Toggle 플립플롭
- 특성표

T	Q(t+1)	
0	Q(t)	No change
1	Q'(t)	Complement



## 모서리-변이형 플립플롭

- Edge-triggered Flip-Flop
- 플립플롭 상태 변이 시점/동기화 시점
- Positive/Negative edge-triggered Flip-Flop



## 여기표, excitation table

- 플립플롭 상태 변이를 유도하기 위한 입력 값
- JK-플립플롭 예 :
  - 특성표와 여기표

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Q'(t)

특성표

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

여기표

# 여기표, excitation table

Q(t)	Q(t+1)	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

SR 플립플롭 여기표

Q(t)	Q(t+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

D 플립플롭 여기표

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

JK 플립플롭 여기표

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

T 플립플롭 여기표



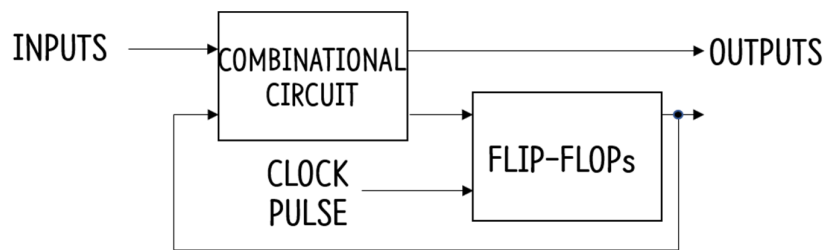
## 순차 회로





## 순차논리회로

- 시간/상태 개념 포함
- 출력과 상태 정보
- 입력 조합 + 현재 상태 → 출력
- 입력조합 + 현재 상태 → 다음 상태
- n개의 입력/k개의 상태/m개의 출력

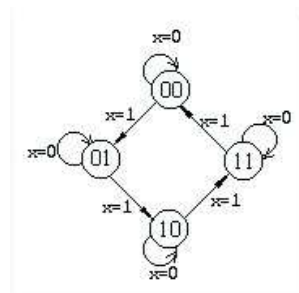
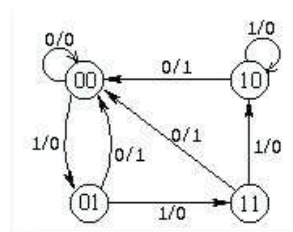


## 순차 논리 회로 설계

- 기능 정의/제시
- 입력/상태/출력의 수 및 문자 기호 결정
- 상태도 작성
- 상태표 작성
- 여기표 작성
- 플립플롭 입력 식/출력 결정
- 논리회로 작성

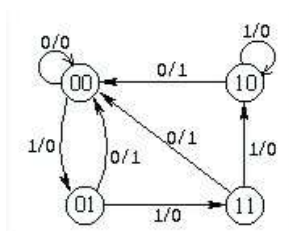
## 상태도

- 입력에 대한 상태의 변이 및 출력 표현
- 구성요소
  - 원 : 상태
  - 화살표 : 상태 전이
  - 화살표 표시 : 입력/출력



## 상태표

- 상태를 표현한 표
- 현재 상태, 입력 조합과 다음 상태, 출력을 나타낸 표



Present State		Input	Next State		Output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

## 플립플롭 입력식

- 상태표를 확장한 여기표 이용
- 입력 조합 및 현재 상태 조합에 대한 플립플롭 각 입력값을 위한 부울 식
  - 입력 및 현재 상태를 이용, 다음 상태 전이를 위한 입력값 결정

Present			Next					
state		Input	state		Flip-Flop		Input	
A	B	X	A	B	J <sub>A</sub>	K <sub>A</sub>	J <sub>B</sub>	K <sub>B</sub>
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	0	1	0	X	X	0
0	1	1	1	0	1	X	X	1
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

X	X	1
X	X	1

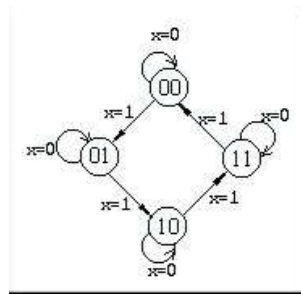
$$K_B = X$$

## 순차 논리회로 설계 예 - 2비트 이진 카운터

- 기능 정의
  - 00→01→10→11→00→01→...
- 외부입력
  - 1 : 카운터
  - 0 : 카운터 중지
- 문자 기호 결정
  - 외부 입력 : x
  - 상태 : A, B
  - 외부 출력 : 없음.

## 순차 논리회로 설계 예 - 2비트 이진 카운터

- 상태도 및 상태표 작성



Present state		Input X	Next state	
A	B		A	B
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

## 순차 논리회로 설계 예 - 2비트 이진 카운터

- 플립플롭 결정 및 여기표 작성
  - JK-플립플롭 2개 사용
    - 각각 A, B로 표시

Present state		Input X	Next state		Flip-Flop Input			
A	B		A	B	J <sub>A</sub>	K <sub>A</sub>	J <sub>B</sub>	K <sub>B</sub>
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	0	1	0	X	X	0
0	1	1	1	0	1	X	X	1
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

## 순차 논리회로 설계 예 - 2비트 이진 카운터

- 플립플롭 입력식 결정

Present state		Input X	Next state		Flip-Flop Input			
A	B		A	B	$J_A$	$K_A$	$J_B$	$K_B$
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	0	1	0	X	X	0
0	1	1	1	0	1	X	X	1
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

		1	
X	X	X	X

 $J_A = Bx$ 

X	X	X	X
		1	

 $K_A = Bx$ 

	1	X	X
	1	X	X

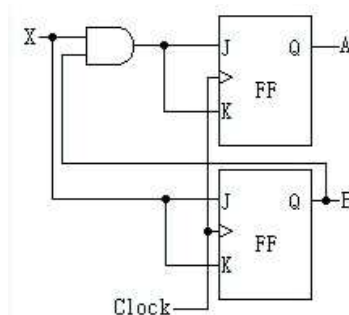
 $J_B = x$ 

X	X	1	
X	X	1	

 $K_B = x$ 

## 순차 논리회로 설계 예 - 2비트 이진 카운터

- 블록 다이어그램 작성



---

Q&A

