Soft Computing And Next generation Network Lab.
Hanyang Univ.

순차논리회로

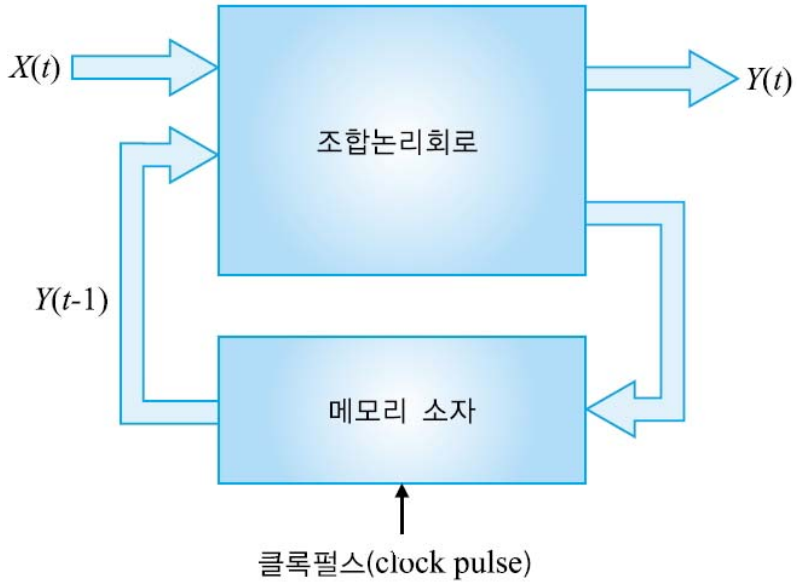
01 동기 순서논리회로 개요

❑ 조합논리회로와 순서논리회로

조합논리회로 (combinational logic circuit)	• 출력이 현재의 입력에 의해서만 결정되는 논리회로
순서논리회로 (sequential logic circuit)	• 현재의 입력과 이전의 출력상태에 의해서 출력이 결정되는 논리회로. • 순서논리회로는 신호의 타이밍(timing)에 따라 동기 순서논리회로와 비동기 순서논리회로로 분류. • 동기 순서회로에서 상태(state)는 단지 이산된(discrete) 각 시점 즉, 클록펄스가 들어오는 시점에서 상태가 변화하는 회로 • 클록펄스에 의해서 동작하는 회로를 동기순서논리회로 또는 단순히 동기순서회로라 한다. • 비동기 순서회로는 시간에 관계없이 단지 입력이 변화하는 순서에 따라 동작하는 논리회로

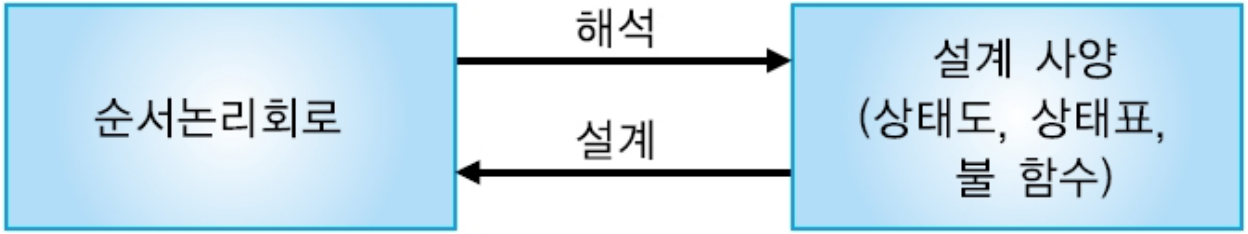
01 동기 순서논리회로 개요

- **순서논리회로의 블록도**



출력 $Y(t)$는 현재 상태의 입력 $X(t)$와 이전 상태의 출력 $Y(t-1)$에 의하여 결정

- **순서논리회로의 해석과 설계 관계**



02 동기 순서논리회로의 해석과정

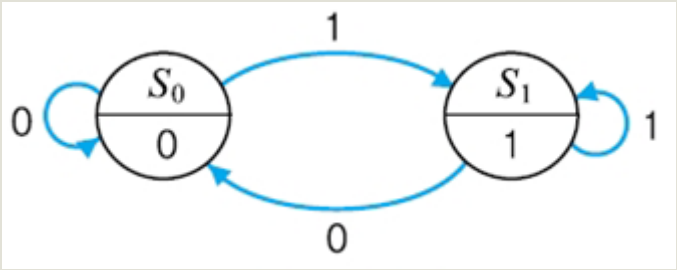
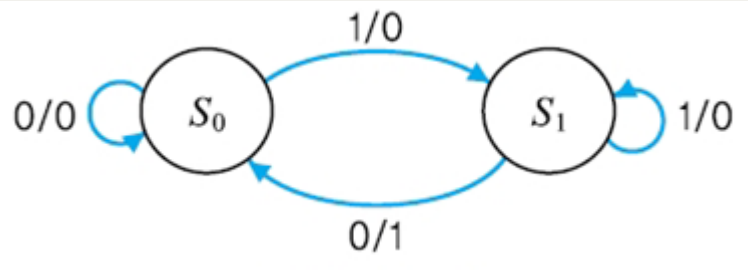
- ❖ 순서논리회로의 동작은 입력과 출력 및 플립플롭의 현재상태에 의해 결정
- ❖ 출력과 차기상태는 현재상태의 함수가 된다.
- ❖ 순서논리회로의 해석은 입력과 출력 및 현재상태에 의해 결정되는 차기상태의 시간순서를 상태표나 상태도로 나타냄으로써 해석이 가능

• 순서논리회로의 해석과정

[단계 1]	회로 입력과 출력에 대한 변수 명칭 부여
[단계 2]	조합논리회로가 있으면 조합논리회로의 부울대수식 유도
[단계 3]	회로의 상태표 작성
[단계 4]	상태표를 이용하여 상태도 작성
[단계 5]	상태방정식 유도
[단계 6]	상태표와 상태도를 분석하여 회로의 동작 설명

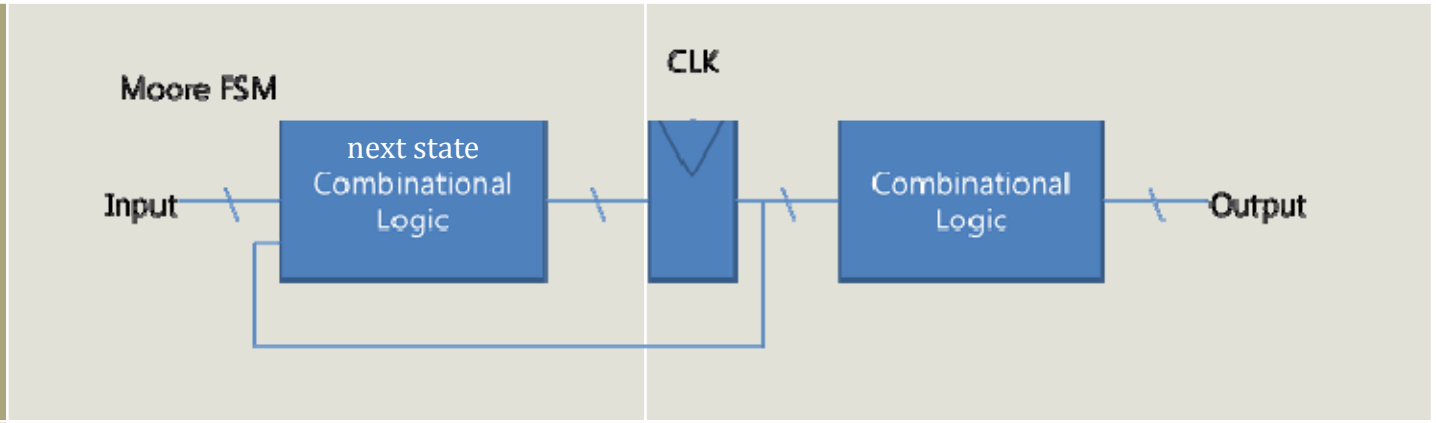
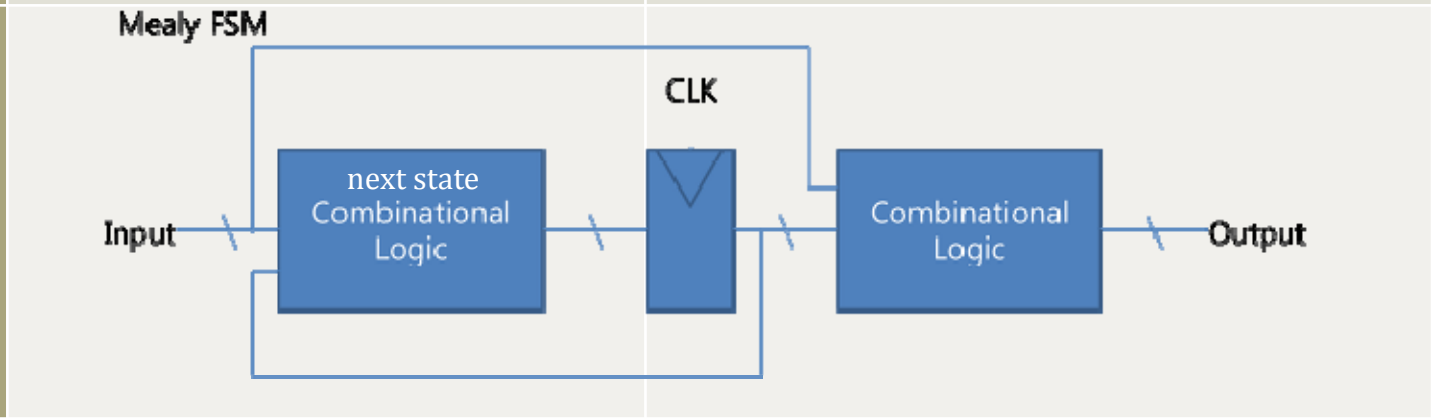
02 동기 순서논리회로의 해석과정

❑ 상태도 종류

무어머신 (Moore machine)	순서논리회로의 출력이 플립플롭들의 현재 상태만의 함수인 회로. 출력이 상태 내에 결합되어 표시된다. **오직 현재의 상태에만 의존하여 결과 값이 도출**	
밀리머신 (Mealy machine)	출력이 현재 상태와 입력의 함수인 회로. 출력은 상태간을 지나가는 화살선의 위에 표시된다. **현재의 결과값과 입력값의 조합으로 결과 값이 도출**	

02 동기 순서논리회로의 해석과정

❑ 상태도 종류

무어머신 (Moore machine)	
밀리머신 (Mealy machine)	

무어머신
(Moore machine)

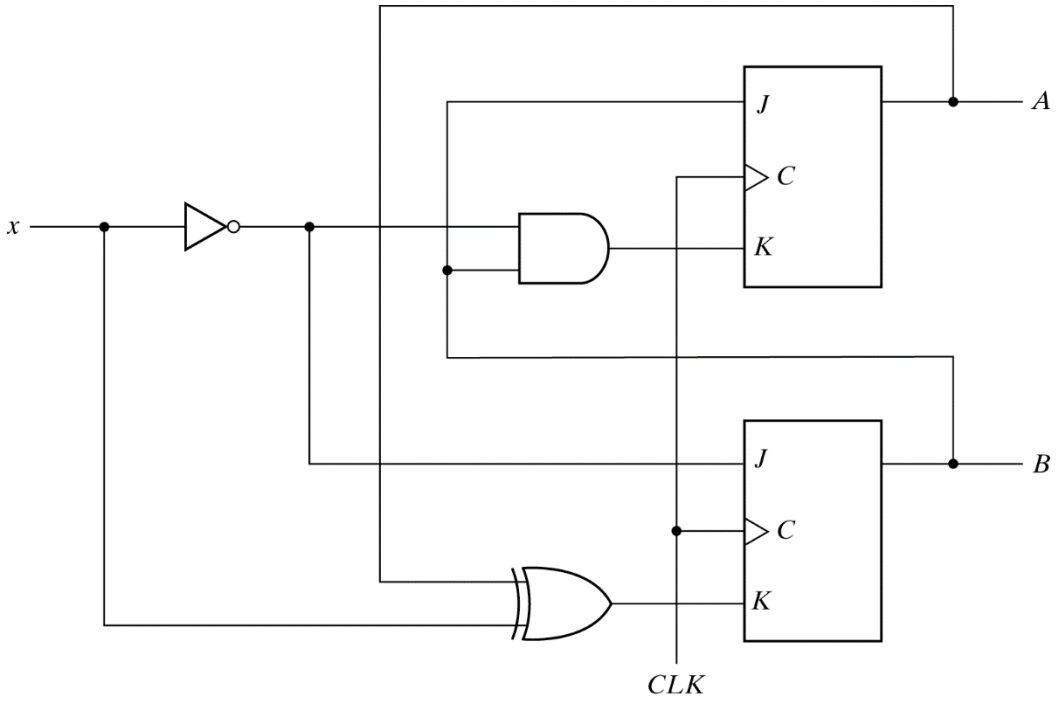


Fig. 5-18 Sequential Circuit with *JK* Flip-Flop

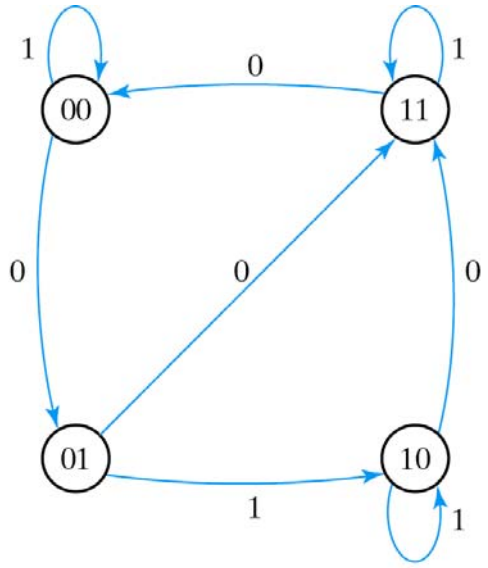


Fig. 5-19 State Diagram of the Circuit of Fig. 5-18

무어머신
(Moore machine)

```
module Moore_Model_Fig_5_19 (output [1: 0] y_out, input x_in, clock, reset);
  reg [1: 0]state;
  parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;

  always @ (posedge clock, negedge reset)
    if (reset == 0) state <= S0;   // Initialize to state S0
    else case (state)
      S0: if (~x_in) state <= S1; else state <= S0;
      S1: if (x_in)   state <= S2; else state <= S3;
      S2: if (~x_in) state <= S3; else state <= S2;
      S3: if (~x_in) state <= S0; else state <= S3;
    endcase

  assign y_out = state;              // Output of flip-flops

endmodule
```

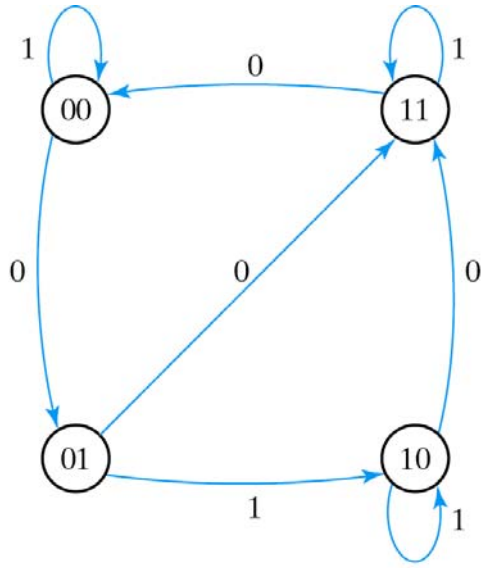


Fig. 5-19 State Diagram of the Circuit of Fig. 5-18

무어머신
(Moore machine)

```
module t_Moore_Fig_5_19;
  wire [1: 0]          t_y_out;
  reg                  t_x_in, t_clock, t_reset;

Moore_Model_Fig_5_19 M0 (t_y_out, t_x_in, t_clock, t_reset);

initial #200 $finish;
initial begin t_clock = 0; forever #5 t_clock = ~t_clock; end

initial fork
      t_reset = 0;  #2 t_reset = 1;  #87 t_reset = 0;  #89 t_reset = 1;  #10 t_x_in = 1;
  #30 t_x_in = 0;  #40 t_x_in = 1;  #50 t_x_in = 0;  #52 t_x_in = 1;  #54 t_x_in = 0;
  #70 t_x_in = 1;  #80 t_x_in = 1;  #70 t_x_in = 0;  #90 t_x_in = 1;  #100 t_x_in = 0;
  #120 t_x_in = 1;  #160 t_x_in = 0;  #170 t_x_in = 1;
  join
endmodule
```

밀리머신
(Mealy machine)

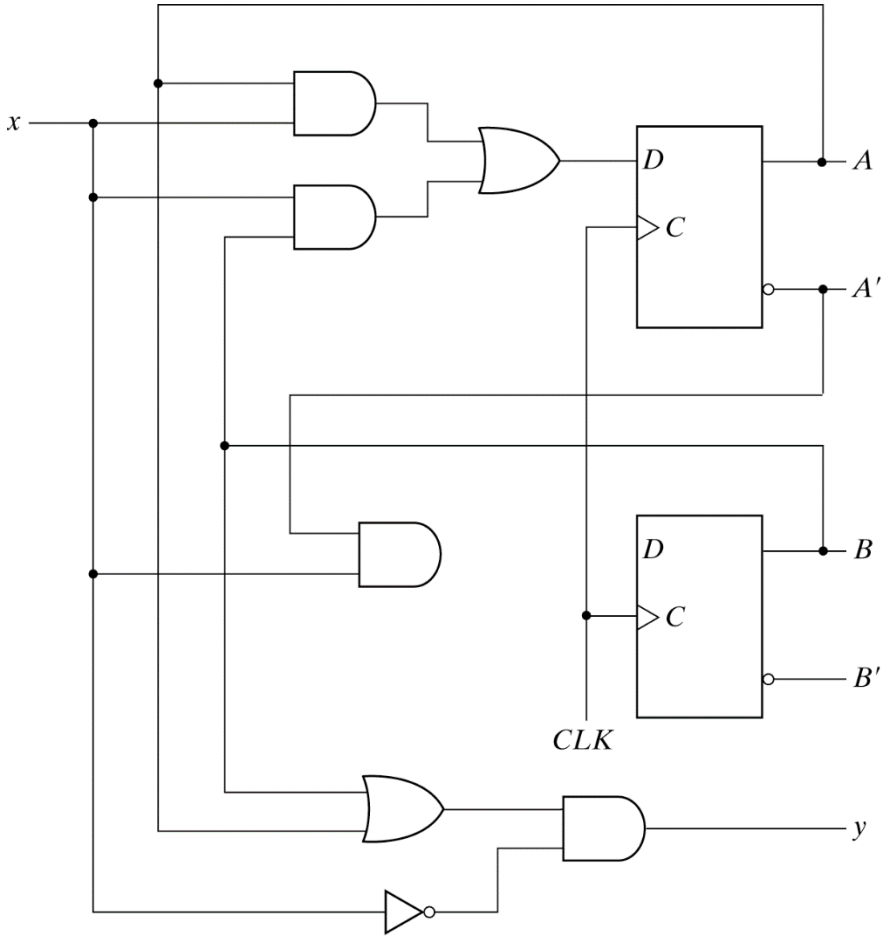


Fig. 5-15 Example of Sequential Circuit

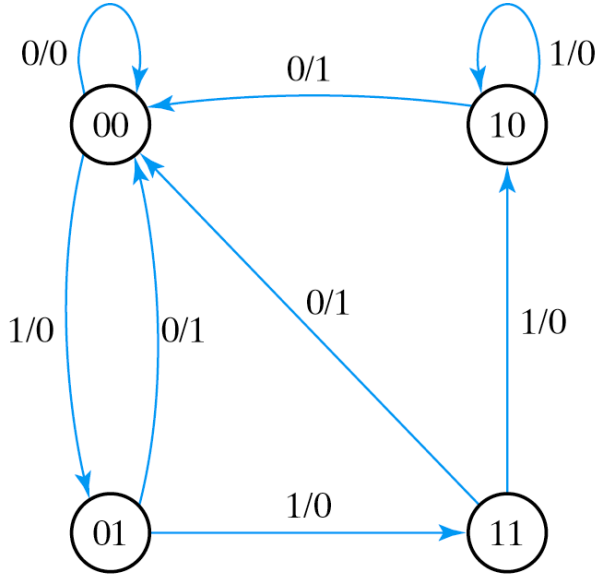


Fig. 5-16 State Diagram of the Circuit of Fig. 5-15

밀리머신
(Mealy machine)

```
module Mealy_Zero_Detector (output reg y_out, input x_in, clock, reset);
  reg [1: 0]   state, next_state;
  parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;

  always @ (posedge clock, negedge reset)          // state transition
    if (reset == 0) state <= S0;
    else state <= next_state;

  always @ (state, x_in) // Form the next state
    case (state)
      S0: if (x_in)   next_state = S1; else next_state = S0;
      S1: if (x_in)   next_state = S3; else next_state = S0;
      S2: if (~x_in) next_state = S0; else next_state = S2;
      S3: if (x_in)   next_state = S2; else next_state = S0;
    endcase

  always @ (state, x_in) // Form the output
    case (state)
      S0: y_out = 0;
      S1, S2, S3: y_out = ~x_in;
    endcase
endmodule
```

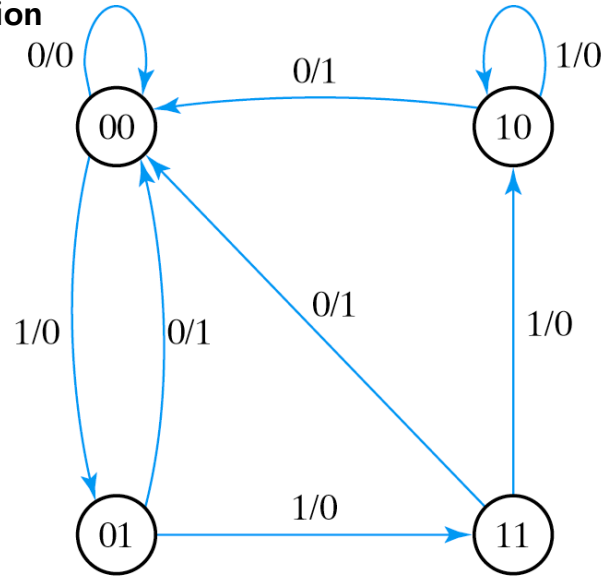


Fig. 5-16 State Diagram of the Circuit of Fig. 5-15

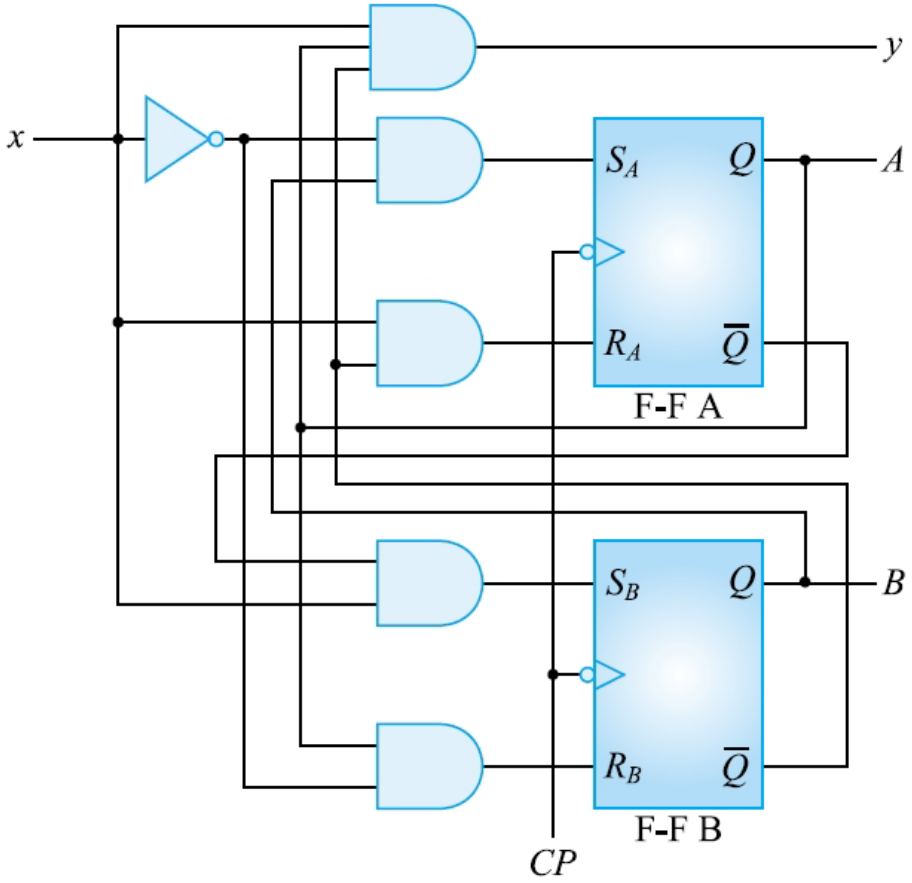
밀리머신
(Mealy machine)

```
module t_Mealy_Zero_Detector;
  wire  t_y_out;
  reg t_x_in, t_clock, t_reset;

Mealy_Zero_Detector M0 (t_y_out, t_x_in, t_clock, t_reset);

initial #200 $finish;
initial begin t_clock = 0; forever #5 t_clock = ~t_clock; end

initial fork
      t_reset = 0;  #2 t_reset = 1;  #87 t_reset = 0;  #89 t_reset = 1;  #10 t_x_in = 1;
  #30 t_x_in = 0;  #40 t_x_in = 1;  #50 t_x_in = 0;  #52 t_x_in = 1;  #54 t_x_in = 0;
  #70 t_x_in = 1;  #80 t_x_in = 1;  #70 t_x_in = 0;  #90 t_x_in = 1;  #100 t_x_in = 0;
  #120 t_x_in = 1;  #160 t_x_in = 0;  #170 t_x_in = 1;
  join
endmodule
```

02 동기 순서논리회로의 해석과정

1. 변수명칭 부여

- **입력변수 :** x
- **출력 변수 :** y
- F-F A **플립플롭의 입력 :** S_A, R_A
- F-F B **플립플롭의 입력 :** S_B, R_B
- F-F A **플립플롭의 출력 :** A
- F-F B **플립플롭의 출력 :** B



2. 부울 대수식 유도

- F-F A 플립플롭의 입력

$$S_A = B\bar{x}, \quad R_A = \bar{B}x$$

- F-F B 플립플롭의 입력

$$S_B = \bar{A}x, \quad R_B = A\bar{x}$$

- 시스템 출력

$$y = A\bar{B}x$$

02 동기 순서논리회로의 해석과정

3. 상태표 작성

- 상태표(state table)는 현재상태와 외부 입력의 변화에 따라 차기상태와 출력의 변화를 정의한 것
- 현재상태란 클록펄스(*CP*)의 인가 전을 나타내며, 차기상태란 클록펄스의 인가 후를 나타낸다.

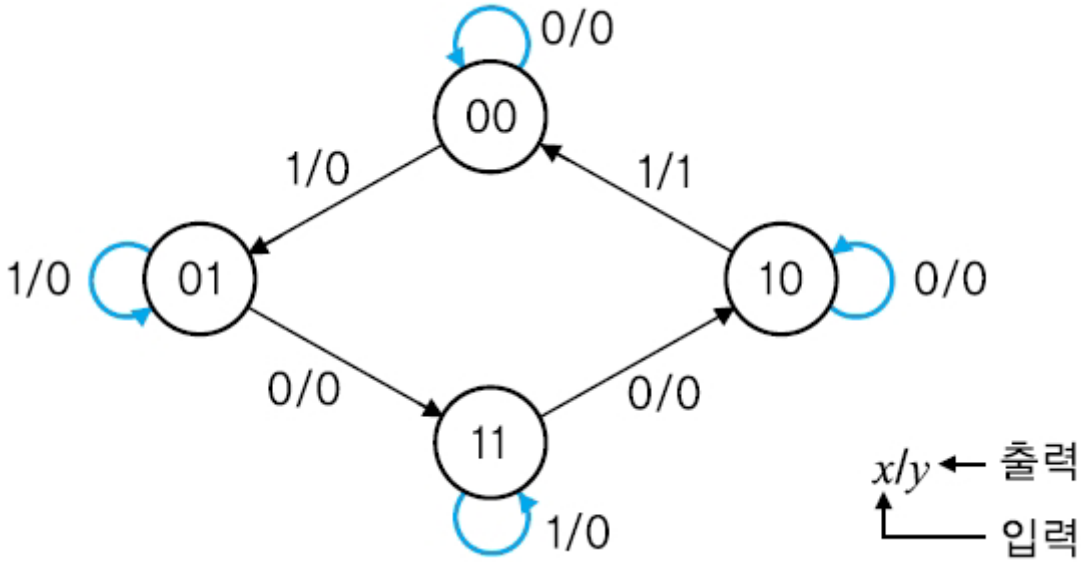
현재상태		차기상태				출력	
		x=0		x=1		x=0	x=1
A	B	A	B	A	B	y	y
0	0	0	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	1	0	0	0	0	1
1	1	1	0	1	1	0	0

상태표

02 동기 순서논리회로의 해석과정

4. 상태도 작성

❖ 상태표로부터 상태도를 그린다.



상태도

02 동기 순서논리회로의 해석과정

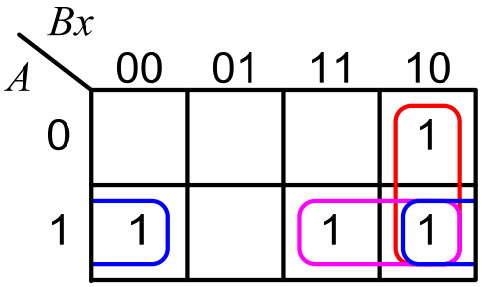
5. 상태방정식 유도

- 상태 방정식(state equation)은 플립플롭 상태 천이에 대한 조건을 지정하는 대수식
- 상태표로부터 플립플롭 A와 B가 논리 1이 되는 상태 방정식은 각각 다음과 같다.

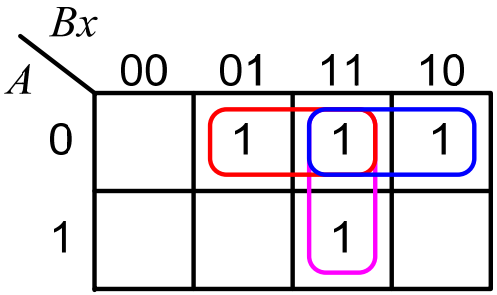
$$A(t+1) = (\overline{A}B + A\overline{B} + AB)\overline{x} + ABx$$
$$B(t+1) = \overline{A}B\overline{x} + (\overline{A}\overline{B} + \overline{A}B + AB)x$$

- 카르노 도표를 이용하여 간소화한 상태 방정식

$$A(t+1) = B\overline{x} + AB + A\overline{x}$$



$$B(t+1) = \overline{A}x + \overline{A}B + Bx$$



02 동기 순서논리회로의 해석과정

❖ *S-R* 플립플롭의 특성방정식과 비교

$A(t+1) = B\bar{x} + AB + A\bar{x}$ $= B\bar{x} + (B + \bar{x})A = B\bar{x} + (\overline{\bar{B}x})A$	⇔	$A(t+1) = (S_A) + (\bar{R}_A)A$
$B(t+1) = \bar{A}x + \bar{A}B + Bx$ $= \bar{A}x + (\bar{A} + x)B = \bar{A}x + (\overline{\bar{A}x})B$		$B(t+1) = (S_B) + (\bar{R}_B)B$

$S_A = B\bar{x}$ $\quad R_A = \bar{B}x$

$S_B = \bar{A}x$ $\quad R_B = A\bar{x}$

6. 회로의 동작설명

❖ 순서논리회로의 동작은 상태도나 상태표를 이용하여 설명 가능

❖ 입력 의 값에 따라 클록펄스가 한번씩 인가될 때마다
0(00)→1(01)→3(11)→2(10)의 순으로 순차적으로 동작하는 순서논리회로

03 플립플롭의 여기표

- 플립플롭의 특성표는 현재상태와 입력값이 주어졌을 때, 차기상태가 어떻게 변하는가를 나타내는 표.
- 플립플롭의 여기표(excitation table)는 현재상태에서 차기상태로 변했을 때 플립플롭의 입력조건이 어떤 상태인가를 나타내는 표.
- 플립플롭의 여기표는 순서논리회로를 설계할 때 자주 사용

03 플립플롭의 여기표

1. S-R 플립플롭의 여기표

특성표

입력		현재상태	차기상태
S	R	$Q(t)$	$Q(t+1)$
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	?

여기표

현재상태	차기상태	요구입력	
$Q(t)$	$Q(t+1)$	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

03 플립플롭의 여기표

2. J-K 플립플롭의 여기표

특성표

입력		현재상태	차기상태
J	K	$Q(t)$	$Q(t+1)$
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	0

여기표

현재상태	차기상태	요구입력	
$Q(t)$	$Q(t+1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

03 플립플롭의 여기표

3. *D* 플립플롭의 여기표

특성표

입력	현재상태	차기상태
D	$Q(t)$	$Q(t+1)$
0	0	0
0	1	0
1	0	1
1	1	1

여기표

현재상태	차기상태	요구입력
$Q(t)$	$Q(t+1)$	D
0	0	0
0	1	1
1	0	0
1	1	1

4. *T* 플립플롭의 여기표

특성표

입력	현재상태	차기상태
T	$Q(t)$	$Q(t+1)$
0	0	0
0	1	1
1	0	1
1	1	0

여기표

현재상태	차기상태	요구입력
$Q(t)$	$Q(t+1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

04 동기 순서논리회로의 설계 과정

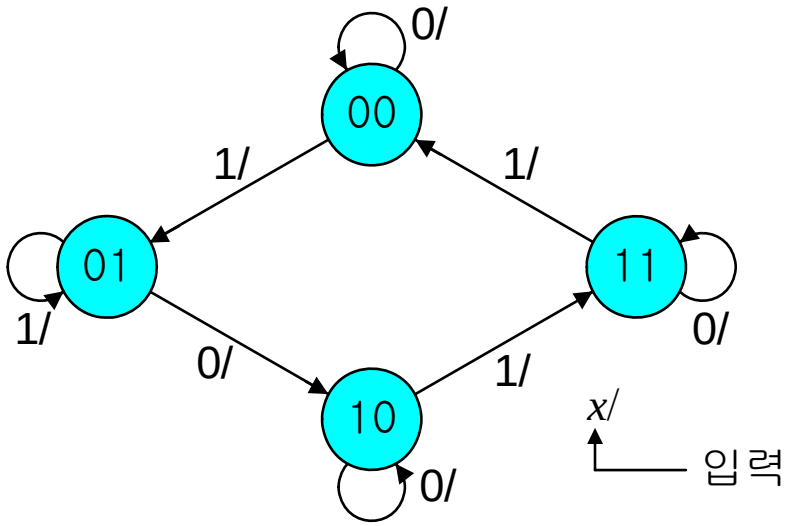
❑ 순서논리회로의 설계 과정

[단계 1]	회로 동작 기술(상태도 작성)
[단계 2]	정의된 회로의 상태표 작성
[단계 3]	필요한 경우 상태 축소 및 상태 할당
[단계 4]	플립플롭의 수와 플립플롭의 종류 결정
[단계 5]	플립플롭의 입력, 출력 및 각각의 상태에 문자기호 부여
[단계 6]	상태표를 이용하여 회로의 여기표 작성
[단계 7]	간략화 방법을 이용하여 출력 함수 및 플립플롭의 입력함수 유도
[단계 8]	순서논리회로도 작성

04 동기 순서논리회로의 설계 과정

1. 회로 동작 기술

❖ 입력 변수만 있고 출력 변수는 없는 상태에서 상태 변화가 일어난다.



동기 순서논리회로에 대한 상태도

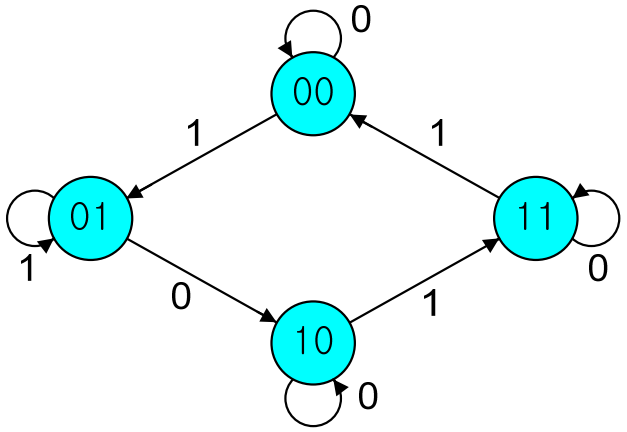
04 동기 순서논리회로의 설계 과정

2. 상태표 작성

❖ 상태도로부터 상태표 유도

현재상태		차기상태			
		x=0		x=1	
A	B	A	B	A	B
0	0	0	0	0	1
0	1	1	0	0	1
1	0	1	0	1	1
1	1	1	1	0	0

상태표



상태도

04 동기 순서논리회로의 설계 과정

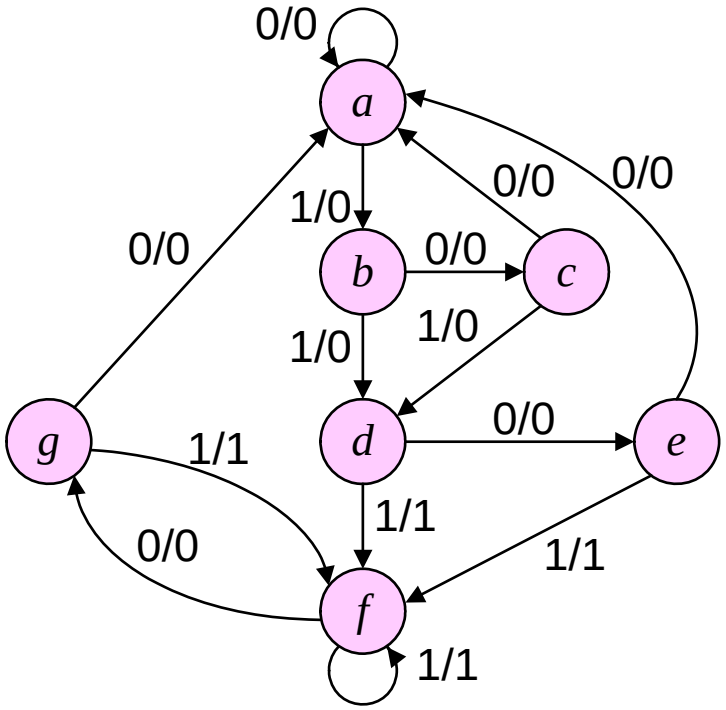
3. 상태 축소 및 상태 할당

- **문자 기호에 의해서 표시된 상태를 가진 상태도로부터 간략화된 상태표를 유도하기 위한 절차에 대해서 알아보기로 한다.**
- **상태도로부터 얻어진 상태표는 하나 또는 그 이상의 불필요한 상태** (redundant state)**를 가질 수 있다.**
- **축소된 최소 상태표**(minimal state table)**를 유도하기 위한 과정은 상태 축소와 상태 할당의** 2**단계에 의해서 수행된다.**

• 상태 축소

- 순서논리회로에서 플립플롭의 수를 줄이는 것
- **플립플롭의 수가** m**이라 가정하면, 이때 요구되는 상태는** 2^m **이 되므로 상태의 수를 줄임으로써 플립플롭의 수를 줄일 수 있다. 그러나 경우에 따라 상태의 수는 감소되지만 플립플롭의 수는 변화하지 않는 경우도 있다.**

04 동기 순서논리회로의 설계 과정



상태도

현재상태	차기상태		출력	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

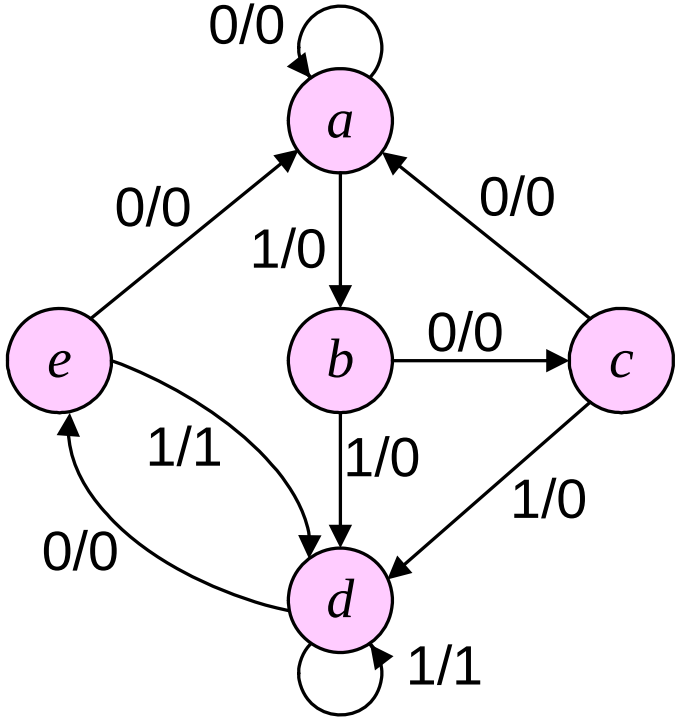
상태표

04 동기 순서논리회로의 설계 과정

현재 상태	차기상태		출력	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f d	0	1
e	a	f d	0	1
f	g e	f	0	1
g	a	f	0	1

최종 상태표

현재 상태	차기상태		출력	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
	a	d	0	1



축소된 상태도

04 동기 순서논리회로의 설계 과정

• 상태 할당

❖ 기호 형태로 표현된 각각의 상태에 대해서 2진수(2진 코드)의 값을 할당하는 과정

상태	할당1	할당2	할당3
a	0 0 1	0 0 0	0 0 0
b	0 1 0	0 1 0	1 0 0
c	0 1 1	0 1 1	0 1 0
d	1 0 0	1 0 1	1 0 1
e	1 0 1	1 1 1	0 1 1

현재 상태	차기상태		출력	
	x=0	*x*=1	*x*=0	*x*=1
0 0 1	0 0 1	0 1 0	0	0
0 1 0	0 1 1	1 0 0	0	0
0 1 1	0 0 1	1 0 0	0	0
1 0 0	1 0 1	1 0 0	0	1
1 0 1	0 0 1	1 0 0	0	1

할당 1에 의한 최소 상태표

04 동기 순서논리회로의 설계 과정

4. 플립플롭의 수와 형태의 결정(참고)

- 정의해야 할 상태의 수가 n가지이면 $\lceil \log_2 n \rceil$개의 플립플롭이 필요.
- 예를 들어 n=16이면, $\lceil \log_2 16 \rceil = 4 \log_2 2 = 4$
- 예를 들어 n=4이면, $\lceil \log_2 4 \rceil = 2 \log_2 2 = 2$

04 동기 순서논리회로의 설계 과정

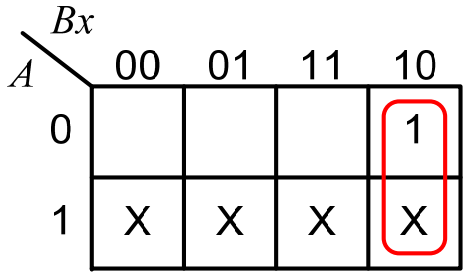
5. 상태 여기표의 유도

조합회로의 입력			차기상태		조합회로의 출력			
현재상태		입력			플립플롭 입력			
A	B	x	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	1	0	1	x	x	1
0	1	1	0	1	0	x	x	0
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

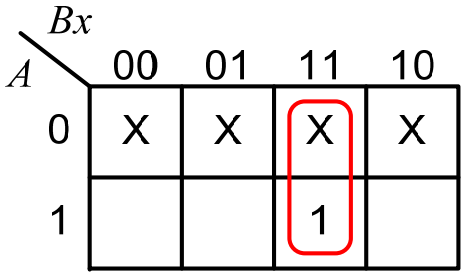
$Q(t)$	$Q(t+1)$	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

J-K 플립플롭의 여기표

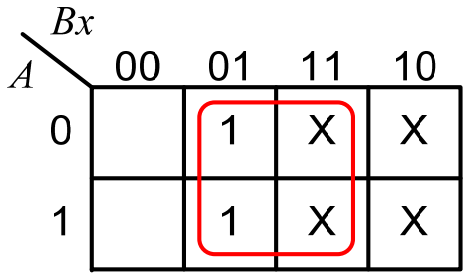
6. 플립플롭의 출력 함수 및 회로의 입력 함수 유도



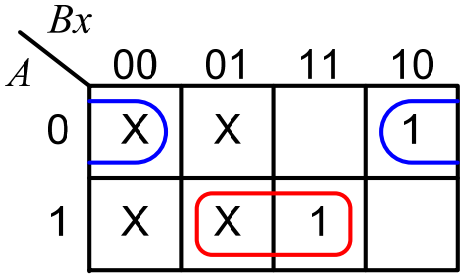
$$J_A = B\overline{x}$$



$$K_A = Bx$$



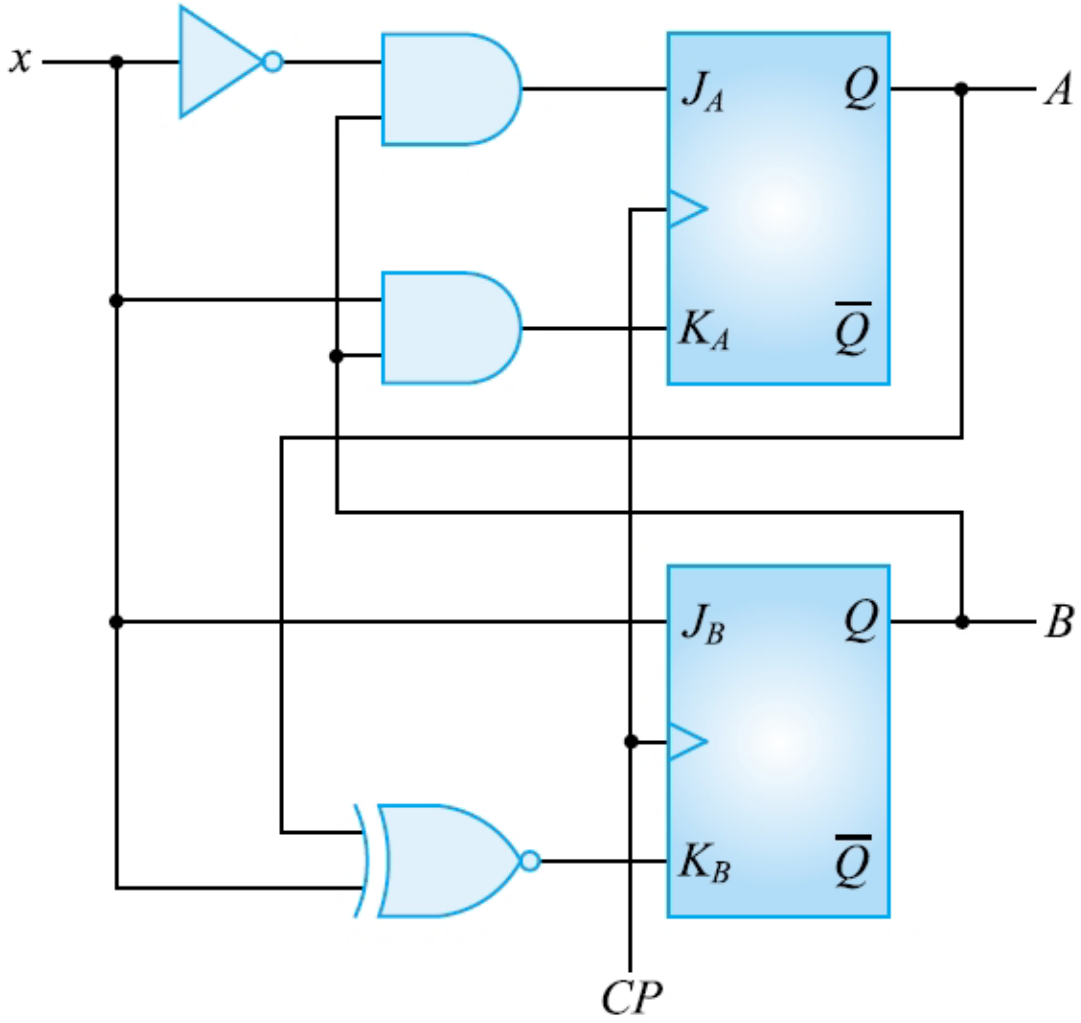
$$J_B = x$$



$$K_B = Ax + \overline{A}\overline{x} = \overline{A \oplus x} = A \odot x$$

04 동기 순서논리회로의 설계 과정

7. 논리 회로의 구현



$$J_A = B\overline{x}$$

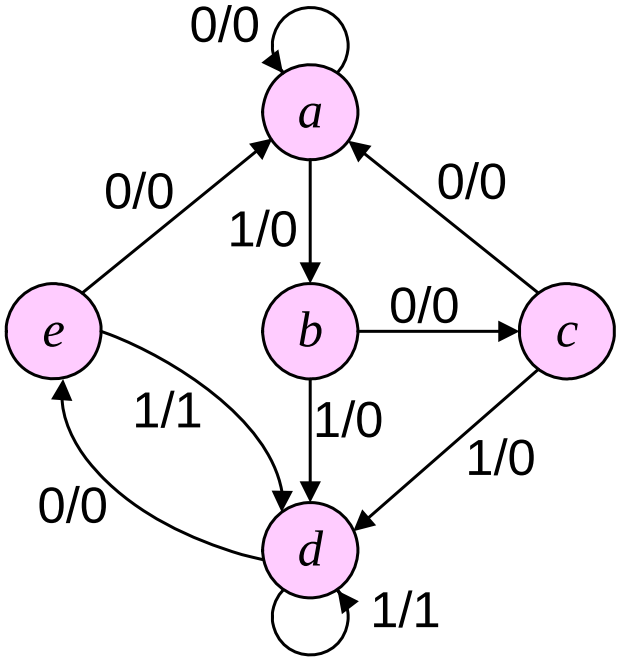
$$K_A = Bx$$

$$J_B = x$$

$$K_B = A \odot x$$

05 동기 순서논리회로의 설계 예

• 상태도



❑ 상태표

현재상태	차기상태		출력	
	x=0	x=1	x=0	x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

• 상태할당 및 플립플롭 수 결정

- ❖ 제어하려는 상태의 수는 5가지이므로 3비트가 필요
- ❖ 3개의 S-R 플립플롭을 순서대로 A, B, C라고 정의
- ❖ 현재 상태 a, b, c, d, e에 각각 000, 001, 010, 011, 100을 할당.

05 동기 순서논리회로의 설계 예

- 순서제어회로의 상태 여기표 작성

	현재상태	외부입력	차기상태	플립플롭의 입력						외부출력
	$A\,B\,C$	x	$A\,B\,C$	S_A	R_A	S_B	R_B	S_C	R_C	y
a	0 0 0	0	0 0 0	0	x	0	x	0	x	0
	0 0 0	1	0 0 1	0	x	0	x	1	0	0
b	0 0 1	0	0 1 0	0	x	1	0	0	1	0
	0 0 1	1	0 1 1	0	x	1	0	x	0	0
c	0 1 0	0	0 0 0	0	x	0	1	0	x	0
	0 1 0	1	0 1 1	0	x	x	0	1	0	0
d	0 1 1	0	1 0 0	1	0	0	1	0	1	0
	0 1 1	1	0 1 1	0	x	x	0	x	0	1
e	1 0 0	0	0 0 0	0	1	0	x	0	x	0
	1 0 0	1	0 1 1	0	1	1	0	1	0	1
don't care	1 0 1	0	x x x	x	x	x	x	x	x	x
	1 0 1	1	x x x	x	x	x	x	x	x	x
	1 1 0	0	x x x	x	x	x	x	x	x	x
	1 1 0	1	x x x	x	x	x	x	x	x	x
	1 1 1	0	x x x	x	x	x	x	x	x	x
	1 1 1	1	x x x	x	x	x	x	x	x	x

03 플립플롭의 여기표

1. S-R 플립플롭의 여기표

특성표

입력		현재상태	차기상태
S	R	$Q(t)$	$Q(t+1)$
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	?

여기표

현재상태	차기상태	요구입력	
$Q(t)$	$Q(t+1)$	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

05 동기 순서논리회로의 설계 예

- 플립플롭의 출력 함수 및 회로의 출력 함수 유도

AB \ Cx	00	01	11	10
00				
01				1
11	X	X	X	X
10			X	

$$S_A = BC\bar{x}$$

AB \ Cx	00	01	11	10
00	X	X	X	X
01	X	X	X	
11	X	X	X	X
10	1	1	X	X

$$R_A = A$$

AB \ Cx	00	01	11	10
00			1	1
01		X	X	
11	X	X	X	X
10		1	X	X

$$S_B = Ax + \bar{B}C$$

AB \ Cx	00	01	11	10
00	X	X		
01	1			1
11	X	X	X	X
10	X		X	X

$$R_B = B\bar{x}$$

AB \ Cx	00	01	11	10
00		1	X	
01		1	X	
11	X	X	X	X
10		1	X	X

$$S_C = x$$

AB \ Cx	00	01	11	10
00	X			1
01	X			1
11	X	X	X	X
10	X		X	X

$$R_C = \bar{x}$$

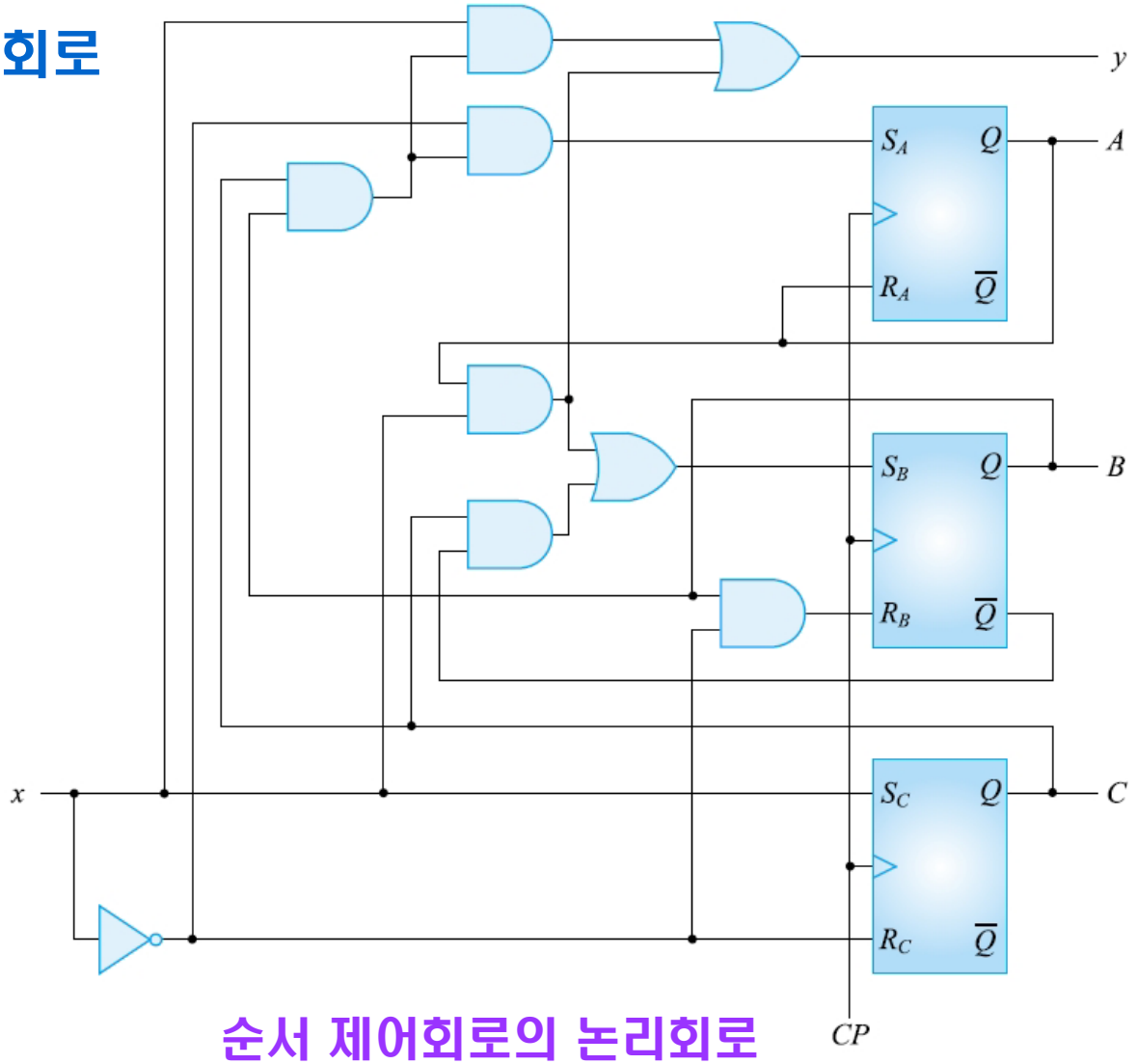
05 동기 순서논리회로의 설계 예

- 순서 제어 회로의 논리회로

AB \ Cx	00	01	11	10
00				
01			1	
11	X	X	X	X
10		1	X	X

$$y = Ax + BCx$$

$$S_A = BC\bar{x} \qquad R_A = A$$
$$S_B = Ax + \bar{B}C \qquad R_B = B\bar{x}$$
$$S_C = x \qquad R_C = \bar{x}$$
$$y = Ax + BCx$$



순서 제어회로의 논리회로

06 미사용 상태의 설계

- 순서논리회로에서는 어떠한 상태도 초기 상태가 될 수 있으므로 현재상태를 순서논리회로에서 모두 사용하지 않는 경우 문제점 발생.
- 따라서 사용하지 않는 상태에 대해 **차기** 상태가 어떤지를 구할 필요가 있다.

현재상태			차기상태					
			x=0			x=1		
A	B	C	A	B	C	A	B	C
0	1	0	0	1	1	0	1	0
0	1	1	0	1	1	1	1	1
1	0	0	1	0	0	1	1	0
1	0	1	1	0	1	1	0	0
1	1	0	1	1	0	0	1	0
1	1	1	1	0	1	1	1	1

상태표

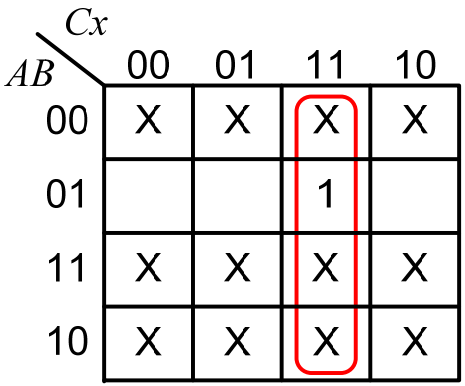
06 미사용 상태의 설계

- 순서논리회로의 상태 여기표

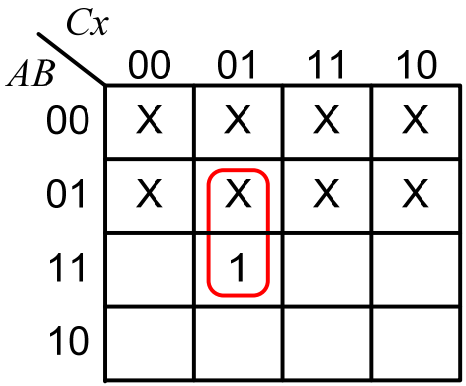
입력	현재 상태			차기 상태			플립플롭 입력					
x	A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	1	0	0	1	1	0	×	×	0	1	×
0	0	1	1	0	1	1	0	×	×	0	×	0
0	1	0	0	1	0	0	×	0	0	×	0	×
0	1	0	1	1	0	1	×	0	0	×	×	0
0	1	1	0	1	1	0	×	0	×	0	0	×
0	1	1	1	1	0	1	×	0	×	1	×	0
1	0	1	0	0	1	0	0	×	×	0	0	×
1	0	1	1	1	1	1	1	×	×	0	×	0
1	1	0	0	1	1	0	×	0	1	×	0	×
1	1	0	1	1	0	0	×	0	0	×	×	1
1	1	1	0	0	1	0	×	1	×	0	0	×
1	1	1	1	1	1	1	×	0	×	0	×	0

06 미사용 상태의 설계

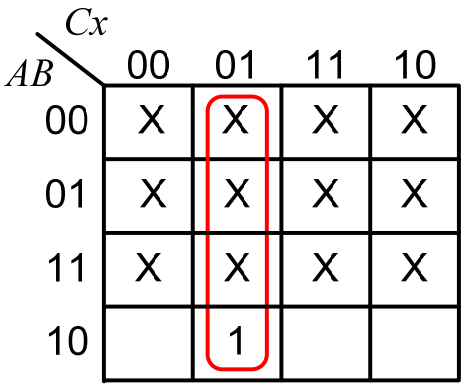
- 사용하지 않은 2개의 상태(000, 001)에 대해서는 카르노 맵에서 무관항으로 처리하여 간소화



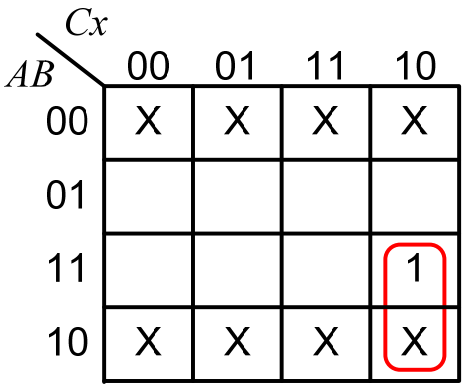
$$J_A = Cx$$



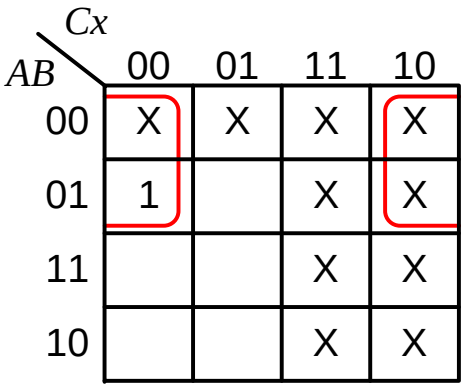
$$K_A = B\overline{C}x$$



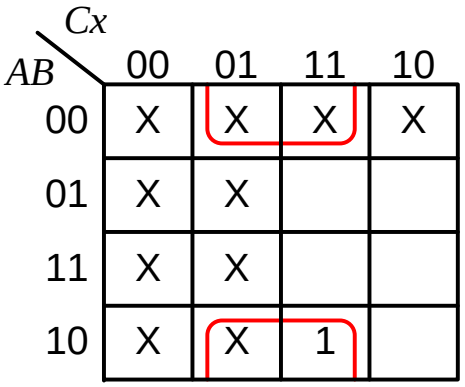
$$J_B = \overline{C}x$$



$$K_B = AC\overline{x}$$



$$J_C = \overline{A}x$$

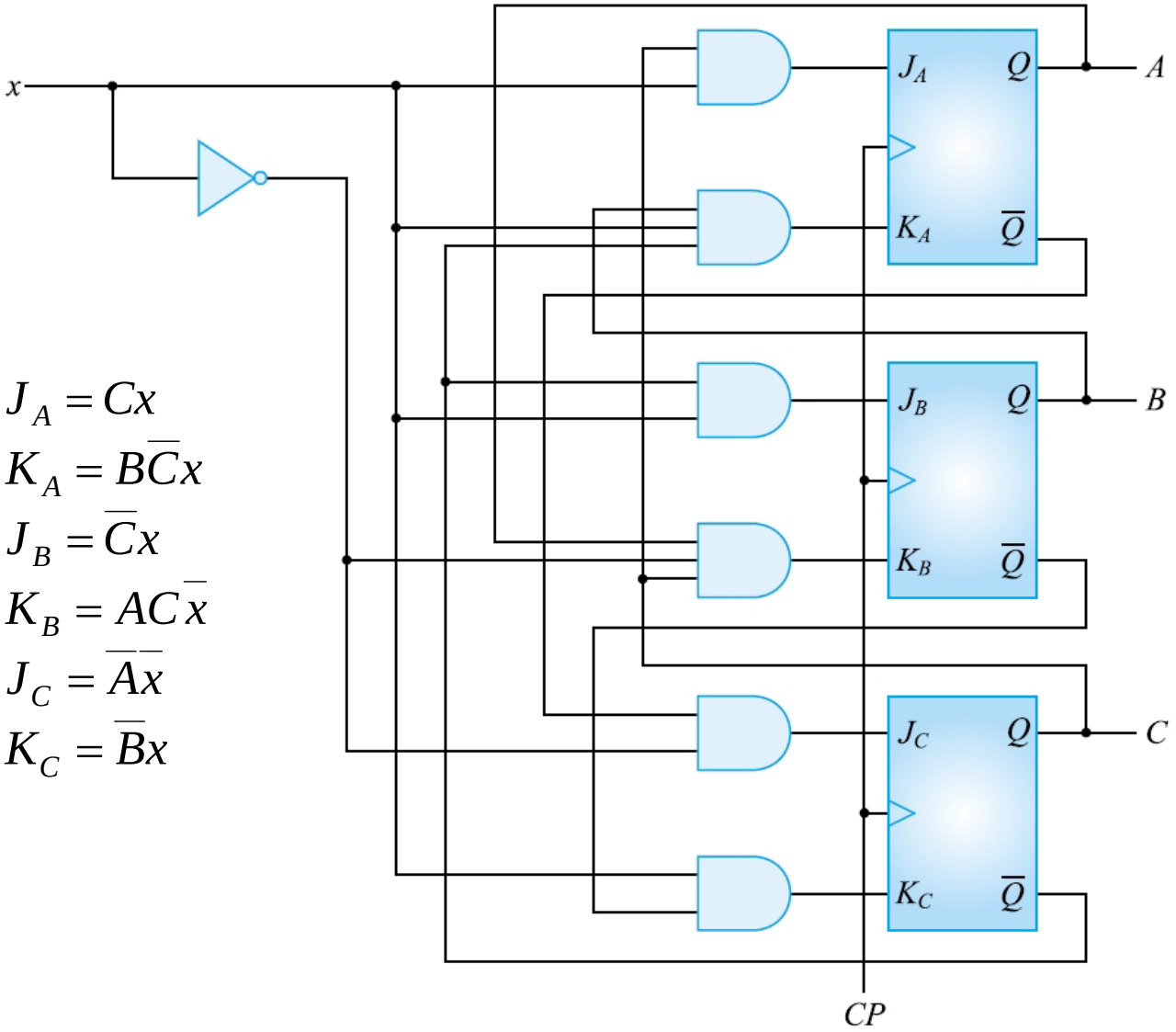


$$K_C = \overline{B}x$$

06 미사용 상태의 설계

- 순서논리회로

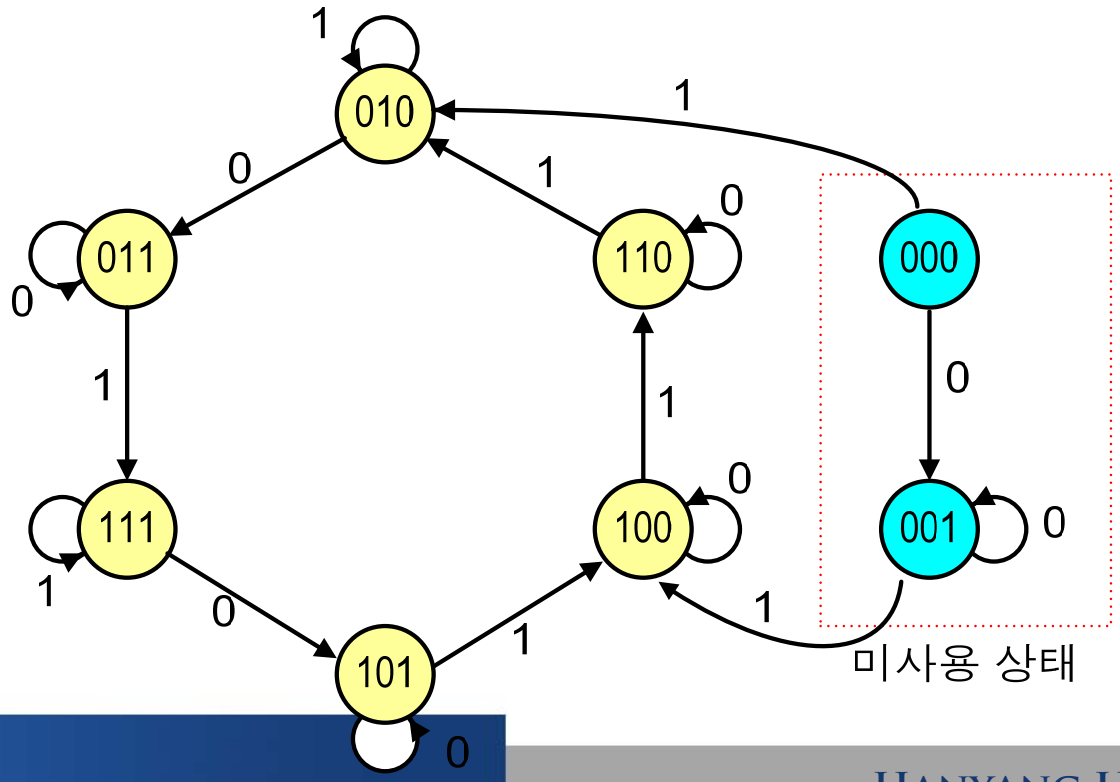
$$J_A = Cx$$
$$K_A = B\overline{C}x$$
$$J_B = \overline{C}x$$
$$K_B = AC\overline{x}$$
$$J_C = \overline{A}\,\overline{x}$$
$$K_C = \overline{B}x$$



06 미사용 상태의 설계

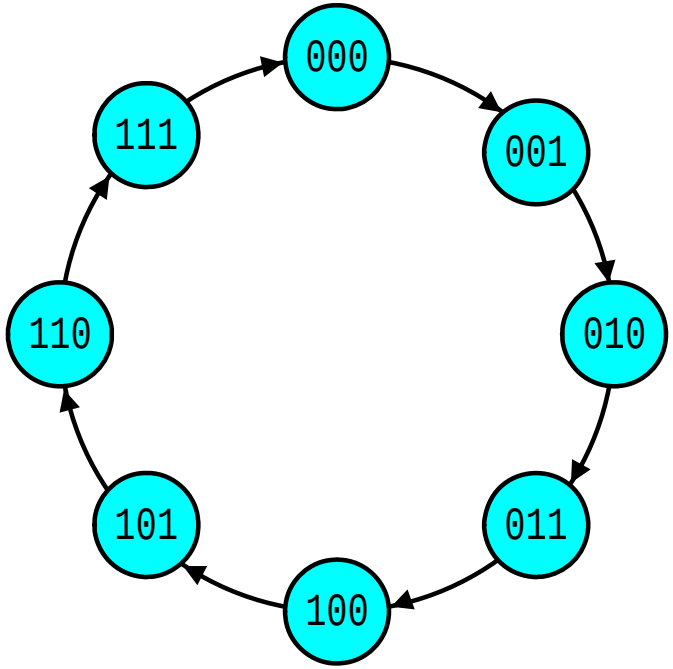
현재상태			차기상태					
			x=0			x=1		
A	B	C	A	B	C	A	B	C
0	0	0	0	0	1	0	1	0
0	0	1	0	0	1	1	0	0

미사용 상태의 상태표



07 카운터의 설계

- 3 비트 2진 카운터 설계



상태도

현재상태			차기상태		
A	*B*	*C*	*A*	*B*	*C*
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

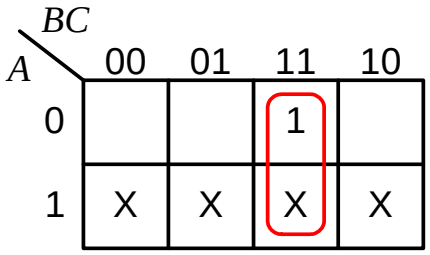
상태표

07 카운터의 설계

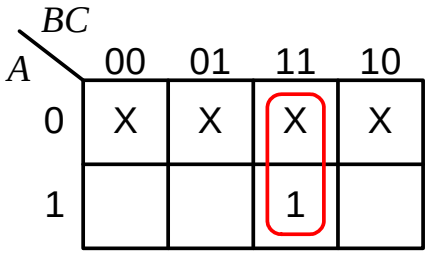
현재상태			차기상태			플립플롭 입력					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	x	0	x	1	x
0	0	1	0	1	0	0	x	1	x	x	1
0	1	0	0	1	1	0	x	x	0	1	x
0	1	1	1	0	0	1	x	x	1	x	1
1	0	0	1	0	1	x	0	0	x	1	x
1	0	1	1	1	0	x	0	1	x	x	1
1	1	0	1	1	1	x	0	x	0	1	x
1	1	1	0	0	0	x	1	x	1	x	1

상태 여기표

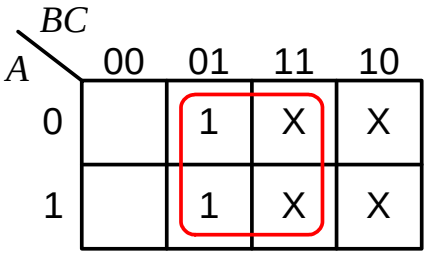
07 카운터의 설계



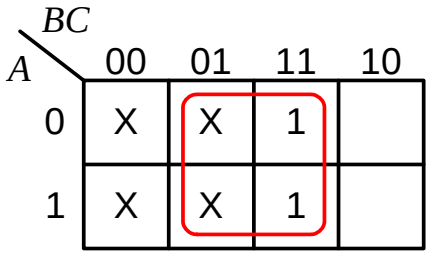
$J_A = BC$



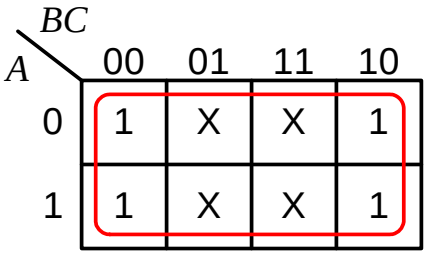
$K_A = BC$



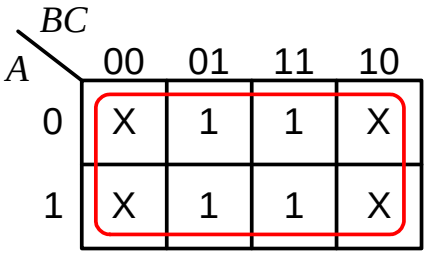
$J_B = C$



$K_B = C$

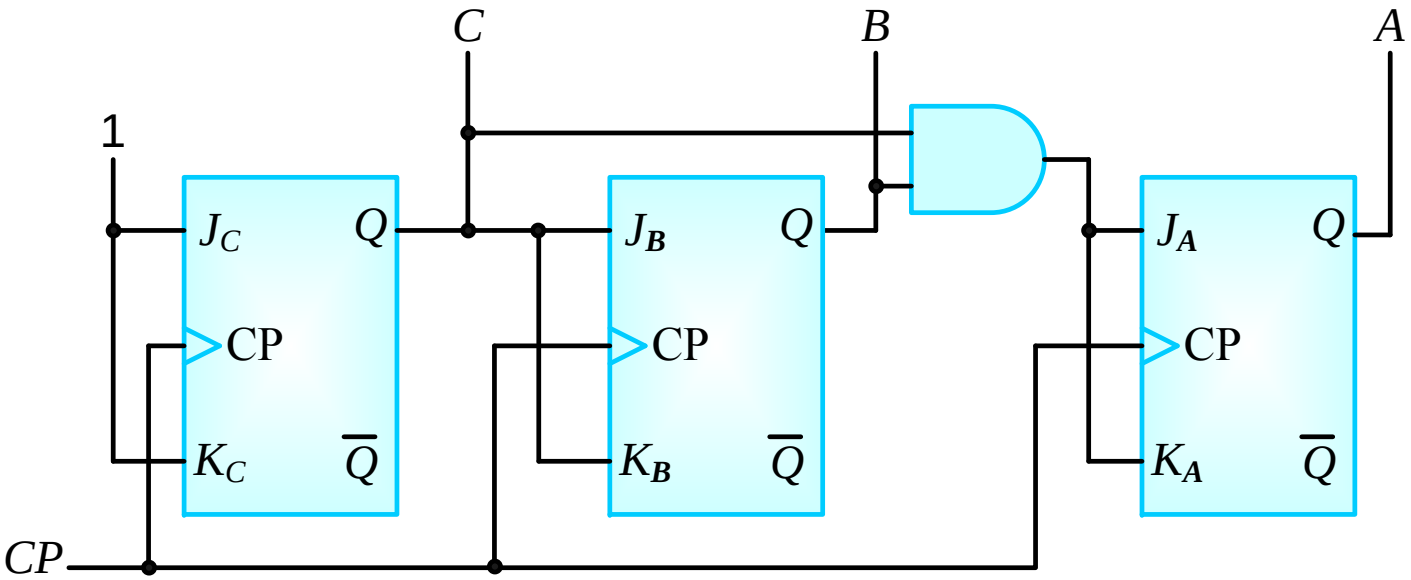


$J_C = 1$



$K_C = 1$

07 카운터의 설계



회로도

$$J_C = 1 \qquad J_B = C \qquad J_A = BC$$
$$K_C = 1 \qquad K_B = C \qquad K_A = BC$$

07 카운터의 설계

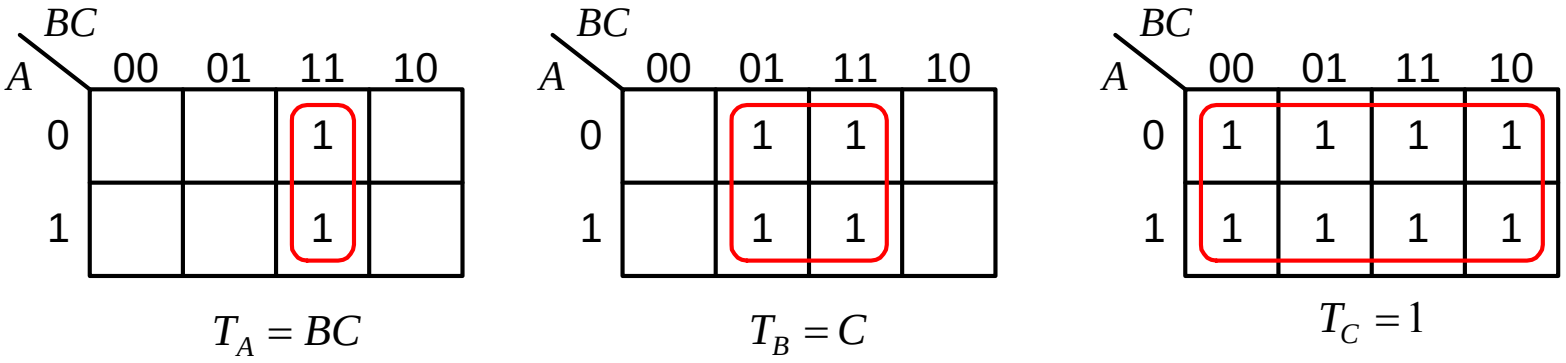
예제 9-1 T 플립플롭을 사용하여 3비트 2진 카운터를 구현하여라.

❑ 상태 여기표

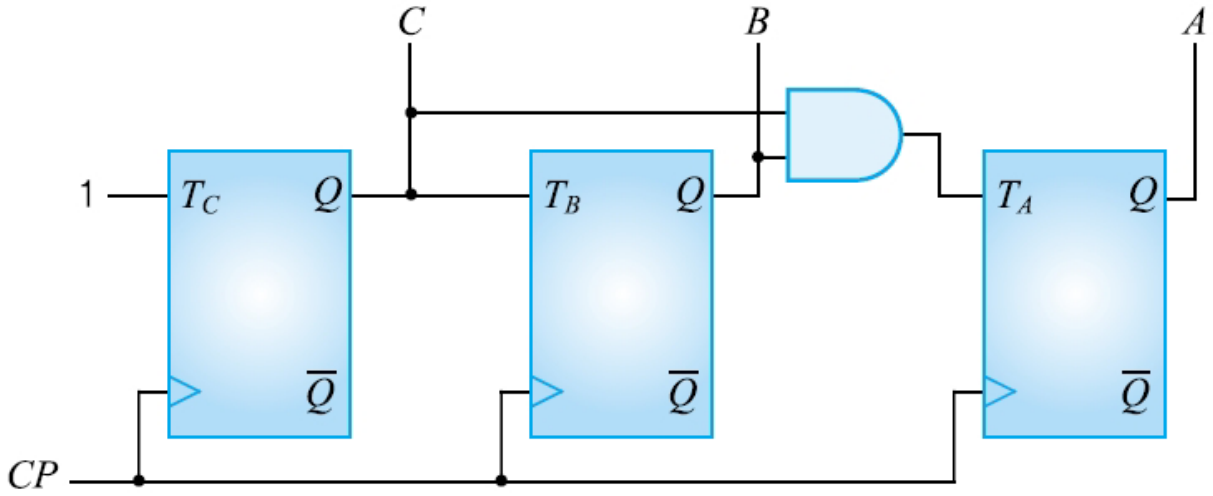
현재상태			차기상태			플립플롭 입력		
A	B	C	A	B	C	T_A	T_B	T_C
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

07 카운터의 설계

- 카르노 맵에 의한 간략화

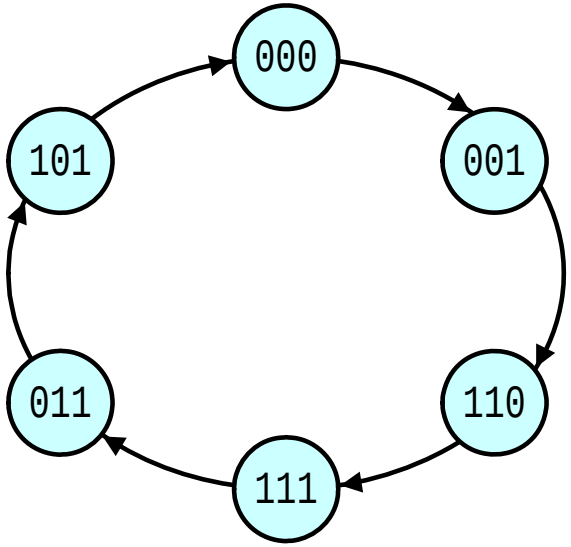


- 3비트 2진 카운터 회로



07 카운터의 설계

예제 9-2 *J-K* 플립플롭을 사용하여 아래의 상태도에 해당하는 카운터를 설계하고, 미사용 상태에 대한 상태도를 구하여라.

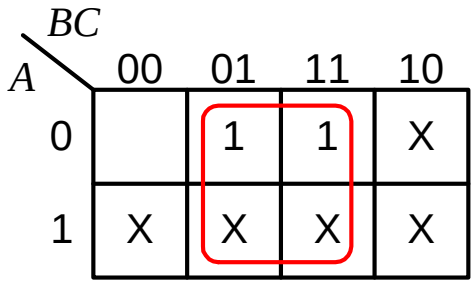


07 카운터의 설계

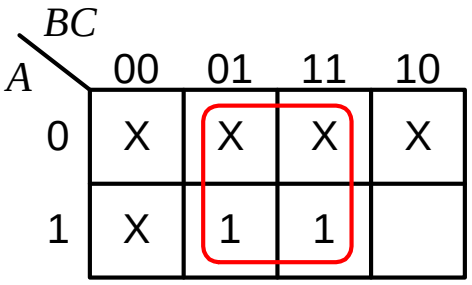
❑ 상태 여기표

현재상태			차기상태			플립플롭 입력					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	×	0	×	1	×
0	0	1	1	1	0	1	×	1	×	×	1
0	1	1	1	0	1	1	×	×	1	×	0
1	0	1	0	0	0	×	1	0	×	×	1
1	1	0	1	1	1	×	0	×	0	1	×
1	1	1	0	1	1	×	1	×	0	×	0

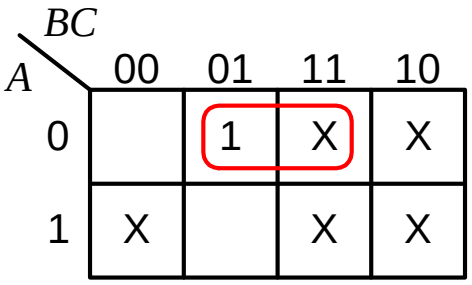
❑ 카르노 맵에 의한 간략화



$J_A = C$



$K_A = C$



$J_B = \overline{A}C$

07 카운터의 설계

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

$K_B = \overline{A}$

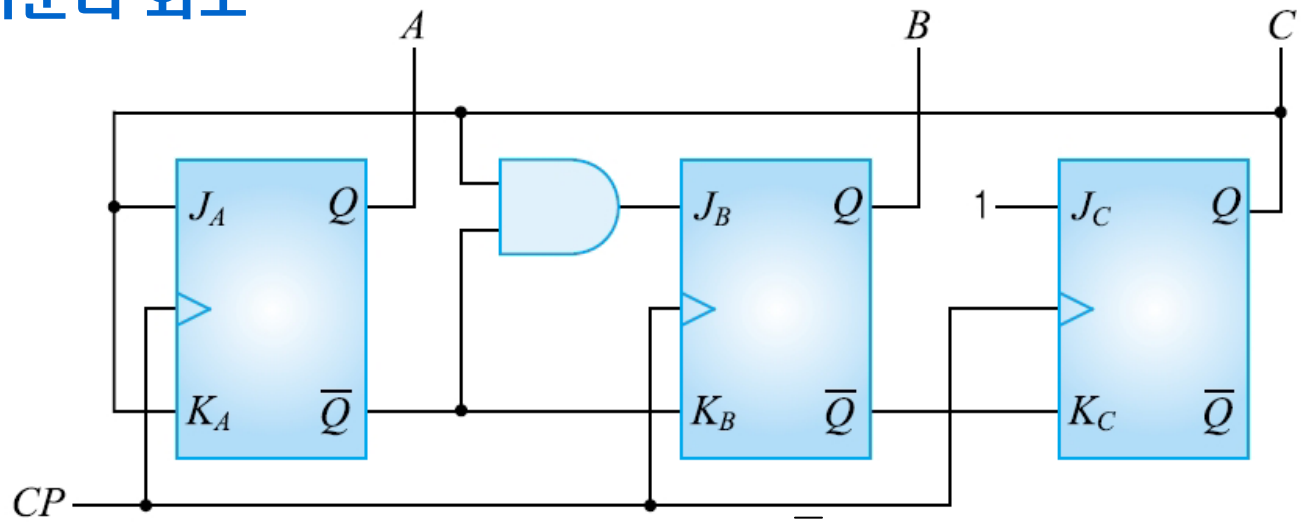
A \ BC	00	01	11	10
0	1	X	X	X
1	X	X	X	1

$J_C = 1$

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

$K_C = \overline{B}$

- 카운터 회로



$J_A = C$

$K_A = C$

$J_B = \overline{A}C$

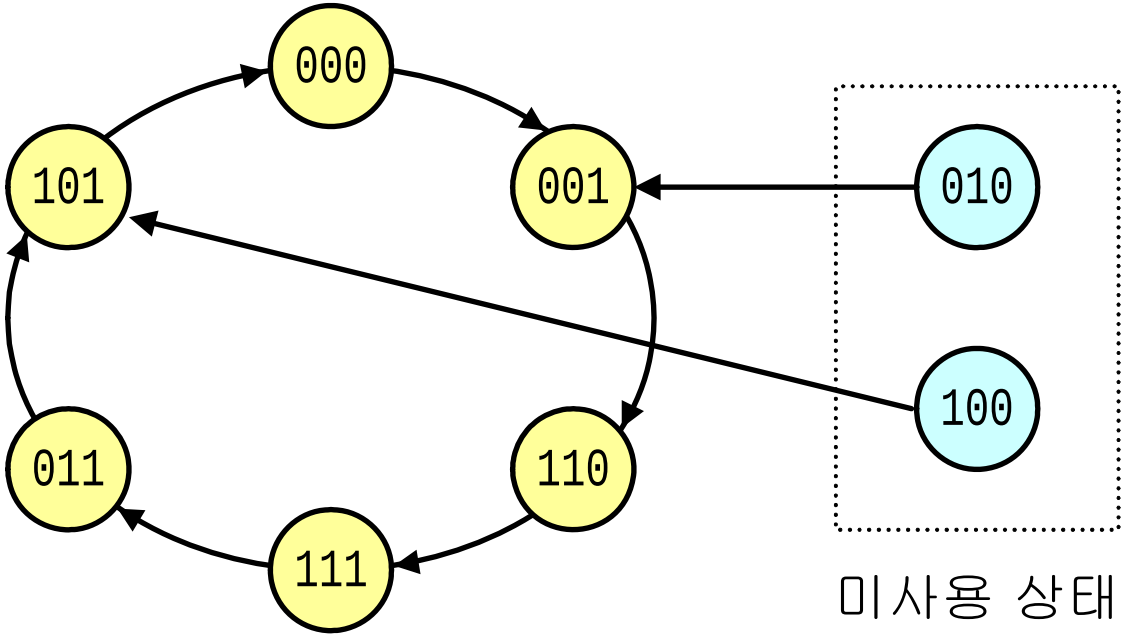
$K_B = \overline{A}$

$J_C = 1$

$K_C = \overline{B}$

07 카운터의 설계

- 미사용 상태를 포함한 카운터의 상태도



08 상태 방정식을 이용한 설계

1. *J-K* 플립플롭을 사용한 상태 방정식

- 순서논리회로의 상태방정식은 상태표에 표시된 정보와 똑같은 내용을 대수적으로 표시하고 있으며, 플립플롭의 특성방정식과 형태가 유사
- 상태방정식은 상태표에서 쉽게 유도할 수 있으며, 모든 순서논리회로는 상태방정식으로 표시할 수 있다.
- 특히 *D* 플립플롭이나 *J-K* 플립플롭을 사용하는 경우 상태방정식을 사용하여 순서논리회로를 설계하는 것이 더욱 편리하다.
- *S-R* 플립플롭이나 *T* 플립플롭을 가진 회로에도 상태방정식을 적용할 수 있으나 많은 대수적 처리가 필요하다.

❑ *J-K* 플립플롭을 사용한 상태 방정식

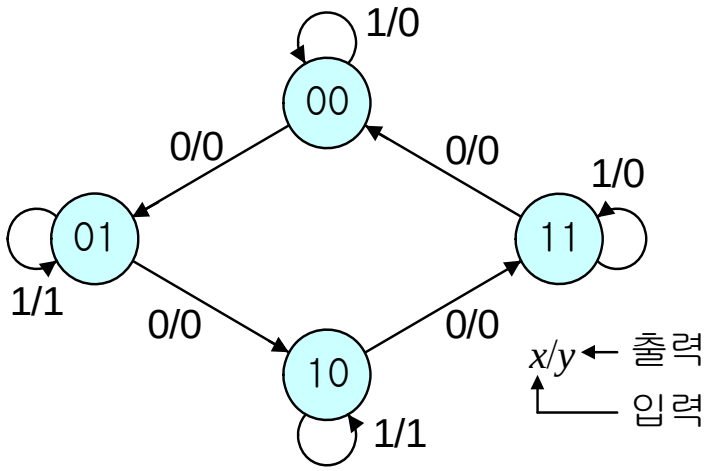
$$Q(t+1) = J\overline{Q} + \overline{K}Q$$

***J-K* 플립플롭의 특성방정식**

- *J-K* 플립플롭의 상태방정식을 *J-K* 플립플롭의 특성방정식과 같은 형태로 변형함으로써 플립플롭의 *J*와 *K*의 입력 함수를 구할 수 있다.

08 상태 방정식을 이용한 설계

- 상태도(상태방정식을 이용하는 경우)



상태도

- 상태표

현재상태		차기상태				출력	
		x=0		x=1		x=0	x=1
A	B	A	B	A	B	y	y
0	0	0	1	0	0	0	0
0	1	1	0	0	1	0	1
1	0	1	1	1	0	0	1
1	1	0	0	1	1	0	0

08 상태 방정식을 이용한 설계

❖ 2개의 *J-K* 플립플롭을 각각 *A*, *B*라 할 때, 상태 여기표에서 플립플롭 *A*, *B*의 차기상태가 논리 1이 되는 항을 최소항으로 하는 부울 함수를 구한다.

$$A(t+1) = \overline{A}B\overline{x} + A\overline{B}\,\overline{x} + A\overline{B}x + ABx$$
$$= (B\overline{x})\overline{A} + (\overline{B}\,\overline{x} + \overline{B}x + Bx)A$$
$$= (B\overline{x})\overline{A} + \overline{(\overline{B}\,\overline{x} + \overline{B}x + Bx)}A$$

$$A(t+1) = J_A\overline{A} + \overline{K}_A A$$
$$J_A = B\overline{x}$$
$$K_A = \overline{\overline{B}\,\overline{x} + \overline{B}x + Bx} = \overline{(\overline{B} + x)} = B\overline{x}$$

$$\overline{A} + A = 1$$

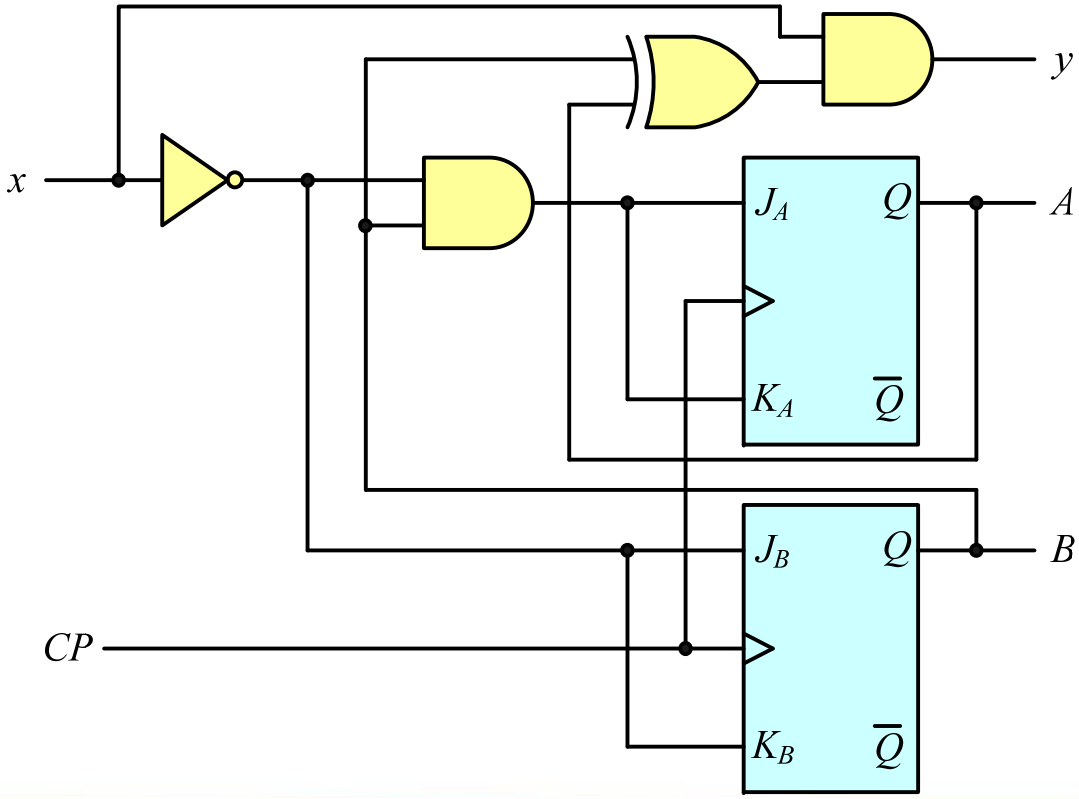
$$B(t+1) = \overline{A}\,\overline{B}\,\overline{x} + A\overline{B}\,\overline{x} + \overline{A}Bx + ABx$$
$$= (\overline{A}\,\overline{x} + A\overline{x})\overline{B} + (\overline{A}x + Ax)B$$
$$= (\overline{A}\,\overline{x} + A\overline{x})\overline{B} + \overline{(\overline{A}x + Ax)}B$$

$$B(t+1) = J_B\overline{B} + \overline{K}_B B$$
$$J_B = \overline{A}\,\overline{x} + A\overline{x} = \overline{x}$$
$$K_B = \overline{\overline{A}x + Ax} = \overline{x}$$

08 상태 방정식을 이용한 설계

$$y = x\bar{A}B + xA\bar{B}$$
$$= x(\bar{A}B + A\bar{B}) = x(A \oplus B)$$

- 회로도(상태 방정식을 이용하는 경우)



$y = x(A \oplus B)$

$J_A = B\bar{x}$

$K_A = B\bar{x}$

$J_B = \bar{x}$

$K_B = \bar{x}$

08 상태 방정식을 이용한 설계

예제 9-3 다음 상태표를 사용하여 순서논리회로를 구현하여라.
(*J-K* 플립플롭 이용)

❑ 상태표

현재상태			차기상태		
A	*B*	*C*	*A*	*B*	*C*
0	0	0	0	1	0
0	1	0	1	1	0
0	1	1	1	0	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	1	1

미사용 상태		
0	0	1
1	0	0

08 상태 방정식을 이용한 설계

- 상태 방정식

$$A(t+1) = \bar{A}B\bar{C} + \bar{A}BC + AB\bar{C} + \boldsymbol{A\bar{B}C} + \boldsymbol{A\bar{B}\bar{C}}$$
$$= (B\bar{C} + BC + \boldsymbol{\bar{B}C})\bar{A} + (B\bar{C} + \boldsymbol{\bar{B}\bar{C}})A$$
$$= (B\bar{C} + BC + \boldsymbol{\bar{B}C})\bar{A} + \overline{(\overline{B\bar{C}} + \overline{\boldsymbol{\bar{B}\bar{C}}})}A$$

$$J_A = B\bar{C} + BC + \boldsymbol{\bar{B}C} = B$$
$$K_A = \overline{B\bar{C} + \boldsymbol{\bar{B}\bar{C}}} = C$$

$$B(t+1) = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + AB\bar{C} + ABC + \boldsymbol{\bar{A}\bar{B}C} + \boldsymbol{A\bar{B}\bar{C}}$$
$$= (\bar{A}\bar{C} + \boldsymbol{\bar{A}C} + \boldsymbol{A\bar{C}})\bar{B} + (\bar{A}\bar{C} + A\bar{C} + AC)B$$
$$= (\bar{A}\bar{C} + \boldsymbol{\bar{A}C} + \boldsymbol{A\bar{C}})\bar{B} + \overline{(\overline{\bar{A}\bar{C}} + \overline{A\bar{C}} + \overline{AC})}B$$

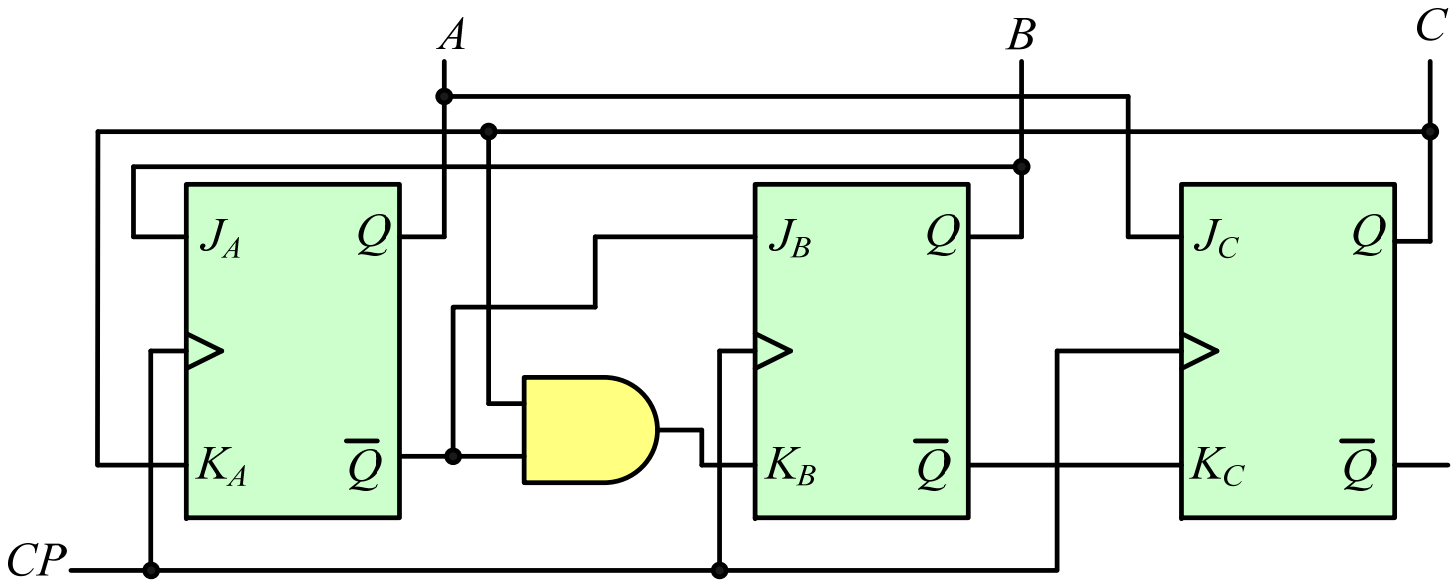
$$J_B = \bar{A}\bar{C} + \boldsymbol{\bar{A}C} + \boldsymbol{A\bar{C}} = \bar{C}\,(\text{or } \bar{A})$$
$$K_B = \overline{\bar{A}\bar{C} + A\bar{C} + AC} = \overline{(A + \bar{C})} = \bar{A}C$$

$$C(t+1) = \bar{A}BC + AB\bar{C} + ABC + \boldsymbol{\bar{A}\bar{B}C} + \boldsymbol{A\bar{B}\bar{C}}$$
$$= (AB + \boldsymbol{A\bar{B}})\bar{C} + (\bar{A}B + AB + \boldsymbol{\bar{A}\bar{B}})C$$
$$= (AB + \boldsymbol{A\bar{B}})\bar{C} + \overline{(\overline{\bar{A}B} + \overline{AB} + \overline{\boldsymbol{\bar{A}\bar{B}}})}C$$

$$J_C = AB + \boldsymbol{A\bar{B}} = A$$
$$K_C = \overline{\bar{A}B + AB + \boldsymbol{\bar{A}\bar{B}}} = \bar{B}$$

08 상태 방정식을 이용한 설계

- 회로도



$J_A = B$
$K_A = C$

$J_B = \overline{C}$ (또는 $\overline{A}$)
$K_B = \overline{A}C$

$J_C = A$
$K_C = \overline{B}$

08 상태 방정식을 이용한 설계

2. D 플립플롭을 사용한 상태 방정식

❖ D 플립플롭의 특성 방정식.

$$Q(t+1) = D$$

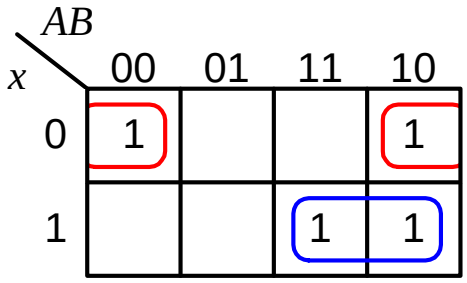
• **상태표**

현재상태		차기상태			
		x=0		x=1	
A	B	A	B	A	B
0	0	1	0	0	0
0	1	0	1	0	0
1	0	1	0	1	1
1	1	0	1	1	1

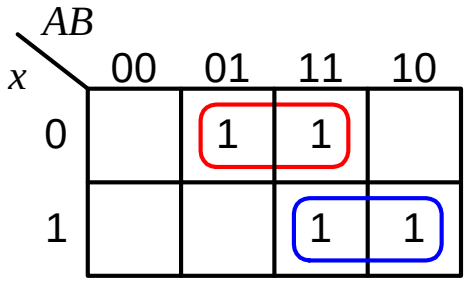
08 상태 방정식을 이용한 설계

- 상태 여기표

조합논리회로 입력			차기상태		플립플롭 입력	
입력	현재상태					
x	A	B	A	B	D_A	D_B
0	0	0	1	0	1	0
0	0	1	0	1	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	1
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	1	1	1
1	1	1	1	1	1	1



$D_A = \overline{B}\overline{x} + Ax$



$D_B = B\overline{x} + Ax$

08 상태 방정식을 이용한 설계

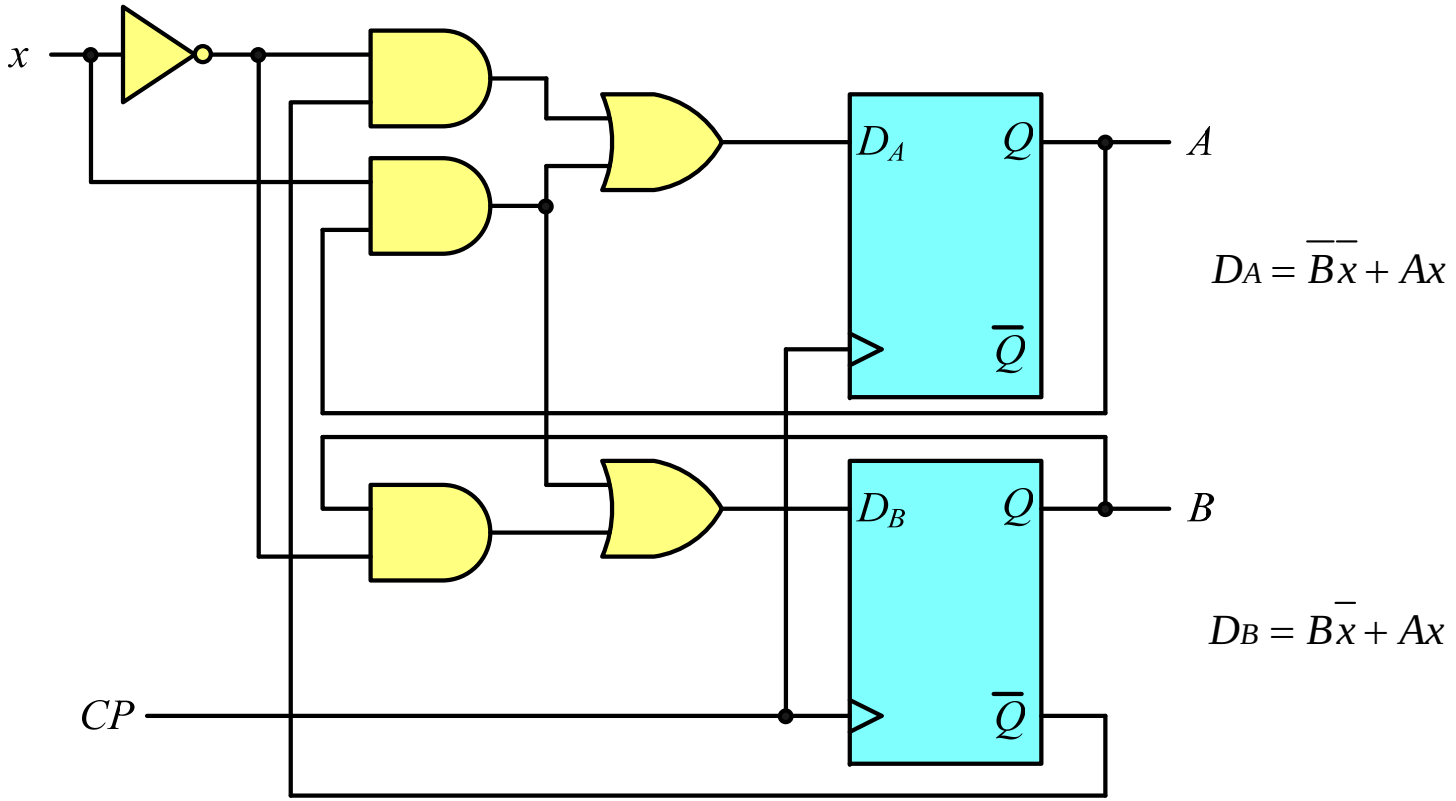
❖ **상태방정식을 특성 방정식의 형태로 변환한다.**

$$A(t+1) = \bar{A}\bar{B}x + A\bar{B}\bar{x} + A\bar{B}x + ABx$$
$$= (\bar{A} + A)\bar{B}\bar{x} + (\bar{B} + B)Ax \quad \Rightarrow \quad D_A = \bar{B}\bar{x} + Ax$$
$$= \bar{B}\bar{x} + Ax$$

$$B(t+1) = \bar{A}B\bar{x} + AB\bar{x} + A\bar{B}x + ABx$$
$$= (\bar{A} + A)B\bar{x} + (\bar{B} + B)Ax \quad \Rightarrow \quad D_B = B\bar{x} + Ax$$
$$= B\bar{x} + Ax$$

08 상태 방정식을 이용한 설계

- 순서논리회로(*D* 플립플롭을 이용하는 경우)



$$D_A = \overline{B}\,\overline{x} + Ax$$

$$D_B = B\overline{x} + Ax$$

09 디코더와 플립플롭을 사용한 설계

- 디코더는 n개의 입력 변수들에 대한 2^n개의 최소항을 출력하는 기능을 수행한다.
- 임의의 부울 함수는 곱의 합형으로 표현될 수 있기 때문에 각각의 곱을 구성하는 최소항들을 구성하는데 디코더를 사용하고 합을 구성하기 위하여 디코더 외에 OR 게이트 또는 NOR 게이트를 사용한다.
- 디코더의 출력이 정상 출력일 때는 OR 게이트를 사용하고, 보수 출력인 경우에는 NOR 게이트를 사용한다.

• **상태표**

현재상태		차기상태			
		x=0		x=1	
A	B	A	B	A	B
0	0	1	0	0	0
0	1	1	1	1	1
1	0	0	1	0	1
1	1	0	0	1	0

09 디코더와 플립플롭을 사용한 설계

- **상태 여기표(S-R 플립플롭 이용)**

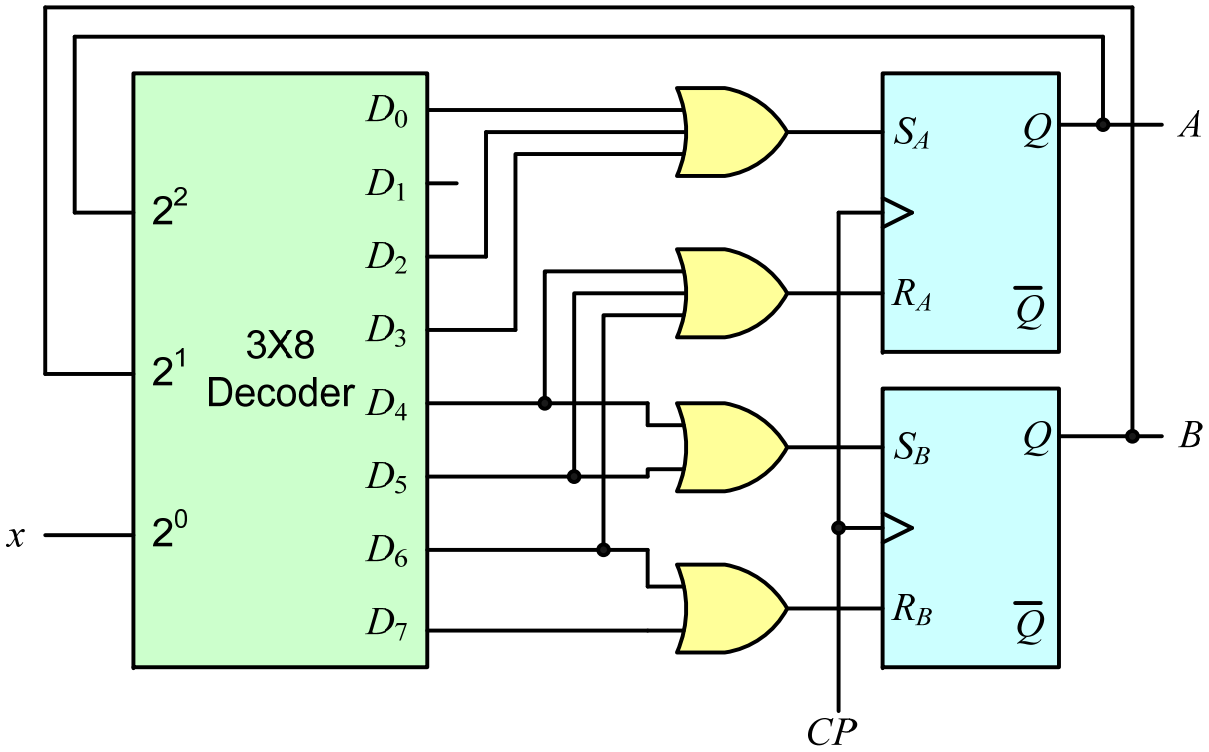
조합논리회로 입력			차기상태		조합논리회로 출력			
현재상태		입력			플립플롭 입력			
A	B	x	A	B	S_A	R_A	S_B	R_B
0	0	0	1	0	1	0	0	x
0	0	1	0	0	0	x	0	x
0	1	0	1	1	1	0	x	0
0	1	1	1	1	1	0	x	0
1	0	0	0	1	0	1	1	0
1	0	1	0	1	0	1	1	0
1	1	0	0	0	0	1	0	1
1	1	1	1	0	x	0	0	1

$$S_A(A,B,x) = \sum m(0,2,3) \qquad R_A(A,B,x) = \sum m(4,5,6)$$

$$S_B(A,B,x) = \sum m(4,5) \qquad R_B(A,B,x) = \sum m(6,7)$$

09 디코더와 플립플롭을 사용한 설계

- 순서논리회로를 설계하기 위하여 플립플롭은 2개가 필요하고, 디코더를 사용하여 조합논리회로를 구현하는 경우 1개의 3×8 디코더와 4개의 OR 게이트가 필요하다.

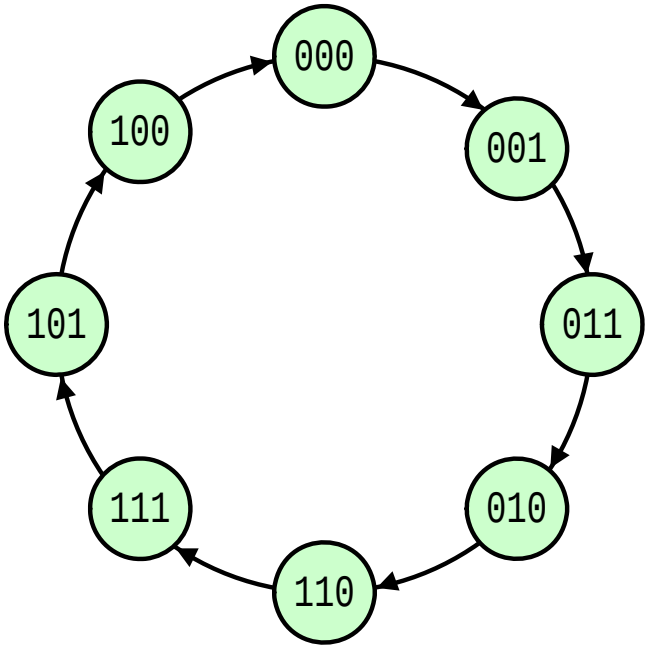


$$S_A(A,B,x) = \sum m(0,2,3)$$
$$R_A(A,B,x) = \sum m(4,5,6)$$
$$S_B(A,B,x) = \sum m(4,5)$$
$$R_B(A,B,x) = \sum m(6,7)$$

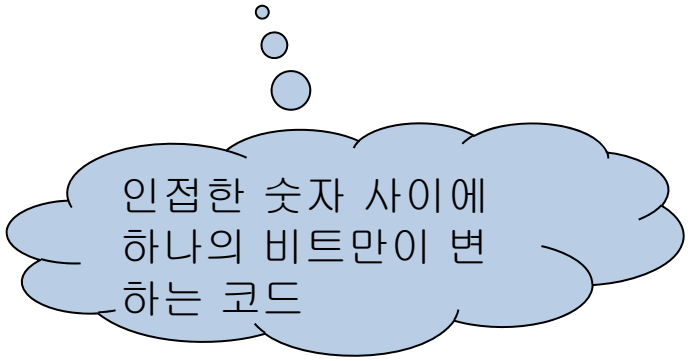
디코더와 S-R 플립플롭을 사용한 순서논리회로

09 디코더와 플립플롭을 사용한 설계

예제 9-4 *J-K* 플립플롭과 디코더를 사용하여 3비트 그레이 코드_카운터를 구현하여라.



상태도



09 디코더와 플립플롭을 사용한 설계

❑ 상태 여기표

현재상태			차기상태			플립플롭 입력					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	×	0	×	1	×
0	0	1	0	1	1	0	×	1	×	×	0
0	1	0	1	1	0	1	×	×	0	0	×
0	1	1	0	1	0	0	×	×	0	×	1
1	0	0	0	0	0	×	1	0	×	0	×
1	0	1	1	0	0	×	0	0	×	×	1
1	1	0	1	1	1	×	0	×	0	1	×
1	1	1	1	0	1	×	0	×	1	×	0

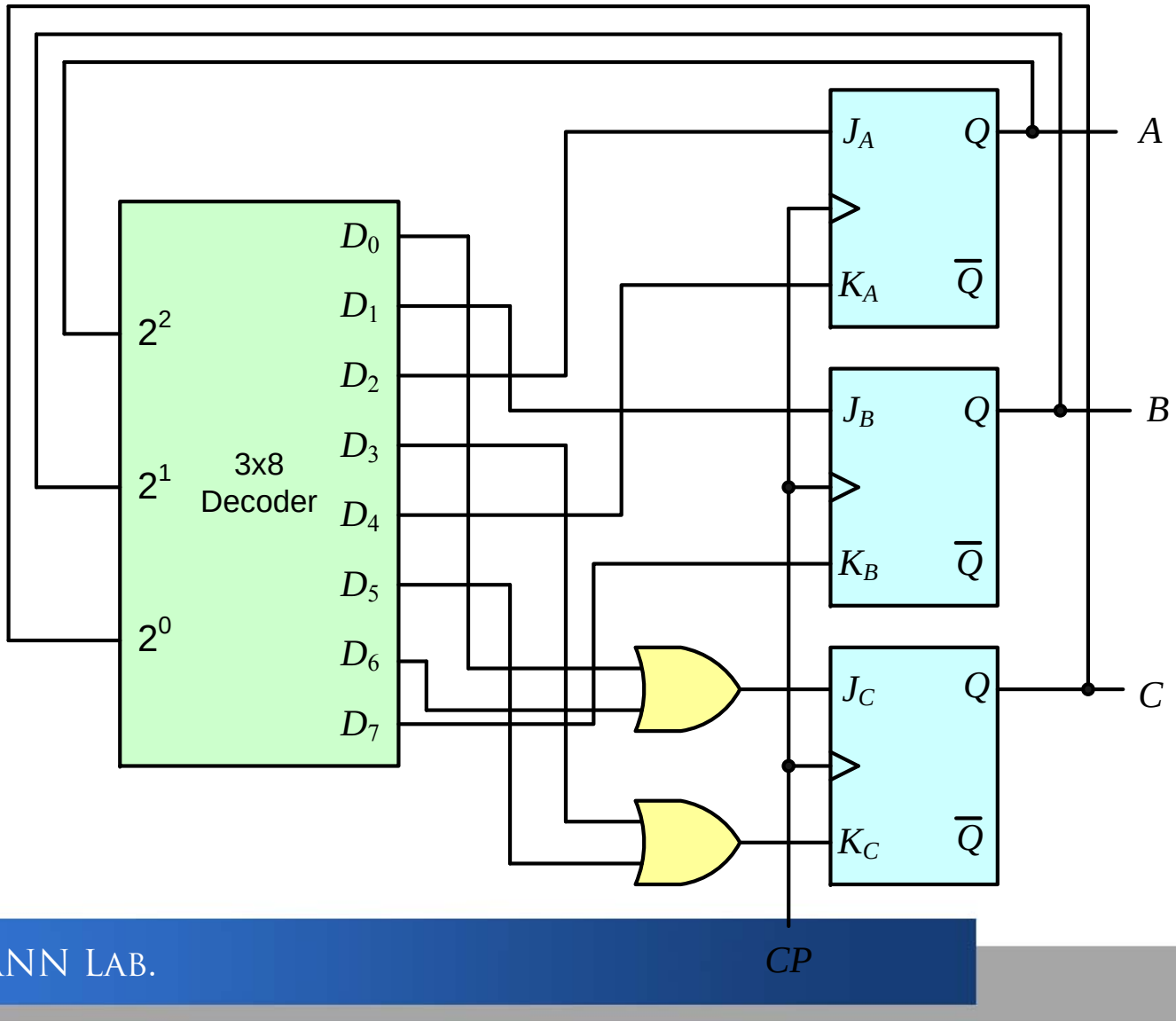
$$J_A(A,B,C)=\sum m(2) \qquad K_A(A,B,C)=\sum m(4)$$

$$J_B(A,B,C)=\sum m(1) \qquad K_B(A,B,C)=\sum m(7)$$

$$J_C(A,B,C)=\sum m(0,6) \qquad K_C(A,B,C)=\sum m(3,5)$$

09 디코더와 플립플롭을 사용한 설계

❑ 그레이 코드 카운터 회로도



$$J_A(A, B, C) = \sum m(2)$$
$$K_A(A, B, C) = \sum m(4)$$
$$J_B(A, B, C) = \sum m(1)$$
$$K_B(A, B, C) = \sum m(7)$$
$$J_C(A, B, C) = \sum m(0,6)$$
$$K_C(A, B, C) = \sum m(3,5)$$