디지털논리회로
(Digital Logic Circuit)

- Chapter 3

조합회로 시스템 설계 과정

1 단계: 각 **입력과 출력**을 2진으로 표현하라.

1.5 단계: 필요하면, 문제를 더 작은 부(sub) 문제로 나누어라.

2 단계: 설계 사양을 **진리표 혹은 대수 식**으로 형식화(formalize)해라.

3 단계: 서술을 **간단히** 하라.

4 단계: 설계 **목표와 제약에 근거**하여,

사용 가능한 부품으로 시스템을 **구현**한다.

Chapter 3 Karnaugh Map (K-Map)

< 이 장의 핵심 >

3.1 K-map 소개

3.2 K-map 을 사용한 최소 SOP (Sum of Product) 표현

3.3 Don't care

< 응용 >

3.4 POS (Product of Sum)

3.5 5변수, 6변수를 위한 맵

3.6 다중 출력 문제

Chapter 3 Karnaugh Map (K-Map)

3.1 K-map 소개

3.2 K-map 을 사용한 최소 SOP (Sum of Product) 표현

3.3 Don't care

3.4 POS (Product of Sum)

3.5 5변수, 6변수를 위한 맵

3.6 다중 출력 문제

어려운 간소화 문제.. K-map?

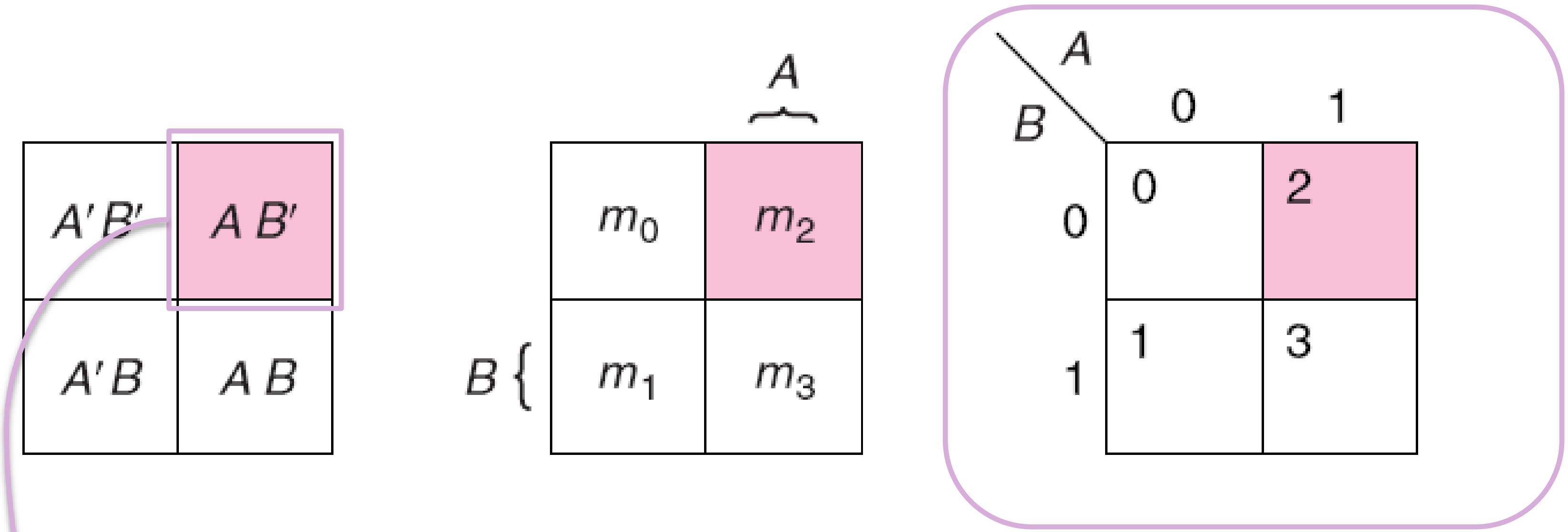
- 어디서부터 어떻게 시작할지 모르겠는데..
  - ➔ 시행착오와 경험
- 어떤 식을 어떤 순서로 사용해야 하는 거지?
  - ➔ 알고리즘 같은 것 없음.. 이것 저것 시도해 보기

K-map: 그래픽적 방법 제공, 6변수 이하까지는 무난히~

- 간소화가 됐는지 안됐는지, 더 해야 하는지 그만둬도 되는지 어떻게 할 수 있지?
  - ➔ 사실은 잘 모른다.

K-map: 100% 보장은 어렵지만 대부분의 경우 minimization 달성이 용이!

K-map 어떻게 생겼나 – *two variables*



A=1, B=0에 해당하는 minterm

함수로부터 K-map 그리기

B \ A	0	1
0	1	
1		1

$f(a, b) = \Sigma m\,(0, 3)$

B \ A	0	1
0	1	X
1		1

$g(A, B) = \Sigma m\,(0, 3) + \Sigma d\,(2)$

K-map 어떻게 생겼나 – *three variables*

C \ AB	A′B′ 00	A′B 01	AB 11	AB′ 10
C′ 0	A′B′C′	A′BC′	ABC′	AB′C′
C 1	A′B′C	A′BC	ABC	AB′C

C \ AB	00	01	11	10
0	0	2	6	4
1	1	3	7	5

P9a. ab + ab’ = a 를 위해서..

m_0+m_1: A′B′C′ + A′B′C =A′B′
m_4+m_6: AB′C′ + ABC′ =AC′
m_7+m_5: ABC + AB′C =AC
m_0+m_4: A′B′C′ + AB′C′ =B′C′
m_1+m_5: A′B′C + AB′C =B′C

…

➔ 인접하는 사각형들끼리 P9a를 적용시킬 수 있도록..

K-map 어떻게 생겼나 – *three variables*

AB / C	A′B′ 00	A′B 01	AB 11	AB′ 10
C′ 0	A′B′C′	A′BC′	ABC′	AB′C′
C 1	A′B′C	A′BC	ABC	AB′C

AB / C	00	01	11	10
0	0	2	6	4
1	1	3	7	5

AB / C	00	01	11	10
0	1			
1	1			

A′B′

AB / C	00	01	11	10
0			1	1
1				

AC′

AB / C	00	01	11	10
0				
1			1	1

AC

AB / C	00	01	11	10
0	1			1
1				

B′C′

AB / C	00	01	11	10
0				
1	1			1

B′C

P9a를
그래픽적으로 표현

➔ 인접하는 사각형들끼리 P9a를 적용시킬 수 있도록..

K-map 어떻게 생겼나 – *three variables*

BC \ A	0	1
00		
01		
11		
10		

BC \ A	0	1
00	1	
01	1	
11		
10		

A′B′

BC \ A	0	1
00		
01		1
11		1
10		

AC

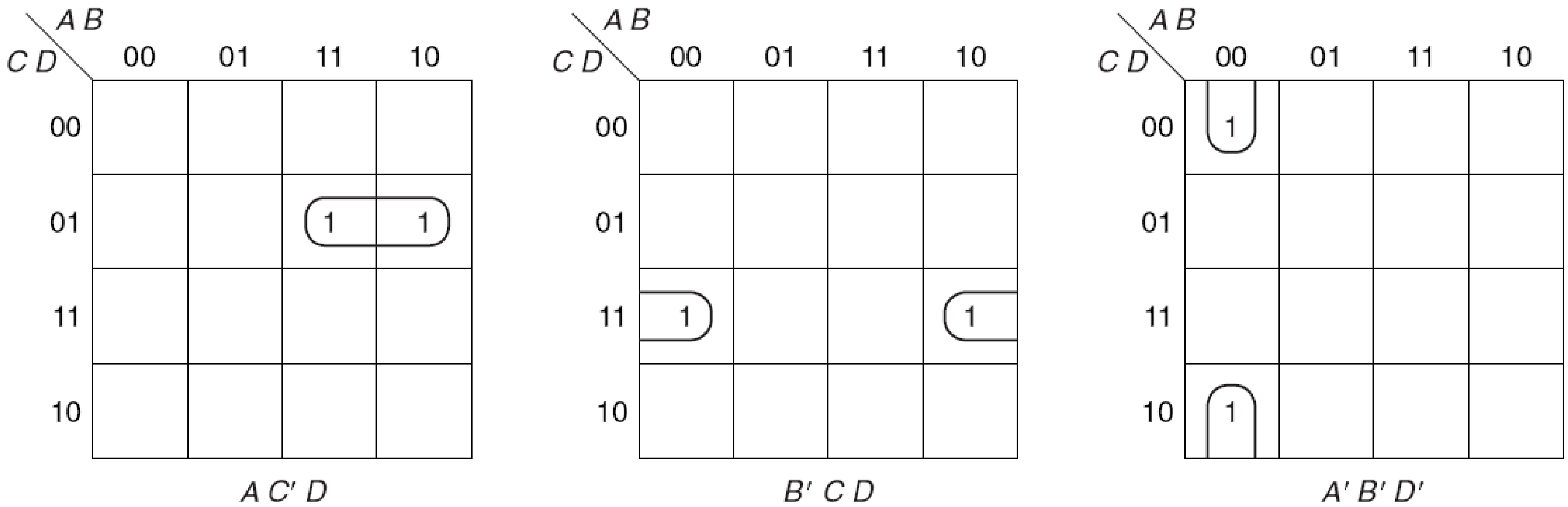
수직으로 그려도 되고~ 편한 대로.

K-map 어떻게 생겼나 – *four variables*

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

CD \ AB	00	01	11	10
00	$A'B'C'D'$	$A'BC'D'$	$ABC'D'$	$AB'C'D'$
01	$A'B'C'D$	$A'BC'D$	$ABC'D$	$AB'C'D$
11	$A'B'CD$	$A'BCD$	$ABCD$	$AB'CD$
10	$A'B'CD'$	$A'BCD'$	$ABCD'$	$AB'CD'$

K-map 에서 grouping 하기 *– 2개씩 묶기*

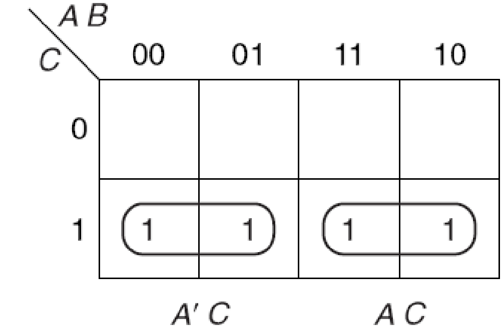


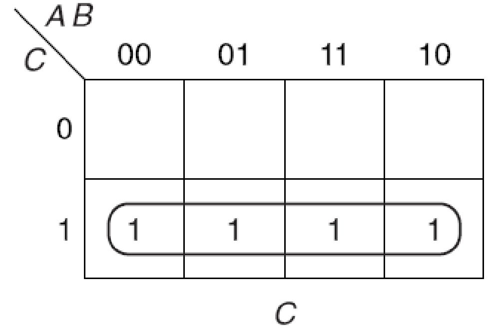
$$m_{13} + m_9: \quad ABC'D + AB'C'D = AC'D$$

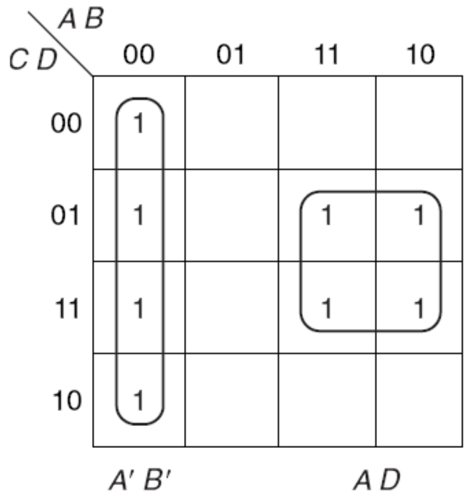
$$m_3 + m_{11}: \quad A'B'CD + AB'CD = B'CD$$

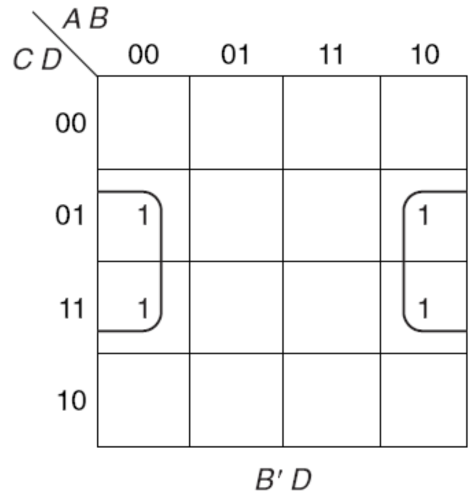
$$m_0 + m_2: \quad A'B'C'D' + A'B'CD' = A'B'D'$$

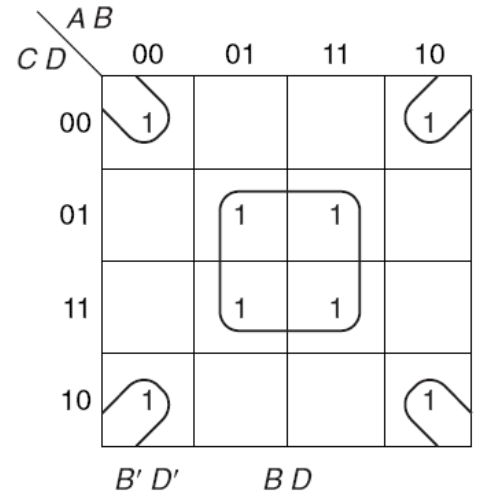
K-map 에서 grouping 하기 *– 4개씩 묶기*



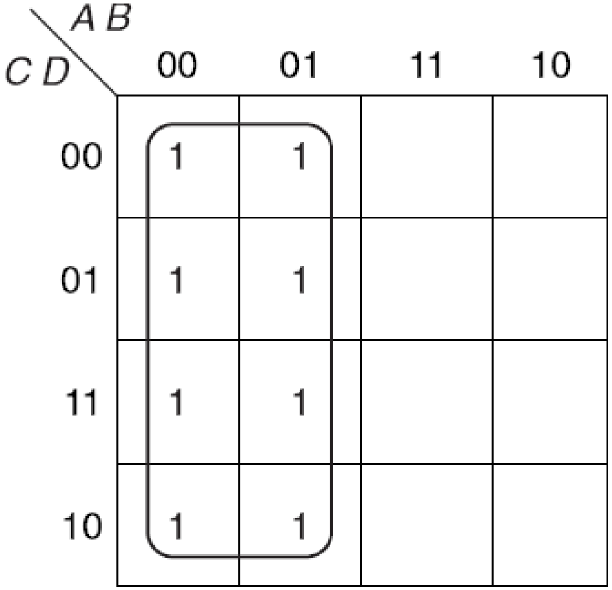




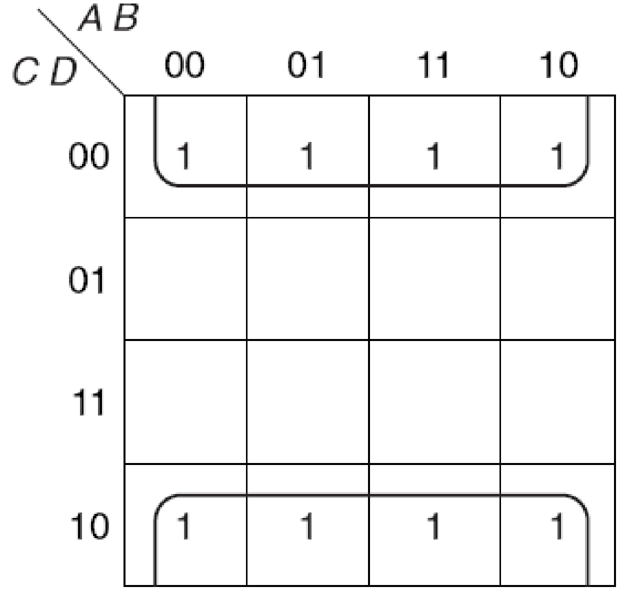




K-map 에서 grouping 하기 *– 8개씩 묶기*



A′



D′

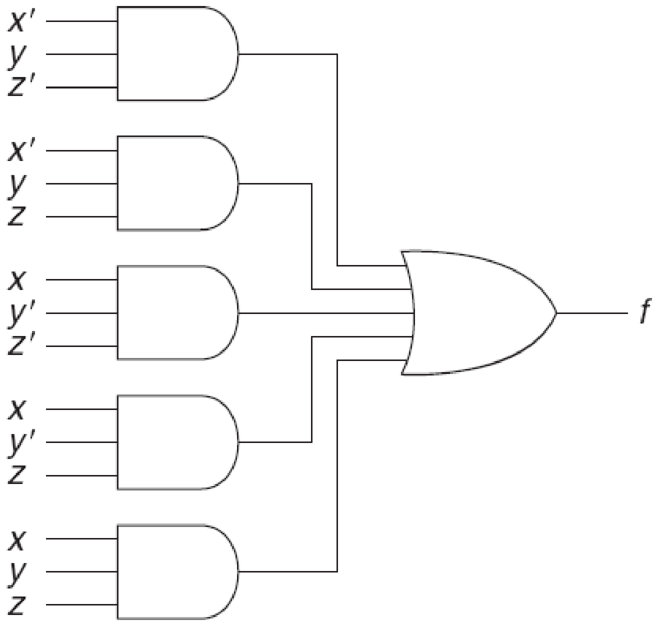
Q 그룹으로 묶기

CD \ AB	00	01	11	10
00				
01	1		1	1
11	1		1	1
10				

CD \ AB	00	01	11	10
00	1			1
01	1			1
11	1			
10	1			

K-map 그리기

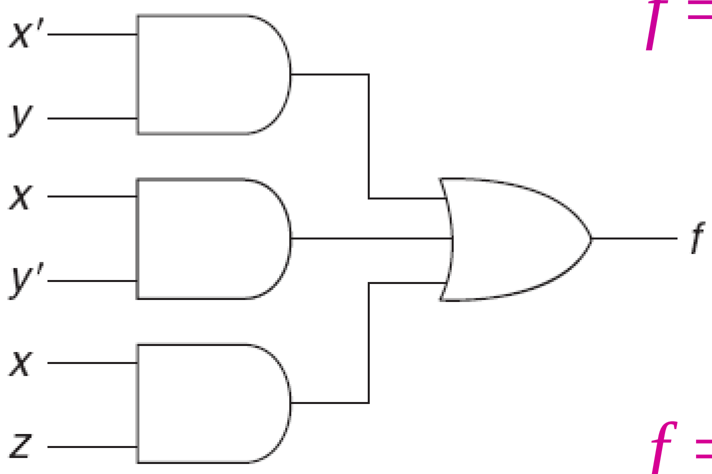
$$f = x'yz' + x'yz + xy'z' + xy'z + xyz$$



z \ x y	00	01	11	10
0		1		1
1		1	1	1

K-map 결과와 비교

- 최소 곱의 합(SOP):



Minimum sum of product implementation of *f*.

$$f = x'yz' + x'yz + xy'z' + xy'z + xyz$$

$$f = x'y + xy' + xz$$

K-map의 몇 가지 용어들

내포항 (*Implicant*)

: 함수를 SOP로 표현했을 때 각 product term

CD \ AB	00	01	11	10
00	1		1	
01			1	
11	1	1	1	1
10				

Minterm인 Implicant

A'B'C'D'
A'B'CD
A'BCD
ABC'D'
ABC'D
ABCD
AB'CD

K-map의 몇 가지 용어들

내포항 (*Implicant*)

: 함수를 SOP로 표현했을 때 각 product term

2개짜리 그룹 Implicant

A'CD

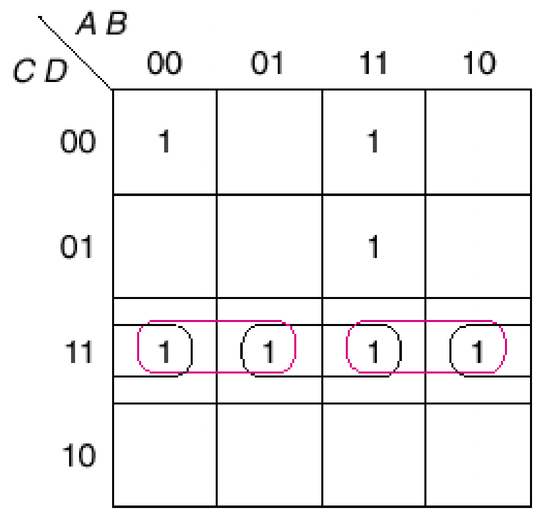
BCD

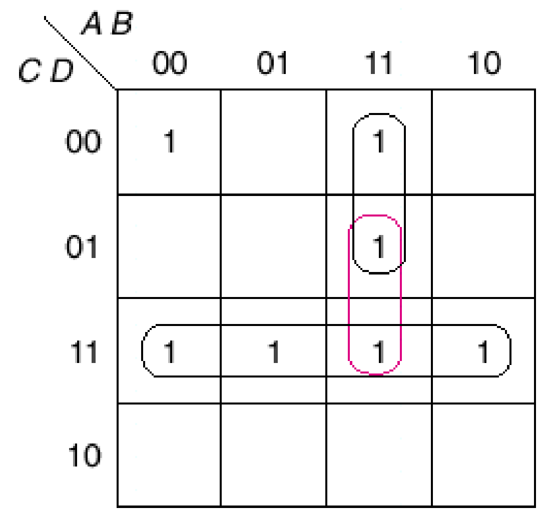
ACD

B'CD

ABC'

ABD



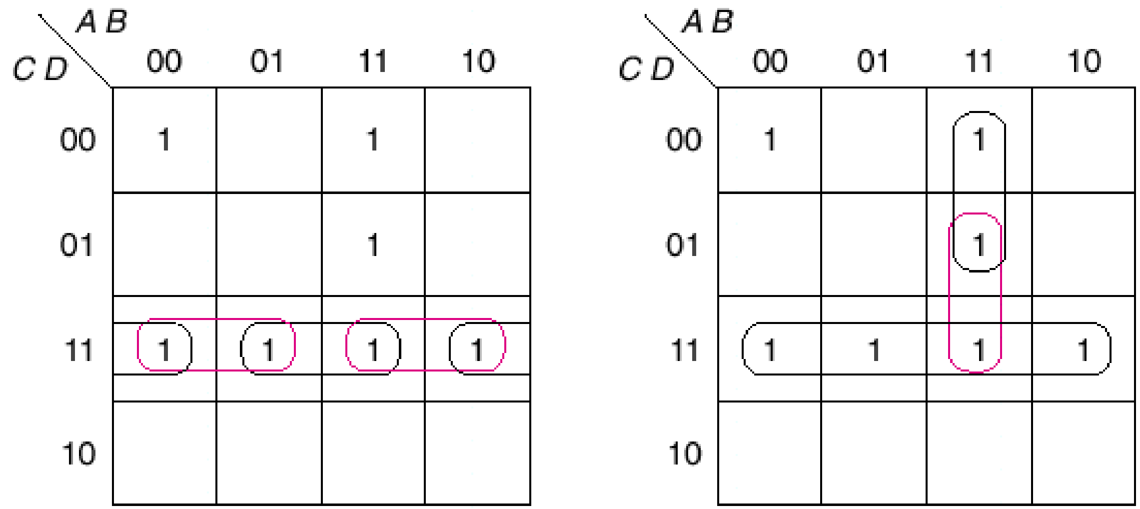


K-map의 몇 가지 용어들

내포항 (*Implicant*)

: 함수를 SOP로 표현했을 때 각 product term

4개짜리 그룹 Implicant
CD



1, 2, 4, 8..(2의 지수승)개의 1로 이루어진 사각형

K-map의 몇 가지 용어들

커버한다 (*Cover*)

Implicant들이 함수 F를 ***Cover*** 한다.

= 가능한 14개의 Implicant 들 중에 ***..*를 고르면 F를 표현할 수 있다.

= 가능한 14개의 Implicant 들 주에 ***..* 를 고르면 K-map의 1이 모두 포함된다.

CD \ AB	00	01	11	10
00	1		1	
01			1	
11	1	1	1	1
10				

최소항	2개로 구성된 그룹	4 개로 구성된 그룹
A'B'C'D'	*A'CD*	*CD*
A'B'CD	*BCD*	
A'BCD	*ACD*	
ABC'D'	*B'CD*	
ABC'D	*ABC'*	
ABCD	*ABD*	
AB'CD		

총 14개의 내포항

K-map의 몇 가지 용어들

커버한다 (*Cover*)

Implicant가 minterm m_0 .. m_N을 ***Cover*** 한다.

Ex) Implicant ACD는 m11과 m15를 Cover 한다.

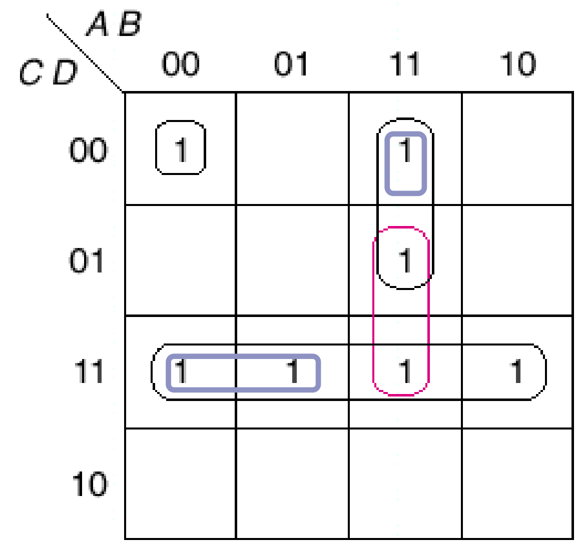
CD \ AB	00	01	11	10
00	1		1	
01			1	
11	1	1	1	1
10				

K-map의 몇 가지 용어들

주 내포항 (Prime Implicant : PI)

: 다른 내포항에 완전히 포함되지 않는 하나의 내포항

Prime Implicant:
A'B'C'D', ABC', ABD, CD



단순화를 위해서는 Implicant가 아닌 **Prime Implicant**를 찾아야함

1) Implicant를 포함하는 Prime Implicant가 있다면 일단 literal 개수가 적음

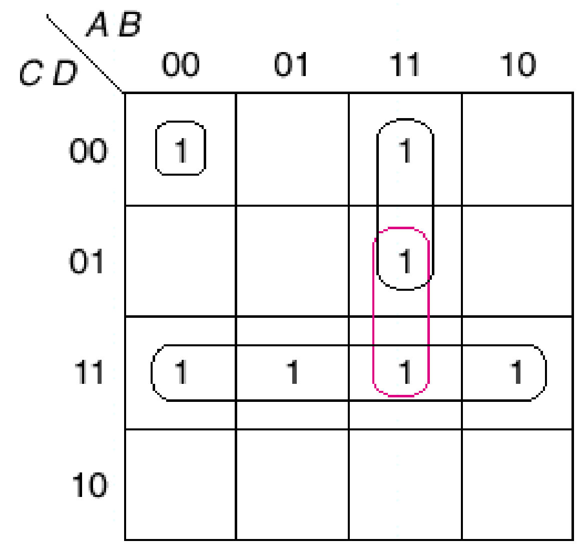
2) 1을 더 많이 포함하는 Prime Implicant위주로 찾으면 결국 term 개수도 줄어듬

K-map의 몇 가지 용어들

필수 주 내포항 (Essential Prime Implicant : EPI)

: 다른 주 내포항에 포함되지 않는 적어도 1개의 1을 포함하는 주 내포항 <u>→ 다른 PI로는 절대 커버 불가능한 1 찾기</u>

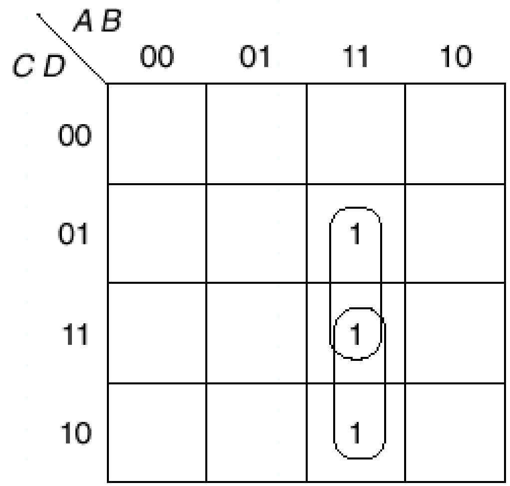
Prime Implicant:
A'B'C'D', ABC', ABD, CD



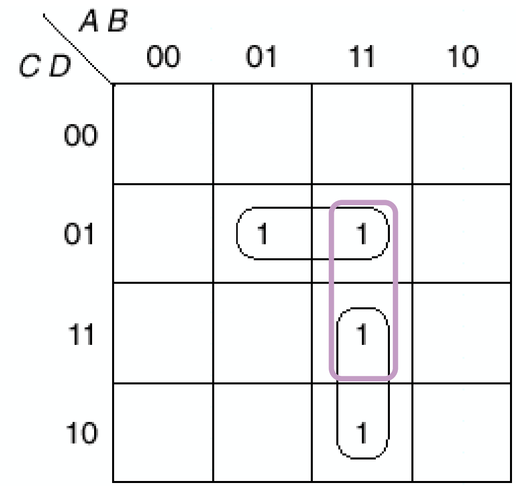
Q. 위의 PI들 중 EPI가 아닌 것은?

K-map의 몇 가지 용어들

예제 3.4



G



H

- 함수 G
  - *최소식 G = ABD + ABC (EPI로만 구성)*
- 함수 H
  - 최소식 *H = BC ′D + ABC (EPI로만 구성)*
  - EPI로 다 커버했으므로 EPI가 아닌 PI ABD 는 사용하지 않아도 됨

Chapter 3 Karnaugh Map (K-Map)

K-map 통한 최소 SOP 찾기

1. **모든 필수 주 내포항 (EPI)**들을 찾음.
   - 맵 상에서 필수 주 내포항들을 묶음
   - 필수 주 내포항으로 만드는 최소항(minterm)들에 * 표시를 한다.
     - 일반적으로 가장 고립된 1들로부터 시작하는 것이 빠르다.

2. **함수를 커버하는 '충분한' 다른 주 내포항 (PI)**들을 찾음.

   (2 가지 기준)
   - 선택된 주 내포항 (PI)에 의해 **된수록 많은 새로운(아직 커버되지 않은) 1을 커버하는 주 내포항**을 선택.
   - 고립되고 커버되지 않은 1을 남겨놓지 않도록 함.

K-map 통한 최소 SOP 찾기

예제 3.5

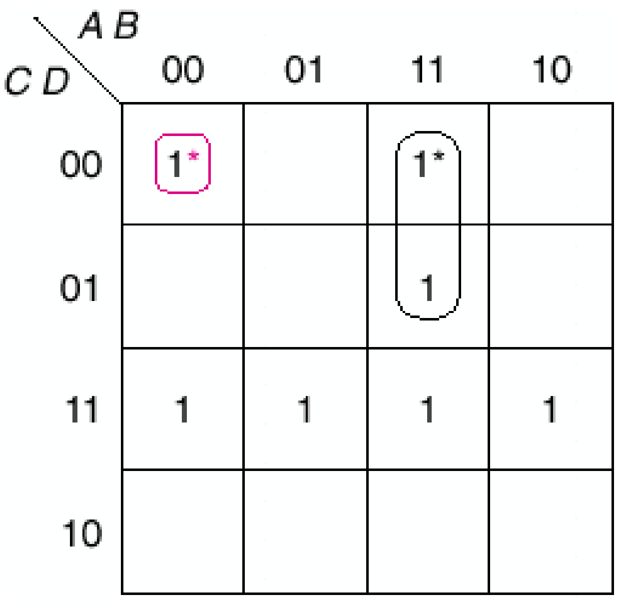
CD \ AB	00	01	11	10
00	1		1	
01			1	
11	1	1	1	1
10				

1.EPI 찾기
(고립된 것부터 찾기)

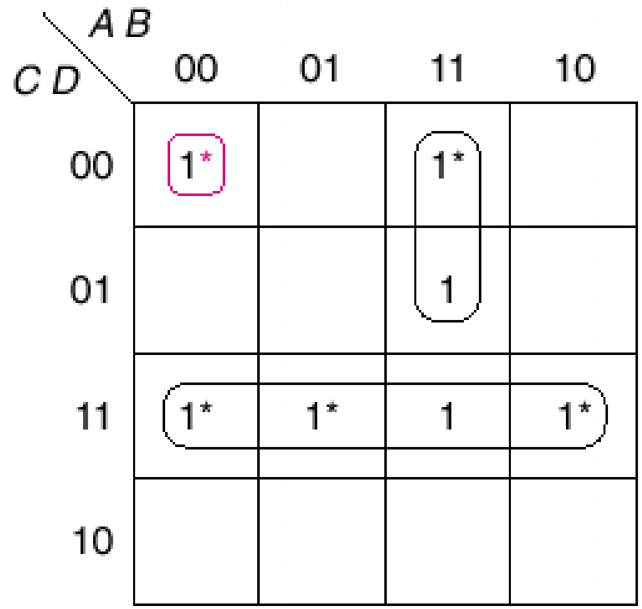
2. PI찾기

K-map 통한 최소 SOP 찾기

예제 3.5



$F = A'B'C'D' + ABC' + \ldots$



$F = A'B'C'D' + ABC' + CD$

1*들은 다른 PI로는 절대 커버될 수 없는 minterm 들이므로
Group을 EPI로 만들어주는 minterm들이다.

→ EPI로 모두 해결된 케이스, 2단계 필요 없음.

K-map 통한 최소 SOP 찾기

예제 3.6 EPI 와 PI로 해결

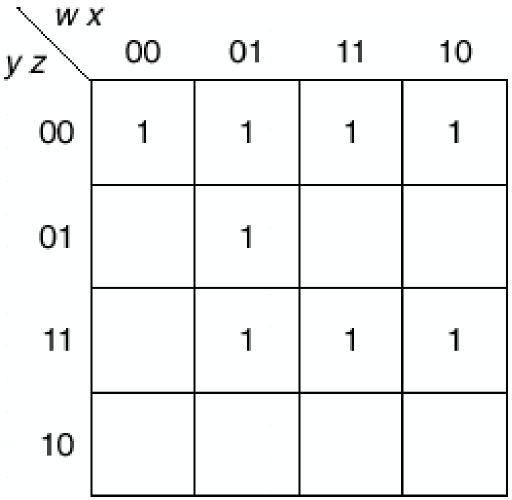
yz \ wx	00	01	11	10
00	1	1	1	1
01		1		
11		1	1	1
10				

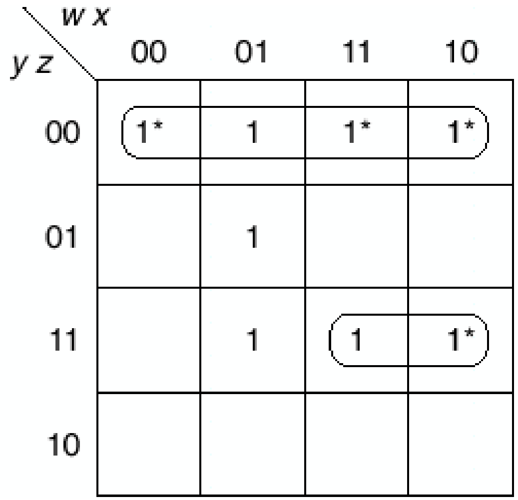
1. EPI 찾기
(고립된 것부터 찾기)

2. PI찾기

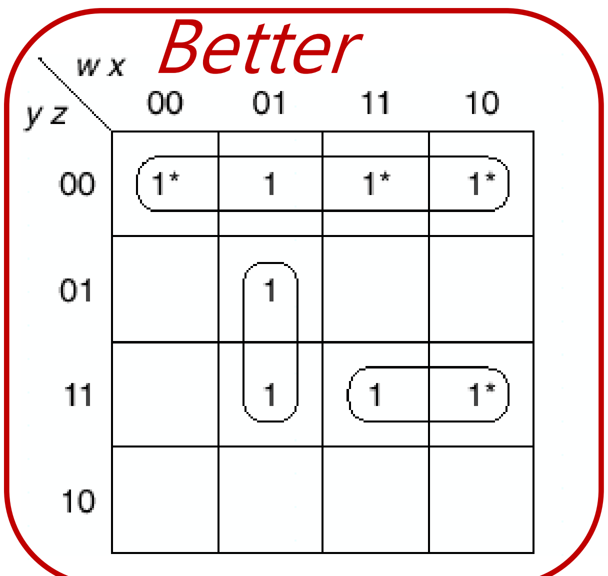
K-map 통한 최소 SOP 찾기

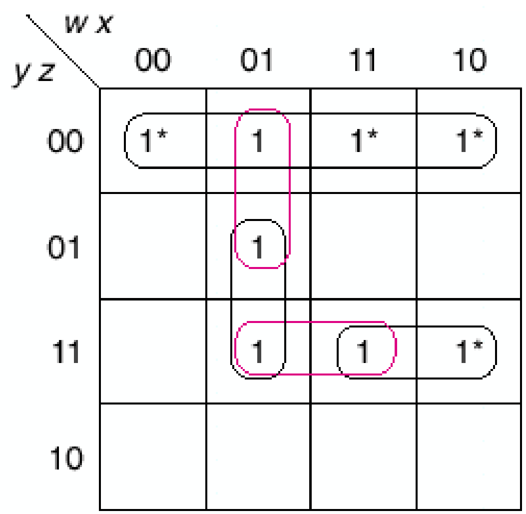
예제 3.6 EPI 와 PI로 해결





- 필수 주 내포항(EPI)
  - 가장 고립된 m_{11} -> wyz
  - m_0, m_{12}, m_8 -> $y'z'$





- 주 내포항(PI)
  - 남은 2개의 1은 $w'xz$ 에 의해 커버

- **최소곱의 합 식**

$$f = y'z' + w\,y\,z + w'xz$$

K-map 통한 최소 SOP 찾기

예제 3.8

$f(a, b, c, d) = \Sigma m(0, 2, 4, 6, 7, 8, 9, 11, 12, 14)$

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

cd \ ab	00	01	11	10
00	1	1	1	1
01				1
11		1		1
10	1	1	1	

cd \ ab	00	01	11	10	
00	1	1	1	1	m_8
01				1	
11		1*		1*	
10	1*	1	1*		

1. **모든 필수 주 내포항 (EPI)**들을 찾음.
   - 맵 상에서 필수 주 내포항들을 묶음
   - 필수 주 내포항으로 만드는 최소항(minterm)들에 * 표시를 한다.
     - 일반적으로 가장 고립된 1들로부터 시작하는 것이 빠르다.

2. **함수를 커버하는 '충분한' 다른 주 내포항 (PI)**들을 찾음.

   (2 가지 기준)
   - 선택된 주 내포항 (PI)에 의해 **될수록 많은 새로운(아직 커버되지 않은) 1을 커버하는 주 내포항**을 선택.
   - 고립되고 커버되지 않은 1을 남겨놓지 않도록 함.

- 필수 주 내포항: $a'd' + bd' + a'bc + ab'd$
- 1개의 1(m_8)이 남음
- 4개로 이루어진 그룹($c'd'$)과 2개로 이루어진 그룹($ab'c'$)에 커버
- 최소 식 $f = a'd' + bd' + a'bc + ab'd + c'd'$

K-map 통한 최소 SOP 찾기

예제 3.10 특이한 경우

CD \ AB	00	01	11	10
00		1		
01		1	1	1
11	1	1	1	
10			1	

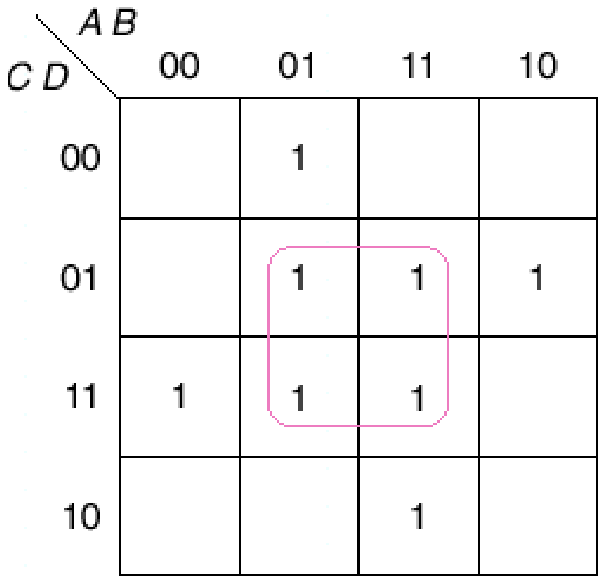
이렇게 묶고 싶죠?
그런데 MAP방법 1을 생각해 보면
EPI부터 찾아야 함.
EPI인가??

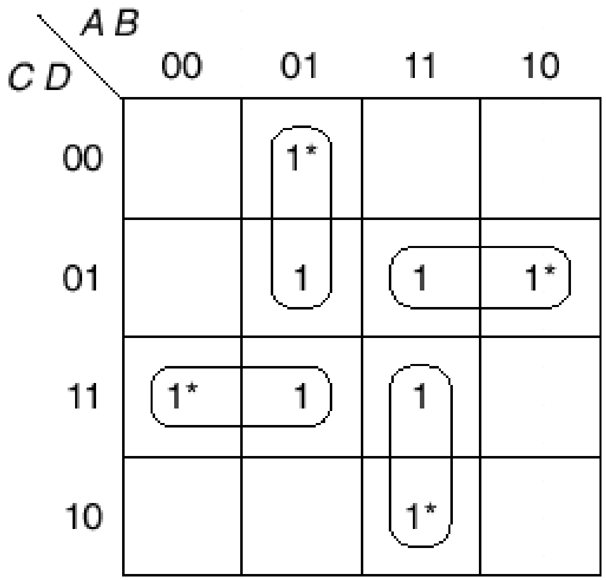
1. EPI 찾기
(고립된 것부터 찾기)

2. PI찾기

K-map 통한 최소 SOP 찾기

예제 3.10 특이한 경우





- 최소 해

$$G = A'BC' + A'CD + ABC + AC'D$$

K-map 통한 최소 SOP 찾기

예제 3.11 최소 SOP가 여러 개 있을 수도 있음

$g(w, x, y, z) = \Sigma m(2, 5, 6, 7, 9, 10, 11, 13, 15)$

yz \ wx	00	01	11	10
00				
01		1	1	1
11		1	1	1
10	1	1		1

yz \ wx	00	01	11	10
00				
01		1*	1	1*
11		1	1	1
10	1	1		1

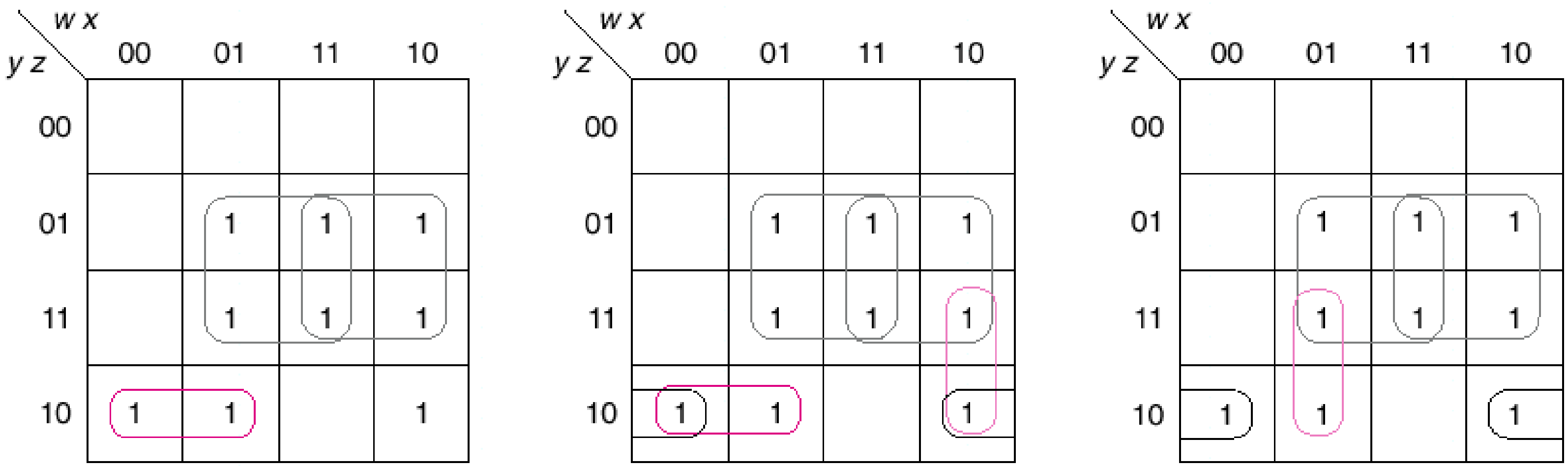
$g = xz + wz + \ldots$

더 이상의 EPI는 없는 것 같으므로 PI로 남은 1을 커버해야 하는데...
2개짜리 그룹 2개면 가능할 것으로 보임

K-map 통한 최소 SOP 찾기

예제 3.11 최소 SOP가 여러 개 있을 수도 있음

$g(w, x, y, z) = \Sigma m(2, 5, 6, 7, 9, 10, 11, 13, 15)$



2개짜리 그룹 2개를 만드는 다수의 방법들

➔ 다수의 최소 SOP 해

$g = xz + wz + w'yz' + wx'y$

$g = xz + wz + w'yz' + x'yz'$

$g = xz + wz + x'yz' + w'xy$

연습

연습문제 2

f=

CD \ AB	00	01	11	10
00				1
01	1	1	1	
11	1	1	1	1
10	1	1		

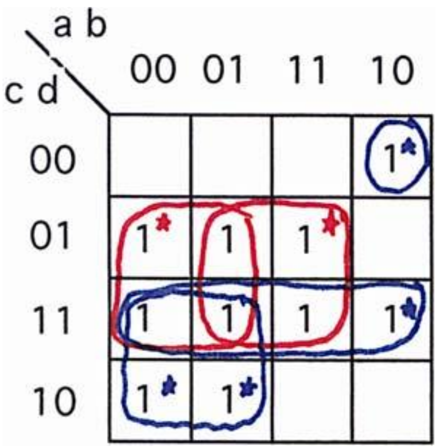
h=

CD \ AB	00	01	11	10
00		1	1	1
01		1	1	
11		1	1	
10	1	1		1

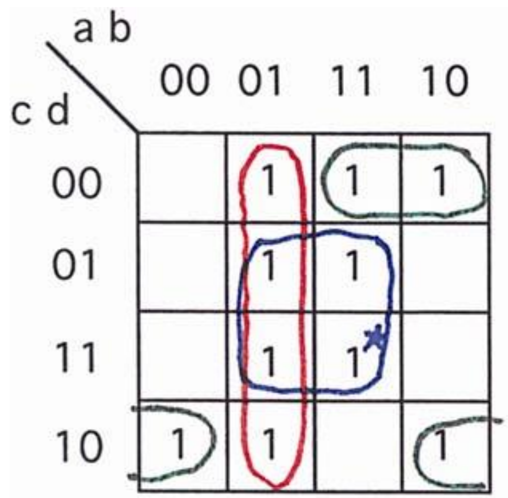
연습

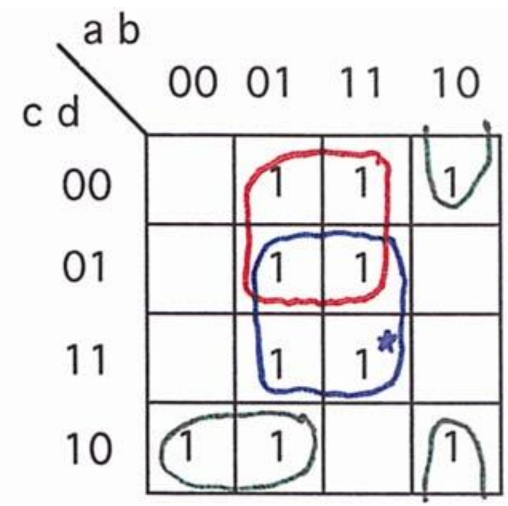
연습문제 2

f=



h=





f = a b' c' d' + c d + a' c + a' d + b d

h = b d + a' b + b' c d' + a c' d'

h = b d + b c' + a' c d' + a b' d'

Chapter 3 Karnaugh Map (K-Map)

3.1 K-map 소개

3.2 K-map 을 사용한 최소 SOP (Sum of Product) 표현

3.3 Don’t care

3.4 POS (Product of Sum)

3.5 5변수, 6변수를 위한 맵

3.6 다중 출력 문제

함수가 don’t care를 포함한다면??

PI(Prime Implicant)와 EPI(Essential Prime Implicant)의 정의를 조금만 바꾸어보자.

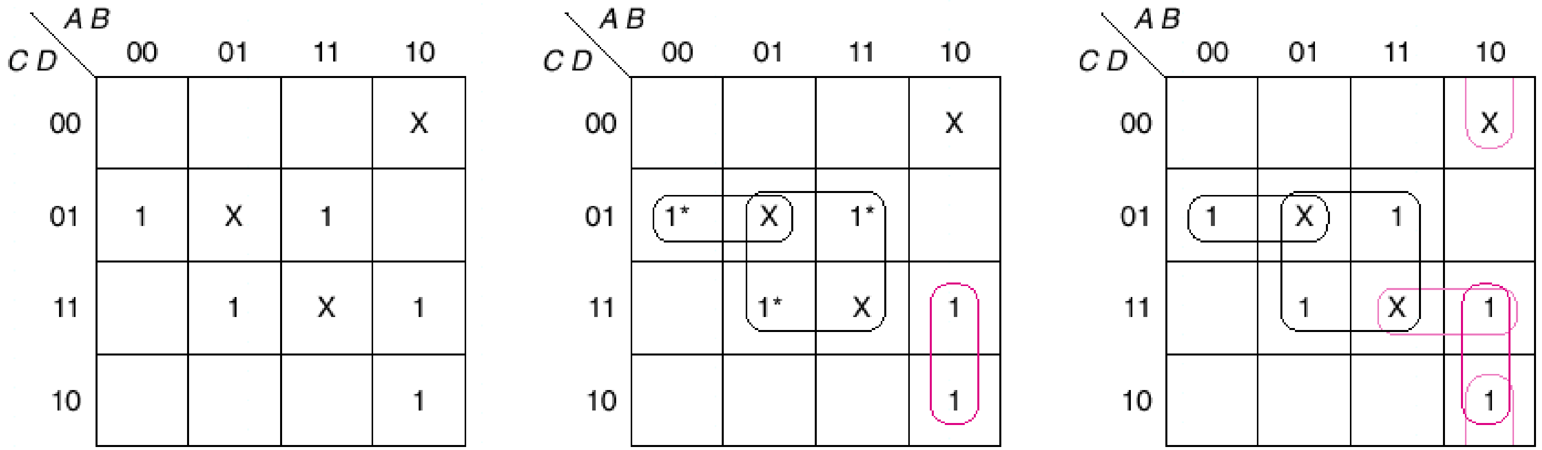
- PI: 다른 더 큰 사각형에 포함되지 않은 1, 2, 4, 8, .. 개의 1 또는 ×의 사각형이다.
  - ×(don’t care)도 1과 같이 동등하게
  - 모두 x로만 이루어진 PI는 의미 없음

- EPI: 다른 PI에 의해 커버되지 않는 1을 적어도 한 개를 커버하는 PI.
  - ×(don’t care)는 EPI로 만들 수 있는 조건에서 빠짐.

** 주의:
grouping의 목적은 1을 모두 cover하는 것. X는 굳이 cover안되어도 됨

Don’t care를 포함하는 K-map

예제 3.20 $F(A, B, C, D) = \Sigma m(1, 7, 10, 11, 13) + \Sigma d(5, 8, 15)$



X를 굳이 묶을 필요 없음

- 최소해 (가운데 맵)

  $F = BD + A'C'D + AB'C$

- 모든 무정의를 1로 고려하는 경우 (오른쪽 맵)

  $F = BD + A'C'D + AB'C + AB'D'$

  $F = BD + A'C'D + ACD + AB'D'$

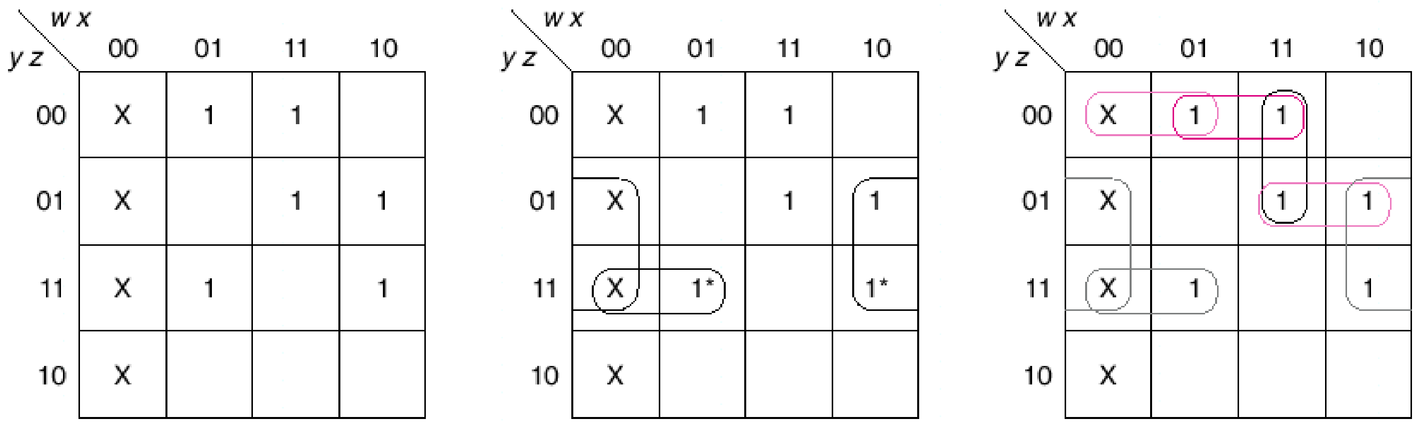
- 모든 무정의를 ‘0’으로 고려한 경우

  $F = A'B'C'D + A'BCD + ABC'D + AB'C$

← *Don’t care로 유연하게 최적화 하는 것보다는 해가 복잡해짐*

Don’t care를 포함하는 K-map

예제 3.21

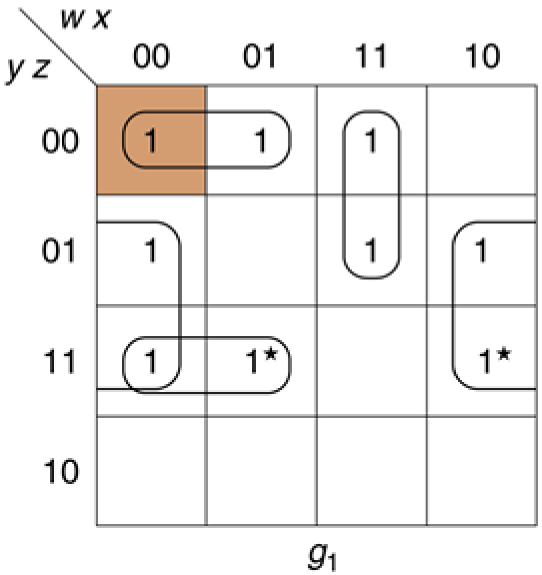


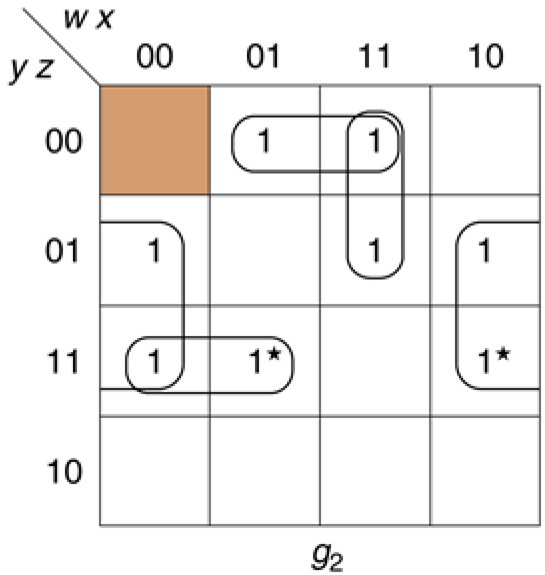
- 2 개의 EPI ***x’z*** 와 ***w’yz*** (가운데 맵)
- 4 개의 don’t care로 이루어진 *w’x’*는 PI지만, EPI는 아님
- 모두 무정의로만 이루어진 주 내포항은 사용할 수 없음

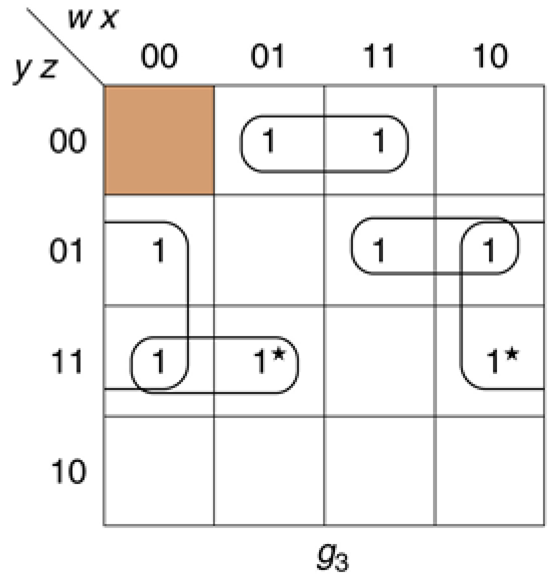
•남은 1을 묶는 여러 가지 방법(오른쪽 맵)

Don’t care를 포함하는 K-map

예제 3.21







- 최소화 해 $g_1 = x'z + w'yz + w'y'z' + wxy'$ ($m_0 = 1$)

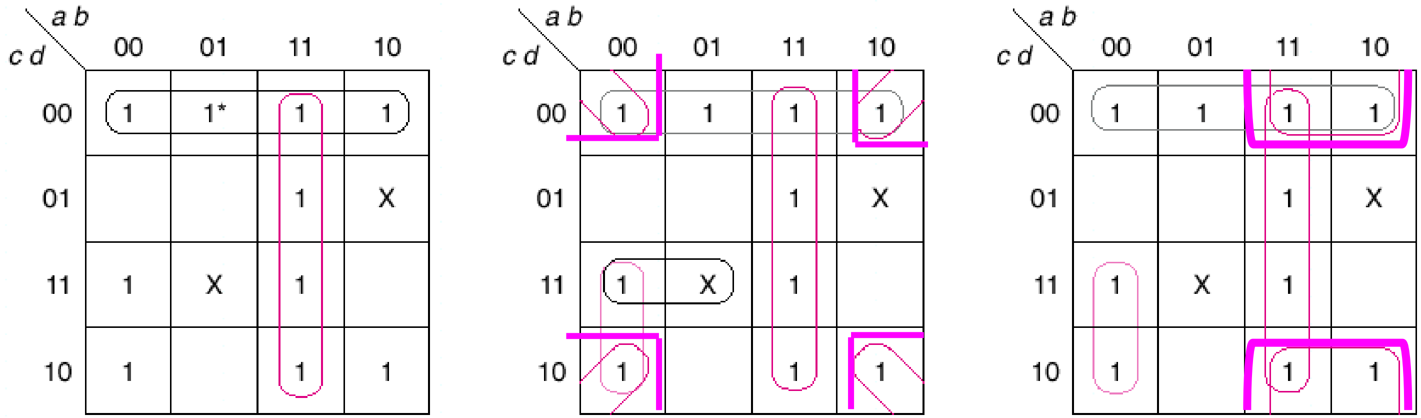
  $g_2 = x'z + w'yz + xy'z' + wxy'$ ($m_0 = 0$)

  $g_3 = x'z + w'yz + xy'z' + wy'z$ ($m_0 = 0$)

  g_2 와 g_3 는 대수학적으로 같은 함수이지만 g_1 과는 다르다.

Don’t care를 포함하는 K-map

예제 3.22



- 첫번째 맵 : 유일한 EPI ***c'd'***와 *ab*, 3 개의 1이 남음
- 두번째 맵 : *b'd'*를 이용한 2개의 해(g_1, g_2)
- 세번째 맵 : *ad'*를 이용한 해

$$g_1 = c'd' + ab + b'd' + a'cd$$

$$g_2 = c'd' + ab + b'd' + a'b'c$$

$$g_3 = c'd' + ab + ad' + a'b'c$$

- 동일성을 검증 : don’t care들이 처리된 값을 표로 구현 그림에서.. 1로 여겨져서 cover가 되었는지 유무에 따라..
- $g_1 \neq g_2 = g_3$

	m_7	m_9
g_1	1	0
g_2	0	0
g_3	0	0

Chapter 3 Karnaugh Map (K-Map)

3.1 K-map 소개

3.2 K-map 을 사용한 최소 SOP (Sum of Product) 표현

3.3 Don't care

3.4 POS (Product of Sum)

3.5 5변수, 6변수를 위한 맵

3.6 다중 출력 문제

POS(Product of Sum)형태의 최소화

드모르간의 정리 이용 (P11)

$(a+b)' = a'b'$ $(ab)'=a'+b'$

POS → SOP → K-map 사용하여 최소화 → min SOP → min POS

step1 step2 step3

Ex) $((a+b)(c+d))'=(a'b')+(c'd')$ Ex) $((a'b')+(c'd'))' = (a+b)(c+d)$

Step 1. 함수의 **보수를 맵에 표현**

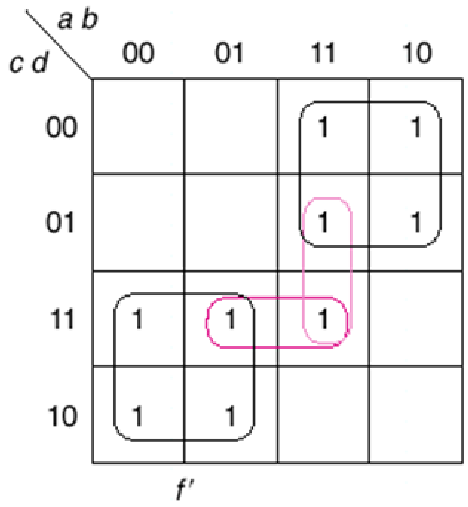
함수에 대한 맵이 이미 존재하는 경우, 모든 0을 1로, 모든 1을 0으로 대치. X는 변환하지 않음

Step 2. 함수의 보수에 대한 **최소 SOP 찾기**

Step 3. POS를 만들기 위하여, 2 에서 구한 SOP에 **다시 보수를 취함**

POS(Product of Sum)형태의 최소화

예제 3.25 $f(a, b, c, d) = \Sigma m(0, 1, 4, 5, 10, 11, 14)$
최소 f_{POS}를 구하라

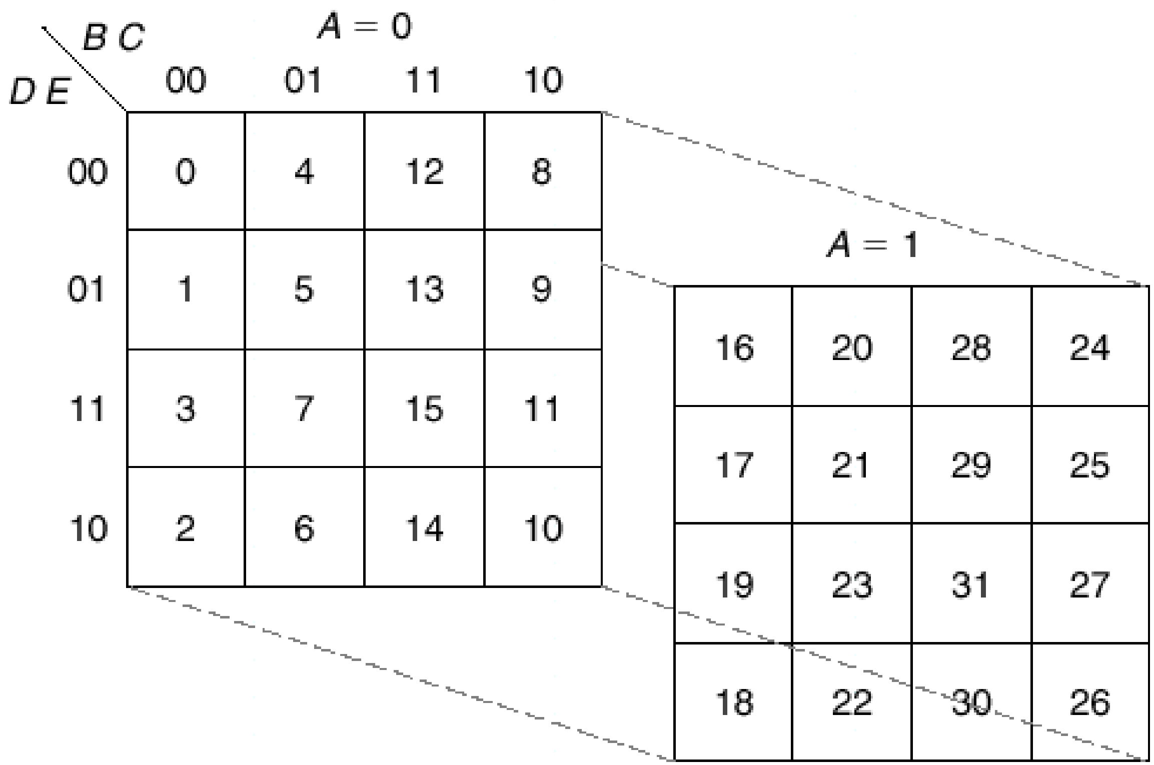


- $f'(a, b, c, d) = \Sigma m(2, 3, 6, 7, 8, 9, 12, 13, 15)$
- $f' = ac' + a'c + abd$ **(K map 결과)** : 최소 f'_{SOP}
  → $f = (a'+c)(a+c')(a'+b'+d')$: 최소 f_{POS}
- $f' = ac' + a'c + bcd$ **(K map 결과)** : 최소 f'_{SOP}
  → $f = (a'+c)(a+c')(b'+c'+d')$: 최소 f_{POS}

Chapter 3 Karnaugh Map (K-Map)

5개 변수 Map

- 5변수 맵은 $2^5 = 32$개로 구성
- 16개의 정사각형을 2개층(*layer*)구조로 나타냄



6개 변수 Map

- 6변수 맵은 $2^6 = 64$개로 구성
- 16개의 정사각형을 4개층(*layer*)구조로 나타냄

$AB = 00$

$EF \backslash CD$	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

$AB = 01$

16	20	28	24
17	21	29	25
19	23	31	27
18	22	30	26

$AB = 11$

48	52	60	56
49	53	61	57
51	55	63	59
50	54	62	58

$AB = 10$

32	36	44	40
33	37	45	41
35	39	47	43
34	38	46	42

5개 변수 함수 Mapping

$$F(A, B, C, D, E) = \Sigma m(4, 5, 6, 7, 9, 11, 13, 15, 16, 18, 27, 28, 31)$$

A = 0

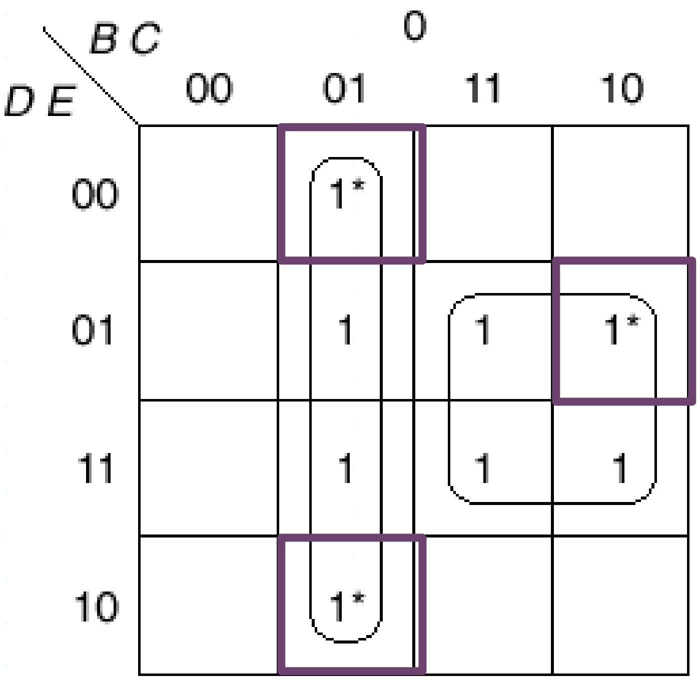
DE \ *BC*	00	01	11	10
00		1		
01		1	1	1
11		1	1	1
10		1		

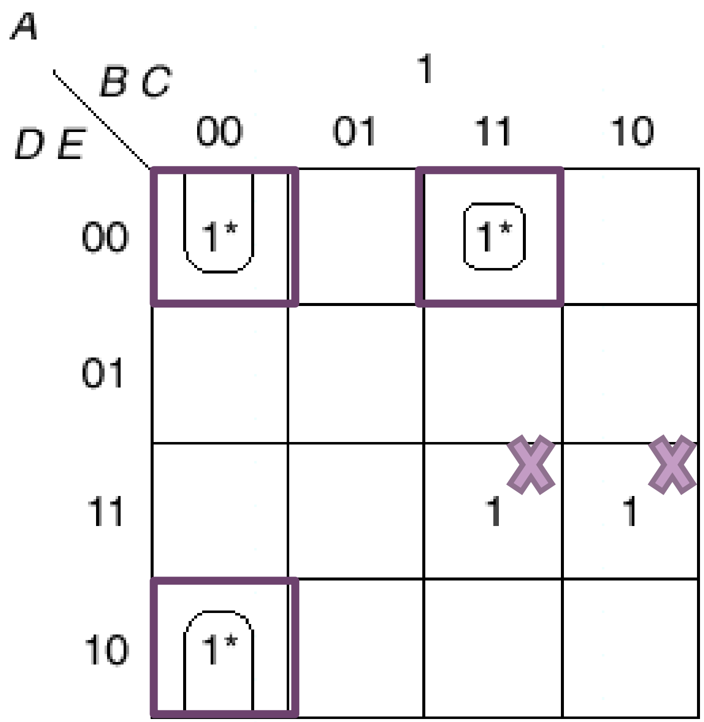
A = 1

DE \ *BC*	00	01	11	10
00	1		1	
01				
11			1	1
10	1			

5개 변수 K-map 풀기 (1)

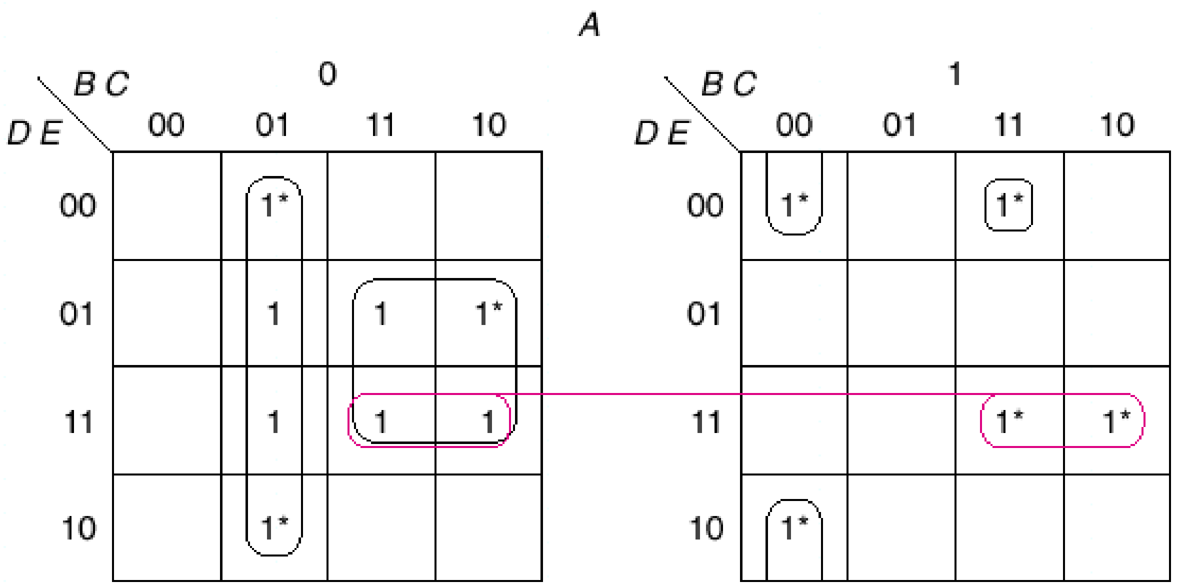
- 늘 그렇듯 EPI 먼저 찾기!
  - ➔ 한 layer에서 EPI 여야 하지만
  - ➔ 인접한 layer에서도 대응하는 정사각형에 1이 없는 것이어야만 함
- 그리고 이러한 1을 포함한 PI는 그 층에만 속하게 됨
  (따라서, 4변수 맵 문제가 된다.)
  ➔A′ 또는 A를 붙이기만 하면 4변수(BCDE)와 동일





5개 변수 K-map 풀기 (2)

- 양쪽의 레이어 상에 1을 커버하는 EPI



- 최종 해

$F = A'B'C + A'BE + AB'C'E' + ABCD'E' + BDE$

Chapter 3 Karnaugh Map (K-Map)

3.1 K-map 소개

3.2 K-map 을 사용한 최소 SOP (Sum of Product) 표현

3.3 Don't care

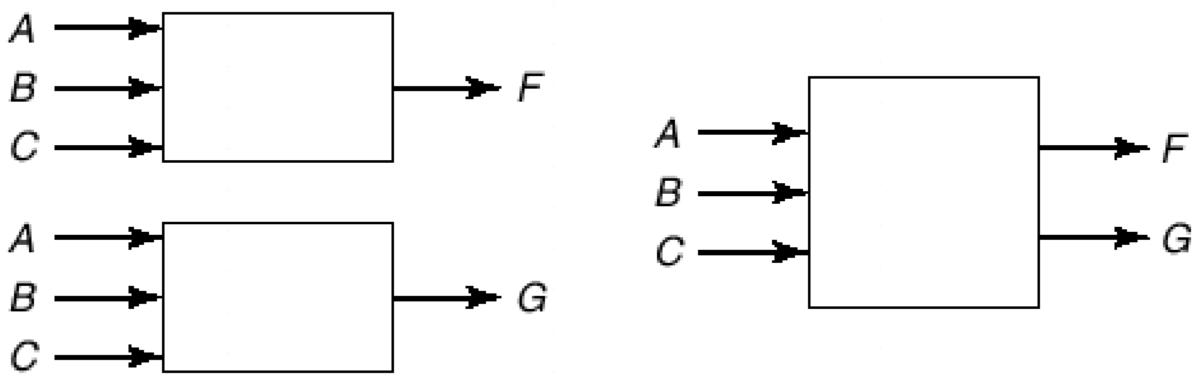
3.4 POS (Product of Sum)

3.5 5변수, 6변수를 위한 맵

3.6 다중 출력 문제

다중 출력 문제

- 다중 출력을 가진 시스템을 설계하는 경우가 많음
- 3개의 입력 *A, B, C*와 2개의 출력 *F, G*
  - 2개의 문제로 나누어서 취급
  - 하나의 시스템: 게이트들의 공유가능 -> 비용 줄임.

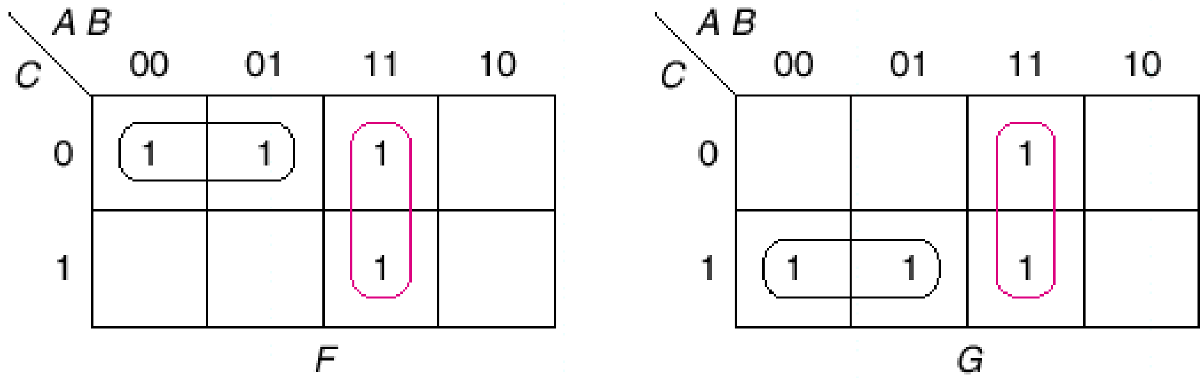


Two Separate Systems

One System

다중 출력 문제 – 결과의 일부를 완전 공유

예제 3.32 $F(A, B, C) = \Sigma m(0, 2, 6, 7)$ $G(A, B, C) = \Sigma m(1, 3, 6, 7)$



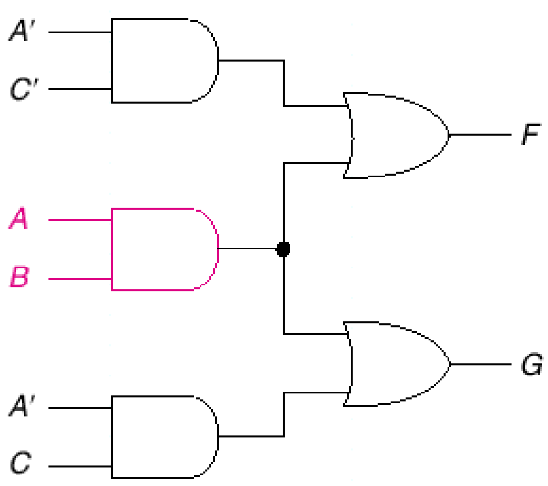
- 각 함수에 대해 아래와 같은 식을 얻을 수 있음

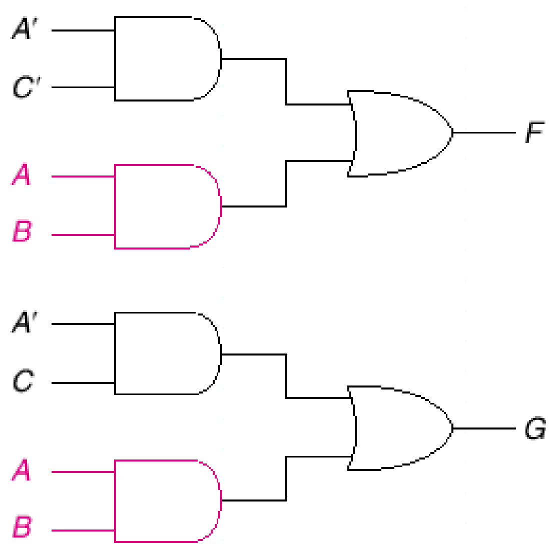
$F = A'C' + AB$ $G = A'C + AB$

다중 출력 문제 – 공유 가능한 항을 찾기

예제 3.32(계속) $F = A'C' + AB$ $G = A'C + AB$

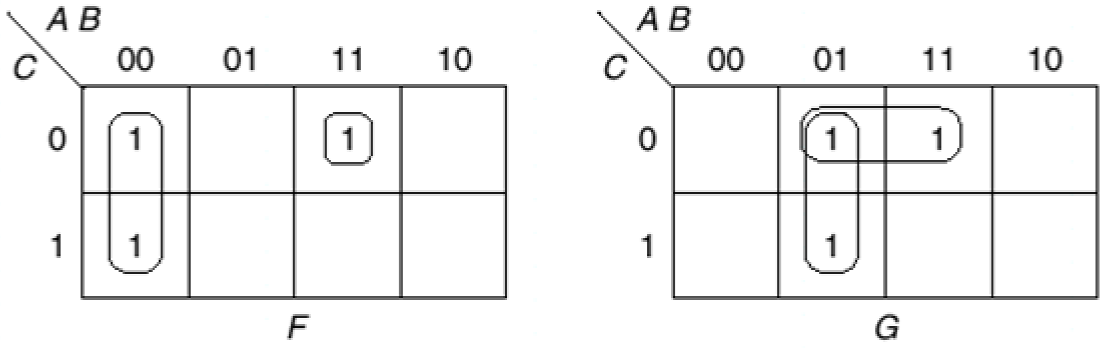
- 우측: 두개의 독립된 문제
- 좌측: 항(AB) 공유함으로써 게이트 수가 줄어듬.





다중 출력 문제 – 공유 가능한 항을 찾기

예제 3.33 $F(A, B, C) = \Sigma m(0, 1, 6)$ $G(A, B, C) = \Sigma m(2, 3, 6)$



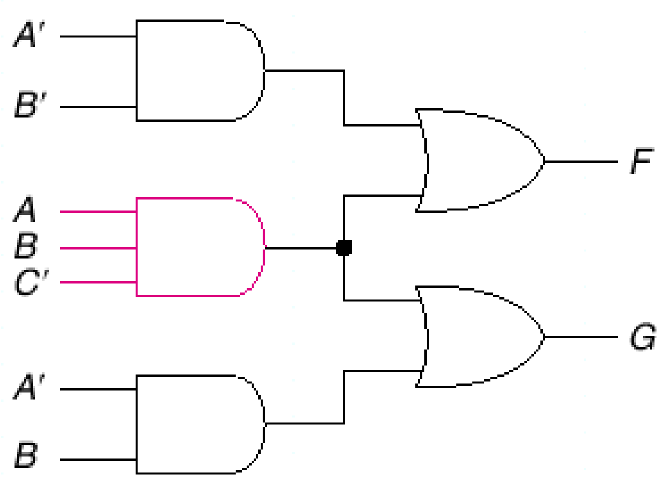
독립된 문제(위쪽 맵) : $F = A'B' + ABC'$ $G = A'B + BC'$

다중 출력 문제 – 공유 가능한 항을 찾기

예제 3.33 (계속)

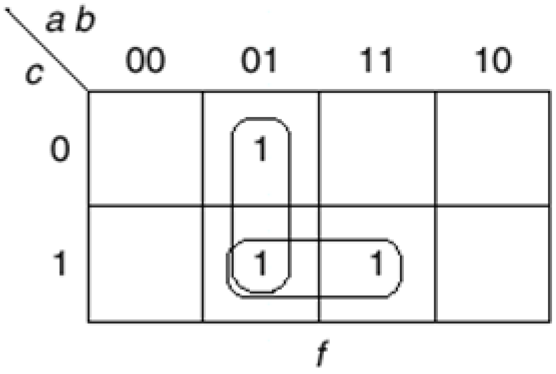
- 독립된 문제:
  $F = A'B' + ABC'$ $G = A'B + BC'$ (6 게이트: 4 AND, 2 OR, 13 inputs)

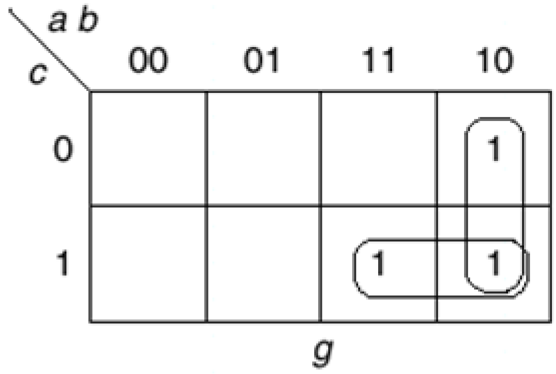
- 공유:
  $F = A'B' + ABC'$ $G = A'B + ABC'$ (5 게이트: 3 AND, 2 OR, 11 inputs)



다중 출력 문제 – 공유 가능한 항을 찾기

예제 3.34 $F(A, B, C) = \Sigma m(2, 3, 7)$ $G(A, B, C) = \Sigma m(4, 5, 7)$





- 독립된 문제 해

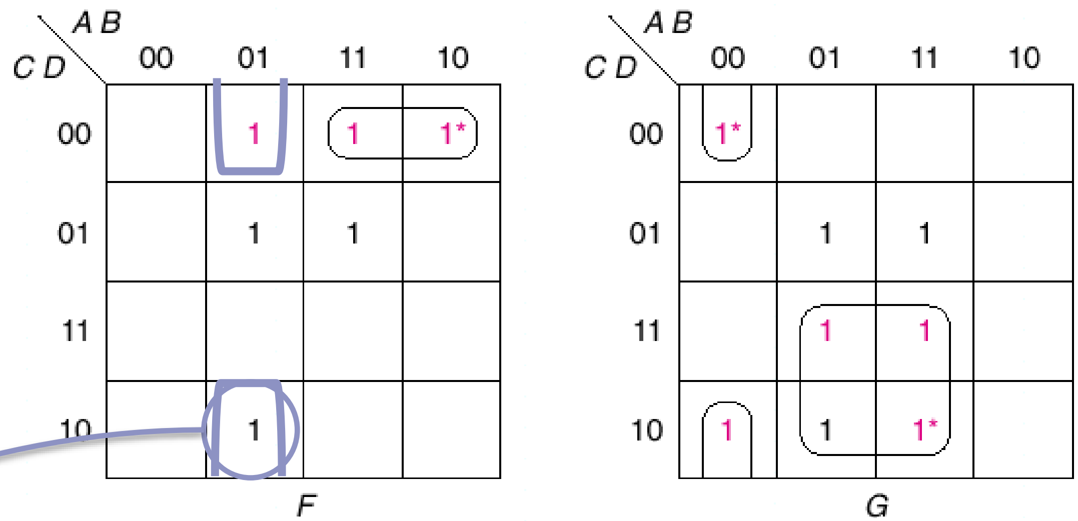
  $f = a'b + bc$

  $g = ab' + ac$

다중 출력 문제 – 공유 가능한 항들이 복잡할 때

예제 3.35 $F(A, B, C, D) = \Sigma m(4, 5, 6, 8, 12, 13)$

$G(A, B, C, D) = \Sigma m(0, 2, 5, 6, 7, 13, 14, 15)$



1. 하나의 함수에서만 1인 것(**적색, 명백히 공유 불가능**)들에서 EPI 찾기

$F = AC'D' + ..$

$G = A'B'D' + BC + ..$

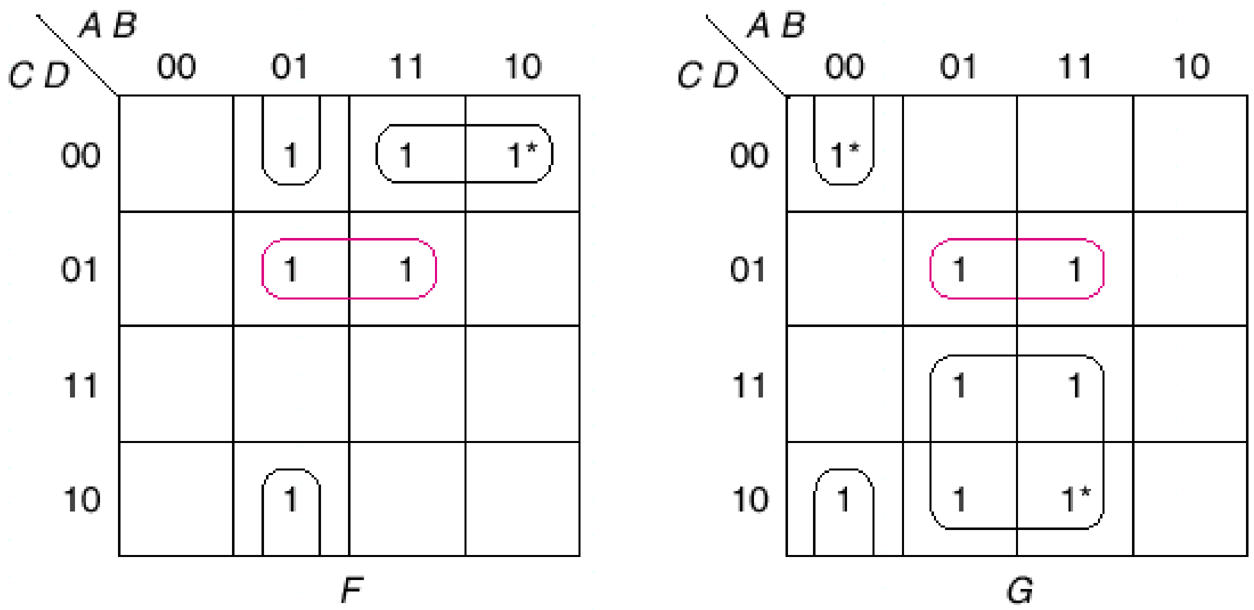
2. 공유 가능하지만 다른 쪽에서 EPI에 포함되었으면 공유 포기하고 나만의 그루핑

$F = AC'D' + A'BD' + ..$

$G = A'B'D' + BC + ..$

다중 출력 문제 – 공유 가능한 항들이 복잡할 때

예제 3.35 (계속)



3. 나머지는 공유!

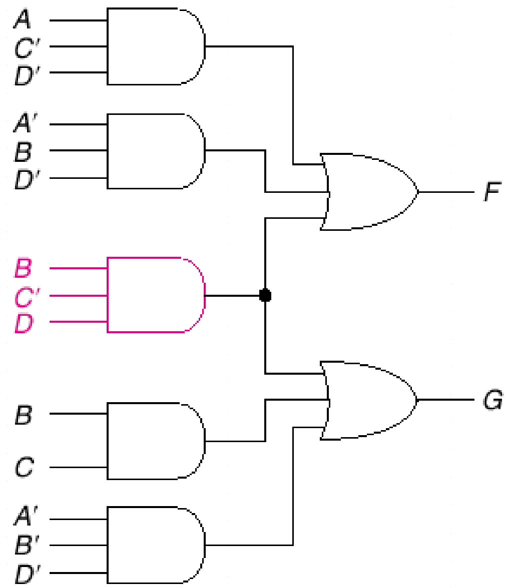
$F = AC'D' + A'BD' + BC'D$

$G = A'B'D' + BC + BC'D$

(20 inputs, 7 게이트)

다중 출력 문제 – 공유 가능한 항들이 복잡할 때

예제 3.35 (계속)



(20 inputs, 7 게이트)

- 독립함수로 풀면

  $F = AC'D' + A'BD' + BC'$

  $G = A'B'D' + BC + BD$

  (21 inputs, 8 게이트)

다중 출력 문제 – 공유 가능한 항들이 복잡할 때

방법 정리하면

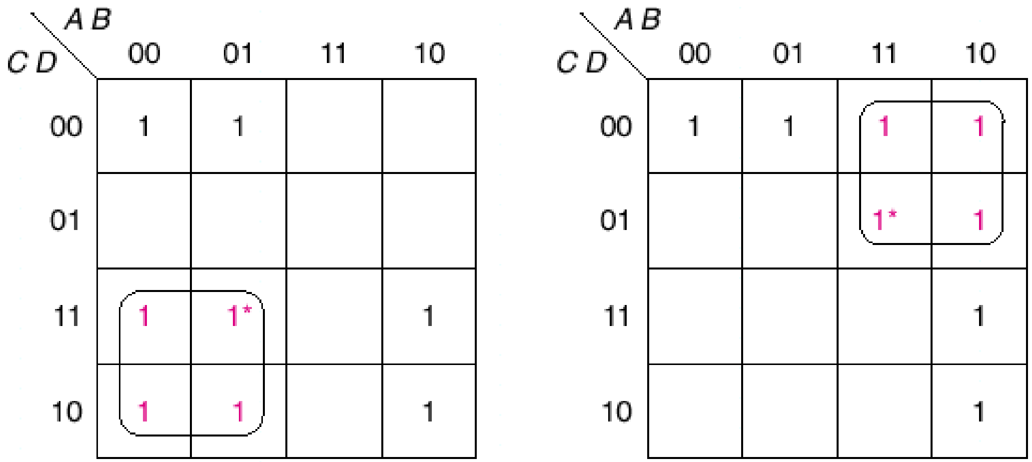
1. 하나의 함수에서만 1인 것(**명백히 공유 불가능**)들에서 EPI 찾기
2. 공유 가능하지만 다른 쪽에서 EPI에 포함되었으면 공유 포기
3. 나머지 공유 가능한 항들 공유

다중 출력 문제 – 공유 가능한 항들이 복잡할 때

예제 3.36 $F(A, B, C, D) = \Sigma m(0, 2, 3, 4, 6, 7, 10, 11)$

$G(A, B, C, D) = \Sigma m(0, 4, 8, 9, 10, 11, 12, 13)$

1. 하나의 함수에서만 1인 것(**명백히 공유 불가능**)들에서 EPI 찾기



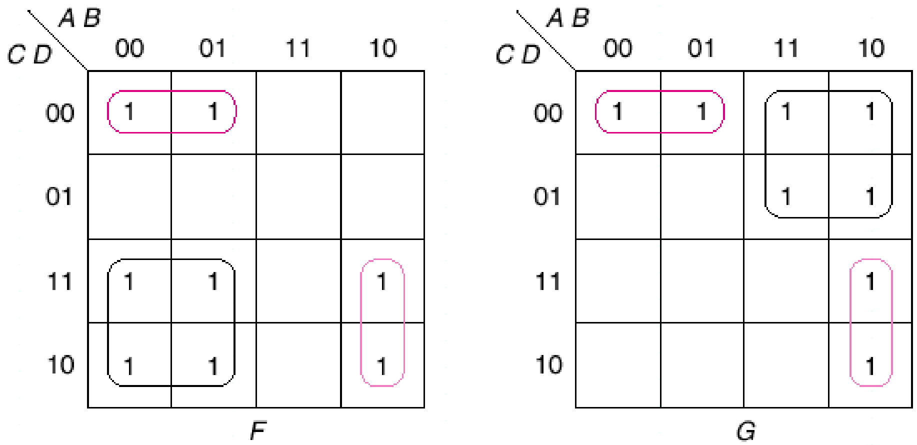
2. 공유 가능하지만 다른 쪽에서 EPI에 포함되었으면 공유 포기

→ 이 문제에서는 그런 것은 없네.

다중 출력 문제 – 공유 가능한 항들이 복잡할 때

예제 3.36 계속

3. 나머지 공유 가능한 항들 공유



$A'C'D'$ 와 $AB'C$ 를 공유
(16 inputs, 6 게이트)

- $F = A'C + A'C'D' + AB'C$
- $G = AC' + A'C'D' + AB'C$