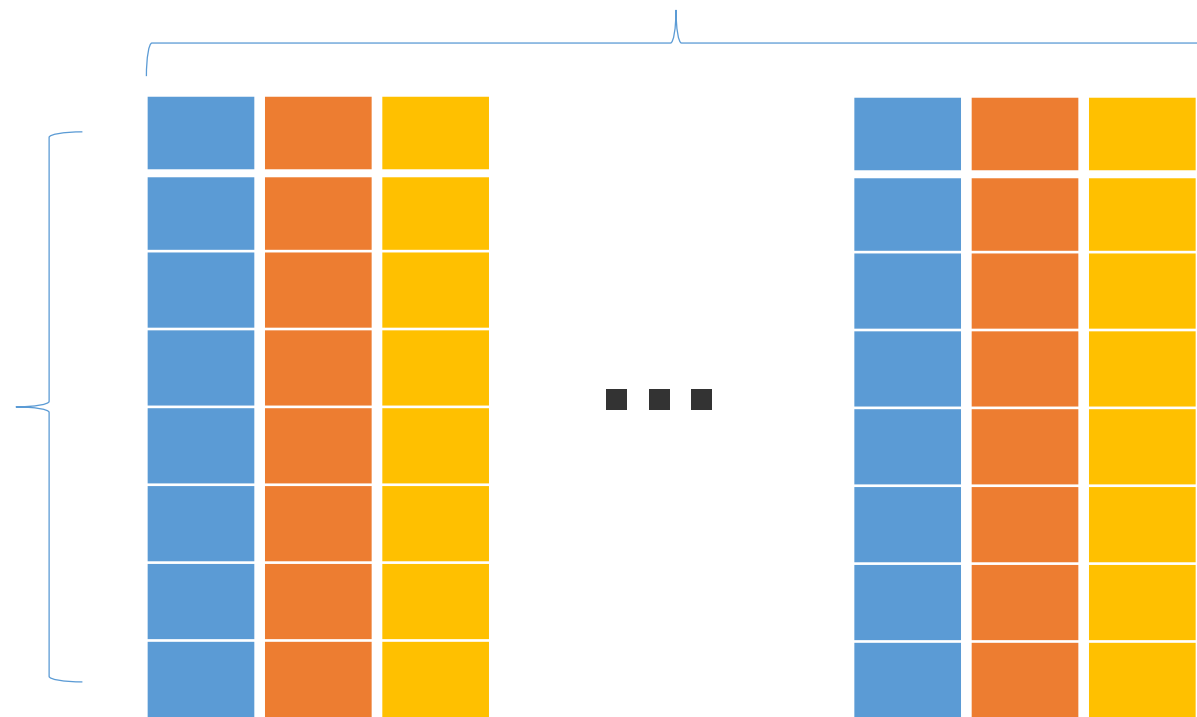
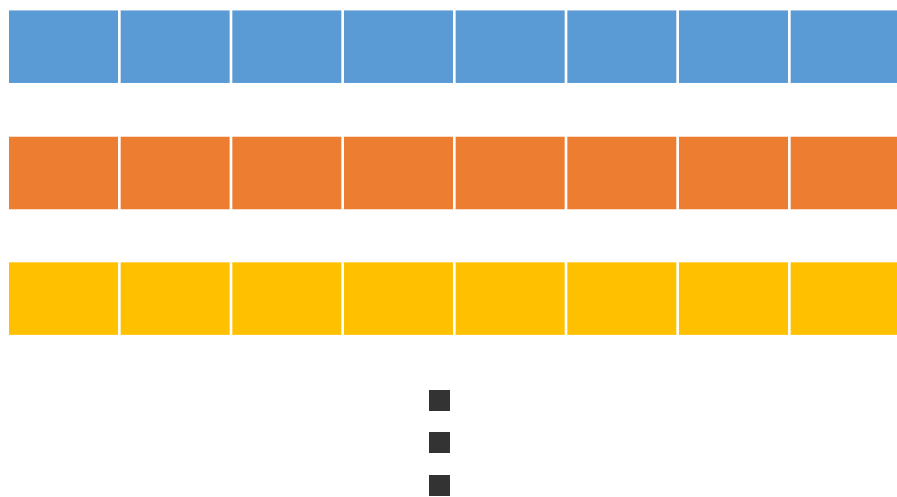


KARNAUGH MAPS를 사용한 S-box 비트슬라이스 변환 방법

<https://youtu.be/N9h4eaMKogM>

비트 슬라이싱

- Eli Biham⁰ | A Fast New DES Implementation in Software 라는 논문에서 처음 사용
- 하드웨어에서 논리 회로를 구현하는 것처럼 단일 비트 논리 연산(*AND* , *XOR* , *OR* , *NOT* 등) 으로 기능을 표현
- 그런 다음 CPU에서 비트 연산을 사용하여 병렬로 함수의 여러 인스턴스를 수행
- 속도, 병렬화, 일정한 실행 시간 장점



KARNAUGH MAPS

- 3-to-2-bit S-box

```
uint8_t SBOX [] = { 1 , 0 , 3 , 1 , 2 , 2 , 3 , 0 };
```

$f(0,0,0) = 0b01$, $f(0,0,1) = 0b00$, $f(0,1,0) = 0b11$, ...

SBOX(a,b,c)

abc	out
000	01
001	00
010	11
011	01
100	10
101	10
110	11
111	00

$f_L(a,b,c)$

abc	out
000	0
001	0
010	1
011	0
100	1
101	1
110	1
111	0

$f_R(a,b,c)$

abc	out
000	1
001	0
010	1
011	1
100	0
101	0
110	1
111	0

KARNAUGH MAPS

- K-맵을 사용한 S-박스 비트슬라이스

uint8_t SBOX [] = { 1, 0, 3, 1, 2, 2, 3, 0 };

$f(0,0,0) = 0b01$, $f(0,0,1) = 0b00$, $f(0,1,0) = 0b11$, ...

SBOX(a,b,c)

abc	out
000	01
001	00
010	11
011	01
100	10
101	10
110	11
111	00

$f_L(a,b,c)$

abc	out
000	0
001	0
010	1
011	0
100	1
101	1
110	1
111	0

$f_R(a,b,c)$

abc	out
000	1
001	0
010	1
011	1
100	0
101	0
110	1
111	0

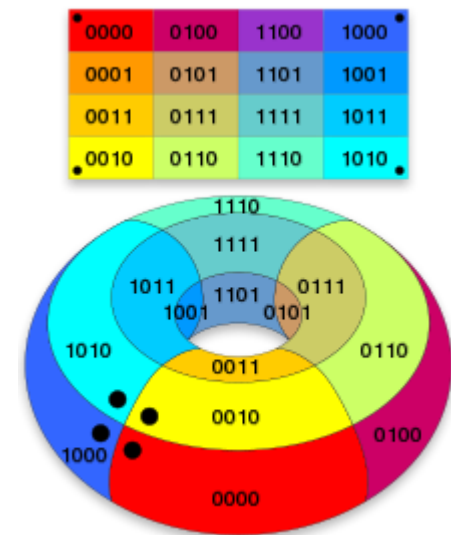
		(b c)			
		00	01	11	10
(a)	0	0	0	0	1
	1	1	1	0	1
		$f_L(a,b,c)$			

		(b c)			
		00	01	11	10
(a)	0	1	0	1	1
	1	0	0	0	1
		$f_R(a,b,c)$			

그룹화

인접 셀 그룹이 1인 값을 찾기 - K-map을 통해 부울 표현식을 단순화

- 2^n 개의 1의 값을 가진 사각형 그룹 (0이 포함될 수 없음)
- 가능한 크고 적은 수의 그룹이 있어야함
- 그룹은 겹칠 수 있음



(a)

		(b c)			
		00	01	11	10
0		0	0	0	1
1		1	1	0	1

$f_L(a,b,c)$

(a)

		(b c)			
		00	01	11	10
0		1	0	1	1
1		0	0	0	1

$f_R(a,b,c)$

(a)

		(b c)			
		00	01	11	10
0		0	0	0	1
1		1	1	0	1

$f_L(a,b,c)$

(a)

		(b c)			
		00	01	11	10
0		1	0	1	1
1		0	0	0	1

$f_R(a,b,c)$

BITSLICED SBOX()

(a)

	(b c)			
	00	01	11	10
0	0	0	0	1
1	1	1	0	1

$f_L(a,b,c)$

(a)

	(b c)			
	00	01	11	10
0	1	0	1	1
1	0	0	0	1

$f_R(a,b,c)$

(a)

	(b c)			
	00	01	11	10
0	0	0	0	1
1	1	1	0	1

$f_L(a,b,c)$

(a)

	(b c)			
	00	01	11	10
0	1	0	1	1
1	0	0	0	1

$f_R(a,b,c)$

• $f_{L(a,b,c)}$

빨간색 그룹 100, 101.

a=1 및 b=0는 그룹에 포함. c는 값의 변화와 무관 ($a \& \sim b$)

녹색 그룹 010, 110.

a 무시, b=1, c=0. ($b \& \sim c$)

```
uint8_t SBOXL(uint8_t a, uint8_t b, uint8_t c) {
    return (a & ~b) | (b & ~c);
}
```

BITSLICED SBOX()

(a)

	(b c)			
	00	01	11	10
0	0	0	0	1
1	1	1	0	1

$f_L(a,b,c)$

(a)

	(b c)			
	00	01	11	10
0	1	0	1	1
1	0	0	0	1

$f_R(a,b,c)$

(a)

	(b c)			
	00	01	11	10
0	0	0	0	1
1	1	1	0	1

$f_L(a,b,c)$

(a)

	(b c)			
	00	01	11	10
0	1	0	1	1
1	0	0	0	1

$f_R(a,b,c)$

• $f_{R(a,b,c)}$

011, 010 ($\sim a \& b$)

010, 110 ($b \& \sim c$)

000, 010 ($\sim a \& \sim c$)

```
uint8_t SBOXR(uint8_t a, uint8_t b, uint8_t c) {
    return (~a & b) | (b & ~c) | (~a & ~c);
}
```

BITSLICED SBOX()

```
uint8_t SBOXL(uint8_t a, uint8_t b, uint8_t c) {  
    return (a & ~b) | (b & ~c);  
}
```

```
uint8_t SBOXR(uint8_t a, uint8_t b, uint8_t c) {  
    return (~a & b) | (b & ~c) | (~a & ~c);  
}
```

```
void SBOX(uint8_t a, uint8_t b, uint8_t c, uint8_t* l, uint8_t* r) {  
    uint8_t na = ~a;  
    uint8_t nb = ~b;  
    uint8_t nc = ~c;  
  
    uint8_t t0 = b & nc;  
  
    *l = (a & nb) | t0;  
    *r = (na & b) | (na & nc) | t0;  
}
```

```
void SBOX(uint8_t a, uint8_t b, uint8_t c, uint8_t* l, uint8_t* r) {  
    uint8_t na = ~a;  
    uint8_t nb = ~b;  
    uint8_t nc = ~c;  
  
    uint8_t t0 = b & nc;  
    uint8_t t1 = b | nc;  
  
    *l = (a & nb) | t0;  
    *r = (na & t1) | t0;  
}
```


n-variable K-map

form 1

	BC			
A \	00	01	11	10
0	01		32	
1	45		76	

form 2

AB \ CD	00	01	11	10
00	0 2		6 4	
01	8 10		14 12	
11	24 26		30 28	
10	16 18		22 20	

 $E=0$

AB \ CD	00	01	11	10
00	1 3		7 5	
01	9 11		15 13	
11	25 27		31 29	
10	17 19		23 21	

E=1

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

form 1

	01		32
	45		76
1213		1514	
89		11	10

form 2

		EF			
		00	01	11	10
A=0	CD 00	01		32	
	01	45		76	
	11				
	10	12 13		15 14	
		89		11 10	

$$A=0$$

CD \ EF	00	01	11	10
00	16 17		19 18	
01	20 21		23 22	
11				
10	28 29		31 30	
	24 25		27 26	

		EF			
		00	01	11	10
A=1	CD 00	32 33		35 34	
	01	36 37		39 38	
	11				
	10	44 45		47 46	
		40 41		43 42	

A=1

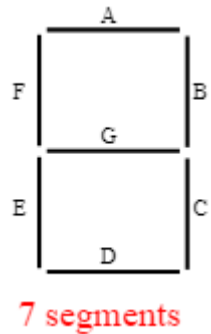
$$B=0$$

CD \ EF	00	01	11	10
00	48 49		51 50	
01	52 53		55 54	
11				
10	60 61		63 62	
	56 57		59 58	

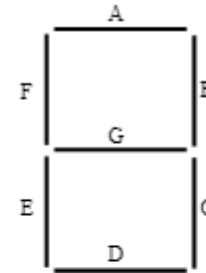
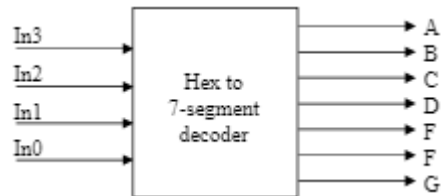
B=1

Example

- 7-segment display decoder



0 1 2 3 4 5 6 7
8 9 A b C d e F



0 1 2 3 4 5 6 7
8 9 A b C d e F

--- = don't care

In3	In2	In1	In0	A	B	C	D	E	F	G
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	X	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	X	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

Example

In3	In2	In1	In0	A	B	C	D	E	F	G
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	1	1	1	0	1	1	1
1	0	1	1	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0
1	1	0	1	0	1	1	1	1	0	1
1	1	1	0	1	0	0	1	1	1	1
1	1	1	1	1	0	0	0	1	1	1

In3 \ In2 \ In1 In0	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	1	0	1	1
10	1	1	0	1

K-map for
A output

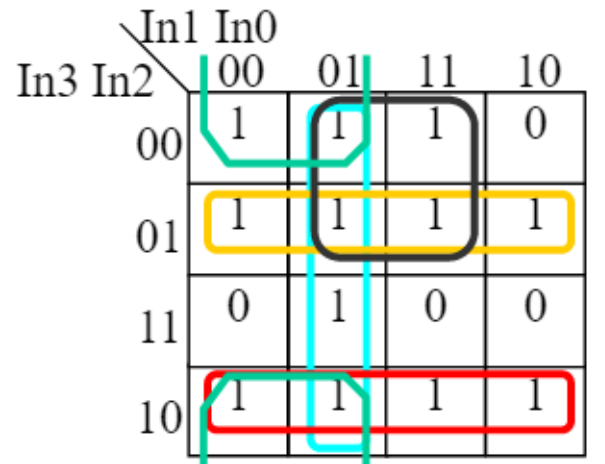
$$A = \text{In2}'\text{In0}' + \text{In3}'\text{In1} + \text{In2 In1} \\ + \text{In3 In0}' + \text{In3}'\text{In2 In0} + \text{In3 In2}'\text{In1}'$$

In3 \ In2 \ In1 In0	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	0	1	0	0
10	1	1	0	1

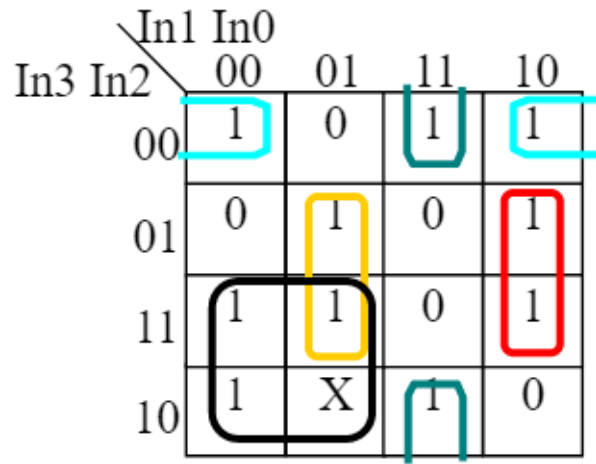
K-map for
B output

$$B = \text{In2}'\text{In0}' + \text{In2}'\text{In1}' + \text{In3}'\text{In1}'\text{In0}' \\ + \text{In3 In1}'\text{In0} + \text{In3}'\text{In1 In0}$$

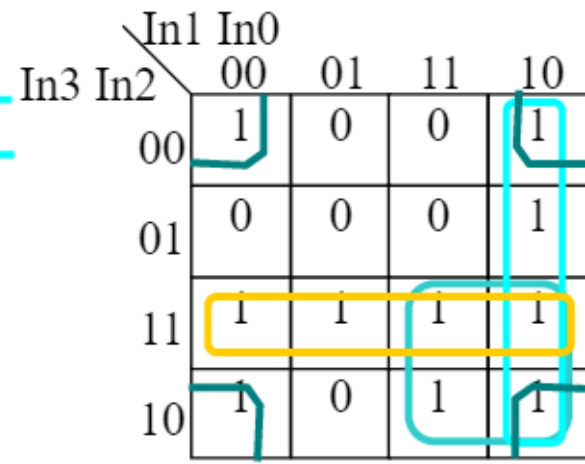
Example



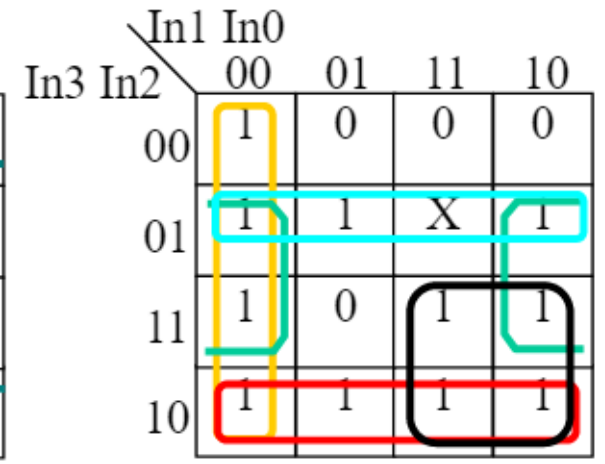
K-map for C output



K-map for D output



K-map for E output



K-map for F output

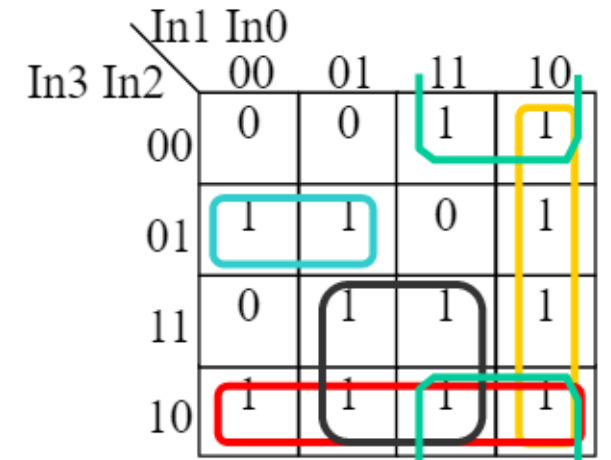
$$C = \text{In3 In2}' + \text{In1}'\text{In0} + \text{In2}'\text{In1}' + \text{In3}'\text{In0} + \text{In3}'\text{In2}$$

$$D = \text{In3}'\text{In2}'\text{In0}' + \text{In2}'\text{In1 In0} + \text{In2 In1}'\text{In0} \\ + \text{In3 In1}' + \text{In2 In1 In0}'$$

$$E = \text{In2}'\text{In0}' + \text{In3 In2} + \text{In1 In0}' + \text{In3 In1}$$

$$F = \text{In1}'\text{In0}' + \text{In3 In2}' + \text{In2 In0}' + \text{In3 In1} + \text{In3}'\text{In2}$$

$$G = \text{In3 In2}' + \text{In1 In0}' + \text{In3 In0} + \text{In3}'\text{In2 In1}' + \text{In2}'\text{In1}$$



K-map for G output

Q & A