Unit 7: Karnaugh Maps

CSE 220: System Fundamental I

Stony Brook University

Joydeep Mitra

Motivation

- Boolean Algebra simplification requires skill and experience
  - You may not always end up with the simplest form
- Fortunately, we have a mechanical and graphical way to simplify Boolean equations called **Karnaugh Maps** (or **K-Maps**)
- Recall that simplification involves combining minterms of the form PA + PA’ to obtain P
- K-Maps make it easy to spot such patterns by placing minterms in a grid

K-Map Rules

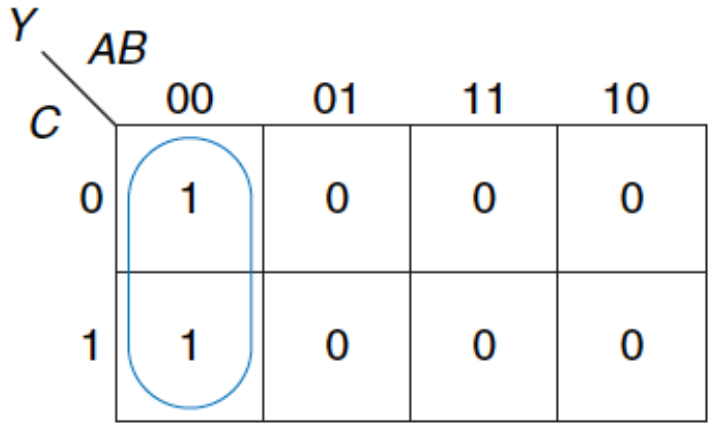
- Works best with 3-4 variables
- The top row of the K-map holds the four possible values of for inputs A and B
- The left column holds the two possible values for input C
- Each square corresponds to a row in the truth table and its value is the output for the row
- Each square (or minterm) differs from its adjacent by a single variable
- This arrangement is called *gray code*; it differs from *regular binary ordering*
  - Regular ordering => 00, 01, 10, 11
  - Gray code => 00, 01, 11, 10
- The squares in a K-map also *wrap* around, i.e., squares in the far right differ from squares in the far left by a single element.
  - Same for top and bottom (applicable when K-map has more than 2 rows)

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Y \ AB	00	01	11	10
C = 0	1	0	0	0
C = 1	1	0	0	0

Simplification Using K-Maps

- We encircle adjacent squares with 1's in the K-map
- For each circle, note the corresponding implicant (product of one or more literals)
- Exclude variables whose true and complement forms are both in the circle



Y = A'B'

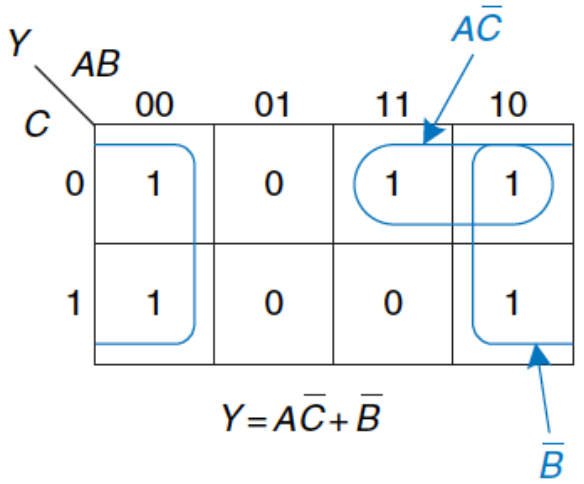
Simplification Rules Using K-Maps

1. Use the fewest circles necessary to cover all the 1's
2. Each circle should be as large as possible
3. All the squares in each circle must contain 1's
4. Each circle must span a rectangular block that is a power of 2 squares in each direction
5. A circle may wrap around the edges of the K-map
6. A 1 may be circled multiple times if that helps using fewer circles

Example 1

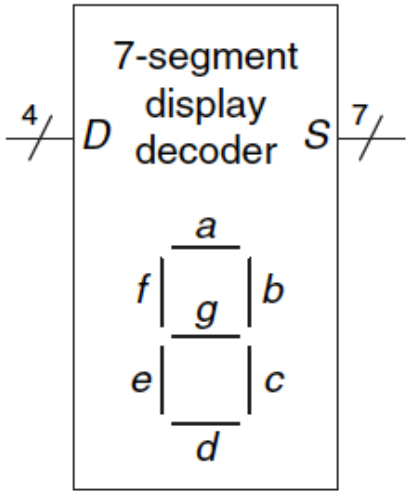
- Simplify a function Y = F(A,B,C) with the following K-map:

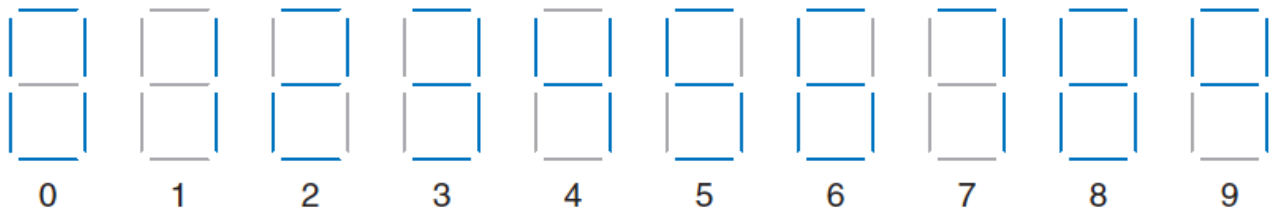
Y, AB / C	00	01	11	10
0	1	0	1	1
1	1	0	0	1



Example 2

- A seven-segment display decoder takes 4-bit data input ($D_{3:0}$) and produces 7 outputs (S_a-S_g) to control the display of numbers 0-9 in a light emitting diode.
- We will write a truth table and use K-maps to find the Boolean equation between inputs and outputs





Inputs marked in blue are set to 1 for each digit

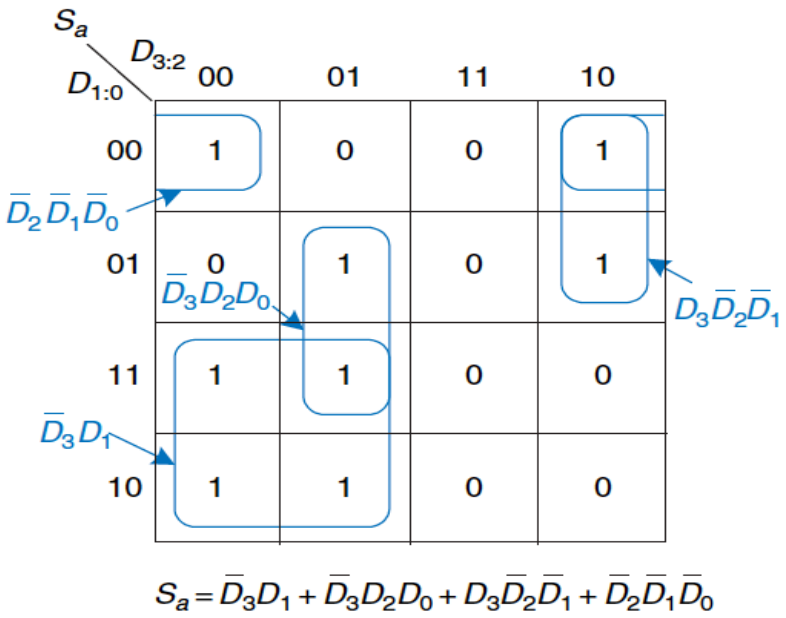
Example 2 – Truth Table

$D_{3:0}$	S_a	S_b	S_c	S_d	S_e	S_f	S_g
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	0	0	1	1
others	0	0	0	0	0	0	0

Example 2

- For each output a K-map needs to be drawn
  - For demonstration, we will draw the K-maps for S_a and then S_b

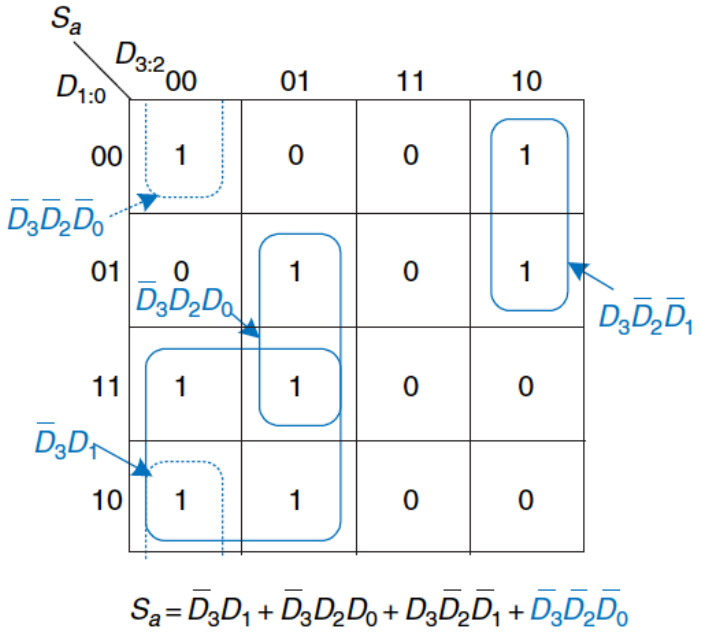
S_a	$D_{3:2}$ 00	01	11	10
$D_{1:0}$ 00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0



Example 2

- Alternative K-map for S_a

S_a: $D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0

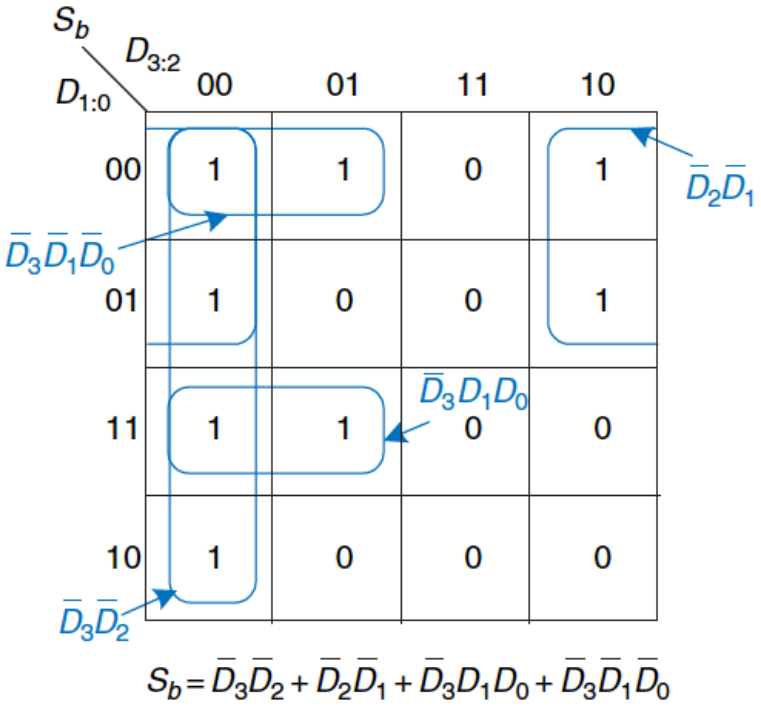


Notice how we can pair the top left and bottom left cells instead of the top left and top right cells

Example 2

- K-map for S_b

S_b : $D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	1	0	1
01	1	0	0	1
11	1	1	0	0
10	1	0	0	0



Don’t Cares

- Recall that don’t care entries can be used to reduce the no. of rows in a truth table
  - X indicates don’t care values meaning it can be 0 or 1
  - They appear when the output values are inconsequential, or the input combinations can never happen

$D_{3:0}$	S_a	S_b	S_c	S_d	S_e	S_f	S_g
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	0	0	1	1
others	X	X	X	X	X	X	X

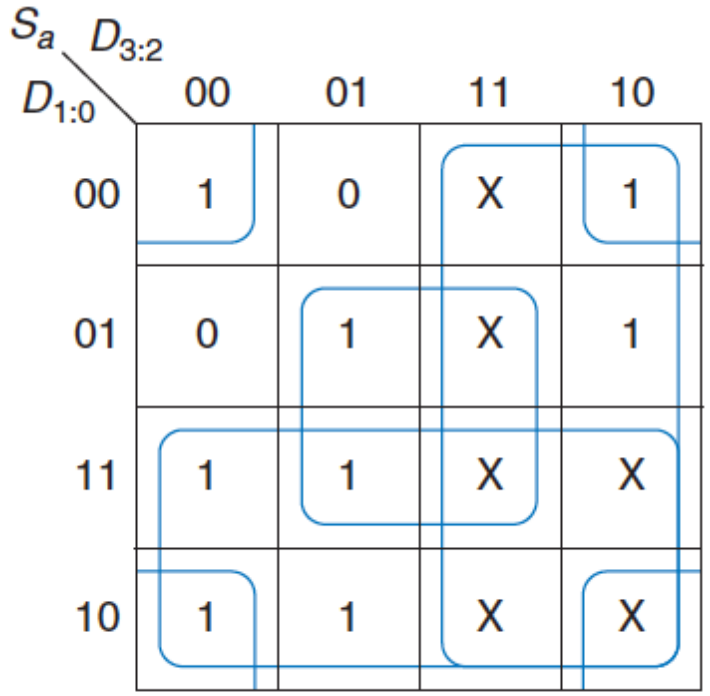
Don’t Cares in K-Maps

- In a K-map, X’s allow for further simplification
- We can treat them as 1’s or 0’s as per our convenience
- We can circle a don’t care if it helps us reduce the expression
- A circled don’t care counts as 1; a non-circled don’t care counts as 0

Example 2 with Don't Cares

- K-map for S_a

S_a	$D_{3:2}$ 00	01	11	10
$D_{1:0}$ 00	1	0	X	1
01	0	1	X	1
11	1	1	X	X
10	1	1	X	X

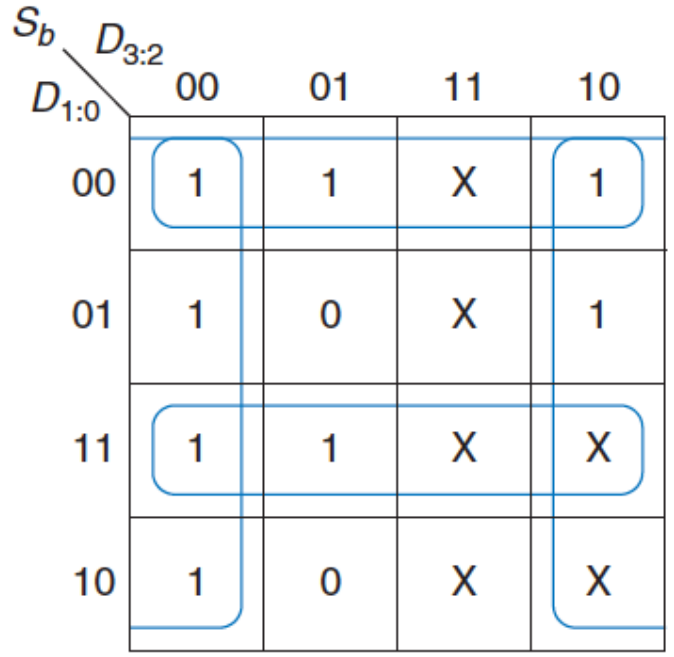


$$S_a = D_3 + D_2D_0 + \bar{D}_2\bar{D}_0 + D_1$$

Example 2 with Don’t Cares

- K-map for S_b

S_b $D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	1	X	1
01	1	0	X	1
11	1	1	X	X
10	1	0	X	X

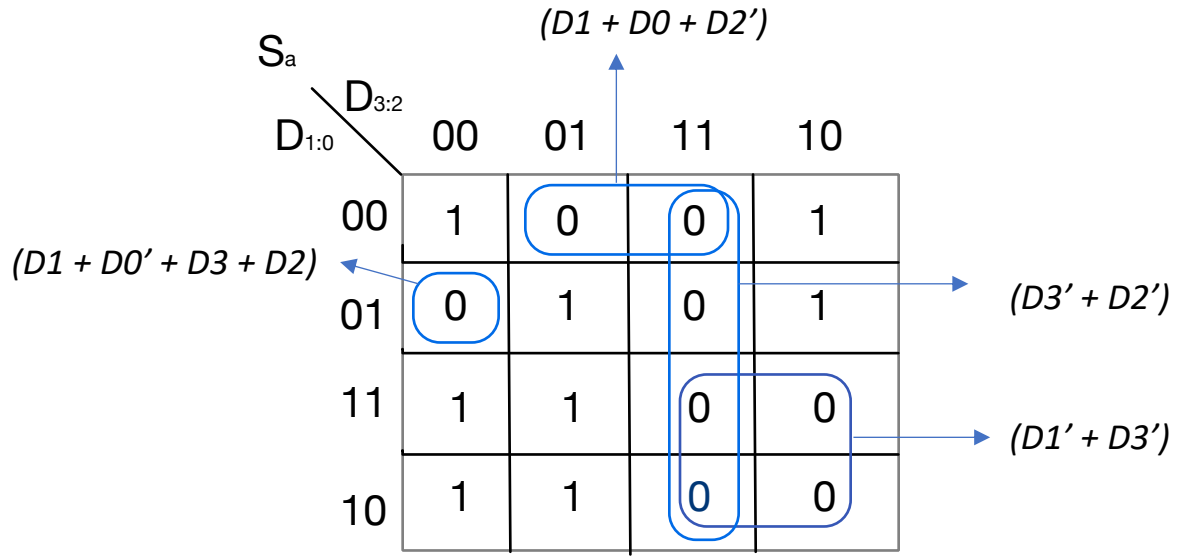


$$S_b = \bar{D}_2 + D_1 D_0 + \bar{D}_1 \bar{D}_0$$

K-Map with POS Form

- For POS form, we group the 0s and not 1s
- Each cell indicates a maxterm (and not minterm)

S_a : $D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0

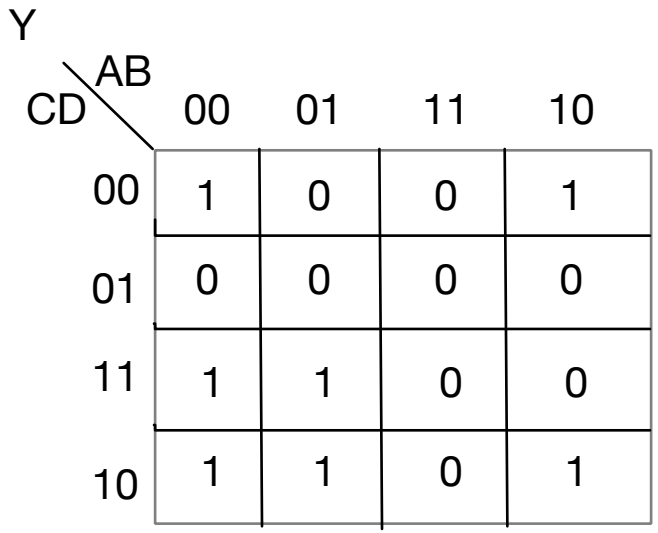


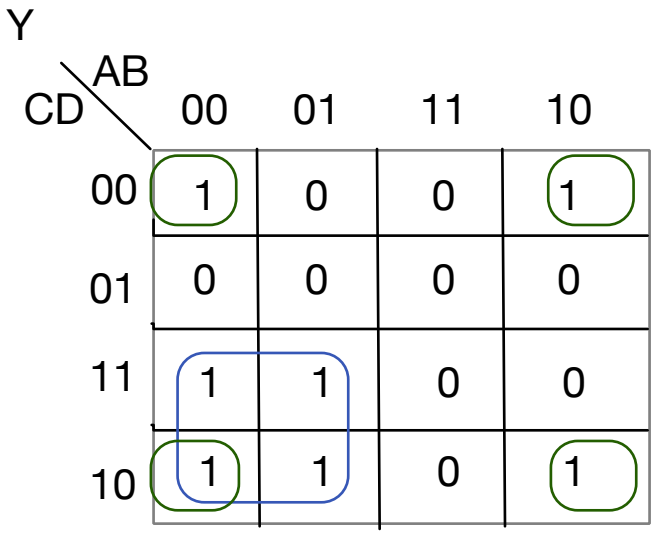
$S_a = (D1 + D0 + D2')(D3' + D2')(D1' + D3')(D1 + D0' + D3 + D2)$

Example 3

- Write a simplified Boolean equation in SOP form for the the truth table

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



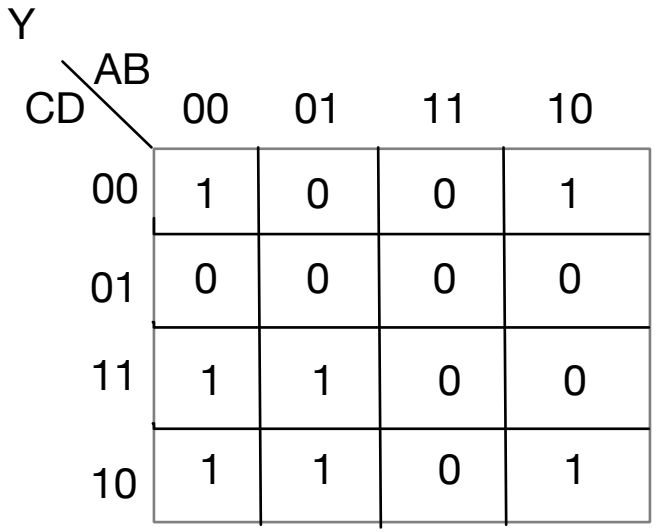


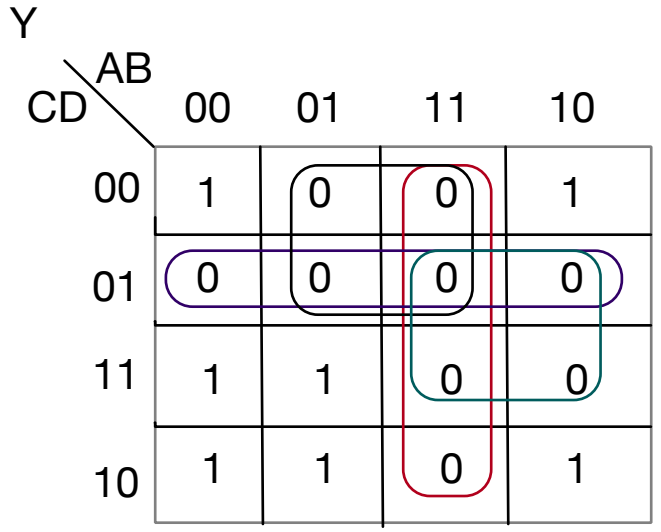
$Y = A'C + B'D'$

Example 4

- Write the equivalent POS form for the same truth table

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



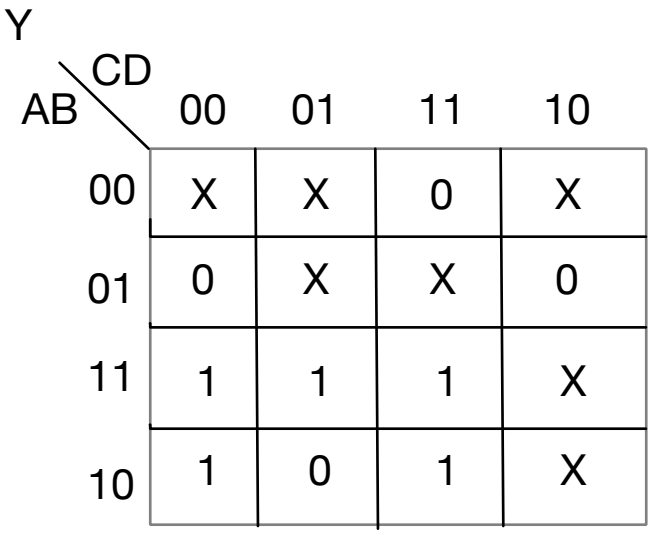


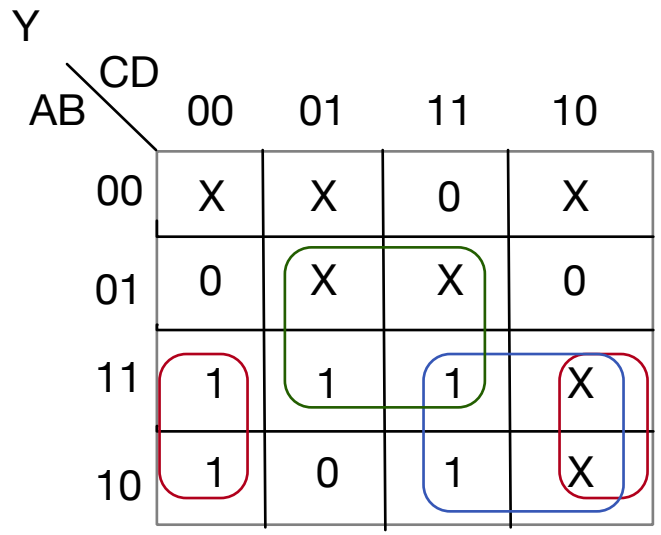
$Y = (A'+B')(C+D')(C+B')(D'+A')$

Example 5

- Find a minimal Boolean equation for the truth table

A	B	C	D	Y
0	0	0	0	X
0	0	0	1	X
0	0	1	0	X
0	0	1	1	0
0	1	0	0	0
0	1	0	1	X
0	1	1	0	0
0	1	1	1	X
1	0	0	0	1
1	0	0	1	0
1	0	1	0	X
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	X
1	1	1	1	1

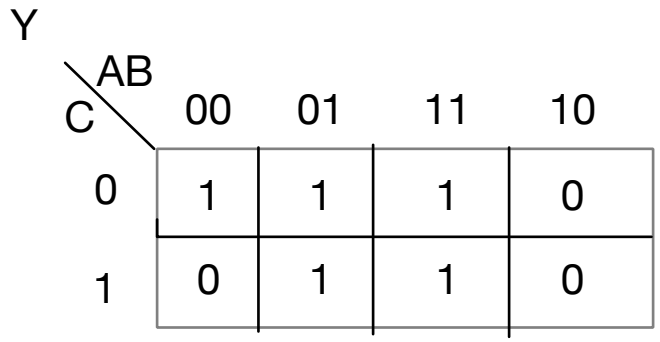


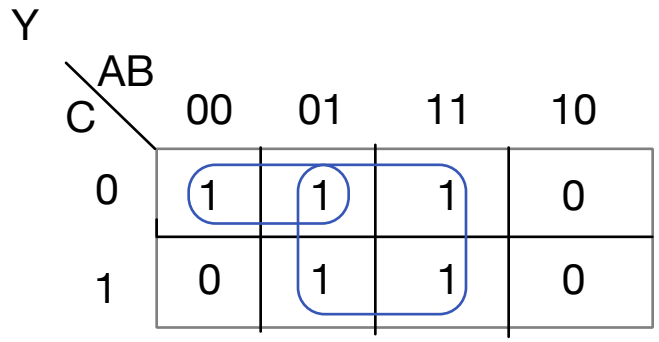


$Y = AD' + AC + BD$

Example 6

- Simplify the Boolean equation $Y = BC + A'B'C' + BC'$ using K-maps
- Convert to SOP form and fill in the k-map

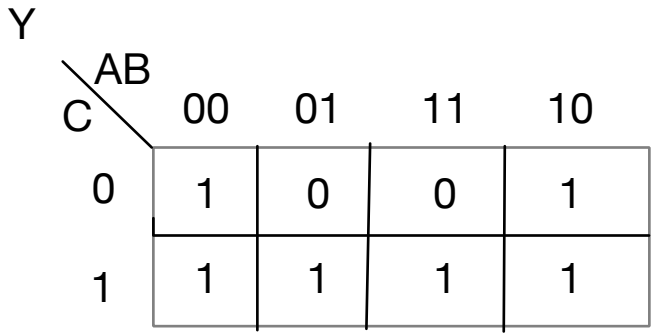


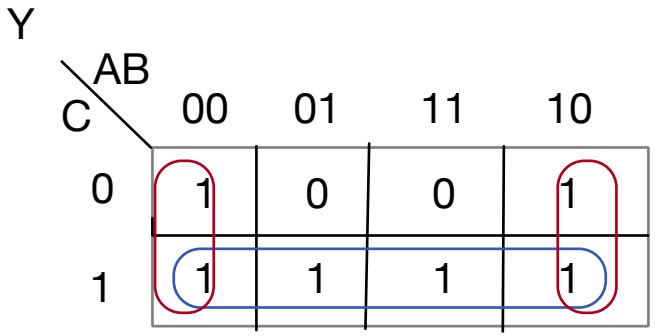


$Y = B + A'C'$

Example 7

- Simplify the Boolean equation `Y = A'BC + (BC')' + BC` using K-maps
- Convert to SOP form and fill in the k-map





`Y = B' + C`

Example 8

- Simplify the Boolean equation
  `Y = ABCD + A'BC'D + (B' + D)'C` using K-maps
- Convert to SOP form and fill in the k-map

Y

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	1	0	1
11	0	0	1	1
10	0	0	0	0

Y

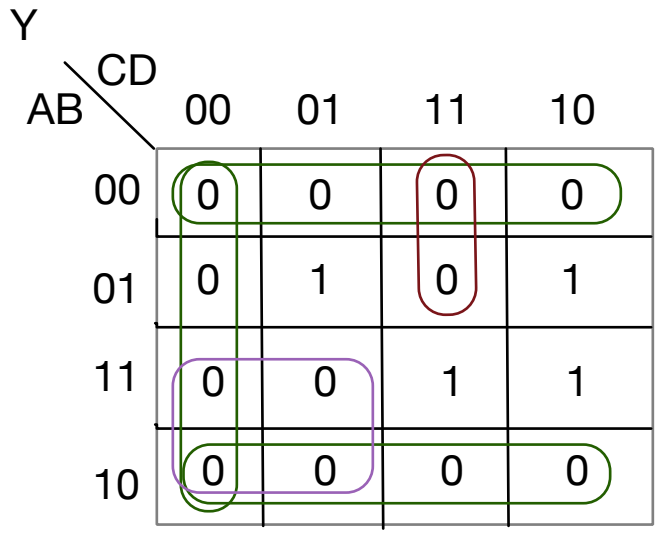
AB \ CD	00	01	11	10
00	0	0	0	0
01	0	1	0	1
11	0	0	1	1
10	0	0	0	0

Y = ABC + BCD' + A'BC'D

Example 9

- Write the simplified POS form for
  `Y = ABCD + A'BC'D + (B' + D)'C` using K-maps
- Convert to SOP form and fill in the k-map

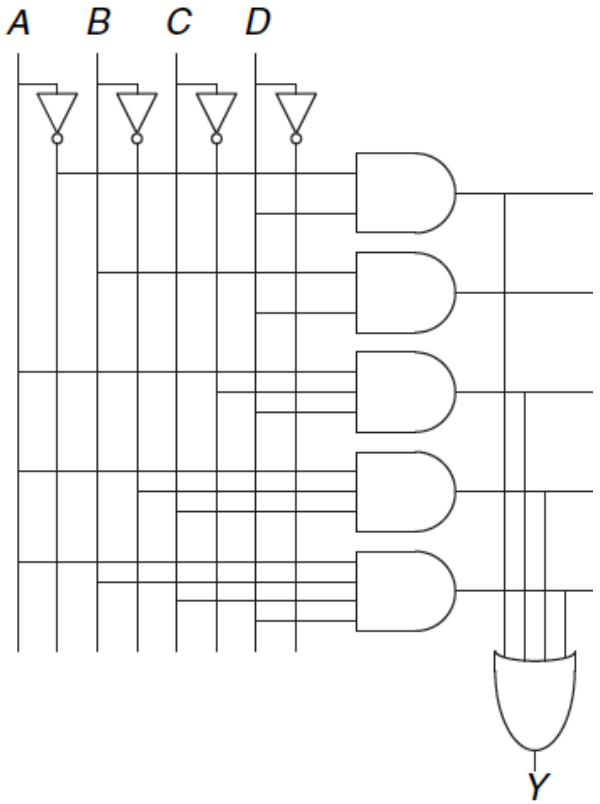
Y	CD 00	01	11	10
AB 00	0	0	0	0
01	0	1	0	1
11	0	0	1	1
10	0	0	0	0



Y = B(C+D) (A'+C) (A+C'+D')

Example 10

- Write a Boolean equation for the following circuit and simplify it using a K-map



Y

AB \ CD	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	1

Y

AB \ CD	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	1

$Y = D + AB'C$