Karnaugh Maps
مخطط كارنوف

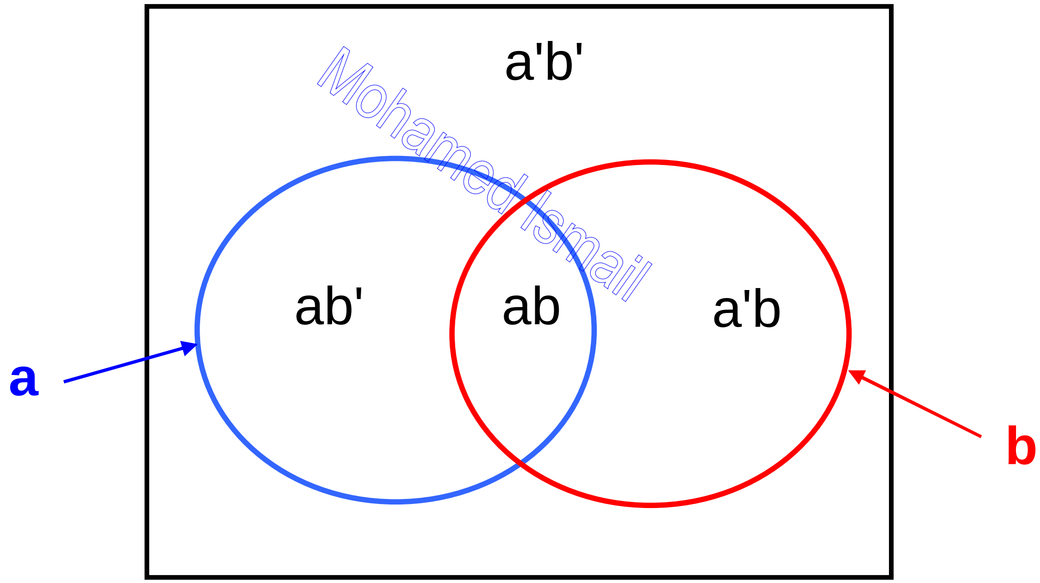
اعداد:محمد اسماعيل
كلية علوم الحاسوب
جامعة امدرمان الاهلية

الاهداء

- الي كل من يسلك طريق العلم والمعرفة
- في هذا المجال

Venn Diagrams

- Venn diagram to represent the space of minterms.
- Example of 2 variables (4 minterms):



Venn Diagrams

- Each set of minterms represents a Boolean function. Examples:

{ ab, ab' } → ab + ab' = a(b+b') = a

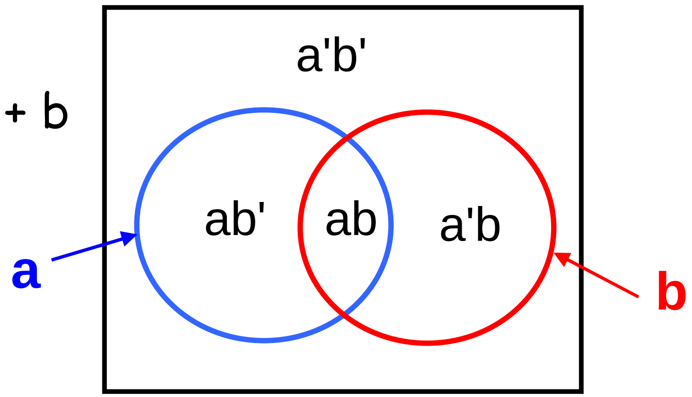
{ a'b, ab } → a'b + ab = (a'+a)b = b

{ ab } → ab

{ ab, ab', a'b } → ab + ab' + a'b = a + b

{ } → 0

{ a'b',ab,ab',a'b } → 1



What are Karnaugh Maps?

A simpler way to handle most (but not all) jobs of <u>manipulating</u> logic functions.

Karnaugh Map Advantages

- Minimization can be done more systematically
- Much simpler to find minimum solutions
- Easier to see what is happening (graphical)

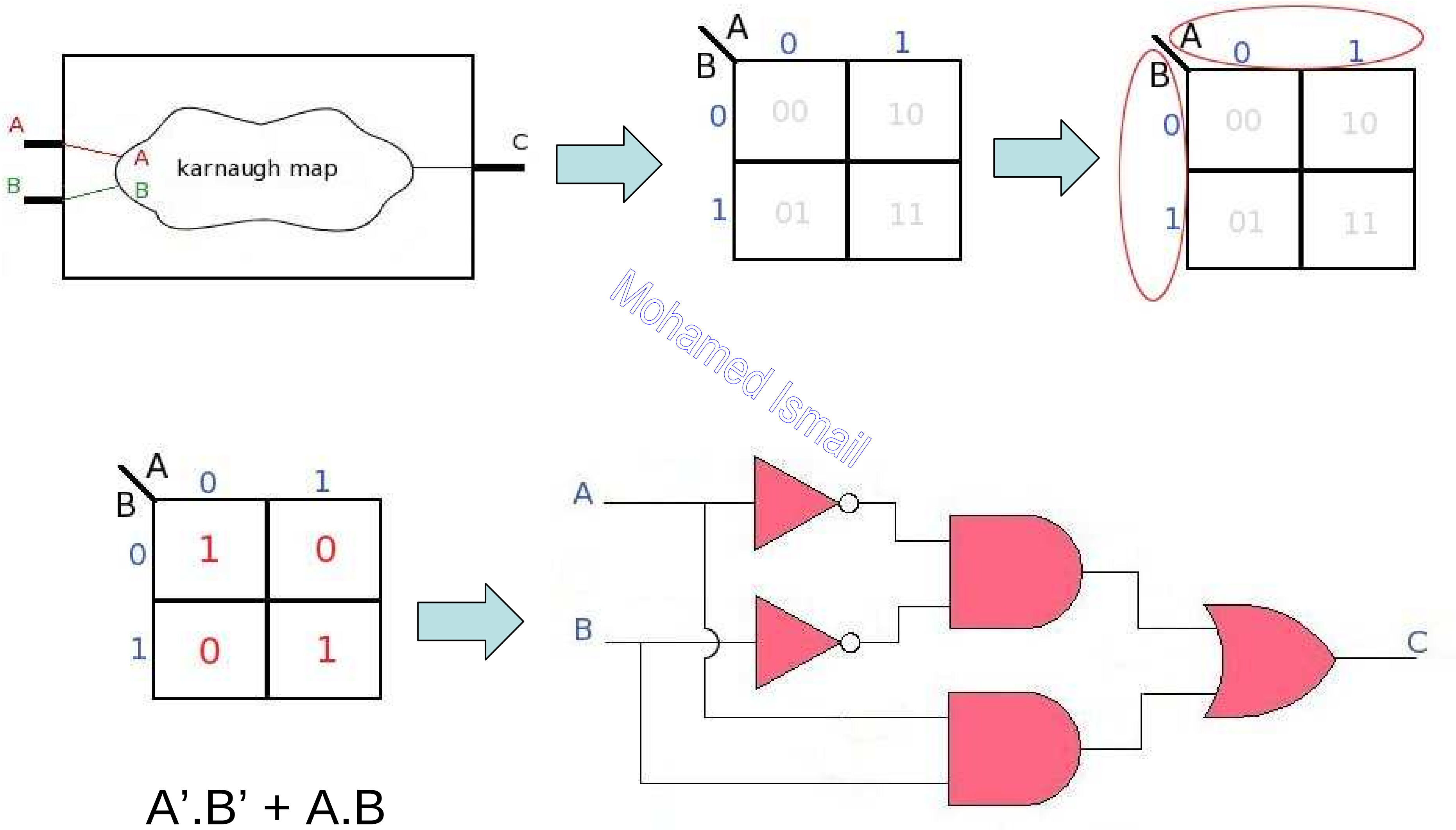
Almost always used instead

of boolean minimization.

Gray Codes

- Gray code is a binary value encoding in which adjacent values only differ by one bit

2-bit Gray Code
00
01
11
10



A’.B’ + A.B

$$F = AB\overline{C} + \overline{A}BC + \overline{A}\,\overline{B}C + A\overline{B}C$$

$$F\left(a,b,c\right) = ab + \overline{b}c$$

$$F(a,b,c) = \sum m(2,3,6,7)$$

$$F(a,b,c) = \overline{a}b + ab = b$$

$$F(a,b,c,d)=\sum m(0,2,3,6,8,12,13,15)$$

$$F=\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}c+\bar{a}c\bar{d}+ab\bar{d}+a\bar{c}\bar{d}$$

$$F(a,b,c,d)=\sum m(0,2,6,8,12,13,15)$$

$$+d(3,9,10)$$

$$F=a\bar{c}+\bar{a}\bar{d}+abd$$

Truth Table Adjacencies

F = A'

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

These are *adjacent in a gray code sense* - they differ by 1 bit

We can apply XY + XY' = X

A'B' + A'B = A'(B'+B) = A'(1) = A'

F = B

A	B	F
0	0	0
0	1	1
1	0	0
1	1	1

Same idea:

A'B + AB = B

Key idea:

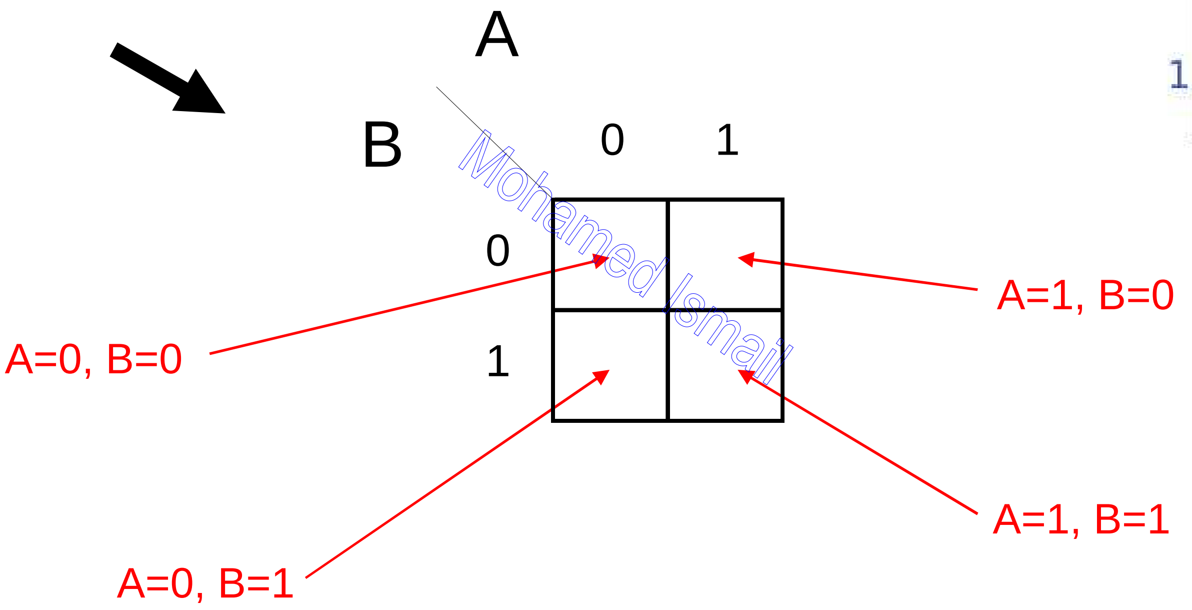
Gray code adjacency allows use of simplification theorems

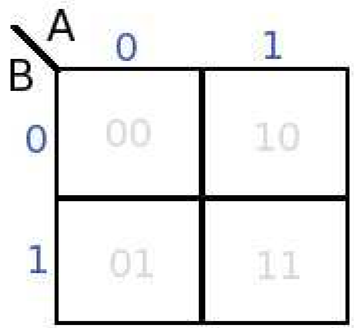
Problem:

Physical adjacency in truth table does not indicate gray code adjacency

2-Variable Karnaugh Map

A	B	F
0	0	
0	1	
1	0	
1	1	

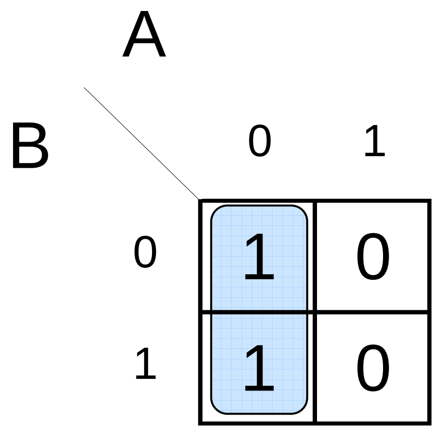




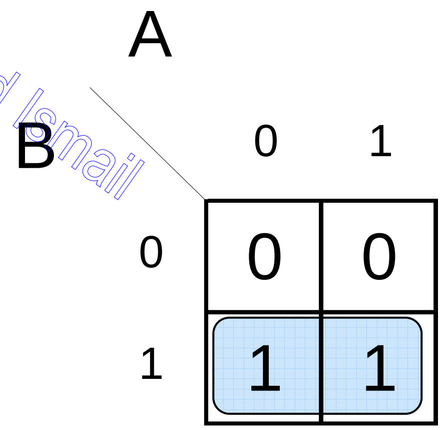
A different way to draw a truth table: by folding it

Karnaugh Map

- In a K-map, physical adjacency *does* imply gray code adjacency



F =A’B’ + A’B = A’



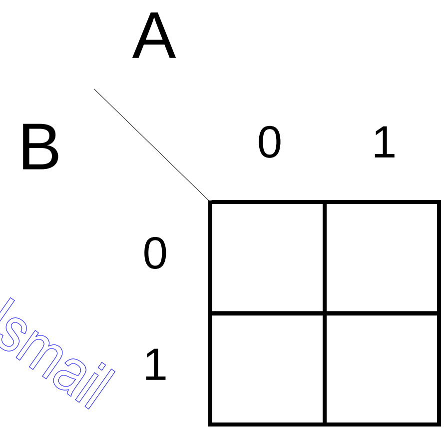
F = A’B + AB = B

2-Variable Karnaugh Map

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

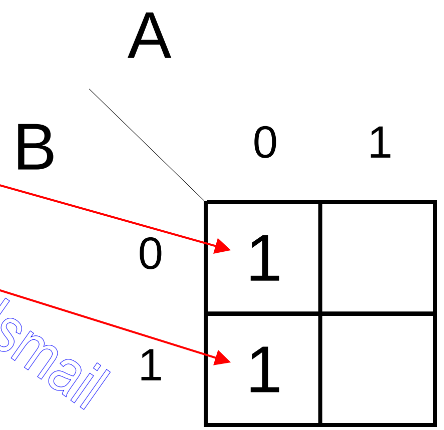
2-Variable Karnaugh Map

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0



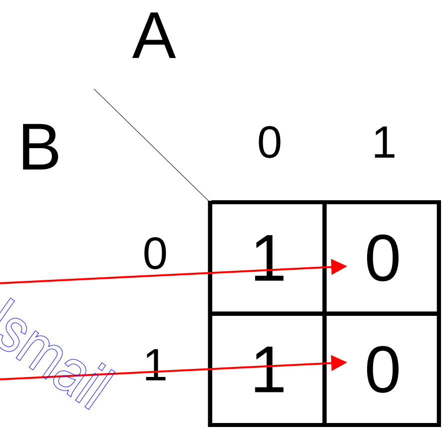
2-Variable Karnaugh Map

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0



2-Variable Karnaugh Map

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0



2-Variable Karnaugh Map

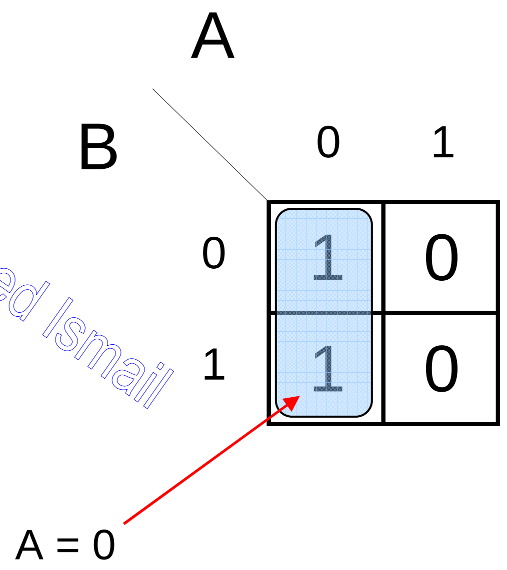
A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

B \ A	0	1
0	1	0
1	1	0

F = A’B’ + A’B = A’

2-Variable Karnaugh Map

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0



F = A'

Another Example

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

B \ A	0	1
0	0	1
1	1	1

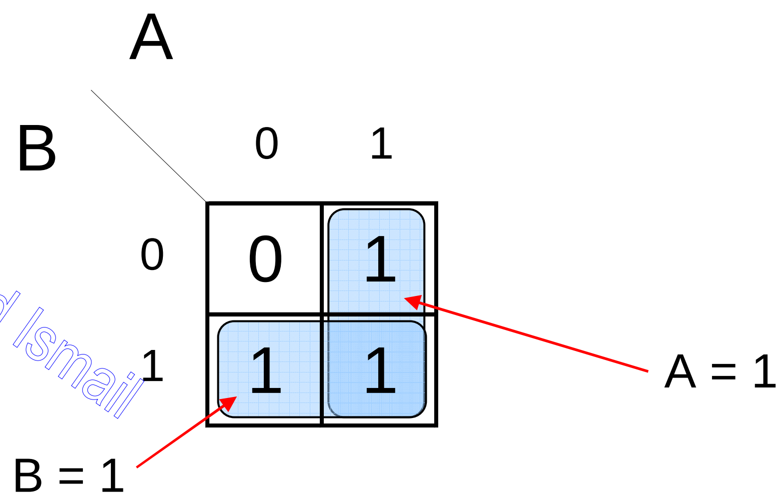
F = A'B + AB' + AB

= (A'B + AB) + (AB' + AB)

= A + B

Another Example

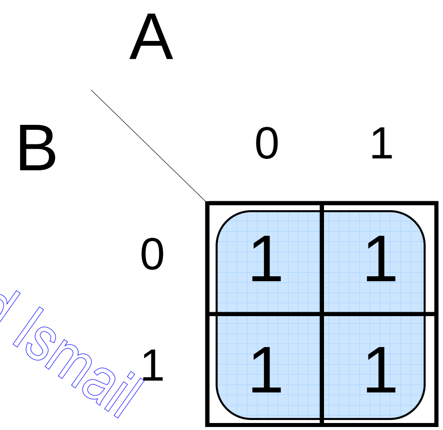
A	B	F
0	0	0
0	1	1
1	0	1
1	1	1



F = A + B

Yet Another Example

A	B	F
0	0	1
0	1	1
1	0	1
1	1	1

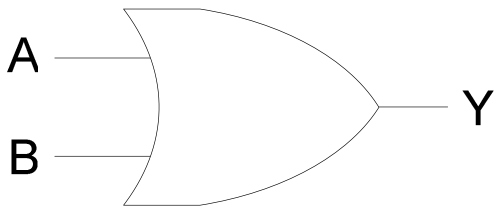


F = 1

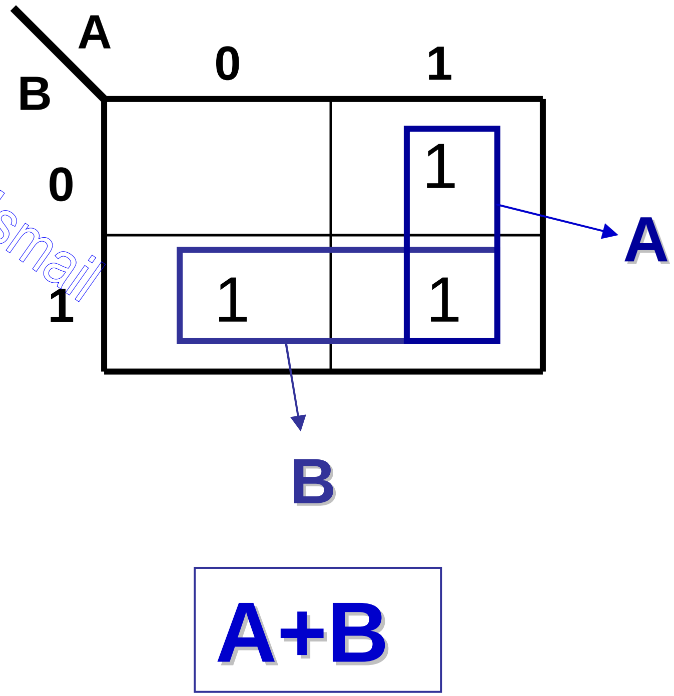
Groups of more than two 1's can be combined

Example

2-variable Karnaugh maps are trivial but can be used to introduce the methods you need to learn. The map for a 2-input OR gate looks like this:



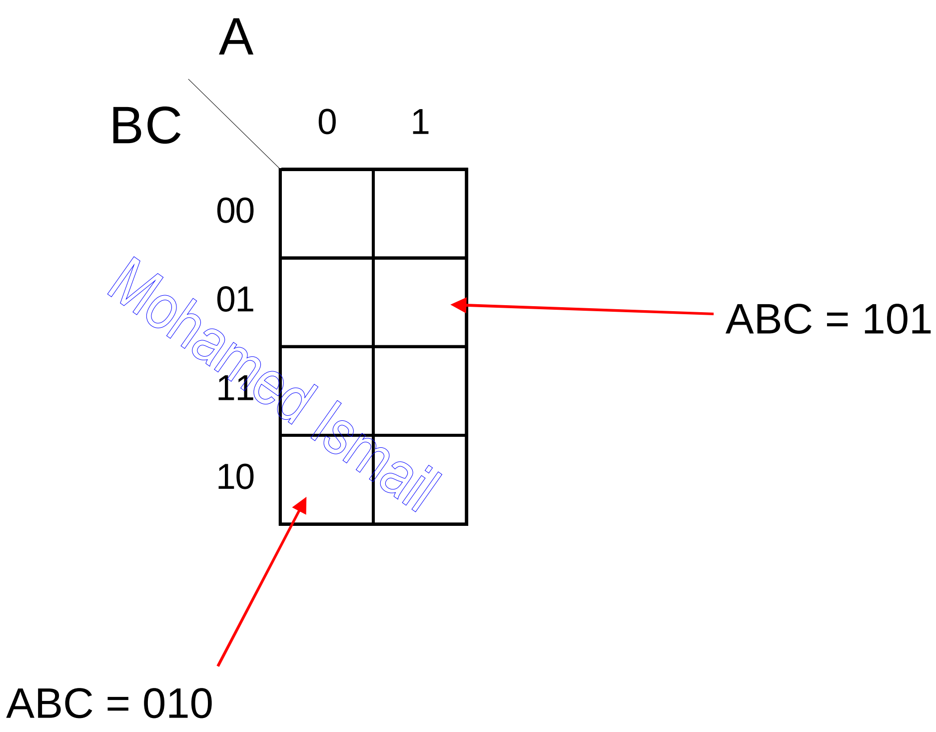
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



3-Variable Karnaugh Map Showing Minterm Locations

Note the order of the B C variables:

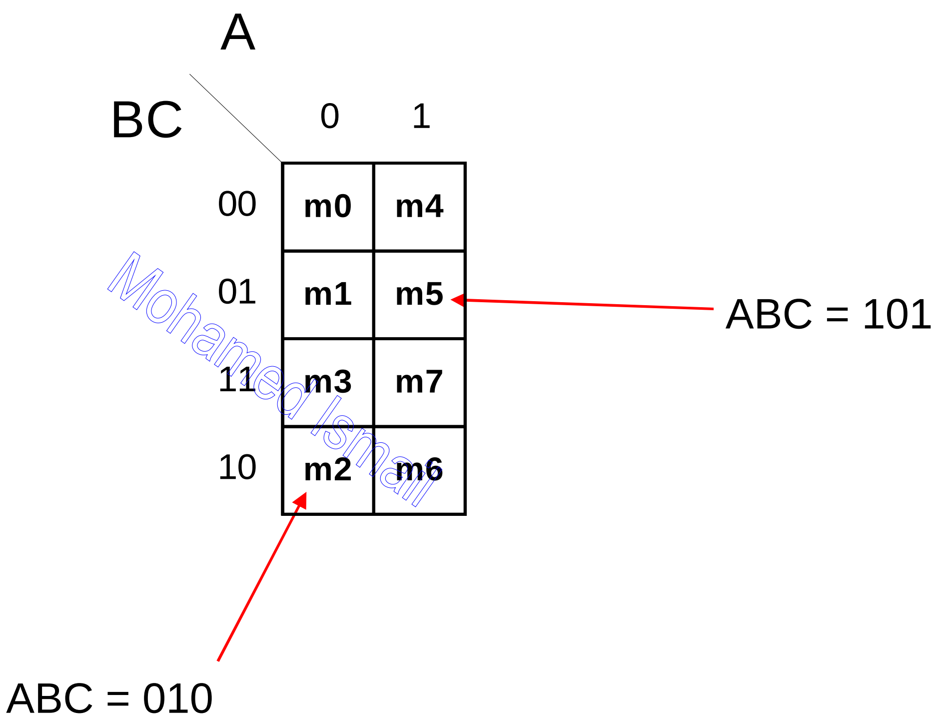
0 0
0 1
1 1
1 0



3-Variable Karnaugh Map Showing Minterm Locations

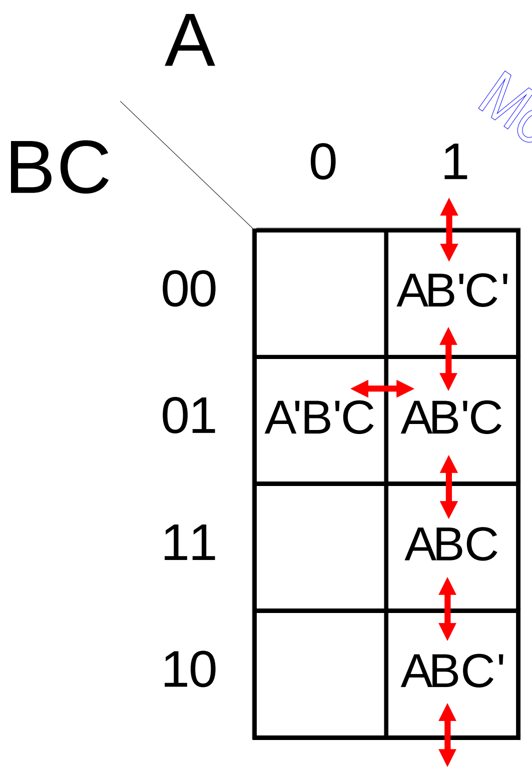
Note the order of the B C variables:

0 0
0 1
1 1
1 0



Adjacencies

- Adjacent squares differ by exactly one variable



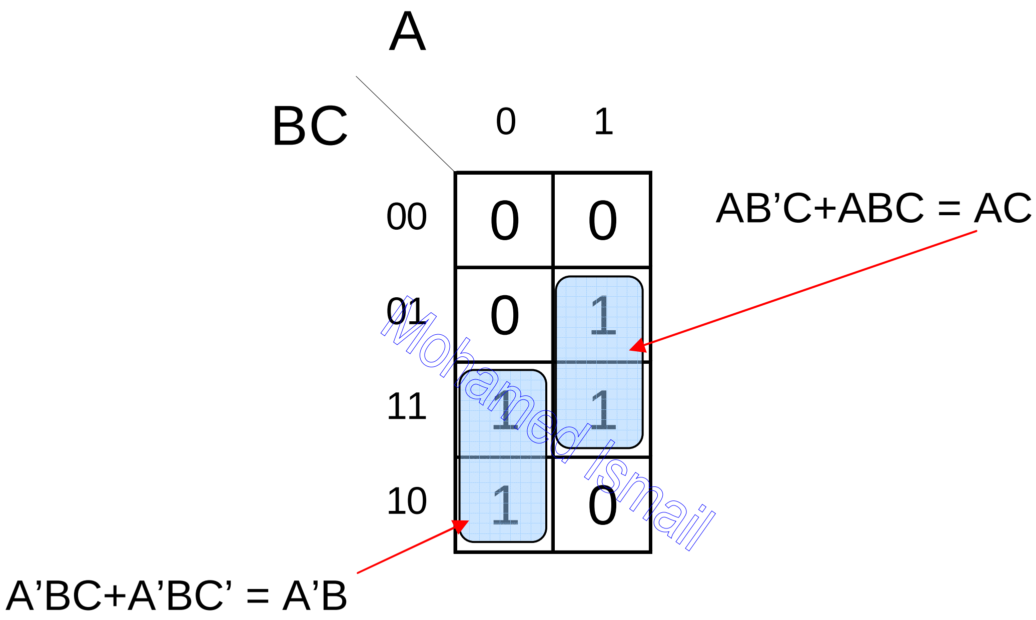
There is wrap-around:
top and bottom rows are adjacent

Truth Table to Karnaugh Map

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

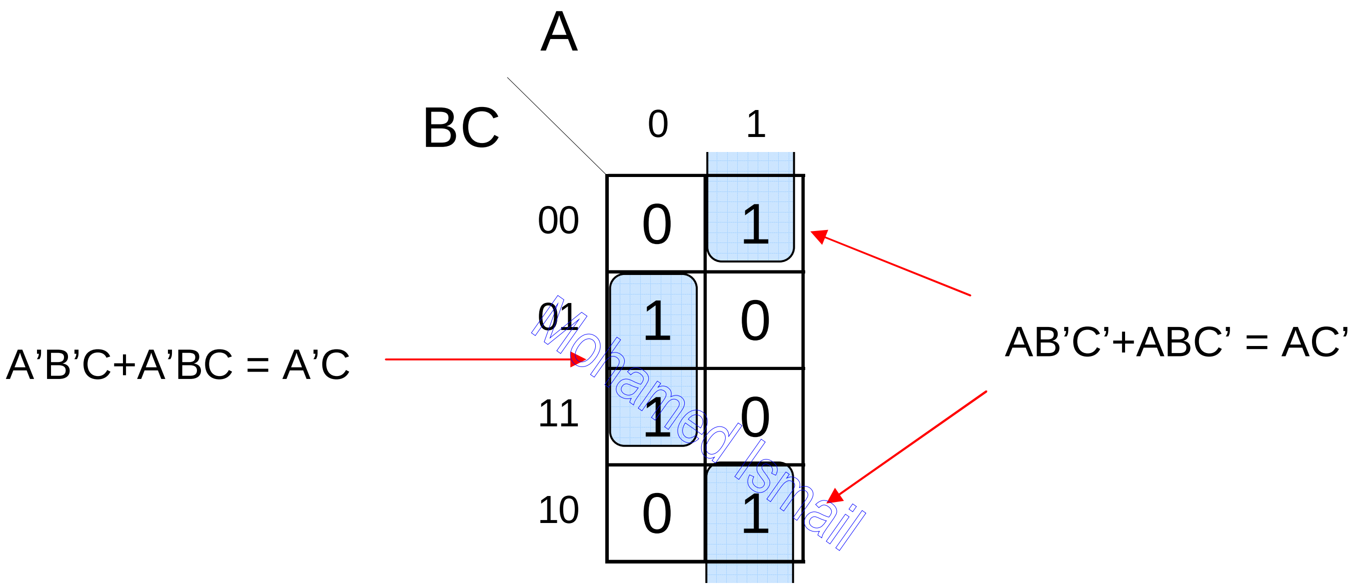
BC \ A	0	1
00	0	0
01	0	1
11	1	1
10	1	0

Minimization Example



F = A’B + AC

Another Example



F = A’C + AC’ = A ⊕ C

Minterm Expansion to K-Map

$$F = \Sigma m(1, 3, 4, 6)$$

BC \ A	0	1
00	m0	m4
01	m1	m5
11	m3	m7
10	m2	m6

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

Minterms are the 1's, everything else is 0

Maxterm Expansion to KMap

$$F = \prod M(0, 2, 5, 7)$$

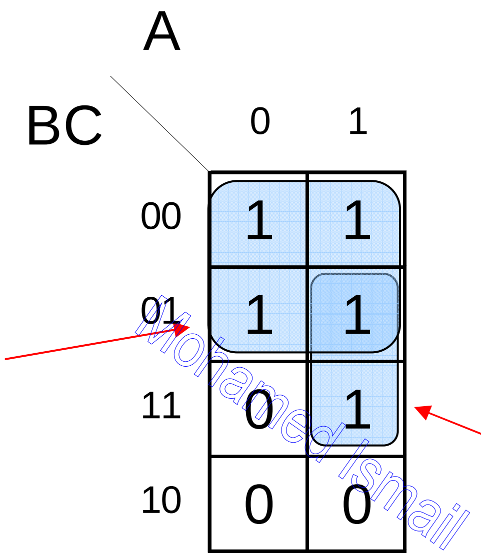
BC \ A	0	1
00	M0	M4
01	M1	M5
11	M3	M7
10	M2	M6

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

Maxterms are the 0's, everything else is 1

Yet Another Example

2^n 1's can be circled at a time
1, 2, 4, 8, … OK
3 not OK



A'B'C'+AB'C'+A'B'C+AB'C = B'

AB'C+ABC = AC

F = B' + AC

The larger the group of 1's
the simpler the resulting product term

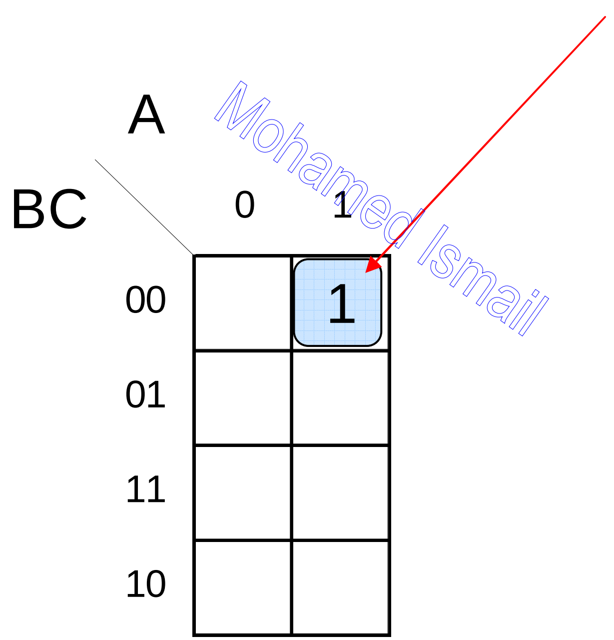
Boolean Algebra to Karnaugh Map

Plot: $ab'c' + bc + a'$

BC \ A	0	1
00		
01		
11		
10		

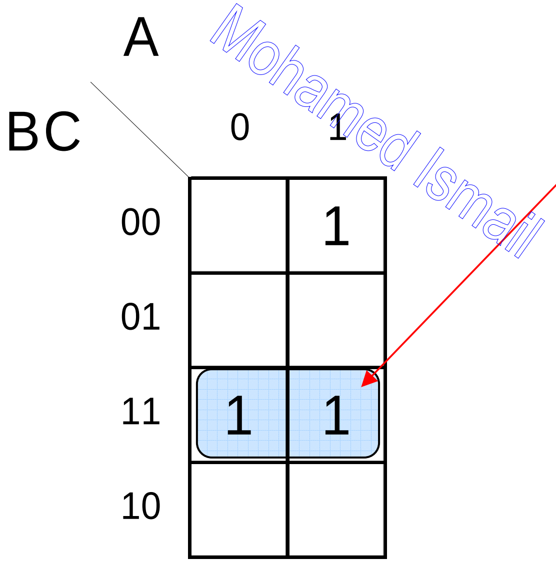
Boolean Algebra to Karnaugh Map

Plot: ab'c' + bc + a'



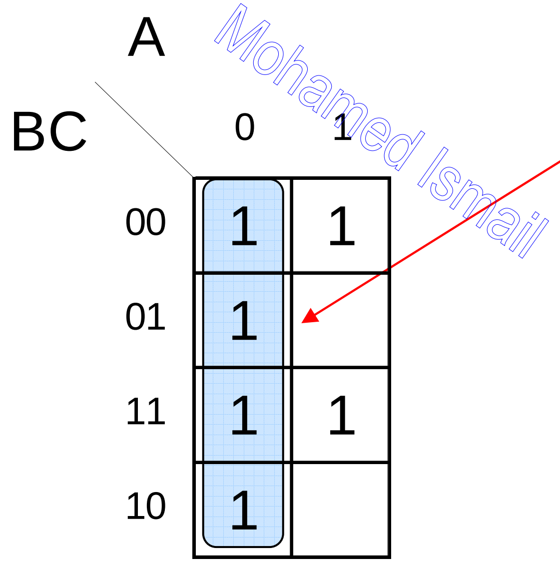
Boolean Algebra to Karnaugh Map

Plot: ab’c’ + bc + a’



Boolean Algebra to Karnaugh Map

Plot: ab'c' + bc + a'



Boolean Algebra to Karnaugh Map

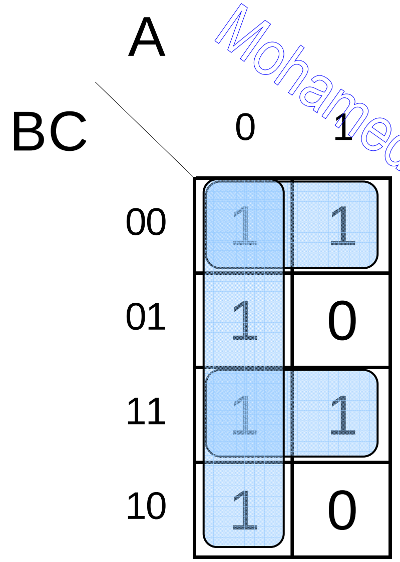
Plot: $ab'c' + bc + a'$

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

Remaining spaces are 0

Boolean Algebra to Karnaugh Map

Now minimize . . .



F = B'C' + BC + A'

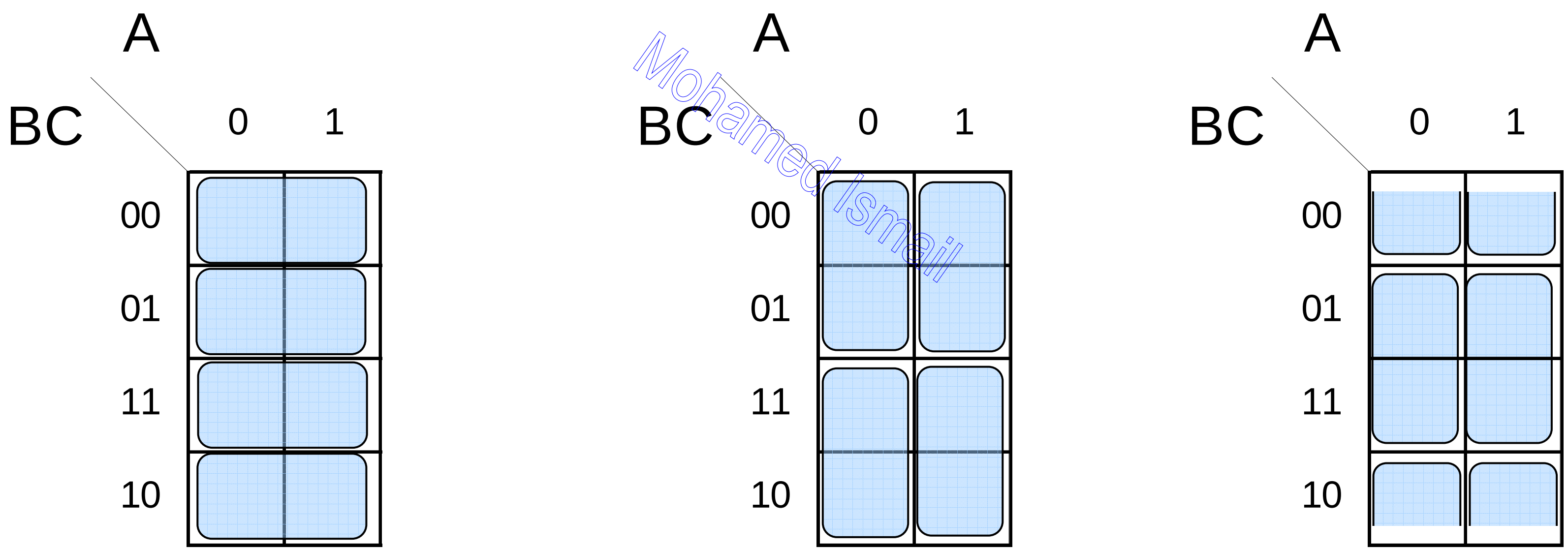
This is a simpler equation than we started with.

Do you see how we obtained it?

Mapping Sum of Product Terms

The 3-variable map has 12 possible groups of 2 spaces

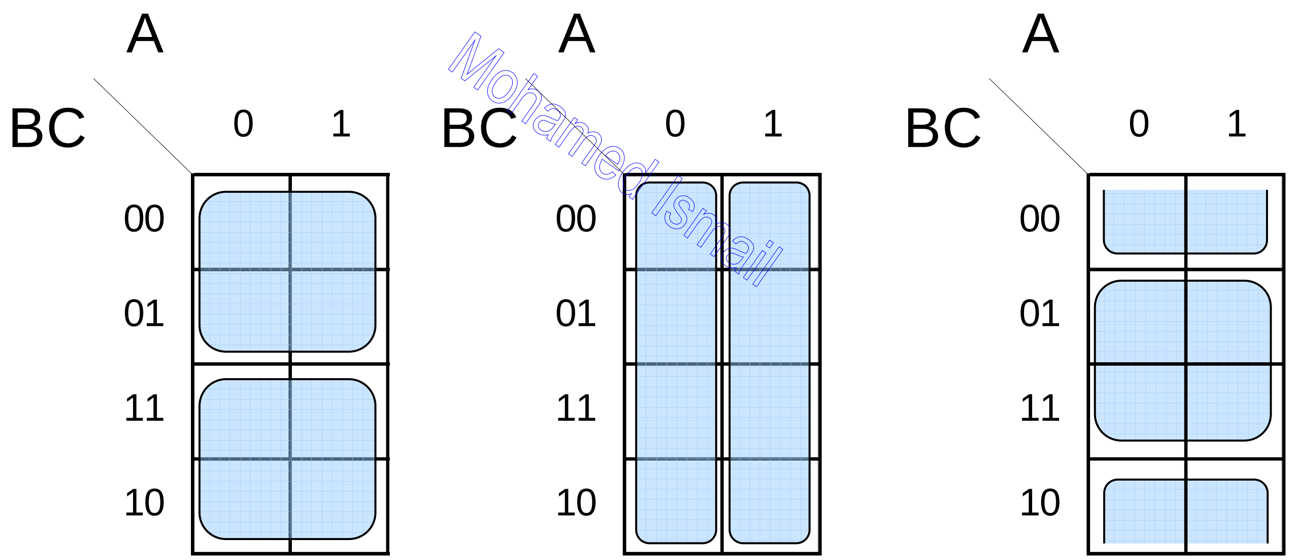
These become terms with 2 literals



Mapping Sum of Product Terms

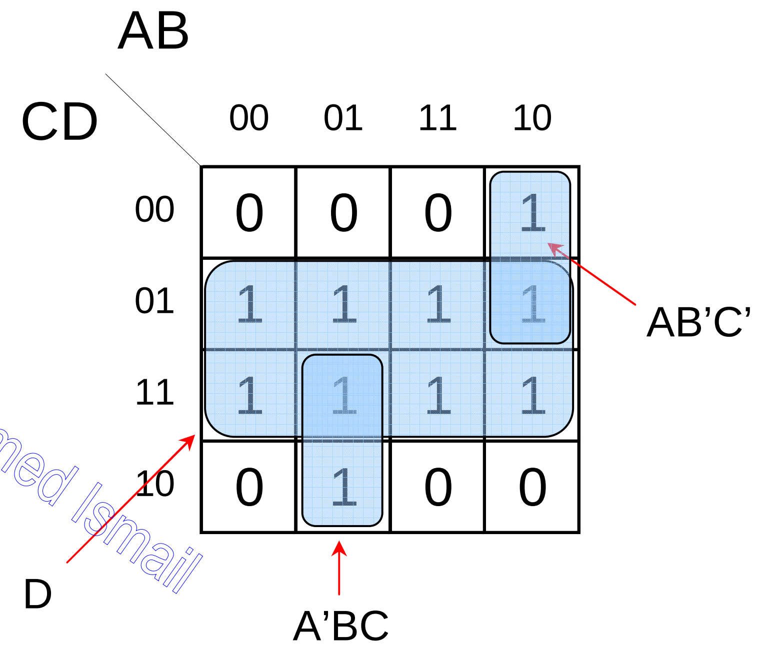
The 3-variable map has 6 possible groups of 4 spaces

These become terms with 1 literal



4-Variable Karnaugh Map

CD \ AB	00	01	11	10
00	**m0**	**m4**	**m12**	**m8**
01	**m1**	**m5**	**m13**	**m9**
11	**m3**	**m7**	**m15**	**m11**
10	**m2**	**m6**	**m14**	**m10**

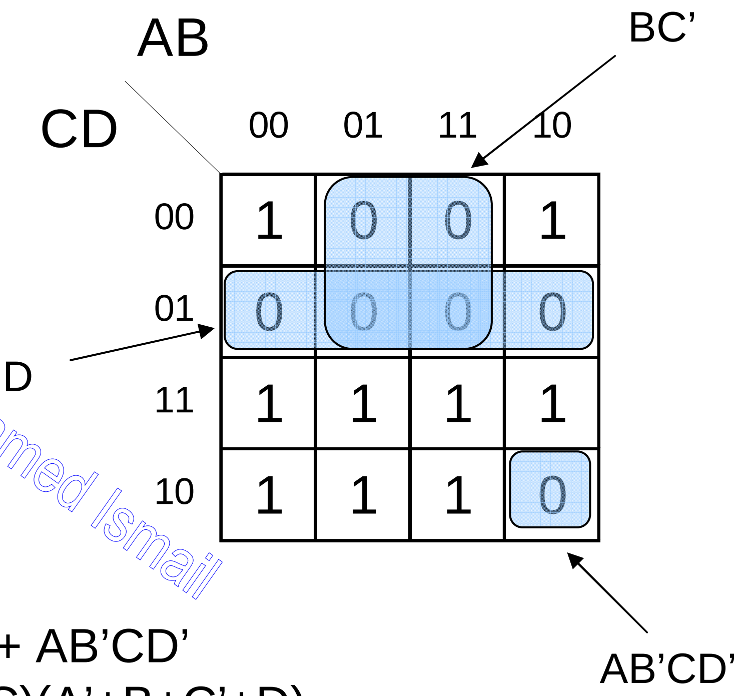


F = A'BC + AB'C' + D

Note the row
and column orderings.

Required for adjacency

Find a POS Solution



F' = C'D + BC' + AB'CD'

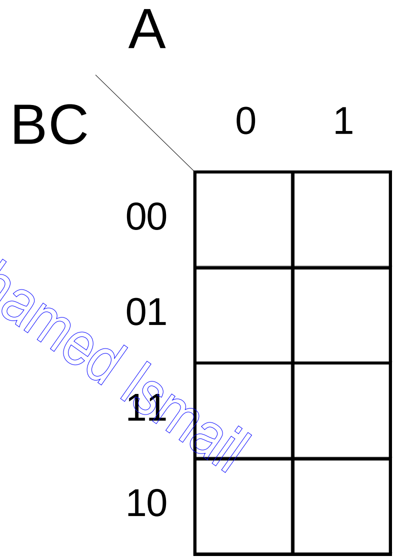
F = (C+D')(B'+C)(A'+B+C'+D)

Find solutions to groups of 0's to find F'

Invert to get F then use DeMorgan's

Dealing With Don't Cares

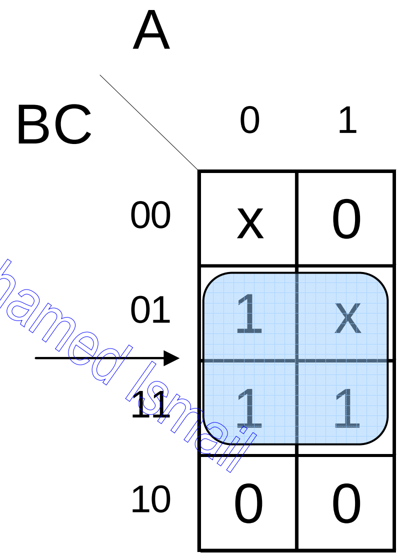
$F = \Sigma m(1, 3, 7) + \Sigma d(0, 5)$



Dealing With Don't Cares

$F = \Sigma m(1, 3, 7) + \Sigma d(0, 5)$

$A'B'C + AB'C + A'BC + ABC = C$



F = C

Circle the x's that help get bigger groups of 1's (or 0's if POS)
Don't circle the x's that don't

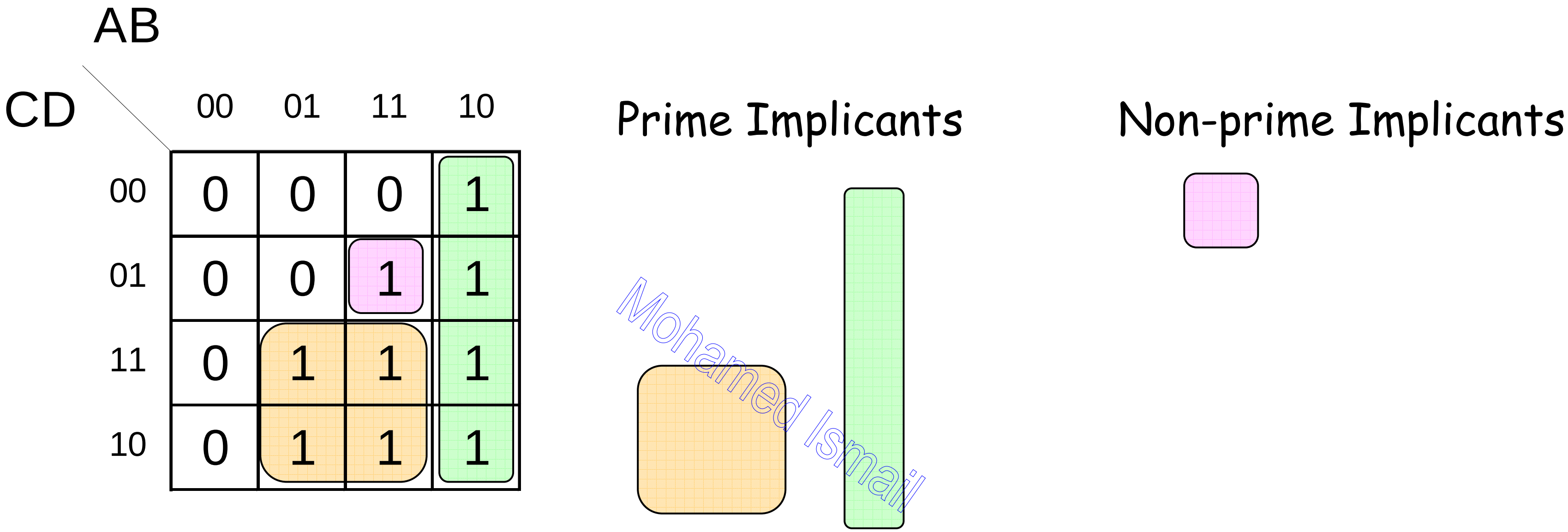
Minimal K-Map Solutions

Some Terminology
and
An Algorithm to Find Them

Prime Implicants

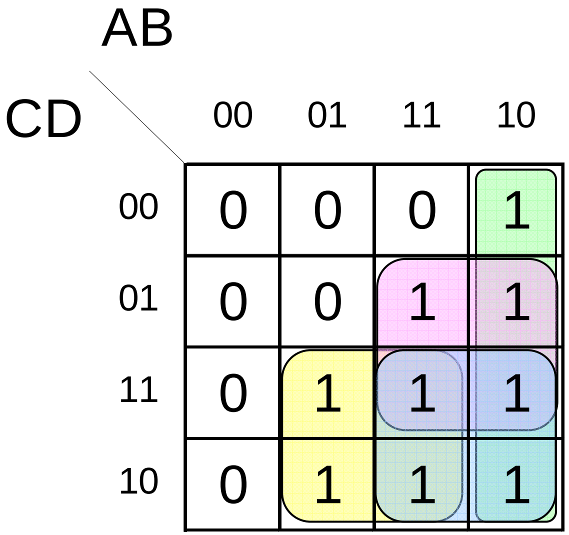
- A group of one or more 1's which are adjacent and can be combined on a Karnaugh Map is called an implicant.
- The *biggest* group of 1's which can be circled to cover a given 1 is called a prime implicant.
  - They are the only implicants we care about.

Prime Implicants

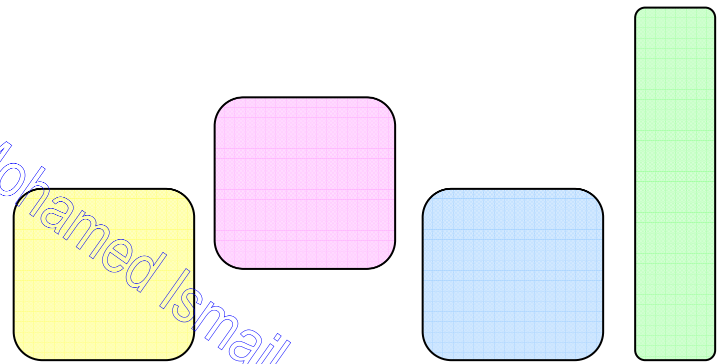


Are there any additional prime implicants in the map that are not shown above?

All The Prime Implicants



Prime Implicants



When looking for a minimal solution – ***only*** circle prime implicants…

A minimal solution will ***never*** contain non-prime implicants

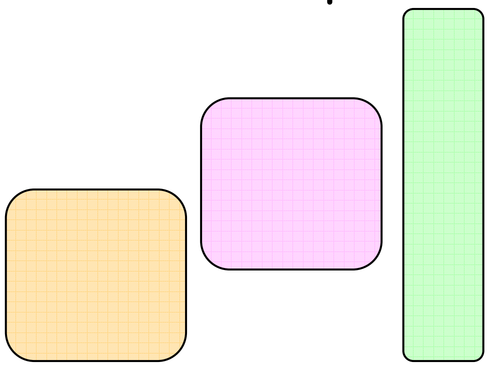
Essential Prime Implicants

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

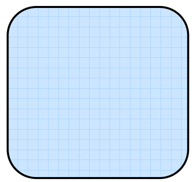
Not all prime implicants are required...

A prime implicant which is the only cover of some 1 is ***essential*** – a minimal solution requires it.

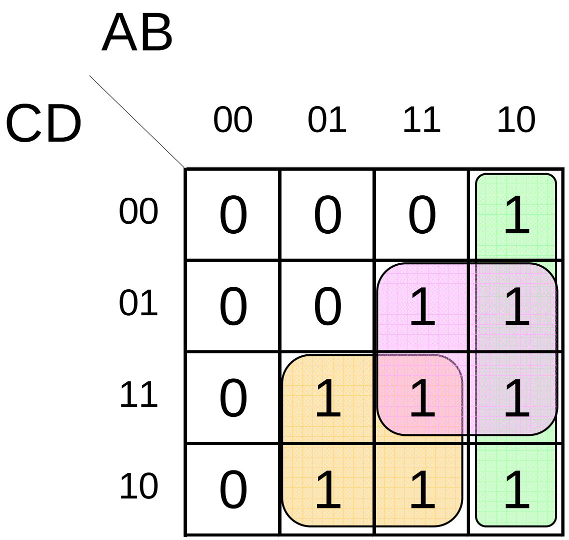
Essential Prime Implicants



Non-essential Prime Implicants



A Minimal Solution Example



Not required...

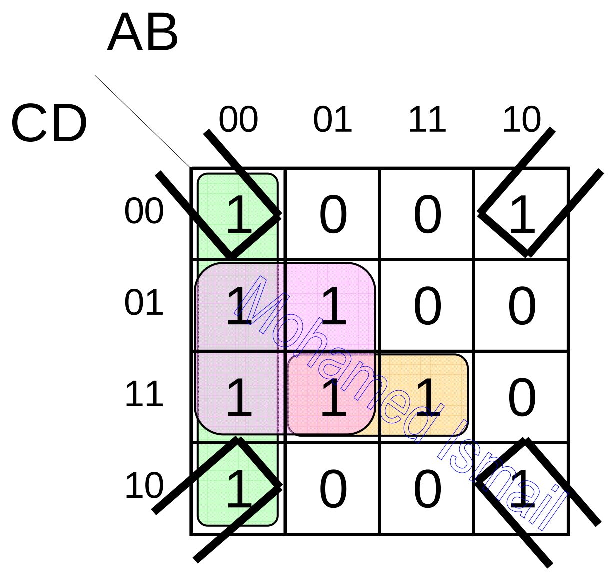
F = AB' + BC + AD

Minimum

Another Example

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

Another Example



A'B' is not required…

Every one one of its locations is covered by multiple implicants

After choosing essentials, everything is covered…

$F = A'D + BCD + B'D'$

Minimum

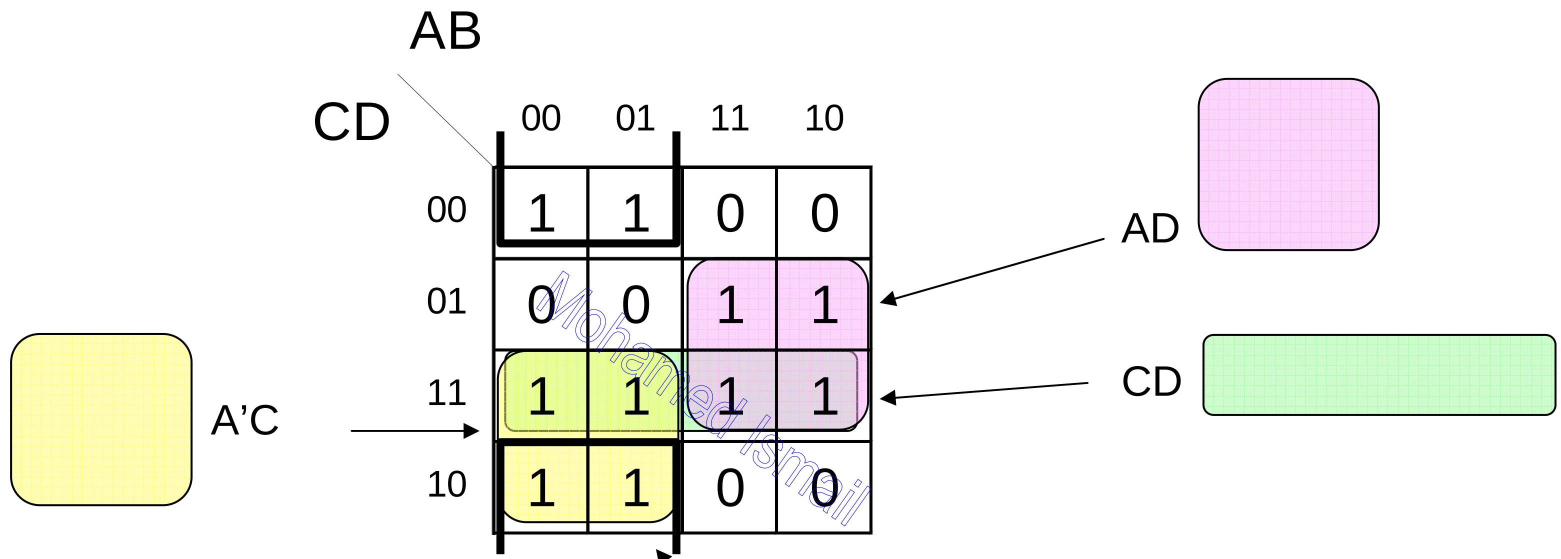
Finding the Minimum Sum of Products

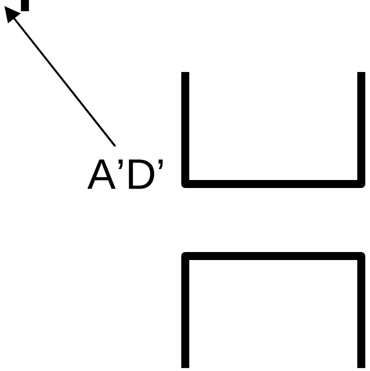
1. Find each essential prime implicant and include it in the solution.
2. Determine if any minterms are not yet covered.
3. Find the minimal # of remaining prime implicants which finish the cover.

Yet Another Example
(Use of non-essential primes)

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

Yet Another Example
(Use of non-essential primes)





Essentials: A'D' and AD
Non-essentials: A'C and CD
Solution: A'D' + AD + A'C
or
A'D' + AD + CD

K-Map Solution Summary

- Identify prime implicants
- Add essentials to solution
- Find a minimum # non-essentials required to cover rest of map

5- and 6-Variable K-Maps

5-Variable Karnaugh Map

DE \ BC	00	01	11	10
00	**m0**	**m4**	**m12**	**m8**
01	**m1**	**m5**	**m13**	**m9**
11	**m3**	**m7**	**m15**	**m11**
10	**m2**	**m6**	**m14**	**m10**

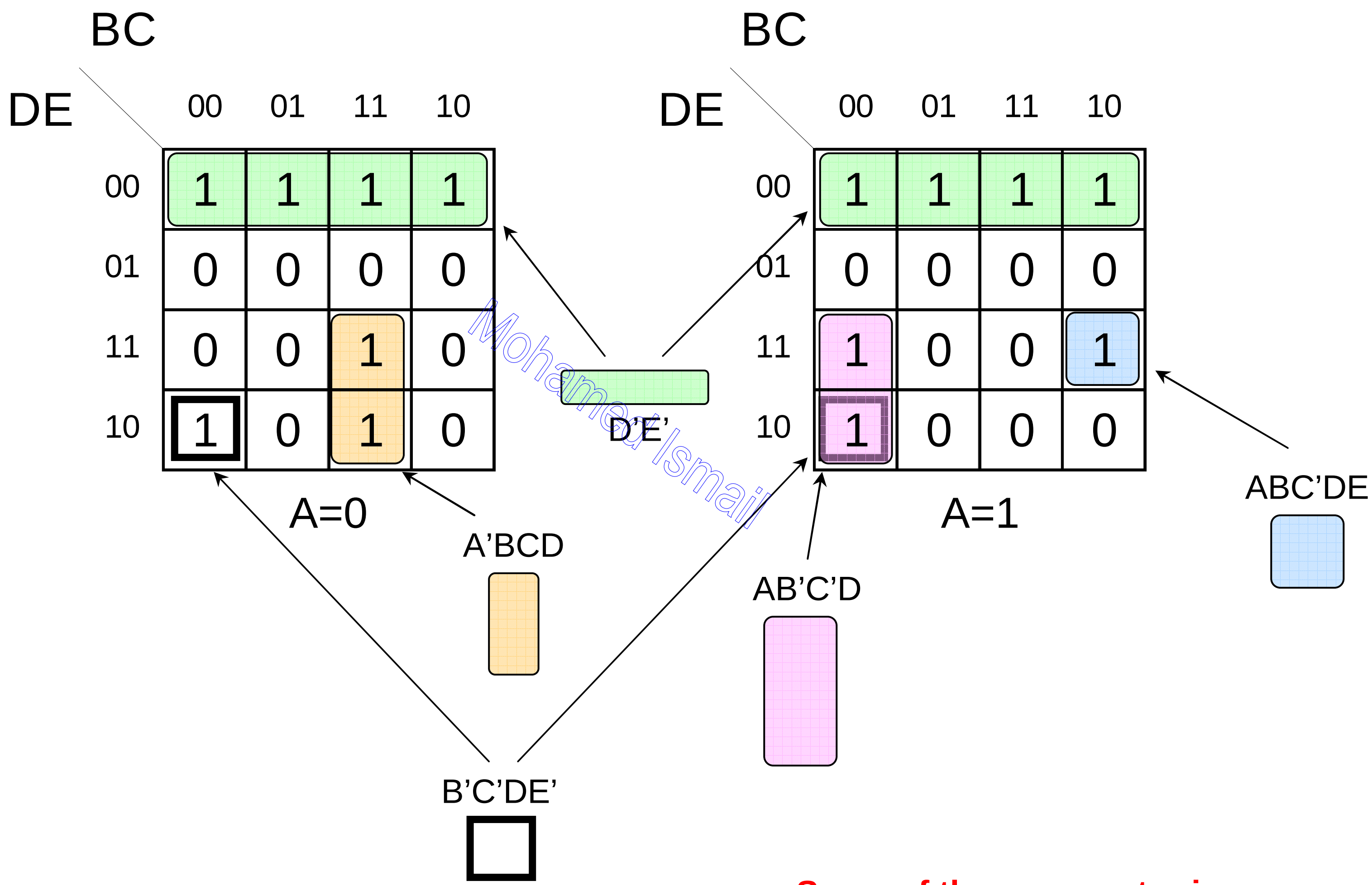
This is the A=0 plane

DE \ BC	00	01	11	10
00	**m16**	**m20**	**m28**	**m24**
01	**m17**	**m21**	**m29**	**m25**
11	**m19**	**m23**	**m31**	**m27**
10	**m18**	**m22**	**m30**	**m26**

This is the A=1 plane

The planes are adjacent to one another (one is above the other in 3D)

Some Implicants in a 5-Variable KMap

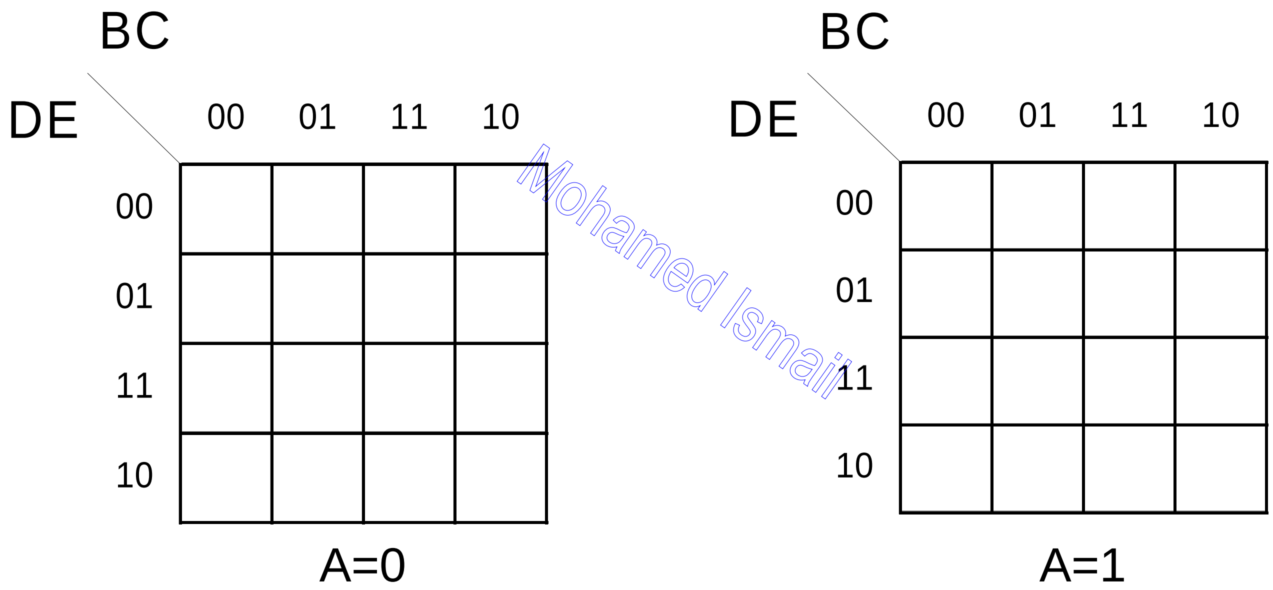


Some of these are not prime…

5-Variable KMap Example

Find the minimum sum-of-products for:

$F = \sum m\,(0,1,4,5,11,14,15,16,17,20,21,30,31)$



5-Variable KMap Example

Find the minimum sum-of-products for:

$F = \sum m\,(0,1,4,5,11,14,15,16,17,20,21,30,31)$

A=0

DE \ BC	00	01	11	10
00	1	1	0	0
01	1	1	0	0
11	0	0	1	1
10	0	0	1	0

A=1

DE \ BC	00	01	11	10
00	1	1	0	0
01	1	1	0	0
11	0	0	1	0
10	0	0	1	0

$F = B'D' + BCD + A'BDE$

6-Variable Karnaugh Map

AB=00

EF \ CD	00	01	11	10
00	**m0**	**m4**	**m12**	**m8**
01	**m1**	**m5**	**m13**	**m9**
11	**m3**	**m7**	**m15**	**m11**
10	**m2**	**m6**	**m14**	**m10**

AB=10

EF \ CD	00	01	11	10
00	**m32**	**m36**	**m44**	**m40**
01	**m33**	**m37**	**m45**	**m41**
11	**m35**	**m39**	**m47**	**m43**
10	**m34**	**m38**	**m46**	**m42**

AB=01

EF \ CD	00	01	11	10
00	**m16**	**m20**	**m28**	**m24**
01	**m17**	**m21**	**m29**	**m25**
11	**m19**	**m23**	**m31**	**m27**
10	**m18**	**m22**	**m30**	**m26**

AB=11

EF \ CD	00	01	11	10
00	**m48**	**m52**	**m60**	**m56**
01	**m49**	**m53**	**m61**	**m57**
11	**m51**	**m55**	**m63**	**m59**
10	**m50**	**m54**	**m62**	**m58**

AB=00

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

AB=10

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

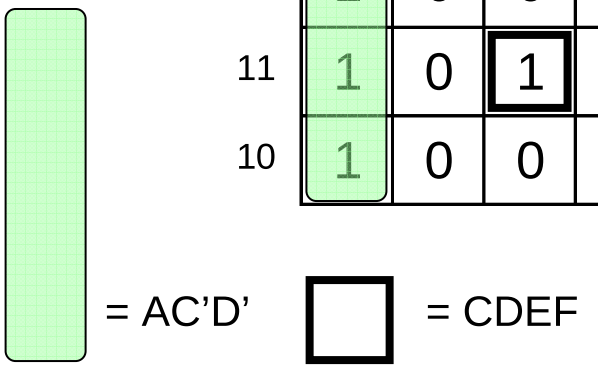
AB=01

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

AB=11

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

Solution = AC'D' + CDEF



KMap Summary

- A Kmap is simply a folded truth table
  - where physical adjacency implies logical adjacency
- KMaps are most commonly used hand method for logic minimization
- KMaps have other uses for visualizing Boolean equations
  - you may see some later.