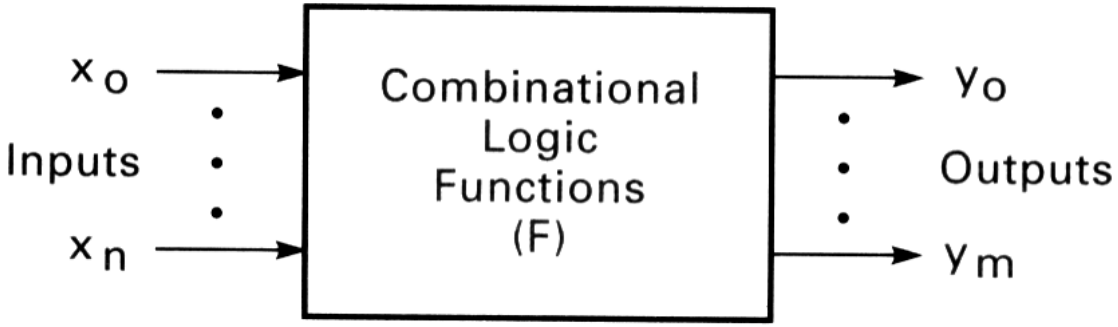
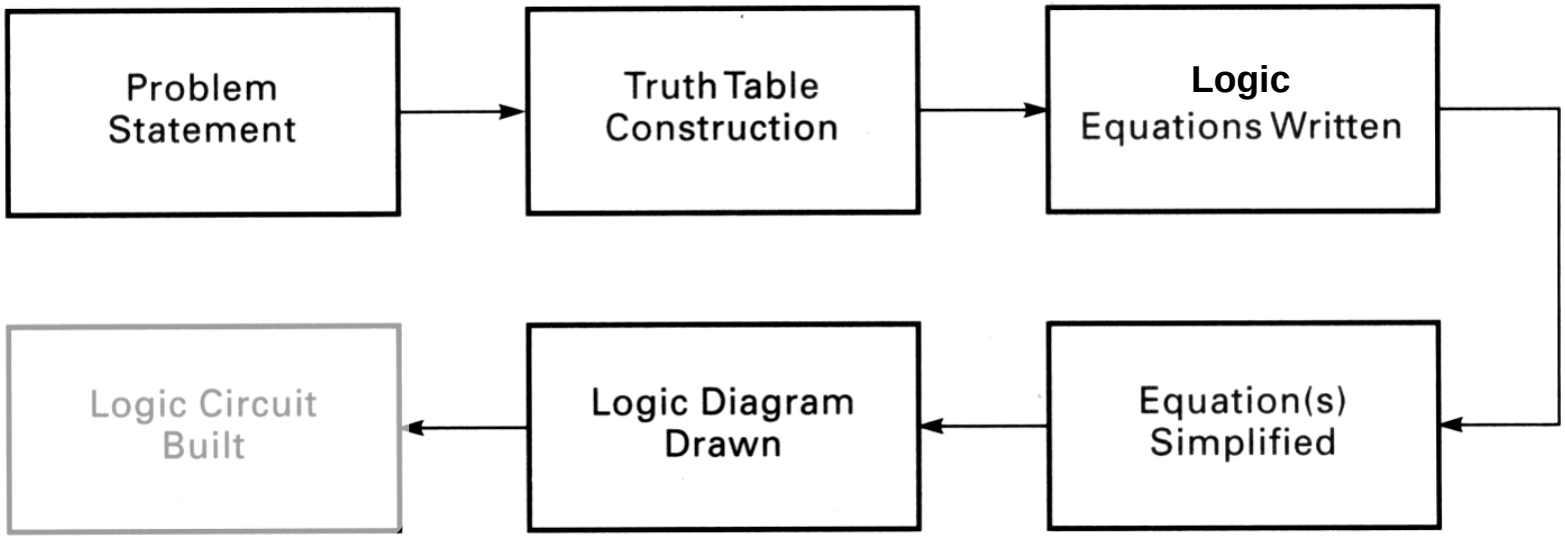
Combinational logic

Logic functions we consider in this course are **combinational** logic functions. The related logic circuits have no memory or feedback from output to input (in contrast to sequential logics).

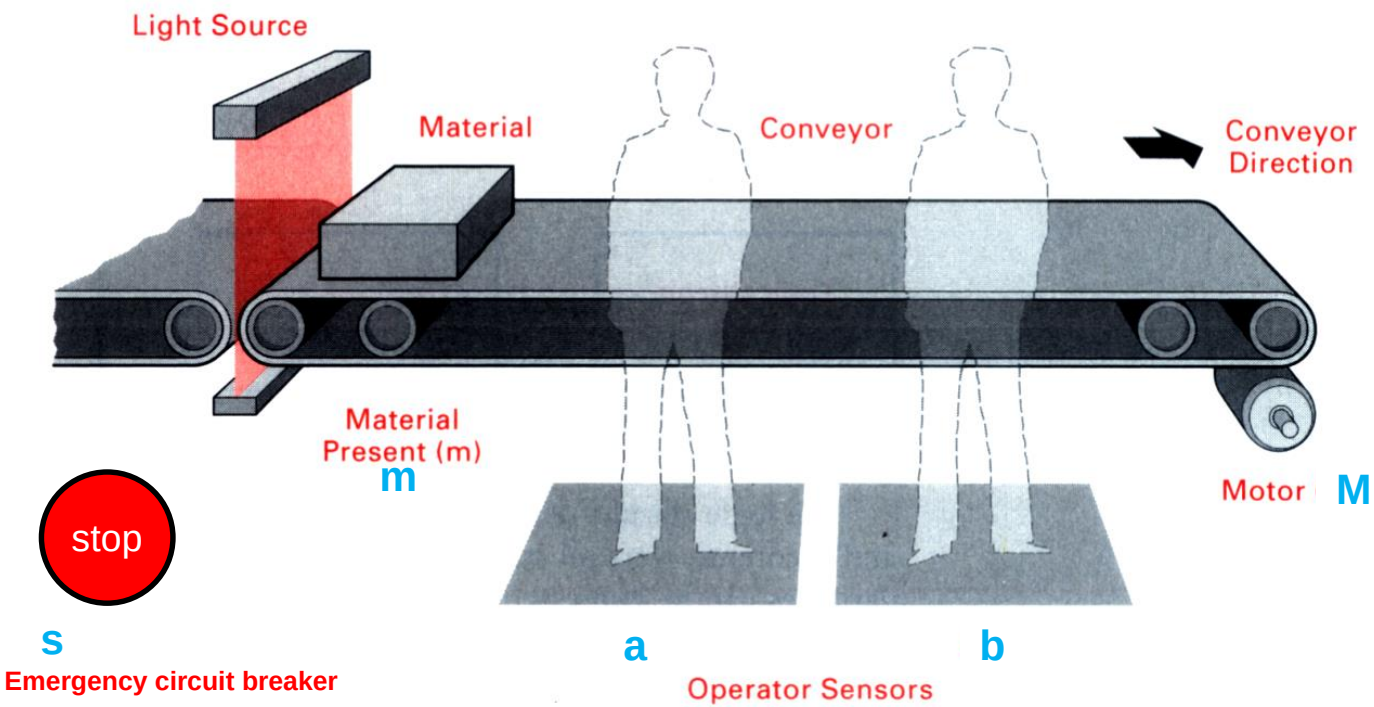
Examples: adder, subtractor, encoder, decoder, multiplier, divider, display driver, keyboard encoder.



General logic design sequence



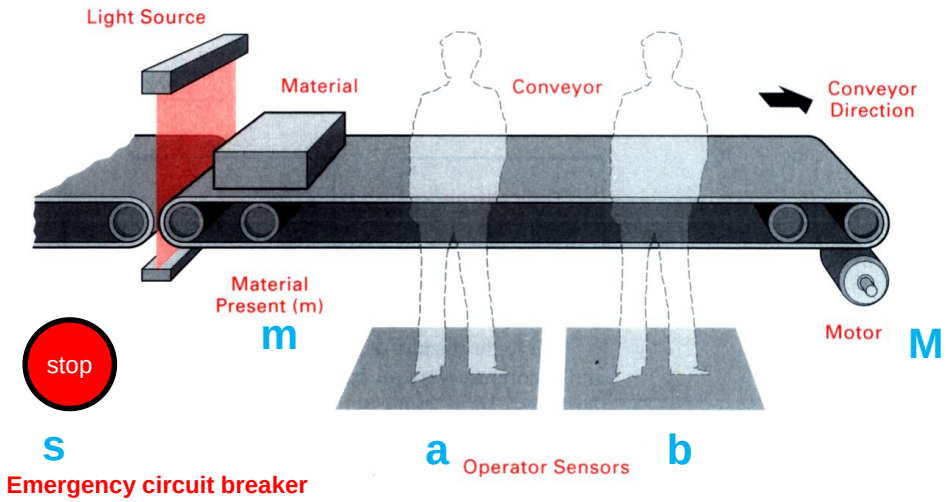
Problem formulation



Establish logical conditions for motor (M) turning on:

- Material (**m**) is present on the conveyor belt.
- At least one of the operators is in place (**a**, **b**).
- Emergency circuit breaker (**s**) is <u>not</u> activated.

Problem statements to truth table

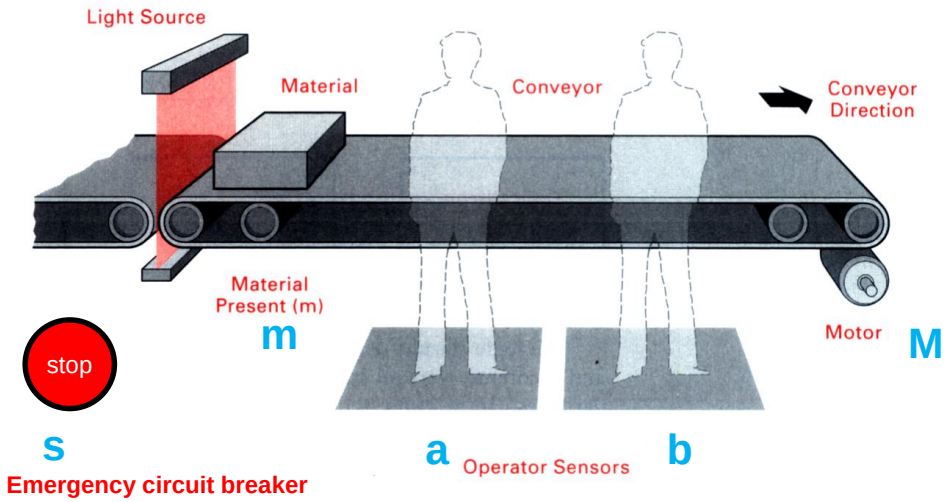


- Convert the established conditions to ‘0’ and ‘1’ states.
- Call **a**, **b**, **s**, and **m** input variables.
- Then **M** is an output variable.
- Number of rows in the truth table: 2^N, N – number of input variables, N=4 hence number of rows: $2^4 = 16$.
- Write down all possible combinations of variables

16

a	*b*	*m*	*s*	*M*
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Problem statements to truth table

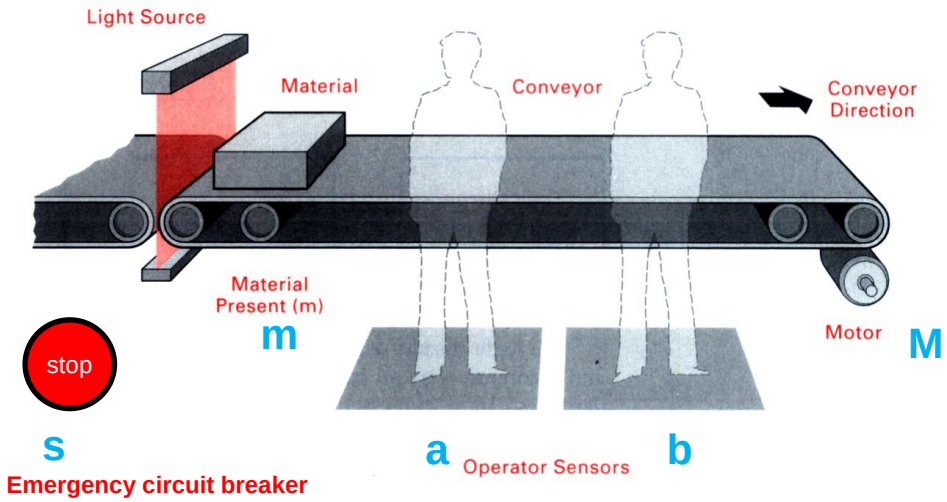


a	b	m	s	M

16

- Convert the established conditions to ‘0’ and ‘1’ states.
- Call **a**, **b**, **s**, and **m** input variables.
- Then **M** is an output variable.
- Number of rows in the truth table: 2^N, N – number of input variables, N=4 hence number of rows: $2^4 = 16$.
- Now we need to write down all possible combinations of the input variables

Problem statements to truth table

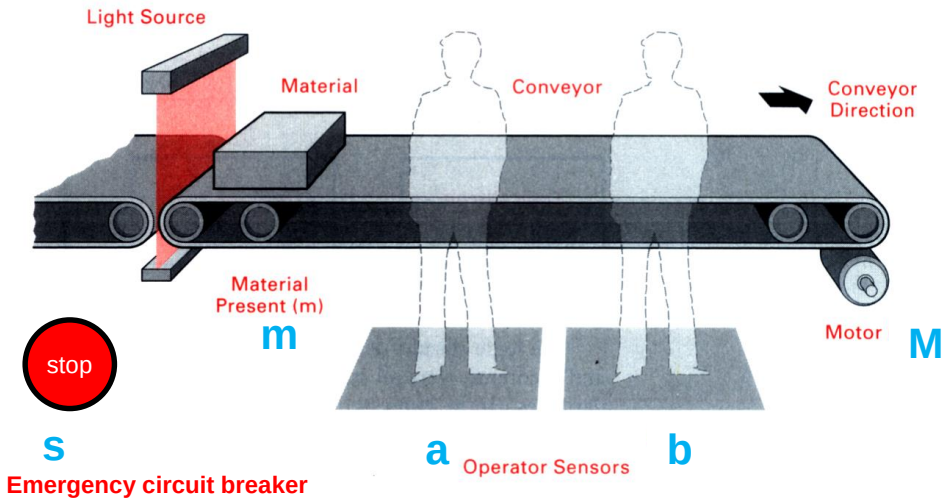


a	*b*	*m*	*s*	*M*
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

One way of filling in all combinations :

- Write down all combinations of **a**, **b**, and **m** variables.
- Then duplicate each row to account for the two states of the emergency switch **s.**

Problem statements to truth table

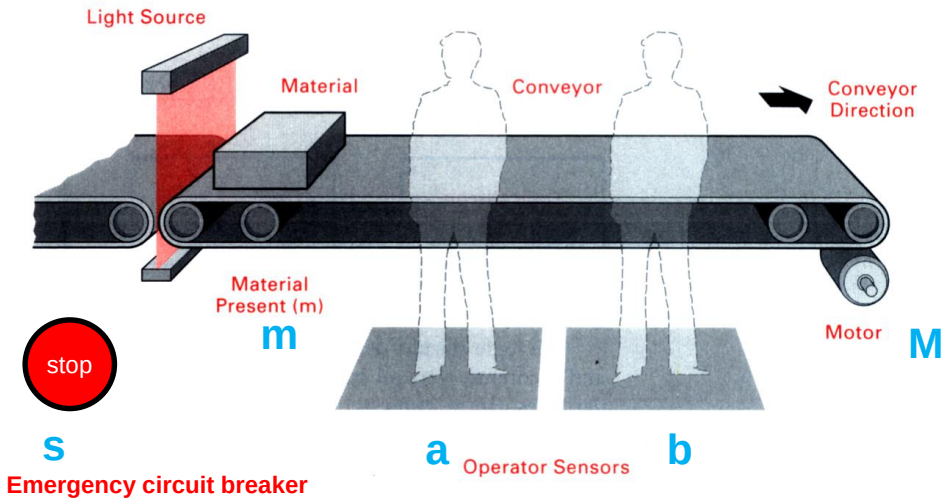


a	*b*	*m*	*s*	*M*
0	0	0	0	
0	0	0	1	
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

One way of filling in all combinations :

- Write down all combinations of **a**, **b**, and **m** variables.
- Then duplicate each row to account for the two states of the emergency switch **s.**

Problem statements to truth table

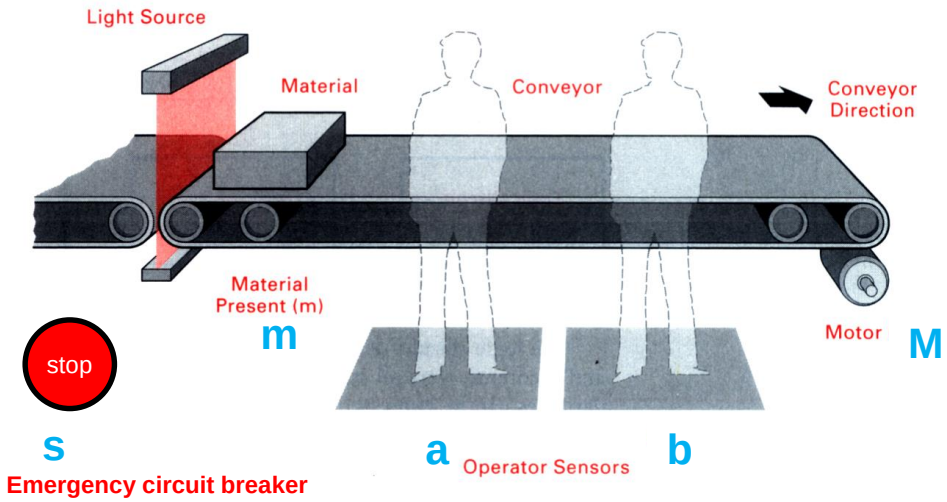


a	*b*	*m*	*s*	*M*
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

One way of filling in all combinations :

- Write down all combinations of **a**, **b**, and **m** variables.
- Then duplicate each row to account for the two states of the emergency switch **s.**

Problem statements to truth table

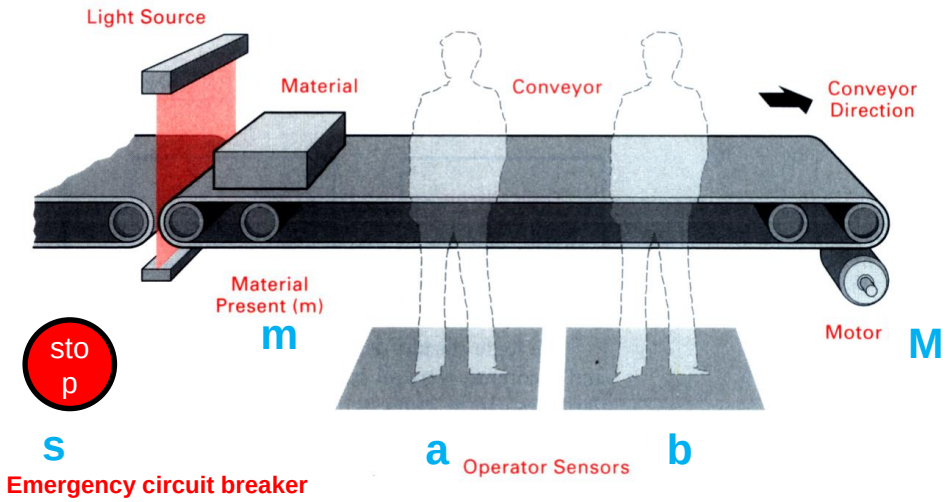


a	*b*	*m*	*s*	*M*
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

One way of filling in all combinations :

- Write down all combinations of **a**, **b**, and **m** variables.
- Then duplicate each row to account for the two states of the emergency switch **s.**

Problem statements to truth table



a	b	m	s	M
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Motor will start (**M**=1) only on three occasions

Recall:

Material (m) is present on the conveyor belt.

At least one of the operators is in place (a, b).

Emergency circuit breaker (s) is <u>not</u> activated.

Writing logic expression: First canonical form

a	b	m	s	M
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

For every entry with '1' at output create AND of inputs

minterms

a'bms

ab'ms

abms

Outputs of these ANDed inputs are ORed together

sum of products

M = a'bms + ab'ms + abms

First canonical form (sum of products)

$$M = f(a, b, m, s) = a'bms + ab'ms + abms$$

If we replace variables with binary values…

$$f(a, b, m, s) = 0111 + 1011 + 1111$$

Convert the binary to decimal values … 7 11 15

We arrive at short form for logic expression

(1st canonical form or sum of products): $f(a, b, m, s) = \mathbf{\Sigma\ (7, 11, 15)}$

sum symbol

First canonical form

Let $f(A,B,C,D) = \Sigma(3, 4, 9, 10)$ be 1st canonical form

To write it as a Boolean expression:

3	→	0011	→	$\bar{A}\bar{B}CD$
4	→	0100	→	$\bar{A}B\bar{C}\bar{D}$
9	→	1001	→	$A\bar{B}\bar{C}D$
10	→	1010	→	$A\bar{B}C\bar{D}$

Hence $F = \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + A\bar{B}\bar{C}D + A\bar{B}C\bar{D}$

Karnaugh map

•Canonical form is not a simplified form of logic expression.

•As we know, Boolean algebra can be used to simplify logic expressions but the process is not straightforward and error prone.

•**Karnaugh map offers a more systematic approach to minimising of logic expressions**, most convenient when the number of input variables < 6.

Building Karnaugh map template

- Two-variable Karnaugh map template. AB sequence.
- All four possible combinations of input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.

The sequence is important!

	$\bar{A}$	A
$\bar{B}$	$\bar{A}\bar{B}$	$A\bar{B}$
B	$\bar{A}B$	AB

!

Building Karnaugh map template

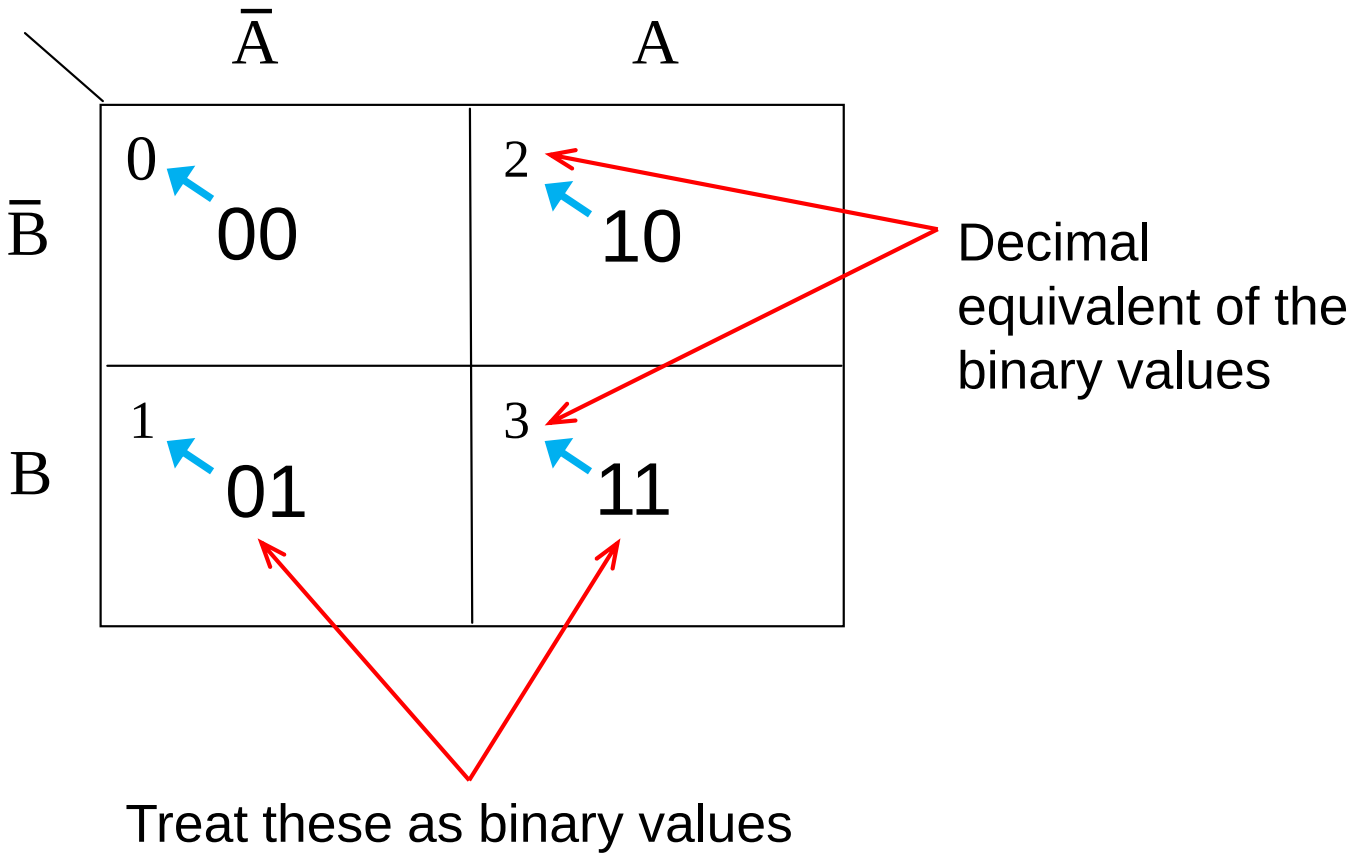
- Two-variable Karnaugh map template. AB sequence.
- All four possible combinations of input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.

	$\bar{A}$	A
$\bar{B}$	00	10
B	01	11

- Use 0 and 1 for simplicity

Building Karnaugh map template

- Two-variable Karnaugh map template. AB sequence.
- All four possible combinations of input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.



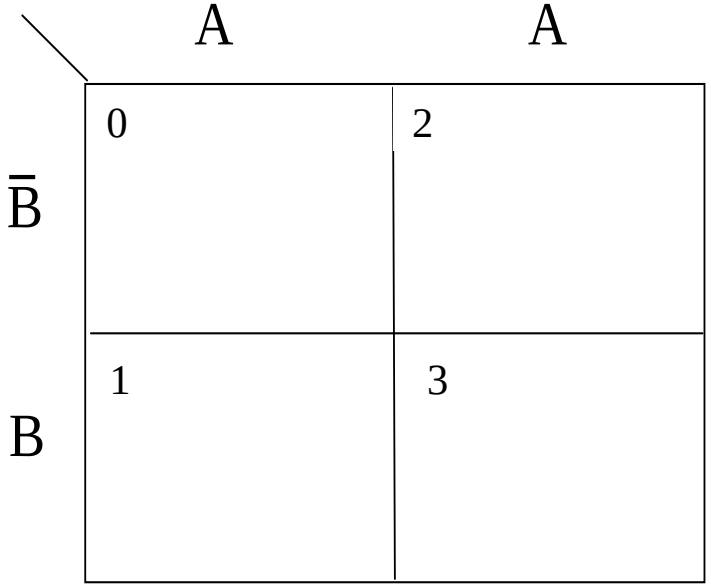
Building Karnaugh map template

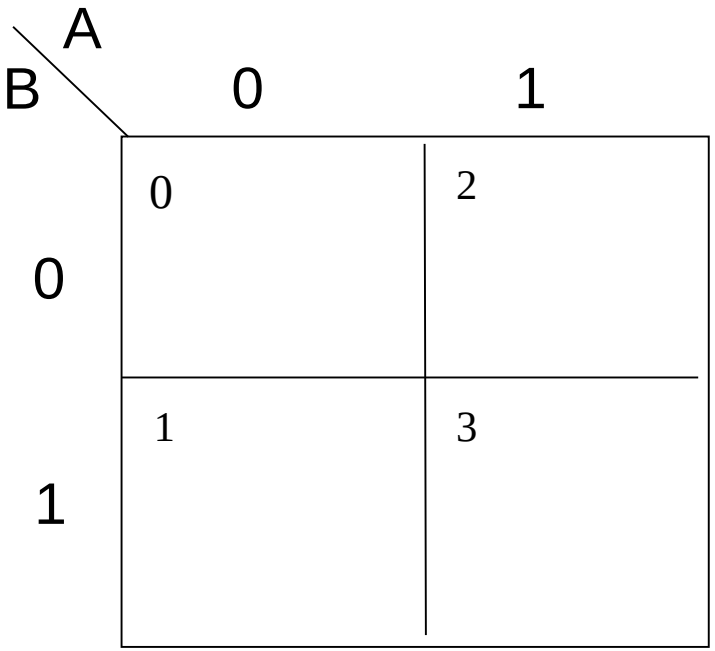
- Two-variable Karnaugh map template. AB sequence.
- All four possible combinations of input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.

	$\bar{A}$	A
$\bar{B}$	0	2
B	1	3

Leave only the decimal indexes for cells

Building Karnaugh map template





Two-variable Karnaugh map template prepared

Loading Karnaugh map

Karnaugh maps contain exactly the same information as truth tables, but this information is arranged visually in a specific way.

$$F(A,B) = A'B + AB' + AB = 01 + 10 + 11 = \sum(\mathbf{1, 2, 3})$$

Truth table

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

Karnaugh map template

B \ A	0	1
0	0	2
1	1	3

Loading Karnaugh map

Karnaugh maps contain exactly the same information as truth tables, but this information is arranged visually in a specific way.

$$F(A,B) = A'B + AB' + AB = 01 + 10 + 11 = \sum(\mathbf{1, 2, 3})$$

Truth table

A	**B**	**F**	
0	0	0	
0	1	1	**01**
1	0	1	**10**
1	1	1	**11**

Karnaugh map template

B \ A	0	1
0	0	2
1	1	3

Loading Karnaugh map

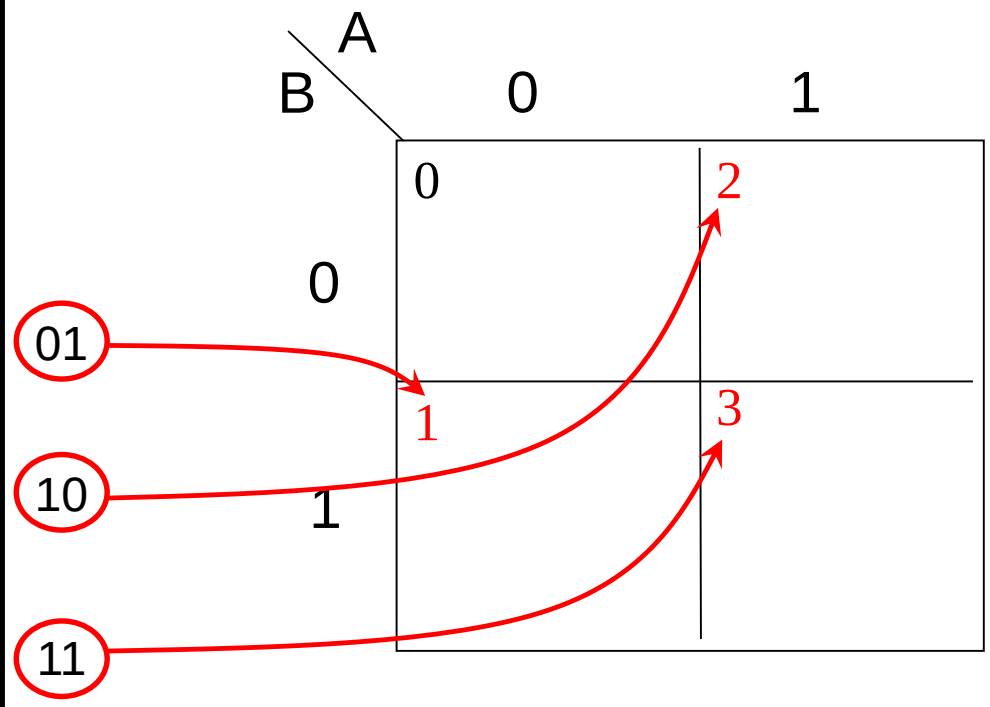
Karnaugh maps contain exactly the same information as truth tables, but this information is arranged visually in a specific way.

$$F(A,B) = A'B + AB' + AB = 01 + 10 + 11 = \sum(\mathbf{1, 2, 3})$$

Truth table

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

Karnaugh map template



Loading Karnaugh map

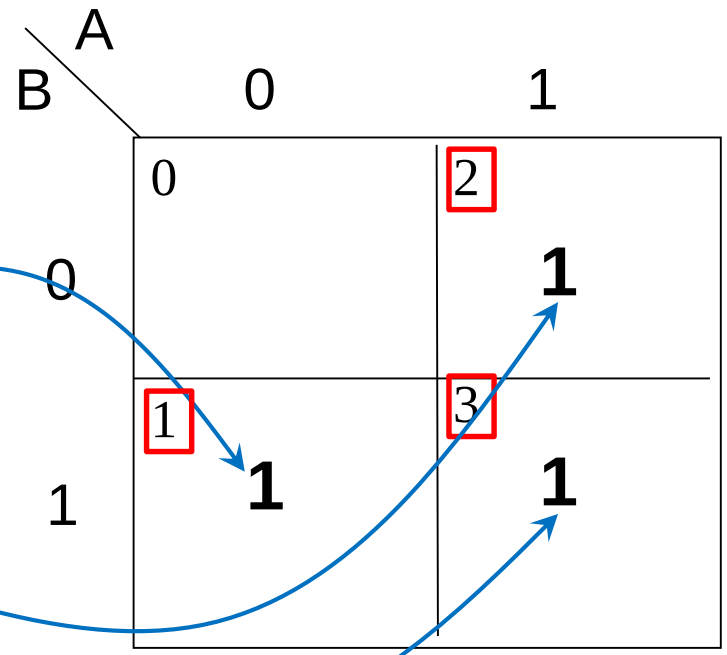
Karnaugh maps contain exactly the same information as truth tables, but this information is arranged visually in a specific way.

$$F(A,B) = A'B + AB' + AB = 01 + 10 + 11 = \sum(\mathbf{1, 2, 3})$$

Truth table

A	B	F	
0	0	0	
0	1	1	01
1	0	1	10
1	1	1	11

Karnaugh map



Loading Karnaugh map

Karnaugh maps contain exactly the same information as truth tables, but this information is arranged visually in a specific way.

$$F(A,B) = A'B + AB' + AB = 01 + 10 + 11 = \sum(\mathbf{1, 2, 3})$$

Truth table

A	B	F	
0	0	0	
0	1	1	01
1	0	1	10
1	1	1	11

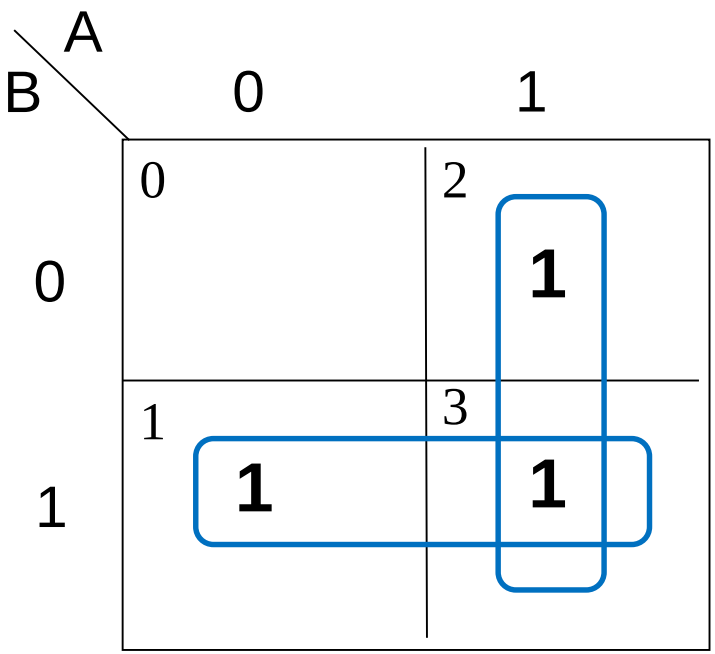
Karnaugh map loaded!

B \ A	0	1
0	0	2 **1**
1	1 **1**	3 **1**

minterms

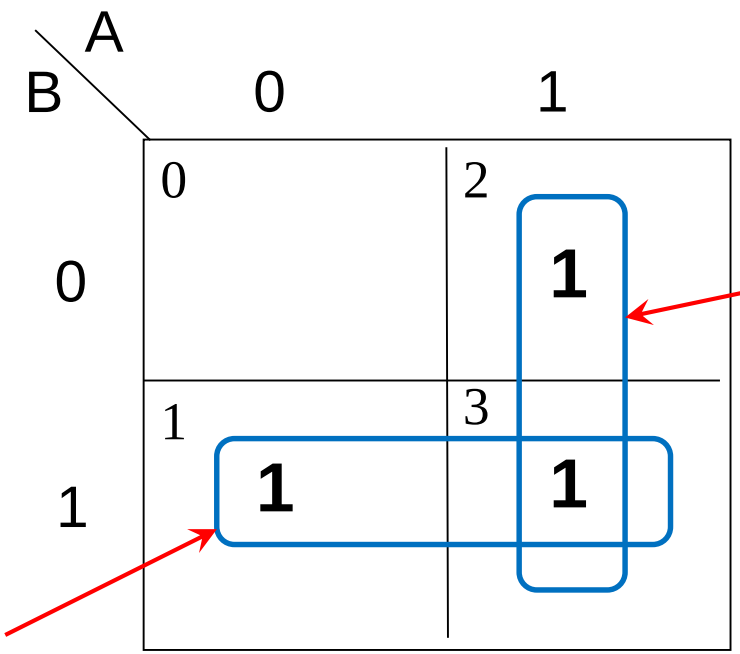
Minimisation using Karnaugh map

- ***Loop all neighbouring '1's***



Minimisation using Karnaugh map

- ***Loop all neighbouring '1's***
- ***Simplify the expressions within each loop***



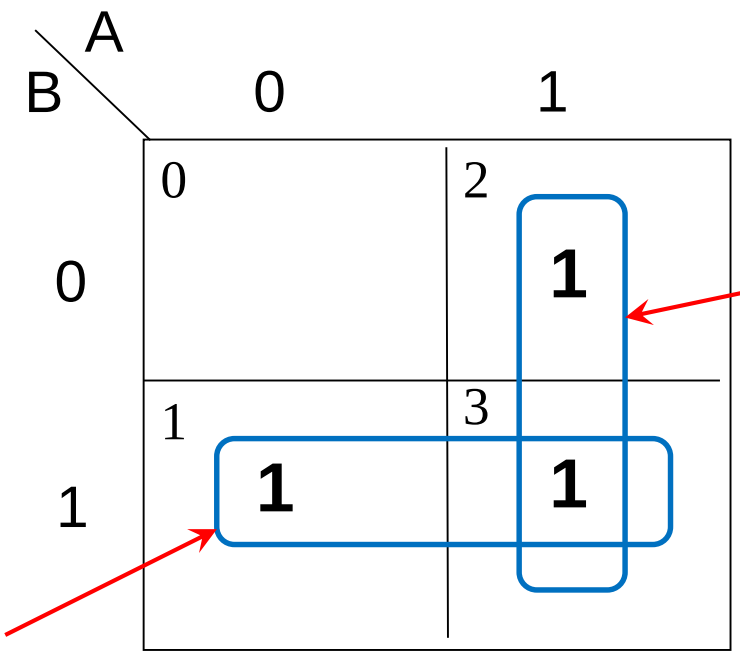
$A\overline{B} + AB = A$

Only one variable changes between the adjacent minterms, hence the rule **xy' + xy = x(y' + y) = x** can be applied.

$\overline{A}B + AB = B$

Minimisation using Karnaugh map

- ***Loop all neighbouring '1's***
- ***Simplify the expressions within each loop***
- ***OR simplified results from all loops***



$A\overline{B} + AB = A$

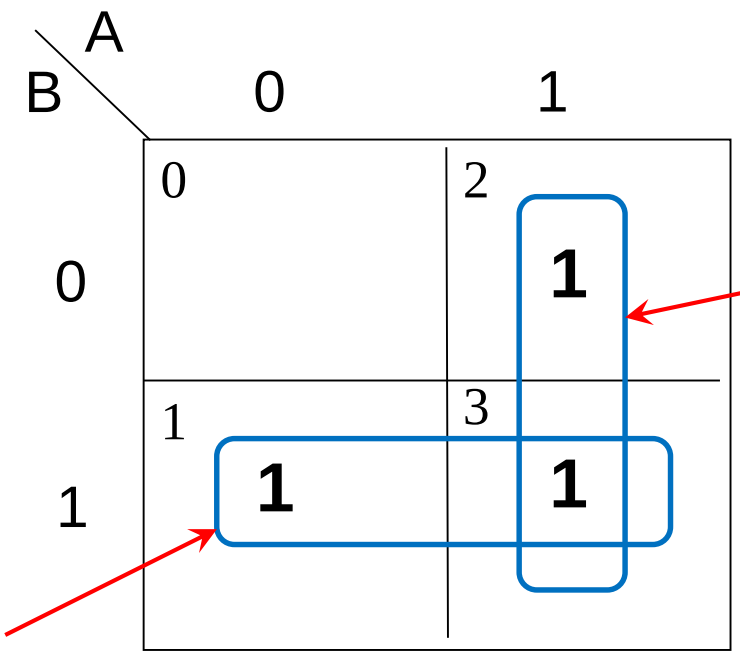
Only one variable changes between the adjacent minterms, hence the rule **xy' + xy = x(y' + y) = x** can be applied.

$\overline{A}B + AB = B$

Final expression: **F = A + B**

Minimisation using Karnaugh map

- **F(A,B) = ∑(1, 2, 3) = 01 + 10 + 11 – three terms** is simplified to **two-terms** function **F = A + B** .



$A\overline{B} + AB = A$

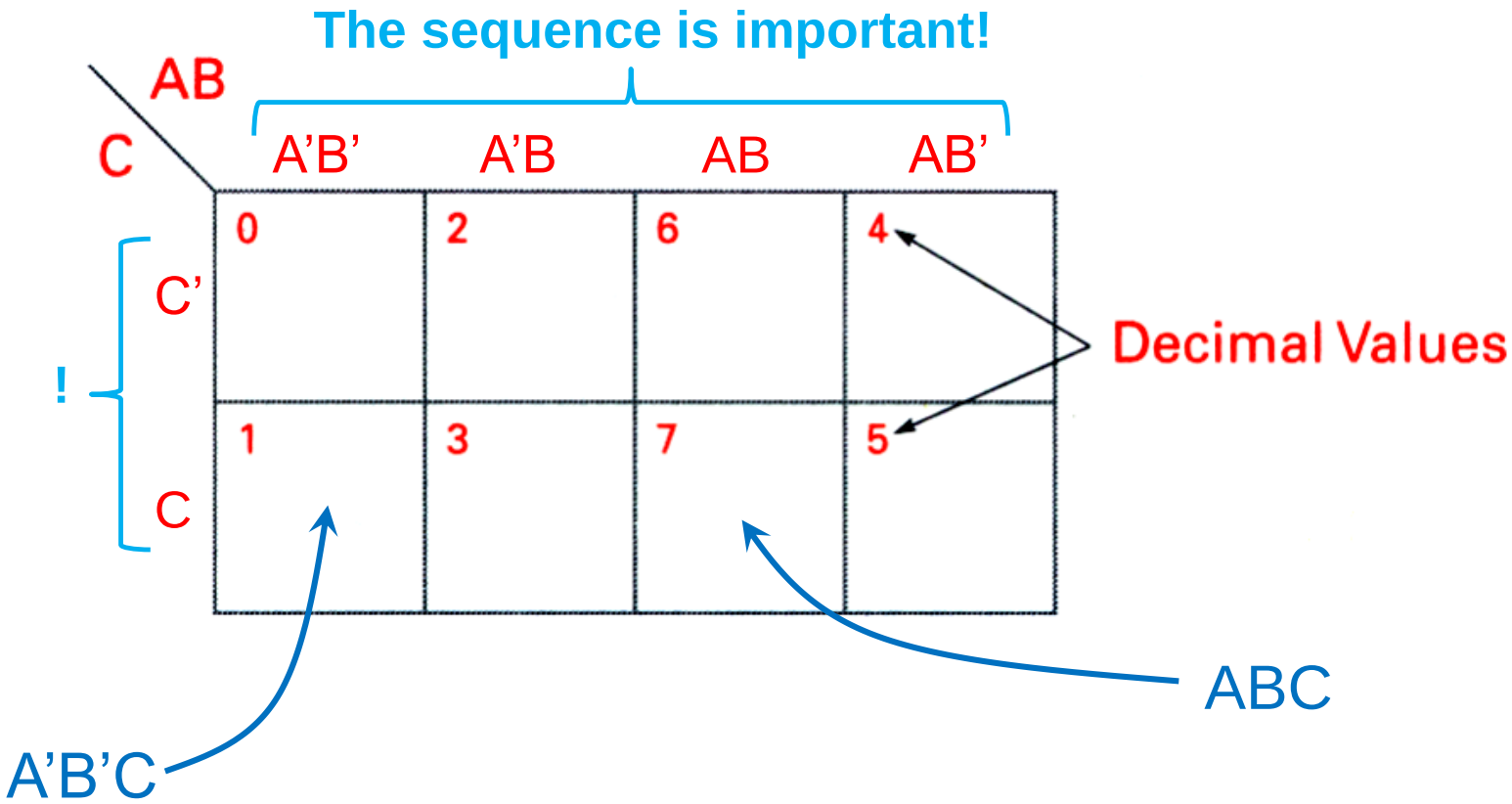
Only one variable changes between the adjacent minterms, hence the rule **xy' + xy = x(y' + y) = x** can be applied.

$\overline{A}B + AB = B$

Final expression: **F = A + B**

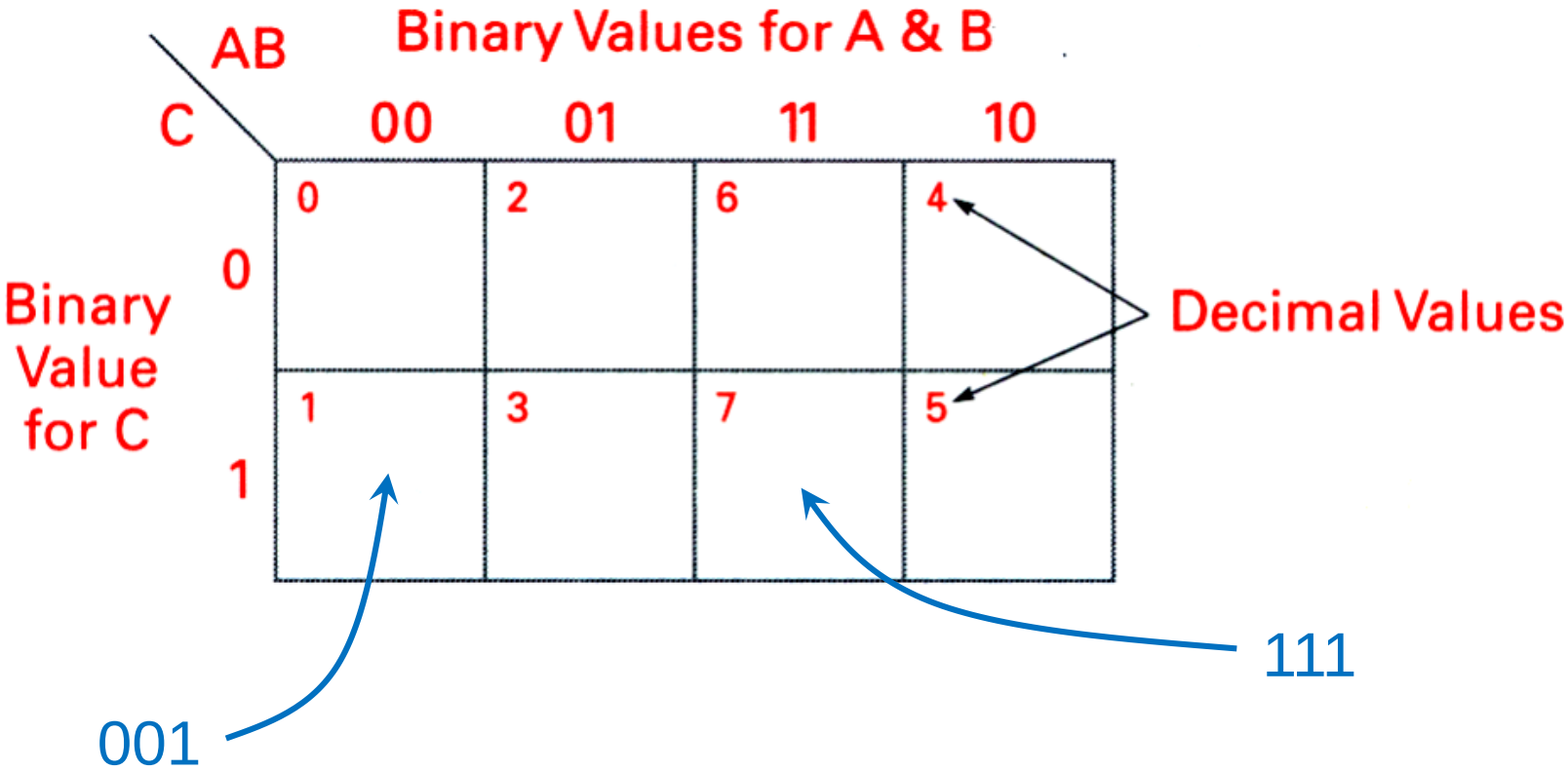
Three-variable Karnaugh map template

- ABC sequence of variables
- All eight possible combinations of three input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.



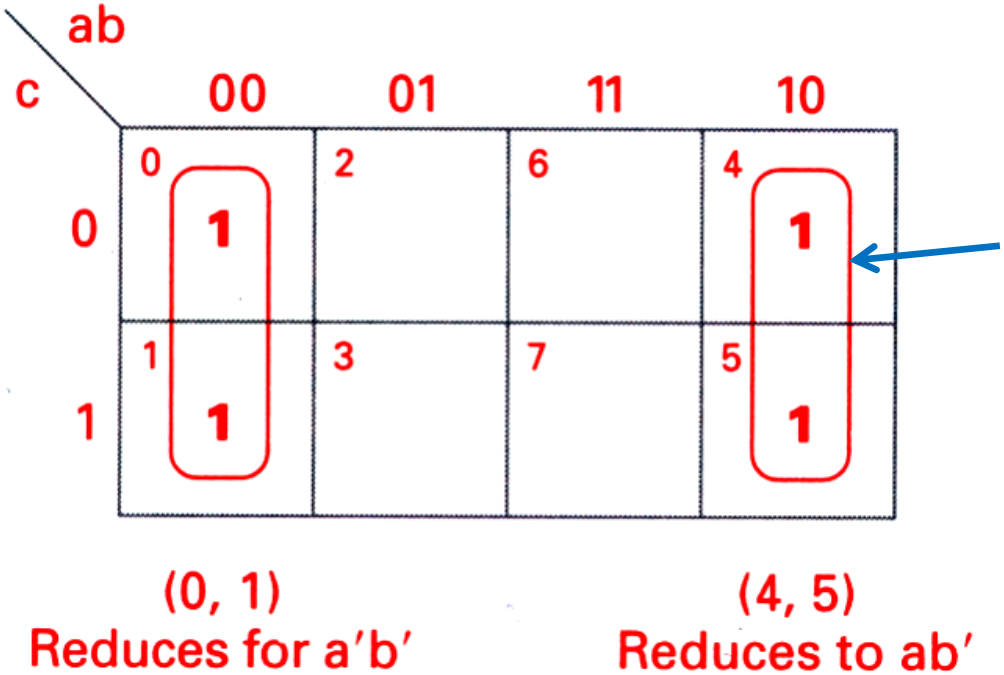
Three-variable Karnaugh map template

- ABC sequence of variables
- All eight possible combinations of three input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.



Three-variable Karnaugh map

$$Y = f(a, b, c) = \sum (0, 1, 4, 5)$$

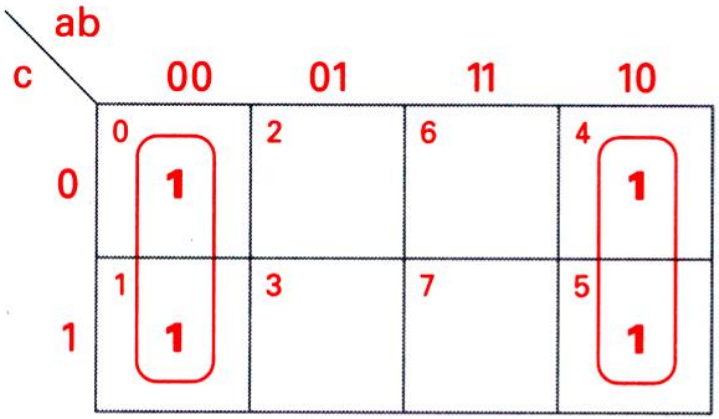


Only one variable (c) changes between the adjacent minterms in this loop, hence **ab'c' + ab'c = ab'(c' + c) = ab'**

$$Y = a'b' + ab'$$

Three-variable Karnaugh map

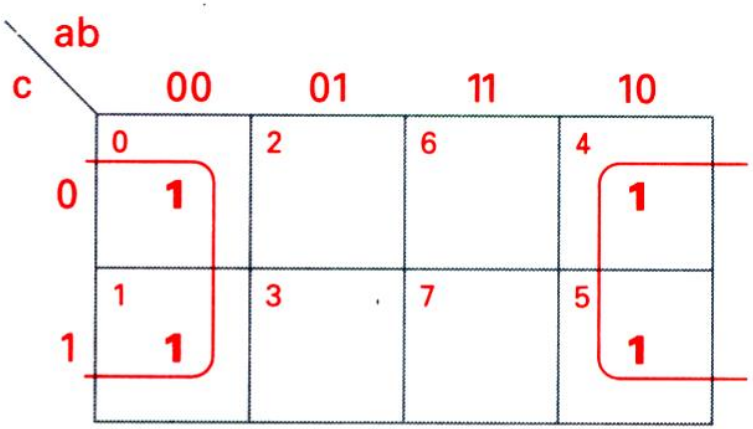
$$Y = f(a, b, c) = \sum (0, 1, 4, 5)$$



(0, 1)
Reduces for a′b′

(4, 5)
Reduces to ab′

Two groups of two minterms



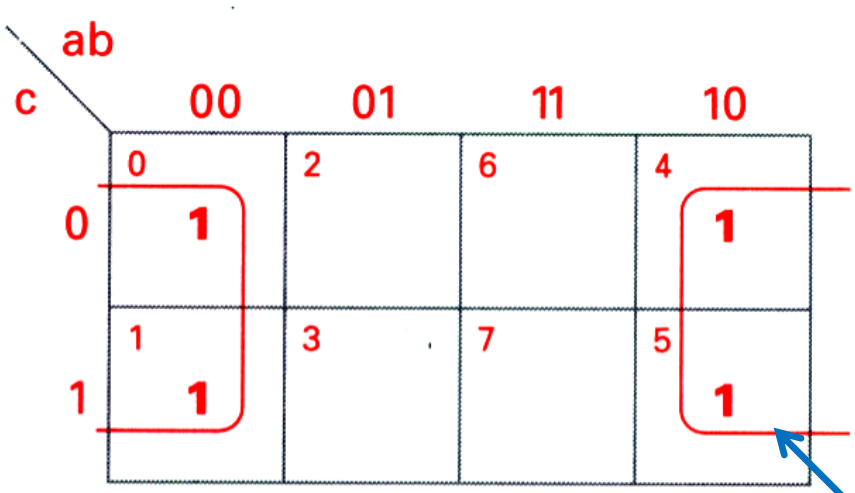
(0, 1, 4, 5)
Reduces to b′

One group of four minterms

More efficient way!

Three-variable Karnaugh map

$$Y = f(a, b, c) = \sum (0, 1, 4, 5)$$



(0, 1, 4, 5)
Reduces to b′

One group of four minterms

$$Y = b'$$

One variable changes in each direction (c↕ and ab↔) between the adjacent minterms in this loop, hence

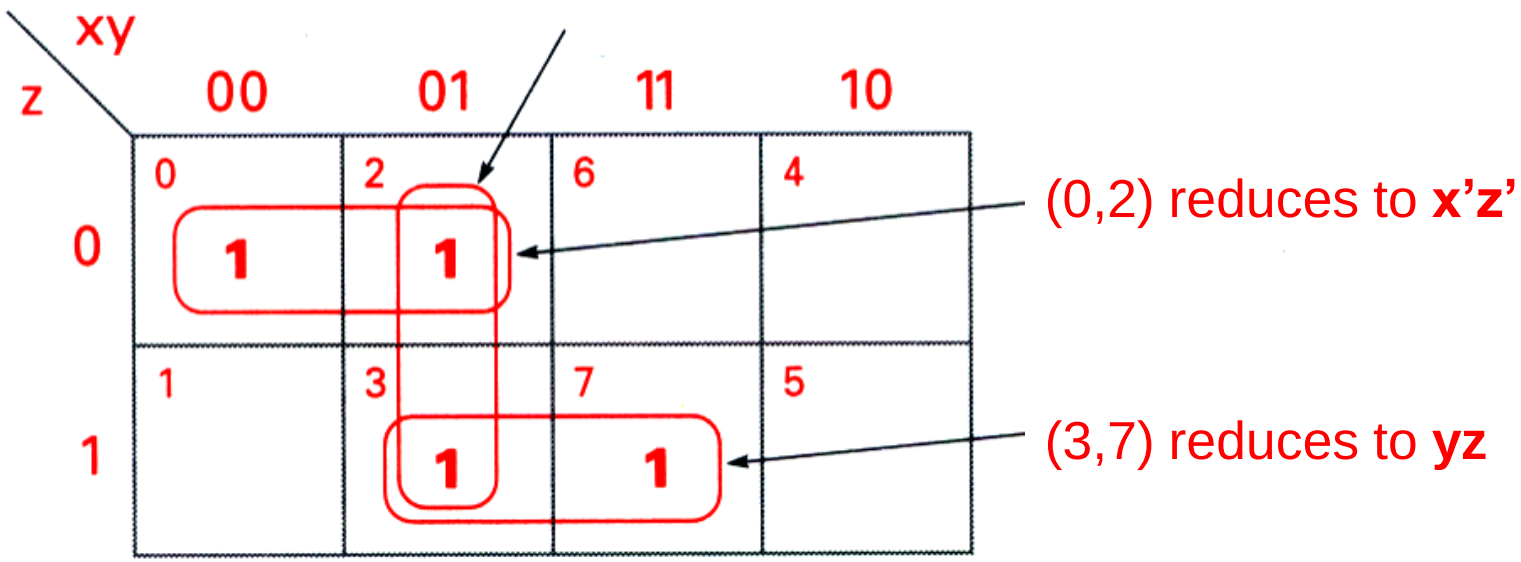
$$ab'c' + ab'c + a'b'c' + a'b'c =$$
$$= ab'(c' + c) + a'b'(c' + c) =$$
$$= b'(a + a') = b'$$

Only non-changing variables survive!

Three-variable Karnaugh map

$$G = f(x, y, z) = \sum (0, 2, 3, 7)$$

(2,3) disappears as not having unique elements



$$G = x'z' + yz$$

Exercise: Obtain Karnaugh map for:

$F = \Sigma\ (1, 2, 5, 6)$

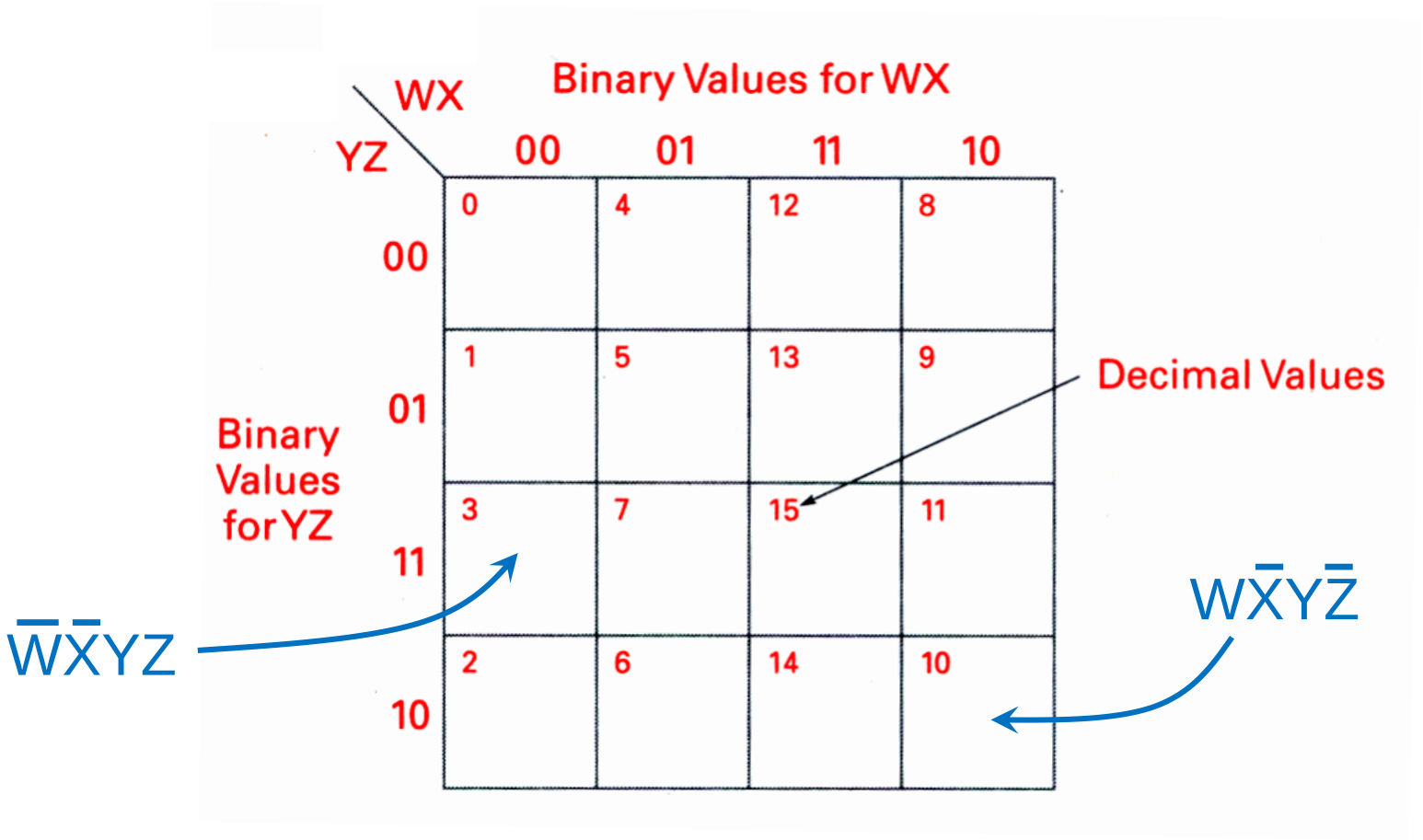
$F = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}C + AB\bar{C}$

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

And simplify by looping neighbouring cells……

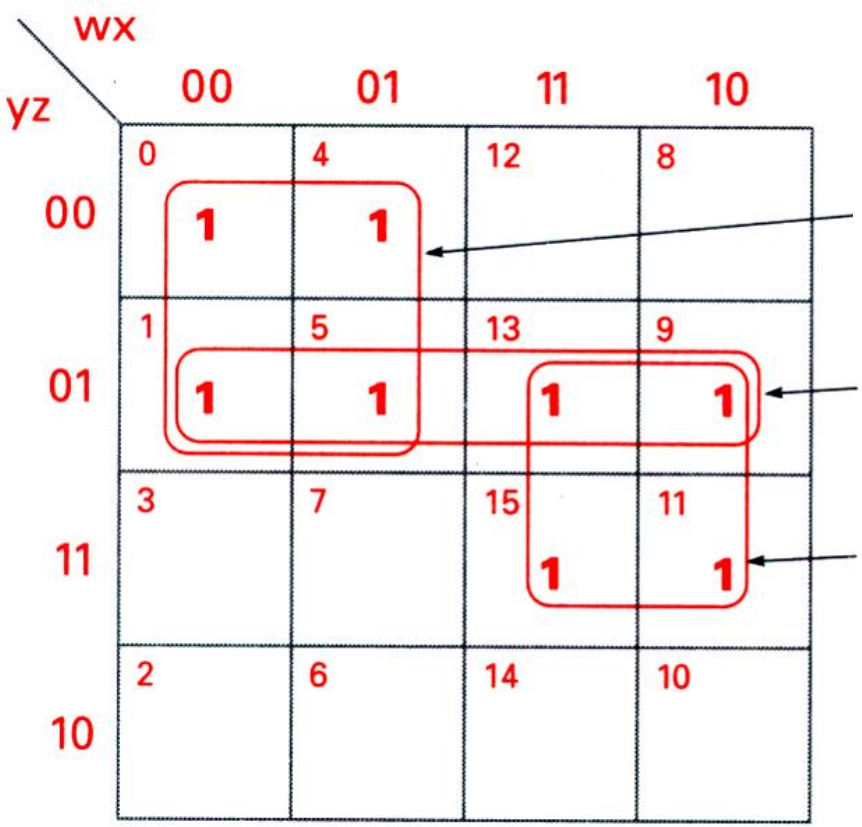
Four-variable Karnaugh map template

- WXYZ variable sequence.
- All 16 possible combinations of four input variables are represented.
- Designed in a way so only one variable changes between the adjacent cells.



Four-variable Karnaugh map

$$K = \sum (0, 1, 4, 5, 9, 11, 13, 15)$$



(0,1,4,5) reduces to **w'y'**

(1,5,13,9) disappears as having no unique elements

(9,11,13,15) reduces to **wz**

$$K = w'y' + wz$$

***Exercise*:**

A logic circuit has inputs A, B, C and D. The output of circuit is given by

$E = \Sigma (1, 3, 4, 5, 7, 10, 12, 13)$

Find the minimum sum of products form (1st canonical) for E.