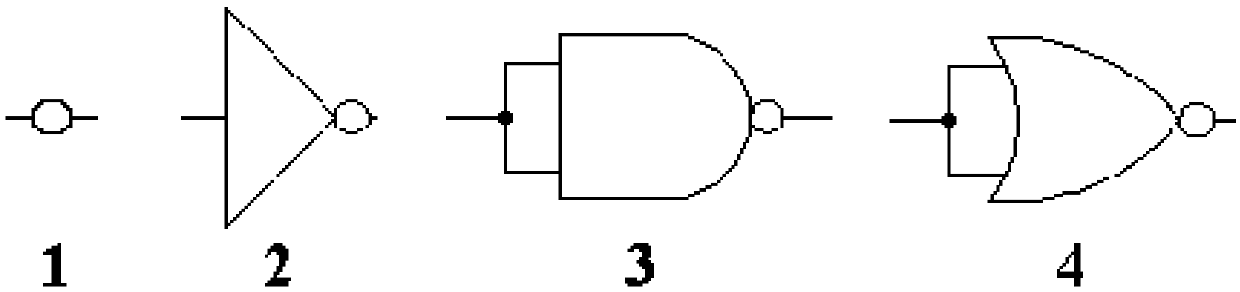
Several realization of NOT



1. This is how a NOT operation is often represented schematically.
2. Sometimes the NOT operation is shown explicitly as a gate.
3. Tying the inputs of a dual-input NAND gate will yield a NOT gate.
4. Tying the inputs of a dual-input NOR gate will similarly yield a NOT gate.

Truth Table Set-Ups for Karnaugh Maps

Each truth table consists of 2^n cells, where n is the number of inputs (logic variables).

Two Inputs (4 cells)

A\B	0	1
0	__	__
1	__	__

Three Inputs (8 cells)

AB\C	0	1
00	__	__
01	__	__
11	__	__
10	__	__

OR

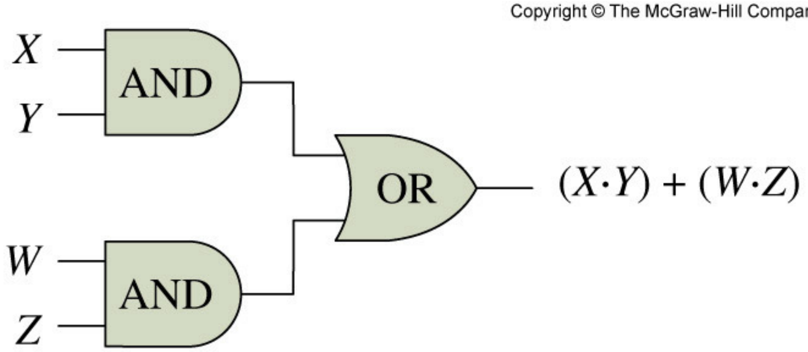
A\BC	00	01	11	10
0	__	__	__	__
1	__	__	__	__

The row and column assignments for two or more variables are arranged so that adjacent terms **change by only one bit.** That is, the possibilities of **AB** are considered as 00, 01, 11, and 10 in turn, rather than 00, 01, 10, and 11 as often is written sequence in ordinary truth tables. (In going from 01 to 10, both bits would be changed.) This type of construction facilitates identification of supercells that are used to simplify gate combinations for realizing the desired truth table.

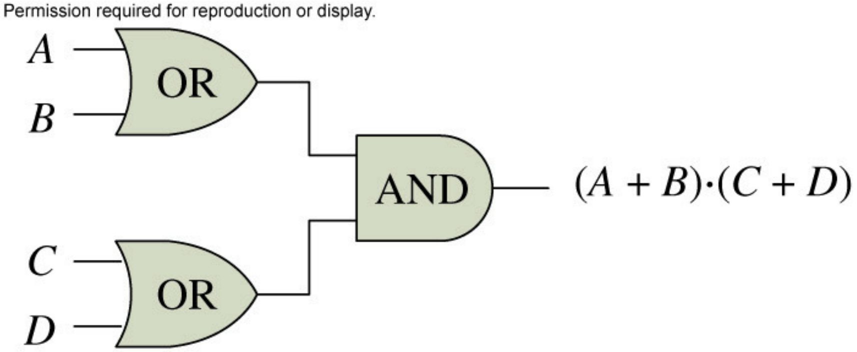
Truth Table Set-Ups for Karnaugh Maps

Four Inputs (16 cells)

AB\ CD	00	01	11	10
00	__	__	__	__
01	__	__	__	__
11	__	__	__	__
10	__	__	__	__



Sum-of-products expression $(X \cdot Y) + (W \cdot Z)$



Product-of-sums expression $(A + B) \cdot (C + D)$

Box Ones for Sum of Products

Consider the following truth table for three inputs:

A	B	C	O
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

The truth table expressed in a form suitable for Karnaugh mapping:

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

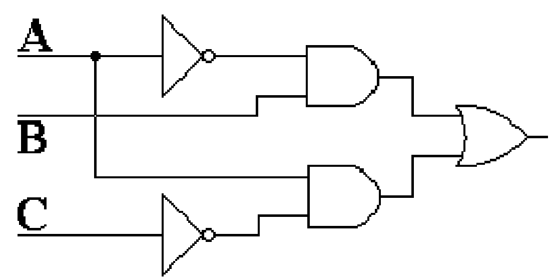
Boxing the ones:

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

The red cells form a 1x2 supercell represented by **A'·B**.

The green cells form a 1x2 wrapped supercell represented by **A·C'**.

The resulting Boolean expression for this truth table, a sum of products, is **A'·B + A·C'**.
This realization requires five gates as shown below:



- Supercell has to contain 2^n ones
- Make cells as big as possible
- Edge of Karnaugh is considered connected
- Finish all the ones

Boxing Zeroes for Sum of Products

Consider again the following truth table for three inputs:

A	B	C	O
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

The truth table expressed in a form suitable for Karnaugh mapping:

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

Boxing the zeroes:

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

The red cells form a 1x2 supercell represented by **A'·B'**.

The green cells form a 1x2 supercell represented by **A·C**.

The resulting Boolean expression for this truth table is the complement of the sum of these cells, **(A'·B' + A·C)'**.

This still looks like a sum of products, just NOTted! Several applications of De Morgan's theorems will yield the desired product of sums:

(A'·B' + A·C)'	**= (A'·B')' · (A·C)'**	by one application of De Morgan's theorem.
	= (A+B) · (A·C)'	by application of the other De Morgan's theorem.
	= (A+B) · (A'+C')	again by application of De Morgan's theorem.
		Thus results the product of sums!

Boxing Zeroes for Product of Sums

Consider again the following truth table for three inputs:

A	B	C	O
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

The truth table expressed in a form suitable for Karnaugh mapping:

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

Boxing the zeroes:

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

The red cells form a 1x2 supercell represented by **A+B**.

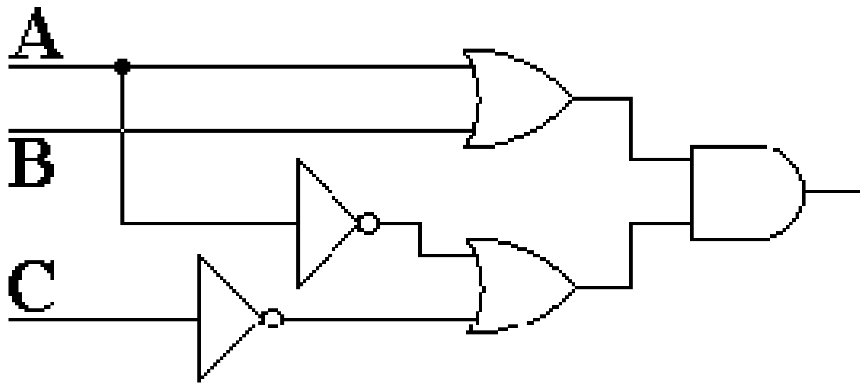
The green cells form a 1x2 supercell represented by **A'+C'**.

The resulting Boolean expression for this truth table is the product of these cells, **(A+B)•(A'+C')**

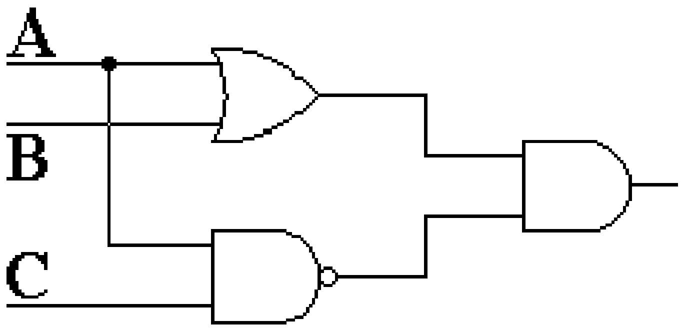
Boxing zeroes: cont.

$$(A' \cdot B' + A \cdot C)' = (A+B) \cdot (A \cdot C)' = (A+B) \cdot (A'+C')$$

This realization again requires five gates as shown below:



NOTE: The minimum realization requires only three gates.
This results from the next to last line in the application of De Morgan's theorems above.



Box one's and zero's: Summary

Sum of Products (X•Y+W• Z)

Box 1's

AND operation among inputs that is not changing

AND operation among inputs that is not changing

Between boxes OR operation

Box 0's

AND operation among inputs that is not changing

AND operation among inputs that is not changing

Between boxes OR operation

Result must be NOT'ed to get the final answer

Production of Sums (X+Y) •(W+Z)

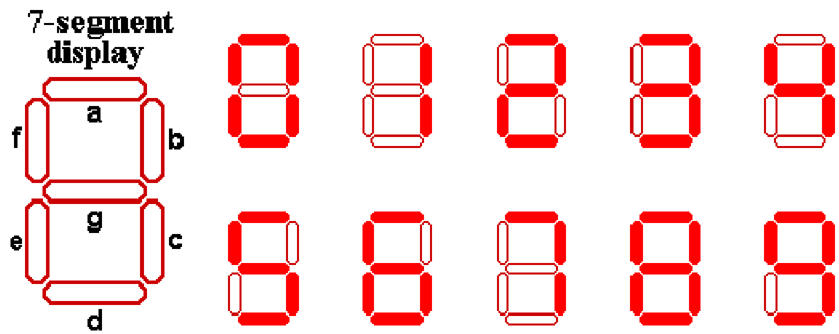
Box 0's

OR operation among NOT'ed inputs that is not changing

AND operation among NOT'ed inputs that is not changing

Between boxes AND operation

Example: 7 Segment Displays



BA	00	01	11	10
DC\				
00	1	0	1	1
01	0	1	1	1
11	x	x	x	x
10	1	1	x	x

Decimal Number	Inputs				Outputs						
	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

Example: 7 Segment Displays, Segment a

DC\ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	x	x	x	x
10	1	1	x	x

Red cells are represented by **B**; this is a 4x2 supercell.

DC\ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	x	x	x	x
10	1	1	x	x

Green cells are represented by **D; this is a 2x4 supercell.**

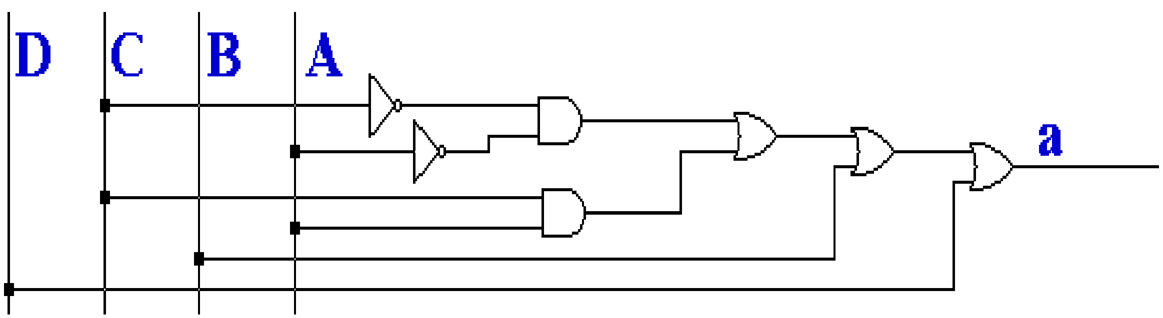
DC\ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	x	x	x	x
10	1	1	x	x

Blue cells are represented by **A·C;**

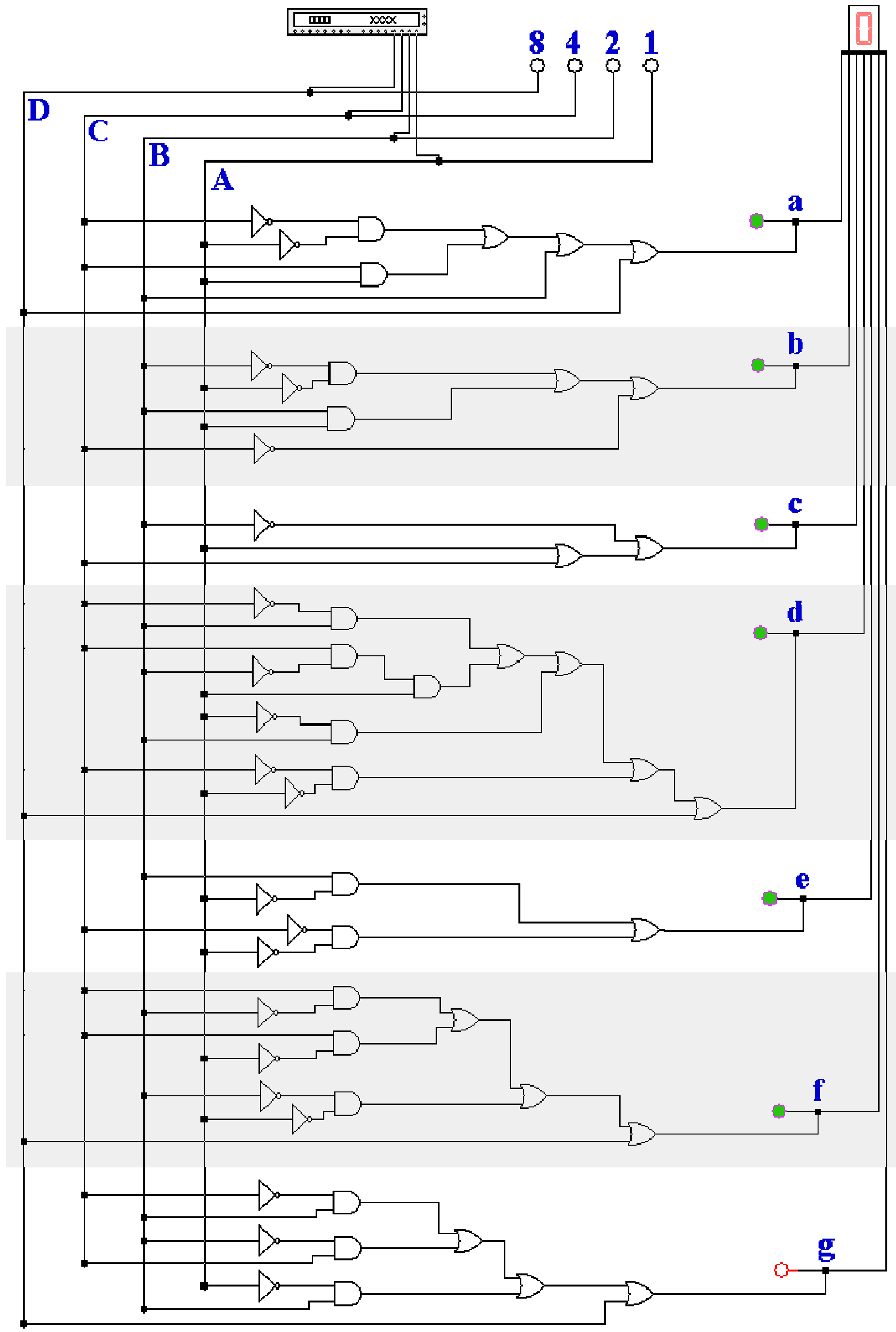
DC\ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	x	x	x	x
10	1	1	x	x

Cyan cells are represented by **A'·C';**

All cells are represented by **B + D + A·C + A'·C'**



7 segment decoder



Karnaugh Map for Simplifying Logic Function

$$f(A,B,C) = A \bullet B + \overline{A} \bullet C + B \bullet C$$

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

$$f(A,B,C) = \overline{A} \bullet C + A \bullet B$$