Fundamentals of Computer Systems

Boolean Logic

Stephen A. Edwards

Columbia University

Fall 2011

Boolean Logic

AN INVESTIGATION

OF

THE LAWS OF THOUGHT,

ON WHICH ARE FOUNDED

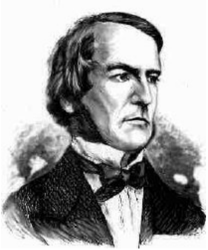
THE MATHEMATICAL THEORIES OF LOGIC
AND PROBABILITIES.

BY

GEORGE BOOLE, LL.D.

PROFESSOR OF MATHEMATICS IN QUEEN'S COLLEGE, CORK.

LONDON:
WALTON AND MABERLY,
UPPER GOWER-STREET, AND IVY-LANE, PATERNOSTER-ROW.
CAMBRIDGE: MACMILLAN AND CO.
1854.



George Boole
1815–1864

Boole's Intuition Behind Boolean Logic

Variables $x, y, \ldots$ represent classes of things

No imprecision: A thing either is or is not in a class

If x is "sheep" and y is "white things," xy are all white sheep,

$$xy = yx$$

and

$$xx = x.$$

If x is "men" and y is "women," $x + y$ is "both men and women,"

$$x + y = y + x$$

and

$$x + x = x.$$

If x is "men," y is "women," and z is "European," $z(x + y)$ is "European men and women" and

$$z(x + y) = zx + zy.$$

The Axioms of (Any) Boolean Algebra

A Boolean Algebra consists of

A set of values A
An “and” operator $\wedge$
An “or” operator $\vee$
A “not” operator $\neg$
A “false” value $0 \in A$
A “true” value $1 \in A$

The Axioms of (Any) Boolean Algebra

A Boolean Algebra consists of

A set of values A
An "and" operator $\wedge$
An "or" operator $\vee$
A "not" operator $\neg$
A "false" value $0 \in A$
A "true" value $1 \in A$

Axioms	
$a \vee b = b \vee a$	$a \wedge b = b \wedge a$
$a \vee (b \vee c) = (a \vee b) \vee c$	$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
$a \vee (a \wedge b) = a$	$a \wedge (a \vee b) = a$
$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
$a \vee \neg a = 1$	$a \wedge \neg a = 0$

The Axioms of (Any) Boolean Algebra

A Boolean Algebra consists of

A set of values A
An "and" operator $\wedge$
An "or" operator $\vee$
A "not" operator $\neg$
A "false" value $0 \in A$
A "true" value $1 \in A$

Axioms	
$a \vee b = b \vee a$	$a \wedge b = b \wedge a$
$a \vee (b \vee c) = (a \vee b) \vee c$	$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
$a \vee (a \wedge b) = a$	$a \wedge (a \vee b) = a$
$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
$a \vee \neg a = 1$	$a \wedge \neg a = 0$

We will use the first non-trivial Boolean Algebra: $A = \{0, 1\}$. This adds the law of excluded middle: if $a \neq 0$ then $a = 1$ and if $a \neq 1$ then $a = 0$.

Simplifying a Boolean Expression

"You are a New Yorker if you were born in New York or were not born in New York and lived here ten years."

$$x \vee \big((\neg x) \wedge y\big)$$

Axioms
$a \vee b = b \vee a$
$a \wedge b = b \wedge a$
$a \vee (b \vee c) = (a \vee b) \vee c$
$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
$a \vee (a \wedge b) = a$
$a \wedge (a \vee b) = a$
$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
$a \vee \neg a = 1$
$a \wedge \neg a = 0$

Lemma:

$$\begin{aligned} x \wedge 1 &= x \wedge (x \vee \neg x) \\ &= x \wedge (x \vee y) \text{ if } y = \neg x \\ &= x \end{aligned}$$

Simplifying a Boolean Expression

"You are a New Yorker if you were born in New York or were not born in New York and lived here ten years."

$$x \vee \big((\neg x) \wedge y\big)$$
$$= \big(x \vee (\neg x)\big) \wedge (x \vee y)$$

Axioms
$a \vee b = b \vee a$
$a \wedge b = b \wedge a$
$a \vee (b \vee c) = (a \vee b) \vee c$
$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
$a \vee (a \wedge b) = a$
$a \wedge (a \vee b) = a$
$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
$a \vee \neg a = 1$
$a \wedge \neg a = 0$

Lemma:

$$\begin{aligned} x \wedge 1 &= x \wedge (x \vee \neg x) \\ &= x \wedge (x \vee y) \text{ if } y = \neg x \\ &= x \end{aligned}$$

Simplifying a Boolean Expression

"You are a New Yorker if you were born in New York or were not born in New York and lived here ten years."

$$x \lor \big((\neg x) \land y\big)$$
$$= \big(x \lor (\neg x)\big) \land (x \lor y)$$
$$= 1 \land (x \lor y)$$

Axioms
$a \lor b = b \lor a$
$a \land b = b \land a$
$a \lor (b \lor c) = (a \lor b) \lor c$
$a \land (b \land c) = (a \land b) \land c$
$a \lor (a \land b) = a$
$a \land (a \lor b) = a$
$a \land (b \lor c) = (a \land b) \lor (a \land c)$
$a \lor (b \land c) = (a \lor b) \land (a \lor c)$
$a \lor \neg a = 1$
$a \land \neg a = 0$

Lemma:

$$x \land 1 = x \land (x \lor \neg x)$$
$$= x \land (x \lor y) \text{ if } y = \neg x$$
$$= x$$

Simplifying a Boolean Expression

"You are a New Yorker if you were born in New York or were not born in New York and lived here ten years."

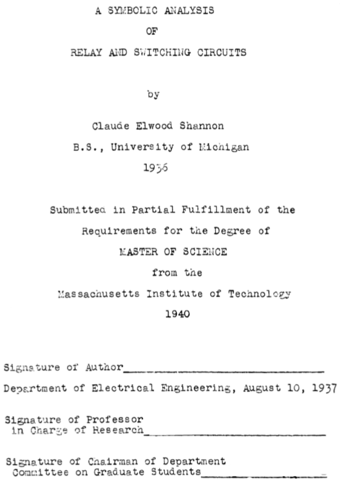
$$
\begin{aligned}
& x \vee \big((\neg x) \wedge y\big) \\
&= \big(x \vee (\neg x)\big) \wedge (x \vee y) \\
&= 1 \wedge (x \vee y) \\
&= x \vee y
\end{aligned}
$$

Axioms
$a \vee b = b \vee a$
$a \wedge b = b \wedge a$
$a \vee (b \vee c) = (a \vee b) \vee c$
$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
$a \vee (a \wedge b) = a$
$a \wedge (a \vee b) = a$
$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$
$a \vee \neg a = 1$
$a \wedge \neg a = 0$

Lemma:

$$
\begin{aligned}
x \wedge 1 &= x \wedge (x \vee \neg x) \\
&= x \wedge (x \vee y) \text{ if } y = \neg x \\
&= x
\end{aligned}
$$

What Does This Have To Do With Logic Circuits?



A SYMBOLIC ANALYSIS

OF

RELAY AND SWITCHING CIRCUITS

by

Claude Elwood Shannon

B.S., University of Michigan

1936

Submitted in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

from the

Massachusetts Institute of Technology

1940

Signature of Author________________________________

Department of Electrical Engineering, August 10, 1937

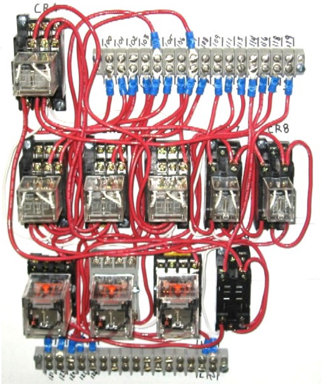
Signature of Professor
in Charge of Research______________________________

Signature of Chairman of Department
Committee on Graduate Students_____________________

Claude Shannon
1916–2001

Shannon’s MS Thesis

“We shall limit our treatment to circuits containing only relay contacts and switches, and therefore at any given time the circuit between any two terminals must be either open (infinite impedance) or closed (zero impedance).



Shannon’s MS Thesis

Fig. 1

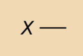
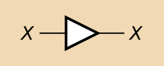
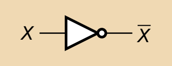
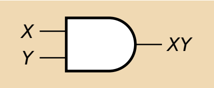
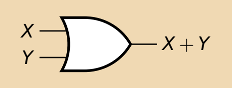
Fig. 2

Fig. 3

“It is evident that with the above definitions the following postulates hold.

$0 \cdot 0 = 0$	A closed circuit in parallel with a closed circuit is a closed circuit.
$1 + 1 = 1$	An open circuit in series with an open circuit is an open circuit.
$1 + 0 = 0 + 1 = 1$	An open circuit in series with a closed circuit in either order is an open circuit.
$0 \cdot 1 = 1 \cdot 0 = 0$	A closed circuit in parallel with an open circuit in either order is an closed circuit.
$0 + 0 = 0$	A closed circuit in series with a closed circuit is a closed circuit.
$1 \cdot 1 = 1$	An open circuit in parallel with an open circuit is an open circuit.
	At any give time either $X = 0$ or $X = 1$

Alternate Notations for Boolean Logic

Operator	**Math**	**Engineer**	**Schematic**
Copy	x	X	 or 
Complement	$\neg x$	$\overline{X}$	
AND	$x \wedge y$	XY or $X \cdot Y$	
OR	$x \vee y$	$X + Y$	

Definitions

Literal: a Boolean variable or its complement

E.g., $X \quad \overline{X} \quad Y \quad \overline{Y}$

Implicant: A product of literals

E.g., $X \quad XY \quad X\overline{Y}Z$

Minterm: An implicant with each variable once

E.g., $X\overline{Y}Z \quad XYZ \quad \overline{X}\,\overline{Y}Z$

Maxterm: A sum of literals with each variable once

E.g., $X+\overline{Y}+Z \quad X+Y+Z \quad \overline{X}+\overline{Y}+Z$

Be Careful with Bars

$$\overline{X}\,\overline{Y} \neq \overline{XY}$$

Let's check all the combinations of X and Y:

X	Y	$\overline{X}$	$\overline{Y}$	$\overline{X} \cdot \overline{Y}$	XY	$\overline{XY}$
0	0	1	1	1	0	1
0	1	1	0	0	0	1
1	0	0	1	0	0	1
1	1	0	0	0	1	0

Truth Tables

A *truth table* is a canonical representation of a Boolean function

X	Y	**Minterm**	**Maxterm**	$\overline{X}$	XY	$\overline{XY}$	$X+Y$	$\overline{X+Y}$
0	0	$\overline{X}\,\overline{Y}$	$X+Y$	1	0	1	0	1
0	1	$\overline{X}\,Y$	$X+\overline{Y}$	1	0	1	1	0
1	0	$X\,\overline{Y}$	$\overline{X}+Y$	0	0	1	1	0
1	1	$X\,Y$	$\overline{X}+\overline{Y}$	0	1	0	1	0

Each row has a unique minterm and maxterm

The minterm is 1 / maxterm is 0 for only its row

Sum-of-minterms and Product-of-maxterms

Two mechanical ways to translate a function's truth table into an expression:

X	Y	**Minterm**	**Maxterm**	F
0	0	$\overline{X}\,\overline{Y}$	$X+Y$	0
0	1	$\overline{X}\,Y$	$X+\overline{Y}$	1
1	0	$X\,\overline{Y}$	$\overline{X}+Y$	1
1	1	$X\,Y$	$\overline{X}+\overline{Y}$	0

The sum of the minterms where the function is 1:

$$F = \overline{X}\,Y + X\,\overline{Y}$$

The product of the maxterms where the function is 0:

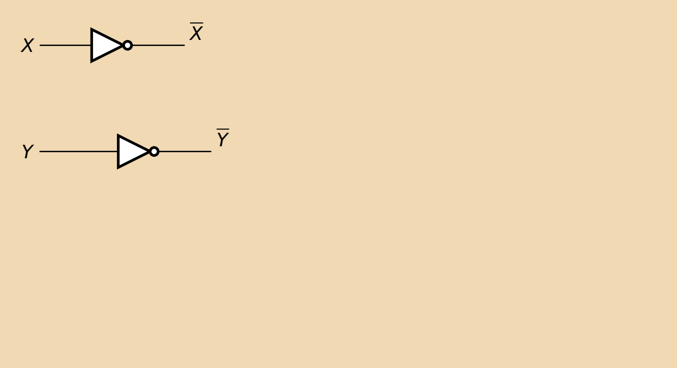
$$F = (X+Y)(\overline{X}+\overline{Y})$$

Expressions to Schematics

$$F = \overline{X}\,Y + X\,\overline{Y}$$

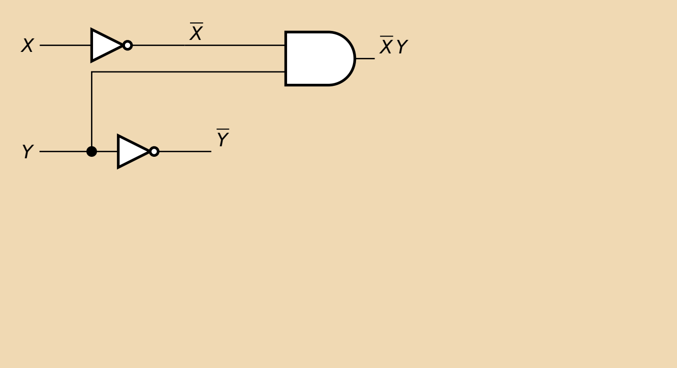
Expressions to Schematics

$$F = \overline{X}\,Y + X\,\overline{Y}$$



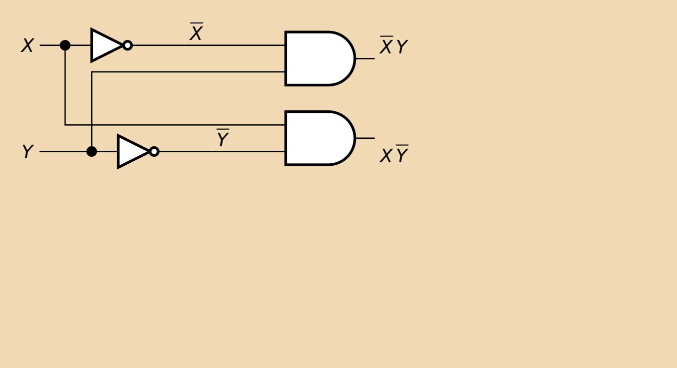
Expressions to Schematics

$$F = \overline{X}\,Y + X\,\overline{Y}$$



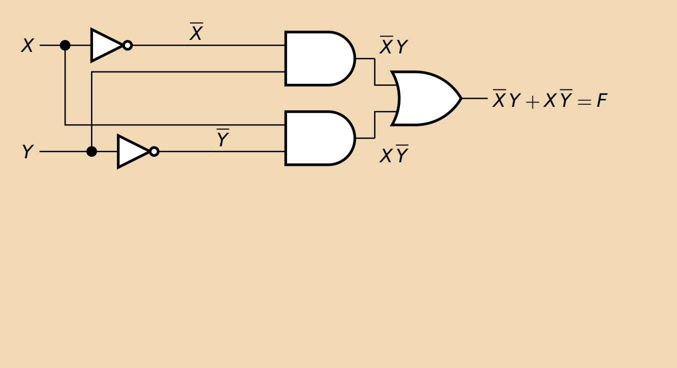
Expressions to Schematics

$$F = \overline{X}\,Y + X\,\overline{Y}$$



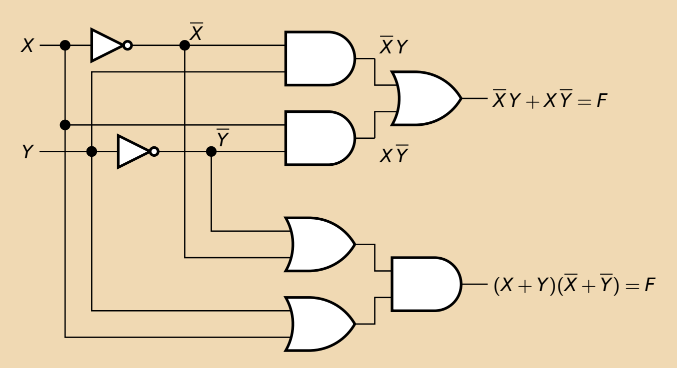
Expressions to Schematics

$$F = \overline{X}\,Y + X\,\overline{Y}$$



Expressions to Schematics

$$F = \overline{X}\,Y + X\,\overline{Y} = (X + Y)(\overline{X} + \overline{Y})$$



Minterms and Maxterms: Another Example

The minterm and maxterm representation of functions may look very different:

X	Y	**Minterm**	**Maxterm**	F
0	0	$\overline{X}\,\overline{Y}$	$X+Y$	0
0	1	$\overline{X}\,Y$	$X+\overline{Y}$	1
1	0	$X\,\overline{Y}$	$\overline{X}+Y$	1
1	1	$X\,Y$	$\overline{X}+\overline{Y}$	1

The sum of the minterms where the function is 1:

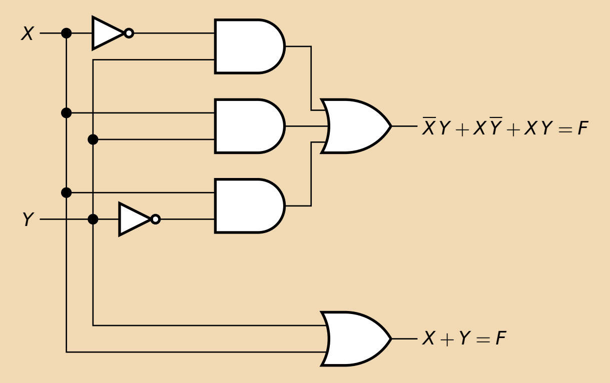
$$F = \overline{X}\,Y + X\,\overline{Y} + X\,Y$$

The product of the maxterms where the function is 0:

$$F = X + Y$$

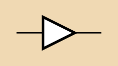
Expressions to Schematics 2

$$F = \overline{X}\,Y + X\,\overline{Y} + X\,Y = X + Y$$



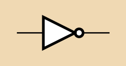
The Menagerie of Gates

Buffer



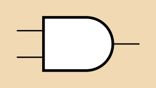
0	0
1	1

Inverter



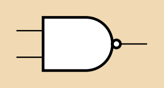
0	1
1	0

AND



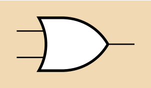
$\cdot$	0	1
0	0	0
1	0	1

NAND



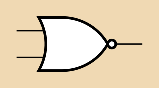
$\overline{\cdot}$	0	1
0	1	1
1	1	0

OR



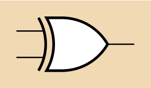
$+$	0	1
0	0	1
1	1	1

NOR



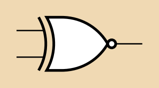
$\overline{+}$	0	1
0	1	0
1	0	0

XOR



$\oplus$	0	1
0	0	1
1	1	0

XNOR



$\overline{\oplus}$	0	1
0	1	0
1	0	1

De Morgan’s Theorem

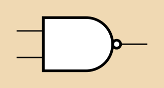
$$\neg(a \lor b) = (\neg a) \land (\neg b) \qquad \neg(a \land b) = (\neg a) \lor (\neg b)$$

Proof by Truth Table:

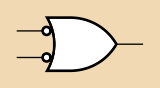
a	b	$a \lor b$	$(\neg a) \land (\neg b)$	$a \land b$	$(\neg a) \lor (\neg b)$
0	0	0	1	0	1
0	1	1	0	0	1
1	0	1	0	0	1
1	1	1	0	1	0

De Morgan's Theorem in Gates

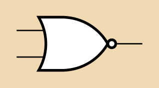
$$\overline{AB} = \overline{A} + \overline{B}$$



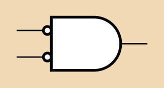
=



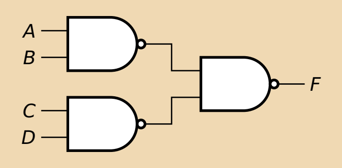
$$\overline{A + B} = \overline{A} \cdot \overline{B}$$



=



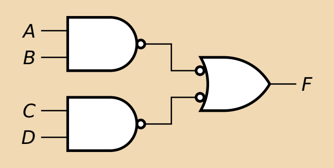
Bubble Pushing



Apply De Morgan’s Theorem:

Transform NAND into OR with inverted inputs

Bubble Pushing

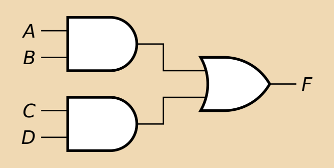


Apply De Morgan’s Theorem:

Transform NAND into OR with inverted inputs

Two bubbles on a wire cancel

Bubble Pushing

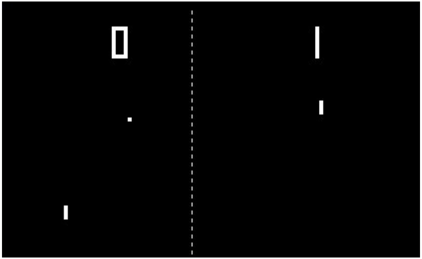


Apply De Morgan’s Theorem:

Transform NAND into OR with inverted inputs

Two bubbles on a wire cancel

PONG



PONG, Atari 1973

Built from TTL logic gates; no computer, no software

Launched the video arcade game revolution

Horizontal Ball Control in PONG

M	L	R	A	B
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	0	0	X	X
1	0	1	1	0
1	1	0	1	1
1	1	1	X	X

The ball moves either left or right.

Part of the control circuit has three inputs: M (“move”), L (“left”), and R (“right”).

It produces two outputs A and B.

Here, “X” means “I don’t care what the output is; I never expect this input combination to occur.”

Horizontal Ball Control in PONG

M	L	R	A	B
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	0	0
1	0	0	0	0
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

E.g., assume all the X's are 0's and use Minterms:

$$A = M\overline{L}R + ML\overline{R}$$

$$B = \overline{M}\,\overline{L}R + \overline{M}\,L\overline{R} + ML\overline{R}$$

3 inv + 4 AND3 + 1 OR2 + 1 OR3

Horizontal Ball Control in PONG

M	L	R	A	B
0	0	0	1	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1

Assume all the X's are 1's and use Maxterms:

$A = (M + L + \overline{R})(M + \overline{L} + R)$

$B = \overline{M} + L + \overline{R}$

3 inv + 3 OR3 + 1 AND2

Horizontal Ball Control in PONG

M	L	R	A	B
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	1	0

Choosing better values for the X's and being much more clever:

$A = M$

$B = \overline{MR}$

1 NAND2 (!)

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	L	R	A	B
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	0	0	X	X
1	0	1	1	0
1	1	0	1	1
1	1	1	X	X

The M’s are already arranged nicely

Karnaugh Maps

Basic trick: put "similar" variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	0	0	X	X
1	0	1	1	0
1	1	0	1	1
1	1	1	X	X

Let's rearrange the *L*'s by permuting two pairs of rows

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	0	0	X	X
1	0	1	1	0
1	1	0	1	1
1	1	1	X	X

Let’s rearrange the *L*’s by permuting two pairs of rows

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	0	0	X	X
1	0	1	1	0

Let’s rearrange the *L*’s by permuting two pairs of rows

1	1	0	1	1
1	1	1	X	X

Karnaugh Maps

Basic trick: put "similar" variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

Let's rearrange the *L*'s by permuting two pairs of rows

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

Let’s rearrange the *L*’s by permuting two pairs of rows

Karnaugh Maps

Basic trick: put "similar" variable values near each other so simple functions are obvious

M	L	R	A	B
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

Let's rearrange the L's by permuting two pairs of rows

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

Let’s rearrange the *L*’s by permuting two pairs of rows

Karnaugh Maps

Basic trick: put "similar" variable values near each other so simple functions are obvious

M	L	R	A	B
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

Let's rearrange the L's by permuting two pairs of rows

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	L	R	A	B
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

The R’s are really crazy; let’s use the second dimension

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	*L*	*R*	*A*	*B*
0	0	0	X	X
0	0	1	0	1
0	1	0	0	1
0	1	1	X	X
1	1	0	1	1
1	1	1	X	X
1	0	0	X	X
1	0	1	1	0

The *R*’s are really crazy; let’s use the second dimension

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	L	R	A	B
00	00	01	X0	X1
00	11	01	0X	1X
11	11	01	1X	1X
11	00	01	X1	X0

The R’s are really crazy; let’s use the second dimension

Karnaugh Maps

Basic trick: put “similar” variable values near each other so simple functions are obvious

M	L	R	A	B
00	00	01	X0	X1
00	11	01	0X	1X
11	11	01	1X	1X
11	00	01	X1	X0

MR

M

Maurice Karnaugh’s Maps

The Map Method for Synthesis of Combinational Logic Circuits

M. KARNAUGH
NONMEMBER AIEE

THE SEARCH for simple abstract techniques to be applied to the design of switching systems is still, despite some recent advances, in its early stages. The problem in this area which has been attacked most energetically is that of the synthesis of efficient combinational that is, nonsequential, logic circuits.

be convenient to describe other methods in terms of Boolean algebra. Whenever the term “algebra” is used in this paper, it will refer to Boolean algebra, where addition corresponds to the logical connective “or,” while multiplication corresponds to “and.”

The minimizing chart,[2] developed at

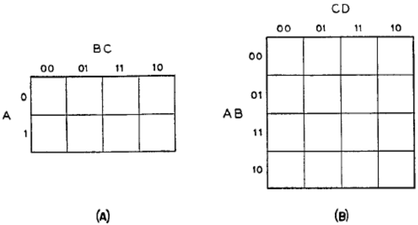
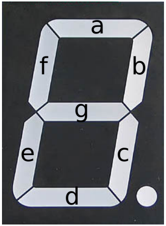


Fig. 2. Graphical representations of the input conditions for three and for four variables

Transactions of the AIEE, 1953

The Seven-Segment Decoder Example



W	*X*	*Y*	*Z*	*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	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X
1	1	1	1	0	0	0	0	0	0	0

Karnaugh Map for Seg. a

W	*X*	*Y*	*Z*	*a*
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0

		Z	*Z*		
	1	0	1	1	
X	0	1	1	1	
X	X	X	0	X	*W*
	1	1	X	X	*W*
			Y	*Y*	

The Karnaugh Map
Sum-of-Products Challenge

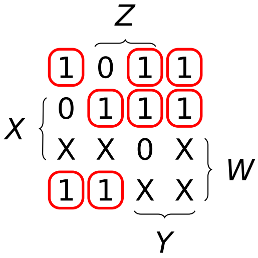
Cover all the 1's and none of the 0's using as few literals (gate inputs) as possible.

Few, large rectangles are good.

Covering X's is optional.

Karnaugh Map for Seg. a

W	X	Y	Z	a
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0



The minterm solution: cover each 1 with a single implicant.

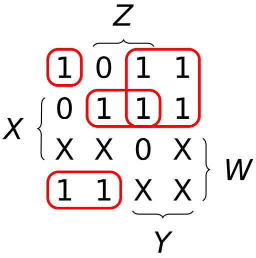
$$a = \overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z} + \overline{W}\,\overline{X}\,Y\,Z + \overline{W}\,\overline{X}\,Y\,\overline{Z} + \overline{W}\,X\,\overline{Y}\,Z + \overline{W}\,X\,Y\,Z + \overline{W}\,X\,Y\,\overline{Z} + W\,\overline{X}\,\overline{Y}\,Z + W\,\overline{X}\,\overline{Y}\,Z$$

$8 \times 4 = 32$ literals

4 inv + 8 AND4 + 1 OR8

Karnaugh Map for Seg. a

W	X	Y	Z	a
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0



Merging implicants helps

Recall the distributive law:
$AB + AC = A(B + C)$

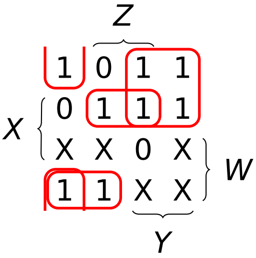
$$a = \overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z} + \overline{W}Y + \overline{W}XZ + W\overline{X}\,\overline{Y}$$

$4 + 2 + 3 + 3 = 12$ literals

4 inv + 1 AND4 + 2 AND3 + 1 AND2 + 1 OR4

Karnaugh Map for Seg. a

W	X	Y	Z	a
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0



Missed one: Remember this is actually a torus.

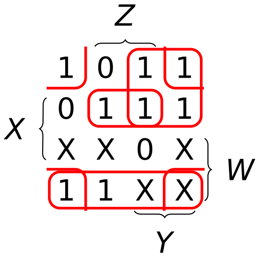
$$a = \overline{X}\,\overline{Y}\,\overline{Z} + \overline{W}Y + \overline{W}XZ + W\overline{X}\,\overline{Y}$$

$3 + 2 + 3 + 3 = 11$ literals

4 inv + 3 AND3 + 1 AND2 + 1 OR4

Karnaugh Map for Seg. a

W	X	Y	Z	a
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0



Taking don't-cares into account, we can enlarge two implicants:

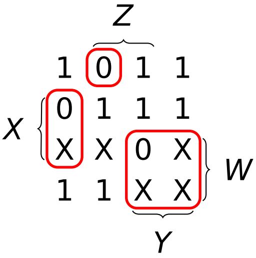
$$a = \overline{X}\,\overline{Z} + \overline{W}Y + \overline{W}XZ + W\overline{X}$$

$2 + 2 + 3 + 2 = 9$ literals

3 inv + 1 AND3 + 3 AND2 + 1 OR4

Karnaugh Map for Seg. a

W	X	Y	Z	a
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0



Can also compute the complement of the function and invert the result.

Covering the 0's instead of the 1's:

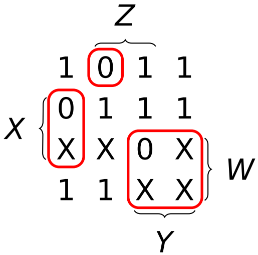
$$\overline{a} = \overline{W}\,\overline{X}\,\overline{Y}Z + X\overline{Y}\,\overline{Z} + WY$$

$4 + 3 + 2 = 9$ literals

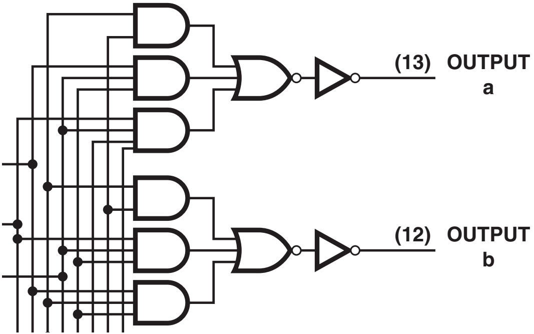
5 inv + 1 AND4 + 1 AND3 + 1 AND2 + 1 OR3

Karnaugh Map for Seg. a

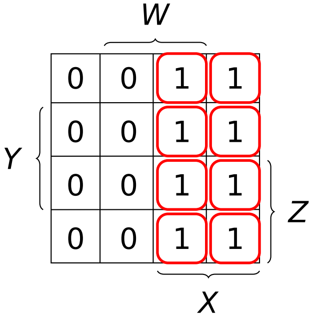
W	X	Y	Z	a
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	0



To display the score, PONG used a TTL chip with this solution in it:



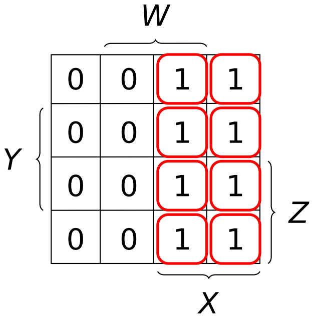
Boolean Laws and Karnaugh Maps



$$
\begin{aligned}
&W X \overline{Y} \overline{Z} + \overline{W} X \overline{Y} \overline{Z} + \\
&W X Y \overline{Z} + \overline{W} X Y \overline{Z} + \\
&W X Y Z + \overline{W} X Y Z + \\
&W X \overline{Y} Z + \overline{W} X \overline{Y} Z
\end{aligned}
$$

Factor out the *W*’s

Boolean Laws and Karnaugh Maps



$$(W+\overline{W})\,X\,\overline{Y}\,\overline{Z}+$$
$$(W+\overline{W})\,X\,Y\,\overline{Z}+$$
$$(W+\overline{W})\,X\,Y\,Z+$$
$$(W+\overline{W})\,X\,\overline{Y}\,Z$$

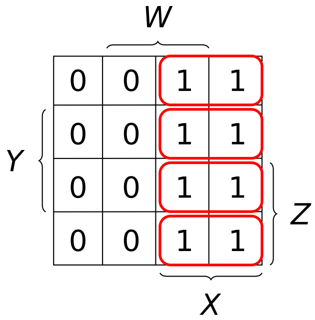
Use the identities

$$W+\overline{W}=1$$

and

$$1X=X.$$

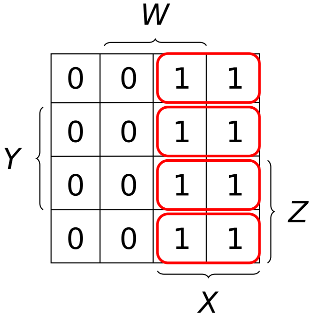
Boolean Laws and Karnaugh Maps



$X\,\overline{Y}\,\overline{Z}+$
$X\,Y\,\overline{Z}+$
$X\,Y\,Z+$
$X\,\overline{Y}\,Z$

Factor out the Y's

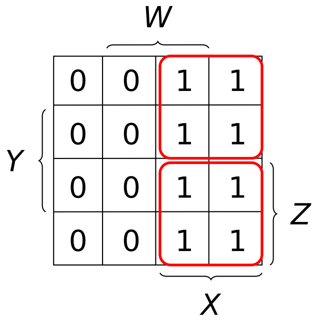
Boolean Laws and Karnaugh Maps



$$(\overline{Y} + Y)X\overline{Z} +$$
$$(\overline{Y} + Y)XZ$$

Apply the identities again

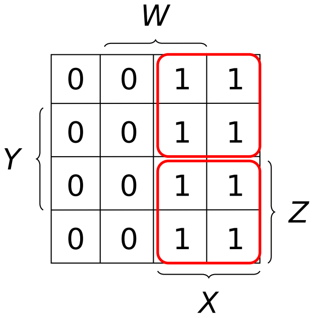
Boolean Laws and Karnaugh Maps



$X\overline{Z}+$
XZ

Factor out Z

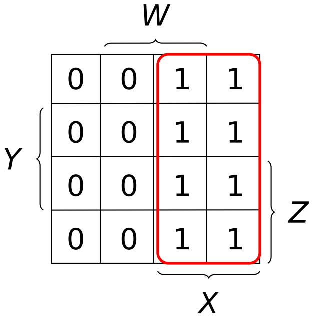
Boolean Laws and Karnaugh Maps



$$X(\overline{Z}+Z)$$

Simplify

Boolean Laws and Karnaugh Maps



X

Done