COMP1071 - Digital Electronics

Boolean Algebra

Ioannis Ivrissimtzis

email: ioannis.ivrissimtzis@durham.ac.uk

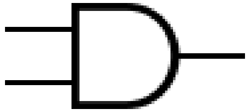
Room: MCS 2023

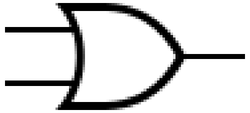
Slide acknowledgements: Eleni Akrida and Farshad Arvin

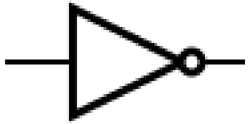
Overview of today's lecture

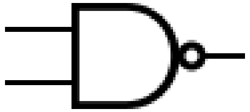
- Functionally complete sets
- Intro to Combinational Logic and Circuits
- Sum of Products vs Product of Sums
- Boolean Algebra

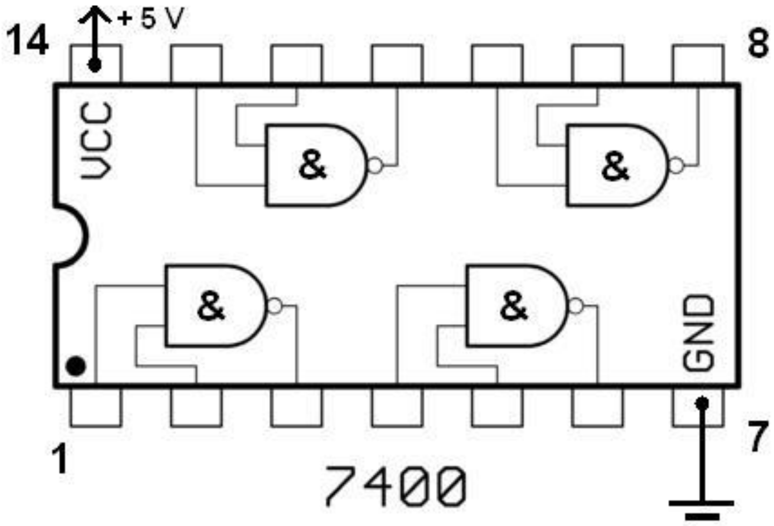
Gates we have seen

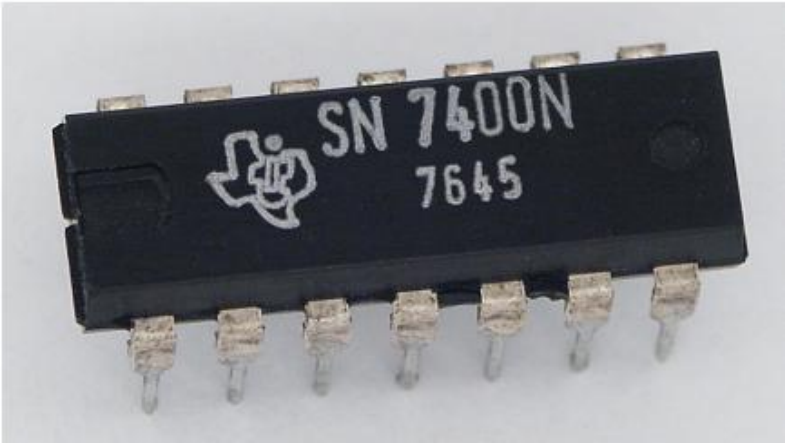
AND gate: 

OR gate: 

NOT gate: 

NAND gate: 





Durham University

More Boolean operations

A	B	Y
0	0	**?**
0	1	**?**
1	0	**?**
1	1	**?**

How many possible Boolean operations are there on 2 inputs?

There are 2^{2^k} possible Boolean operations on k inputs – *i.e.* 16 on 2 inputs.

The trivial operations are: 0, 1, A, B

We have seen $\overline{A}$, $\overline{B}$, A·B, A+B

What else is there?

Adding rule:

0 + 0 = 0

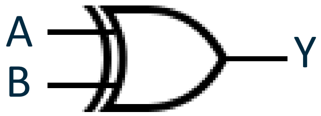
0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 (with Carry)

Durham University

Truth tables: Exclusive OR (XOR)

- **XOR gate:** 

Algebraic expression: $Y = A \oplus B$

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

Linear truth table

		B	
	XOR	**0**	**1**
A	**0**	**0**	**1**
	1	**1**	**0**

Rectangular/Coordinate table

Exclusive OR (XOR)

We can construct XOR using AND, OR and NOT:

$$A \oplus B = (A + B) \bullet (\overline{A \bullet B})$$

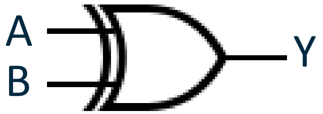
Check:

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

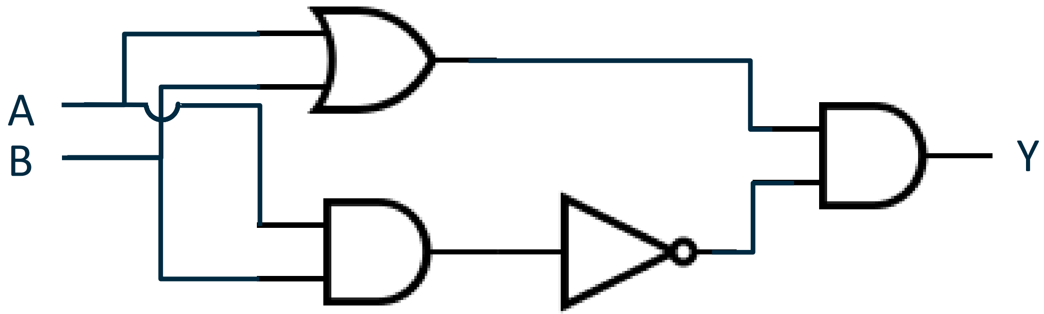
Durham University

Exclusive OR (XOR)

$$A \oplus B = (A + B) \bullet (\overline{A \bullet B})$$



is the same as



Durham University

Overview of today's lecture

- Functionally complete sets
- Intro to Combinational Logic and Circuits
- Sum of Products vs Product of Sums
- Boolean Algebra

Functionally Complete Sets

In logic, a **functionally complete set of Boolean operators** is one which can be used to *express all possible truth tables* by combining members of the set into a Boolean expression.

Functionally Complete Sets

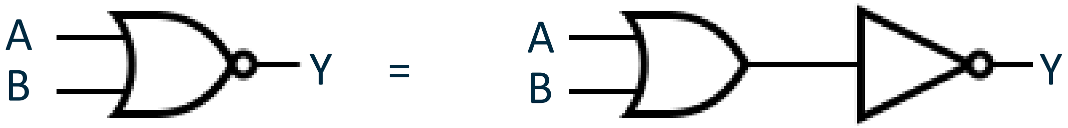
Every Boolean expression can be turned into a conjunctive normal form, see PoS!

Any logic circuit can be constructed with just these three operators

- **AND, OR,** and **NOT**
- They form a **functionally complete set.**

Charles Sanders Peirce (1880) showed that **NOR gates alone** form a functionally complete set.

NOR: inverse of OR, $Y = \overline{A + B}$



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

NOR gates

AND:

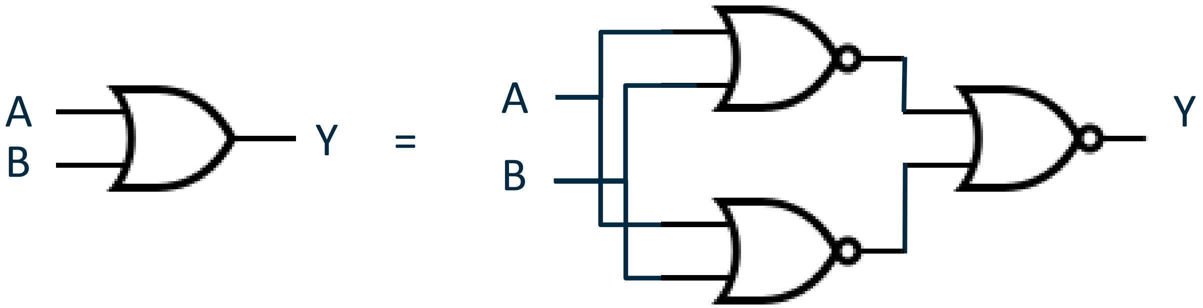
$$A \cdot B = \overline{\overline{(A + A)} + \overline{(B + B)}}$$

OR:

$$A + B = \overline{\overline{(A + B)} + \overline{(A + B)}}$$

NOT:

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



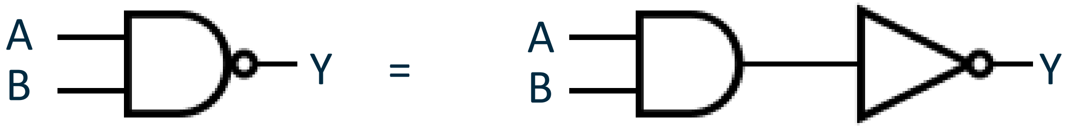
Functionally Complete Sets

Any logic circuit can be constructed with just these three operators

- **AND, OR,** and **NOT**
- They form a **functionally complete set.**

Henry M. Sheffer (1913) showed that the **NAND gates alone** form a functionally complete set.

NAND: inverse of AND, $Y = \overline{A \cdot B}$



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

NAND gates

AND:

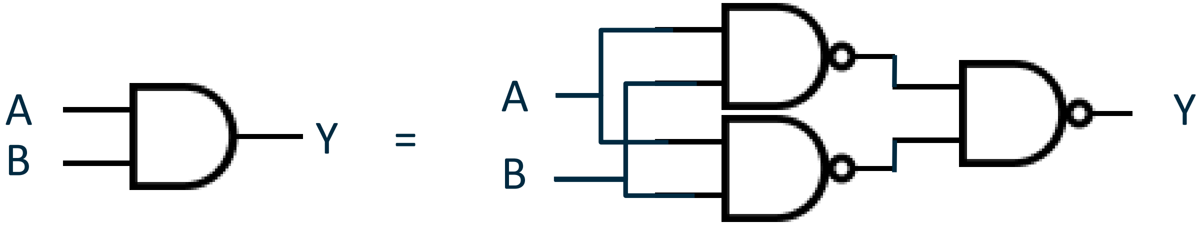
$$A \cdot B = \overline{(\overline{A \cdot B}) \cdot (\overline{A \cdot B})}$$

OR:

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

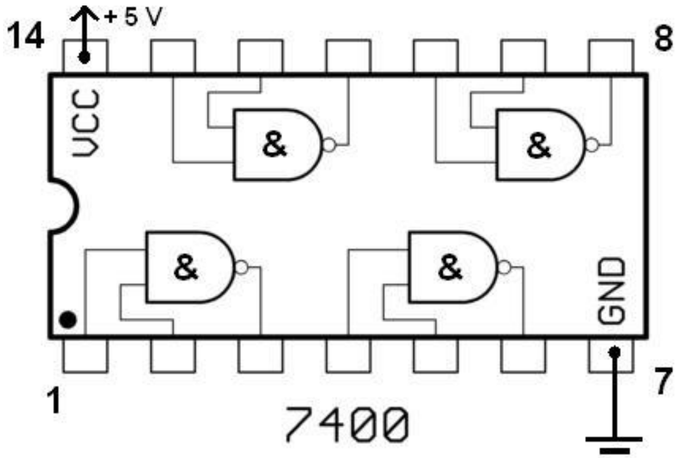
NOT:

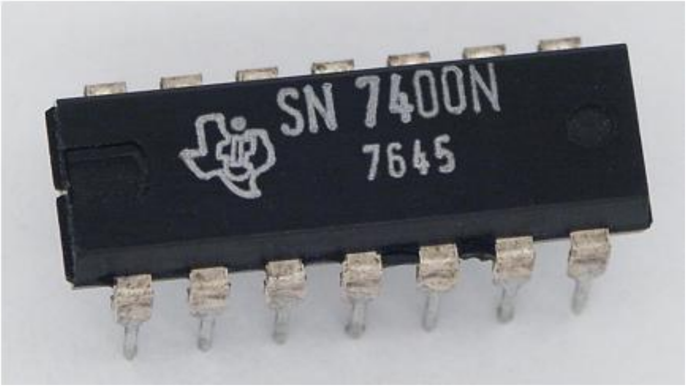
$$\overline{A} = \overline{A \cdot A}$$



NAND chips

NAND gates are easier to make (use less silicon for same performance) than NOR gates. So often used as universal gates. E.g. the 7400:





Overview of today's lecture

- Functionally complete sets
- Intro to Combinational Logic and Circuits
- Sum of Products vs Product of Sums
- Boolean Algebra

Digital design principles

Digital design is all about **managing the complexity** of huge numbers of interacting elements. Some principles help humans do this:

Abstraction: Hiding details when they aren't important.

Discipline: Restricting design choices to make things easier to model, design and combine. E.g. the logic families and the digital abstraction.

The three –y's:

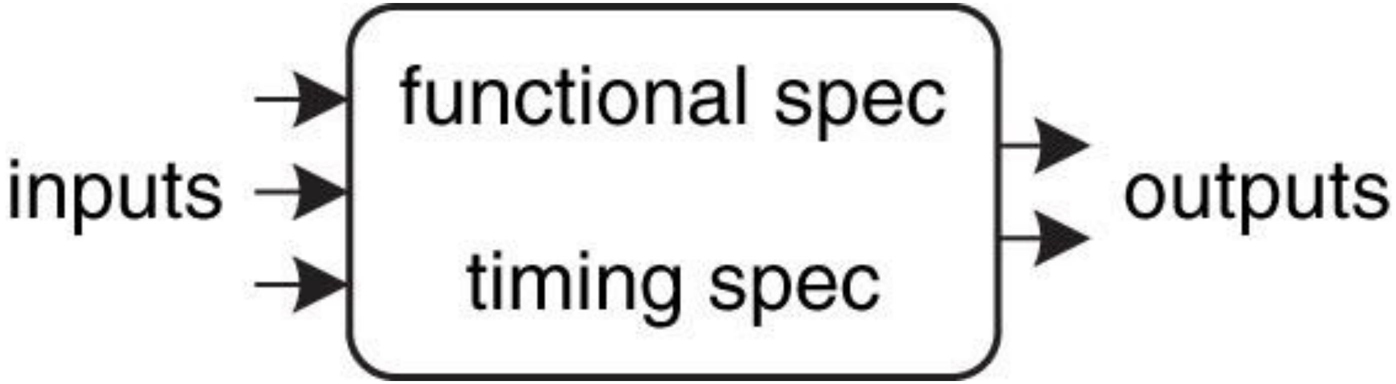
- **Hierarchy**: dividing a system into modules and submodules
- **Modularity**: well-defined functions and interfaces for modules
- **Regularity**: encouraging uniformity so modules can be swapped or reused.

Circuits

Network that processes discrete-valued variables.

A **circuit** has:

- one or more discrete valued input terminals
- one or more discrete valued output terminals
- a specification of the relationship between inputs and outputs
- a specification of the delay between inputs changing and outputs responding

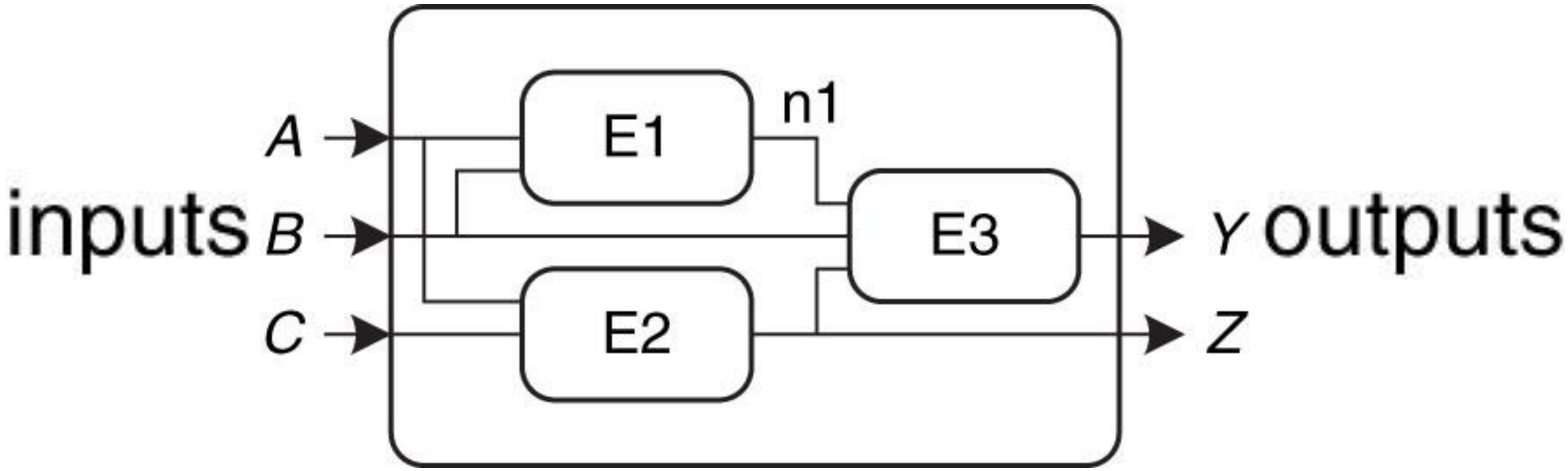


Durham University

Circuits

The circuit is made up of **elements** and **nodes:**

- An **element** is itself a circuit with inputs, outputs and specs.
- A **node** is a wire joining elements, whose voltage conveys a discrete valued variable.



Durham University

Combinational logic

We wish to design **very large** circuits to perform functions for us.

We will restrict what we allow, for now, firstly to **combinational logic*** and circuits.

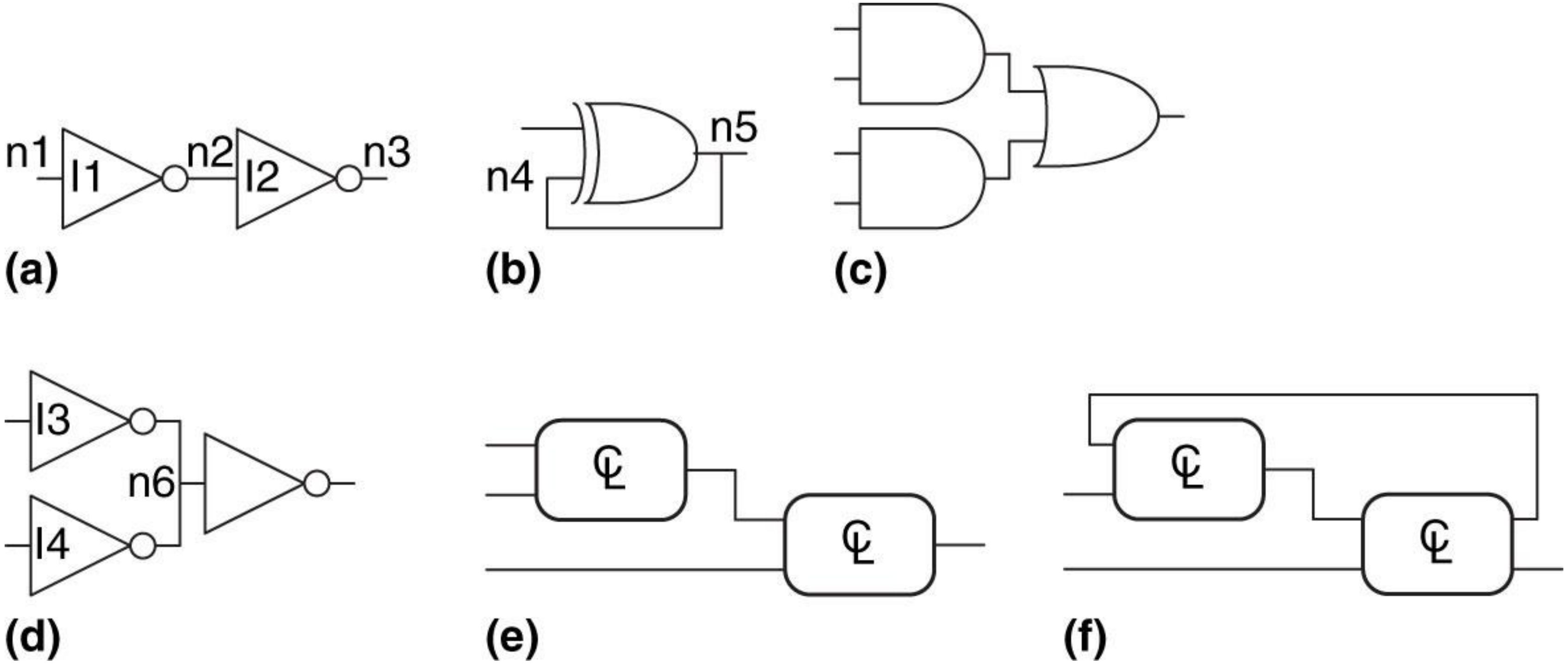
Combinational logic rules:

- **Individual gates** are combinational circuits.
- Every circuit **element** must be a combinational circuit.
- Every node is either an **input** to the circuit or connecting **to exactly one output** of a circuit element
- The circuit has **no cyclic paths** – every path through the circuit visits any node at most once.

*And later **sequential logic.**

Combinational logic

Which of these are combinational circuits and why?



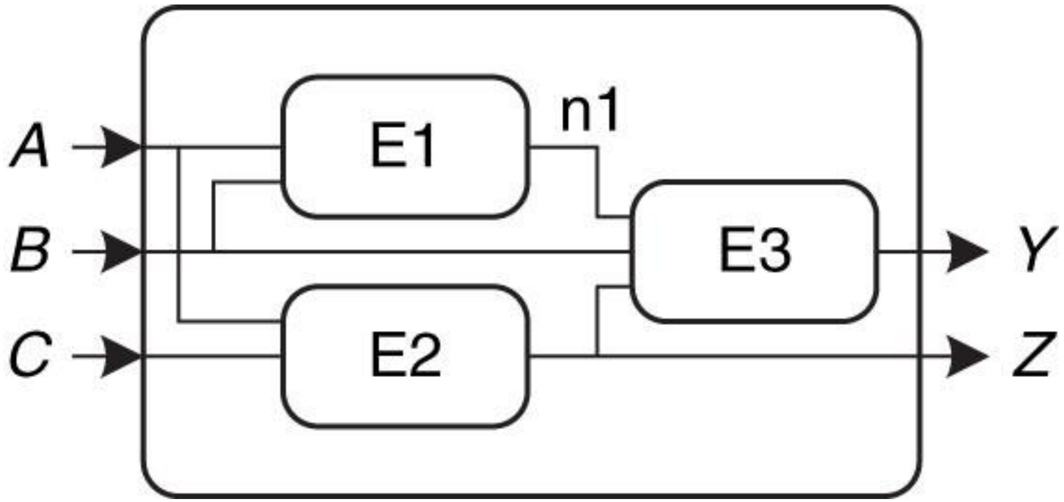
Overview of today's lecture

- Functionally complete sets
- Intro to Combinational Logic and Circuits
- Sum of Products vs Product of Sums
- Boolean Algebra

Durham University

Boolean Algebra

- The algebra of 0/1 variables.
- Used for specifying the function of a combinational circuit
- Used to analyse and simplify the circuits required to give a specified truth table.



Boolean Algebra

- **Variables** are represented by letters, e.g. $A, B, C \ldots$

  The **complement** or **inverse** of a variable is written with a bar, e.g. $\overline{\boldsymbol{A}}$.

- A variable or its complement is called a **literal**, e.g. $A, \overline{A}, B$ or $\overline{B}$.

- The **AND** of several literals is called a **product**, e.g. $A\overline{B}C$ or $A\overline{C}$,

  Products may be written $A \cdot B \cdot C$, ABC, $A \cap B \cap C$ or $A \wedge B \wedge C$.

  A **minterm** is a product in which **all** the inputs to a function appear once each (either in its complemented or uncomplemented form).

- The **OR** of several literals is called a **sum** or **implicant**, e.g. $A + B + C$ or $A + C$

  Sums may be written $A + B + C$, $A \cup B \cup C$ or $A \vee B \vee C$.

  A **maxterm** is a sum in which **all** the inputs to a function appear once each (either in its complemented or uncomplemented form).

Durham University

Truth table to Boolean eqn.

X	Y	Z	F(X,Y,Z)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Truth table to Boolean eqn.

"Sum of products (SoP)" form

Every Boolean expression can be written as minterms **ORed together**:

$$(A \cdot B \cdot C) + (A \cdot \overline{B} \cdot \overline{C}) + (\overline{A} \cdot B \cdot C)$$

"Product of sums (PoS)" form

Also every Boolean expression can be written as maxterms **ANDed together**:

$$(\overline{A} + \overline{B} + \overline{C}) \cdot (\overline{A} + B + C) \cdot (A + \overline{B} + \overline{C})$$

Truth table to SoP

X	Y	Z	F(X,Y,Z)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

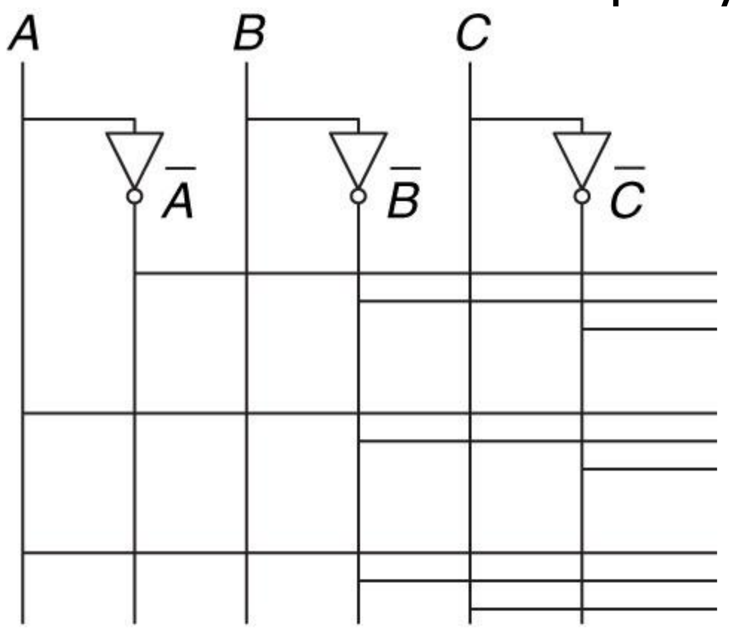
Each row can be represented by a minterm that is true:

OR together the **1 values** of the function, to give SOP form:

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

Example of circuit design:

$$Y = \bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$$



This layout can be used for any sum-of-products expression.

Truth table to PoS

X	Y	Z	F(X,Y,Z)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Each row can be represented by a maxterm that is false:

$$X + Y + Z$$

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

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

AND together the **0 values** of the function, to give POS for F:

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

Compare to $F(X,Y,Z) = \overline{X} \cdot \overline{Y} \cdot \overline{Z} + \overline{X} \cdot Y \cdot Z + X \cdot \overline{Y} \cdot Z + X \cdot Y \cdot \overline{Z}$

Overview of today's lecture

- Functionally complete sets
- Intro to Combinational Logic and Circuits
- Sum of Products vs Product of Sums
- Boolean Algebra

Durham University

Boolean Algebra

Two equivalent expression for the same logical formula:

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

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

Which is simpler?

Is there another equivalent expression that is simpler than either?

We will use **Boolean algebra** and **Karnaugh maps** to produce the simplest equivalent expression that can then be turned into circuitry.

Durham University

Axioms of Boolean Algebra

	Axiom		Dual axiom	Name
A1	$B = 0 \; if \; B \neq 1$	A1’	$B = 1 \; if \; B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2’	$\overline{1} = 0$	NOT
A3	$0 \cdot 0 \; = \; 0$	A3’	$1 + 1 \; = \; 1$	AND/OR
A4	$1 \cdot 1 \; = \; 1$	A4’	$0 + 0 \; = \; 0$	AND/OR
A5	$0 \cdot 1 \; = \; 1 \cdot 0 \; = \; 0$	A5’	$1 + 0 = 0 + 1 = 1$	AND/OR

Axioms cannot be proven – they are defined or assumed.
Each axiom has a dual obtained by interchanging AND and OR, and 0 and 1.

Durham University

Theorems of one variable

	Theorem		Dual theorem	Name
T1	$B \cdot 1 = B$	T1’	$B + 0 = B$	Identity
T2	$B \cdot 0 = 0$	T2’	$B + 1 = 1$	Null element
T3	$B \cdot B = B$	T3’	$B + B = B$	Idempotency
T4	$\overline{\overline{B}} = B$			Involution
T5	$B \cdot \overline{B} = 0$	T5’	$B + \overline{B} = 1$	Complements

Theorems can be proved by applying the axioms and checking cases

Theorems of several variables

	Theorem	Dual	Name
T6	B•C = C•B	B+C = C+B	Commutativity
T7	(B•C)•D= B•(C•D)	(B+C)+D = B+(C+D)	Associativity
T8	B•(C+D) = B•C+B•D	(B+C)•(B+D) = B+(C•D)	Distributivity
T9	B•(B+C) = B	B+(B•C) = B	Covering
T10	$B \cdot C + B \cdot \overline{C} = B$	$(B+C) \cdot (B+\overline{C}) = B$	Combining
T11	$B \cdot C + \overline{B} \cdot D + C \cdot D = B \cdot C + \overline{B} \cdot D$		Consensus
T11’		$(B+C) \cdot (\overline{B}+D) \cdot (C+D) = (B+C) \cdot (\overline{B}+D)$	
T12	$\overline{B_0 \cdot B_1 \cdot B_2 \ldots} = \overline{B_0} + \overline{B_1} + \overline{B_2} \ldots$		De Morgan’s
T12’		$\overline{B_0 + B_1 + B_2 \ldots} = \overline{B_0} \cdot \overline{B_1} \cdot \overline{B_2} \ldots$	

DeMorgan's Theorem

Proof of two variable case:

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

Proof:

A	B	$A \cdot B$	$\overline{A \cdot B}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	**0**	**1**	**1**	**1**	**1**
0	1	**0**	**1**	**1**	**0**	**1**
1	0	**0**	**1**	**0**	**1**	**1**
1	1	**1**	**0**	**0**	**0**	**0**

Durham University

Theorems of several variables

	Theorem	Dual	Name
	Key principle for simplification: use T10 and T11 to remove variables or terms; use others to rearrange so that T10 or T11 can be applied		
T10	$B \cdot C + B \cdot \overline{C} = B$	$(B+C) \cdot (B+\overline{C}) = B$	Combining
T11	$B \cdot C + \overline{B} \cdot D + C \cdot D = B \cdot C + \overline{B} \cdot D$		Consensus
T11'		$(B+C) \cdot (\overline{B}+D) \cdot (C+D) = (B+C) \cdot (\overline{B}+D)$	

General form of T10: for any implicant (i.e., product or sum) P and variable A, $PA + P\overline{A} = P$

Example

Draw the circuit corresponding to this truth table.

1. Sum of products form
2. Simplification using Boolean Algebra

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

Sum of products: $Y = \overline{A}\,B\,\overline{C} + \overline{A}\,B\,C + A\,B\,\overline{C}$

Now minimise this equation: $Y = \overline{A}\,B\,(\overline{C} + C) + A\,B\,\overline{C}$

$Y = \overline{A}\,B + A\,B\,\overline{C}$

Durham University

Example

Draw the circuit corresponding to this truth table.

1. Sum of products form
2. Simplification using Boolean Algebra

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

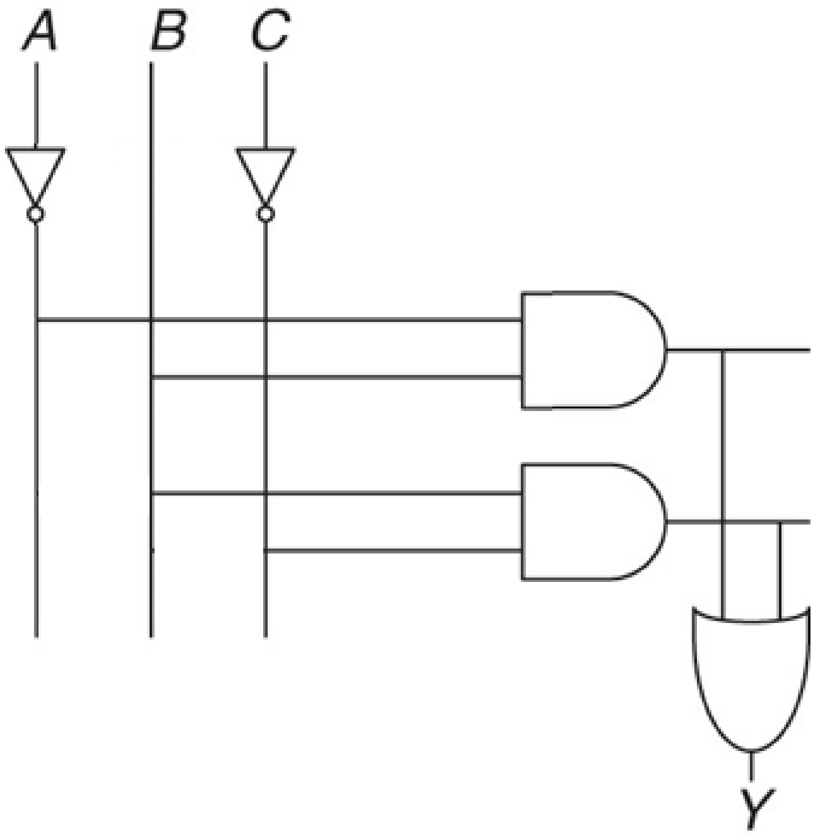
Sum of products: $Y = \underbrace{\boldsymbol{\overline{A}\,B\,\overline{C}} + \overline{A}\,B\,C} + \underbrace{\boldsymbol{\overline{A}\,B\,\overline{C}} + A\,B\,\overline{C}}$

Now minimise this equation: $Y = \overline{A}\,B\left(\overline{C} + C\right) + (A + \overline{A})\,B\,\overline{C}$

$Y = \overline{A}\,B + B\,\overline{C}$

Example

$$Y = \overline{A}\,B + B\,\overline{C}$$



The simplified expression gives the same logical output with much less hardware.

Summary

- Functionally Complete Sets
- Combinational Logic and Circuits
- Sum of Products and Product of Sums
- Boolean Algebra