P2 Digital Electronics

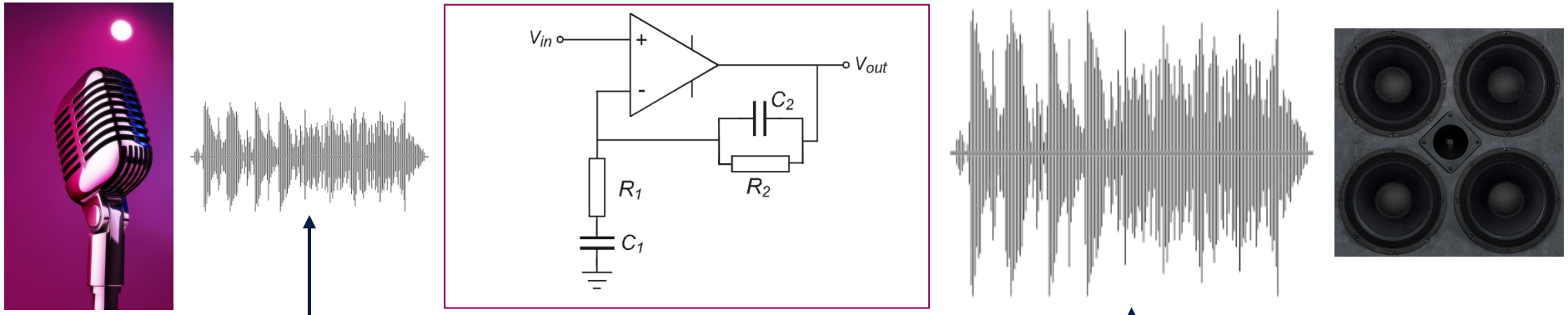
Lecture 1: Logic functions and gates

Mark Cannon

mark.cannon@eng.ox.ac.uk

Trinity Term 2026

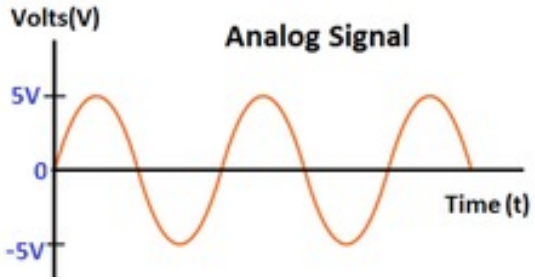
Analogue electronics

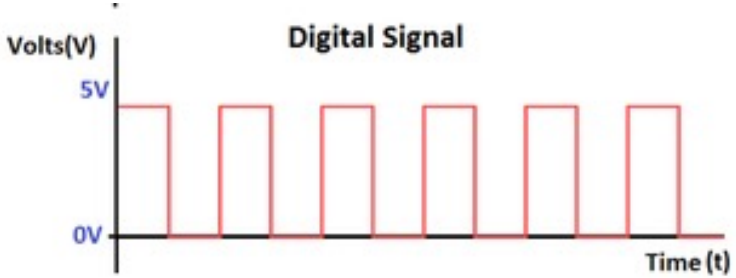


Digital electronics

Embedded Systems







Only two possible values 0 (LOW) and (HIGH) corresponding to two different voltage le1vels

Advantages:

- More efficient processing and transmission
- Reduced storage need
- Digital data is less susceptible to noise than analogue

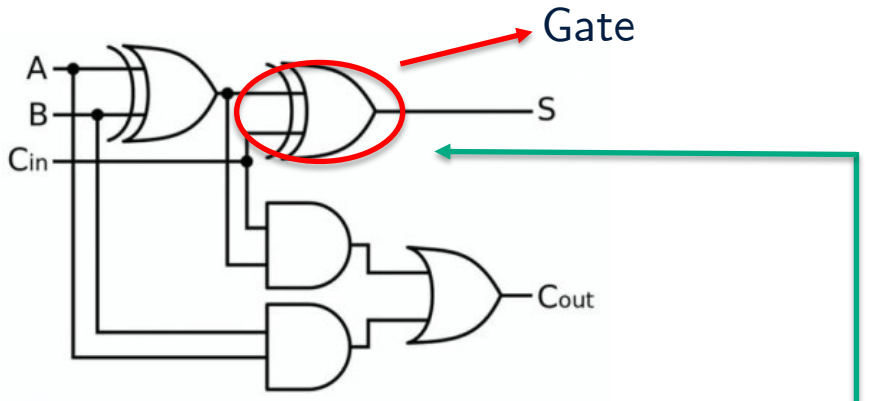
Digital electronics

Embedded Systems



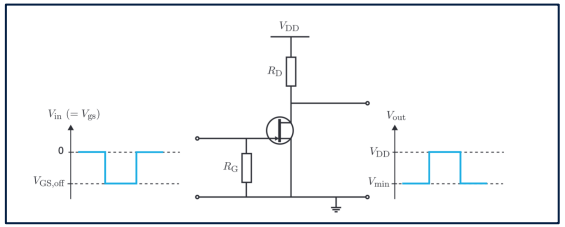
Functions:

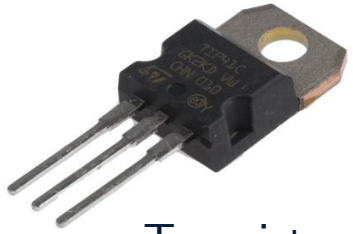
- Communication
- Computation
- Device Control
- ....



Digital Circuit

From previous P2 lectures:



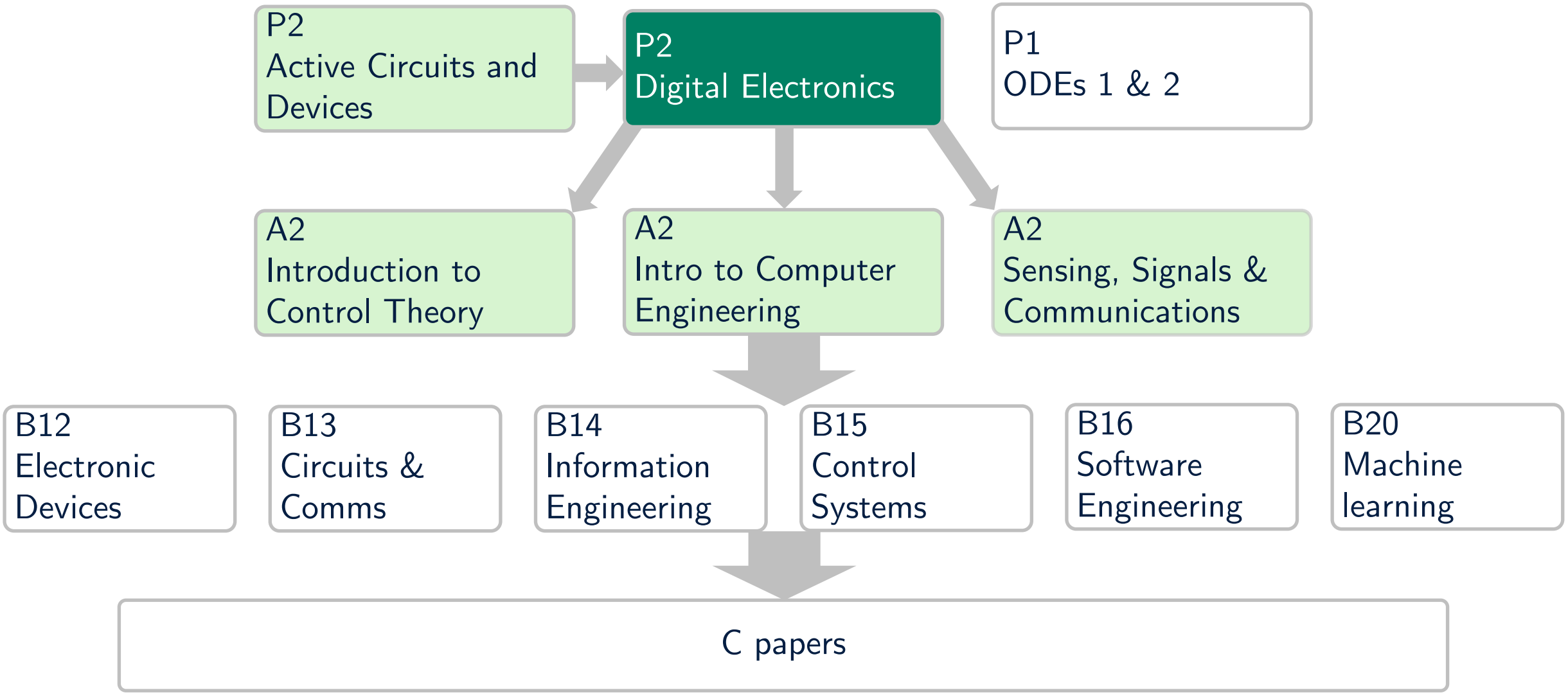


Transistor

Overview of this course

1. Logical functions and logic gates
2. Low level logic design
3. Binary number representation
4. Binary arithmetic
5. Integration of digital logic components
6. Memory and sequential circuits
7. Design of sequential logic
8. Data converters: analogue-to-digital and digital-to-analogue

Overview of this course



Suggested reading

- **Logic and Computer Design Fundamentals**, M.M. Mano, C.R.Kime, Pearson, 2015 (ISBN 0131911651)
- **Digital Fundamentals**, Thomas L Floyd, Pearson, 2015 (ISBN 1292075988)
- **Digital Logic and Microprocessors**, F.J. Hill and G.R. Peterson, Wiley, 1984 (ISBN 047182979X)
- **Applied Digital Electronics**, D.C. Green, Longman, 1999 (ISBN 0582356326)
- **The Essence of Digital Design**, B. Wilkinson, 1997 (ISBN 0135701104)

Logic expressions

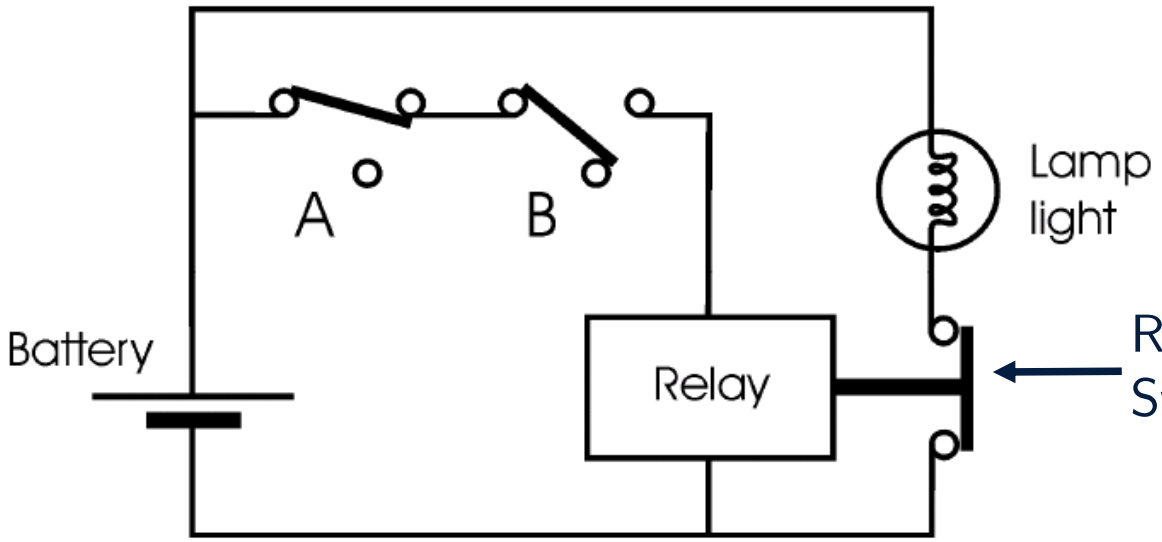
"The light should be on during the night if the motion sensor detects someone is in the room, or during the day if the switch is pressed"

Define some symbols to help us:

- LIGHT means "the light is on"
- DAY means "during the day" (also opposite of "during the night")
- SENSOR means "the motion sensor is activated"
- SWITCH means "the switch is pressed"

LIGHT = ((not DAY) and SENSOR) or (DAY and SWITCH)

A simple logic circuit



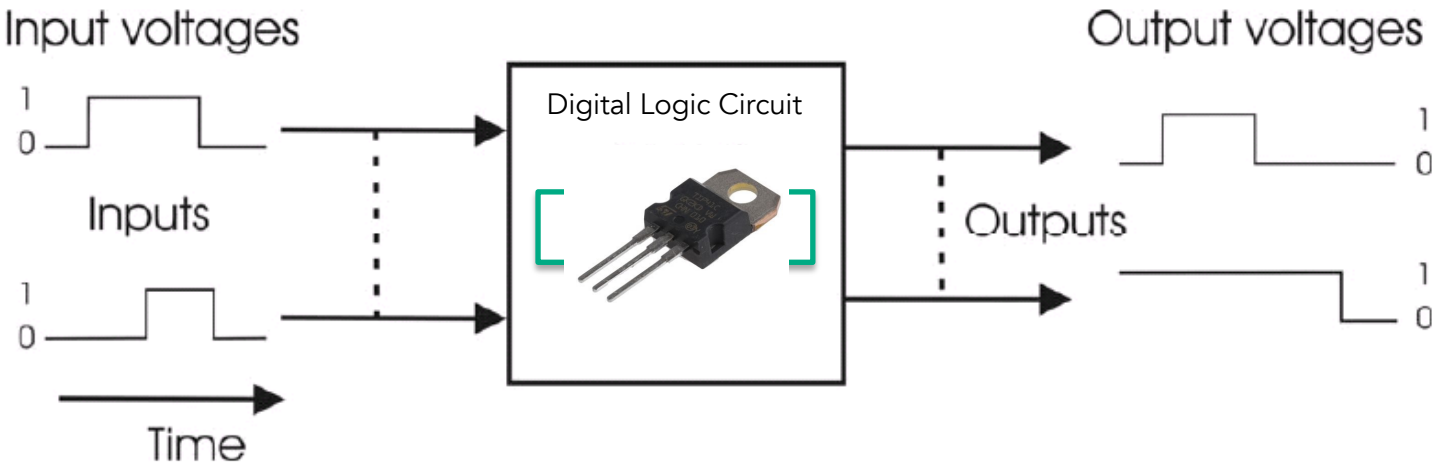
Relay is closed when no current flows through it
Switch opens when current flows through relay

Inputs = switch conditions **Outputs = states**

Switch A	Switch B	Relay state	Lamp state
Open	Open	OFF	ON
Open	Closed	OFF	ON
Closed	Open	OFF	ON
Closed	Closed	ON	OFF

Binary variables:
On/Off or TRUE/FALSE or 1/0

Logic states

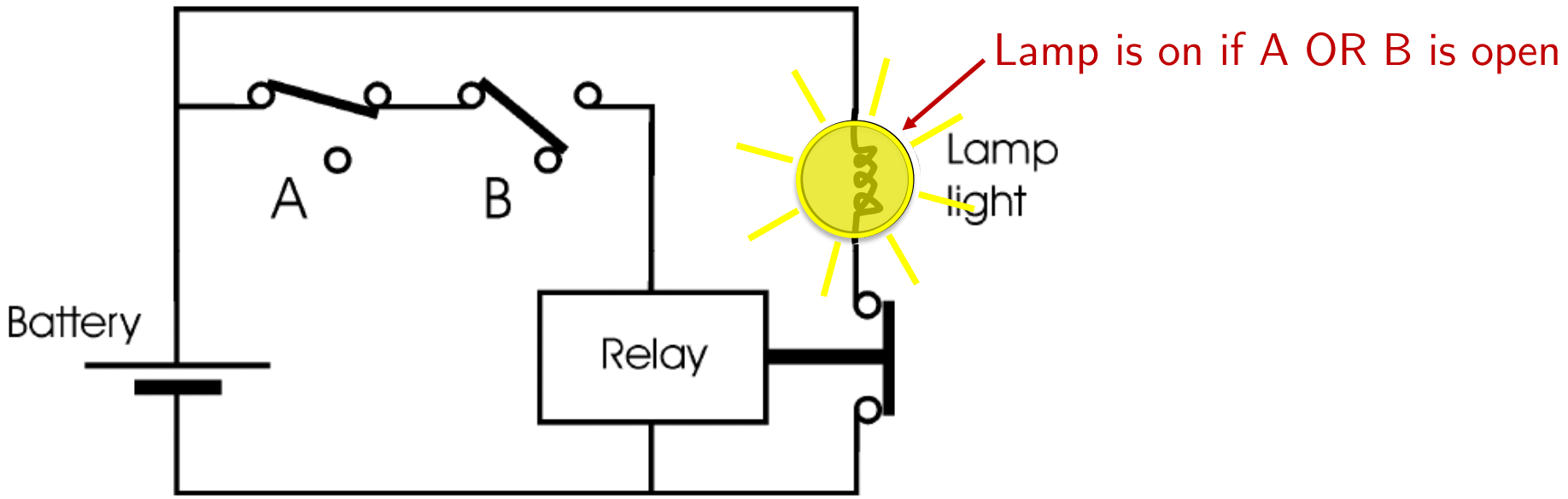


Voltage level	Logic value
0 V	FALSE, logic 0
5 V	TRUE, logic 1

N.B.: states depend on logic type in use: some circuits use 0 and 5V, others 0 and 3V and some work as low as 0 and 1.5V

(Electrical power is proportional to voltage^2, hence desire for lower voltages)

Truth tables and logic



Truth table:

A	B	Lamp	$\overline{\text{Lamp}}$
0	0	1	0
0	1	1	0
1	0	1	0
1	1	0	1

Logic function:

$$\text{Lamp} = (\text{not A}) \text{ or } (\text{not B}) \implies \text{Lamp} = \overline{A} + \overline{B}$$

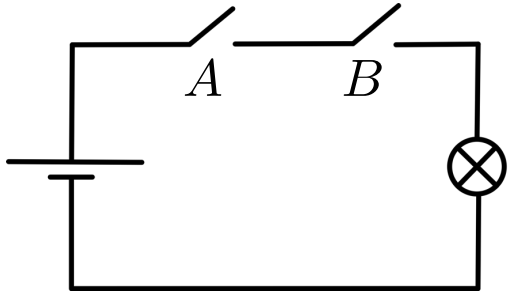
$$\text{not Lamp} = \text{A and B} \implies \overline{\text{Lamp}} = A.B$$

$$\text{Lamp} = \text{not } (\text{A and B}) \implies \text{Lamp} = \overline{A.B}$$

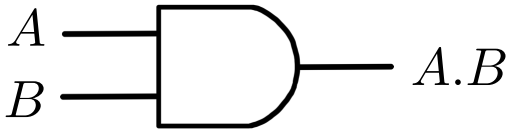
Logic gates

Associated with binary variables are **Operators**: AND, OR, NOT (the basic 3), NOR, NAND and EXCLUSIVE-OR

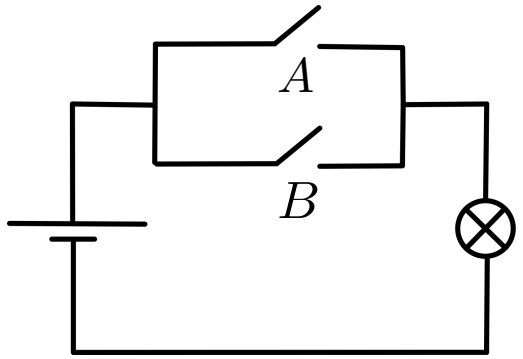
AND: $A.B$ (or $A \wedge B$)



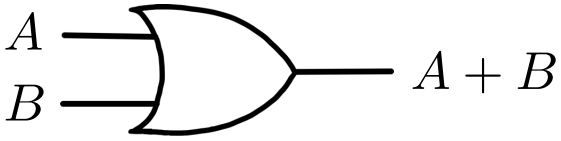
A	B	$A.B$
0	0	0
0	1	0
1	0	0
1	1	1



OR: $A + B$ (or $A \vee B$)



A	B	$A + B$
0	0	0
0	1	1
1	0	1
1	1	1



Clearly: $A.A = A$ $A.0 = 0$ $A.1 = A$ $A.\overline{A} = 0$

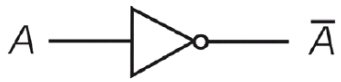
$A + A = A$ $A + 0 = A$ $A + 1 = 1$ $A + \overline{A} = 1$

Logic gates

Truth table

A	$\overline{A}$
0	1
1	0

NOT gate symbol



Truth table

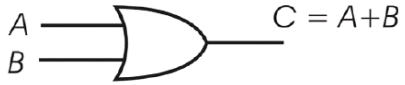
A	B	$C = A.B$
0	0	0
0	1	0
1	0	0
1	1	1

AND gate symbol

Truth table

A	B	$C = A+B$
0	0	0
0	1	1
1	0	1
1	1	1

OR gate symbol

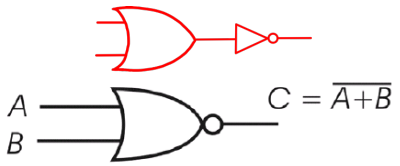


A or B, or both

Truth table

A	B	$C = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

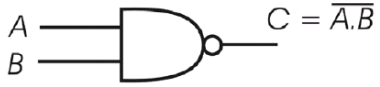
NOR gate symbol



Truth table

A	B	$C = \overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

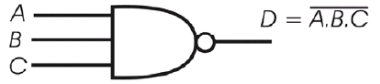
NAND gate symbol



Truth table

A	B	C	$D = \overline{A.B.C}$
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	0

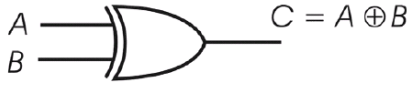
Three input NAND gate symbol



Truth table

A	B	$C = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

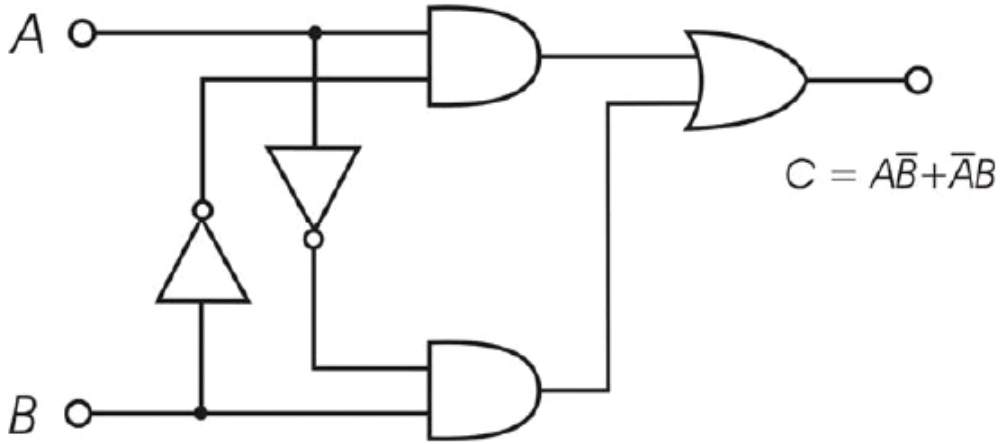
Exclusive OR gate symbol



A or B, but not both

Basic Boolean algebra: XOR

$$C = A \text{ xor } B = \big(A \text{ and } (\text{not } B)\big) \text{ or } \big(B \text{ and } (\text{not } A)\big) = A.\overline{B} + B.\overline{A}$$

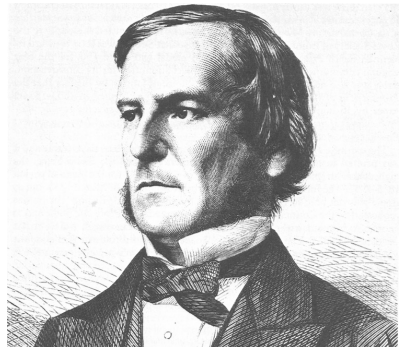


A	B	C = A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A.B̄	Ā.B	C = A.B̄+Ā.B
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

EQUIVALENT

Laws of Boolean algebra



George Boole

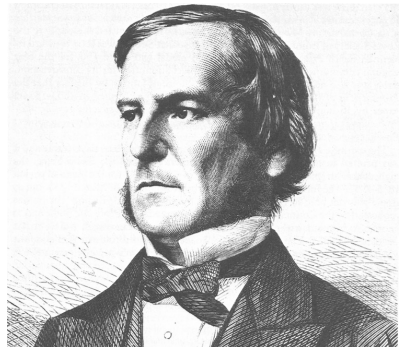
Axioms

$X + Y = Y + X$		$X.Y = Y.X$		Commutativity (order of args)
$(X + Y) + Z = X + (Y + Z)$		$(X.Y).Z = X.(Y.Z)$		Associativity
$X + 0 = X$	$X + 1 = 1$	$X.0 = 0$	$X.1 = X$	Dominance or identity
$X.(Y + Z) = X.Y + X.Z$				Distributive law 1
$X + Y.Z = (X + Y)(X + Z)$				Distributive law 2

(HLT section Electricity/Digital Logic)

Laws of Boolean algebra

Let's prove some of these axioms



George Boole

Dominance

$X + 0 = X$ $\quad$ $X + 1 = 1$

X	Y	$X+Y$	
0	0	0	$= X$
0	1	1	$= 1$
0	0	1	$= X$
0	0	1	$= 1$

Distributivity (OR over AND)

$$X + Y.Z = (X + Y).(X + Z)$$

$$\begin{aligned}
\text{RHS} &= (X + Y).(X + Z) \\
&= \underbrace{X.X}_{=X} + \underbrace{X.Z + Y.X}_{=X.(Y+Z)} + Y.Z \\
&= X + X.(Y + Z) + Y.Z \\
&= X.\underbrace{\underbrace{(1 + Y + Z)}_{=1}}_{=X} + Y.Z \\
&= X + Y.Z \\
&= \text{LHS}
\end{aligned}$$

Laws of Boolean algebra

Augustus De Morgan

Axioms

$X + Y = Y + X$		$X.Y = Y.X$		Commutativity (order of args)
$(X + Y) + Z = X + (Y + Z)$		$(X.Y).Z = X.(Y.Z)$		Associativity
$X + 0 = X$	$X + 1 = 1$	$X.0 = 0$	$X.1 = X$	Dominance or identity
$X.(Y + Z) = X.Y + X.Z$				Distributive law 1
$X + Y.Z = (X + Y)(X + Z)$				Distributive law 2

and some basic theorems

$\overline{X + Y} = \overline{X}.\overline{Y}$	De Morgan's law 1
$\overline{X.Y} = \overline{X} + \overline{Y}$	De Morgan's law 2

Laws of Boolean algebra

Augustus De Morgan

$\overline{X+Y} = \overline{X}.\overline{Y}$	De Morgan's law 1
$\overline{X.Y} = \overline{X} + \overline{Y}$	De Morgan's law 2

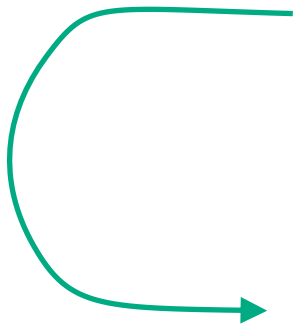
Let's prove De Morgan's Law 1

X	Y	$X+Y$	$\overline{X+Y}$	$\overline{X}$	$\overline{Y}$	$\overline{X}.\overline{Y}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Universal gates

A **universal gate** is a gate that can implement any Boolean function without needing any other type of gate. NAND and NOR are universal gates

e.g. We can replace OR with a combination of NAND and NOT gates:



$$\overline{X+Y} = \overline{X}.\overline{Y} \qquad \text{De Morgan's Law 1}$$

$$\overline{X.Y} = \overline{X} + \overline{Y} \qquad \text{De Morgan's Law 2}$$

$$\implies \quad X + Y = \overline{\overline{X+Y}} = \overline{\overline{X}.\overline{Y}}$$

therefore X OR Y is equivalent to (NOT X) NAND (NOT Y)

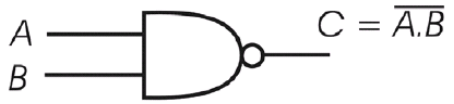
and we can also express NOT using NAND gates . . .

NAND universality

Truth table

A	B	$C = \overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

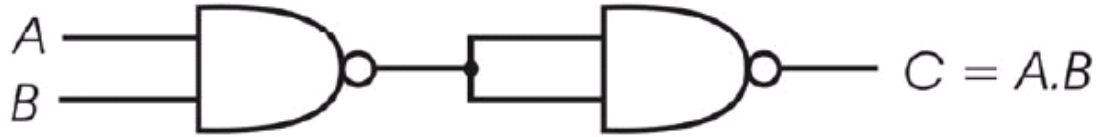
NAND gate symbol



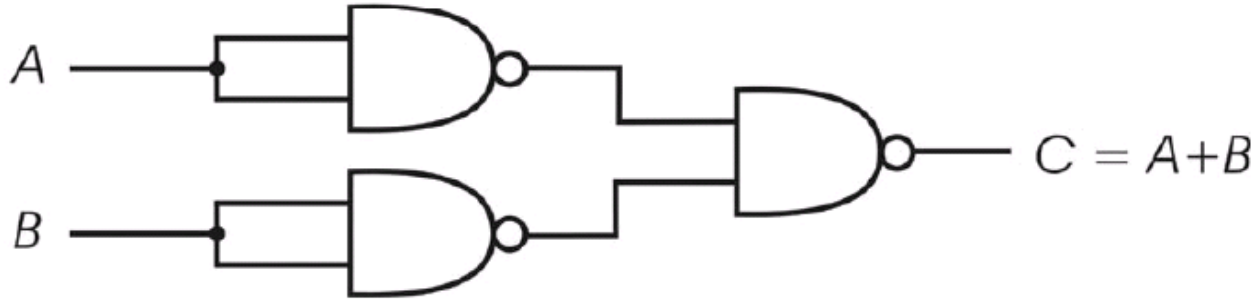
All logic functions can be implemented using NAND gates alone!

NOT

AND



OR

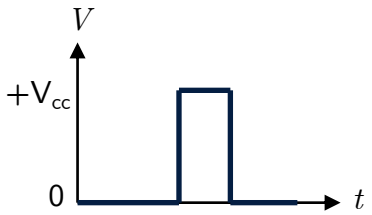


Positive versus negative logic

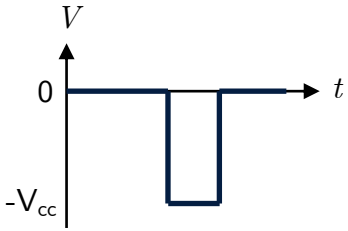
- In positive logic, *high* voltage represents 1 and *low* voltage represents 0
  In negative logic, *low* voltage represents 1 and *high* voltage represents 0

- NAND is the universal gate for positive logic

  We can change from positive logic to negative logic by inverting all inputs and outputs, and changing all NAND gates to NOR gates



Positive logic



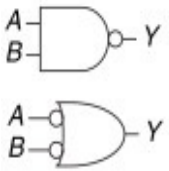
Negative logic

$A \leftarrow \overline{A}$, $B \leftarrow \overline{B}$ etc.

De Morgan: $\overline{X+Y} = \overline{X}.\overline{Y}$

$\overline{X.Y} = \overline{X} + \overline{Y}$

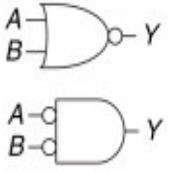
NAND



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

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

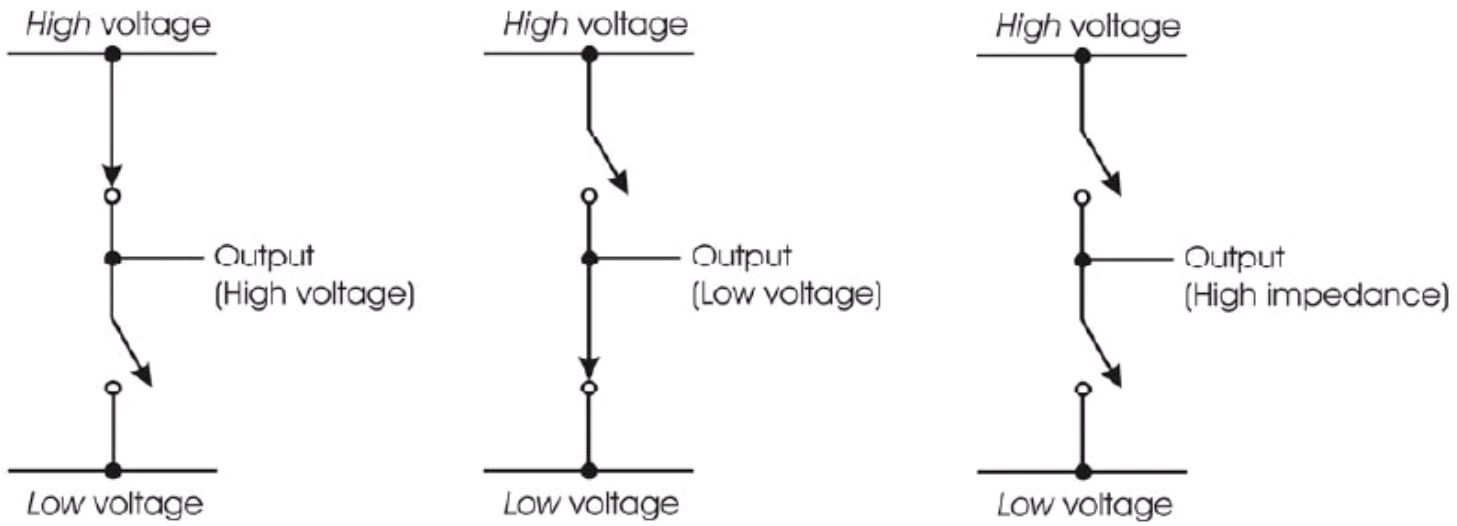
NOR



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

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

Logic as switches

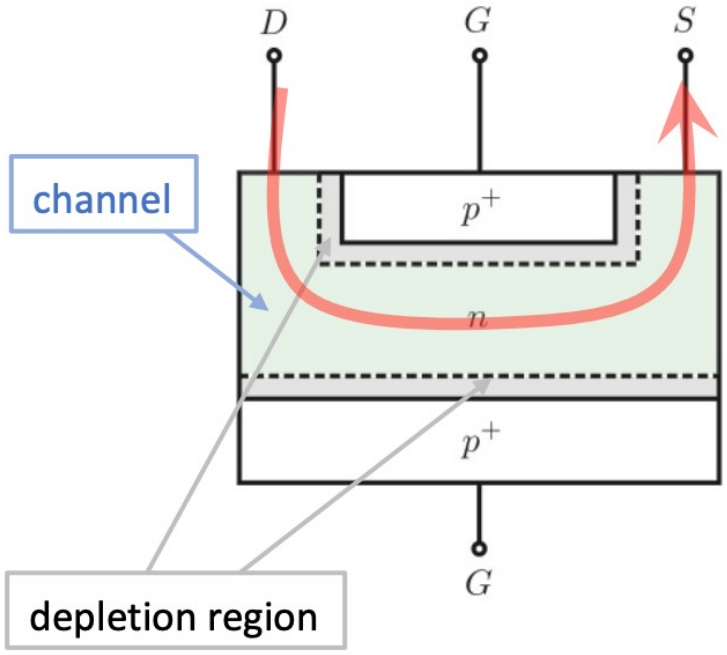


- To implement logic circuits, we use CMOS FET (Complementary Metal-Oxide-Semiconductor, Field Effect Transistors) which are thermally stable and power efficient
- A logic level is represented by two transistors used as switches. The output is controlled by deciding which switch is open or closed

Switches using MOSFETs

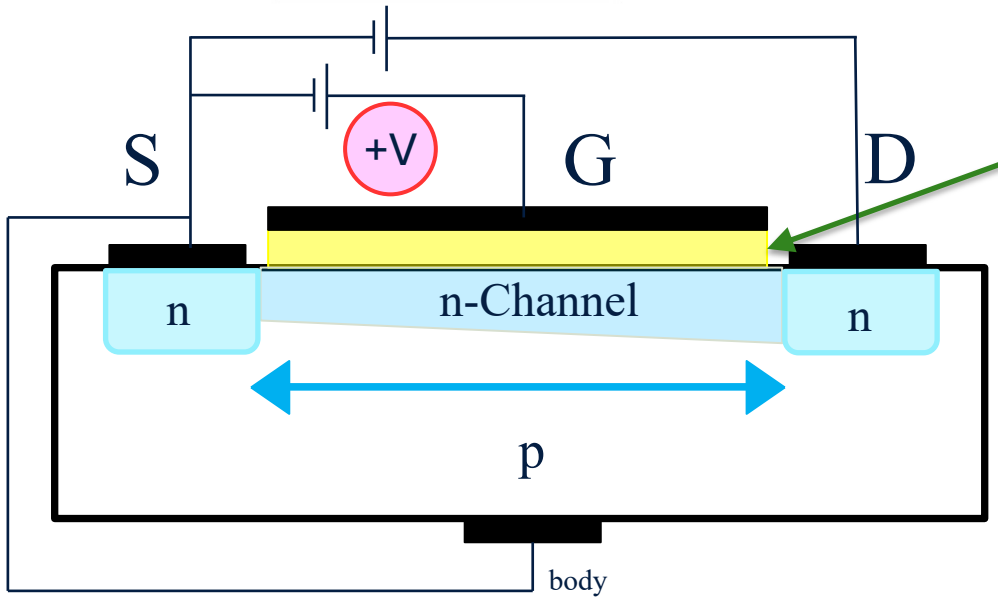
JFET

(P2 Active devices lectures)



- Conduction through n-channel
- Gate voltage controls current between source-drain by changing depletion region

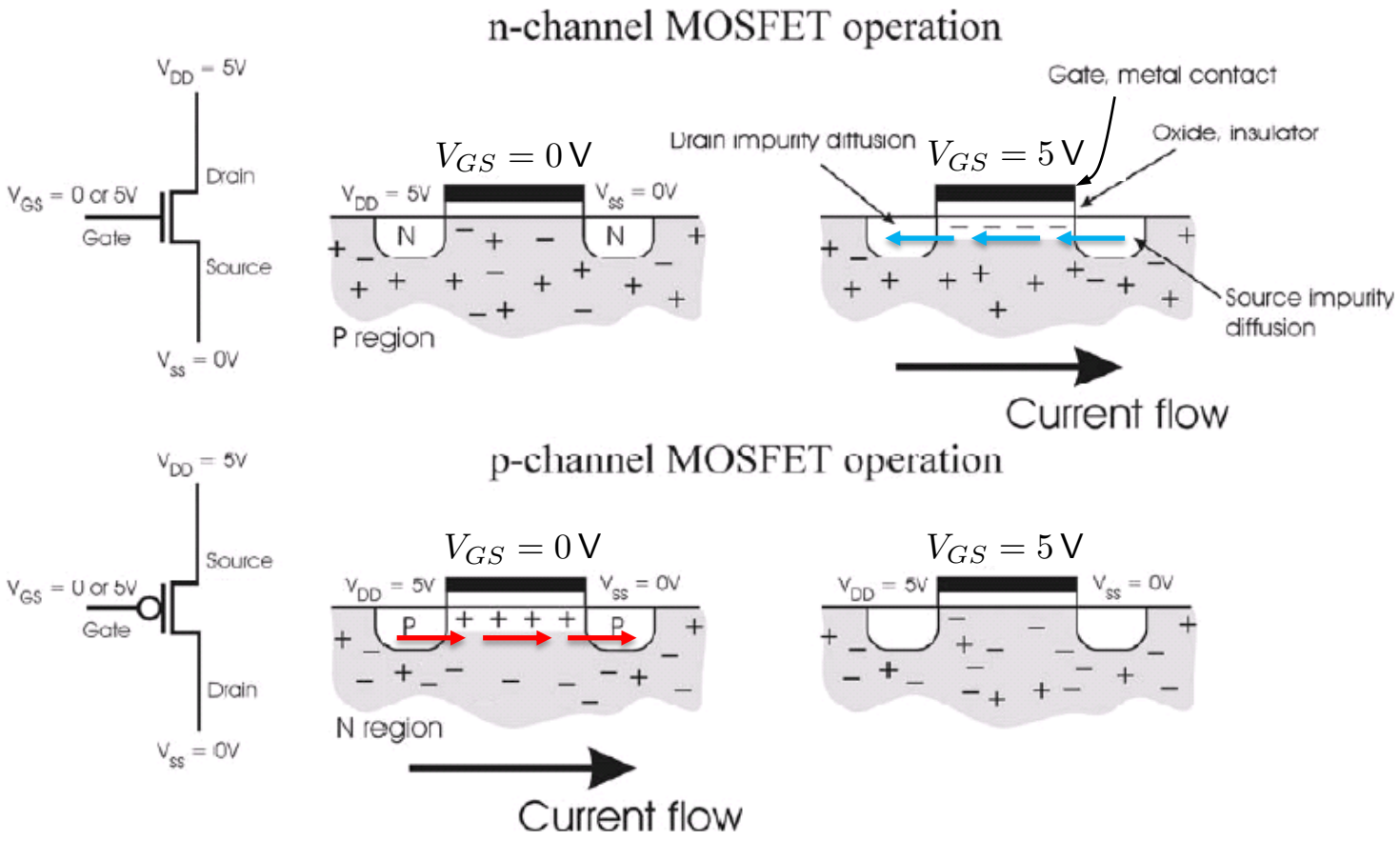
MOSFET



Metal oxide layer (insulator)

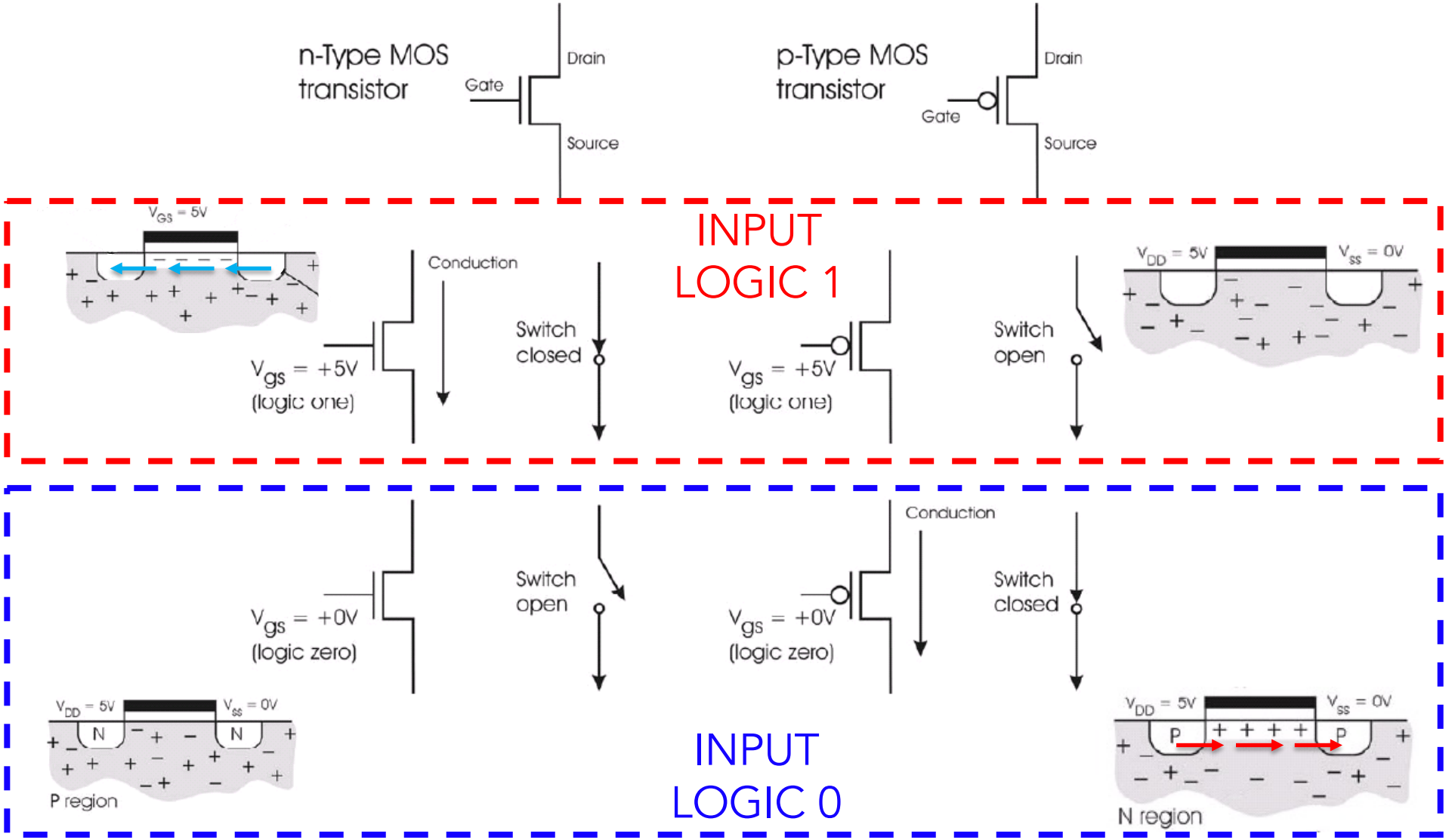
- Long p region
- Controlled by electric fields
- n-channel created by positive gate voltage

p-MOS and n-MOS devices

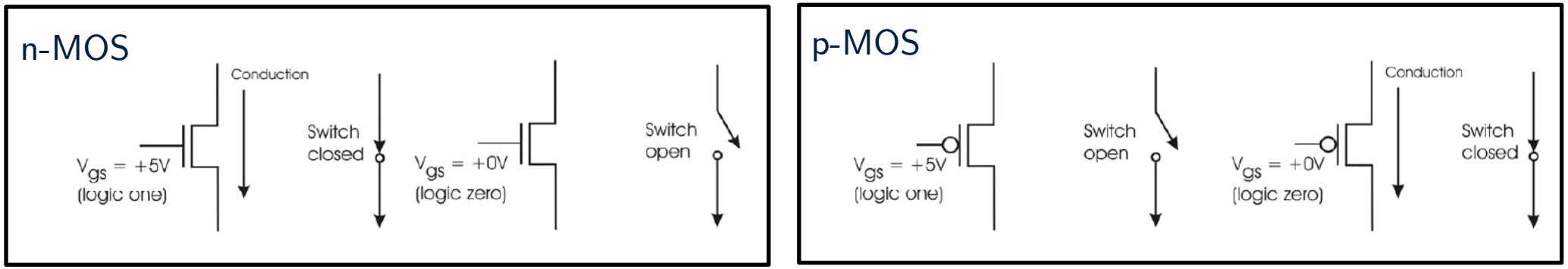


- CMOS: **Complementary Metal-Oxide-Semiconductor**
- Uses p and n type conductors (complementary)
- Electric field driven devices (fast); only consume power during change of state

p-MOS and n-MOS logic



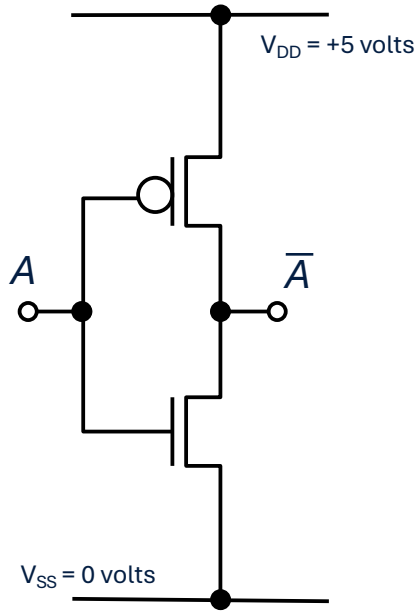
CMOS digital logic implementation



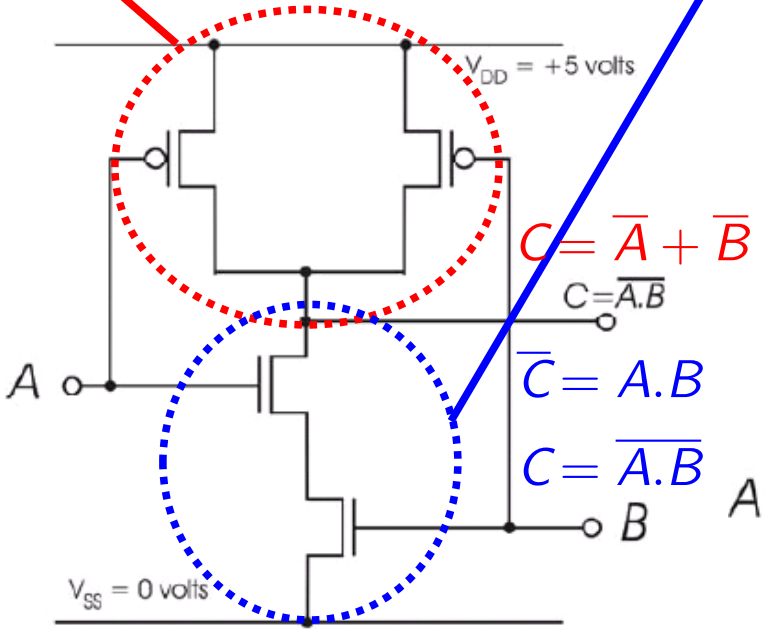
Output connected high if A or B is low

Output connected low if A and B are high

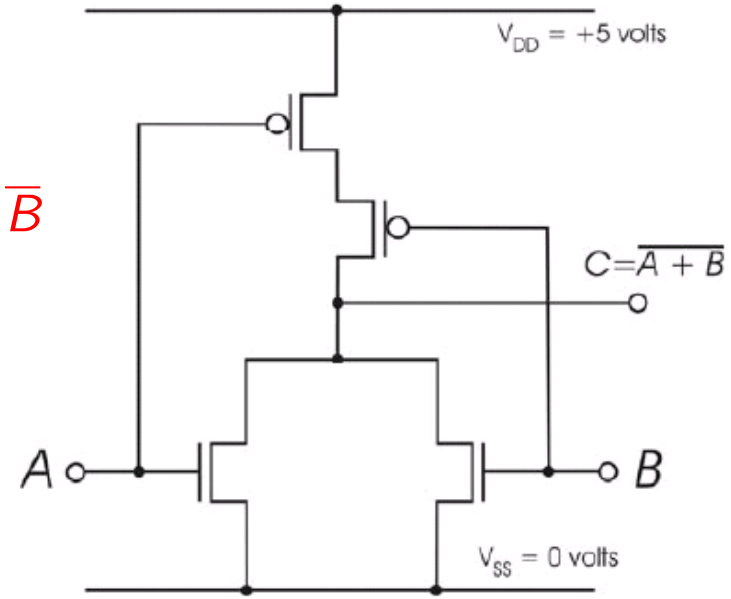
CMOS NOT



CMOS NAND gate



CMOS NOR gate



Overview of lectures

1. **Logical functions and logic gates**
2. Low level logic design
3. Binary number representation
4. Binary arithmetic
5. Integration of digital logic components
6. Memory and sequential circuits
7. Design of sequential logic
8. Data converters: analogue to digital / digital to analogue

Please send feedback, comments and corrections to mark.cannon@eng.ox.ac.uk