BIOE40002 – Computer Fundamentals and Programming 1

Part I – Digital Logics, Lab 1

Binghuan Webster Li | *Department of Bioengineering*

binghuan.li19@imperial.ac.uk

January 20, 2022

Term schedule at a glance

- **MBE Group 13: Thursdays 11pm – 1am, via Teams**

- **Week 2-6 (5 weeks):** Part I - Digital Logics
- **Week 7:** Reading week, *no* computer session scheduled
- **Week 8–11(4 weeks):** Part II – Programming 1

- For each session, your registration record will be kept to fulfil your learning credentials.

Part I – Digital Logic

Imperial College London

- **Aim:** Understanding the basic of digital logic circuits and the mechanisms.
- **Your Tasks:**
  - Follow the guidance on the task sheet, build the circuit and perform a series of simulations in Quartus.
  - Upon the completion of five sessions, you are expected to finish the first 11 tasks in your work sheet
- **What are available to you?**
  - Key resources on BlackBoard: *(1) Lab book (2) FAQ & Tips (3) course slides*
  - Me ☺
- **What should you prepare?**
  - A computer installed Quartus
  - Your logbook
  - Be a friendly partner

Part II – Programming 1

Imperial College London

- Python (3 weeks)
  - Variables, data types
  - Input and Output (I/O)
  - Flow of control: `for` loops, `while` loops, `if…else…` conditions
  - Functions and modular programming
  - Useful/popular libraries
- Arduino (1 week)
  - Microcontrollers

Lab Sessions Guides

Imperial College London

- **Time, time and time!**
- Guide No.1
  - You are deffo encouraged to show your face… I promise I will hi ☺
- ***Do*s and *Don't*s**
  - ***Do*** follow your lab book – read all descriptions and figures!
  - ***Do*** think independently. Persuade yourself first.
  - ***Do*** ask questions. Unmute yourself when you need me.
  - ***Do*** take your own time, ***don't*** rush through the tasks.
  - ***Do*** keep your log-book.

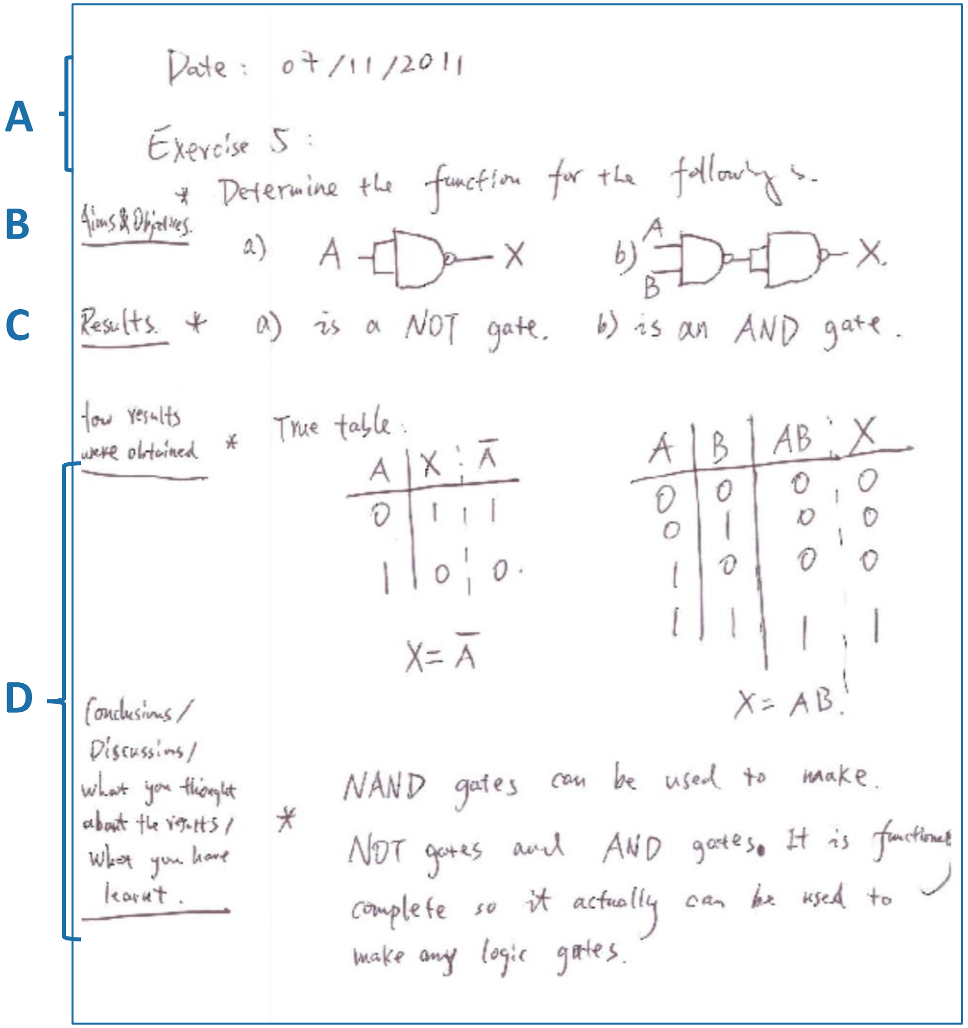
Your log-book

Imperial College London

- Log-book keeps a detailed record of your work...
  - Will *not* be assessed directly
  - Does not have to be a physical notebook
  - Take notes, regardless of your results are correct or wrong

- How would you structure your log-book?
  - A. Title - Date & task number
  - B. Aims - What you planned to do
  - C. Results - What results were obtained
  - D. Discussion -
    - a) How they were obtained
    - b) How you think about the results

- Use tables and sketches where appropriate



Imperial College London

Questions?

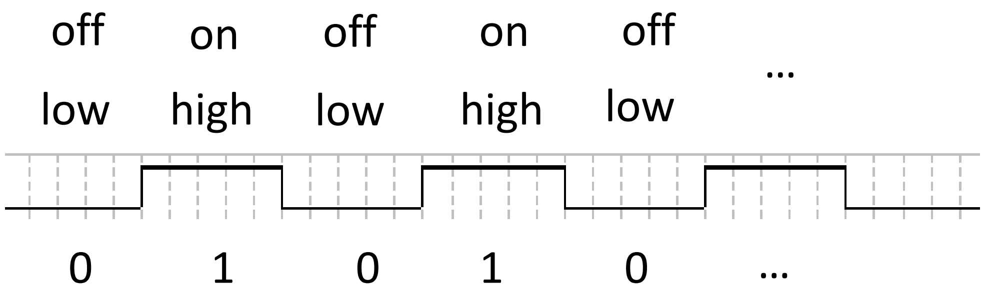
Today's Schedule

Imperial College London

- Recap (~ 10 mins)
  - Notations in digital logic
  - Logic gates and truth tables
  - Laws of Boolean algebra
  - CMOS
- Lab work kick-off

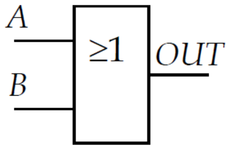
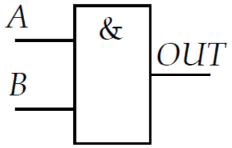
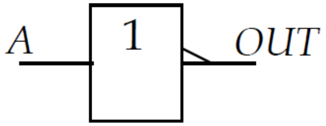
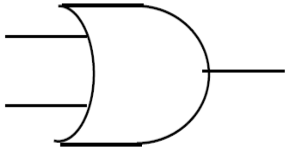
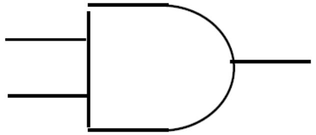
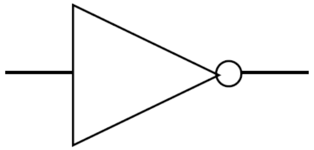
Common notations in digital logic

- 1 – 'high voltage', 'true', 'on'…
- 0 – 'low voltage', 'false', 'off'…



Logic Gates and Truth Tables - 1

Imperial College London

	OR	AND	NOT
	$OUT = A + B$	$OUT = A \cdot B$	$OUT = \bar{A}$
IEEE symbols			
Mil-spec symbols (in your task sheet)			

OR truth table:

A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	1

AND truth table:

A	B	OUT
0	0	0
0	1	0
1	0	0
1	1	1

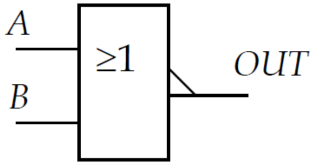
NOT truth table:

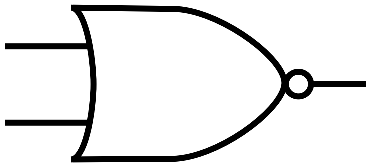
A	OUT
0	1
1	0

Logic Gates and Truth Tables - 2

NOR

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

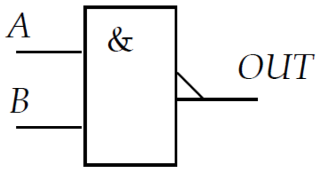


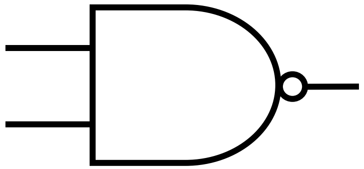


A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	0

NAND

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

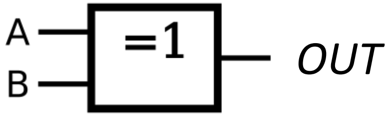


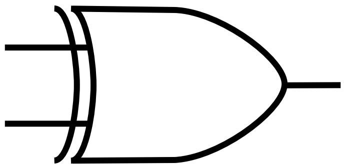


A	B	OUT
0	0	1
0	1	1
1	0	1
1	1	0

XOR

$$OUT = A \oplus B$$

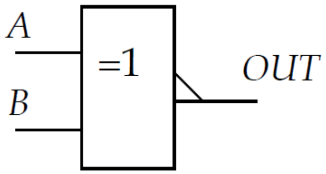


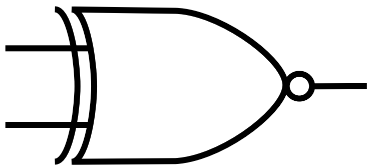


A	B	OUT
0	0	0
0	1	1
1	0	1
1	1	0

XNOR

$$OUT = \overline{A \oplus B}$$





A	B	OUT
0	0	1
0	1	0
1	0	0
1	1	1

"X" – exclusive

Produce TRUE when two inputs are different

Laws of Boolean Algebra

Imperial College London

- Commutative

$$A \cdot B = B \cdot A\,, \qquad A + B = B + A$$

- Associative

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

- Distributive

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

- De Morgan's Theorem

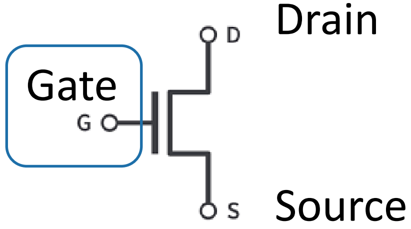
$$\bar{A} \cdot \bar{B} = \overline{A + B}, \qquad \bar{A} + \bar{B} = \overline{A \cdot B}$$

- Absorption

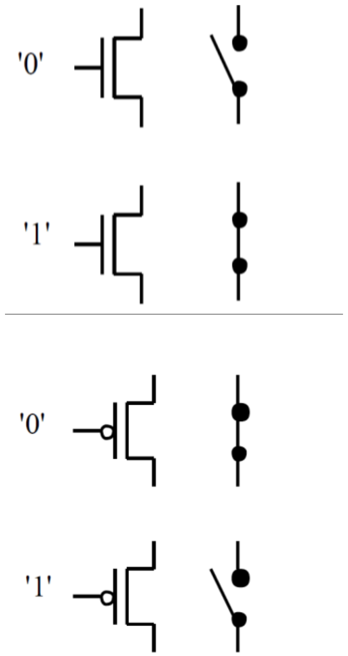
$$A + (A \cdot B) = A$$
$$A \cdot (A + B) = A$$
$$A + (\bar{A} \cdot B) = A + B$$

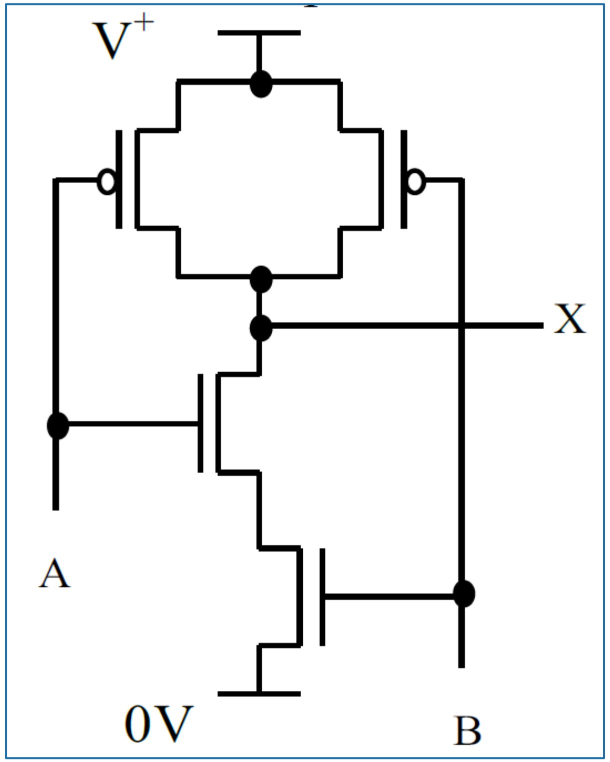
CMOS - Metal-oxide semiconductor

- CMOS work as switches



- N-channel CMOS
  - ‘1’ – switch on
  - ‘0’ – switch off
- P-channel CMOS
  - ‘1’ – switch off
  - ‘0’ – switch on





A	*B*	*OUT*
0	0	1
0	1	1
1	0	1
1	1	0

That’s it for now.

You can now proceed to the Exercise 1-5.

Task 1 – Gliders and Airliners

- Gliders have wings but no engines
- Airliners have both wings and engines

Wings (A)	Engine (B)	Glider (X)	Airliner (Y)
0	0	**0**	**0**
0	1	**0**	**0**
1	0	**1**	**0**
1	1	**0**	**1**

$X = A \cdot \bar{B}$ $\quad$ $Y = A \cdot B$

Task 2 – Verify De Morgan's Theorem

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

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

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

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

Task 3 – Boolean Simplification

Imperial College London

$$(\bar{V} + X) \cdot \left(W \cdot (\bar{Y} + Z)\right) + \overline{(\bar{V} + X)} \cdot \left(W \cdot (\bar{Y} + Z)\right)$$

$$= \underbrace{\left((\bar{V} + X) + \overline{(\bar{V} + X)}\right)}_{1 \quad \text{Since } A + \bar{A} \equiv 1} \cdot \left(W \cdot (\bar{Y} + Z)\right)$$ Distributive law

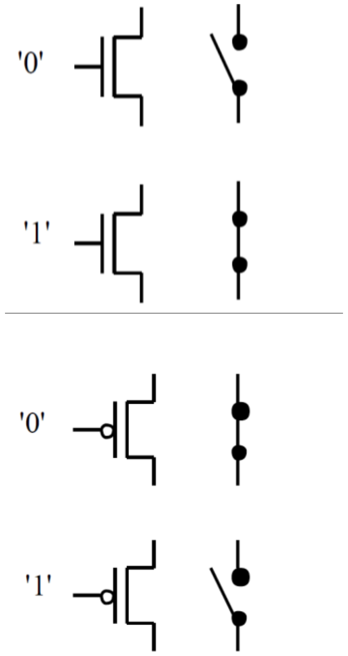
$$= \left(W \cdot (\bar{Y} + Z)\right)$$

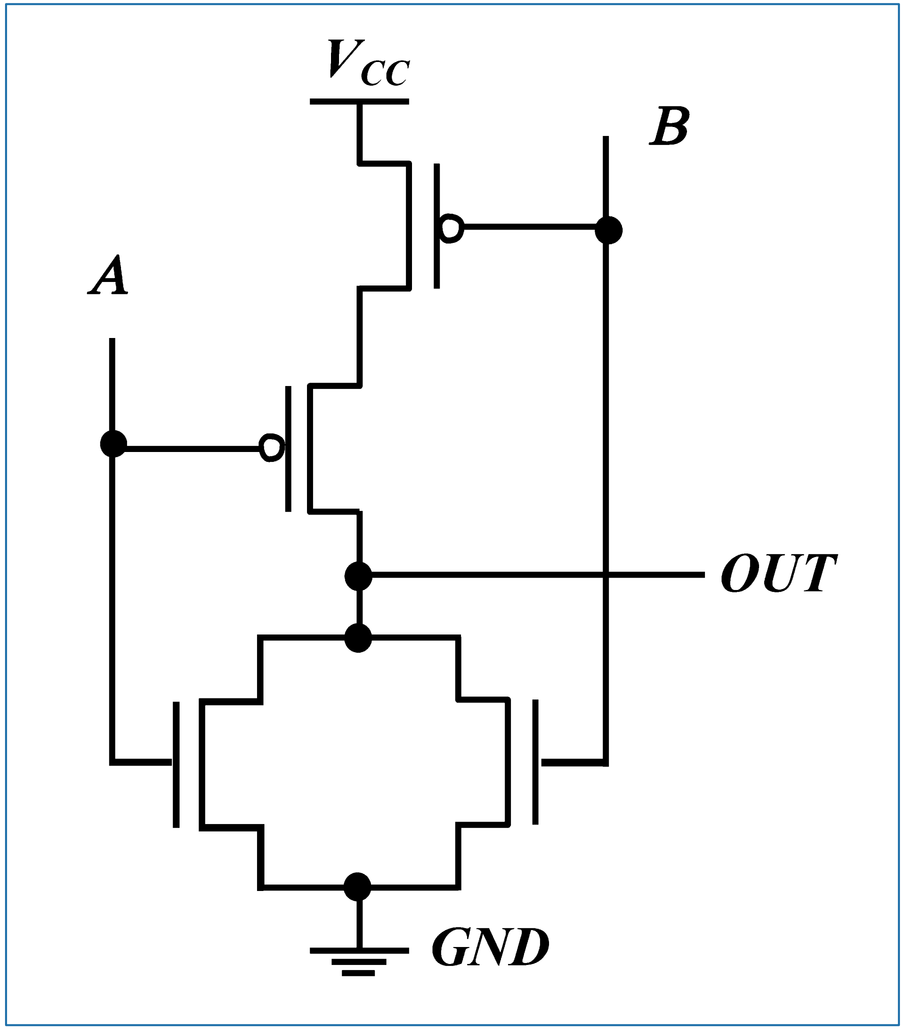
$$= \boldsymbol{W \cdot \bar{Y} + W \cdot Z}$$ Distributive law

Task 4 – design a *NOR* gate using CMOS

Imperial College London

- CMOS work as switches
- N-channel CMOS
  - ‘1’ – switch on
  - ‘0’ – switch off
- P-channel CMOS
  - ‘1’ – switch off
  - ‘0’ – switch on

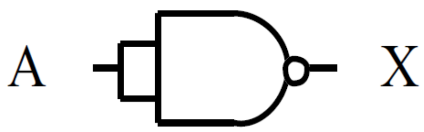




Imperial College London

Task 5.1 – *NAND* gates analysis

a)

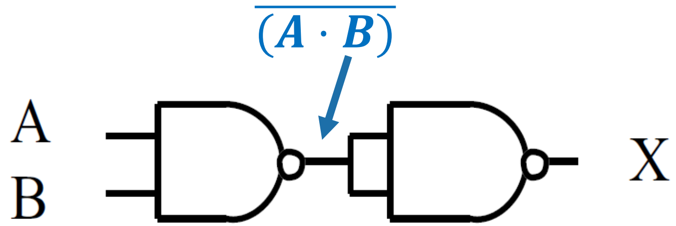


A	X
0	*1*
1	*0*

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

NAND gates can be used to create *NOT* gates

b)



A	B	$\overline{(A \cdot B)}$	X
0	0	*1*	*0*
0	1	*1*	*0*
1	0	*1*	*0*
1	1	*0*	*1*

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

NAND gates can be used to create *AND* gates

Task 5.2 – design an *OR* gate with *NAND* gates

Imperial College London

- We have seen $\bar{A} \cdot \bar{B} = \overline{A + B}$ (De Morgan's Theorem), thus $\overline{\bar{A} \cdot \bar{B}} = \underbrace{A + B}_{OR}$
- The question has simplified to "use *NAND* gates to represent $\overline{\bar{A} \cdot \bar{B}}$
- *NOT* / *AND* gates are also available.

A	B	$\bar{A}$	$\bar{B}$	$X = \overline{(\bar{A} \cdot \bar{B})}$
0	0	1	1	0
0	1	1	0	0
1	0	1	1	0
1	1	0	0	1