Computer Science

Lesson Breakdown

- 4 Lessons of Paper 2
- 3 Lessons of Paper 1
- 2 Lessons Programming Time

Recommended Course Books

- Tackling A Level Projects in Computer Science AQA 7517 Paperback – 30 Jan. 2020
- AQA AS and A Level Computer Science Paperback – 24 April 2016
- AQA A Level Computer Science Workbook by Isaac Computer Science
- C# Programming Yellow Book “Cheese” Edition 8.2, Rob Miles – November 2016

Other Resources in Use

- Craig & Dave Videos
- IsaacComputerScience.org
- CodeWars.com
- Programming IDEs (Visual Studio, VS Code, IDLE etc.)
- Microsoft Teams

Course Structure

- AQA 7517
- Paper 1, 150 mins (40%)
- Paper 2, 150 mins (40%)
- NEA (20%)

Previous Project Ideas

- Rocket Simulator
- Games
- Tools for TTRPGs
- Mobile Fitness Apps
- Machine Learning
- Satellite Simulator
- Minecraft Mod Investigation Modelling
- F1 Cornering Simulator
- Football Management Tools

Extra-Curricular Opportunities

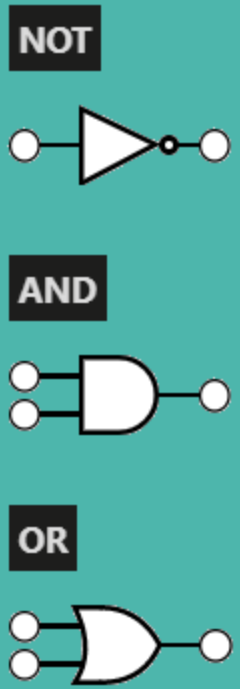
-(Government Guidance Permitting)-

- Space Lab
- Girls Who Code
- Bletchley Park & The National Museum of Computing
- Computer Science in Action Talk
- RU Hacking
- British Informatics Olympiad
- Other Higher Education Activities

Lesson Time!

Boolean Logic

Recapping the Basics



Which Truth Table?

Table 1		
In A	In B	Out
0	0	0
0	1	0
1	0	0
1	1	1

Table 2	
In	Out
0	1
1	0

Table 3		
In A	In B	Out
0	0	0
0	1	1
1	0	1
1	1	1

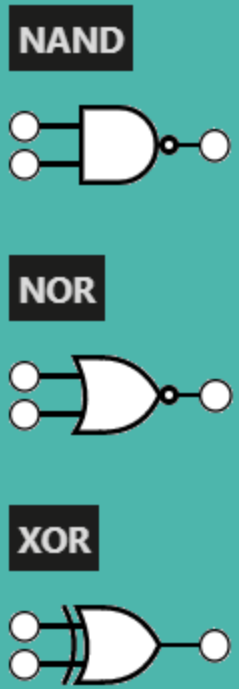
Which Truth Table?

AND		
InA	InB	Out
0	0	0
0	1	0
1	0	0
1	1	1

NOT	
In	Out
0	1
1	0

OR		
InA	InB	Out
0	0	0
0	1	1
1	0	1
1	1	1

New Gates



Which Truth Table?

Table 1		
InA	InB	Out
0	0	0
0	1	1
1	0	1
1	1	0

Table 2		
InA	InB	Out
0	0	1
0	1	1
1	0	1
1	1	0

Table 3		
InA	InB	Out
0	0	1
0	1	0
1	0	0
1	1	0

Which Truth Table?

XOR (eXcusive OR)		
InA	InB	Out
0	0	0
0	1	1
1	0	1
1	1	0

NAND (Not AND)		
InA	InB	Out
0	0	1
0	1	1
1	0	1
1	1	0

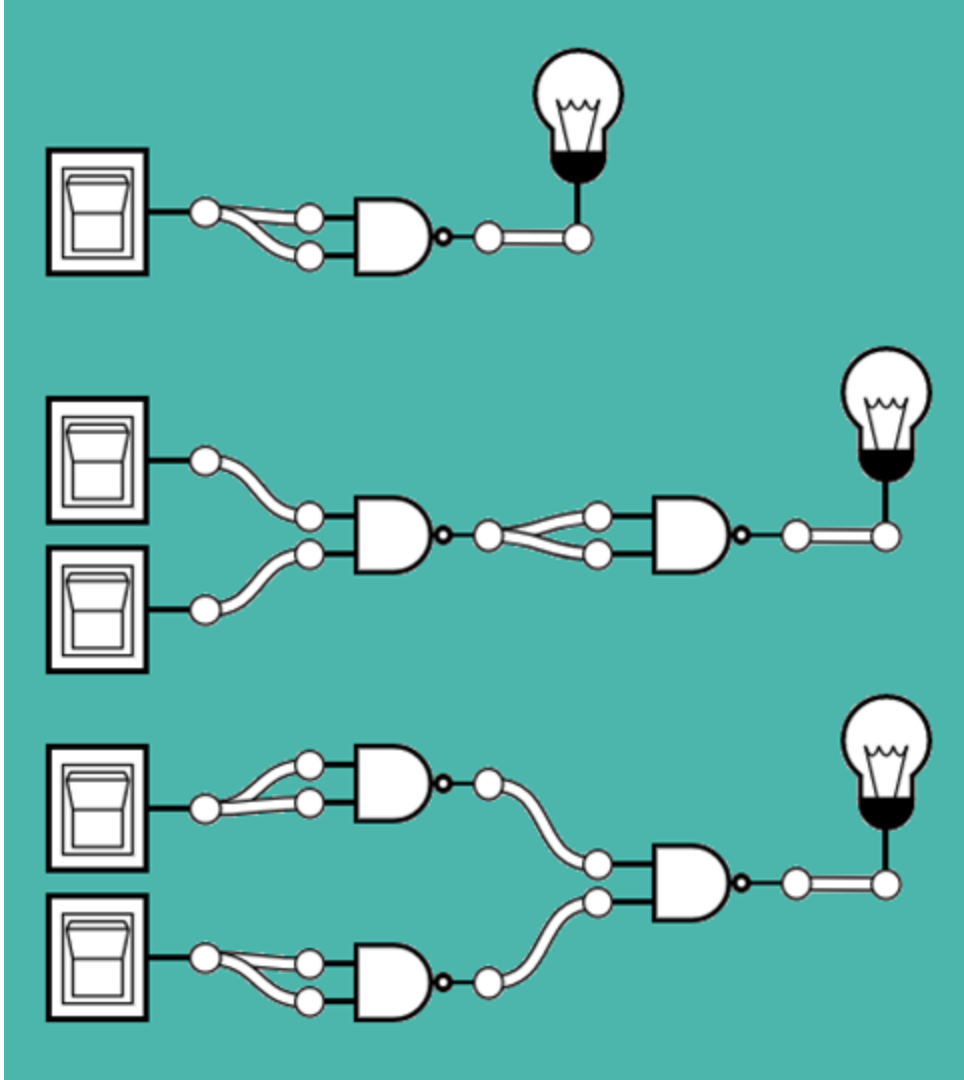
NOR (Not OR)		
InA	InB	Out
0	0	1
0	1	0
1	0	0
1	1	0

Universal Gates

NAND & NOR

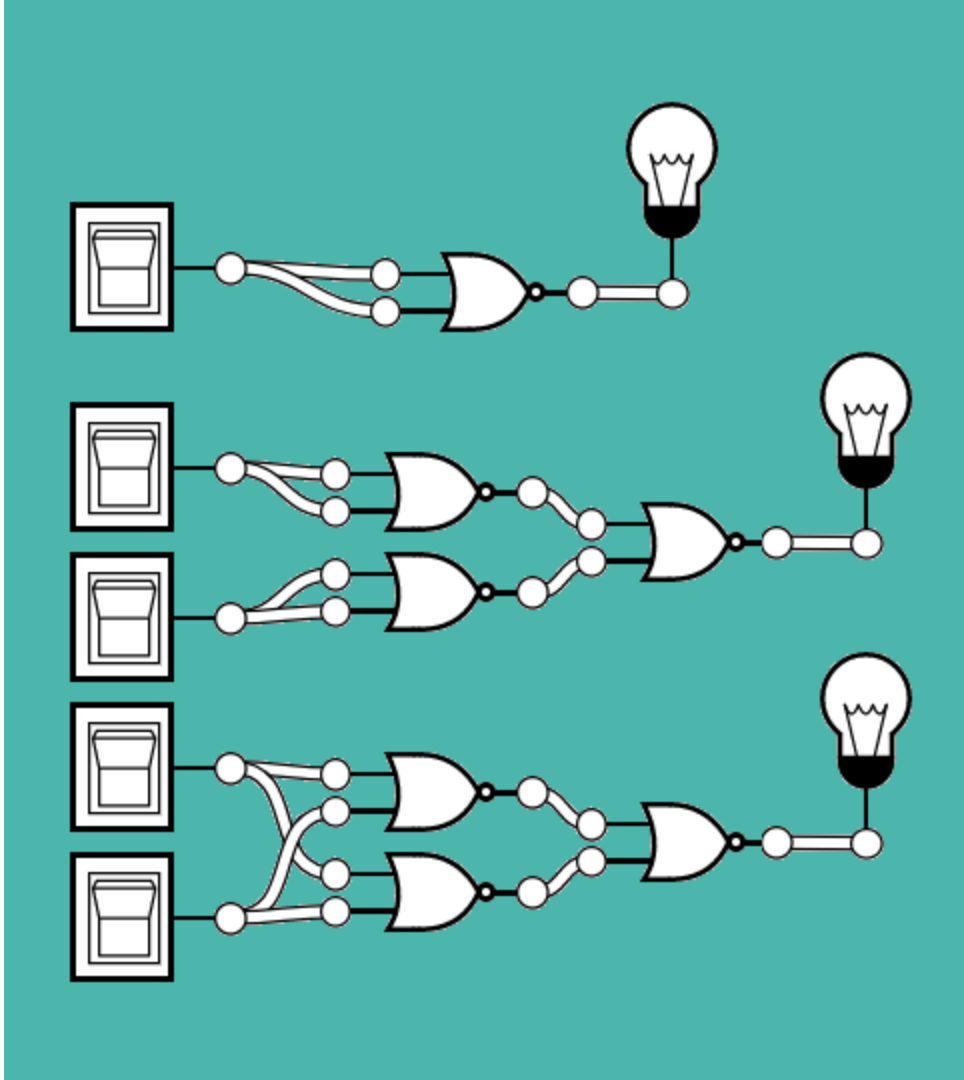
Universal Gate: NAND

NOT, AND, OR



Universal Gate: NOR

NOT, AND, OR



Boolean Algebra

Boolean Algebra

Basic Gates

$$NOT = \overline{A}$$

$$OR = A + B$$

$$AND = A \cdot B$$

Boolean Algebra

Advanced Gates

$$NOR = \overline{A + B}$$

$$XOR = A \cdot \overline{B} + \overline{A} \cdot B \text{ or } A \oplus B$$

$$NAND = \overline{A \cdot B} \text{ or } \overline{A} + \overline{B}$$

Expressions to Circuits

BNAO - Brackets, Not, And, Or
(Like BODMAS / BIDMAS)

Can you create circuits for the following equations using logic.ly/demo?

$$U = \overline{(A \oplus B)}$$

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

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

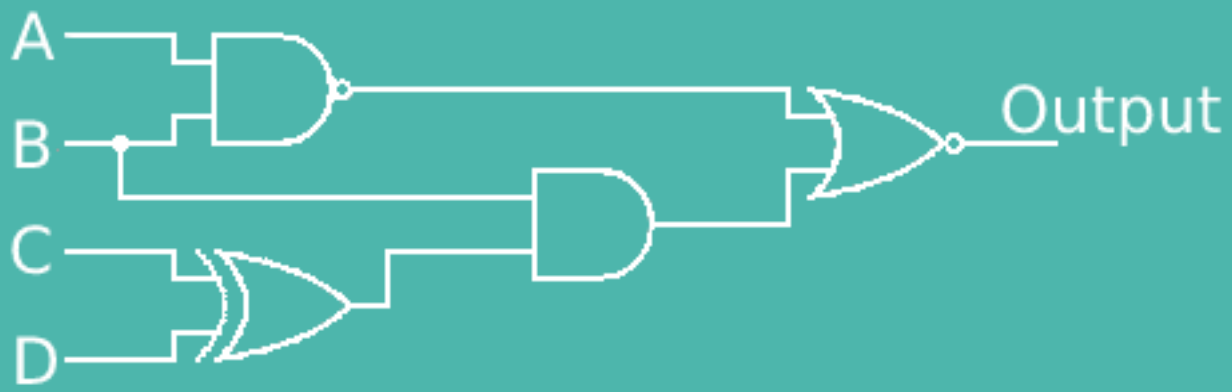
$$X = A + (B \oplus C)$$

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

$$Z = A \bullet B + \overline{(D \bullet E)}$$

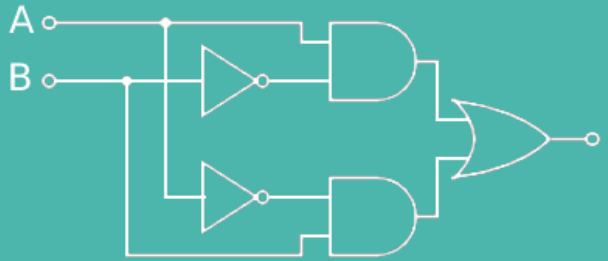
Circuit to Expression

BNAO - Brackets, Not, And, Or
(Like BODMAS / BIDMAS)



Equivalent Circuits

Each circuit has a counterpart. Can you match them up?



Equivalent Expressions

Each expression has a counterpart. Can you match them up?

You may need to do some truth tables.

$\overline{A + B}$

$\overline{A \cdot B}$

$(A \cdot B) + A$

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

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

$A \cdot \overline{A}$

$A + B$

0

$1 \cdot A \cdot B$

A

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

$A \cdot B$

Why is this important?

Computer circuits are made up from these logic gates, they allow computers to perform calculations and store things in memory. Reducing the complexity and size gives us more powerful computers!

Not sure how that’s possible? One of our year 13s made a simulated CPU using logic gates. When you're here in September, I'll show you.