1.4.3 Boolean Algebra

Revision sheet

1 Boolean Algebra

In the following sections, the Boolean algebra have been expressed using the standard Electronics notation. This will be accepted in the exams however questions will be given using the following OCR notation.

1.1 OCR Boolean Algebra Notation

Operator	OCR	Electronics
AND	$A \bigwedge B$	$A \cdot B$
OR	$A \bigvee B$	$A + B$
NOT	$\neg A$	$\overline{A}$
XOR	$A \veebar B$	$A \oplus B$

Table 1: Comparison of OCR and electronics Boolean algebra operators

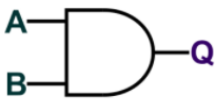
For expressions where the components are only shown in Electronics notation below, they can be constructed from multiple of the OCR symbols shown above.

2 Logic Gates and Truth Tables

For the most part, logic gates can have multiple inputs and one output. Pictured here are the smallest arrangement they come in. In real life, logic gates come in chips, generally bundled into sets of 4 or 6 in one chip.

2.1 AND Gate

This only outputs when all inputs are logic high.



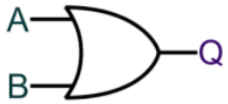
B	A	Q
0	0	0
0	1	0
1	0	0
1	1	1

Table 2: Truth Table for AND Gate

Boolean Expression: $Q = A \cdot B$

2.2 OR Gate

This outputs 1 when either or both are logic high.



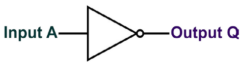
B	A	Q
0	0	0
0	1	1
1	0	1
1	1	1

Table 3: Truth Table for OR Gate

Boolean Expression: $Q = A + B$

2.3 NOT Gate

This is a single input gate. It inverts the signal in.



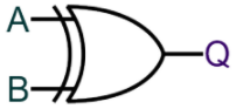
A	Q
0	1
1	0

Table 4: Truth Table for NOT Gate

Boolean Expression: $Q = \overline{A}$

2.4 XOR Gate

This outputs only when input A or B is logic high - eXclusiveOR.



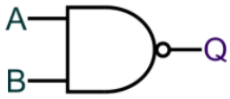
B	A	Q
0	0	0
0	1	1
1	0	1
1	1	0

Table 5: Truth Table for XOR Gate

Boolean Expression: $Q = A \oplus B$

2.5 NAND Gate

This is the same as an AND gate, except it inverts the output.



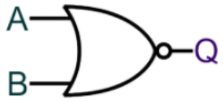
B	A	Q
0	0	1
0	1	1
1	0	1
1	1	0

Table 6: Truth Table for NAND Gate

Boolean Expression: $Q = \overline{A \cdot B}$

2.6 NOR Gate

This only outputs logic high if both inputs are logic 0 - NotOR. This makes it useful for checking if something is 0.



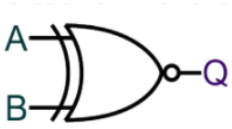
B	A	Q
0	0	1
0	1	0
1	0	0
1	1	0

Table 7: Truth Table for NOR Gate

Boolean Expression: $Q = \overline{A + B}$

2.7 XNOR Gate

This outputs if all inputs are logic 0 or if all inputs are logic 1 - eXclusiveNotOR. This is useful for checking if the inputs are the same.



B	A	Q
0	0	1
0	1	0
1	0	0
1	1	1

Table 8: Truth Table for XNOR Gate

Boolean Expression: $Q = \overline{A \oplus B}$

3 Manually Simplifying Boolean Algebra

3.1 Boolean Identities

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

3.2 Single Variable Rules

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

3.3 Rules

3.3.1 Commutative

The order doesn't matter.

$$A \cdot B = B \cdot A$$
$$A + B = B + A$$

3.3.2 Associative

Some can be re-grouped to make simplification easier.

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

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

3.3.3 Distributive

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

3.3.4 Absorption / Redundancy

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

3.3.5 Double Inversion

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

3.4 deMorgan's Theorem

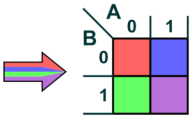
Break the bar, change the sign.

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

4 Karnaugh Maps

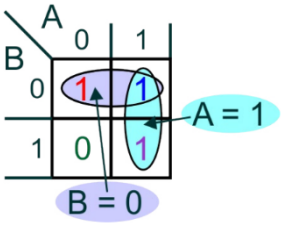
Karnaugh maps can be used to simplify boolean algebra expressions. In their simplest form, they have a grid of two across and two down. This is the size which would be needed to simplify a boolean expression for a single logic gate.

Inputs		Output
B	A	Q
0	0	
0	1	
1	0	
1	1	



4.1 Simplification using a Karnaugh Map

Karnaugh maps allow us to simplify boolean expressions graphically. This works by identifying groups of 2, 4 or 8 neighbouring cells containing logic 1 (these can also wrap around the edge of the table); finding the term common to each groups then combining the terms together using the OR operator. Every cell containing logic 1 must be included in at least one group.

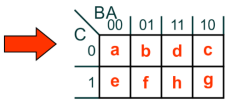


In the example above, the top right cell is in two groups, this is okay as long as each group as at least one of the cells in the group is only in one group. The cells in the blue group are in the A=1 column, this means the term common to them is A. The cells in the purple group are in the B=0 row, this means the term common to them is $\overline{B}$. Combining the blue and purple group gives us $Q = \overline{B} + A$.

4.2 Bigger Karnaugh Maps

Karnaugh maps can have many more inputs. Shown below are three-input and four-input Karnaugh maps.

Inputs			Output
C	B	A	Q
0	0	0	a
0	0	1	b
0	1	0	c
0	1	1	d
1	0	0	e
1	0	1	f
1	1	0	g
1	1	1	h



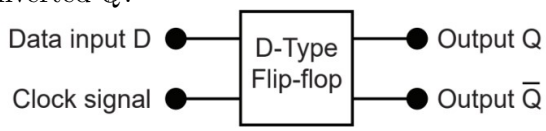
C \ BA	00	01	11	10
0	a	b	d	c
1	e	f	h	g

Inputs				Output
D	C	B	A	Q
0	0	0	0	a
0	0	0	1	b
0	0	1	0	c
0	0	1	1	d
0	1	0	0	e
0	1	0	1	f
0	1	1	0	g
0	1	1	1	h
1	0	0	0	i
1	0	0	1	j
1	0	1	0	k
1	0	1	1	l
1	1	0	0	m
1	1	0	1	n
1	1	1	0	o
1	1	1	1	p

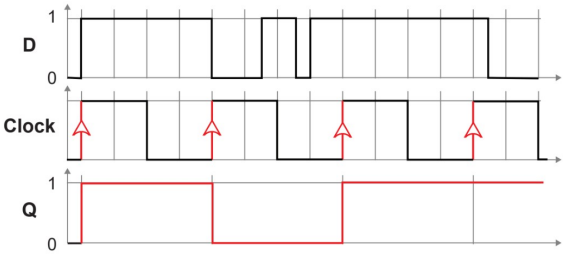
DC \ BA	00	01	11	10
00	a	b	d	c
01	e	f	h	g
11	m	n	p	o
10	i	j	l	k

5 D Flip-Flops

A *D-type flip-flop* is a positive edge-triggered flip-flop (this means that the output will only change when the clock is at a rising edge). A D-type flip-flop has four connections, two input and two output. The inputs are D and CLK, which stands for clock, and the outputs are Q and $\overline{Q}$, which is inverted Q.



Whenever the clock pulse is recieved into the flip flop, whatever the D value is will be moved to the output. This is the same for when D is logic high or logic low. The relationship between these three ports can be shown on a timing diagram (shown below).



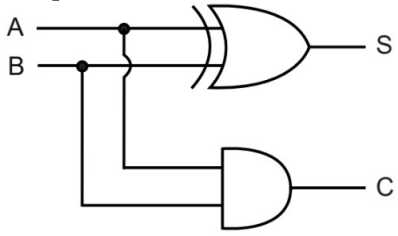
D flip-flops can be used as register memory locations as they will only store the data for one clock cycle.

6 Adders

With the right combination of logic gates, it is possible to output the result of a binary addition or subtraction including the value of any carry bit as a second output. There are two types of adders.

6.1 Half-Adder

This can take an input of two bits and give a two-bit output.

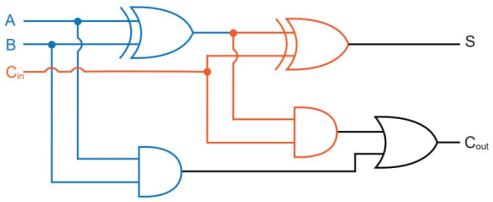


A half adder has the following truth table.

A		B		S	C
0	+	0	=	0	0
0	+	1	=	1	0
1	+	0	=	1	0
1	+	1	=	0	1

6.2 Full-Adder

A full adder combines two half adders together which allows three bits to be added together (A, B and a carry bit C).



A full adder has the following truth table.

A		B		C_{in}		S	C_{out}
0	+	0	+	0	=	0	0
0	+	0	+	1	=	1	0
0	+	1	+	0	=	1	0
0	+	1	+	1	=	0	1
1	+	0	+	0	=	1	0
1	+	0	+	1	=	0	1
1	+	1	+	0	=	0	1
1	+	1	+	1	=	1	1

6.3 Combining Adders

To add numbers which are bigger than one bit together, you will need to combine multiple full adders together. They can be done so as follows.