3 DIGITAL ELECTRONICS.

3.1 The binary numeral system.

The DECIMAL system, or base-10, represents numeric values using 10 symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9.

The BINARY numeral system, or base-2 number system, represents numeric values using two symbols, 0 and 1.

Binary numbers are closely related to digital electronics. **With digital electronics a '1' means that a voltage signal is high and '0' means it is low**. The binary system is used internally by all modern computers.

1 What electronic component can work as a binary switch? ....

When we put together many of them in a single piece of silicon it is called ....

In computing and telecommunications a binary digit is called a _ _ _. It is the basic unit of information in a binary system.

2a The binary system is positional, like the decimal one. To count in binary we put in "ones" from the right. Look at the table on the right and try to figure out the rule. Fill in the missing digits.

Binary	Decimal	Binary	Decimal
0	0	1000	8
1	1	1___	9
10	2	__10	10
11	3	1011	11
100	4	1100	12
10_	5	1__1	13
1_0	6	1110	14
1__	7	1111	15
__00	8	1____	16
1001	9	1____	17

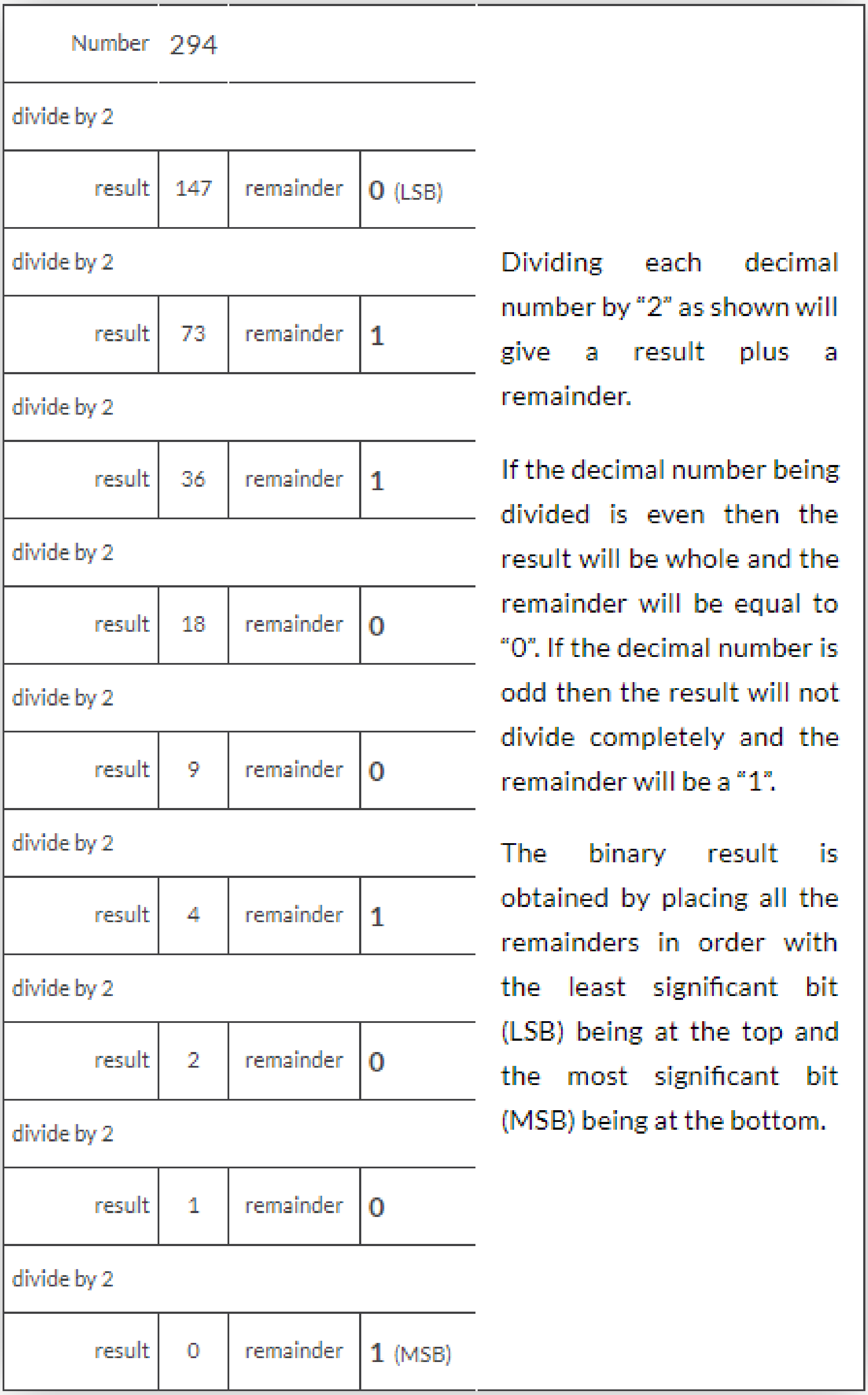
2b It is easy to CONVERT any binary number to decimal because each position has a weight.

Look at the example and convert binary numbers b), c) and d) to decimal. Check the answers with your partner.

	Binary	Binary weight						Decimal
		32	*16*	*8*	*4*	*2*	*1*	
a)	001100	0	0	1	1	0	0	8+4=12
b)	010101							
c)	101010							
d)	100001							

What is the decimal equivalent of one one zero?

So for example. Convert the decimal number 294_{10} into its binary number equivalent.



Number	294		
divide by 2			
result	147	remainder	0 (LSB)
divide by 2			
result	73	remainder	1
divide by 2			
result	36	remainder	1
divide by 2			
result	18	remainder	0
divide by 2			
result	9	remainder	0
divide by 2			
result	4	remainder	1
divide by 2			
result	2	remainder	0
divide by 2			
result	1	remainder	0
divide by 2			
result	0	remainder	1 (MSB)

Dividing each decimal number by "2" as shown will give a result plus a remainder.

If the decimal number being divided is even then the result will be whole and the remainder will be equal to "0". If the decimal number is odd then the result will not divide completely and the remainder will be a "1".

The binary result is obtained by placing all the remainders in order with the least significant bit (LSB) being at the top and the most significant bit (MSB) being at the bottom.

This divide-by-2 decimal to binary conversion technique gives the decimal number 294_{10} an equivalent of 100100110_2 in binary, reading from right to left. This divide-by-2 method will also work for conversion to other number bases.

2c In order to convert from decimal to binary you have to do the inverse process. Convert the following numbers and check your answers with your partner orally.

	Decimal	Binary weight						Binary
		32	*16*	*8*	*4*	*2*	*1*	
a)	41							
b)	20							
c)	33							
d)	63							

Adding binary numbers is a very simple task. As with decimal numbers, you start by adding the bits (digits) from right to left:

Rules	Examples		
		11 1	*11*
0+0 = 0	1001100	1001001	1000111
1+0 = 1	+ 0010010	+ 0011101	+ 1010110
0+1 = 1	---------	---------	---------
1+1 = 10			
1+1+1 = 11	1011110	1100110	10011101

It is also possible to subtract, multiply and divide. This is how electronic devices operate.

3a Add the following numbers. Your teacher will ask some of you to read the additions to all the class. Follow the example and practise reading the procedure to prepare.

```
    1
   001  (1)
 + 101  (4+1=5)
 -----
   110  (4+2=6)
```

One plus one equals zero and I carry one.
One plus zero plus zero equals one.
Zero plus one equals one.
The result is one one zero in binary,
which is six in decimal.

```
a)
   0011
 + 1010
 ------
```

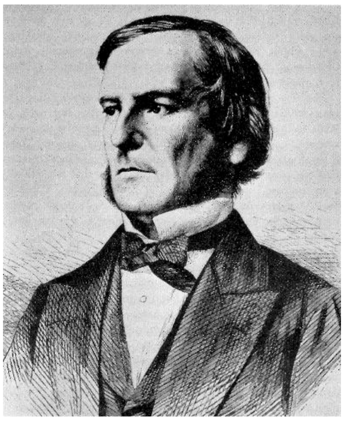
```
b)
   1011
 + 0111
 ------
```

3.2 Boolean logic. Logic gates.

In the last lesson you used BINARY DIGITS to represent NUMERIC VALUES.

BINARY DIGITS can also be used to represent LOGIC STATES like “true” (1) or “false” (0).

BOOLEAN LOGIC (or Boolean algebra) is a complete system for logical mathematical operations. It was developed by the English Mathematician and philosopher George Boole in the 1840s. Boolean logic has many applications in electronics, computer hardware and software, and is the basis of all modern digital electronics.



George Boole (1815-1864)

These are examples of Boolean operations:

1 or 0 = 1	1 and 0 =0	not 0 =1	1 and 1= 1	0 or 0 = 0	not 1 = 0

4a Read the text about Boolean operation representation and fill in the table with the expressions below.

Boolean algebra is based on these logical operations: conjunction x ∧ y (AND), disjunction x ∨ y (OR), and complement or negation ¬x (NOT).

In electronics, the AND is represented as a multiplication, the OR is represented as an addition, and the NOT is represented with an overbar

General	*Maths*	*Electronics*
a AND b		
a OR b		
NOT a		

a ∨ b	$\overline{a}$	a ·b	¬a	a + b	a ∧ b

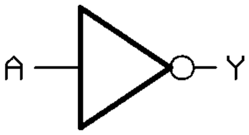
Digital circuits are built from simple on/off switches called **GATES**. These gates have two states: logic high (ON or 1) and logic low (OFF or 0). **TRUTH TABLES** are used to analyse all the possible alternative states of a digital circuit.

You can see the gates symbols on next page. There are two sets of symbols for gates: The traditional ones from America and the new square symbols, a standard by the IEC (International Electrotechnical Commission). You should use the IEC symbols. Anyway the traditional ones are still widely used for simple gates.

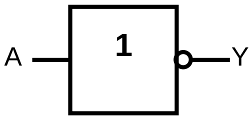
4b Read the gate descriptions and fill in the truth table for each one.

NOT gate: A NOT gate or inverter has just one input. The output is ON if the input is OFF, and OFF if the input is ON.

$\boxed{Y=\overline{A}}$



NOT symbol

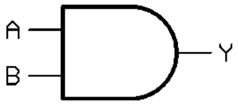


NOT IEC symbol

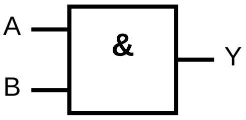
A	Y
0	
1	

AND gate: The output is ON (1) if **both** input signals are ON (1).

$\boxed{Y=A\cdot B}$



AND symbol

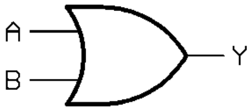


AND IEC symbol

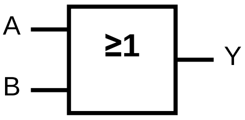
A	B	Y
0	0	
0	1	
1	0	
1	1	

OR gate: The output is ON if **either** or both inputs are ON.

$\boxed{Y=A+B}$



AND symbol

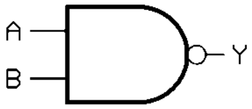


AND IEC symbol

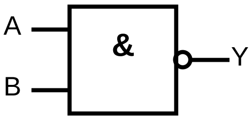
A	B	Y
0	0	
0	1	
1	0	
1	1	

NAND gate: The output is ON **unless both** inputs are ON.

$\boxed{Y=\overline{A\cdot B}}$



NAND symbol

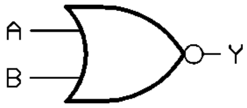


NAND IEC symbol

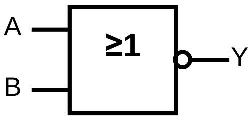
A	B	Y
0	0	
0	1	
1	0	
1	1	

NOR gate: The output is ON if both inputs are OFF.

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



NOR symbol

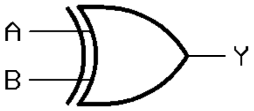


NOR IEC symbol

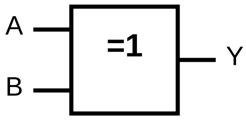
A	B	Y
0	0	
0	1	
1	0	
1	1	

XOR gate: The output is ON if one input is ON and the other is OFF, but will not work if both are ON.

$$\boxed{Y = A \oplus B}$$



XOR symbol



XOR IEC symbol

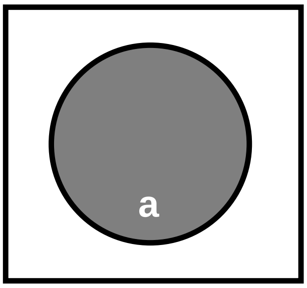
A	B	Y
0	0	
0	1	
1	0	
1	1	

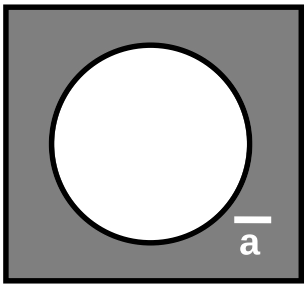
4c Let's test if you remember the IEC symbols and the truth tables. In turns, choose one gate and ask your partner for the function description and the IEC symbol gate. Here you have an example:

Can you explain how a NAND gate works?
What is the symbol of a NAND Gate?

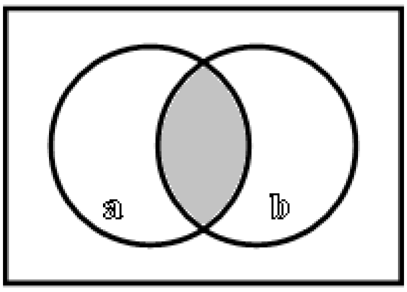
In a NAND gate the output is 0 when both inputs are 1.
It is a square with a "&" symbol inside and with a small circle at the output.

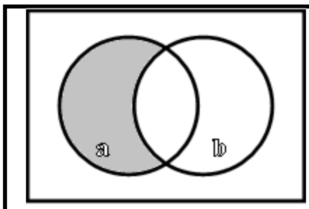
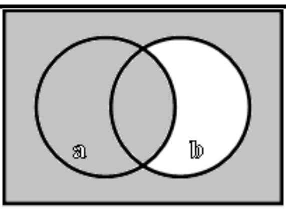
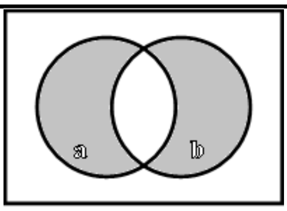
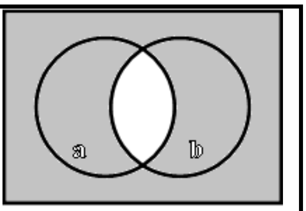
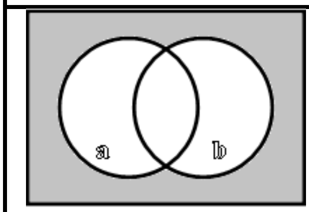
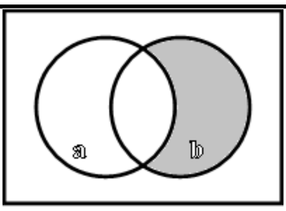
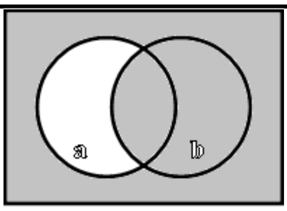
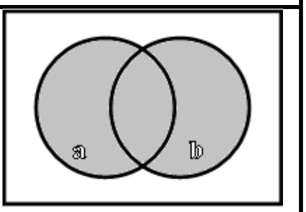
4d It is possible to represent logic functions with Venn diagrams. Look at the examples. Then identify the 8 diagrams as $\overline{a}\cdot b$, $a\cdot\overline{b}$, $a+b$, $\overline{a}+b$, $a+\overline{b}$, $a\oplus b$, $\overline{a\cdot b}$, $\overline{a+b}$.



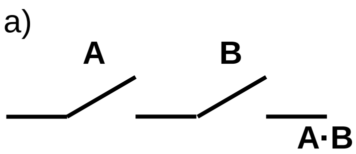


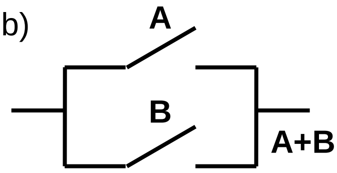
a·b



5 Logic functions can be implemented **electrically with switches** as in these examples.





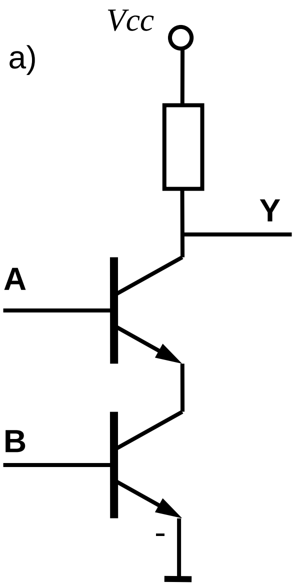
a) AND: The output will only be on when both switches A and B are on.
b) OR: The output will go on if either switch A or B is on.

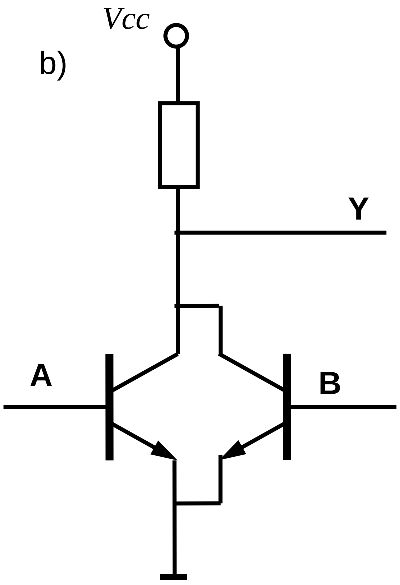
Real electronic gates are implemented **with transistors**. High voltage means 1 and low voltage means 0. These are simplified circuits of a NAND and a NOR gate. Think how the circuits work and fill in the blanks with these words:

parallel	high	low	NAND	series	NOR

In circuit "a" both transistors are connected in _________. The output will go low only when both inputs are ________. So it is a ______ gate.

In circuit "b" both transistors are connected in ________. If either input goes up the output goes _______. So it is a _______ gate.



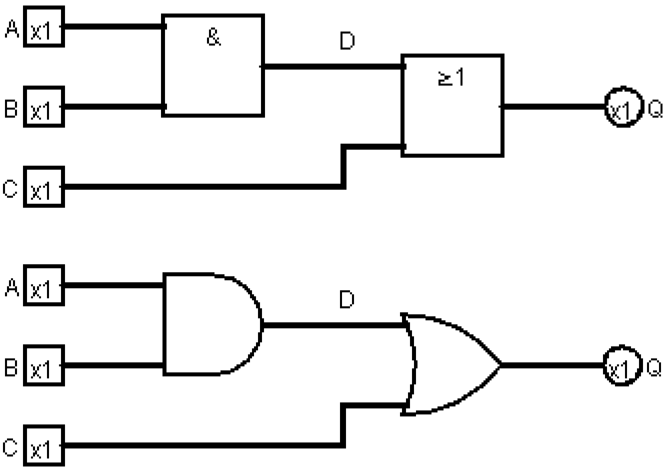


3.3 Logic circuits.

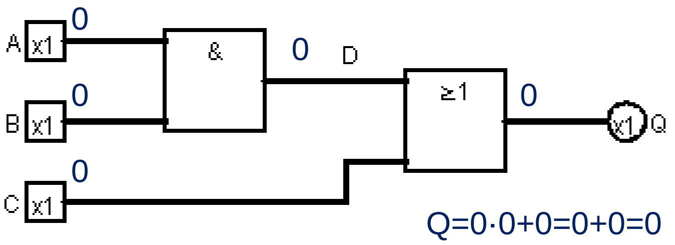
Logic circuits can have many gates, many inputs and more than one output. In this lesson we are going to work with circuits that have a maximum of 3 inputs and 1 output.

6a The diagram below shows a complex logic gate combining two simple gates. There are three inputs and eight possible outcomes. To complete a truth table go row by row. For each combination of input find first D and then Q.

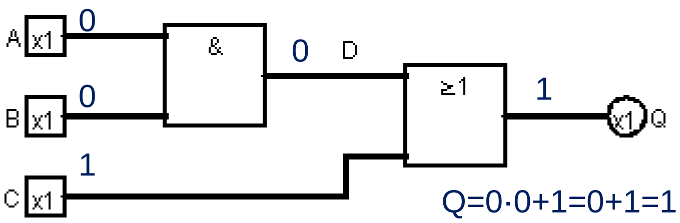
The two first combinations of the truth table are done as an example. Complete the 6 remaining values.



Expression: Q=A·B+C



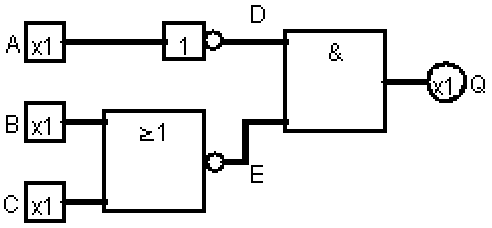
Q=0·0+0=0+0=0



Q=0·0+1=0+1=1

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

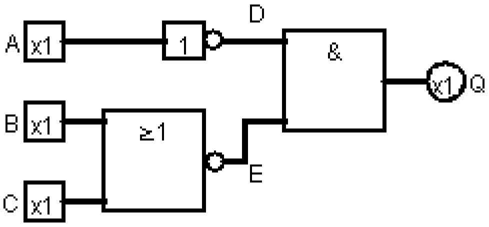
6b For the next circuit find the expression, draw the gate diagram with the traditional symbols and complete the truth table.



IEC diagram

Traditional diagram

Expression:



A	B	C	Q

7 You have to describe orally a logic circuit from the A/B worksheet to your partner. Your partner will describe one for you. Draw the diagram using IEC symbols.

Then you must find the logic expression and fill in the logic table. Finally check results with your partner.

This is an example of descriptions for the circuit in exercise 6b:

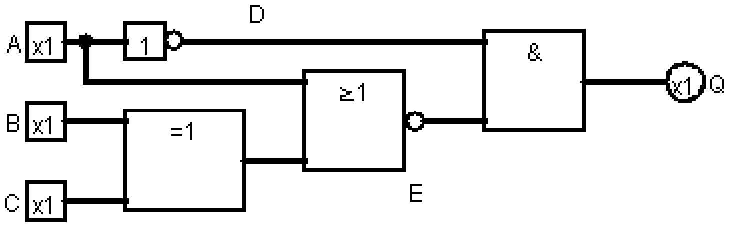
> Input A is fed to an inverter. The output from the inverter is called D. Inputs B and C are fed into a NOR gate, whose output is called E. D and E are fed through an AND gate to output Q.

Circuit:

A	B	C	Q

Expression: Q=

8 For the next circuit find the expression, draw the gate diagram with the traditional symbols and complete the truth table.



Expression:

Traditional diagram:

A	B	C	Q

Look at the example in order to do exercise 9.

DESIGN A LOGIC SYSTEM to control heating like this: In automatic mode heating must be on when it is cold and there is somebody inside. In forced mode heating is always on.

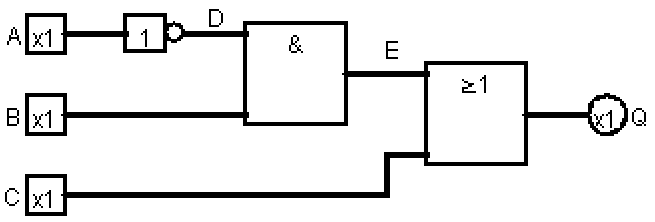
Inputs:
A: temperature (0 cold, 1 warm)
B: presence (0 nobody, 1 somebody)
C: mode (0 automatic, 1 forced)

Output:
Q= heating (0 off, 1 on)

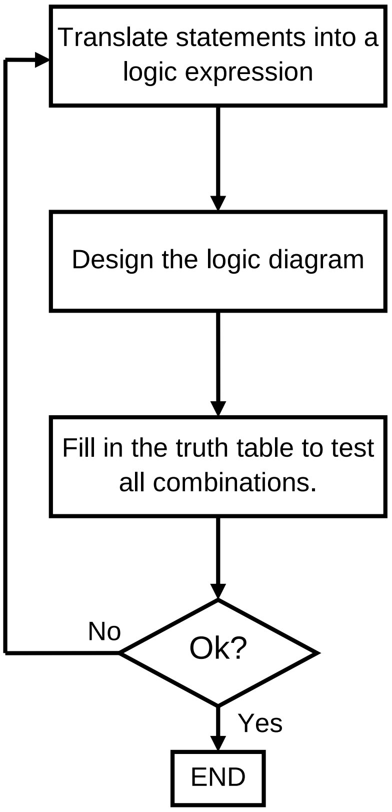
Design process:

Heating= (NOT temperature AND presence) OR mode

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



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


Translate statements into a logic expression
Design the logic diagram
Fill in the truth table to test all combinations.
No
Ok?
Yes
END

Karnaugh Maps

Introduction

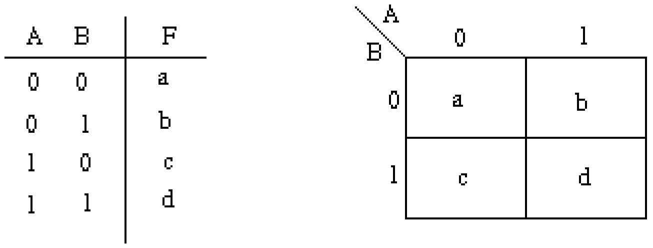
So far we can see that applying Boolean algebra can be awkward in order to simplify expressions. Apart from being laborious (and requiring the remembering all the laws) the method can lead to solutions which, though they appear minimal, are not.

The Karnaugh map provides a simple and straight-forward method of minimising boolean expressions. With the Karnaugh map Boolean expressions having up to four and even six variables can be simplified.

So what is a Karnaugh map?

A Karnaugh map provides a pictorial method of grouping together expressions with common factors and therefore eliminating unwanted variables. The Karnaugh map can also be described as a special arrangement of a truth table.

The diagram below illustrates the correspondence between the Karnaugh map and the truth table for the general case of a two variable problem.



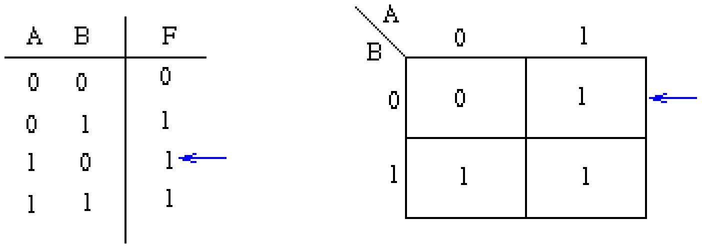
A	B	F
0	0	a
0	1	b
1	0	c
1	1	d

Truth Table.

B \ A	0	1
0	a	b
1	c	d

F.

The values inside the squares are copied from the output column of the truth table, therefore there is one square in the map for every row in the truth table. Around the edge of the Karnaugh map are the values of the two input variable. A is along the top and B is down the left hand side. The diagram below explains this:



A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

Truth Table.

B \ A	0	1
0	0	1
1	1	1

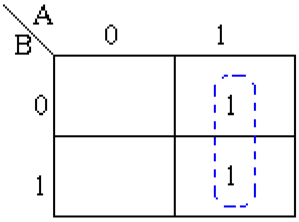
F.

The values around the edge of the map can be thought of as coordinates. So as an example, the square on the top right hand corner of the map in the above diagram has coordinates A=1 and B=0. This square corresponds to the row in the truth table where A=1 and B=0 and F=1. Note that the value in the F column represents a particular function to which the Karnaugh map corresponds.

Examples

Example 1:

Consider the following map. The function plotted is: $Z = f(A,B) = A\bar{B} + AB$



- Note that values of the input variables form the rows and columns. That is the logic values of the variables A and B (with one denoting true form and zero denoting false form) form the head of the rows and columns respectively.
- Bear in mind that the above map is a one dimensional type which can be used to simplify an expression in two variables.
- There is a two-dimensional map that can be used for up to four variables, and a three-dimensional map for up to six variables.

Using algebraic simplification,

$Z = A\bar{B} + AB$

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

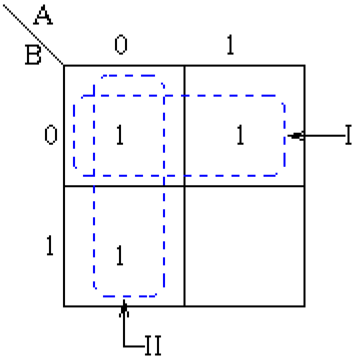
$Z = A$

Variable B becomes redundant due to Boolean Theorem T9a.

Referring to the map above, the two adjacent 1's are grouped together. Through inspection it can be seen that variable B has its true and false form within the group. This eliminates variable B leaving only variable A which only has its true form. The minimised answer therefore is Z = A.

Example 2:

Consider the expression Z = f(A,B) = $\bar{A}\bar{B} + A\bar{B} + \bar{A}B$ plotted on the Karnaugh map:



Pairs of 1's are *grouped* as shown above, and the simplified answer is obtained by using the following steps:

Note that two groups can be formed for the example given above, bearing in mind that the largest rectangular clusters that can be made consist of two 1s. Notice that a 1 can belong to more than one group.
The first group labelled I, consists of two 1s which correspond to A = 0, B = 0 and A = 1, B = 0. Put in another way, all squares in this example that correspond to the area of the map where B = 0 contains 1s, independent of the value of A. So when B = 0 the output is 1. The expression of the output will contain the term $\bar{B}$

For group labelled II corresponds to the area of the map where A = 0. The group can therefore be defined as $\bar{A}$. This implies that when A = 0 the output is 1. The output is therefore 1 whenever B = 0 and A = 0
Hence the simplified answer is Z = $\bar{A} + \bar{B}$

Problems

Minimise the following problems using the Karnaugh maps method.

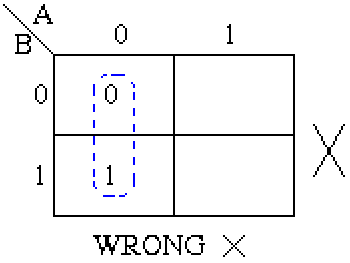
Z = f(A,B,C) = $\bar{A}\bar{B}\bar{C} + \bar{A}B + AB\bar{C} + AC$

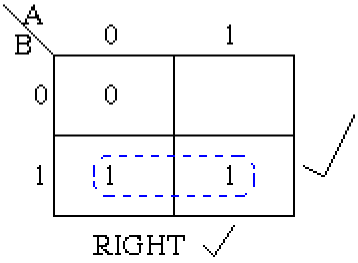
Z = f(A,B,C) = $\bar{A}B + B\bar{C} + BC + A\bar{B}\bar{C}$

Karnaugh Maps - Rules of Simplification

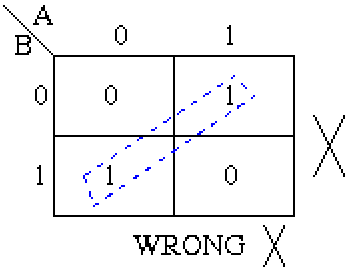
The Karnaugh map uses the following rules for the simplification of expressions by *grouping* together adjacent cells containing *ones*

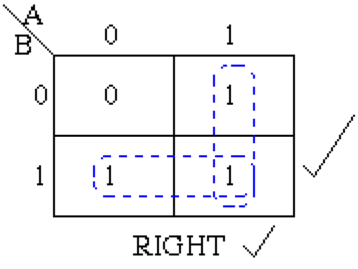
- **Groups may not include any cell containing a zero**



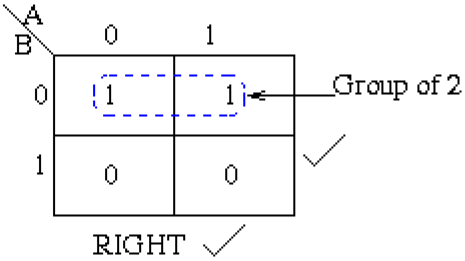


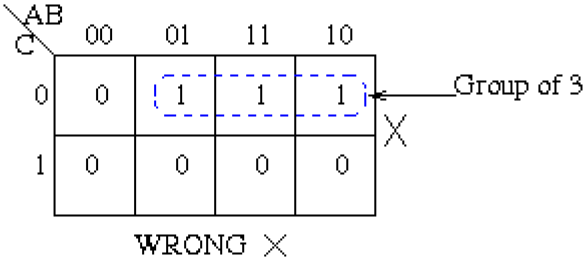
- **Groups may be horizontal or vertical, but not diagonal.**

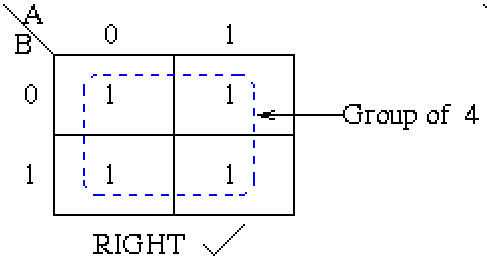


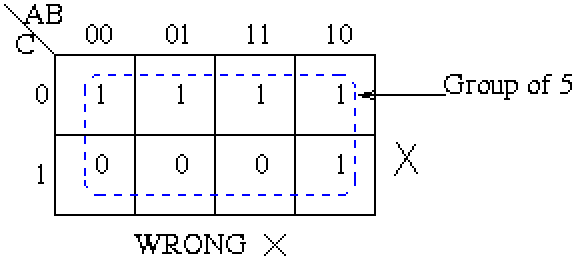


- **Groups must contain 1, 2, 4, 8, or in general 2^n cells.**
  That is if n = 1, a group will contain two 1's since $2^1 = 2$.
  If n = 2, a group will contain four 1's since $2^2 = 4$.

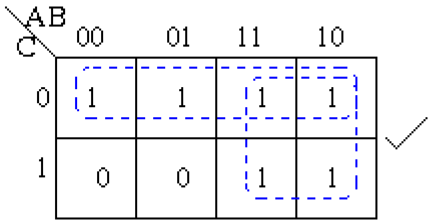






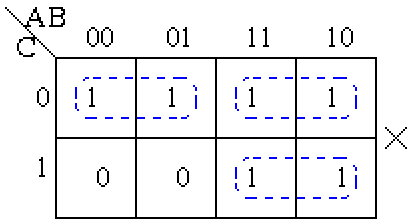


- **Each group should be as large as possible.**



C \ AB	00	01	11	10
0	1	1	1	1
1	0	0	1	1

RIGHT ✓

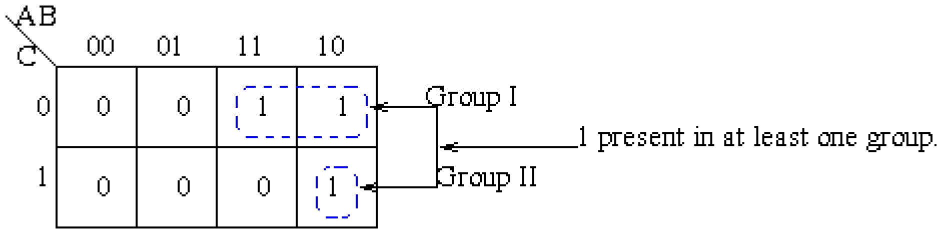


C \ AB	00	01	11	10
0	1	1	1	1
1	0	0	1	1

WRONG ×

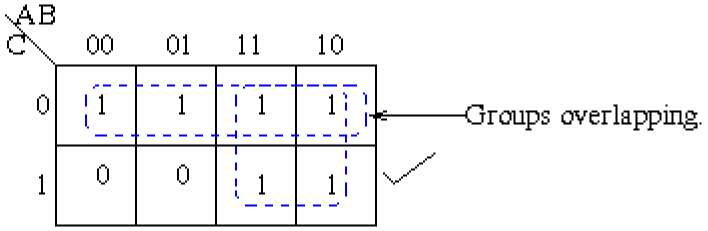
(Note that no Boolean laws broken,
but not sufficiently minimal)

- **Each cell containing a *one* must be in at least one group.**



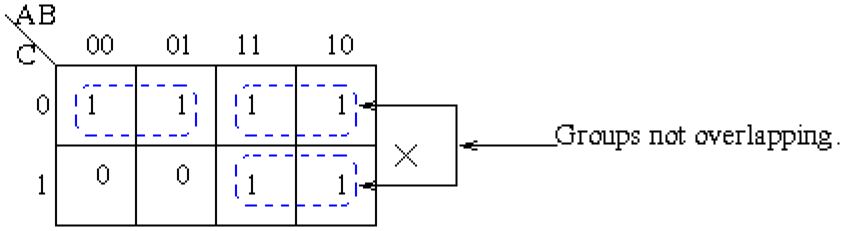
C \ AB	00	01	11	10
0	0	0	1	1
1	0	0	0	1

- **Groups may overlap.**



C \ AB	00	01	11	10
0	1	1	1	1
1	0	0	1	1

RIGHT ✓

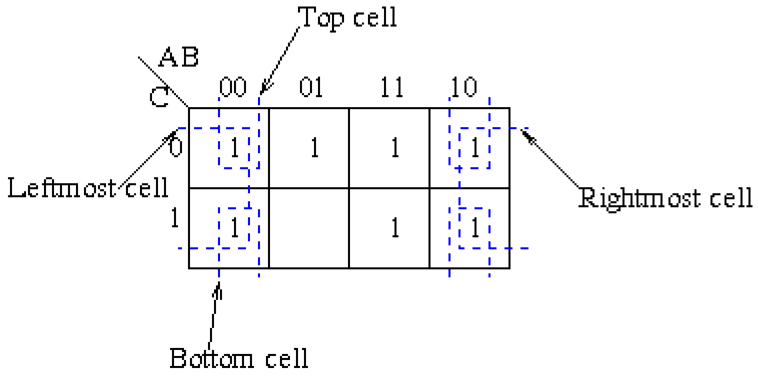


C \ AB	00	01	11	10
0	1	1	1	1
1	0	0	1	1

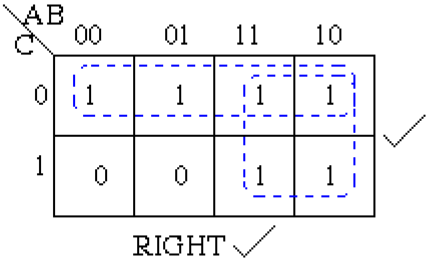
WRONG ×

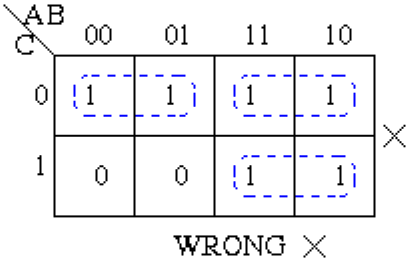
- **Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be**

grouped with the bottom cell.



- **There should be as few groups as possible, as long as this does not contradict any of the previous rules.**





Summmary:

1. No zeros allowed.
2. No diagonals.
3. Only power of 2 number of cells in each group.
4. Groups should be as large as possible.
5. Every one must be in at least one group.
6. Overlapping allowed.
7. Wrap around allowed.
8. Fewest number of groups possible.

9a Design a logic system to control an automatic light like this: The light must come on when it is dark and somebody passes in front of it.

Inputs:
A: presence (0 nobody, 1 somebody)
B: light_sensor (0 dark, 1 light)

Output:
Q= light (0 off, 1 on)

Expression:

Diagram:

A	B	Q
0	0	
0	1	
1	0	
1	1	

9b Design a logic system to control an alarm bell like this: the alarm bell must ring when the alarm switch is on and either the window or the door are opened.

Inputs:
A: window_open(0 closed, 1 open)
B: door_open (0 closed, 1 open)
C: alarm_switch (0 off, 1 on)

Output:
Q= alarm_bell (0 off, 1 on)

Expression:

Diagram:

A	B	C	Q

SELF ASSESSMENT: Before you move on make sure that you can answer yes to all these questions.

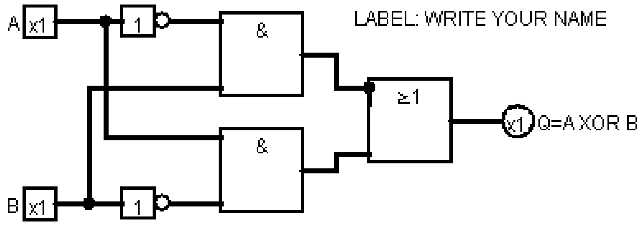
QUESTION	No	More or less	Yes
Can I convert between decimal and binary?			
Can I add binary numbers?			
Can I operate using Boole algebra?			
Can I translate logical expressions to gates?			
Can I obtain truth tables from a logic system?			
Can I use simulators to analyse logic systems?			
Can I design logic circuits in order to solve simple technological problems?			

3.4 Simulation work.

You are going to simulate logic systems using a simulator such as https:// circuitverse.org/simulator

3.4.1 Logic Simulation Basics.

Practice 1: Follow your teacher's instructions to build a XOR gate with AND, OR and NOT gates. Label the final design with your name.



Practice 2: Build and simulate the design you did in activity 9b to control an alarm system. Paste screen shot below

3.4.2 Further design of logic circuits.

Practice 3: Enter this expression: $Q= A\cdot\overline{B\cdot C}+B$ into the simulator and run observing the output.

Practice 4: **CHALLENGE** - Design a detector of prime numbers. The input will consist of four binary digits and the output has to be 1 when the input combination is a prime number (2, 3, 5, 7, 11 or 13).

3.4.3 Adding and visualising.

Practice 5: **Using libraries with integrated circuits.**

Electronic gates are implemented in integrated circuits. The 74XX series of logic gates is built with bipolar transistors.

You have to find out what pins and what circuits to use to build this logic function:

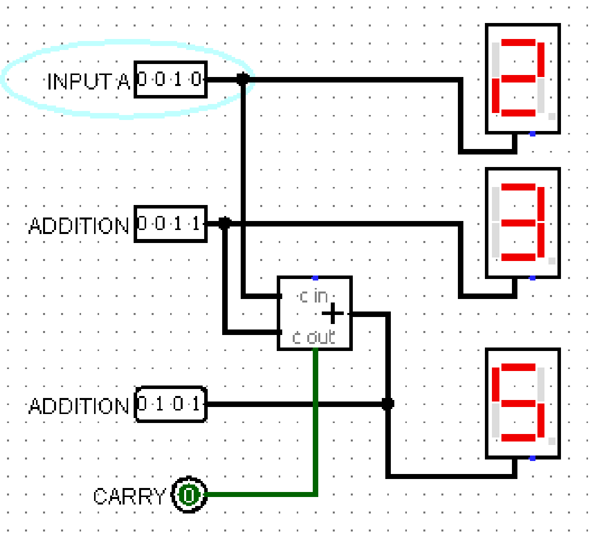
Q = (A NOR B) AND (NOT C)

These are the microchips you may need to use:

- 7400: quad 2-input NAND.
- 7404: hex inverter.
- 7402: quad 2-input NOR gate.
- 7408: quad 2-input AND gate.
- 7432: quad 2-input OR gate.

Practice 6: **Using Adding binary numbers with.** Build the circuit in the picture. You will need:

- Normal inputs and outputs set to 4 bits.
- An *adder* from the *Arithmetic* folder.
- Three *hex digit display* from the *Input/output* folder.



The hexadecimal code has 16 different digits. What are they?