Logical Foundations

Teacher Notes to support Tenderfoot Unit 6: Bits AND Chips – Boolean logic and machine architecture

A taught session establishing ways to represent the behaviour of the three logical constructs of AND, OR and NOT. Introduces truth tables, combinational logic and symbolic representation of circuits. Further gates are introduced (NAND, NOR and XOR) with their algebraic representation before demonstrating that all logical constructs can be reduced to combinations of AND, OR & NOT. Practical activities to introduce the ideas are highlighted throughout.

Preparation required:
Optional sets of batteries / bulbs etc. (common in science departments). Combinational logic exercise.
5 dice and Dotsy scorecards per pair.
Human logic circuit resources plus sticky tape per group (7-9 children)

Concepts and Computational Thinking

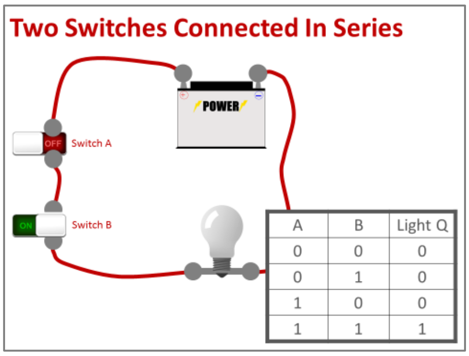
Collectively the concepts we identify as key to Computational Thinking can all be thought of as part of the skills of logical analysis and reasoning, so what better place to start than by looking at the foundations of logic. Our starting point is a mathematician, George Boole, who was born just over 200 years ago. In 1854 he published a very influential book "An Investigation Into the Laws of Thought" which continued "on which are founded the mathematical theories of logic and probability". In it, Boole wrote: "... no general method for the solution of questions … can be established which does not explicitly recognise ... those universal laws of thought which are the basis of all reasoning ..."

The book demonstrated the power of a few simple logical operations. Their application to computing became apparent through the work of Claude Shannon nearly a hundred years later. In 1938 his Masters thesis from MIT formed the basis for "A Symbolic Analysis of Relay and Switching Circuits". In it, he demonstrated that you could construct electrical circuits to represent Boole's logical operations.

Boole's logical constructs can be represented in the behaviour of logic gates. A logic gates has one or more inputs, and one output. They can be made out of anything, but in the first class activity, we use basic electrical components, batteries, bulbs and switches, which are common in school science labs. This makes a good group activity, with pupils working in pairs initially (if equipment makes that possible), then combining as the larger circuits require groups to share equipment.

Consider a simple electrical circuit. The battery is connected to a switch, which connects to a bulb. The bulb connects back to the battery to form a circuit. The behaviour of the bulb is controlled by the switch. We can say when the input (switch) is 0 (off), the output (bulb) is 0 (off), and vice versa. Expressing the behaviour in this fashion, is known as a truth table. The slides that follow provide a narrative to introduce this.

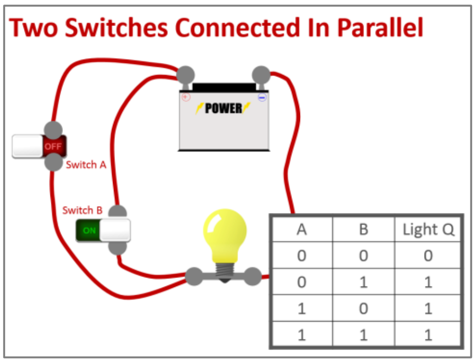
Through these simple activities, constructing circuits, we hope the children will discover behaviour that equates to that of logic gates. Start by connecting two switches 'in series', i.e. daisy chained together. Traditionally we refer to the output as Q. We need to calculate the output for every combination of the switches, labelled A and B. Always complete all combinations of inputs first. To ensure students don't miss any, fill the input columns as if you were counting in binary. If students don't make that connection, point the pattern out. Once created, students can complete the output column (Q).

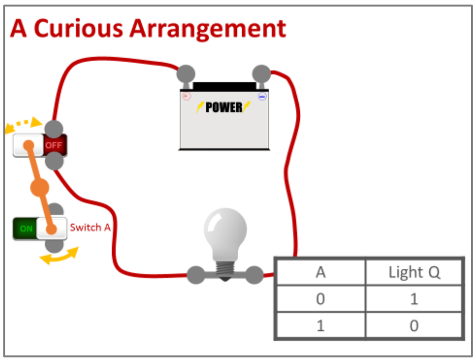


A	B	Light Q
0	0	0
0	1	0
1	0	0
1	1	1

Ask them to explain, in their own words what the truth table shows. We are looking for answers that articulate that both Switch A <u>AND</u> Switch B need to be on before the bulb lights.

Pupils should now rearrange their circuits, so the switches are connected 'in parallel'. Ask the pupils to develop a new truth table. Point out that it still has two input switches, so the same combinations in the truth table need to be tested, but the output will be different. This time when A <u>OR</u> B <u>OR</u> both are on the bulb lights.





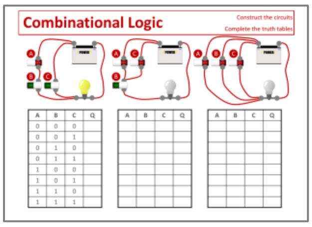
It is unlikely pupils will be able to create this arrangement, but they can imagine the scenario.

Two switches are connected by a pivot, so when one switch slides on, the other will slide off. As the switches are connected there is only one switch, Switch A that we can operate. The truth table therefore only has one input, in two states – on and off. When articulating this behaviour, encourage answers along the lines of 'when the switch is on, the bulb is <u>NOT</u>.

We have developed truth tables for three different circuits. These circuits provide a way of controlling output (lighting a bulb) by altering one or more inputs. They are the three fundamental logic gates

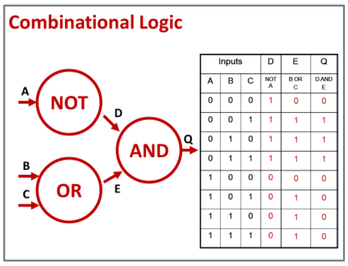
- AND: both inputs must be true (on) for the output to be true (on)
- OR: either input (or both) being true (on) means the output will be true (on) and
- NOT: the output is the opposite of the input.

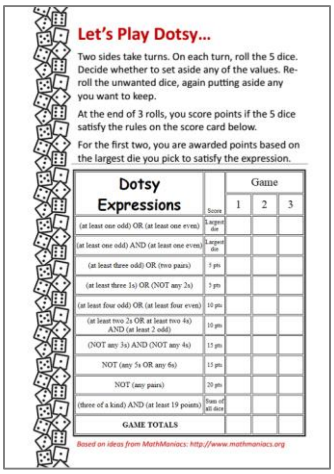
AND, OR and NOT are the three constructs identified by George Boole as the foundation for all logical thought. Any complex logical statement can be expressed in combinations of these constructs.



How would we develop a truth table for circuits with 3 switches? Exactly the same way. Put all the input headings in a line, then fill in the table with incrementing binary numbers. With 3 inputs there are 8 possible combinations to check (binary 000 to 111). Give out the activity sheet and let children construct the circuits and complete the truth tables. When they have the outputs, a sequence of clicks will reveal the answers.

To calculate the output from several gates combined we first label all the inputs, the final output *<u>and</u>* any interim outputs, that feed into the next logic gate. Being systematic avoids making silly errors. We create a truth table as before, and fill in all the input combinations: A, B and C. Each interim value can then be calculated. Ask a student to fill in D, before revealing the answer. Similarly, ask students to complete E and the final output Q. Clicks will reveal each correct answer in turn.





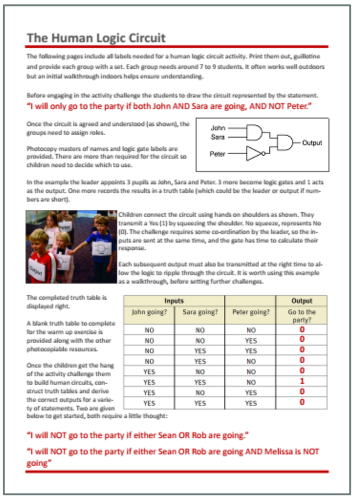
There are many fun ways to reinforce basic understanding of the logical constructs. George Boole says… is played in the same manner as Simon says, but with an emphasis on using statements that include AND, OR and NOT e.g. … put your right elbow on your right knee AND do NOT lift your right foot off the ground.

A reinforcement game, that focuses on interpreting logical expressions is a simple 'poker' dice based game, where each player has 3 rolls of 5 dice, and can set aside any on each roll. At the end of the 3 rolls the player scores points, based on the criteria shown. Although any number can participate, it works best for groups of 2 or 3 players, depending on how many dice are available, or seating arrangements. If larger groups are required, splitting into two teams so players collaborate can also generate good discussion.

Human Logic Circuits

As George Boole articulated all those years ago, logic gates can be used to frame human statements. A short two minute clip from an old Canadian television series, The Essence of the Computer, broadcast in 1983 makes the point well (youtu.be/6wU2NoAtWKM) and provides a good link for students between circuits and logical expressions. How could we express the statement "Jane will only go to the cinema tonight if a good film is on AND she has enough money." in a truth table? What would be the two inputs? Here the inputs (conditions) are a good film and enough money. The output will only be yes (true) IF both conditions are true. The truth table is that for an AND gate. Rather than just using circles to represent gates, each type of logic gate has its own symbol. Two more statements introduce both the OR and NOT symbols.

Human logic circuits are a great way to develop an understanding of combinational logic. Supporting material, based on activities from the Royal Institution are included in the resources. It requires groups of 7 to 9 students. It often works well outdoors but an initial walkthrough indoors helps ensure understanding. Start by studying the statement. The challenge is to represent it as a human circuit. Once it is understood, the groups need to assign roles. Children connect using hands on shoulders as shown in the help sheet.


The Human Logic Circuit

The following pages include all labels needed for a human logic circuit activity. Print them out, guillotine and provide each group with a set. Each group needs around 7 to 9 students. It often works well outdoors but an initial walkthrough indoors helps ensure understanding.

Before engaging in the activity challenge the students to draw the circuit represented by the statement.

"I will only go to the party if both John AND Sara are going, AND NOT Peter."

Once the circuit is agreed and understood (as shown), the groups need to assign roles.

Photocopy masters of names and logic gate labels are provided. There are more than required for the circuit so children need to decide which to use.

In the example the leader appoints 3 pupils as John, Sara and Peter. 3 more become logic gates and 1 acts as the output. One more records the results in a truth table (which could be the leader or output if numbers are short).

Children connect the circuit using hands on shoulders as shown. They transmit a Yes (1) by squeezing the shoulder. No squeeze, represents No (0). The challenge requires some co-ordination by the leader, so the inputs are sent at the same time, and the gate has time to calculate their response.

Each subsequent output must also be transmitted at the right time to allow the logic to ripple through the circuit. It is worth using this example as a walkthrough, before setting further challenges.

The completed truth table is displayed right.

A blank truth table to complete for the warm up exercise is provided along with the other photocopiable resources.

Once the children get the hang of the activity challenge them to build human circuits, construct truth tables and derive the correct outputs for a variety of statements. Two are given below to get started, both require a little thought:

Inputs			Output
John going?	Sara going?	Peter going?	Go to the party?
NO	NO	NO	0
NO	NO	YES	0
NO	YES	YES	0
NO	YES	NO	0
YES	NO	NO	0
YES	YES	NO	1
YES	NO	YES	0
YES	YES	YES	0

"I will NOT go to the party if either Sean OR Rob are going."

"I will NOT go to the party if either Sean OR Rob are going AND Melissa is NOT going"

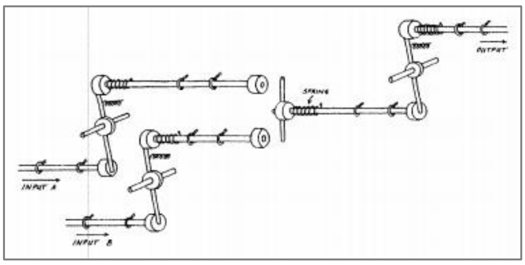
They transmit a Yes (1) by squeezing the shoulder. No squeeze, represents No (0). The challenge requires some co-ordination by the leader, so inputs are sent at the same time, and each gate has time to calculate their response. Each subsequent output must also be transmitted on time to allow the logic to ripple through the circuit. A blank truth table is included for the example activity, but pupils should develop their own for any further challenges.

The activity comes from the website that supports the Royal Institution Christmas Lectures 2008, presented by Chris Bishop. Each of the five lectures is well worth viewing for ideas about how to present many of the fundamental aspects of computing. Lecture 2 'Chips With Everything' looks at the development of several recent technologies and is particularly pertinent as a follow up to this unit. They can be viewed from the RI website (goo.gl/FQCjH4) – each one is around 40 minutes.

Visualising Logic

Logical constructs can be visualised using Venn diagrams. By making links to set theory, work in this area can be linked to, for example, exercises developing smart web search techniques. This can be helpful to illustrate how an AND restricts output, which can be a common misunderstanding. An OR will return elements from both sets. And a NOT will restrict to a given attribute, removing the intersecting cohort.

A logic gate takes one or more inputs, and generates one output. This is an abstract representation of the gate – hiding the particulars of what it is constructed from, and concentrating on its behaviour. We noted that gates can be made from anything. We have used switches, bulbs and humans, but the same behaviour can be built from many components. An example constructs them from dominos. The more complex circuit shown might make a good activity if children can't work out what it does (AND). A video of a slightly different approach (youtu.be/H-53TVR9EOw) also provides an example using marble runs.



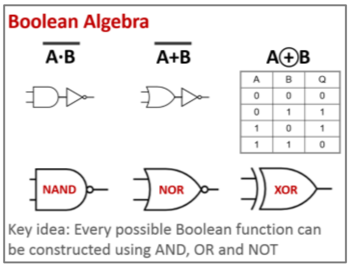
Diagrams demonstrate how we could fashion an OR and a NOT function using simple mechanical components. They come from a wonderful book "The Pattern On The Stone" by Danny Hillis (see further reading). Ask students to develop a truth table for the combinational logic circuit, constructed from 3 NOT gates and an OR gate. Point out that it performs as an AND gate. Only when both inputs are pushed (on), will the output also move forward.

Introducing Boolean Algebra

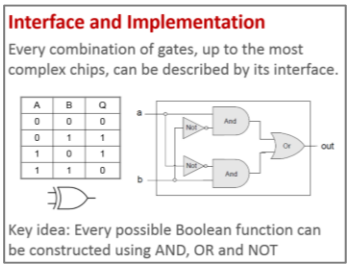
The point is that the behaviour of one logic gate can be reproduced by combining other logic gates. This key idea underpins an understanding of how complex circuits can be constructed. We use it to introduce three new gates, and the idea of Boolean algebra. We are familiar with the 3 core functions, AND, OR and NOT, and their symbols and truth tables. We can use algebraic expressions to indicate the same function. AND is represented as A•B. OR, somewhat confusingly is expressed as A+B, NOT is written $\bar{A}$ to indicate inversion.

Three new gates are introduced via truth tables. The first (opposite of an AND) is known as a NAND. A NOR is the negation of OR. The last symbol is XOR, short for 'exclusive OR'; one OR the other, but not both.

Each expression can also be written in algebraic form as shown. Remember the key idea: every possible Boolean function can be constructed using combinations of AND, OR and NOT. This is easy to see with a NAND, which could be accomplished by connecting the output from an AND gate through a NOT gate, as shown. It's the same with NOR, but what about the XOR? It's not at all clear, either from the symbol, or the algebraic expression how this could be constructed.



Every logical behaviour, beyond a simple AND, OR or NOT, can be described by its interface, that is the inputs it takes, and the corresponding outputs. The XOR interface is described in the truth table. How it could be implemented is shown, using 2 AND, an OR and 2 NOT gates. But once it is implemented, we can forget about the detail, and consider only the interface when designing more complex circuits. Complex circuits are built from less complex chips, but the design is based on the interface, rather than worrying about implementation.

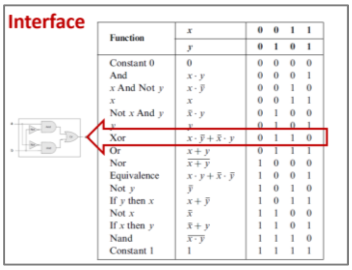


Any chip can be decomposed down to individual AND, OR and NOT components. There is no better example of the relationship between decomposition and abstraction than chip design. Once constructed, the detail of the implementation can be abstracted away. The circuit designers can focus on connecting interfaces, without worrying about the detail of implementation.

To prove that this is possible we can take every possible permutation of output, for the combination of two inputs. This is provided in the table shown. It has 4 columns, one for the output for each combination of input 00,01,10,11, like our truth tables, but arranged as columns rather than rows. Take time to understand this – it is central to the upcoming proof. In the column underneath, we show each truth table output. The first two are shown initially, representing a constant zero output and the output derived from an AND gate. To ensure we consider every possible permutation of output, in the same way as we construct a truth table, we simply list the binary combinations from 0000 to 1111.

We now have every possible interface for two inputs. Each row represents a named function: for example we can see the interface for AND in the second row down. All the others are listed below it in the first column. Finally, the second column expresses the implementation in algebraic form.

If this seems a little complicated take one row and investigate it. You can see, for example how the table lists the interface, and implementation for the XOR we used as a demonstration.



Function	x	0	0	1	1
	y	0	1	0	1
Constant 0	0	0	0	0	0
And	$x \cdot y$	0	0	0	1
x And Not y	$x \cdot \bar{y}$	0	0	1	0
x	x	0	0	1	1
Not x And y	$\bar{x} \cdot y$	0	1	0	0
y	y	0	1	0	1
Xor	$x \cdot \bar{y} + \bar{x} \cdot y$	0	1	1	0
Or	$x + y$	0	1	1	1
Nor	$\overline{x + y}$	1	0	0	0
Equivalence	$x \cdot y + \bar{x} \cdot \bar{y}$	1	0	0	1
Not y	$\bar{y}$	1	0	1	0
If y then x	$x + \bar{y}$	1	0	1	1
Not x	$\bar{x}$	1	1	0	0
If x then y	$\bar{x} + y$	1	1	0	1
Nand	$\overline{x \cdot y}$	1	1	1	0
Constant 1	1	1	1	1	1

You may wonder why we bother with an algebraic representation. When logic circuits are combined, it is often the case that some gates will be redundant, or the circuit could be simplified to use less gates, whilst keeping the interface the same. Boolean algebra makes such simplification easier, as the algebraic expressions can be simplified in much the same way as mathematical equations can also be simplified. This isn't of interest at KS3, but it does feature on courses at A Level.