# Week 1 Lab: Basics of computers

## Objectives

Develop understanding and experience of:

1. Basic logic gates and truth tables with an introduction to Logisim
2. Binary and hexadecimal
3. Combining logic gates to create more complex components

### Part 1 Basic Logic Gates

The tutor will talk through the basic logic gates of AND, NOT, OR, NAND, NOR and XOR.

1. Getting started with Logisim – do this part along with the tutor
   1. Open Logisim and create a circuit that has two input pins labelled A and B that feed into a 2-input AND gate then into a NOT gate and to an output pin labelled X.
   2. Click on the hand icon to test the circuit manually by trying all possible combinations of inputs A and B.
   3. Check the truth table generated by Logisim for your circuit, go to Project then Analyze Circuit and click the Table tab.
   4. Get Logisim to generate the circuit using NAND gates only and create it as a separate circuit.
   5. Save your Logisim file. You are expected to save all your work but won’t always be reminded after this point.

Your tutor will demonstrate two ways in which three inputs (A, B and C) could be combined using two AND gates. Using brackets to show the groupings, the possibilities are

* (A AND B) AND C
* A AND (B AND C)

You will see that the truth tables are the same in both cases.

1. We will use the same approach to compare other combinations. You should work in a pair and choose one of the first two comparisons (labelled i or ii) below. Check with pairs around you to make sure that you don’t all do the same exercise. In your pair, one person should create the circuit for the first bullet point and one the second. You should compare the truth tables. When you have finished, try part iii.
2. Two NAND gates comparing the truth tables for
   * (A NAND B) NAND C
   * A NAND (B NAND C)
3. One AND and one OR gate that is comparing the truth tables for

* (A AND B) OR C
* A AND (B OR C)

1. Using NOT gates before or after an OR gate comparing the truth tables for
   * NOT A OR NOT B
   * NOT (A OR B)

The tutor will lead a discussion of the results so you can see what other pairs found out.

### Part 2 Binary and Hexadecimal

Your tutor will talk through the basics of binary representation of numbers and how hexadecimal relates to binary.

Try the following exercises in pairs before the tutor looks at the solutions with the group. For those who have studied binary before, we are only looking at natural numbers here (that is zero and positive integers).

1. Convert these binary numbers to decimal

010101 21

110000 48

101010 42

1. What is the result (in binary) of adding 1 to the following binary numbers?

101 110

111111 1000000

110011 110100

1. Add the following binary numbers

1001 + 111 1111

10101 + 111 11010

1. Convert these decimal numbers to binary

12 1100

21 10101

45 101001

1. Convert the following numbers from binary to hex

1101000001001110

1011011011000101

1. Convert the following from hex to binary

1a4d

c0f5

1. Convert the following from hex to decimal

ff

a1

1. What is the result (in hex) of adding 1 to the following hexadecimal numbers?

ff

abc

ef9

### Part 3 Combining logic gates to create components

The tutor will recap on the addition of binary numbers giving all the outcomes for adding two single-bit numbers. These outcomes can be considered as a truth table with two outputs, one for the sum and one for the carry bit. You will consider what gates can be used to give the two outputs needed. The component that does this is called a half adder.

1. Based on the gates agreed in the discussion, create a half-adder using Logisim. There should be two input pins, labelled A and B and two output pins, labelled S (sum) and C (carry).
2. To add longer numbers, we need to be able to include the carried bit into the calculation, so we need an adder that can take three bits of input, known as a full adder.

Create a copy of your half adder and try to extend to a full adder. There will now be three inputs, A, B and C\_In and two outputs, S and C\_Out. The full adder can be created by combining two half-adders and an additional OR gate. Spend a few minutes working in small groups to consider what the truth table should look like and how you could try to create it.

The tutor will talk through some solutions for the half and full adders before explaining the concept of a multiplexer (sometimes called a selector).

1. Following the tutor’s introduction, create a multiplexer that selects one of two 1-bit data signals (D0 and D1) according to the value on a 1-bit select signal (S). In summary, the multiplexer will output 1 when (D0 is 1 AND S is 0) OR (D1 is 1 AND S is 1), otherwise the multiplexer will output zero. You should be able to do this using AND, NAND, NOT, OR or XOR gates and it may help to start with the parts in brackets.
2. Create a copy of your multiplexer and extend it to have four separate input bits (D0, D1, D2, D3) and a 2-bit selector signal (input as two 1-bit signals, S1 and S0). Spend a few minutes discussing this in pairs and trying it out before the tutor reviews the work. You might want to use gates with more than two inputs for this task.

The tutor will discuss some solutions to the exercises.