KS4 – Computer systems

**Enjoyed teaching these lessons? Found a mistake? Share feedback at** [**the-cc.io/feedback**](http://the-cc.io/feedback)**.**

## Unit introduction

In this unit, learners will gain an understanding and knowledge of how computer systems work. Starting with the building blocks of the microprocessor — logic gates — learners will discover how a computer system works and executes instructions.

The term ‘note taking’ is used throughout this unit, how this is applied is left to your judgement. Some computing teachers use exercise books or paper folders with learners, some use files on the computer, and some use online note-taking software. You should adapt the lesson to your preferred way of working.

Ideally, learners would have prior knowledge of the binary number system and programming in a high-level language.

## Overview of lessons

| **Lesson** | **Brief overview** | **Learning objectives** |
| --- | --- | --- |
| Computer systems and system software | This is the first lesson in the ‘Computer systems’ unit. It will introduce learners to the two types of computer systems: general purpose and embedded systems. They will explore the characteristics of these systems, and will learn how to identify embedded systems through practical activities.  Following this, learners will discover the need for system software to facilitate communication between software and hardware in computer systems. You will explain the role of an operating system, and learners will match each description to the component, to help them with exam questions. | * Compare embedded and general purpose computer systems * Describe the role of system software as part of a computer system * Explore the role of the operating system and utility software |
| Introduction to the CPU | In this lesson the learners are introduced to the CPU and von Neumann architecture. They will learn about the individual components of the CPU and their roles in computation. Learners will find out about von Neumann and his theories that form the basis of modern computer architecture. | * Describe the basic components of the CPU * Describe the roles and purpose of each component of the CPU in computation |
| The FDE cycle | In this lesson, the learners’ knowledge about the components that make up the CPU and main memory will be furthered with the introduction of the fetch-decode-execute cycle (FDE). Learners will be able to connect the parts of the CPU to their role in executing instructions visually. | * Explain how the fetch-decode-execute cycle works by describing what happens at each stage * Describe the role of each part of the CPU as part of the fetch-decode-execute cycle |
| Main memory | In this lesson learners will be introduced to main memory, RAM and ROM, as well as cache. This lesson builds on the core knowledge from the previous lesson about CPU components. | * Describe the characteristics of RAM and ROM * Explain the role of main memory as part of a computer system * Define cache memory * Describe the role of cache in a computer system |
| Secondary storage | In this lesson, learners will be introduced to secondary storage and take an in-depth look at solid-state storage. You will guide them to discover the need for secondary storage, through assessing the devices they have learnt about already. By the end of the lesson, learners will be able to explain how solid-state storage stores data, and describe the advantages and disadvantages of those devices. | * Explain why a computer system needs secondary storage * State the different types of secondary storage and describe their functional characteristics * State how solid-state memory works and describe its characteristics |
| Optical and magnetic storage | This lesson builds on from the first secondary storage lesson; it involves exploring optical and magnetic storage devices. Learners will need to be aware of how each type of storage operates, and to explain how data is written and read from each device. The last activity has them synthesising the characteristics of the storage mediums to compare them. Using a combination of their learning, and a reference table, learners will actively rank the mediums in each of the key areas of comparison. | * Explain how optical and magnetic memory stores data in the form of binary * Describe how data is read from and written to optical and magnetic memory * Apply knowledge of storage devices to compare the three mediums of storage |
| Selecting a storage device | This is the last of three lessons on secondary storage. The previous two lessons have equipped the learners with the knowledge they need to systematically select and justify a device for a given use. This lesson will model that process for the learners, and then lead to them completing the full process on their own. The second half of the lesson explores the limits of physical storage and how cloud storage can fill the gaps. Learners will examine cloud storage and answer GCSE-style questions on the impacts of cloud storage. | * Apply the knowledge of storage devices to recommend an appropriate device * Describe the limitations of secondary storage * Explain the definition of ‘cloud storage’ and describe the characteristics of cloud storage |
| Computer specifications | This lesson will teach your learners how to evaluate a computer based on its specifications. They will discover the factors that limit a CPU’s performance: clock speed, cache, and the number of cores.  Learners will then use a computer component website to build computer systems of their own. This activity is designed to put their understanding of computer systems into a real-world context. Selecting components to a budget will force learners to make compromises and to examine the importance of the components. | * Explore the factors that impact a CPU’s performance * Select components to create a computer system * Evaluate a computer’s suitability for a given task |
| Computer systems quiz | The beginning of this unit has introduced the essential components of computer systems to your learners. This lesson offers them an opportunity to revise that knowledge while engaging in a fun software project.  The learners will be using the questions they have produced in the previous plenaries to make a quiz they can use for revision. The quiz project serves as a fun way for them to revisit the different components, but the project will also help them with the next few lessons. After this, you will be looking at how computers execute programs; the quiz serves as a knowledge activator for that learning. | * Revise computer systems content covered so far * Design and implement a software project |
| Logic gates | In this lesson, learners will discover logic gates — the building blocks of processors at the heart of a computer system. Through the activities they will build an understanding of how logic gates are used to address real-world problems. | * Discover the logic gates AND, NOT, and OR, including their symbols and truth tables * Learn how logic gates are used in carrying out computation * Design a logical circuit, combining logic gates to solve a problem |
| Logic problems | In this lesson, learners will be introduced to the concept of three-input logic problems, and will be taught how to construct a three-input logic diagram, truth table, and expression. Links will be made to show that computer circuits are made of logic gates, through an explanation of how you can use logic gates to construct a binary adder. | * Construct truth tables for a three-input logic circuit * Write a Boolean expression to describe a logical circuit * Describe how combinations of logic gates can perform mathematical operations |
| Assembly language programming I | This is the final set of lessons for the ‘Computer systems’ unit, a double lesson in which learners will design and write their own assembly language program. You will build them up to this task, first modelling the translation of a Python program into assembly, and then moving on to analysing and tracing an assembly language program as it runs. Over the course of the lesson, learners will build up a ‘toolbox’ of assembly language commands. | * Determine that assembly language has a 1:1 relationship with machine code * Explain the basic commands in the LMC’s assembly code: INP, OUT, STA, LDA, ADD, SUB, and BRP |
| Assembly language programming II | This final lesson involves one last project: the learners will be given a set of requirements and tasked with first designing a Python solution and then an assembly language program to meet the requirements. There is an assessment for the learners to complete to round off the unit and test their understanding. | * Design and write your own program in assembly language |

## Progression

This unit progresses learners’ knowledge and understanding of computer systems.

Please see the learning graphs for this unit for more information about progression.

## Curriculum links

[**National curriculum links**](https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study)

* Develop their capability, creativity, and knowledge in computer science, digital media, and information technology
* Develop and apply their analytic, problem-solving, design, and computational thinking skills
* Understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to report a range of concerns

## Assessment

### Summative assessment

* Please see the assessment question and answer documents for this unit.

## Subject knowledge

This unit focuses on the components that make up a computer system and their roles in computation. It also teaches learners about the generations of programming languages and how software and hardware work together in computation.

Enhance your subject knowledge to teach this unit through the following training opportunities:

### Online training courses

* [Understanding Computer Systems](http://rpf.io/computersystems)
* [Understanding Maths and Logic in Computer Science](http://rpf.io/mathsandlogic)
* [How Computers Work: Demystifying Computation](http://rpf.io/hcw)

Resources are updated regularly - the latest version is available at: [the-cc.io/curriculum](http://the-cc.io/curriculum).



This resource is licensed by the [Raspberry Pi Foundation](https://www.raspberrypi.org/) under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International license. To view a copy of this license, visit, see [creativecommons.org/licenses/by-nc-sa/4.0/](https://creativecommons.org/licenses/by-nc-sa/4.0/).