Computer Science Year 1

SECTION 1

COMPUTER SYSTEMS A – DATA REPRESENTATION AND COMPUTER STRUCTURE

Computer Architecture and Organisation

Data Storage and Manipulation

INTRODUCTION

Inside today's computers, all information is encoded (represented) in the form of 0s and 1s data. These digits are called **bits**, which are short for binary digits. **Bits** are the basic building blocks of computers (just like letters are the building blocks for words), telling the computers what to do and how to represent data. Patterns of bits combine to represent text, numbers, images and sound. This topic is fundamental to computer science because, at the end of the day, computers are machines that manipulate digital signals, which are either on or off, either 1 or 0. So, regardless of whether you are programming, doing machine learning, just using a computer or engaging in more advanced topics like robotics, you are also ultimately manipulating bits.

At the end of this section, you should be able to:

- Describe information as bit patterns.
- Apply knowledge of browser cache to solve runtime issues (e.g. opening browsers).
- Describe the functions of the parts of the CPU: Arithmetic and Logic Unit (ALU), Control Unit (CU) and registers.
- Understand and explain the control bus, address bus, data bus and the internal clock, Machine Cycle, Fetch-Decode-Execute-Store Cycle, instruction set for a CPU and describe embedded systems.

Key Ideas

- A **bit** is the smallest unit of data.
- 8 bits make a **byte** (8 bits = 1 byte).
- Different types of information (numbers, text, images, etc.) are encoded into binary in different ways. This means the raw data/binary representation needs to be decoded according to its type to make sense as useful information.
- **Boolean logic** is a type of algebra in which results are calculated as either True or False.
- **Buses** are used to connect these components, allowing data and signals to pass between them.
- The **buses** that form the system bus play a key role in the machine cycle, because it is they who ferry around the necessary addresses, data and control signals.

HOW DATA IS REPRESENTED IN A COMPUTER SYSTEM

Data as Bit Pattern Representations

A bit is the smallest unit of data in computing or the digital world. A bit can only be in one of two 'binary' states: '0' or '1. The '0' often signifies 'Off' or 'False' and the '1' signifies 'On' or 'True' as in Figure 1.1.

Fun fact*:** Did you know that the word binary comes from the late Latin word 'bini***' for '***two together***', and before computers and bits were invented, it meant '*dual*' or '*pair*', which is quite fitting since it now means two things that are opposite ('On' and 'Off' /1 and 0).

Fig 1.1 Switch illustrating the two binary states

Eight contiguous bits make one **byte** (8 bits = 1 byte).

A series of 0s and 1s is known as a **bit pattern**.

Representing data as bit patterns involves using sequences of 0s and 1s (bits) to encode different types of data. For example, a wall socket switch, which can be 'On' or 'Off' could be represented by one bit (0 or 1).

Bits can often be found in the real world as labels for switches indicating which direction/state is 'On' (1) and which is 'Off' (0). For example, see the switch of the extension cable shown in Figure 1.2.

Fig 1.2 Extension cable with switch labelled with bits

We will learn how the following types of data are represented in binary (as bits): ***Integers (numbers)***, ***Text***, ***Images***, ***Audio***, ***Videos***, ***Files***, and ***Data for Transmission***. We will discuss each in turn below.

1. Integer Representation

At the basic level, you were taught how to convert an integer (whole number) from one base to another.

When encoding numbers in binary, we are converting from base-10 (decimal) to base-2 (binary). In binary/base-2, the place values for each digit go up in 2s, going from right to left.

For example, the first five place values are:

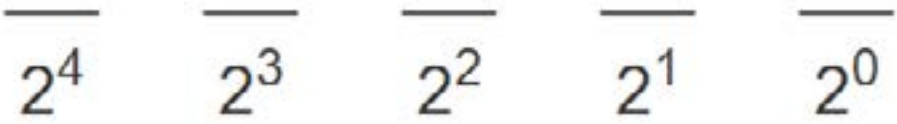


Fig 1.3 Binary representation

a. Decimal to Binary

The process of converting decimal numbers to binary (base-2) notation is shown through two examples. Figure 1.4 converts 13 to binary and Figure 1.5 converts 25 to binary.

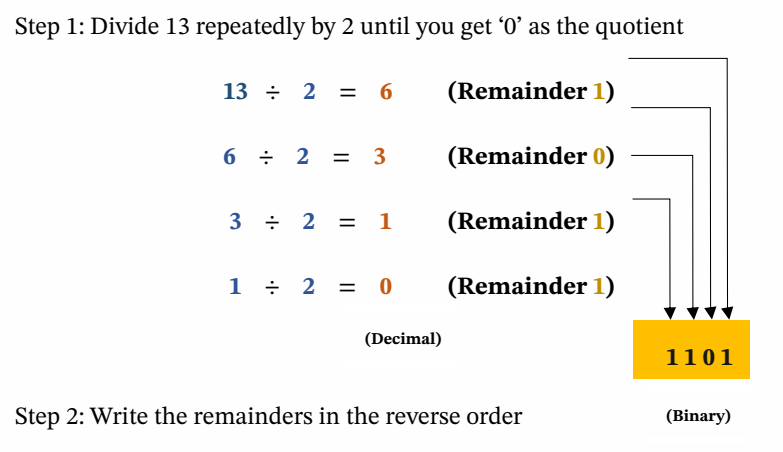


Fig 1.4 Converting 13 to binary

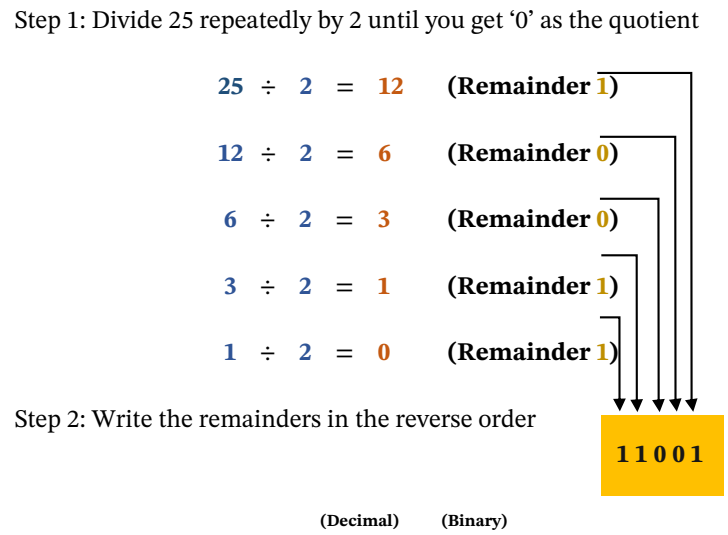


Fig 1.5 Converting 25 to binary

Activity 1.1

1. Count the number of chairs in your class and convert this number to binary.
2. Convert your age to binary.

Leading zeros can be added to make a byte (8 bits), as shown in Table 1.1

Decimal	Binary (8 bits)
0	00000000
1	00000001
2	00000010
3	00000011
4	00000100
5	00000101
6	00000110
7	00000111
8	00001000
9	00001001

Table 1.1: Table representing the 8-bit representation of numbers 0 to 9.

b. Binary to Decimal

Now that we are done with the conversion from decimal to binary, let us look at how to convert from binary to decimal. To convert from binary to decimal, you can use the positional notation system. Each digit in a binary number represents a power of 2, starting from the rightmost digit , then , , and so on.

Note: When counting the binary numbers, you start counting from 0, from the right to the left.

Let us follow the steps below to convert binary numbers to decimals.

Firstly, assign powers of 2 to each digit, starting from the rightmost digit. For example, if you have the binary number 10110, the rightmost digit is 0, followed by 1, then 1, in that order.

$$10110_2 = (1\times2^4) + (0\times2^3) + (1\times2^2) + (1\times2^1) + (0\times2^0)$$

$$= 16 + 0 + 4 + 2 + 0$$

$$= 22$$

$$\mathbf{10110_2} = \mathbf{22_{10}}$$

(Binary) (Decimal

Fig 1.6 Converting 10110 to decimal

Another example is finding the decimal number 1101010_2

$$1101010_2 = (1\times2^6) + (1\times2^5) + (0\times2^4) + (1\times2^3) + (0\times2^2) + (1\times2^1) + (0\times2^0)$$

$$= 64 + 32 + 0 + 8 + 0 + 2 + 0$$

$$= 106$$

$$\mathbf{1101010_2} = \mathbf{106_{10}}$$

(Binary) (Decimal)

Fig 1.7 Converting 1101010 to decimal

Activity 1.2

1. The timekeeper of the 2023 African Games tournament held in Ghana recorded the binary number 111010 as the number of seconds run by an athlete. Calculate the time spent by the athlete, in decimal.
2. Agya Yao's mark in an eating competition was recorded as 101111 in binary. What is the mark in decimal?

2. Text Representation

In the numeric representation we learnt earlier, we saw that numbers can be represented as binary numbers the computer understands. In the same way, text characters also have their own binary representation. These are done using ASCII (American Standard Code for Information Interchange) and Unicode.

Standard ASCII uses a 7-bit code to represent 128 characters. These include 95 printable characters, like letters (A-Z, a-z), digits (0-9), and symbols, along with 33 non-printable characters for things like formatting and control (example carriage return, line feed).

The extended ASCII character set utilises an 8-bit code to represent a wider range of characters. With 8 bits, extended ASCII can accommodate an additional 128 characters and symbols. Extended ASCII is assumed when ASCII is mentioned for the remainder of these notes.

ASCII is a standard that allows computers to understand and communicate with each other. In ASCII, each character (letter, number, and symbol) has its unique code. For example, the letter 'A' is represented by the binary number 01000001 (65), while 'a' is represented by the binary number 1100001 (97). This system helps computers know what characters to display on the screen or how to store them in memory. So, when you type a letter on your keyboard, the computer translates it into its corresponding ASCII code to understand what you are saying. ASCII makes it possible for computers to communicate with each other and makes it easy for us to also interact with them through the text we type. Another encoding scheme is Unicode, which is a more recent standard, developed to overcome the limitations of how many characters ASCII can represent by assigning 16 bits per character. Extended ASCII is a subset of Unicode (comprising its first 256 characters). Unicode aims to provide a unique number for every character, regardless of the platform, programme, or language, creating a global standard for text representation.

Sometimes, an ASCII table can also show the hexadecimal (base-16) and octal (base-8) equivalent of the binary ASCII code as shown in the ASCII table extract in Table 1.2.

Decimal	Hexadecimal	Binary	Octal	Char
48	30	110000	60	0
49	31	110001	61	1
50	32	110010	62	2
51	33	110011	63	3
52	34	110100	64	4
53	35	110101	65	5

Decimal	Hexadecimal	Binary	Octal	Char
54	36	110110	66	6
55	37	110111	67	7
56	38	111000	70	8
57	39	111001	71	9
58	3A	111010	72	:
59	3B	111011	73	;
60	3C	111100	74	<
61	3D	111101	75	=
62	3E	111110	76	>
63	3F	111111	77	?
64	40	1000000	100	@
65	41	1000001	101	A
66	42	1000010	102	B
67	43	1000011	103	C
68	44	1000100	104	D

Table 1.2: Extract from an ASCII table

The ASCII binary codes for letters are shown in Table 1.3.

Letter	ASCII Code (in decimal form)	ASCII Code (Binary)	Letter	ASCII Code	Binary
a	097	01100001	A	065	01000001
b	098	01100010	B	066	01000010
c	099	01100011	C	067	01000011
d	100	01100100	D	068	01000100
e	101	01100101	E	069	01000101
f	102	01100101	F	070	01000110
g	103	01100101	G	071	01000111
h	104	01100101	H	072	01001000
i	105	01101001	I	073	01001001

Letter	ASCII Code (in decimal form)	ASCII Code (Binary)	Letter	ASCII Code	Binary
j	106	01101010	J	074	01001010
k	107	01101011	K	075	01001011
l	108	01101100	L	076	01001100
m	109	01101101	M	077	01001101
n	110	01101110	N	078	01001110
o	111	01101111	O	079	01001111
p	112	01110000	P	080	01010000
q	113	01110001	Q	081	01010001
r	114	01110010	R	082	01010010
s	115	01110011	S	083	01010011
t	116	01110100	T	084	01010100
u	117	01110101	U	085	01010101
v	118	01110110	V	086	01010110
w	119	01110111	W	087	01010111
x	120	01111000	X	088	01011000
y	121	01111001	Y	089	01011001
z	122	01111010	Z	090	01011010

Table 1.3: ASCII-Binary character code set

Activity 1.3

Encode your first name using the ASCII-Binary character code set and show the result to your study mate.

Activity 1.4

What name does this sequence of ASCII-Binary characters encode?

01001010 01101111 01110011 01100101 01110000 01101000 00100000 01000001 01101101 01101111 01100001 01101000

3. Image Representation

When images are represented using bits, we use **pixels**. A **pixel** is the smallest unit of a digital image or display. One way of thinking of an image to be displayed and stored by a computer is as lots of pixels. We then give each pixel a binary code to show its colour. When we put all these binary codes together, we get what is called a **bitmap**, which is an array/grid of binary data representing the colour values of pixels in an image or display. We call this method of representing images on a computer **bitmapped graphics**. With bitmapped graphics, if only 1 bit is used to represent each pixel, the binary code for each pixel is 1 or 0, giving 2 possible colours (such as black and white).

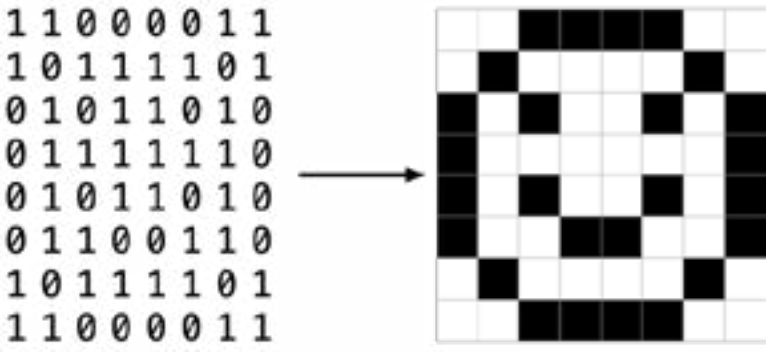


Fig 1.8 Black and white bitmap

From Figure 1.8, '1' is for 'white' and '0' is for 'black'. If more bits are used to represent each pixel, then there are more combinations of binary numbers so more colours can be represented. For example, if two bits are assigned to each pixel, then four colours can be represented using the binary codes 00, 01, 10, and 11.

Activity 1.5

For each of the bitmaps in Figure 1.9, produce the pixel array that encode them as a bitmapped graphics with white represented by '0' and black represented by '1'. In other words, you are producing the array on the left of Figure 1.8, but for the images in Figure 1.9.

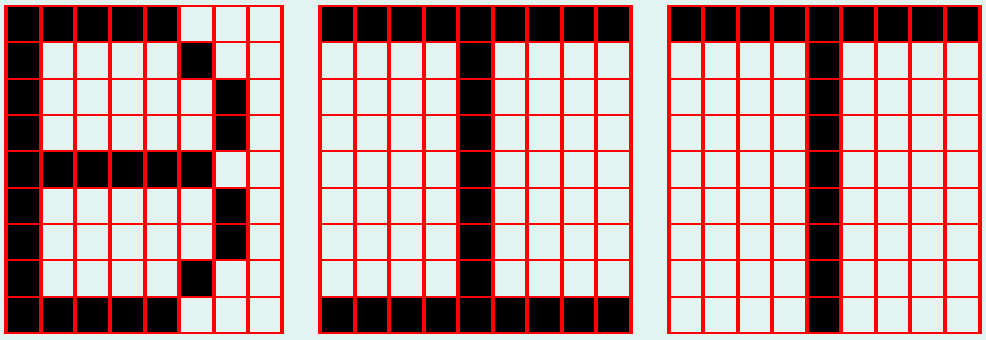


Fig 1.9 Word 'BIT' represented as a bitmap

4. Audio Representation

One method of representing sound on a computer is digital audio. Digital audio is where sound is recorded as a sequence of samples, with each sample encoding the amplitude of the sound wave at a specific time as seen in Figure 1.9. The amplitude value is represented using a series of bits.

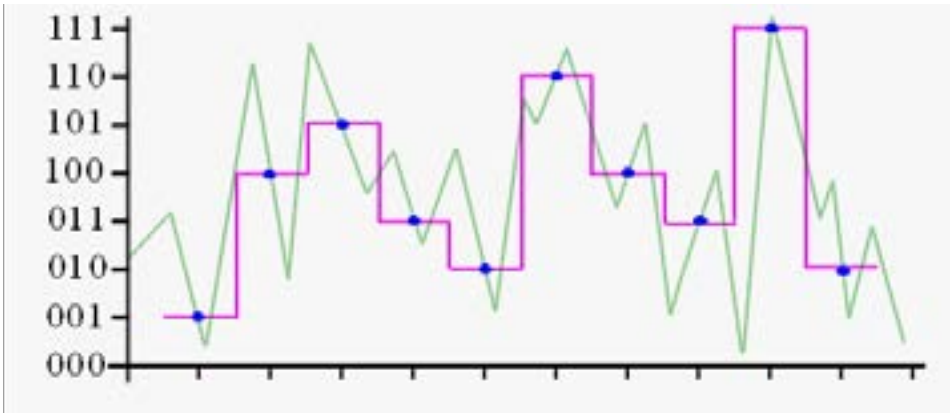


Fig 1.10 Sample audio of bit representation.

Here are examples of how audio can be represented using different bit depths:

a. **8-bit audio**: Each sample is represented by an 8-bit binary number, which can range from 0 to 255 (- 1). This provides relatively low resolution and dynamic range, suitable for some basic audio applications like early video games or simple sound effects.

b. **16-bit audio**: Each sample is represented by a 16-bit binary number, which can range from 0 to 65,535 (- 1). This is the standard format for CD-quality audio and offers higher resolution and dynamic range compared to 8-bit audio.

c. **24-bit audio**: Each sample is represented by a 24-bit binary number, which can range from 0 to 16,777,215 (- 1). 24-bit audio provides even greater resolution and dynamic range, suitable for professional audio recording and mastering, as well as high-fidelity audio playback.

These examples illustrate how the number of bits used to represent each sample affects the quality and fidelity of the digital audio signal.

Activity 1.6

Share, with your study mate, how sound quality would be affected by adjusting the rate at which samples are taken. For example, how would an audio file, where samples are recorded every second, compare to one with samples taken every 0.1s.

Activity 1.7

Find out, using the internet, what sample rate CDs are typically recorded at, exploring what unit of measurement is used and how that corresponds to samples per second. As an extension, you can also find out why that is the standard rate in relation to faithfully recording sound and how humans hear.

5. Video Representation

Digital videos are represented as a series of images (frames), where each frame is encoded using bit patterns to store the colour of each pixel in the frame. Videos are essentially a sequence of images played in rapid succession.

6. File Representation

Files, such as documents, images, videos, and executables are represented as collections of bits. The arrangement of these bits follows specific file formats, which dictate how the data should be interpreted and displayed.

Examples of file formats: JPEG for pictures, DOC or DOCX for Word documents and XLS or XLSX for Excel spread sheets.

7. Data for Transmission

When data is transmitted over networks, the binary representation of the data is broken into same-sized pieces called packets for transmission and then reassembled back into its original form at the destination.

BIT PATTERNS

Representing data as bit patterns allows computers to process, store, and transmit data efficiently. These representations conquer the binary nature of computing, enabling the use of simple electronic circuits that can distinguish between two states (0 and 1) to have the great utility of modern computers. These bit patterns serve as the foundation of modern computing systems, enabling the vast array of applications and services we use in our digital world.

Application of bit patterns in everyday life

All the apps on your smartphone, be it for social media, navigation or productivity, rely on bit patterns for their functionality and data representation. Every time you use a smartphone, tablet, or computer, you are interacting with bit patterns. From browsing the web to sending text messages, the digital data that powers these devices is stored and processed as bit patterns.

Bit patterns are at the core of the digital world, facilitating the efficient storage, processing, and communication of information in numerous aspects of our daily lives.

BOOLEAN LOGIC AND BINARY

Boolean logic is a type of algebra in which results are calculated as either true or false. Boolean logic drives modern digital devices, such as computers. It is used to describe electromagnetically charged memory locations or circuit states in a computer that are either charged (1 or true) or not charged (0 or false).

Bit and their storage

To understand how individual bits are stored and manipulated inside a computer, it is convenient to imagine that the bit 0 represents the value false and the bit 1 represents the value true.

Logic operations

These operations are based on the Boolean logic, which was developed by mathematician George Boole and is widely used in computer science and digital electronics, including applications in programming, digital circuit design, and data processing.

There are three main logic operations. These are: **AND**, **OR** and **NOT**.

Now, let's look at the meaning of these logical operations.

1. AND operation

The AND operation takes two input values, often represented as A and B, and produces an output based on the following rules:

a. If both A and B are 1 (true), the output is 1 (true).

b. Otherwise, if either A or B (or both) is 0 (false), the output is 0 (false).

In simple terms, the AND operation checks whether both input values are true, and if they are, the result is also true; otherwise, the result is false.

The behaviour of logical operation can be summarised into a *truth table*. A ***truth table*** is a breakdown of all the possible truth values returned by a logical expression. Let us consider the truth table for the AND operation in Table 1.5.

Truth table for AND operation:

A (INPUT)	B (INPUT)	A AND B (OUTPUT)
0	0	0
0	1	0
1	0	0
1	1	1

Table 1.5 AND logic truth table

Activity 1.8

Complete the truth table below.

A (Input)	B (Input)	A AND B (Output)
0	0	
0	1	
1	0	
1	1	

Activity 1.9

Based on the output (**A AND B**) of input A and input B, provide the corresponding inputs A and B that satisfy the output.

A (Input)	B (Input)	A AND B (Output)
		1
		0
		0
		1

2. OR Operation

The OR operation takes two input values, also often represented as A and B, and produces an output based on the following rules:

a. If either A or B (or both) is 1 (true), the output is 1 (true).

b. If both A and B are 0 (false), then the output is 0 (false).

In simple terms, the OR operation checks if at least one of the input values is true and if it is, the result is true; otherwise, the result is false.

Table 1.6 shows the truth table for the OR operation.

Truth Table for OR operation:

A (INPUT)	B (INPUT)	A OR B (OUTPUT)
0	0	0
0	1	1
1	0	1
1	1	1

Table 1.6 OR logic truth table

Activity 1.10

Use the truth table of table 1.6 to complete the truth table below.

A (Input)	B (Input)	A OR B (Output)
0	0	
0	1	
1	0	
1	1	

Activity 1.11

Complete the truth table below such that each row of inputs is different.

A (Input)	B (Input)	A OR B (Output)
		1
		1
		1
		0

3. NOT Operation

The NOT operator takes a single input value (often referred to as A) and produces the opposite value as the output.

a. If A is 0 (false), the output is 1 (true).

b. If A is 1 (true), the output is 0 (false).

In simple terms, the NOT operator negates or flips the input value.

Truth Table for NOT operation:

A (Input)	A-NOT / NOT A (Output)
0	1
1	0

Table 1.7 NOT Logic truth table

Activity 1.12

Use the truth table in table 1.7 to complete the table below.

A (Input)	A-NOT (Output)
	0
	1

Activity 1.13

Use the truth table in table 1.7 to complete the table below.

A (Input)	A-NOT (Output)
1	
0	

Understanding logic operations is important for programming and working with digital circuits, as they form the basis for making decisions, comparisons, and computations in computer systems. By using truth tables and practical examples, you can gain an understanding of logic operations and their significance in digital computing.

One example is a truth table for analysing the statement '*The dog is black, and you are the dog's owner*'. This is a statement with two inputs or *propositions*, where the logic operator is the **AND**. Therefore '*the dog is black*' (labelled as A) and '*you are the dog's owner*' (labelled as B) can be combined into the truth table of table 1.8 to summarise the statement.

A (Input)	B (Input)	A AND B (Output)
0	0	0
0	1	0
1	0	0
1	1	1

Table 1.8 Truth table for the given statement above

COMPUTER MEMORY: UNITS OF MEMORY

Units of memory in computing are typically measured in bytes.

Below is a breakdown of some common memory units:

1. Bit (b): the smallest unit of memory, representing a binary digit (0 or 1).
2. Byte (B): a group of 8 bits.
3. Kilobyte (KB): 1 KB is equal to 1024 bytes.
4. Megabyte (MB): 1 MB is equal to 1024 KB or 1,048,576 bytes.
5. Gigabyte (GB): 1 GB is equal to 1024 MB or 1,073,741,824 bytes.

Activity 1.14

Discuss, with your study mates, why 1KB is 1024 bytes and not 1000 bytes.

Activity 1.15

Convert 3MB into both KBs and bytes.

Activity 1.16

Explore how much storage is needed for different types of data by conducting online research or using available storage calculators to investigate the storage requirements for a movie, a mobile phone photo, and a popular video game.

Structure of a computer system

A computer system is structured and comprises a processor/CPU (Central Processing Unit) and memory, input, output, and storage devices. The CPU is the part of the computer where all the calculating, sorting, searching, and decision-making happens. The CPU is discussed in more detail later in this material.

Main or working memory stores data that is currently being used by the CPU. Secondary or backing storage is where saved computer data is stored.

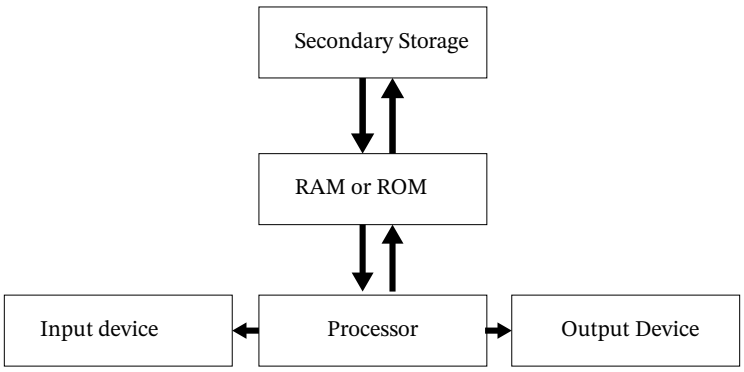


Fig 1.11 Basic block diagram of the structure of a computer system

Input devices are pieces of equipment used to provide data and control signals to the computer. The keyboard and mouse are examples of input devices. Output devices provide data and control signals from the computer to the user. Screens and printers are examples of output devices.

Flip-flop

Having understood what bits are, 'the smallest pieces of data' (either 0 or 1), the question is, how are bits stored in the computer's memory? A flip-flop is a basic electronic circuit used for storing bits. A flip-flop can store only one bit of data. Some types of memory (for example, a type of RAM called SRAM) use flip-flop circuits.

TYPES OF MEMORY: RAM AND ROM AS BIT STORAGE

Below are some types of memory:

1. Random Access Memory

RAM (Random Access Memory) serves as the primary storage location for data that the CPU is actively working on or requires for quick access. This type of working memory holds data and instructions that are currently being processed by the CPU. RAM is directly accessible by the CPU.

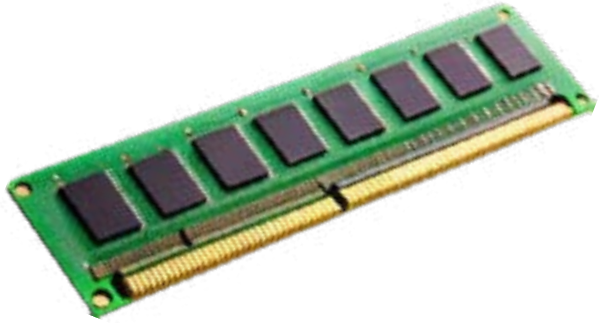


Fig 1.12 A RAM module

RAM has a limited capacity compared to secondary storage devices which will be studied in the next section. The capacity of RAM is typically measured in gigabytes (GB) or terabytes (TB). The amount of RAM in a computer affects its ability to run multiple programmes simultaneously and handle large amounts of data efficiently.

Binary numbers, called memory addresses, are used to identify storage locations in RAM. RAM is described as volatile, meaning it requires power to retain data. When the computer is turned off, the data stored in a RAM is lost.

2. ROM - Read Only Memory

Read-Only Memory (ROM) is another type of working memory in a computer system. Unlike RAM, which is volatile and loses its contents when the power is turned off, ROM is non-volatile, meaning it retains its data even when the power is switched off. ROM is generally read-only, meaning that while data can be read from ROM, it cannot be written to or modified by normal programme execution. ROM is used to store instructions that tell the computer how to perform housekeeping functions such as initialising hardware components or loading the operating system.

Fig 1.13 A ROM module

3. Secondary Memory/Storage

Secondary memory is external, non-volatile memory where data can be stored permanently. Examples include internal/external hard disk drives (HDDs), Pen drives (Flash drives), CDs, and solid-state drives (SSDs). Secondary storage has a much larger capacity than main memory but is slower in terms of access times. Secondary storage is also sometimes termed auxiliary storage.

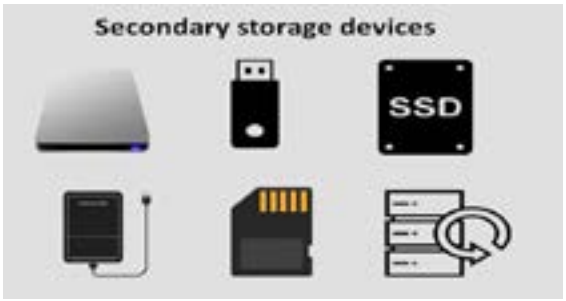


Fig 1.14 Secondary storage devices/media

4. Cache Memory

Cache memory or CPU cache is a small, high-speed memory for temporary data storage. It is located directly on the CPU or close to it. It acts as a buffer between the CPU and RAM, holding frequently used data and instructions that the processor may require next. This reduces the need for frequent slower memory retrievals from RAM which may otherwise keep the CPU waiting.

5. Registers

Registers are the fastest access and smallest capacity storage units. They are located within the CPU itself. They serve as temporary storage areas that store the data, instructions and memory addresses that the CPU is currently processing.

Memory Hierarchy

Memory hierarchy is a hierarchical arrangement of different types of memory in a computer system, organised in levels according to their speed, capacity and cost. The memory hierarchy is designed to optimise data access and storage, ensuring that the computer can efficiently process and retrieve data at various speeds. It consists of several levels, each with different characteristics, purpose and proximity to the CPU. The levels in the memory hierarchy go from the fastest and smallest access time to the slowest and largest capacity.

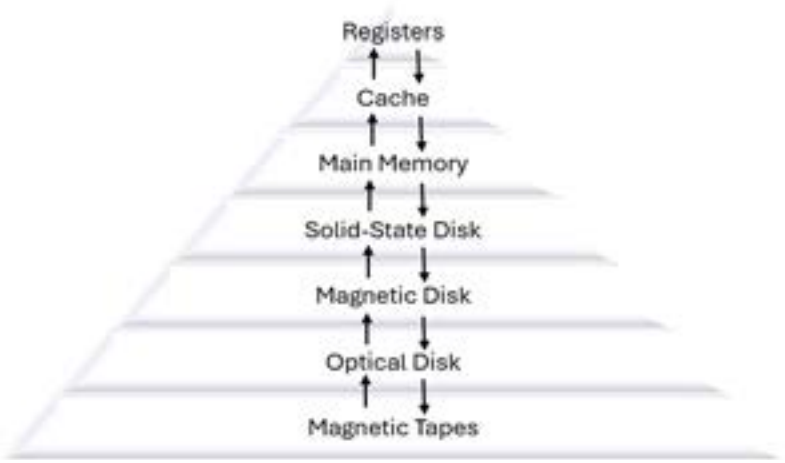


Fig 1.15 Memory hierarchy

Since CPU registers are the fastest to read and write, they are placed at the top of the hierarchy. On the other hand, secondary storage devices like optical drives (CDs and DVDs) are the slowest and largest and therefore occupy the last level in the pyramid. A significant way to increase system performance is minimising how far down the memory hierarchy data needs to be retrieved.

Activity 1.17

Use Figure 1.15 and any search engine of your choice to complete the task below.

a. Expand figure 1.15 into a poster that gives more details about each level.
b. Go online and find out detailed explanation of each level.
c. Prepare PowerPoint slides of your finding.
d. Share your finding with your colleagues.

CACHING

CPU cache is just an example of the more general concept of caching. In computing, a cache is a hardware or software component that stores data so that future requests for that data can be served faster. Another example of caching is browser cache, also known as web cache.

Browser cache/Web cache

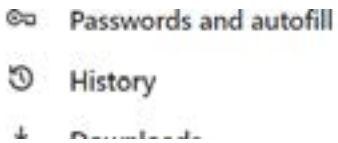
Browser cache is a temporary storage area in RAM or on disk that holds the most recently downloaded web pages. As you jump from web page to web page, the caching of those pages in memory lets you quickly go back to a page without it having to be downloaded again. To ensure that the latest page is displayed, the browser compares the dates of the cached page with the current web page. If the web page has not changed, the cached page is displayed immediately. If the web page has changed, it is downloaded, displayed and cached. When you quit the browser session, the cached pages are stored on disk.

Errors and clearing browser cache

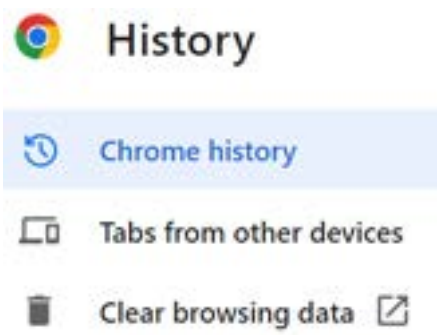
If a user gets a run-time error message when trying to access a particular webpage, this can indicate a corrupt browser cache. The user may also be unable to load the webpage if the website owner has made updates, and the user's browser programme tries to use an old file stored in its cache. A possible solution to these issues is to clear the browser cache.

Below are the instructions for clearing the browser cache in Chrome.

1. Open Chrome and click on the three dots (top right of the window)
2. Click / hover over 'History' to display a submenu.



3. In the submenu, select 'History'.
4. On the left of the 'History' screen, click 'Clear browsing data'.



Activity 1.18

With the aid of the internet, use the steps above to clear the browser cache of the device you are using.

CENTRAL PROCESSING UNIT

The part of a computer that manages all the work with data is called the Central Processing Unit or CPU. It is often just called the 'processor'. The CPU is often referred to as the 'brain' of the computer, as it is responsible for executing instructions and performing calculations that enable various computing tasks. It is also the part of the computer system where data is processed into information. CPUs in computers are small flat chips that can fit in your hand.

They have connecting pins that go into a socket on the motherboard (a large circuit board that hosts all the essential components of a computer, including the CPU). In smartphones, tablets, and other mobile devices, CPUs are even smaller; about half the size of a postage stamp. So, there are different types of CPUs and CPU specifications vary. CPUs in washing machines and smart televisions, for example, will have considerably less processor speed than in a brand new desktop computer.

Below is a picture of a CPU on a motherboard.

Fig 1.16 A picture of a CPU on a motherboard.

Activity 1.19

Write the names of other devices that have CPU in them in your jotter and share with your friends in class.

Components of a CPU

A CPU consists of three main parts: the Control Unit (CU), the Arithmetic and Logic Unit (ALU), and registers. We will look at each in turn, along with other components of the CPU including its cache, clock and bus.

1. Control Unit (CU)

This is the part of the CPU that manages the operations within the CPU. Some of its functions includes:

a. Controls the order in which instructions are executed.

b. Manages the other CPU components, such as the ALU and registers. The CU communicates with these parts using the CPU internal buses (we'll come to those later).

c. Fetches instructions from memory.

d. Decodes those instructions.

e. Fetches the data required by the instructions from memory.

f. Sends the data and instructions to ALU for processing.

g. Outputs the data after processing.

Activity 1.20

Table 1.9 shows real life examples of processing, breaking processes down into their input, processor and output. For example, the first line demonstrates how the process of washing clothes can be broken down into the input of dirty clothes, the processor of a washing machine, and the output of clean clothes. Complete the rest of the table individually, asking your teacher for feedback on your completed table.

Input	Processor	Output
Dirty clothes	Washing machine	Clean clothes
	Garage	
	Fire	
A person with long hair		

Table 1.9: Task on processing in real life

Possible solution to the activity 1.20

A valid completion of table 1.9 is seen in Table 1.10. Notice how each output is the result of some processing being done, by the processor, to the input. Consider whether this also holds for your solution.

Input	Processor	Output
Dirty clothes	Washing machine	Clean clothes
Broken car	Garage	Fixed car
Raw chicken	Fire	Roast chicken
A person with long hair	Hairdressers	A person with short hair

Table 1.10: Valid solution to activity on processing real life

2. Arithmetic and Logic Unit (ALU)

The arithmetic operations manipulate numbers and include addition (+), subtraction (-), multiplication (*) and division (/). The logic operations involve performing comparisons and making decisions, including the logical operations we learnt about the last topic (AND, OR, and NOT), and comparison operations including equal to (=), not equal to (≠), less than (<), greater than (>), less or equal to (≤), and greater or equal to (≥).

Functions:

a. Performs arithmetic operations to process numbers.

b. Performs logical operations to make decisions.

3. Registers

We have already touched on this part of the CPU in the last topic, where we learned the details of computer memory. Recall that registers are the fastest access and smallest capacity storage units, and that they are located within the CPU itself. They serve as temporary storage areas that store the data, instructions and memory addresses that the CPU is currently processing. Different types of registers perform different functions. The names and functions of three of the CPU registers include the following:

a. The **Memory Address Register** (**MAR**) stores the address in memory currently being read from/written to.

b. The **Program Counter (PC)** stores the memory address of the next instruction to be executed.

c. The **Memory Data Register** (**MDR**) holds data being transferred to and from the processor.

4. Cache

Cache memory is a fast random-access memory that temporarily stores a small amount of data and instructions that the CPU is likely to use, so that it can run more efficiently.

Functions:

a. Temporarily store data and instructions for later.

b. Allows faster processing, as the CPU can get the data and instructions directly from the cache instead of having to wait to retrieve them from RAM.

Activity 1.21

1. Use one of your school's computers to find out:
   a. The name and manufacturer of the processor it has.
   b. The quantity of RAM installed.

Possible Solution to the activity 1.21

For a Windows device: click on the Windows Start button, then click on Settings (the gear icon). In the Settings menu, click on 'System', scroll down and click on 'About'. The information you are looking for will be here, e.g. processor = Intel Xeon 6, RAM = 16GB.

5. Clock

The clock is the part of the CPU that sends out regular pulses that keep the CPU and its related components in step with one another. The CPU clock speed, also

known as clock rate or clock frequency, is measured in Gigahertz (GHz) and indicates the number of cycles per second that the CPU's clock executes.

Each cycle of the CPU clock represents one basic unit of work that the CPU can perform, such as fetching, decoding, or executing an instruction. A higher clock speed means the CPU can execute more cycles per second, leading to faster processing of instructions and improved performance.

Function: Generates electrical pulses at a specific frequency that determines how quickly the CPU can process.

6. System Bus

A bus is a communication channel/*series of lines that connect the processor to another part of the computer's architecture.*

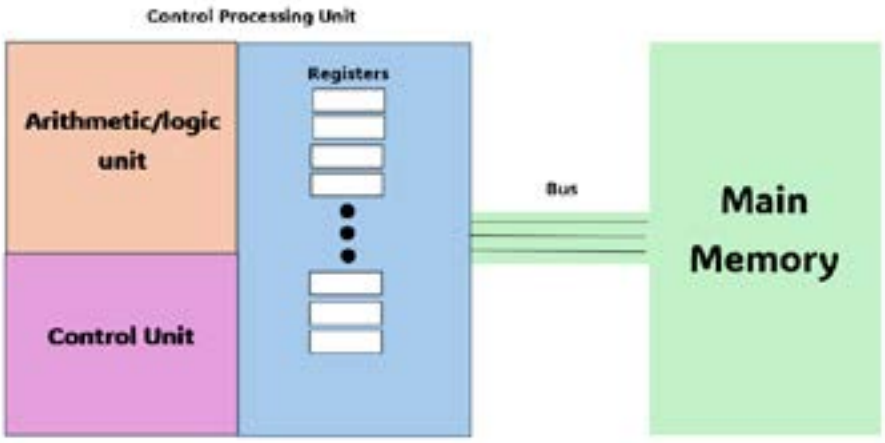


Fig 1.17 CPU is linked to main memory via buses

The processor also has a set of internal buses that allow signals and data to pass between its different components.

The three types of buses are address bus, data bus and control bus. These buses are collectively called the system bus. Functions include:

a. The address bus sends memory addresses to other components.

b. The data bus sends the actual data to/from other components.

c. The control bus sends control signals to other devices.

Out of all the components looked at, the control unit, arithmetic and logic unit, and registers are considered to be the three main parts of the CPU.

Activity 1.22

Together with one or more of your study mates, create an informative display communicating all that you have learnt about the CPU. For example, you might

want to create a labelled model of the CPU including all the relevant parts (CU, ALU, registers, etc.).

Activity 1.23

In pairs, pick one of the following processors and research it: Intel Core i5-14600K, Intel Core i9-14900K, Intel Core i7-14700K, AMD Ryzen 7 5800X3D and AMD Ryzen 97950X. Create a report of your findings that could be presented to the class.

COMPUTER BUSES – OUTSIDE OF THE CPU

We have already learnt that the system bus (the collective name for the control, address and data buses) is used within the CPU to connect its components, transporting control signals, addresses and data around, but that is not where their job ends. Buses on the motherboard are also used to connect the CPU to other components and main memory (see Figure 1.18) allowing the CPU to transfer instructions and data to and from the main memory.

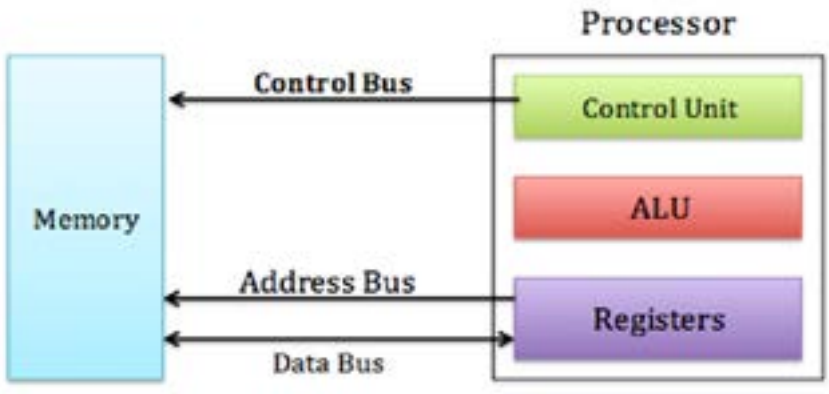


Fig 1.18 Buses connecting the processor and main memory

Activity 1.24

1. How do system buses contribute to the functioning of a computer system?
2. When the processor is at work, what are the three types of buses that are crucial for its operation? Explain your response.

Next, as we go through the machine cycle, we will highlight how the buses are used.

Machine Cycle

The CPU does its work by following a specific process called the *machine cycle*, which consists of four main steps: fetch, decode, execute and store. For this reason, the machine cycle is also known as the *Fetch-Decode-Execute-Store cycle*. This process is a routine that the CPU repeats to run a programme/set of instructions. Figure 1.19 shows stages of the machine cycle and what parts of the computer are involved.

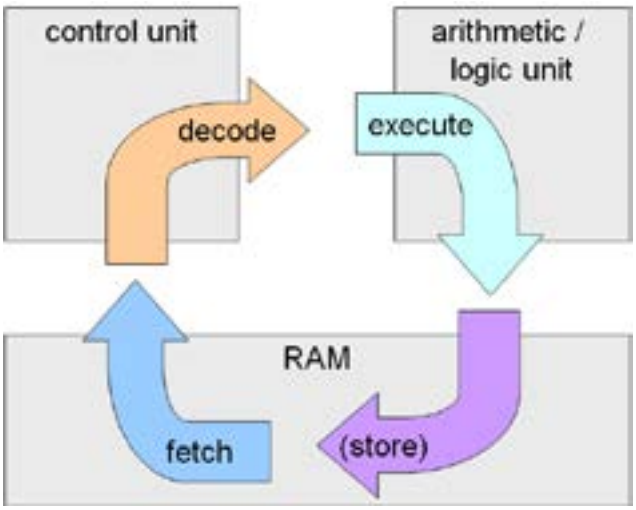


Fig 1.20 The stages involved in the machine cycle.

1. Fetch

During the fetch stage, the CPU retrieves an instruction from memory. This is achieved by the address bus providing the location (or *address*) of the instruction in the RAM, and the data bus transferring this instruction to the processor. The programme counter is a register that keeps track of the memory address of the next instruction. This is where the address bus finds the address, and after it has passed the address of the current location, the programme counter is incremented to point to the next instruction (by storing its address). The fetched instruction is then stored in a designated register within the CPU for further processing.

2. Decode

In the decode stage, the CPU interprets the fetched instruction. It identifies the operation to be performed and determines the operands involved. The instruction is decoded into a format that the CPU can understand and execute.

3. Execute

Once the instruction has been decoded, the CPU executes the instruction; that is, carrying out the necessary operations specified by the instruction. This stage may involve arithmetic or logical calculations, data manipulation, or control flow modifications.

4. Store

The final stage of the machine cycle is the store stage. Here the results of the executed instruction are stored back in memory or designated registers. This stage ensures that the changes made during the execution are preserved going forward for future operations or the overall programme's functioning. The four main processes are shown in Figure 1.20.

The FDES cycle repeats continuously, with each cycle processing one instruction at a time. By following this cycle, a computer system can execute complex programmes by sequentially fetching, decoding, and executing instructions, storing any results. The **CPU clock**, with its regular electronic pulses, synchronises the various stages of the cycle, ensuring each step happens at the right time.

Activity 1.25

1. With reference to the machine cycle, explain how a computer runs a programme.
2. What are the steps involved in the machine cycle? Use your own words to explain them in turn.
3. How would you describe what occurs during the decode stage of the machine cycle? Taking the task of playing the song 'Putuu' by Stonebwoy as a case study.
4. Explain the fetch stage of the machine cycle, with specific reference to the buses involved.

CPU INSTRUCTION SET

A CPU's instruction set, also known as instruction set architecture (ISA), is a set of commands the CPU can understand and execute. It defines the operations that can be performed, the data types that can be manipulated, and the addressing modes for accessing memory. The ISA is the CPU's programming language. It is how we tell the CPU to perform certain tasks. Some instructions are simple read, write and move commands that direct data to different hardware elements. Programmes written in programming languages are *compiled* (translated) down to these instructions so they can be run. Whatever task you get a computer to do boils down to a sequence of ISA instructions that the CPU executes using the machine cycle.

Embedded systems

An embedded system is a special-purpose computer system, which is completely encapsulated by the device it controls. Its purpose is to control the device and to allow a user to interact with it.

An embedded system has specific requirements and performs pre-defined tasks, unlike a general-purpose personal computer. Embedded systems are a combination of hardware and software which facilitates mass production and a variety of applications. Embedded devices are not usually programmable by a user – the programming is usually done beforehand by the manufacturer. However, it is often possible to upgrade the software on an embedded device. For example, fitness trackers are embedded systems whose software can be upgraded. However, the upgrade can only be done by connecting the tracker to a computer/smartphone (wirelessly or wired) and installing the new software.

Domestic Appliances

Electronic Calculators

Central Heating Systems

Fitness Tracker

Digital Watch

GPS Systems

Fig 1.20 Examples of embedded systems

Programmes on an embedded system often run in real time with limited hardware resources. For example, often there is no disk drive. The software may be stored in a type of memory called Flash ROM. If a user interface is present, it is often a small keypad and screen or touch screen.

Advantages of an embedded system

1. They can be used in a wide variety of products and devices, and to create new ones.
2. They are ROM based so operate very quickly.
3. These systems are tiny and can fit into all sorts of gadgets, making them perfect for things like smartphones or smartwatches.
4. They can react fast to what's happening around them, like in cars or phones, because they are designed to work in real time.
5. Embedded systems are specialized, so they are good at doing their jobs without messing up.

Activity 1.25

1. Look around your home, school and place of work and name two examples of embedded systems you might find in each place. Explain what the embedded system does in each case.
2. What do you think are two important characteristics shared by most embedded systems? Justify your answer.
3. Create a presentation demonstrating how embedded systems are employed in managing household tasks, improving healthcare or enhancing transportation systems.

Review Questions

1. Differentiate between the following terms:
   a. Data and Information
   b. Bit and Byte
   c. Pixel and Bitmap
2. Papa Eyram was asked by his daughter, Mawusi, to aid her in a computing homework. She was asked by her computing facilitator, Mr. Elikem, to convert the following decimal numbers into their binary representations. As a computing student, help Papa Eyram to also help his daughter.
   a. 25
   b. 57
   c. 128
3. In a Science and Maths Quiz competition between three schools in the Upper West Region, the competing schools were to convert the following numbers into their decimal representations. Assuming you were one of the representatives of the schools, what would have been your answers to these questions, showing steps of working?
   a. 11010
   b. 100111
   c. 1110100
4. If a computer has a 4-bit binary system, which of the following decimal numbers cannot be accurately represented, and why?
   a. 7
   b. 15
   c. 16
   d. 9
5. A digital speedometer of a moving vehicle shows 8 bits reading as 11001100. What is the speed of the vehicle in decimal value?
6. The binary representation of 9 is 1011. (True/False)
7. Match the decimal numbers with their binary representations:
   - Decimal: 4
   - Binary: ___
   - Decimal: 17
   - Binary: ___
   - Decimal: 35
   - Binary: ___

8. Sir Kojo was passing by to attend to one of his lessons and heard students in 3A General Arts class making noise. He then tasked the class to convert decimal number 22 into its binary representation. What would be your answer if you were a member of the class that day?

   A. 10101
   B. 11010
   C. 10110
   D. 11011

9. What does the AND operation do if both input values are 1?

   A. Outputs 0
   B. Outputs 1
   C. Outputs either 0 or 1 randomly

10. Which logic operation inverts the input value?

   A. AND
   B. OR
   C. NOT

11. The OR operation requires both inputs to be 1 to produce an output of 1. True or False?

   A. True
   B. False

12. Which of the following best describes the difference between RAM and ROM?

   A. RAM is volatile and ROM is non-volatile.
   B. RAM is used for permanent storage and ROM is used for temporary storage.
   C. RAM is non-volatile and ROM is volatile.

13. Given that 1 KB equals 1024 bytes, how many bytes are there in 7 KB?

14. Why is ROM considered non-volatile?

   A. It retains data without power.
   B. It loses data when power is turned off.
   C. It can be easily rewritten.
   D. It operates at high speeds.

15. Given two binary numbers A = 01101 and B = 10011, after performing the AND operation point wise (bit by bit) on A and B, how many 1s are in the output?

   A. 1
   B. 2
   C. 3

16. If a computer has 8GB of storage and a programme takes up 8000 MB of memory, can the programme be stored? Explain your reasoning.
17. A student claims that deleting files from a computer's ROM can speed it up. Evaluate this statement and explain why it is correct or incorrect.
18. Create a truth table for the following statement and evaluate when it is true:
    'Kojo is not an artist and Abena is not a musician'
19. What does CPU stand for?
20. Which part of the human body is the best analogy for what the CPU is to a computer?
    A. Heart
    B. Brain
    C. Eyes
    D. Skin
21. Which component is primarily responsible for executing instructions in a computer?
    A. RAM
    B. ROM
    C. CPU
    D. SSD
22. Identify the three main components of a CPU.
    A. Arithmetic Logic Unit (ALU), Control Unit (CU), and Registers
    B. Hard Drive, Monitor, and Keyboard
    C. Motherboard, Power Supply, and Fan
    D. RAM, ROM, and SSD
23. Explain the role of CPU registers; specifically, how they help make computers more efficient.
24. Explain the role of the ALU.
25. Explain the role of the CU.
26. How does increasing the clock speed of a CPU affect its performance?
27. If Efo Danny's personal computer's CPU can process 10 thousand instructions per second, how many instructions can it process in one minute?
28. Describe how cache memory benefits CPU performance.
29. What are the four stages of the machine cycle?
30. Name the three buses that make up the system bus.

31. What is the name of the set of commands that a processor can understand and execute?
32. Describe the steps involved in the fetch phase in the Fetch-Decode-Execute-Store cycle.
33. Research and prepare a presentation to illustrate, with examples, how a typical instruction is processed through the Fetch-Decode-Execute-Store cycle.
34. Explain the role the CPU clock plays in the machine cycle.

Answers to Review Questions

1. **a.** Data is the raw bits which is meaningless while information is the processed data which has meaning.

   b. A bit is the smallest unit of data and byte is eight contiguous bits.

   c. A pixel is the smallest unit of a digital image or display while bitmap is an array/grid of binary data representing the colour values of pixels in an image or display.`

2. 25 in binary is 11001 (Refer to pages 3 and 4 for steps to convert Decimal to Binary).

   57 in binary is 111001.

   128 in binary is 10000000.

3. 11010 in decimal is 26.

   100111 in decimal is 39.

   1110100 in decimal is 116.

4. In a 4-bit binary system, the highest number it can represent is 15 (1111 in binary). Therefore, the decimal number 16 cannot be accurately represented. 16 exceeds the maximum value a 4-bit system can represent, while 7, 15, and 9 can be accurately represented within a 4-bit limit.

5. The digital speedometer reading of 11001100 in binary corresponds to a speed of 204 in decimal.

6. False, 1011 represents 11.

7. - Decimal: 4, Binary: 100

   - Decimal: 17, Binary: 10001

   - Decimal: 35, Binary: 100011

8. C

9. B: output 1

10. C: NOT

11. B: False

12. A

13. Answer: 7KB = 7 * 1024 = 7168 bytes

14. B: it retains data without power

15. A:. 1

16. Yes, the programme can be stored. 8 GB is equal to 8192MB (since 1GB = 1024MB), which is more than the 8000MB required by the programme.

17. Incorrect. ROM (Read-Only Memory) is non-volatile and used to store firmware or permanent software to boot the computer. It is not used for general file storage, and users typically cannot modify or delete its contents.

18. Truth table:

    A = Kojo is an artist

    B = Abena is a musician

A	B	NOT(A)	NOT(B)	NOT(A) AND NOT(B)
1	1	0	0	0
1	0	0	1	0
0	1	1	0	0
0	0	1	1	1

    The statement is true only in case 4.

19. CPU stands for Central Processing Unit.

20. B: Brain

21. C: CPU

22. A: Arithmetic Logic Unit (ALU), Control Unit (CU), and Registers

23. CPU registers act as small, fast storage spaces within the computer's brain (the CPU). They store data, programme instructions and memory addresses. This quick access to data helps computers work more efficiently by allowing the CPU to store and retrieve information rapidly, enhancing the speed and performance of tasks like running programmes and performing calculations.

24. The Arithmetic Logic Unit (ALU) helps the CPU to do calculations and solve problems. For example, when you add numbers on a computer or compare two values, the ALU is the part of the CPU that does these tasks.

25. The CU interprets programme instructions and directs the flow of data between the CPU, memory, and other input/output devices, ensuring that tasks are executed in the correct sequence and according to the instructions provided.

26. Increasing the clock speed of a CPU improves its performance by allowing it to execute more instructions per second, leading to faster processing and better overall system performance.

27. The CPU can process 600,000 instructions in one minute.

28. Cache memory benefits CPU performance by providing a small, but extremely fast, memory area that stores copies of the data and instructions that are frequently accessed from the main memory (RAM).

29. Fetch, Decode, Execute and Storage.

30. Control bus, Address bus and Data bus.

31. Instruction Set or Instruction Set Architecture (ISA).

32. Steps involved in the fetch phase in the Fetch-Decode-Execute-Store cycle.
    a. The phase begins with the address of the instruction to be fetched stored by the Program Counter (PC).
    b. The address bus collects this address from the PC (which is then incremented to store the next location) and informs the CPU of the location of the instruction to be fetched.
    c. The data bus goes to this location and brings the instruction back to the CPU.

33. The presentation should include examples and visuals to explain each phase clearly and how they connect to form a complete cycle of instruction processing.

34. The clock generates regular electronic pulses that synchronise the various stages of the cycle, ensuring each step happens at the right time.

EXTENDED READING

- Click on the link below to learn more on ASCII Code and Binary https://youtu.be/H4l42nbYmrU?si=BxrlW76QNrAbi6zS
- Click on the link below to watch a video on binary numbers
  https://www.youtube.com/watch?v=kcTwu6TFZ08&t=2s
- Click on the link below to play a binary game
  https://learningcontent.cisco.com/games/binary/index.html
- CSC 170 – Introduction to Computers and their Applications – Lecture #1 – Digital Basics

Computational Fairy Tales:

- Pixels, Peacocks, and the Governor's Turtle Fountain
  https://computationaltales.blogspot.com/2011/06/pixels-peacocks-and-governors-turtle.html
- Using Binary to Warn of Snow Beasts
  https://computationaltales.blogspot.com/2012/01/using-binary-to-warn-of-snow-beasts.html
- Unhappy Magic Flowers and Binary
  https://computationaltales.blogspot.com/2011/06/unhappy-magic-flowers-and-binary.html
- Introduction to Boolean logic https://youtu.be/-hhZIFEGIIU
- Difference between SRAM and DRAM https://youtu.be/n974TtK0qnA
- What is a Browser Cache? How Do I Clear It? (youtube.com)
- CPU Cache Explained - What is Cache Memory? - YouTube

Computational Fairy Tales:

- Computer Memory and Making Dinner https://computationaltales.blogspot.com/2011/03/computer-memory-and-making-dinner.html
- Caching and the Librarian of Alexandria https://computationaltales.blogspot.com/2011/03/caching-and-librarian-of-alexandria.html

Watch this video on how a CPU works:

- https://www.youtube.com/watch?v=cNN_tTXABUA
- The Fetch-Execute Cycle: What›s Your Computer Actually Doing? - YouTube (9 mins)

REFERENCES

Appiah O. K., Birbal R., Taylor M. (2008). ICT for Senior High Schools, Students' Book. Sedco Publishing Limited, pg. 38 - 40.

Computer Science Wiki (2016). Fetch-execute-cycle.png retrieved from https://computersciencewiki.org/index.php?title=File:Fetch-execute-cycle.png.

Fenwick, P. (2014). Introduction to computer data representation. Bentham Science Publishers.

Harvard, Indonesia. (2022). Black and White Bitmap. In CS50 for Teachers. Retrieved from https://cs50.harvard.edu/indonesia/2023/psets/4/filter/less/

NaCCA (2023). Computing Teacher Manual (Year One - Book One). Pages – 47.

Teach Computer Science (2024). Sound Representation. Retrieved from https://teachcomputerscience.com/sound-representation/

Thapliyal, N. (2023). Difference Between RAM and ROM retrieved from https://www.scaler.com/topics/difference-between-ram-and-rom/

ACKNOWLEDGEMENTS

List of Contributors

Name	Institution
Mark Kwadwo Ntoso	Krachi SHS
Mahama Seidu	Daboya Community SHS
Miheso Daniel	Wa SHTS, Wa
Francis Bennet Kouadio Kouame	Odorgonno SHS, Awoshie