C readings

Computers

Learning Outcomes

After reading this section, you will be able to:

- Describe the major components of a modern computer
- Describe the software that controls a modern computer
- Outline the contents of these notes

Introduction

Computers are available in many flavours: mobile devices, smart phones, laptops, tablets, desktops, workstations and servers to name a few. All of these devices control their operations through software. Programmers create this software. Users rely on this software to operate their devices.

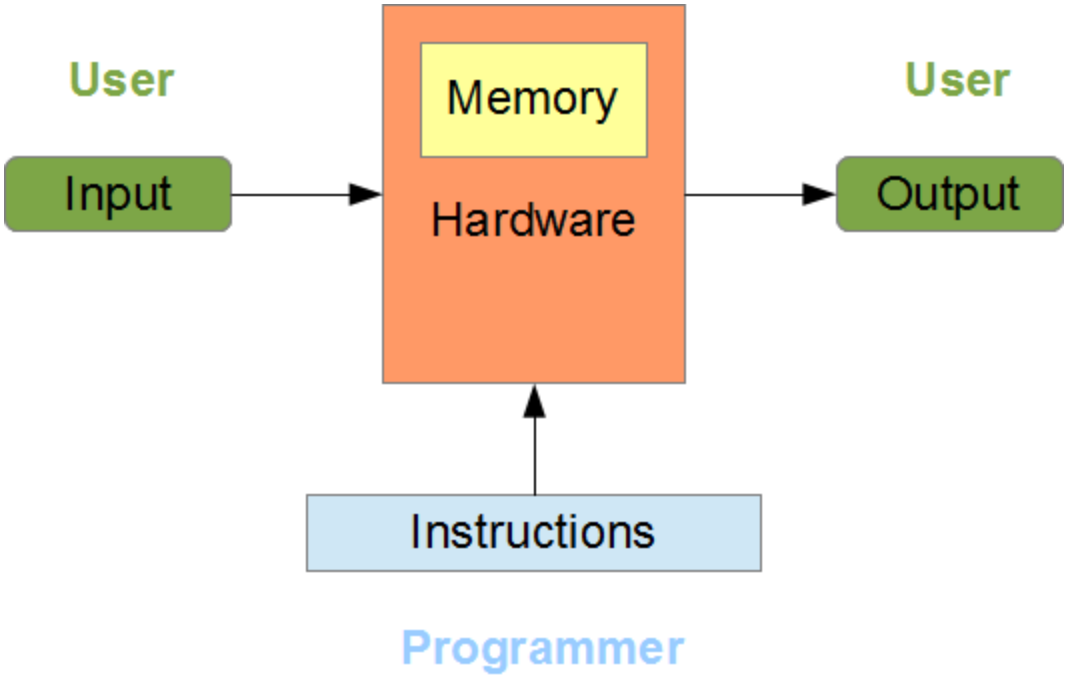
We refer to the software with which a user operates their device as application software. Application software consists of one or more programs. Each program is a full set of instructions that performs a well-defined task on the host device. Programmers code these instructions in a programming language.

The instructions that programmers code represent algorithms. An algorithm is a step-by-step procedure that describes how to achieve a specified task. Examples include searching, sorting and mixing. Application programmers study different algorithms and create their own if required. They find the code for some algorithms in libraries and write the code for their own algorithms.

This introductory chapter describes the major components of a modern computer, components to which programmers often refer. Subsequent chapters show how to write programs to use the principal features of these components efficiently.

Hardware

The figure below illustrates the relation of the programmer to a user of a software application. The boxes identify the principal elements for any programmer. The green boxes identify the elements of concern to the user.

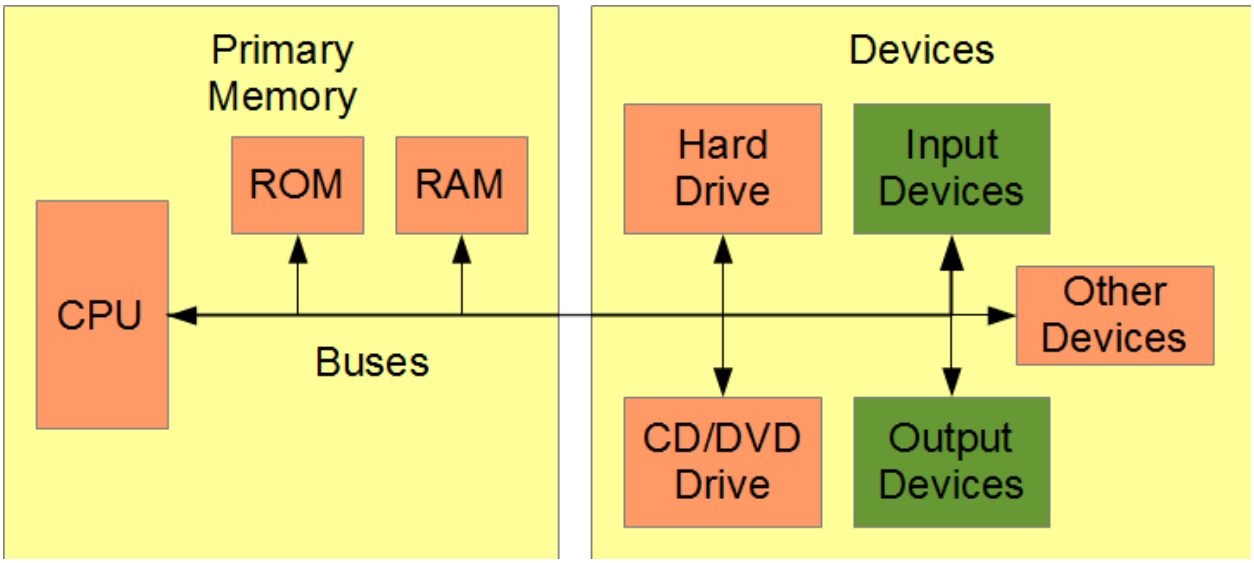


Computer hardware stores the program instructions in its own memory, accepts input from the user, processes that input according to the stored instructions, and generates the output for the user. The user can rerun the program to process different inputs and produce corresponding outputs.

Modern Computers

In 1945, John von Neumann, noting that instructions are pieces of information just like data, proposed a new computer architecture, in which instructions and data are stored alongside one another. We call this idea the stored-program concept. All modern computers are stored-program computers.

The figure below shows the components of a stored-program computer. They include a central processing unit (CPU), a clock, primary memory and a set of devices. Buses interconnect these components and are part of the motherboard. The CPU, primary memory and a clock are also part of the motherboard. The clock controls the rate at which the CPU executes the instructions.



Primary Memory

Primary memory is memory directly accessible by the CPU. Primary memory includes read-only memory (ROM), random-access memory (RAM) and memory within the CPU itself.

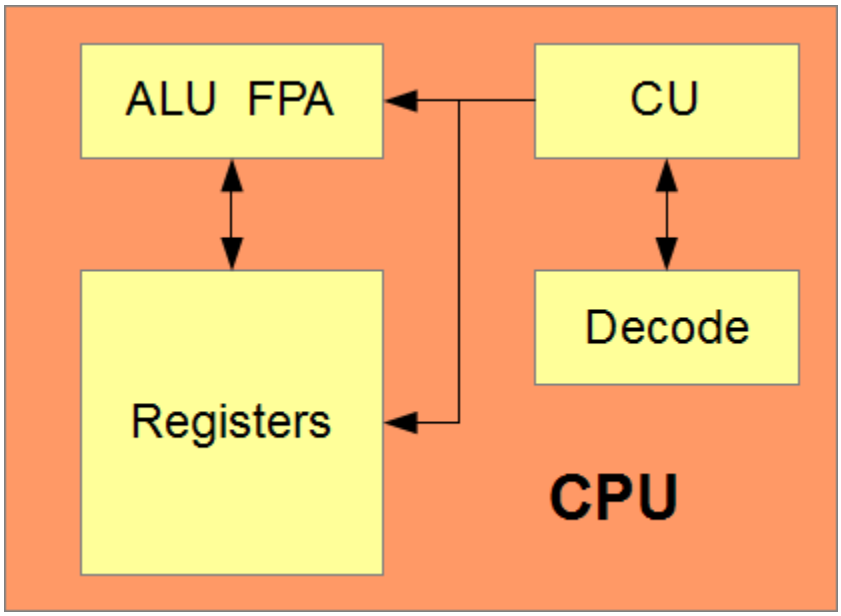
ROM holds the instructions for starting the system. ROM is not volatile: it persists if we turn off power.

RAM holds the program instructions and the program data. RAM is volatile: its contents are lost if we turn off power.

Central Processing Unit

The CPU is the work-horse of any modern computer. The CPU executes program instructions serially (one at a time). A modern CPU consists of:

- registers
- a decode unit
- a control unit (CU)
- an arithmetic and logic unit (ALU)
- a floating-point accelerator (FPA)



Registers are the CPU's internal memory. They hold the data used by the ALU and FPA and any new data that the ALU and FPA produce. Register data is volatile: we lose the contents of each register as soon as we turn off power.

The Decode unit extracts each incoming instruction from the instruction queue and decodes that instruction. The CU moves data between registers, RAM and the devices, and passes the decoded instruction to the ALU or the FPA for processing. The CU manages the data, but does not change it.

The ALU performs the comparisons and integer calculations, changes the data and creates new data as directed by the CU. The FPA performs the calculations on floating-point data. The ALU and FPA work solely with register memory inside the CPU.

Devices

The devices of a modern computer include peripheral and other devices. Peripheral devices include a keyboard, a mouse, and a monitor, which provide user interfaces for input and output. Hard drives, USB keys and DVD/CD-ROMs constitute secondary memory, which provides persistent storage of program instructions and program data.

Memory Comparison

Secondary memory is inexpensive compared to primary memory, but considerably slower. Compare the following data transfer rates:

- Registers ~10 nanoseconds
- ROM and RAM ~60 nanoseconds
- Hard disk ~12,000,000 nanoseconds

A nanosecond is 1e-9 seconds. To appreciate the differences, consider the following analogy. The ratio of the time that the CPU takes to transfer data between registers to the time that a hard disk takes to transfer that same information is the ratio of the width of an average-sized room to the distance once around the earth along the equator.

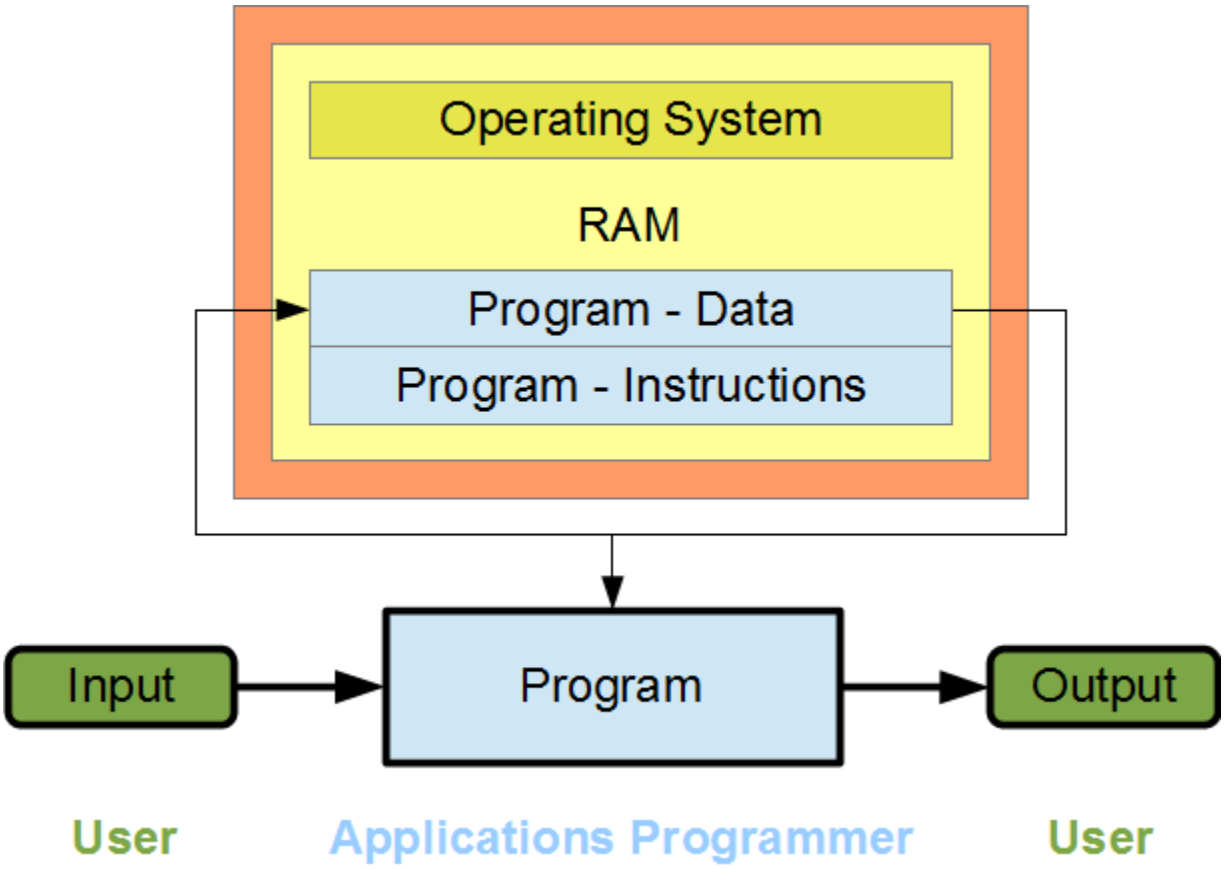
Software

The software that controls a modern computer includes the programs that are currently executing and the operating system that manages them. The operating system is a program that executes as long as power is on. The operating system resides in RAM along with the other currently executing programs.

When a user starts an application program, the operating system loads that program's instructions into part of the RAM and transfers control to the program. The program starts executing, requests data from the user, sends output to the user, and eventually terminates its own execution and returns control to the operating system.

An application program transforms raw data from the user (the input) into equivalent data stored in RAM, operates on the data in RAM and transforms the resultant data into some user-readable form (the output). Programming languages define how internal data is stored in RAM.

As application developers, we focus on input and output processing and transformation of input data into output data. The figure below relates this focus to the software that we code.



Outline

These notes introduce the fundamental concepts of software development. The chapters are grouped into six parts:

1. Introduction
2. Computations
3. Data Structures
4. Modularity
5. Secondary Storage
6. Refinements

The introductory part describes modern computers, the storage of information in memory, and how to create your first program.

The computations part introduces the concept of type, shows how to describe program variables using types, shows how to handle basic input and output, how to calculate a

new value from existing values, how to create optional paths through a program, how to write code in a friendly readable style, and how to test and debug sets of program instructions, including how to determine the output produced by those instructions.

The data structures part describes how to organize groups of values into data types in memory. This part introduces the grouping concepts of arrays and structures.

The modules part describes how to organize program instructions into self-contained cohesive units, called functions, where each function implements a single algorithm. This part shows how to divide a program into independent modules and how to pass information from one module to another.

The secondary storage part describes how to move text data between an installed device and RAM. This part introduces files and describes the syntax for working with secondary memory.

The refinements part elaborates on concepts introduced in the other parts. It covers character strings as specialized versions of arrays and shows how to work with the library functions that handle them. This part describes the relation between pointers and arrays, including arrays of structures. This part also covers two-dimensional arrays and shows how to store tabular data using strings. This part concludes with introductions to some of the standard algorithms and the guidelines for portability of program code.

Optional Sections

Some chapters include sections or sub-sections marked optional. These sections contain information that elaborates specific details related to the topic covered in these notes. Feel free to skip these sections on first reading, without disrupting presentation flow. Subsequent mandatory sections do not rely on any information covered in these optional sections. These optional sections simply add depth to the material.