

3. Basic Computer Architecture

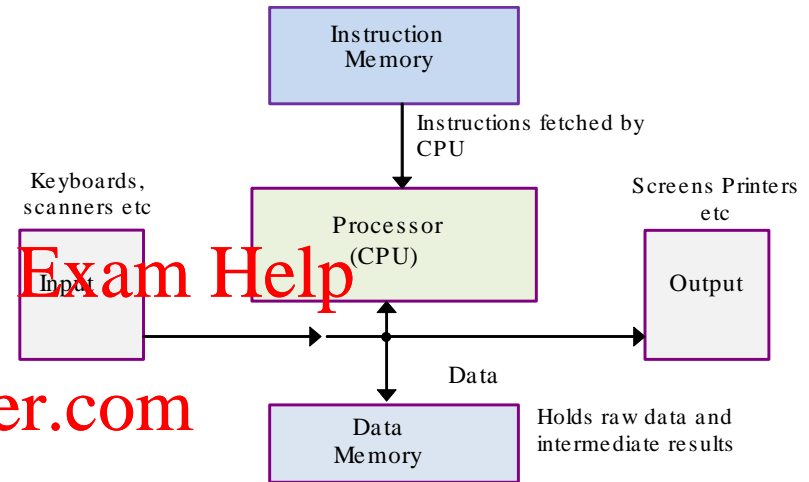
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Computer Subsystems

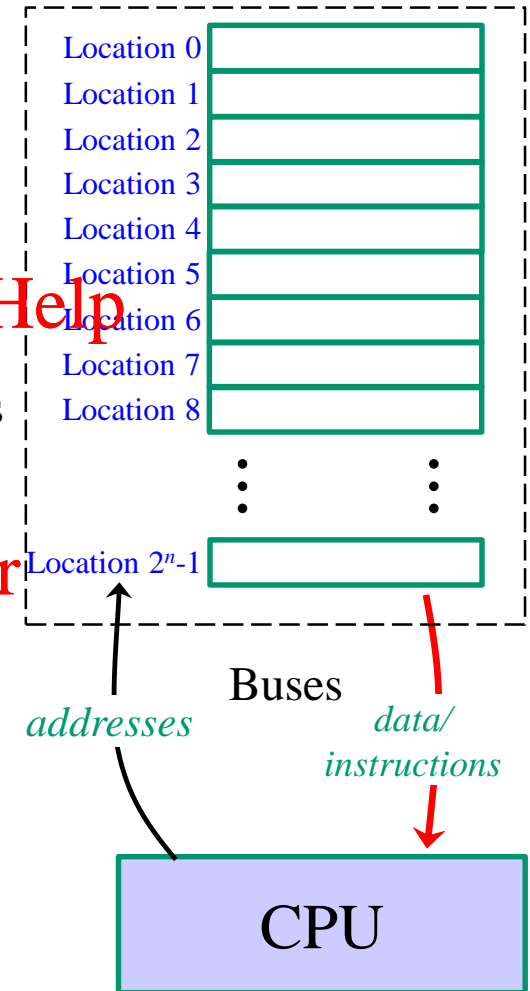
- A computer has several key subsystems:
 - **processor** or **central processing unit (CPU)** which fetches, decodes and executes sequences of binary coded instructions, known as **machine code (m/c)**.
 - **instruction memory** where a program is stored as a list of m/c instructions.
 - **data memory** where binary coded data is held.
 - **input/output subsystem** which allows data to enter the computer via I/O devices like keyboards, screens, disks etc.
 - **system interconnect** which allows the various components to communicate. This is usually implemented as one or more **buses**.



- Electronic memory can store only binary coded information.
- Each CPU has a set of m/c instructions it can interpret
- This **instruction set** is different for different CPU designs.

Memory

- Instruction and data memory have same structure.
 - Each is organised as an indexed array of registers called **locations** each capable of holding a word.
 - Each location labelled by unsigned binary index or **address**.
- CPU generates addresses when it wants to **read** from or **write** to a location.
- The length (in bits) of an address and of a memory word is set by each CPU design.
 - set of all addresses CPU can generate is its **address space**.
- Memory hardware needs to be physically mapped to any location CPU is to use.
 - If no memory cells at an address, address is **unpopulated**.
 - Unpopulated addresses cannot store anything!
 - large address spaces are often only partly **populated**



Harvard and Princeton

- Randomly addressable memory such as that used for instructions and data is called **primary memory**.
- Instructions and data are stored in primary memory locations as binary coded words.
 - If an instruction or data item has a code word too big to fit in one location it must be split across several.
- In most current computers the instructions and data are stored in a single address space (often on the same chips) accessed using the same buses.
 - A computer where instructions and data are stored in the same address space is called a **von Neumann** or **Princeton architecture**;
 - A computer with separate address spaces is called a **Harvard architecture**.

More Memory

- Physical memory is called **volatile** if its content is lost when power is removed; otherwise it is **non-volatile**.
- Some physical memory in the computer cannot be written to in normal use. Such memory is called **read only memory (ROM)**.
- ROM is non-volatile and is used, for example, to hold programs that are essential to start the computer up. Writing to conventional ROM requires special hardware.
- Other physical memory is read/write and is usually called **random access memory (RAM)**. Conventional RAM is volatile.
- Another type of memory is **flash memory** which behave like RAM but is non-volatile.
- Computers also have **secondary memory** in the form of disks and other mass storage devices. Such devices are not directly addressable by the CPU and are accessed via I/O interfaces.

The CPU

- A CPU is a compound subsystem comprising:
 - A **control unit (CU)** which controls the rest of the CPU and, through it, the computer.
 - An **execution unit (EU)** which contains hardware to perform operations on binary words (e.g. includes an adder).
 - A **register file** consisting of a set of general registers for holding data or addresses.
 - A **bus interface** through which the CPU communicates with primary memory and I/O interfaces.
- These devices are connected via buses internal to the CPU.
- A CPU on a single chip is a **microprocessor** or **MPU**.
- A complete computer on a single chip is an **MCU**

Control Unit

- The CU includes some special-purpose registers of its own.
- One of these is a register called the **program counter (PC)** that holds a memory address.
 - This always points at the next instruction to be fetched.
- When an instruction has been fetched, the CU decodes it.
- It then controls execution of the instruction and using the other subsystems as necessary.
- Unless otherwise informed, the CU will assume that the next instruction is stored in the next memory address after the last one. Some instructions (e.g. **jumps** or **branches**) can change this normal flow.
- The sequence of fetch-decode-execute is called the **instruction cycle** and is repeated for each instruction in the program.

Execution Unit

- The EU is able to perform a variety of operations on one or two binary words supplied to it by the CU.
- The core of the EU is a device called an **arithmetic logic unit (ALU)**. This device can typically:
 - Perform Boolean logic operations such as (bitwise) **NOT**, **AND**, **OR** etc. These ops can be used on any binary words regardless of code being used.
 - Add, **subtract** or compare two binary code words (both these operations work on unsigned and two's complement codes). The ALU will generate **status bits** (or **flags**) such as C and V to tell if overflow has occurred.
 - Other status bits may also be generated to inform future instructions: e.g. a **Z bit** if the result is 0.
- An EU will also have a **shifter**: hardware to shift words left or right.
- An EU may also have additional hardware to perform **fast multiply** or divide. Without this hardware, operations like multiply have to be built up from addition and shifting.

CPU Registers

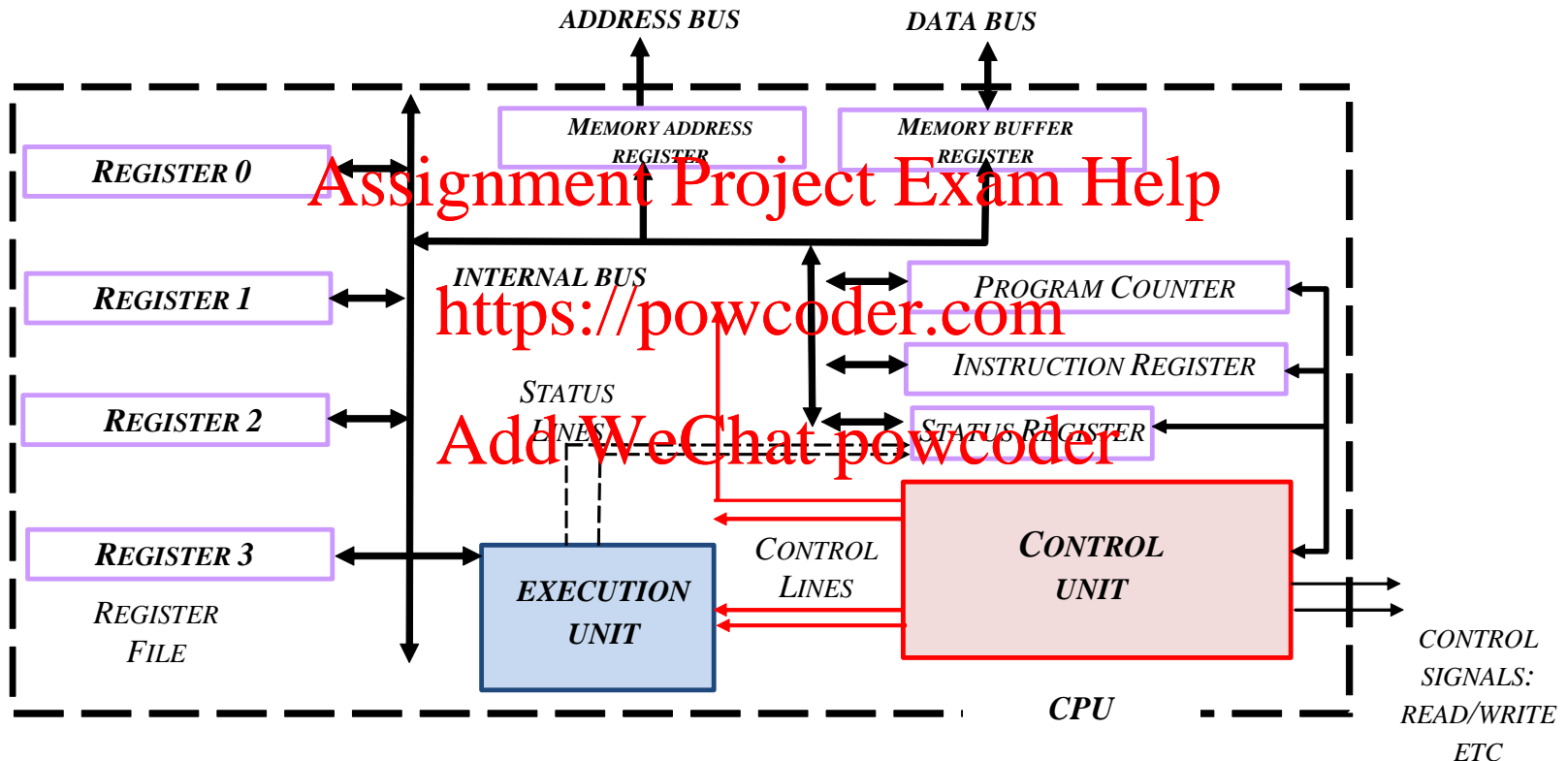
- A CPU has two types of register:
 - Registers in the **register file** for general use by instructions and are numbered.
 - **Special purpose registers** usually associated directly with the CU.
- In some CPUs some registers in the register file have special functions.
- The register file is often used to supply codewords to the EU during instruction execution and is often where results from the EU are written.
- There are two kinds of special purpose register:
 - Registers which contain useful information at the end of an instruction.
 - E.g. the PC; the **status register (SR)** present in most CPUs, which stores the status bits or flags (V,C,Z etc.) from the last instruction
 - Registers which are only used inside an instruction and have no useful information after the instruction finishes execution.
 - E.g. the **instruction register (IR)**, holds the first code-word of the instruction currently being processed (often m/c instructions have only one word).

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

CPU Internal Structure



Writing Programs

- A programmer can write a program directly as a list of m/c instructions.
- It is hard to write m/c directly. To help, each m/c instruction has a symbolic form. A program written in this form is called **assembly language**.
- Assembly language can be translated into m/c automatically by a program called an **assembler**.
- Because each type of CPU has its own m/c, programs written in assembly language are machine specific.
- Programs written in a **high-level language (HLL)** like Java or C must also be translated into m/c before being executed. The HLL is machine independent before it is translated by a **compiler**.
- A compiler must generate m/c for a specific CPU type and so the compiler itself is machine specific, even though the HLL is not
- A single HLL statement will compile into many m/c instructions.
- A single assembly statement will assemble to exactly one m/c instruction.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Instructions

- Instruction formats vary from one CPU type to another.
- In any given machine each instruction must tell the CU exactly what to do.
- To say **ADD** is not enough. The instruction must say where to find the data to be operated on (the **operands**) and where the result should be stored.
 - The locations of operands and results may be in registers or in memory locations.
 - Not all instructions allow total freedom to specify operand and result locations.
 - For example, in some CPUs, arithmetic and logic instructions can only use CPU registers. Operands must be read from memory and results written to memory using other instructions (names like **LOAD**, **STORE** or **MOVE** but CPU specific).
- The set of CPU registers that can be used to carry information from one instruction to another is called the **programmer's model**. The model typically contains all general purpose registers, the PC and the SR.
- Registers like the IR are not, however, included in the programmer's model. A programmer may ignore these registers although they are relevant to a hardware designer.

Sigma16: a 16-Bit Architecture

- In this course we will make use of a 16-bit modern-style architecture called Sigma16
 - Data words and addresses are all 16-bits long
 - There are 16 registers in the register file: R0...R15
 - R0 is always set to 0 (special purpose)
- This is not a physical machine but implemented in software.
- It is deliberately simplified so as to help teach basic principles.
- Designed by John O'Donnell; still being developed.
- Software version we will use is Sigma16-0.1.7.
- You can run this from IT Lib or download the entire folder and run it from your own (Windows) computer.
 - No installation is needed
 - Run by executing Sigma16-0.1.7\Sigma16.exe
 - Documentation at Sigma16-0.1.7\Sigma16-0.1.7\index.htm

The Instruction Cycle

1. CU sends PC address to Bus Interface **Memory Address Register (MAR)**
2. CPU reads first word of instruction from memory into **Memory Buffer Register (MBR)**
3. First word placed in **Instruction Register (IR)**
4. CU **decodes** instruction.
5. If instruction has more than one word, CU commands bus interface to get the rest.
6. CU organises execution. This depends on what type of instruction. Assume ADD.
7. CU advances PC to point to next instruction.
8. CU determines where operands are. If any operands are in memory, CU commands bus interface to get them. (In Sigma16, ADD operands are only allowed in register file).
9. CU supplies EU with operands and commands EU to add.
10. CU determines where result goes and writes it. In some CPUs this can also mean a memory access (in Sigma16, ADD result must go into register file).
11. EU generates **status bits (C, V, Z etc.)** and stores them in **status register (SR)**.
12. Finished. Start fetch of first word of next instruction

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder