5 Computer Organisation and Architecture

5.25 Internal Computer Hardware

- **External components** include input/output and storage devices.
- **Internal components** are those within the CPU.
  - Processor
  - Main memory
  - Address/control/data bus
  - I/O controllers

The Processor

The processor **responds to and processes the instructions** that drive the computer. It contains

- The **control unit** coordinate s and controls all operations carried out by the computer. It operates by repeating the **fetch-decode-execute cycle**.
- ALU performs operations on data, such as **arithmetic operations** and **logical operations** and **shift operations**.
- Registers are **special memory cells** that operate at very high speeds. All arithmetic and logical operations take place within Registers.

Buses

Each bus is a **shared transmission medium**, only one device can transmit at a time..

- **Control bus**, a bidirectional bus to **transmit command** between components, and ensure the access to and use of the data and address buses by different components **does not lead to conflict**.

  The control bus is made of **control lines**, including

  - Memory read/write
  - Interrupt request
  - Bus request/grant
  - Clock
  - Reset
- **Data bus**, a bidirectional bus for **moving data and instructions** between components. The width of the data bus is a key factor in determining **overall system performance**.

- **Address bus** - specify an address to access a particular memory location.

  The memory is divided into **words** handled as a unit by the processor, each word in memory has its own address. The width of the address bus determines the **maximum possible memory capacity** of the system.

I/O Controller

An I/O controller is a device which **interfaces between and I/O device and the processor**. Each device has a **separate controller** which connects to the control bus.

The controller consists of

- An interface that allows connection of the controller **to the system bus**.
- A set of data, command, and status **registers**.
- An interface that enables connection of the controller **to the cable** connecting the device to the computer.

An **interface** is a standardised form of connection defining things such as signal, voltage levels, etc.

- The **von Neumann architecture** - a shared memory and bus is used for both data and instructions.
- The **stored program concept** - a program **must be in main memory** to be executed, and instructions are fetched from memory one at a time.
- The **Harvard architecture** - physically separate memories for instructions and data. It is used in **embedded systems** as instruction can use a read-only memory.

Harvard architecture is faster than von Neumann because data and instructions can be **fetched in parallel**.

5.26 The Processor

- The **ALU** perform arithmetic and logical operations on the data, as well as **shift operations** and **boolean logic operations**.
- The **control unit** controls and coordinate the activities of the CPU - directing the **flow of data** between the CPU and other devices.
  - Accepts the **next instruction**.
  - **Break down** its processing into several sequential steps.
  - Manages its **execution**.
  - **Stores** the resulting data back in memory or registers.

- The **system clock** generates a series of signals to **synchronise CPU operations**.
- **General-purpose registers** are very fast memory, all arithmetic, logical or shift operations take place in registers.

Dedicated registers include

- The **program counter** holds the address of the **next instruction** to be executed.
- The **current instruction register** holds the **current instruction** being executed.
- The **memory address register** holds the address of the memory location from which data is to be fetched or written.
- The **memory buffer register** is used to temporarily store the data read from or written to memory.
- The **status register** contains bits that are set or cleared depending on the result of an instruction.

The Fetch-Execute Cycle

This cycle is repeated over and over as each instruction of the program is executed.

1. **Fetch phase**

   The address of the next instruction is copied **from PC to MAR**, the address is **sent via address bus** to memory.
2. The instruction held at that address is returned along the data bus **to the MBR**. Simultaneously, the content of the **PC is incremented** so it holds the address of the next instruction.
3. The content of the MBR is **copied to the CIR**.
4. **Decode phase**

   The instruction held in the CIR is decoded - it is split into **opcode** and **operand**.

   - The opcode determine the **type of instruction** and what hardware to use to execute it.
   - **Additional data** is fetched if necessary and passed to the registers.
5. **Execute phase**

   The instruction is executed using the ALU if necessary, the results are stored in general purpose registers or memory.

Processor Performance

- **Number of cores**: each core is able to process a different instruction at the same time with **its own fetch-execute cycle**.

  However some software may not be able to take full advantage of multiple processors.

- **Cache** is a very small amount of expensive, very fast memory inside the CPU. An instruction fetched from main memory is copied into the cache if it is **needed again soon**.

  As cache fills up, unused data are **replaced with more recent ones**.

- **Clock speed**: all processor activities begin on a clock pulse. The greater the clock speed, the faster the instructions will be executed.
- **Word length** is the number of bits that the CPU can **process simultaneously**.
- The **width of data bus** determines how many bits can be transferred simultaneously.

  The **width of address bus** determines the **maximum memory address** that can be directly referenced.

Interrupts

An interrupt is a signal sent by a software program or a hardware device to the CPU.

- **Software interrupt** occurs when an application terminates or requests certain services from the operating system.
- **Hardware interrupt** occur when an I/O operation is complete or an error occurs.

When the CPU receives an interrupt signal

1. **Suspends execution** or running program.
2. Puts values of each register and PC onto the **system stack**.
3. An **interrupt service routine** is called to deal with the interrupt.
4. Once served, the original values of the registers are retrieved from the stack, and the fetch-execute cycle resumes.