Understanding Computer System Components

Components of a Computer System

A computer system consists of three essential components: memory, processors, and input/output peripherals.

Types of Computer Systems

Computer systems can be classified into two types:

- General Purpose: Systems designed for various tasks, such as laptops and desktops.
- Special Purpose: Systems built for a specific purpose, such as embedded devices in appliances or industrial machinery.

System on Chip (SoC) vs. System Board

System on Chip (SoC)

- Integrates all components (processor, memory, peripherals) onto a single chip.
- Smaller in size and lower in cost.
- Limited flexibility for upgrades or expansions.

System Board

- Separate components assembled on a circuit board.
- Allows for easier upgrades and customization.
- Higher cost compared to SoCs.

Integrated Circuits (ICs)

ICs are small chips that perform specific functions within a computer system.

- Discrete ICs: Single-chip ICs with limited functionality.
- VLSI (Very Large Scale Integration) ICs: ICs with millions of transistors, enabling complex functions and high performance.

Microprocessor

- The primary processor in a computer system.
- Handles instructions and computations.
- Consists of a control unit, registers, and arithmetic/logic unit.

Microcontroller Unit (MCU)

- A small computer system on a single chip.
- Contains a microprocessor, memory, and peripherals.
- Used in embedded and control systems.

System on Chip Plus Peripheral (SoC+)

- An MCU with additional high-performance peripherals.
- Offers greater processing power and functionality than MCUs.

Understanding the Fundamentals of a Computer System

Components of a Computer System

A computer system consists of three main components:

- Memory Unit: Stores data and instructions.
- Central Processing Unit (CPU): Executes the instructions and performs calculations.
- Input/Output (I/O) Devices: Connect the computer to the outside world, allowing communication with users and peripherals.

The CPU

The CPU is the brain of the computer, responsible for processing and executing instructions. It consists of three key components:

- Control Unit: Fetches and decodes the instructions, directing the execution process.
- Arithmetic Logic Unit (ALU): Performs mathematical and logical operations on data.
- Registers: Temporary storage locations that hold data and results during processing.

Instruction Execution Cycle

When an instruction is executed, it goes through the following cycle:

1. Fetch: The instruction is retrieved from memory.
2. Decode: The instruction is analyzed to determine the operation to perform.
3. Execute: The ALU performs the operation on the specified data.
4. Writeback: The result of the operation is stored in memory or a register.

Memory

Memory stores data and instructions in a sequence of numbered locations called 'addresses.' Each location can hold a fixed amount of data, typically 8 or 16 bits.

Read Access: Retrieving data from memory.
Write Access: Storing data into memory.

Registers

Registers are high-speed temporary storage locations within the CPU. They hold:

Data: Input data, intermediate results, or final outputs.
Flags: Status information, such as the result of an operation (zero or non-zero).

Memory Types: DRAM, SRAM, and ROM

DRAM (Dynamic Random Access Memory)

- Definition: RAM is a volatile memory that stores data as long as power is supplied.
- Characteristics:
- Cannot retain data without power: Once power is removed, data is lost.
- Requires periodic refresh: To prevent data loss, a refresh circuit constantly rewrites the data.
- Advantages:
- High density: Can store large amounts of data in a small form factor.
- Relatively low cost: Compared to other memory types.
- Disadvantages:
- Slow speed: Due to the need for refreshing.
- Power consumption: Requires constant power for operation.

SRAM (Static Random Access Memory)

- Definition: SRAM is a volatile memory that stores data without the need for refreshing.

- Characteristics:
- Faster than DRAM: Does not require refreshing.
- Lower density: Can store less data in the same form factor as DRAM.
- Higher cost: More expensive than DRAM.
- Advantages:
- Faster speed: Ideal for high-speed applications.
- No need for refreshing: Eliminates the speed penalty associated with DRAM.
- Disadvantages:
- Lower density: Limits storage capacity.
- Higher cost: Can be a significant cost factor.

ROM (Read-Only Memory)

- Definition: ROM is a non-volatile memory that stores data permanently.
- Characteristics:
- Cannot be overwritten: Data is programmed into ROM during manufacturing and cannot be changed.
- Retains data without power: Unlike DRAM, ROM stores data even when power is removed.
- Advantages:
- Permanent data storage: Ideal for storing firmware, boot instructions, and other critical data.
- Low power consumption: Does not require constant power like DRAM.
- Disadvantages:
- High cost: More expensive than DRAM and SRAM.
- Limited capacity: Can only store a fixed amount of data.

A Comprehensive Guide to Computer Memory: ROM and RAM

Introduction

This article provides a detailed overview of read-only memory (ROM) and random-access memory (RAM), two essential components of computer hardware responsible for data storage and processing.

ROM (Read-Only Memory)

- Definition: ROM is a type of non-volatile memory that stores permanent data. Once data is written to ROM, it cannot be overwritten or erased.
- Purpose: ROM is primarily used to store firmware, such as the boot sequence and low-level BIOS settings, which are essential for the computer to start and operate.
- Types:

- PROM (Programmable ROM): Data can be written to PROM only once using a special programmer.
- EPROM (Erasable PROM): Data can be erased by exposing the chip to ultraviolet light and then reprogrammed.
- EEPROM (Electrically Erasable PROM): Data can be erased and reprogrammed electrically, allowing for quick updates.

RAM (Random-Access Memory)

- Definition: RAM is a volatile memory that stores data temporarily. Data in RAM is lost when the computer is turned off.
- Purpose: RAM is used to store the operating system, programs, and data that are actively being processed by the computer.
- Types:
- SRAM (Static RAM): Retains data as long as power is supplied.
- DRAM (Dynamic RAM): Requires periodic refresh cycles to retain data.

Comparison of ROM and RAM

Feature	ROM	RAM
Data Persistence	Non-volatile (permanent)	Volatile (temporary)
Writability	Read-only	Read-write
Primary Use	Firmware, BIOS settings	Operating system, programs, data
Speed	Slower	Faster
Cost	More expensive	Less expensive

Hybrid Memory (Flash)

- Definition: Flash memory combines characteristics of both ROM and RAM.
- Type: Flash is a non-volatile memory that allows data to be overwritten electrically.
- Advantages:
- Faster than ROM
- Can be reprogrammed multiple times
- Less expensive than SRAM

Cache Memory

- Definition: Cache memory is a high-speed memory that stores frequently accessed data.
- Purpose: Cache memory reduces the time required to retrieve data from the slower main memory (RAM).
- Levels: Multiple levels of cache memory (L1, L2, L3) are typically organized based on speed and size.

Cache Memory

Cache memory is a faster type of memory that sits between the CPU and RAM. It stores frequently used data and instructions, allowing the CPU to access them more quickly.

Types of Cache

There are multiple levels of cache, each with its own speed and capacity. Level 1 (L1) cache is the fastest and smallest, followed by L2 and L3 caches.

Memory and Cache Interaction

When the CPU needs to access data, it first checks the L1 cache. If the data is not found, it checks the L2, then the L3 cache, and finally the RAM. If the data is not found in any of these caches, it is retrieved from the disk drive, which is the slowest but most spacious storage device.

The size and speed of the cache memory can significantly impact the performance of a computer system. By storing frequently used data in cache, the CPU can quickly access it without having to wait for the slower RAM or disk drive.

Bus Interfaces

The CPU, memory, and cache are connected by a set of buses:

- Address Bus: Carries the memory address of the data being requested.
- Data Bus: Carries the data itself.
- Control Bus: Manages the flow of data between the CPU, memory, and cache.

Address Decoding

When the CPU requests data, it provides an address on the address bus. The control bus then determines which memory or cache device is responsible for that address. This process is known as address decoding.

Cache Coherency

When multiple cores or CPUs share the same cache, it is important to ensure that they have consistent copies of the data. This is known as cache coherency. Special hardware mechanisms are used to maintain cache coherency and prevent data corruption.

Harvard vs. Von Neumann Architecture

Introduction

In computer architecture, Harvard architecture and von Neumann architecture are two distinct design approaches for storing and processing data. Understanding their differences is crucial for system optimization.

Harvard Architecture

- Utilizes separate memory and data buses for instructions and data, allowing for faster data access.
- E.g., Used in microcontrollers where code and data are stored separately.

Von Neumann Architecture

- Employs a shared memory bus for both instructions and data, simplifying design but potentially slowing down execution.
- E.g., Suitable for PCs that primarily interact with RAM.

Advantages and Disadvantages

Feature	Harvard Architecture	Von Neumann Architecture
Memory Access	Faster	Slower
Design Complexity	More complex	Simpler
System Speed	Higher	Lower

Clock Generation

- Clock is crucial for system timing.
- Clock frequency determines the maximum number of instructions processed per second.

Oscillators

- Types of oscillators used to generate clock signals:
- Asymmetric multivibrator (Astable multivibrator): Uses a resistor and capacitor to generate a square wave.
- Crystal oscillator: Utilizes a piezoelectric material that vibrates at a specific frequency when powered.
- Ceramic oscillators: Offer lower cost but lower accuracy compared to crystal oscillators.

Noise Immunity

- Noise can interfere with clock signals.
- Crystal oscillators are more prone to noise than ceramic oscillators.
- RC oscillators have the lowest noise immunity but also the lowest cost.

Application Considerations

- The choice of oscillator depends on factors such as:
- Noise environment
- Frequency accuracy requirements
- System cost

Comprehensive Understanding of System Design Using Protocols

Introduction

System design involves connecting various components, including processors, memory, and peripherals, through communication protocols. These protocols define the rules for transmitting data between devices.

Bus Protocols

Two common bus protocols are:

- Peripheral Bus Protocol (PBP): Connects multiple peripheral devices to a master device directly.
- Memory Bus Protocol (MBP): Connects memory devices to a master device, providing access to data stored in memory.

System Address Space

Each peripheral or memory device has a specific address range assigned to it. The address range for PBP devices is determined by the protocol, while the address range for MBP devices is determined by the memory map defined in the system design.

Register Mapping

Registers are memory locations within devices that control specific functions. The address offset for each register is used to access it.

Microcontroller Design

In the example, a designer named Hamada wants to create a custom microcontroller with a CPU, GPIO, and other peripherals. The microcontroller connects to the system bus through a master device, and each peripheral connects to the system bus through a slave device.

Memory Map and Register Access

The memory map defines the address range and size of each memory and register section in the microcontroller. To access a register, its address offset is used.

Data Writes to Registers

When a data write operation is performed on a register, a transaction is initiated on the system bus. The transaction includes the address of the register, the data to be written, and a control signal. The register is then updated with the new data value.

Protocol Considerations

Different protocols have different features. For example, the I2C protocol is commonly used for small devices with limited bandwidth, while the PCI Express protocol is used for high-performance applications.

Protocol Conversion (Bridging)

Sometimes, it is necessary to bridge between different protocols. This is done using a bridge device that converts signals from one protocol to another.

Example: Debugging a Master Device

In the example, suppose the microcontroller has a 12-bit register that cannot store a 16-bit value. The issue lies in the size of the register, which should be increased to accommodate the larger data size.

Embeddded Systems Processor Fundamentals - Lectures Summary - Lecture 4

Registers - what is an RD and WR register?

In this lecture, we discussed registers. In the context of embedded systems, registers are special memory locations within the processor that can be directly accessed by software instructions. We learned about two types of registers:

- Read-only (RO) registers can only be read from, not written to.
- Read-write (RW) registers can be both read from and written to.

We also learned that the type of a register (RO/RW) is determined by the hardware design of the processor.

Using the GPIOB_ODR register

We used the GPIOB_ODR register as an example to demonstrate how to access registers in C code. The GPIOB_ODR register is a RW register that controls the output data value of the GPIOB port.

To access the GPIOB_ODR register, we used the following steps:

1. Obtained the base address of the GPIOB peripheral from the device's datasheet.
2. Created a pointer to the GPIOB_ODR register using the base address and the appropriate offset.
3. Used the pointer to read and write the value of the GPIOB_ODR register.

Other ways to access registers

We also discussed other ways to access registers in C code, including:

- Using bit fields
- Using a structure
- Using a union

DMA and memory access

We briefly touched on the topic of Direct Memory Access (DMA). DMA is a technique that allows peripherals to access memory directly without involving the processor. This can improve the performance of I/O operations.

We also discussed how to configure the DMA controller to transfer data between memory and peripherals.

AMBA bus protocol

We concluded the lecture by discussing the AMBA bus protocol. AMBA is an open standard that defines a common interface between peripherals and processors. This allows peripherals from different manufacturers to be used with different processors.