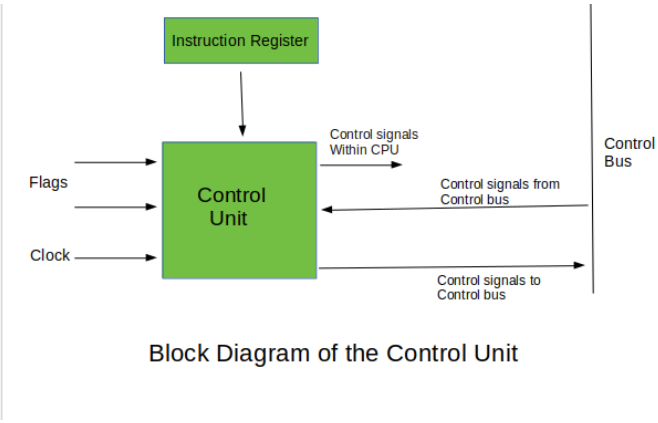
Chapter 2.1 Design of control unit



Block Diagram of the Control Unit

Control unit is responsible for controlling the overall operation of the CPU. It is responsible for fetching instructions from memory, decoding them, and executing them. There are two types of control units: hardwired control units and microprogrammed control units.

Hardwired Control Unit

- *Uses a combinational logic circuit to generate control signals for instruction execution*
- *Control signals are generated using a fixed set of gates and flip-flops*
- *Fast, simple, and reliable*
- *Inflexible and cannot be easily modified or updated*
- *Suitable for simple and well-defined instruction sets*

Advantages of Hardwired Control Unit:

- *Fast and simple design*
- *Reliable operation*
- *Lower cost and complexity*
- *Suitable for simple and well-defined instruction sets*

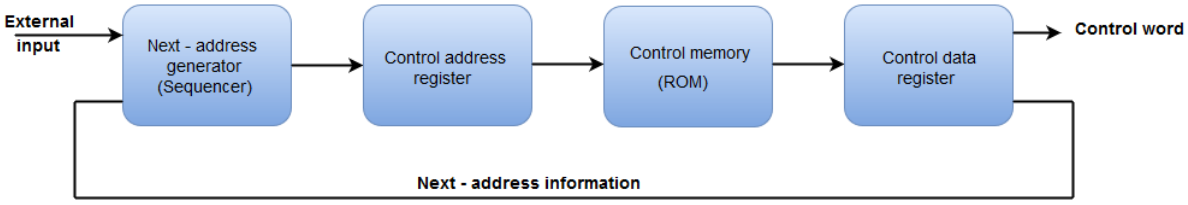
Disadvantages of Hardwired Control Unit:

- *Inflexible and cannot be easily modified or updated*
- *Difficult to implement for complex instruction sets*
- *Larger hardware footprint*
- *Limited functionality*

Microprogrammed Control Unit:

- *Uses microcode to generate control signals for instruction execution*
- *Microcode is stored in a ROM or a RAM*
- *Microcode is executed by a control unit called a micro sequencer*
- *More flexible than hardwired control units and can be easily modified or updated*
- *Slower and more complex than hardwired control units*
- *Suitable for complex instruction sets and applications that require flexibility and adaptability*

Microprogrammed Control Unit of a Basic Computer:



Advantages of Microprogrammed Control Unit:

- *More flexible and adaptable than hardwired control units*
- *Can be easily modified or updated*
- *Easier to design for complex instruction sets*
- *Smaller hardware footprint*
- *Can be easily upgraded to support new instructions or features*

Disadvantages of Microprogrammed Control Unit:

- *Slower than hardwired control units*
- *More complex design*
- *Higher cost and power consumption*
- *Microcode can introduce errors if not properly designed and tested.*

Chapter 2.2 Memory organization

Memory Hierarchy:

- *Memory hierarchy is a concept of organizing memory in a computer system in different levels of speed and size.*
- *The levels of memory hierarchy include registers, cache memory, main memory, and secondary storage devices such as hard disk drives and solid-state drives (SSDs).*
- *The purpose of the memory hierarchy is to provide the CPU with quick access to frequently used data while keeping the cost of the memory system reasonable.*

Cache Memory:

- *Cache memory is a type of fast memory that is used to temporarily store frequently accessed data and instructions from main memory.*
- *Cache memory is smaller in size but faster in access speed than main memory.*
- *Cache memory works by storing a copy of frequently accessed data in a faster memory location, reducing the number of memory accesses required to fetch the data.*
- *Cache memory is typically implemented using SRAM technology, which is faster but more expensive than DRAM technology.*

Associative Memory:

- *Associative memory is a type of memory that allows data to be accessed by content instead of by address.*
- *In an associative memory, data is stored with a tag, which is a unique identifier associated with the data.*
- *To retrieve data from an associative memory, the search is performed using the data content rather than the memory address.*

Cache Size vs Block Size:

- *Cache size refers to the amount of data that can be stored in cache memory.*
- *Block size refers to the size of the data block that is stored in cache memory.*
- *A larger cache size increases the hit rate but also increases the cache access time and the cost of the memory system.*
- *A larger block size reduces the number of cache misses but also increases the waste of cache memory space due to storing more data than needed.*

Mapping Functions:

- *Mapping functions are used to determine the location of data blocks in cache memory.*
- *The most common mapping functions are direct mapping, set-associative mapping, and fully-associative mapping.*
- *Direct mapping maps each block to a unique location in cache memory.*
- *Set-associative mapping maps each block to a set of locations in cache memory.*
- *Fully-associative mapping allows each block to be stored in any location in cache memory.*

Replacement Algorithms:

- *Replacement algorithms are used to decide which block of data should be replaced when the cache is full and a new block of data needs to be stored.*
- *The most common replacement algorithms are Least Recently Used (LRU), First-In-First-Out (FIFO), and Random Replacement.*
- *LRU replaces the block that has not been accessed for the longest time.*
- *FIFO replaces the block that has been in the cache for the longest time.*
- *Random Replacement replaces a random block from the cache.*

Write Policy:

- *Write policy is used to determine when and how data is written back to main memory after it has been modified in cache memory.*
- *The most common write policies are Write-Through and Write-Back.*
- *Write-Through writes data back to main memory immediately after it is modified in cache memory.*
- *Write-Back writes data back to main memory only when the block is replaced or the cache is flushed.*

Basic Optimization Techniques in Cache Memory:

- *Increasing the cache size.*
- *Increasing the associativity of the cache.*
- *Using multiple levels of cache memory.*
- *Using a combination of mapping functions and replacement algorithms.*
- *Using prefetching techniques to predict which data will be accessed next and fetch it into cache memory in advance.*

Cache Memory with Associative Memory:

- *Cache memory can be combined with associative memory to create a hybrid memory system that takes advantage of the strengths of both types of memory.*
- *In a cache memory with associative memory, the associative memory is used to perform fast searches for frequently accessed data, while the cache memory is used to temporarily store the data for fast access.*
- *The hybrid memory system can improve the hit rate and access speed while keeping the cost of the memory system reasonable.*

Virtual Memory:

- *Virtual memory is a technique that allows a computer to use more memory than physically available by temporarily transferring data from main memory to secondary storage.*
- *Virtual memory provides the illusion of a larger memory space by dividing the memory space into smaller pages or segments.*
- ***Paging** and **segmentation** are the two common techniques used for implementing virtual memory.*

1) Paging:

- *Paging is a technique used for implementing virtual memory that divides the memory space into fixed-size pages.*
- *Each page is mapped to a frame in main memory, and the mapping is stored in a page table.*
- *When a process needs to access a page that is not in main memory, the page is swapped into a free frame, and the mapping is updated in the page table.*

2) Segmentation:

- *Segmentation is a technique used for implementing virtual memory that divides the memory space into variable-size segments.*
- *Each segment is mapped to a contiguous block of memory in main memory, and the mapping is stored in a segment table.*
- *Segmentation allows for more efficient use of memory by allocating memory segments of different sizes to different processes based on their requirements*
-

*In summary, the **memory hierarchy** is a system of organizing memory in a computer system in different levels of speed and size. **Cache memory** is a type of fast memory that temporarily stores frequently accessed data, while **associative memory** allows data to be accessed by content instead of by address. **Virtual memory** is a technique that allows a computer to use more memory than physically available by temporarily transferring data from main memory to secondary storage, and can be implemented using paging or segmentation. **To optimize cache memory performance**, techniques such as increasing the cache size, increasing the associativity, and using prefetching can be used.*

Chapter 2.3 Input output organization

Input/output (I/O) organization is the method of controlling the flow of data between input/output devices and the CPU of a computer system. Asynchronous data transfer is a method of data transfer where data is transferred without the use of a fixed clock signal.

Source Initiated

- *In source-initiated data transfer, the input/output device controls the transfer of data and signals the CPU when the transfer is complete.*
- *The CPU sends a command to the input/output device to transfer data, and the input/output device transfers data to the CPU when ready.*
- *This method is suitable for low-speed devices and devices that have a simple interface.*

Example 1: *A mouse sending data to the CPU only when it has new data available to send.*

Example 2: *A digital camera transferring data to a computer only when the user initiates a transfer.*

Destination Initiated

- *In destination-initiated data transfer, the CPU initiates the transfer of data and signals the input/output device when ready to receive data.*
- *The input/output device waits for a signal from the CPU indicating that it is ready to receive data, and then transfers data to the CPU.*
- *This method is suitable for high-speed devices and devices that have a complex interface.*

Example 1: *A disk controller initiating data transfer only when the CPU signals that it is ready to receive data.*

Example 2: *A sound card initiating the transfer of audio data to the CPU only when the CPU is ready to process the data.*

Handshaking

- *Handshaking is a method of asynchronous data transfer where the input/output device and the CPU exchange signals to control the transfer of data.*
- *In this method, the input/output device sends a signal to the CPU indicating that it is ready to transfer data, and the CPU responds with a signal indicating that it is ready to receive data.*

- *The input/output device then transfers data to the CPU, and the CPU sends a signal indicating that the data has been received.*
- *This method is suitable for devices that require data transfer control.*

Example 1: *A serial communication device waiting for the CPU to signal that it is ready to receive data before sending data.*

Example 2: *A printer waiting for the CPU to signal that it is ready to receive data before printing a document.*

Programmed I/O

- *Programmed I/O is a method of data transfer where the CPU controls the transfer of data by reading and writing data directly to and from the input/output device.*
- *The CPU uses a series of input/output instructions to transfer data to and from the input/output device.*
- *This method is suitable for devices that have a simple interface and low-speed devices.*

Example 1: *Reading data from a keyboard using an input instruction and writing data to a printer using an output instruction.*

Example 2: *A microcontroller reading sensor data from an ADC and storing it in memory using input/output instructions.*

Interrupts

- *Interrupts are a method of data transfer where the input/output device interrupts the CPU when it is ready to transfer data.*
- *The input/output device sends an interrupt signal to the CPU, and the CPU saves its current state and starts processing the interrupt request.*
- *The input/output device then transfers data to the CPU, and the CPU resumes its previous task.*
- *This method is suitable for devices that require fast and efficient data transfer.*

Example 1: *A network card interrupting the CPU when a packet has been received and needs to be processed.*

Example 2: *An audio card interrupting the CPU when it has finished playing a sound.*

DMA

- *DMA (Direct Memory Access) is a method of data transfer where a DMA controller transfers data directly between the input/output device and main memory without the intervention of the CPU.*
- *The CPU sets up the DMA controller to transfer data, and the DMA controller handles the transfer of data between the input/output device and main memory.*
- *This method is suitable for devices that require high-speed data transfer and devices that need to transfer large amounts of data.*

Example 1: *A hard disk controller transferring data from the hard disk to main memory using a DMA controller.*

Example 2: *A graphics card transferring large amounts of image data from its memory to main memory using a DMA controller.*

IOP

- *IOP (Input/Output Processor) is a dedicated processor that controls the transfer of data between input/output devices and main memory.*
- *The IOP communicates with the CPU to transfer data between input/output devices and main memory.*
- *This method is suitable for devices that require high-speed data transfer and devices that need to transfer large amounts of data.*

Example 1: *A RAID controller managing multiple hard disks and handling the storage and retrieval of data on its own.*

Example 2*: A network controller that manages packet processing, including checksum calculation and packet routing.*