MICROPROCESSOR SYSTEMS

Mitchell Aaron Thornton,
Department of Electrical and Computer Engineering, Mississippi State University, PO Box 9571, Mississippi State, MS, 39762-9571, United States.

Keywords: Computer, Microprocessor, Fetch and Execute Cycle, Memory, Peripheral Device, Register, Hardware, Software, Synchronous, Asynchronous, Bus, Network, Transaction, Interrupt, Bandwidth, Latency, Microcontroller, Multimedia

Contents

Summary

The purpose of a microprocessor is to perform mathematical calculations (computations) in an artificial manner. The combination of a microprocessor with memory and peripherals results in a system that is designed to be useful for controlling other devices or for performing computations quickly. This combination is a microprocessor system and is the focus of this article. The principle components of a microprocessor system as well as their interaction and interconnection are discussed. Included is a discussion of the system software necessary for a microprocessor system to be useful.

Later sections include more focused discussions on differing types of system configurations including those designed for specific as well as general applications. Particular emphasis is given to various methods for interconnecting system components, often referred to as "interfaces".

1. Introduction

A complete computer system can be considered as a machine that "computes" or performs calculations. In the 19th century, Charles Babbage envisioned these types of machines as a complicated series of mechanical cogs and levers. Later, in the mid-20th century, Alan Turing developed a mathematical model of a computational machine based on the notion of accessing a memory system in a sequential manner and containing a stored computer program. Near the same time, the first electronic computer systems were developed and were comprised of large banks of vacuum tubes. Since this period in time, society has witnessed rapid advances in computer system implementation methods and performance. In 1971, the first central processing unit to be entirely contained on a single integrated circuit, the 4004, was produced by Intel. Due to the small size of an integrated circuit, this device was dubbed the *microprocessor*. Computer systems that are based on the use of microprocessors are referred to as *microprocessor systems*. For more information on the history and models of computation, see *History of Computation and Models of Computation.*

One necessary component of a microprocessor system in addition to the microprocessor integrated circuit is a set of peripherals. The microprocessor alone is a device that is generally designed as a circuit that repeatedly retrieves an instruction from an external memory source and then performs the operation indicated. This endless *fetch and execute cycle* is performed by the microprocessor until power is withdrawn or a signal indicating that it should cease this cycle is present. For the microprocessor to perform useful work, it is necessary for devices to be present that allow humans to interact with it. Since electrical signals are used by the microprocessor as both input and output, it is necessary to integrate *peripheral devices* into a microprocessor system. Devices that are responsible for generating input signals to the microprocessor transform human stimuli such as pressing a key on a keyboard into an electrical signal. Alternatively, devices that provide microprocessor output to a human may are designed to transform electrical signals into signals perceptible by humans such as the visible light from a monitor.

Microprocessor systems require memory for the storage of instructions and input data and for the storage of computational results. All microprocessors have a very small amount of memory referred to as *registers* or as a *register file*. Some also have larger amounts of memory integrated onto the system chip. Most microprocessor systems do have external memory that is typically implemented in several different technological forms. As a minimum, a microprocessor system must have enough memory to store a sufficient amount of instructions to perform useful work and to store the required amount of input and output data.

Another very important component of a microprocessor system is the *system software*, the set of instructions that govern system operation. Software must be present even when no useful work is being accomplished by the microprocessor system in order to maintain system stability. Likewise, software is typically used to translate electrical signals generated by a peripheral into signals that are compliant with the design of the microprocessor. Finally, it is necessary to have software available that can transform commands generated by a human into those that are understood by the microprocessor.

2. System Components

Microprocessor systems are traditionally described in terms of two major classes of components described as *hardware* or *software*. In general, the hardware class includes all components that have physical substance while the software portion is an instance of electrical or magnetic patterns of charge that represent information. Memory subsystems are a crucial portion of the hardware portion of a microprocessor system because they enable the storage and updating of the electrical or magnetic charges that represent the software portion of the system. Peripheral devices are important because they allow humans to interact with microprocessor systems. Figure 1 contains a block diagram of a typical microprocessor system.

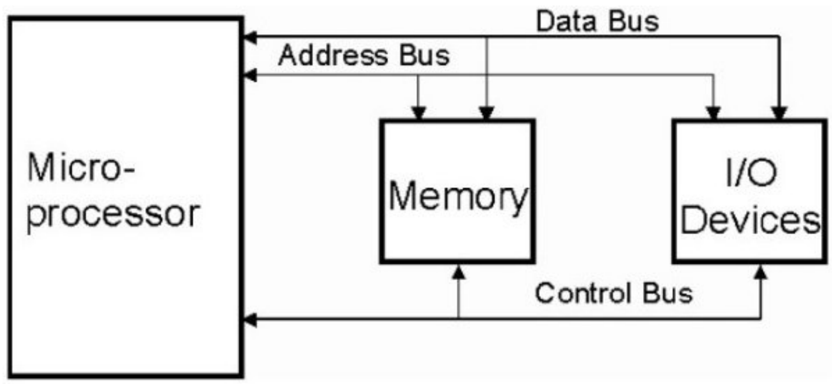

Figure 1: Block Diagram of Microprocessor System

2.1 Hardware Subsystem

The hardware subsystem is composed of the microprocessor, the memory devices, the peripheral (or I/O) devices and the media used to connect each of these. In Figure 1, the interconnection is accomplished by the use of *busses.* Busses are sets of wires that transmit data and control signals from one block to another. Depending on the type of information being transmitted, the busses have different names. The bus responsible for transmitting microprocessor instructions and data is the *data bus*. The *address bus* is used to specify the location of a particular piece of data or an instruction within the memory storage devices. The *control bus* transmits signals that synchronize internal communication among the various components of the microprocessor system. As an example, a particular control signal may be asserted that indicates data is being transmitted from the microprocessor to the memory. A peripheral device could monitor this signal and would not attempt to also transmit data at the same time. More details are available about bus based computer systems in general in *Chapter Bus Architectures*.

2.1.1 Microprocessor

The microprocessor is the portion of the system that actually performs the task desired by the user. These tasks are performed by specifying a sequence of operations in the form of a *program*. Virtually all microprocessors are synchronous circuits; that is, they utilize a periodic clock waveform that governs when the operations present in a program may be performed. The program consists of a set of instructions that dictate the particular operation and the order in which those operations are performed. Programs are stored in the memory hardware of the microprocessor system. Based on the value of the clock signal at any particular instance in time, the microprocessor retrieves an instruction from a program and then performs the operation that it represents. This is known as the *fetch-and-execute cycle.* Details of the internal organization and operation of microprocessors are given in section (see *Microcontrollers*)

2. 1.2. Memory

All microprocessor systems have some form of memory for the storage of instructions and data. There are many different kinds of memory with various characteristics. One way to classify different kinds of memory is through a memory hierarchy model that classifies the differing types based on the characteristics of access time, storage density and cost. Ideally, the user of a microprocessor system would prefer to have a memory subsystem that is as inexpensive as possible, has a very large capacity and has a very fast access time. As is the case with many different types of system implementations, these characteristics represent design tradeoffs. Optimizing the memory system for one characteristic is often at the expense of degrading another. The memory hierarchy describes these tradeoffs and is a convenient way to introduce different types of memory technology. Figure 2 contains a diagram illustrating a sample memory hierarchy. It is noted that Figure 2 is intended to serve as a representative example only since many other memory storage devices also exist.

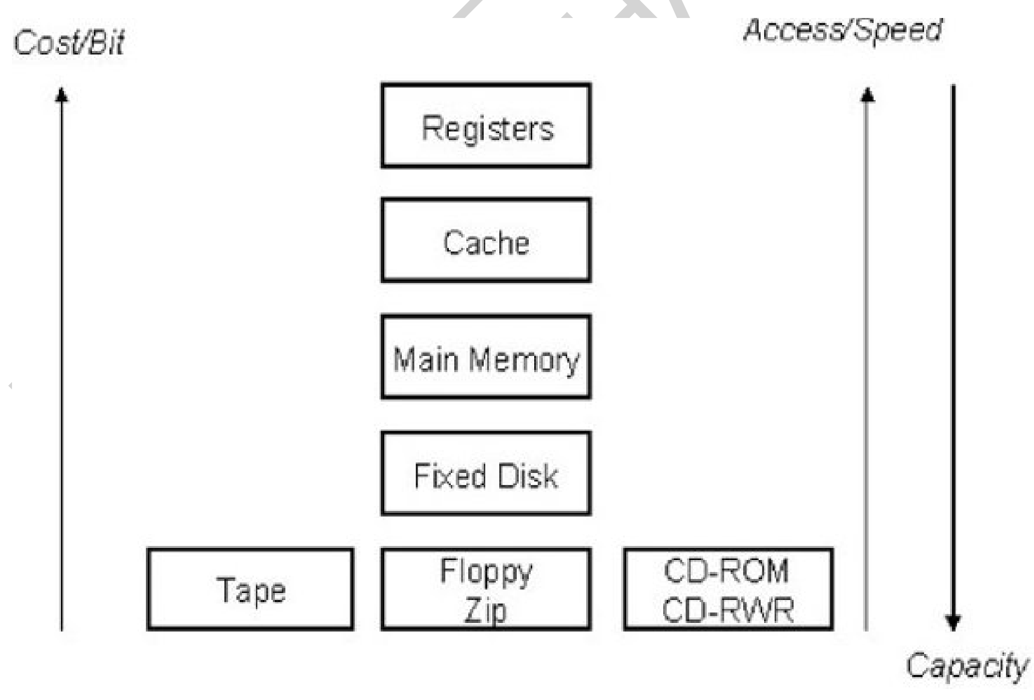

Figure 2: Diagram of the Memory Hierarchy in a Microprocessor System

Register memory represents the fastest and most flexible storage technology but incurs the greatest cost per information unit (bit) and the lowest capacity. Microprocessors have register memory circuits designed internally. All instructions that are executed are eventually transferred from a lower level of the memory hierarchy into a register. Additionally, most data that is manipulated or transformed is also resident in a register followed by a subsequent migration to a lower level of the memory hierarchy.

The blocks labeled *Cache* and *Main Memory* refer to storage units that are constructed from semiconductor devices and are present in electronic components that can be separate from the microprocessor circuit. A typical cache memory unit requires six transistors for each bit of storage while a similar storage cell in main memory requires only one or two transistors. Cache and main memory have more capacity than registers but less than a hard disk. However, the access time of these classes of memory is generally much smaller (thousands of times smaller) than that for a *fixed disk*. This is because instructions and data are stored as electronic charges and may be accessed in an electrical manner purely, whereas magnetic charges are used for the storage of fixed disk data, and the access is dependent on mechanical movement of the internal disk platter and heads in addition to electrical signals.

The next level in the hierarchy shown in Figure 2 is the *fixed disk*. A fixed disk is a type of memory that utilizes platters coated with magnetic material. The platter then rotates mechanically in close proximity to a sensor referred to as the read/write head. The read/write head is capable of sensing the polarity of a magnetized spot on the platter surface or of magnetizing a particular spot to some polarity. In this manner, data is read from or written to the fixed disk. Due to the involvement of mechanical movement, access times are much slower for this class of storage elements. However, the storage density and cost per bit are superior to that of the semiconductor memory cells.

The last level in the memory hierarchy is composed of storage units that use media such as *tape*, *floppy* or *zip* diskettes and *compact data* units (CDs). In the case of the tape, floppy and zip diskette examples, magnetic polarization is used to record the value of a single bit similar to the fixed disk. In the case of the tape system, purely sequential accesses are required. As an example, if a microprocessor system accesses a location at the beginning of the tape, a subsequent access to the end of the tape requires that the tape be forwarded to the end. This mechanical process can add significantly long access times to a tape storage subsystem. For this reason tape is generally used only for archival purposes rather than being used during the execution of a program. Floppy and zip diskettes operate in a manner that is very similar to the fixed disk; however, they are instances of removable storage media. The actual platters can be removed from the host system and replaced with other platters. This necessitates a looser requirement in the mechanical tolerances used to manufacture these types of storage subsystems that ultimately translates into slower access times and reduced capacity. The CD storage technology is based on optics for the storage of information and microprocessor instructions. In these types of systems, a platter is coated with a special material that is capable of either reflecting or absorbing the light from a laser. Depending on whether the incident light is reflected or absorbed, the value of a bit is determined to be either "1" or "0". CD-ROM (compact disk – read only memory) storage indicates that only a read operation can be performed. CD-RWR (compact disk – read/write recordable) indicates that the media may be reused and data may be written to the same location multiple times.

-
-
-

Bibliography

Aho A. V., Ravi S. and Ullman J. D. (1985), *Compilers: Principles, Techniques and Tools*, Addison-Wesley. [Compiler theory and design concepts]

Brey B. B. (2000), *The Intel Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, and Pentium Pro Processor Architecture, Programming, and Interfacing*, Prentice Hall. [Comprehensive description of programming and interfacing the x86 family of microprocessors]

Cady F. M. (1997), *Microcontrollers and Microcomputers Principles of Software and Hardware Engineering*, Oxford Press. [Introduction to microprocessor and microcontroller system architecture]

Miller M. A. (1992), *The 68000 Microprocessor Family*, Merrill Publishing. [Overview of programming and interfacing the popular 68000 series of microprocessors]

Patterson D. A. and Hennessy J. L. (1998), *Computer Organization & Design The Hardware/Software Interface*, Morgan Kaufman. [Comprehensive description of computer systems organization from both the hardware and software systems]

Rosch W. L. (1995), *Multimedia Bible*, Sams Publishing. [Good description of how multimedia peripherals operate in a microprocessor system]

Silberschatz A., Peterson J. and Galvin, P. (1991), *Operating System Concepts*, Addison-Wesley. [Introduction to the structure of operating systems and associated algorithms]

Tanenbaum A. S. (1992), *Modern Operating Systems*, Prentice Hall. [Introduction to the structure of operating systems and associated algorithms]

Tanenbaum A. S. (1999), *Structured Computer Organization*, Prentice Hall. [Overview of computer system architecture and interaction with operating systems]

Uffenbeck J. (1997). *The 80x86 Family:Design, Programming and Interfacing*, Prentice-Hall. [General background on the assembly language programming and the system interfaces of the popular Intel x86 series of microprocessors]