PART I COMPUTER HARDWARE

CHAPTER 1 PRINCIPLES OF COMPUTER ORGANIZATION

1.1 COMPUTER HARDWARE

We build computer to solve problems. Early computer solved mathematical and engineering problems, and later computers emphasized information processing for business applications. Today, computers also control machines as diverse as automobile engines, robots, and microwave ovens. A computer system solves a problem from any of these domains by accepting input, processing it, and producing output. Fig. 1-1 illustrates the function of a computer system.

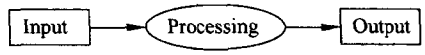

Fig. 1-1 The three activities of a computer system

Computer systems consist of hardware and software. *Hardware* is the physical part of the system. Once designed, hardware is difficult and expensive to change. *Software* is the set of programs that instruct the hardware and is easier to modify than hardware. Computers are valuable because they are general-purpose machines that can solve many different kinds of problems, as opposed to special-purpose machines that can each solve only one kind of problem. Different problems can be solved with the same hardware by supplying the system with a different set of instructions, that is, with different software.

Every computer has four basic hardware components:

- Input devices.
- Output devices.
- Main memory.
- Central processing unit(CPU).

Fig. 1-2 shows these components in a block diagram. The lines between the blocks represent the flow of information flows from one component to another on the *bus*, which is simply a group of wires connecting the components.[1] Processing occurs in the CPU and main memory. The organization in Fig. 1-2, with the components connected to each other by the bus, is common. However, other configurations are possible as well.

Computer hardware is often classified by its relative physical size:

- *Small* microcomputer;
- *Medium* minicomputer;

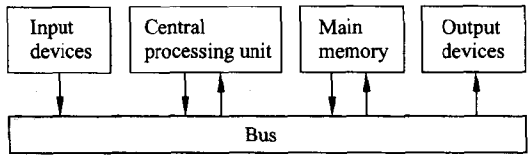

Fig. 1-2 Block diagram of the four components of a computer system

• *Large* mainframe.

Just the CPU of a mainframe often occupies an entire cabinet. Its input/output (I/O) devices and memory might fill an entire room. Microcomputers can be small enough to fit on a desk or in a briefcase. As technology advances, the amount of processing previously possible only on large machines becomes possible on smaller machines. Microcomputers now can do much of the work that only minicomputers or mainframes could do in the past.

The classification just described is based on physical size as opposed to storage size. A computer system user is generally more concerned with storage size, because that is a more direct indication of the amount of useful work that the hardware can perform. Speed of computation is another characteristic that is important to the user. Generally speaking, users want a fast CPU and large amounts of storage, but a physically small machine for the I/O devices and main memory.

When computer scientists study problems, therefore, they are concerned with space and time-the space necessary inside a computer system to store a problem and the time required to solve it. They commonly use the metric prefixes of Table 1-1 to express large or small quantities of space or time.

Table 1-1 Prefixes for powers of 10

Multiple	Prefix	Abbrev
10^9	giga-	G
10^6	mega-	M
10^3	kilo-	K
10^{-3}	milli-	m
10^{-6}	micro-	μ
10^{-9}	nano-	n

Example Suppose it takes 4.5 microseconds, also written 4.5 μs, to transfer some information across the bus from one component to another. (a) How many seconds are required for the transfer? (b) How many transfers can take place during one minute?[2]

(a) A time of 4.5μs is 4.5×10^{-6} from Table 1-1 or 0.000 004 5 s. (b) Because there are 60 seconds in one minute, the number of times the transfer can occur is (60 s)/(0.000 004 5 s/transfer) or 13 300 000 transfers. Note that since the original value was given with two significant figures, the result should not be given to more than two or three significant figures.

Table 1-1 shows that in the metric system the prefix kilo-is 1 000 and mega-is 1 000 000. But in computer science, a kilo-is 2^{10} or 1024. The different between 1 000 and 1 024 is less than 3%, so you can think of a computer science kilo-as being about 1 000 even though it is a little more. The same applies to mega-and giga-, as in Table 1-2. This time, the approximation is a little worse, but for mega-it is still within 5%.

Table 1-2 Computer science values of the large prefixes

Prefix	Computer science value
giga-	$2^{30}=1\ 073\ 741\ 824$
mega-	$2^{20}=1\ 048\ 576$
kilo-	$2^{10}=1\ 024$

NOTES

[1] 总线不仅仅是一组线缆,它还包括一些逻辑电路。此处是一种形象说法,有关总线的概念请见本章1.6节。

[2] 因为没有给出总线宽度,故每次传送可以理解为每秒钟或每分钟所传送的字节数、字数等。

KEYWORDS

computer	计算机	input device	输入设备
information processing	信息处理	output device	输出设备
hardware	硬件	main memory	主存储器
software	软件	central processing unit (CPU)	中央处理器
program	程序	bus	总线
general-purpose machine	通用(计算)机	microcomputer	微型计算机
special-purpose machine	专用(计算)机	minicomputer	小型计算机
instruction	指令	mainframe	主机,特大型机
set of instruction	指令集,指令系统		

EXERCISES

1. Multiple choices.

(1) When we store a problem into a computer, _____ is necessary.

a. space　　b. time

c. input device　　d. output device

(2) Early computer solved _____ problems.

a. control　　b. business applications

c. engineering　　d. mathematical

(3) We can use prefix micro to express _____.

a. time metric　　b. space metric

c. both time and space metrics　　d. 10^{-6}

(4) We can say a bus is simply _____.

a. a group of wires　　b. a wire

c. a 8-bit bus　　d. a 16-bit bus

(5) A computer system user generally more cares for _____.

a. physical size of the computer

b. storage size

c. speed of computation

d. efficiency of the computer

(6) According to the physical size of computers we can classify the computers into _____.

a. microcomputer　　b. minicomputer

c. mainframe　　d. supercomputer

(7) Prefix "mega" used for computer science is _____.

a. larger than 10^6　　b. smaller than 10^6

c. equal to 2^{20}　　d. 1 048 576

(8) The basic hardware components of any computer include _____.

a. CPU　　b. main memory

c. input devices　　d. output devices

2. Fill in the blank with appropriate words or phrases found behind this exercise.

(1) A computer system solves a problem by _____.

(2) The amount of effective work of a computer can be indicated by _____ directly.

(3) Computer systems consist of _____.

(4) Computer that can solve only one kind of problem is a _____.

(5) Computer that can solve many different kinds of problems is a _____.

(6) _____ instruct the hardware.

(7) _____ is difficult and expensive to change.

(8) We usually show the computer components in a _____.

a. general-purpose machine

b. hardware

c. accepting input, processing problem, and producing output

d. block diagram

e. software

f. storage size

g. special-purpose machine

h. hardware and software

1.2 COMPUTER CODES

Although the capacity of their storage locations can vary, every computer stores numbers, letters, and other characters in a coded form.

Every character in storage is represented by a string of 0s and 1s—the only digits found in the binary numbering system. Let's see how it's possible to use just two digits to represent any character.

Most computers use a coded version of true binary to represent decimal numbers.

Although many coding schemes have been developed over the years, the most popular of these use a binary coded decimal (BCD) approach.

The BCD Approach.

With BCD, it's possible to convert each decimal number into its binary equivalent rather than convert the entire decimal value into a pure binary form.

The BCD equivalent of each possible symbol is shown in Fig. 1-3.

Since 8 and 9 place value, all decimal digits are presented in BCD by 4 bits.

You've just seen that 202 is equal to 1100 1010 in a pure binary form.

Converting 202 into BCD, however, produces the following result: 202(10) in BCD = 0010 0000 0010 or 0010 0000 0010

Decimal Digit	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Fig. 1-3 BCD numeric bit configurations and their decimal equivalent

Zone bits		Numeric bits			
B	A	8	4	2	1

(a) Format for 6-bit BCD code

Character	Standard BCD interchange code
0	00 0000
1	00 0001
2	00 0010
3	00 0011
4	00 0100
5	00 0101
6	00 0110
7	00 0111
A	11 0001
B	11 0010
C	11 0011
D	11 0100
E	11 0101

(b) The coding used to represent selected characters in a standard 6-bit code

Fig. 1-4 The 6-bit BCD code

Six-bit BCD Code.

Instead of using 4 bits with only 16 possible characters, computer designers commonly use 6, 7, 8 bits to represent characters in alphanumeric versions of BCD.

In the 6-bit code, the four BCD numeric place positions (1, 2, 4, and 8) are retained, but two additional zone positions are included (Fig. 1-4(a)).

With 6 bits, it's possible to represent 64 different characters (2^6).

This is a sufficient number to code the decimal digits (10), capital letters (26), and other special characters and punctuation marks (28).

Fig. 1-4(b) shows you how a few of the 64 possible characters are represented in a standard 6-bit BCD code.

Seven and Eight-Bit Codes.

Since 64 possible bit combinations isn't sufficient to provide decimal numbers (10), lower-case letters (26), capital letters (26), and a large number of other characters (28), designers have extended the 6-bit BCD code to 7 and 8 bits.

With 7 bits, it's possible to provide 128 different arrangements (2^7); with 8 bits, 256 variations are possible (2^8).

In addition to the four numeric place positions, there are three zone bit positions in a 7-bit code, and four zone bit positions in an 8-bit code.

The 7-bit American Standard Code for Information Interchange (ASCII) is widely used in data communications work and is by far the most popular code used to represent data internally in personal computers.

The ASCII format and the coding used to represent selected characters are shown in Fig. 1-5.

These are other two 8-bit codes in common use.

One is the Extended Binary Coded Decimal Interchange Code (EBCDIC).

This code is used in IBM mainframe models and in similar machines produced by other manufactures.

The other 8-bit code is ASCII-8, an 8-bit version of ASCII that is frequently used in the larger machines produced by some vendors.

Fig. 1-6 presents the 8-bit format and shows how selected characters are represented in these 8-bit codes.

Zone bits			Numeric bits			
Z	Z	Z	8	4	2	1

(a) Format for 7-bit ASCII code

Character	ASCII
0	01 10000
1	01 10001
2	01 10010
3	01 10011
4	01 10100
5	01 10101
6	01 10110
7	01 10111
A	10 00001
B	10 00010
C	10 00011
D	10 00100
E	10 00101

(b) The coding used to represent selected characters in the ASCII 7-bit code

Fig. 1-5 The 7-bit ASCII format and selected ASCII character codes

Zone bits				Numeric bits			
Z	Z	Z	Z	8	4	2	1

(a) Format for 8-bit EBCDIC and ASCII-8 codes

Character	Extended BCD Interchange code (EBCDIC)	ASCII-8
0	1111 0000	0101 0000
1	1111 0001	0101 0001
2	1111 0010	0101 0010
3	1111 0011	0101 0011
4	1111 0100	0101 0100
5	1111 0101	0101 0101
6	1111 0110	0101 0110
7	1111 0111	0101 0111
A	1100 0001	1010 0001
B	1100 0010	1010 0010
C	1100 0011	1010 0011
D	1100 0100	1010 0100
E	1100 0101	1010 0101

(b) The coding used to represent selected characters in the EBCDIC and ASCII 8-bit codes

Fig. 1-6 The format for EBCDIC and ASCII 8-bit codes and selected characters represented by these codes. Each 8-bit unit used to code data is called a byte

The main difference is in the selection of bit patterns to use in the zone positions.

Detecting Code Errors.

Computers are very reliable, but they're not infallible.

If just one bit in a string of 6, 7, or 8 bits is lost during data input, processing, or output operations, an incorrect character code will be created.

Such an error can be caused by dust particles on storage media, by improper humidity levels near the computer, or by many other factors.

Fortunately, however, computer designers have developed a method for detecting such errors by adding an extra check bit or parity bit to each 6, 7, or 8-bit character represented in storage.

Thus, as you can see in Fig. 1-7, a total of 7, 8, or 9 bits may actually be stored.

The designers of a particular computer model may then use the check bit to make sure that every valid character code will always have an even number of 1 bits.

The 6-bit coding used to represent selected characters was presented earlier in Fig. 1-4.

Several of these characters have been reproduced in Fig. 1-8.

Check Zone

bit	bits		Numeric bits			
C	B	A	8	4	2	1

(a)

Check

Bit	bits		Numeric bits				
C	Z	Z	Z	8	4	2	1

(b)

Check

bit	Zone bits				Numeric bits			
C	Z	Z	Z	Z	8	4	2	1

(c)

Fig. 1-7

(a) Format of 6-bit BCD code with check bit included.
(b) Format of 7-bit ASCII code with check bit included.
(c) Format of 8-bit codes with check included.

Check Zone

	bit	bits		Numeric bits			
1	1	0	0	0	0	0	1
2	1	0	0	0	0	1	0
3	0	0	0	0	0	1	1
A	1	1	1	0	0	0	1

Fig. 1-8 Using the check bit to make sure that every character has an even number of 1 bits. This is an example of the use of an even-parity format. Other computers are also designed to have the check bit to produce an odd-parity format

You'll notice that if the basic code for a character such as 1, 2, or A requires an odd number of 1 bits, an additional 1 bit is added in the check-bit location so that there will always be an even number of such bits.

This is an example of an even-parity format, but other computers use the check bit to produce an odd parity.

Since every valid character in a computer that uses even parity must always have an even number of 1 bits, circuits for parity checking are built into the computer to constantly monitor the data moving through the system.

The computer operator is notified if a bit is lost and a parity error is detected. Of course, parity checking will only detect coding errors.

It cannot signal the fact that incorrect data have been entered into the system if the data are properly coded.

NOTES

[1] 短语 by far 修饰比较级与最高级，强调数量、程度等，意为"……得多，最……"，这是一个用 and 连接的并列句。

[2] to make sure 是不定式短语作目的状语，其后的 that 引导的是 make 的宾语从句。

KEYWORDS

Binary Coded Decimal(BCD)	二进制编码的十进制，二-十进制
ASCII	美国信息交换用标准代码
EBCDIC	扩展的二-十进制交换码
even-parity	偶检验
odd-parity	奇检验
parity checking	奇偶检验

EXERCISES

1. True / False.

(1) _____ The possible symbols in the binary numbering system are 0 to 9.

(2) _____ The decimal value of 16 is represented in 4 bit BCD as 00010101.

(3) _____ Alphanumeric versions of BCD commonly use 7, or 8 bits to represent characters.

(4) _____ A 6 bit alphanumeric code can represent 128 different characters.

(5) _____ There are four 8 bit codes in current use.

(6) _____ Each 8 bit unit used to code data is called a byte.

(7) _____ Eight bit codes are limited to representing 128 different characters.

(8) _____ An extra check (or parity) bit is often added to each 7, or 8 bit character represented in storage so that it will be possible to detect coding errors that may occur.

(9) _____ The digits that can be found in the binary numbering system are only 0s and 1s .

(10) _____ Using BCD we can convert the entire decimal value into a pure binary form.

2. Fill in the blanks with appropriate words or phrases found behind this exercise.

(1) Every computer stores numbers, letters and characters in a _____ form.

(2) The abbreviation BCB is derived from its full name _____.

(3) Using BCD we can convert each decimal number into its binary _____.

(4) In addition to four numeric place positions in an 8-bit code there are another four _____ positions.

(5) The most popular code used to represent data internally in personal computer is _____.

(6) The main difference between EBCDIC and ASCII-8 is _____ positions.

(7) Parity checking has two types, they are _____.

(8) Any character can be represented by using just _____ of 0 and 1.

a. equivalent

b. even-parity and odd-parity

c. ASCII

d. coded

e. two digits

f. patterns of zone

g. zone bit

h. binary coded decimal

1.3 WHAT IS A PROCESSOR

A processor is a functional unit that interprets and carries out instructions. Every processor comes with a unique set of operations such as ADD, STORE, or LOAD that represent the processor's instruction set. Computer designers are fond of calling their computers machines[1], so the instruction set is sometimes referred to as machine instructions and the binary language in which they are written is called machine language! You shouldn't confuse the processor's instruction set with the instructions found in high-level programming languages, such as BASIC or Pascal.

An instruction is made up of operations that specify the function to be performed and operands that represent the data to be operated on. For example, if an instruction is to perform the operation of adding two numbers, it must know (1) what the two numbers are and (2) where the two numbers are. When the numbers are stored in the computer's memory, they have their addresses to indicate where they are. So if an operand refers to data in the computer's memory it is called an address. The processor's job is to retrieve instructions and operands from memory and to perform each operation. Having done that , it signals memory to send it the next instruction.

This step-by-step operation is repeated over and over again at awesome speed. A timer called a clock releases precisely timed electrical signals that provide a regular pulse for the processor's work. The term that is used to measure the computer's speed is borrowed from the domain of electrical engineering and is called a megahertz (MHz), which means million cycles per second. For example, in an 8-MHz processor, the computer's clock ticks 8 million times to every 1 second tick of an ordinary clock.

A processor is composed of two functional units—a control unit and an arithmetic/logic unit—and a set of special workspaces called registers.

1. The Control Unit

The control unit is the functional unit that is responsible for supervising the operation of the entire computer system. In some ways, it is analogous to a telephone switch-board with intelligence because it makes the connections between various functional units of the computer system and calls into operation each unit that is required by the program currently in operation[2].

The control unit fetches instructions from memory and determines their types or decodes them. It then breaks each instruction into a series of simple small steps or actions. By doing this, it controls the step-by-step operation of the entire computer system.

2. The Arithmetic and Logic Unit

The arithmetic and logic unit (ALU) is the functional unit that provides the computer with logical and computational capabilities. Data are brought into the ALU by the control unit, and the ALU performs whatever arithmetic or logic operations are

required to help carry out the instruction[3].

Arithmetic operations include adding, subtracting, multiplying, and dividing. Logic operations make a comparison and take action based on the results. For example, two numbers might be compared to determine if they are equal. If they are equal, processing will continues; if they are not equal, processing will stop.

3. Registers

A register is a storage location inside the processor. Registers in the control unit are used to keep track of the overall status of the program that is running. Control unit registers store information such as the current instruction, the location of the next instruction to be executed, and the operands of the instruction. In the ALU, registers store data items that are added, subtracted, multiplied, divided, and compared. Other registers store the results of arithmetic and logic operations.

An important factor that affects the speed and performance of a processor is the size of the registers. Technically, the term word size (also called word length) describes the size of an operand register, but it is also used more loosely to describe the size of the pathways to and from the processor. Currently, word sizes in general purpose computers range from 8 to 64 bits. If the operand registers of a processor are 16 bits wide, the processor is said to be a 16-bit processor.

NOTES

[1] be fond of doing ... 是短语"乐于……,喜欢……";call computers machines 意为"把计算机称为机器"。

[2] because 后的原因状语从句,由 makes 和 calls 带出的两个并列分句组成。Calls into operation each unit 中的双宾语倒装,正常语序为 calls each unit into operation。原意为"传唤各部件进行操作",这里指微操作,实际上是指"完成微操作"。

[3] 这是一个 and 连接的并列句。后一个分句中的 whatever 是关系代词,引导后面的宾语从句。

KEYWORDS

instruction	指令	clock	时钟
instruction set	指令系统,指令集	megahertz (MHz)	兆赫
processor	处理器	control unit	控制器,控制部件
operation	操作、操作码、操作码指令	arithmetic and logic unit (ALU)	算术/逻辑部件
operand	操作数	word size (word length)	字长
register	寄存器	machine language	机器语言

EXERCISES

1. Match the following terms to the appropriate definition.

(1) _____ Processor.
(2) _____ Instruction set.
(3) _____ Clock.
(4) _____ Machine language.
(5) _____ Operation.
(6) _____ Operand.
(7) _____ Megahertz (MHz).
(8) _____ Control unit.
(9) _____ Arithmetic and logic unit (ALU).
(10) _____ Register.
(11) _____ Word size.

a. The part of an instruction that specifies the function that is to be performed.

b. The binary language in which a computer's instruction set is written.

c. A timer in a processor that releases precisely timed signals that provide a pulse for the processor's work.

d. A functional unit that interprets and carries out instructions.

e. A unique set of operations that comes with every processor.

f. The part of an instruction that tells where data that are operated on are located .

g. Million cycles per second.

h. The function unit that is responsible for supervising the operation of the entire computer system.

i. A function unit that provides the computer with logical and computational capability.

j. The term used to describe the size of operand registers and buses.

k. A storage location inside the processor.

2. Fill in the blanks with appropriate words or phrases

(1) We usually call our computers _____.

(2) An instruction set can sometimes be referred to as _____.

(3) The binary language is called _____.

(4) We don't confuse the processor's instruction set with the instructions of _____.

(5) An instruction consists of _____.

(6) An operand that refers to data in the memory is called an _____.

(7) A timer can give precisely timed _____.

(8) A processor includes two functional units, they are _____.

(9) The ways by which the control unit works are analogous to _____.

(10) The control unit takes out the _____ from memory.

a. address
b. high-level programming languages
c. instructions
d. machines
e. electrical signals
f. machine instructions
g. a telephone switch-board with intelligence
h. operations and operands
i. machine language
j. control unit and ALU

1.4 MEMORY SYSTEMS

1. Memory System Desiderata

The memory system has three desiderata.

(1) Size: infinitely large, no constraints on program or data set size.

(2) Speed: infinitely fast, latency equal to the fastest memory technology available[1].

(3) Cost: the per bit cost should approach the lowest-cost technology available.

Clearly these specifications cannot all be achieved as they are mutually exclusive. However, with the semiconductor and magnetic memory technology of today, these specifications are closely approximated.

2. Hierarchical Memory

In this section it is shown how designers implement a practical memory that approaches the performance of an ideal memory at reasonable cost. This memory system has a hierarchy of levels: The memory closest to the processor is fast and relatively

small, but has a high cost per bit. This level is called the cache; The real memory, sometimes known as main memory, is slower, larger, and has a lower cost per bit than the cache; The lowest level in the hierarchy is usually a magnetic disk that has the longest latency and the lowest bandwidth; however, it can be very large and has a very low cost per bit. This hierarchy is illustrated in Fig. 1-9.

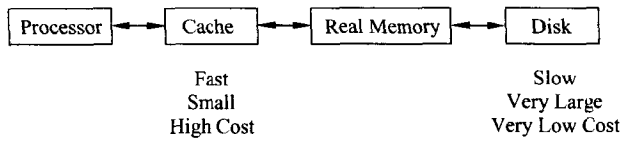

Fig. 1-9 Hierarchical memory

Note that Fig. 1-9 does not include the processor register file in the memory hierarchy.[2] The register file is a program-managed cache and is generally not included in the memory system. Also, there can be more than one cache in the hierarchy.

3. Paged Virtual Memory

Paged virtual memory provides the solution to the first desideratum of a very large memory's being available to the processor. Because of the importance of this desideratum, the relationship between virtual and real memory is discussed first. With virtual memory, the processor addresses the disk, the large, slow end of the hierarchy.[3] The memories between the processor and the disk are there to increase the effective performance (reduced latency and increased bandwidth) of the very slow disk. If every instruction and data reference were made to the disk, the processor performance would be slow indeed.

Then why is virtual memory so important? Large memory is needed for large programs and data sets. Early computers with small real memory required that the transfer of data between real memory and disk be managed explicitly by the operating system or the user. Virtual memory provides for automatic management of this portion of the memory hierarchy through a combination of hardware and software aids.

The virtual memory interface is shown in Fig. 1-10. A real memory of 16M bytes and a virtual memory of 2G bytes are shown for illustration; many modern virtual-memory systems are much larger than this. Virtual-memory space is divided into equal-sized groups called pages. A page in a modern computer is 1K, 2K, or 4K bytes. Real memory is also divided into the same equal-sized groups, called page frames. When information is moved between virtual-memory space and real-memory space, a complete page is moved.

4. Caches

Section 3 discussed how virtual memory extends the address space of a processor. However, the latency of real memory is too long to support high-performance processors. Even with the high-speed DRAMs used today for real memory, something must be done to overcome this latency problem.

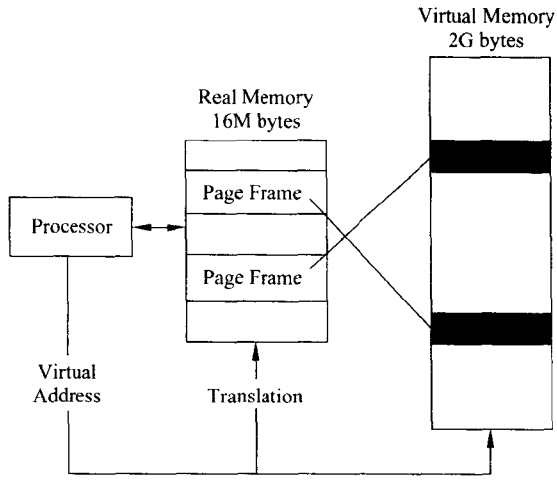

Fig. 1-10 Page allocation and address translation

The solution to this performance problem is to add another level to the memory hierarchy, called a cache, shown in Fig. 1-9. The allocation of spaces in a three-level memory is shown in Fig. 1-11. As discussed in Section 3, the virtual-address space is divided into equal-sized pages. These pages are placed in real-memory frames of the same size. Because a page can be placed in any vacant frame, there is no particular order to the allocation of pages to frames. With the addition of a cache, blocks of 16-32 bytes are taken from the page frames in real memory and placed into a block slot for access by the processor. For modern processors the cache usually has a latency of one processor clock, so that instructions and data may be fetched without delay except when the referenced item is not in the cache.

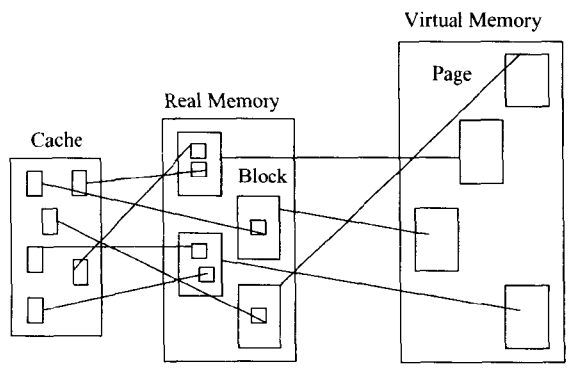

Fig. 1-11 Real memory and cache allocation

5. Memory Devices

1) RANDOM-ACCESS MEMORY

Random-access memory, or RAM, is the kind of memory we usually refer to when

we speak of computer memory. It is the most widely used type, and consists of rows of chips with locations established in tables maintained by the control unit[4].

As the name suggests, items stored in RAM can be gotten (accessed) both easily and in any order (randomly) rather than in some sequence. RAM relies on electric current for all its operations; moreover, if the power is turned off or interrupted, RAM quickly empties itself of all your hard work. Thus, we say RAM is volatile, or nonpermanent.

2) READ-ONLY MEMORY

Read-only memory, or ROM, typically holds programs. These programs are manufactured, or "hard-wired" in place on the ROM chips. For example, a microcomputer has a built-in ROM chip(sometimes called ROM BIOS, for ROM basic input/output system) that stores critical programs such as the one that starts up, or "boots", the computer. ROM is "slower" than RAM memory, and as a result, items in ROM are transferred to RAM when needed for fast processing.

Items held in ROM can be read, but they cannot be changed or erased by normal input methods. New items cannot be written into ROM. The only way to change items in most forms of ROM is to change the actual circuits.

3) MAGNETIC DISKS

The magnetic disk is a circular platter with a smooth surface and a coating that can be magnetized. Data is stored on it as magnetized spots. The reading and recording device, the disk drive, spins the disk past read/write heads that detect or write the magnetized spots on the disk.

4) CD-ROMS

Optical disks need thin beams of concentrated light to store and read data. It is a form of laser storage, called CD-ROM.

There are two types of optical disks that can be user-recorded: WORM and erasable optical. WORM stands for "write once, read many": Data can be written to this disk just one time, but the data can be read many times. Erasable optical disks can be written to, read, and erased.

5) MAGNETIC TAPE

A magnetic tape is a narrow plastic strip similar to the tape used in tape recorders. The tapes are read by tape drive moving the tape past a read/write head, which detects or writes magnetized spots on the iron-oxide coating of the tape. Each pattern of spots matches the byte code for character being stored.

NOTES

[1] latency 等待时间,潜伏时间。此处指存储器访问时间。

[2] register file 寄存器组。是作为指令或数据临时存放的一种多位寄存器组,有时称为栈。

[3] the processor addresses the disk 处理器对磁盘编址。实际上使用的是虚拟地址。

[4] 在控制器中有一张内存地址列表,此处 locations 翻译成为存储单元。

KEY WORDS

hierarchical memory 存储器层次结构

cache	高速缓冲存储器,高速缓存
magnetic disk	磁盘
main memory	主存储器
paged virtual memory	页式虚拟存储器
Random-Access Memory (RAM)	随机(访问)存储器
Read-Only Memory (ROM)	只读存储器
Compact Disk ROM (CD-ROM)	只读光盘
disk drive	磁盘驱动器
floppy disk (diskette)	软磁盘
write once, read many (WORM)	一次写多次读
magnetic tape	磁带
register file	寄存器组
latency	潜伏时间、等待时间
page frame	页帧
real memory (storage)	实存储器
Dynamic RAM (DRAM)	动态随机存储器

EXERCISES

1. Fill in the blanks from using the sentences and definitions found on text.

(1) There are three desiderata in the memory system, they are _____.

(2) Even with the high-speed DRAM as real memory, the high-performance processors still have the _____ problem.

(3) A microcomputer has a built-in ROM chip, it is called sometimes _____.

(4) Items stored in RAM can be accessed both easily and in _____.

(5) Magnetic disk has two types, they are _____.

(6) CD-ROM is abbreviated from _____.

(7) _____ detects or writes magnetized spots on the iron-oxide coating of the tape.

(8) The locations of a RAM are maintained by _____.

(9) Main memory is known as _____ sometimes.

(10) The processor addresses the disk with _____.

a. ROM BIOS	b. the control unit
c. Compact Disk-ROM	d. size, speed and cost
e. virtual memory	f. any order (randomly)
g. real memory	h. latency
i. read/write head	j. hard disks and floppy disks

2. Multiple choices.

(1) Cache is _____.

a. fast	b. slow
c. relatively small	d. high cost

(2) Information stored in semiconductor RAM is _____.

a. permanent	b. nonpermanent
c. volatile	d. non-volatile

(3) We use paged virtual memory to _____.

a. extend the size of memory	b. increase bandwidth of the disk
c. reduce latency of the disk	d. store large program and data set

(4) We read data on the disk by _____.

a. detecting the magnetized spots randomly

b. writing the magnetized spots
c. detecting the magnetized spots sequentially
d. moving the read/write head

(5) The three desiderata of memory system are _____.
a. independent
b. exclusive
c. closely approximated today
d. less approximated tomorrow

(6) The processor performance would be high as we reference instructions and data from _____.
a. hard disk
b. cache
c. floppy disk
d. memory

(7) Page frame is used in _____.
a. real memory
b. virtual memory
c. disk
d. cache

(8) The register file is _____.
a. included in the memory system
b. a program-managed cache
c. called as a stack sometimes
d. included in the CPU

1.5 INPUT/OUTPUT SYSTEM

Processors need a source of input data and a destination for output data. Processors also require storage space for large volumes of intermediate data. Thus the I/O system provides these storage facilities by connecting the processor's memory to the I/O devices. [1] These devices vary widely in latency and bandwidth; [2] thus, a major function of the I/O system is to match their latencies and bandwidths to that of the memory system. I/O devices consist of such diverse items as keyboards, modems, and hard disks. Each of these devices has its own latency and bandwidth requirements that must be satisfied for a properly functioning system.

The major tasks of the I/O system are the following:

(1) to establish the connection between the memory and the I/O devices;

(2) to synchronize and manage the control of the data transfer;

(3) to provide buffering when the source and sink bandwidths and latencies differ;

(4) to perform code conversions if required;

(5) to interrupt the processor when required;

(6) to terminate the operation when it is completed.

These tasks are accomplished by hardware and software, in varying amounts of each. The more hardware-oriented, the higher the performance and the higher the hardware cost. The more software-oriented, the lower the performance and the lower the hardware cost. Control can be classified into two broad categories: programmed I/O and coprocessor I/O.

Programmed I/O

Programmed I/O, also known as direct I/O, is accomplished by a program executing on the processor itself to control the I/O operations and transfer the data. With programmed I/O, shown in Fig. 1-12, there are several architectural registers (minimally an input register and an output register) that are addressed by special move instructions. [3]

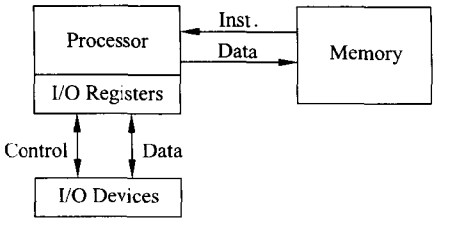

Fig. 1-12 Programmed I/O

These move instructions move information from the accumulator (or general-purpose register) to/from the I/O registers. The peripherals are connected directly to the bits of these registers. [4]

Memory-Mapped I/O

Memory-Mapped I/O is another form of programmed I/O that maps the device connections to bits in memory address space, as shown in Fig. 1-13. [5] An output command is a normal store instruction that stores a pattern of bits into a designated memory location. [6] An input command is a normal load instruction that reads a memory location.

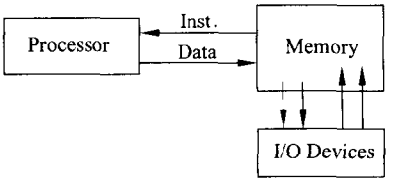

Fig. 1-13 Memory-mapped I/O

Memory-mapped I/O requires the full participation of the processor, as with programmed I/O. Thus there is a similar loss of efficiency that is due to executing the I/O control program.

The Intel Pentium Pro supports memory-mapped I/O in addition to programmed I/O. Any of the processor's instructions that reference memory can address memory-mapped I/O locations. An example of an allocation is shown in Fig. 1-14.

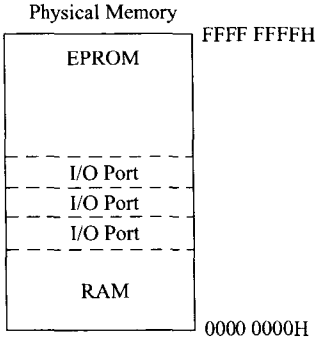

Fig. 1-14 Pentium Pro memory-mapped I/O Allocation