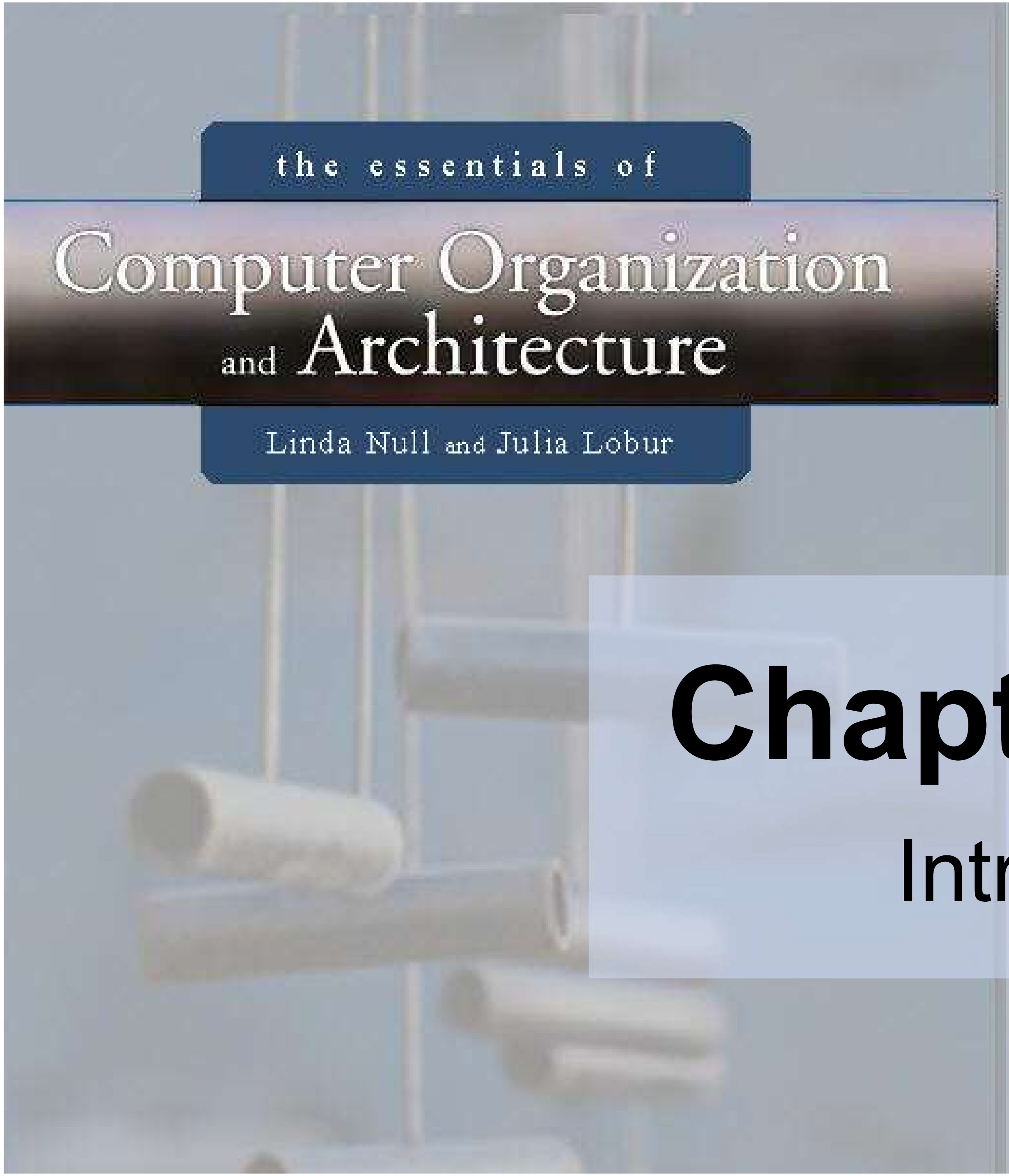


Chapter 1

Introduction

1.1 Overview

Why study computer organization and architecture?

– Design better programs, including system software such as compilers(hardware features), operating systems, and device drivers(I/O).

– Program optimization and system tuning.

– Evaluate (benchmark) computer system performance.

– Understand time, space, and price tradeoffs.

Chapter 1 Objectives

- Know the difference between computer organization and computer architecture.
- Understand units of measure common to computer systems.
- Appreciate the evolution of computers.
- Understand the computer as a layered system.
- Be able to explain the von Neumann architecture and the function of basic computer components.

1.1 Overview

- Computer organization
  - How various circuits and components fit together to create working computer systems
  - Encompasses all ***physical aspects*** of computer systems.
  - E.g., circuit design, control signals, memory types.
  - *How does a computer work?*
- Computer architecture
  - ***Logical aspects*** of system implementation as seen **by the programmer.**
  - E.g., instruction sets, instruction formats, data types, the number and types of registers, addressing modes, memory access methods, I/O mechanisms.
  - *How do I design a computer?*

1.2 Computer Components

- Principle of Equivalence of Hardware and Software:
  - ***Anything that can be done with software can also be done with hardware, and anything that can be done with hardware can also be done with software.***
  - hardware provides more speed, software provides more flexibility
  - They can solve problems equally, although solutions are often easier in one versus the other

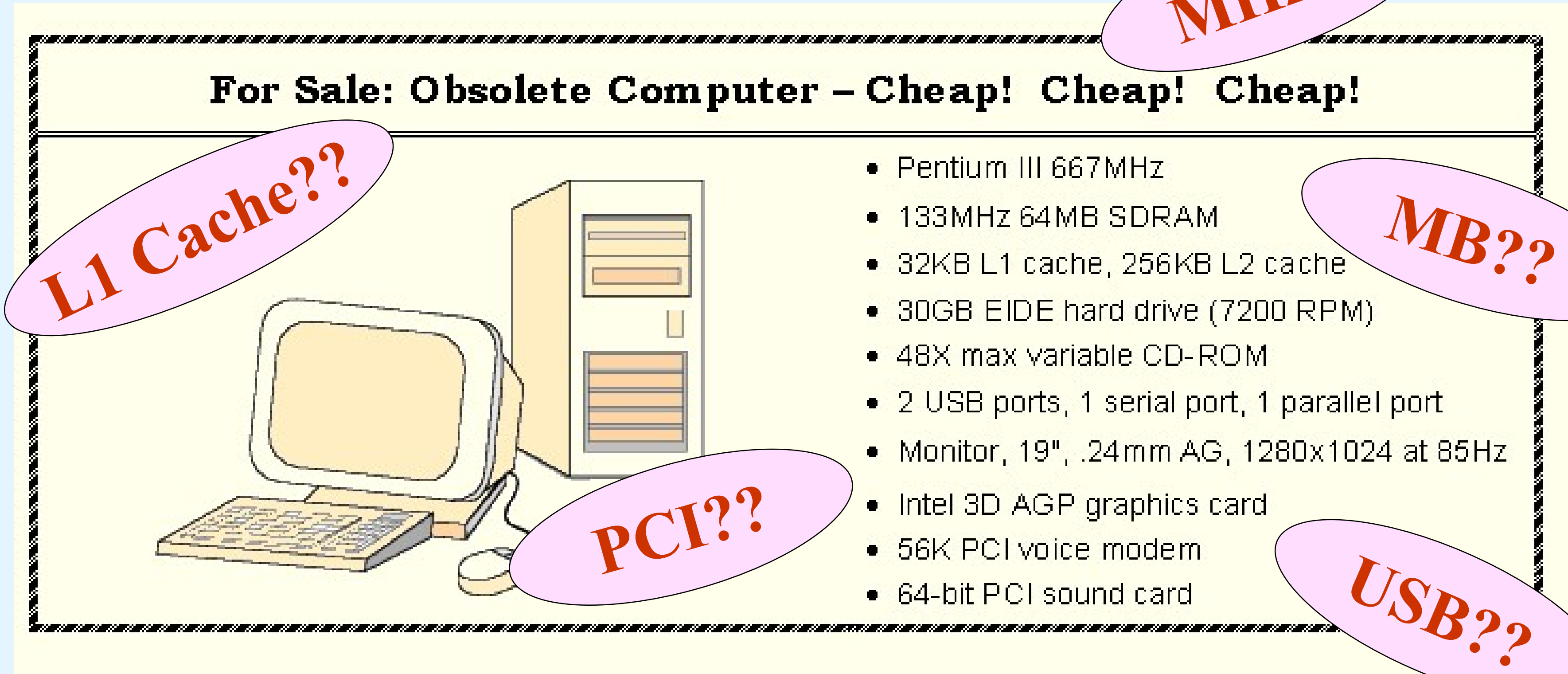
1.2 Computer Components

- At the most basic level, a computer is a device consisting of three pieces:
  - A processor to interpret and execute programs
  - A memory to store both data and programs
  - A mechanism for transferring data to and from the outside world.

  - Data processing, data storage, data transferring &controlling

1.3 An Example System

Consider this advertisement:



MHz??

L1 Cache??

MB??

PCI??

USB??

What does it all mean??

1.3 An Example System

Measures of capacity and speed:

- Kilo- (K) = 1 thousand = 10^3 and 2^{10}
- Mega- (M) = 1 million = 10^6 and 2^{20}
- Giga- (G) = 1 billion = 10^9 and 2^{30}
- Tera- (T) = 1 trillion = 10^{12} and 2^{40}
- Peta- (P) = 1 quadrillion = 10^{15} and 2^{50}

Whether a metric refers to a power of ten or a power of two <u>typically</u> depends upon what is being measured.

1.3 An Example System

- Hertz = clock cycles per second (frequency)
  - 1MHz = 1,000,000Hz
  - Processor speeds are measured in MHz or GHz.
- Byte = a unit of storage
  - 1KB = 2^{10} = 1024 Bytes
  - 1MB = 2^{20} = 1,048,576 Bytes
  - Main memory (RAM) is measured in MB
  - Disk storage is measured in GB for small systems, TB for large systems.

1.3 An Example System

Measures of time and space:

- Milli- (m) = 1 thousandth = 10^{-3}
- Micro- (μ) = 1 millionth = 10^{-6}
- Nano- (n) = 1 billionth = 10^{-9}
- Pico- (p) = 1 trillionth = 10^{-12}
- Femto- (f) = 1 quadrillionth = 10^{-15}

1.3 An Example System

- Millisecond = 1 thousandth of a second
  - Hard disk drive access times are often 10 to 20 milliseconds.
- Nanosecond = 1 billionth of a second
  - Main memory access times are often 50 to 70 nanoseconds.
- Micron (micrometer) = 1 millionth of a meter
  - Circuits on computer chips are measured in microns.

1.3 An Example System

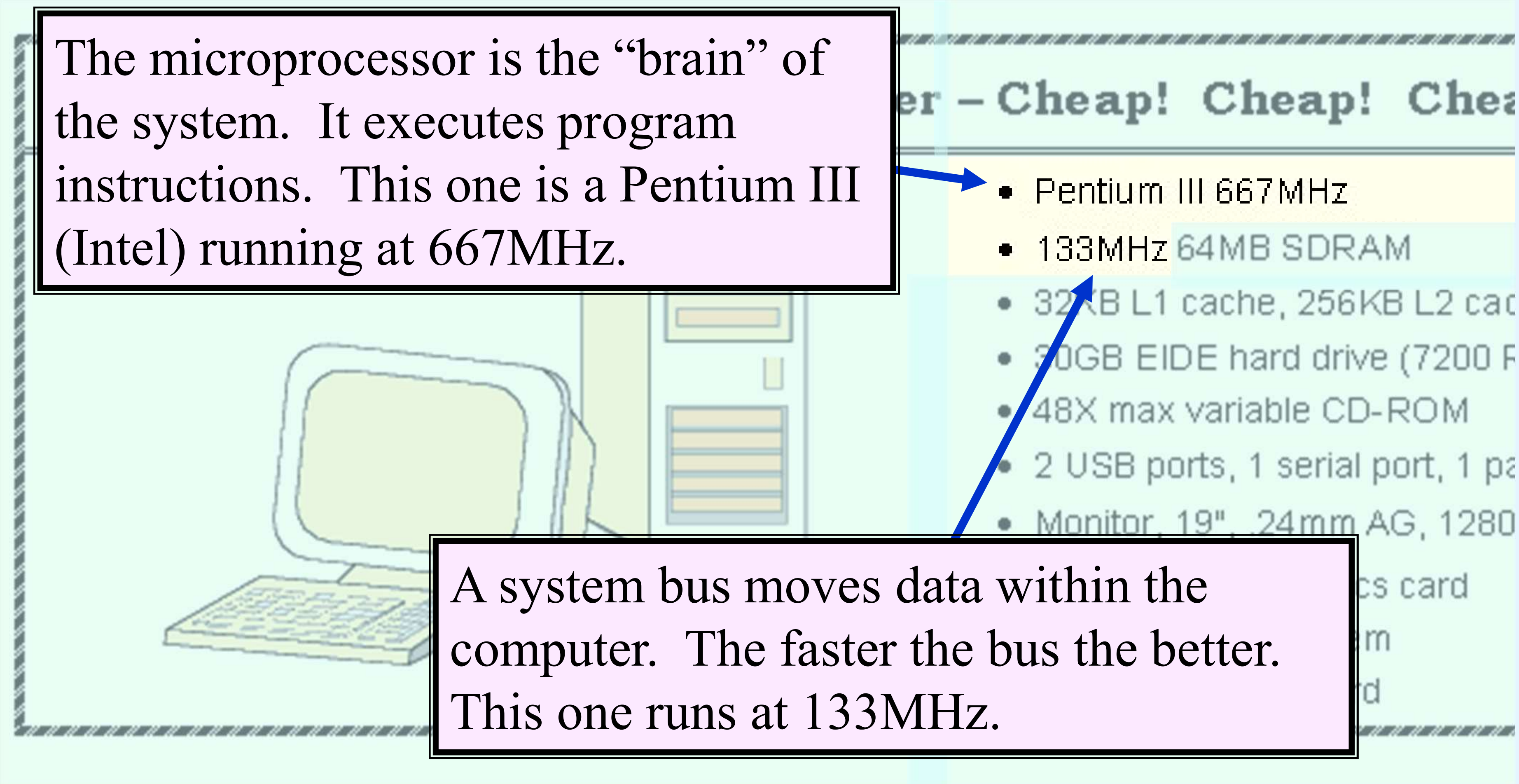
- We note that cycle time is the reciprocal of clock frequency.
- A bus operating at 133MHz has a cycle time of 7.52 nanoseconds:

$$133{,}000{,}000 \text{ cycles/second} = 7.52\text{ns/cycle}$$

Now back to the advertisement ...

1.3 An Example System

The microprocessor is the “brain” of the system. It executes program instructions. This one is a Pentium III (Intel) running at 667MHz.

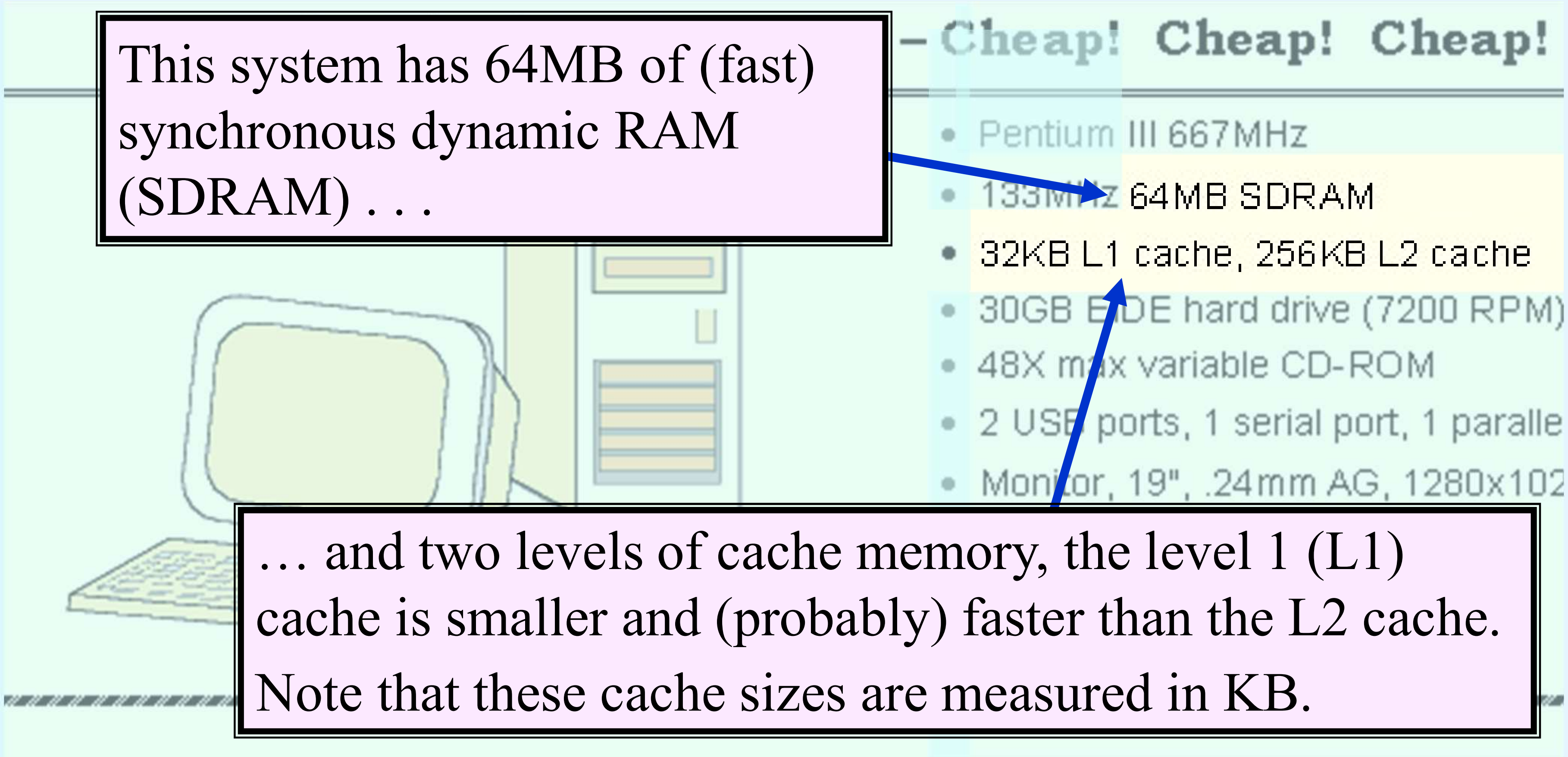


A system bus moves data within the computer. The faster the bus the better. This one runs at 133MHz.

1.3 An Example System

- Computers with large main memory capacity can run larger programs with greater speed than computers having small memories.
- RAM ： *random access memory*. Random access means that memory contents can be accessed directly if you know its location.
- Cache is a type of temporary memory that can be accessed faster than RAM.

1.3 An Example System

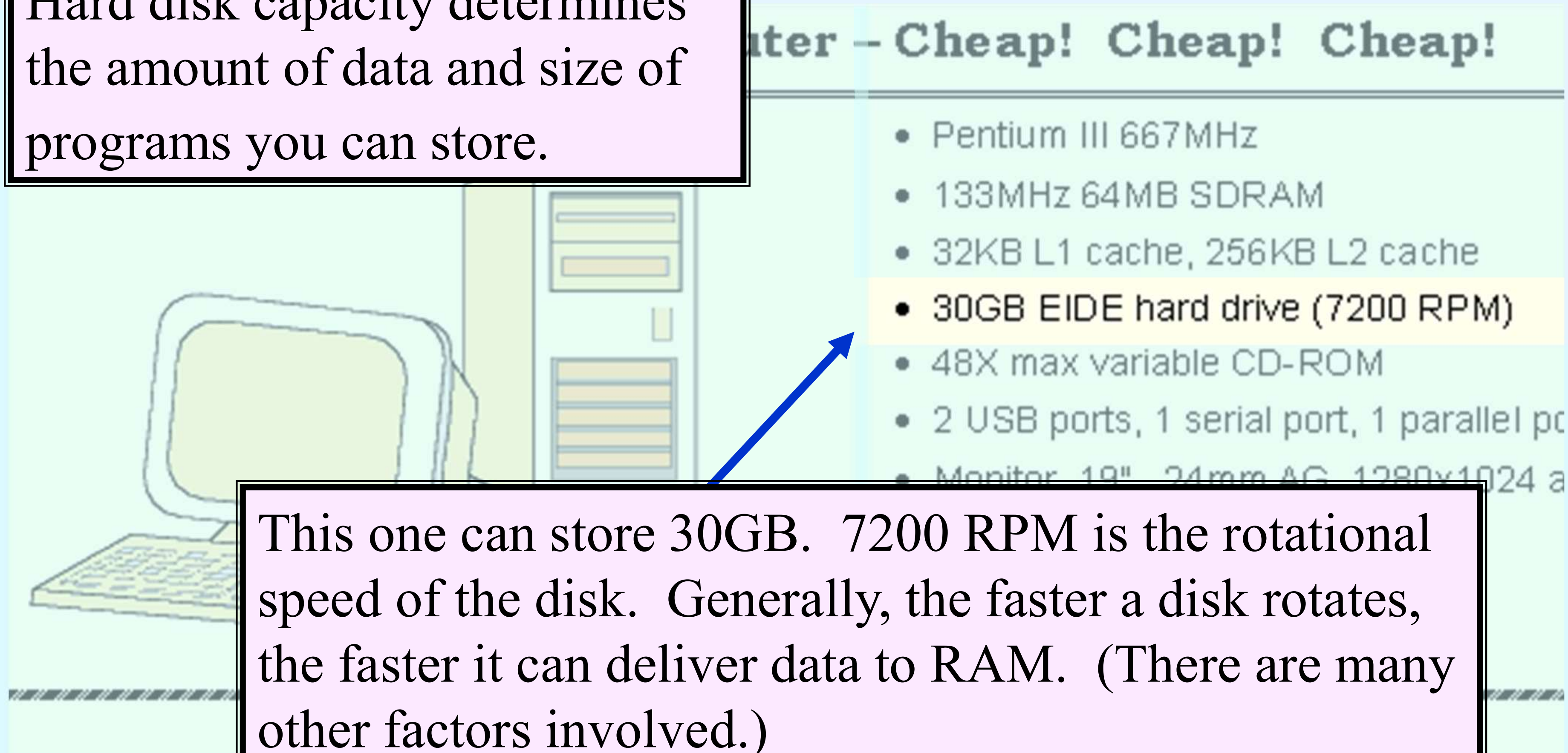


This system has 64MB of (fast) synchronous dynamic RAM (SDRAM) . . .

… and two levels of cache memory, the level 1 (L1) cache is smaller and (probably) faster than the L2 cache. Note that these cache sizes are measured in KB.

1.3 An Example System

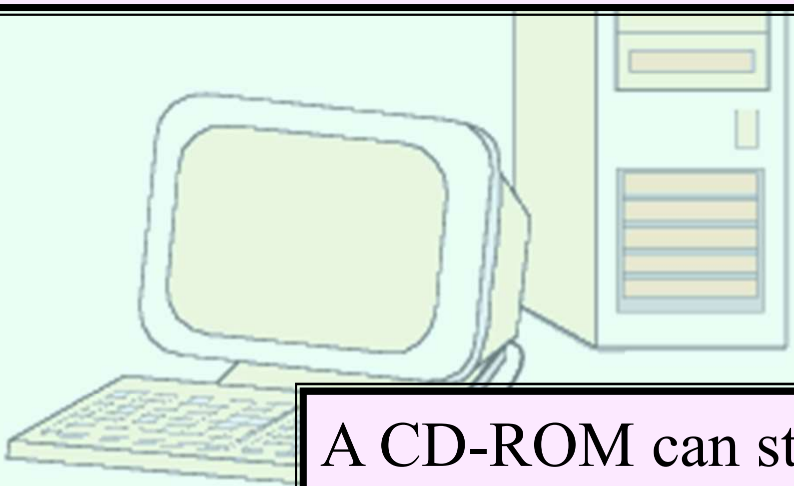
Hard disk capacity determines the amount of data and size of programs you can store.



This one can store 30GB. 7200 RPM is the rotational speed of the disk. Generally, the faster a disk rotates, the faster it can deliver data to RAM. (There are many other factors involved.)

1.3 An Example System

EIDE stands for *enhanced integrated drive electronics*, which describes how the hard disk interfaces with (or connects to) other system components.



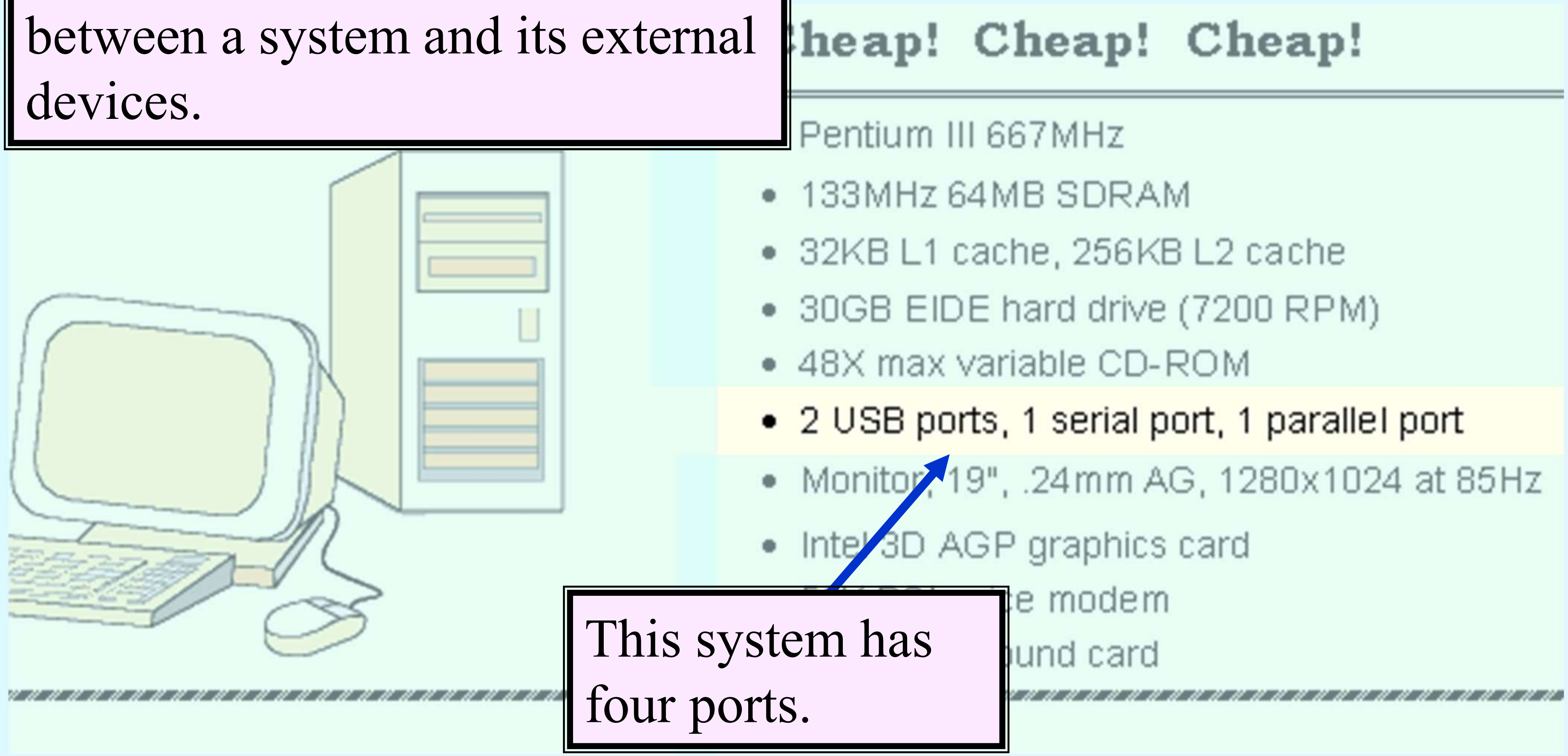
! Cheap!

- AM
- 32KB L1 cache, 256KB L2 cache
- 30GB EIDE hard drive (7200 RPM)
- 48X max variable CD-ROM
- 2 USB ports, 1 serial port, 1 parallel po
- Monitor, 19", .24mm AG, 1280x1024 a

A CD-ROM can store about 650MB of data, making it an ideal medium for distribution of commercial software packages. 48x describes its speed.

1.3 An Example System

Ports allow movement of data between a system and its external devices.



This system has four ports.

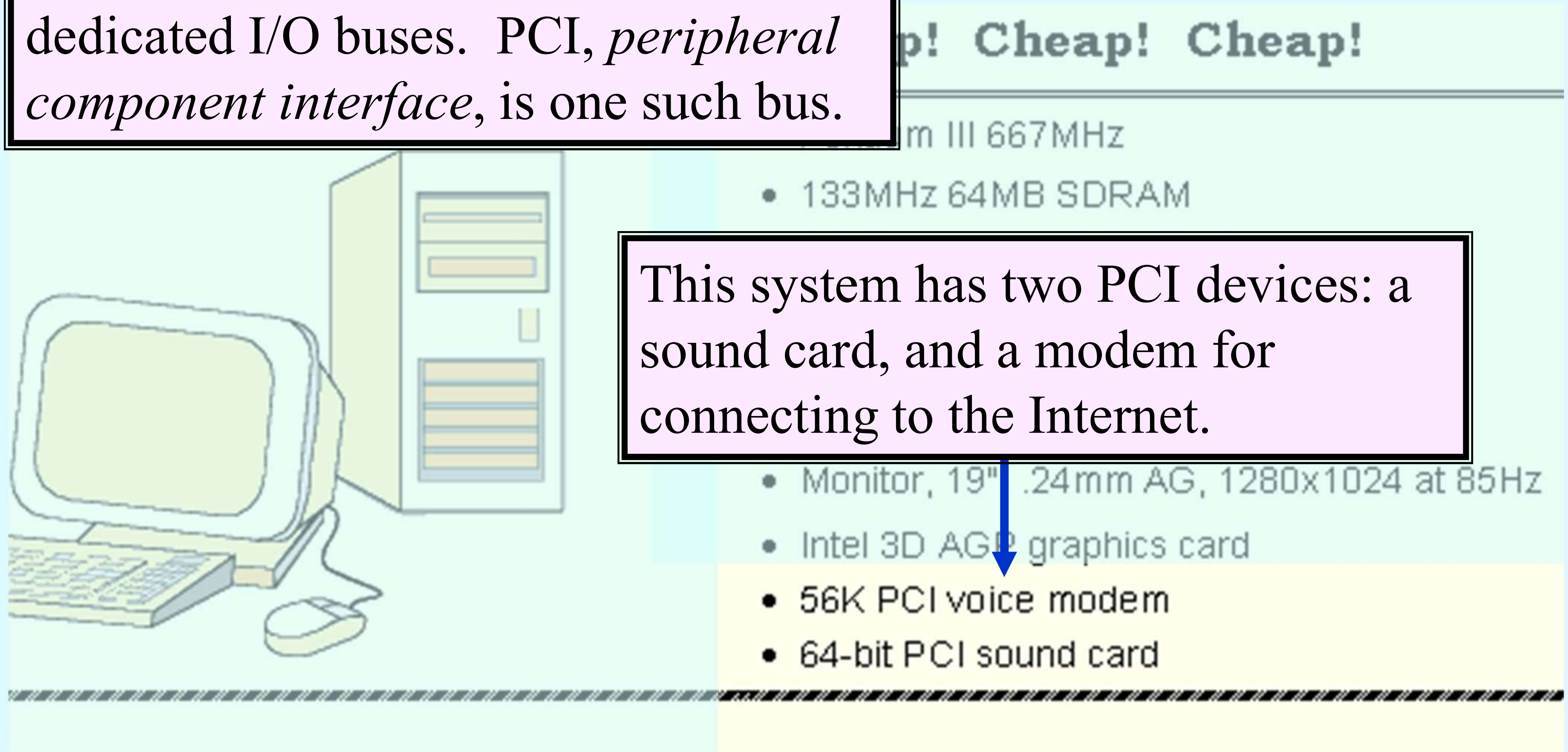
1.3 An Example System

- Serial ports send data as a series of pulses along one or two data lines.
- Parallel ports send data as a single pulse along at least eight data lines.
- USB, universal serial bus, is an intelligent serial interface that is self-configuring. (It supports "plug and play.")

1.3 An Example System

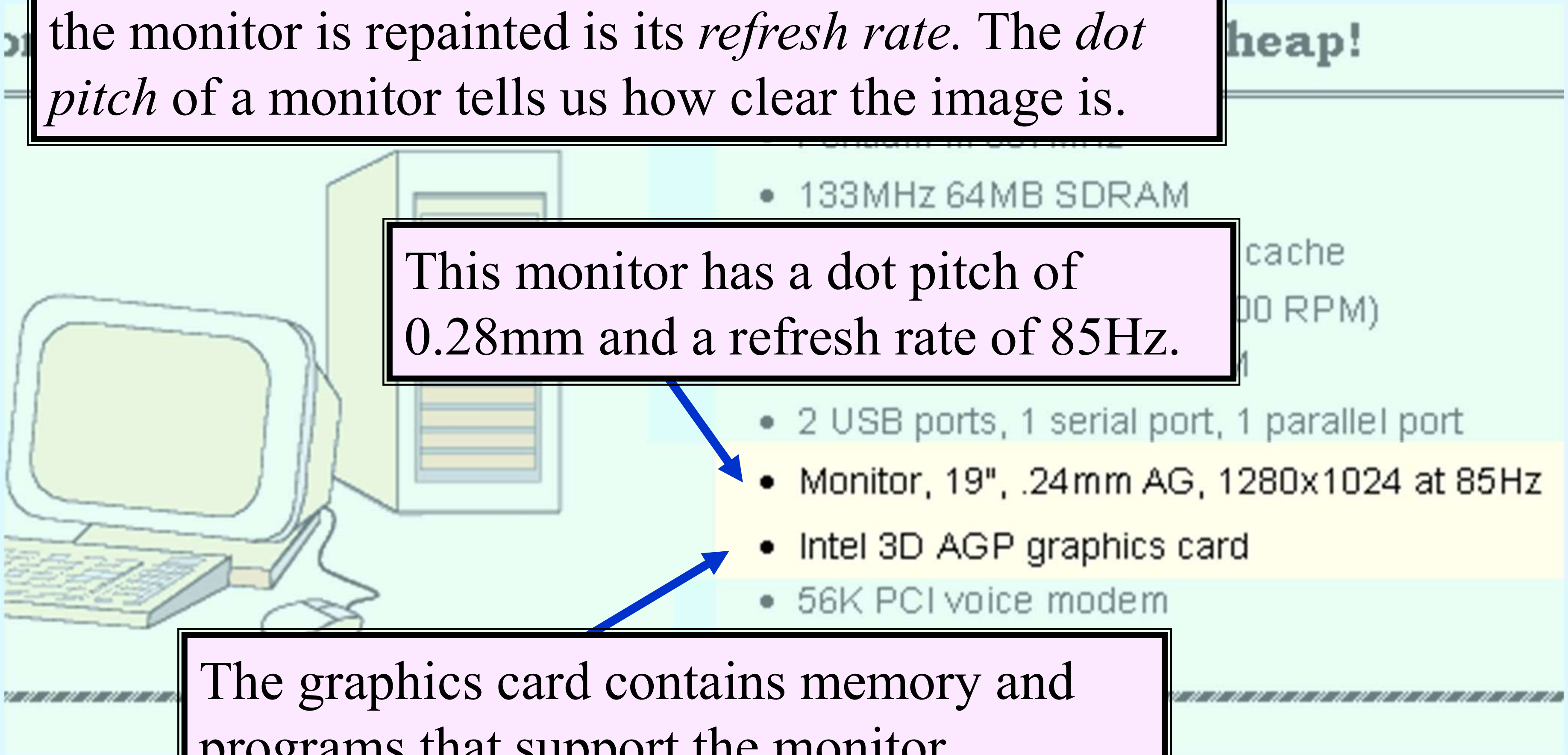
System buses can be augmented by dedicated I/O buses. PCI, *peripheral component interface*, is one such bus.

This system has two PCI devices: a sound card, and a modem for connecting to the Internet.



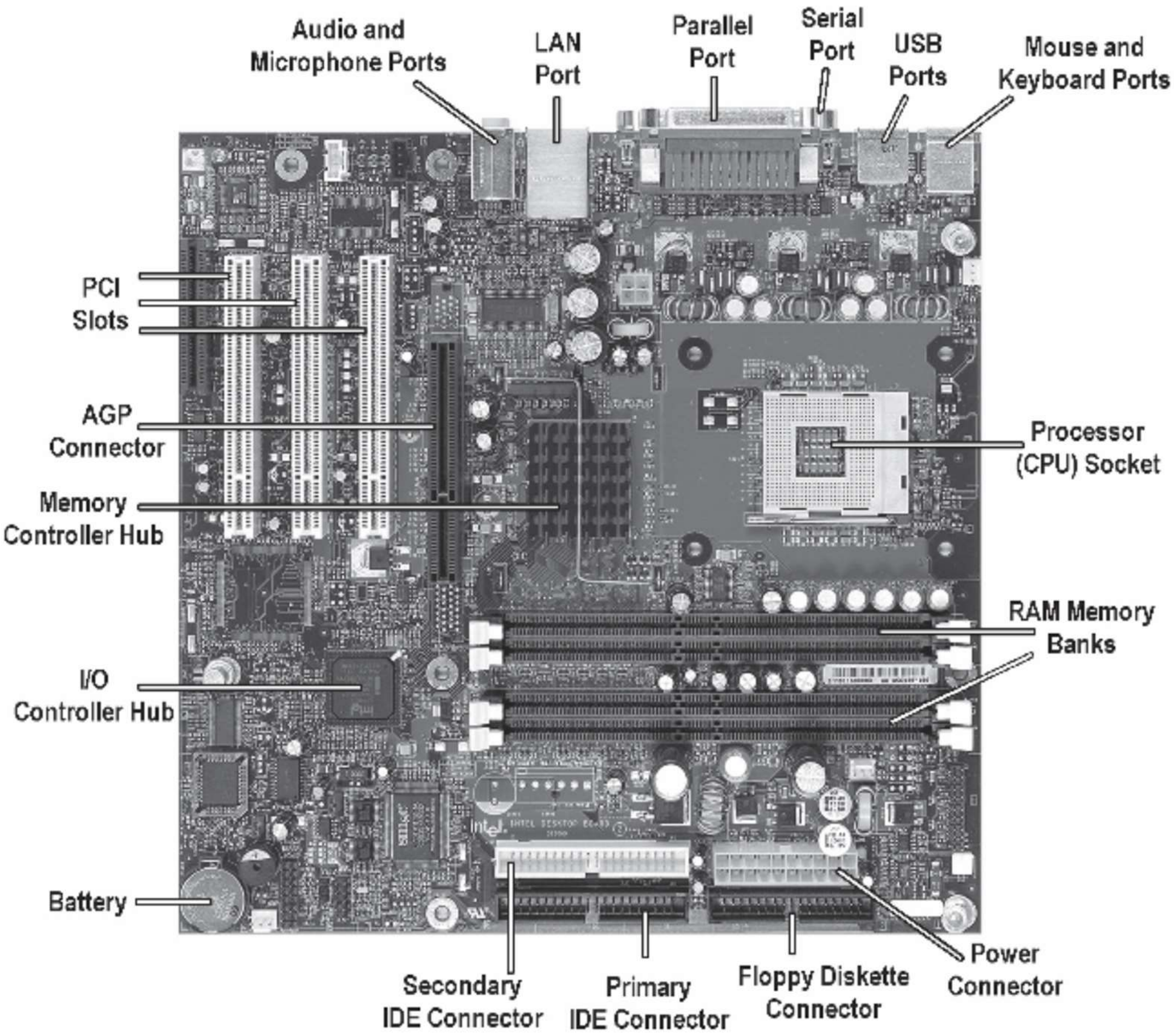
1.3 An Example System

The number of times per second that the image on the monitor is repainted is its *refresh rate*. The *dot pitch* of a monitor tells us how clear the image is.



This monitor has a dot pitch of 0.28mm and a refresh rate of 85Hz.

The graphics card contains memory and programs that support the monitor.


Audio and Microphone Ports
LAN Port
Parallel Port
Serial Port
USB Ports
Mouse and Keyboard Ports
PCI Slots
AGP Connector
Memory Controller Hub
Processor (CPU) Socket
RAM Memory Banks
I/O Controller Hub
Battery
Secondary IDE Connector
Primary IDE Connector
Floppy Diskette Connector
Power Connector

1.3 An Example System

Throughout the remainder of this book you will see how these components work and how they interact with software to make complete computer systems.

What assurance do we have that computer components will operate as we expect?

And what assurance do we have that computer components will operate together?

1.4 Standards Organizations

- There are many organizations that set computer hardware standards-- to include the interoperability of computer components.
- Throughout this book, and in your career, you will encounter many of them.
- Some of the most important standards-setting groups are . . .

1.4 Standards Organizations

- The Institute of Electrical and Electronic Engineers (IEEE)
  - Promotes the interests of the worldwide electrical engineering community.
  - Establishes standards for computer components, data representation, and signaling protocols, among many other things.

1.4 Standards Organizations

- The International Telecommunications Union (ITU)
  – Concerns itself with the interoperability of telecommunications systems, including data communications and telephony.
- National groups establish standards within their respective countries:
  – The American National Standards Institute (ANSI)
  – The British Standards Institution (BSI)

1.4 Standards Organizations

- The International Organization for Standardization (ISO)
  - Establishes worldwide standards for **everything** from screw threads to photographic film.
  - Is influential in formulating standards for computer hardware and software, including their methods of manufacture.
    - ISO/IEC/IEEE 12207 Systems and software engineering – Software life cycle processes

Note: ISO is **not** an acronym. ISO comes from the Greek, *isos,* meaning “equal.”

1.5 Historical Development

- To fully appreciate the computers of today, it is helpful to understand how things got the way they are.
- The evolution of computing machinery has taken place over several centuries.
- In modern times computer evolution is usually classified into four generations according to the salient technology of the era.

We note that many of the following dates are approximate.

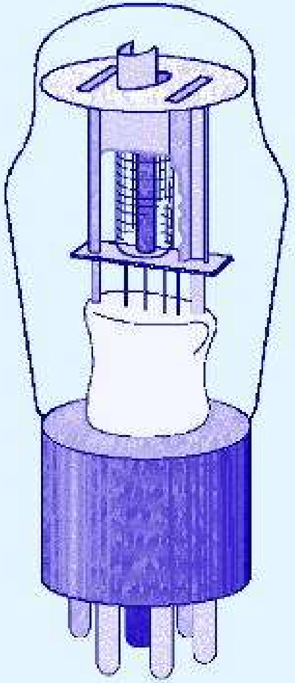
1.5 Historical Development

- Generation Zero: Mechanical Calculating Machines (1642 - 1945)
  - Calculating Clock - Wilhelm Schickard (1592 - 1635).
  - Pascaline - Blaise Pascal (1623 - 1662).
  - Difference Engine - Charles Babbage (1791 - 1871), also designed but never built the Analytical Engine.
  - Punched card tabulating machines - Herman Hollerith (1860 - 1929).

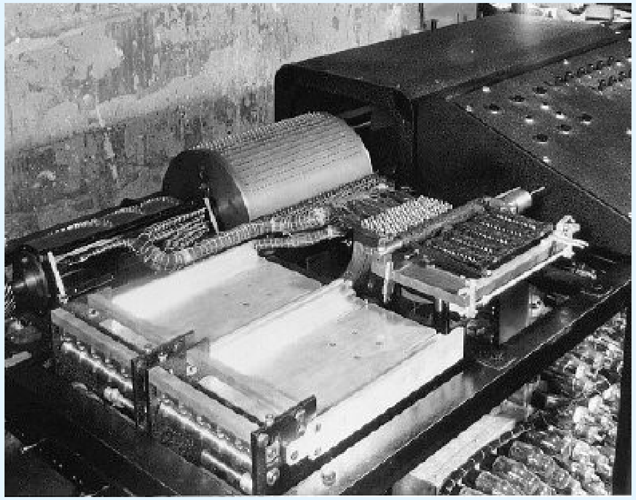
Hollerith cards were commonly used for computer input well into the 1970s.

1.5 Historical Development

- The First Generation: Vacuum Tube Computers (1945 - 1953)



  - Atanasoff Berry Computer (1937 - 1938) solved systems of linear equations.
  - John Atanasoff and Clifford Berry of Iowa State University.



1.5 Historical Development

- The First Generation: Vacuum Tube Computers (1945 - 1953)
  - Electronic Numerical Integrator and Computer (ENIAC)
  - John Mauchly and J. Presper Eckert
  - University of Pennsylvania, 1946

The first *general-purpose* computer.

1.5 Historical Development

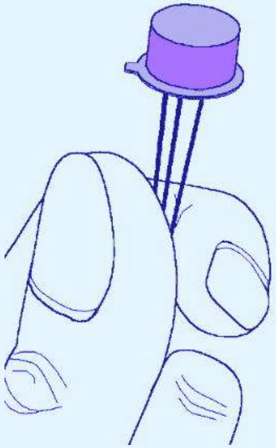
- The First Generation: Vacuum Tube Computers (1945 - 1953)

  - IBM 650 (1955)
  - Phased out in 1969.

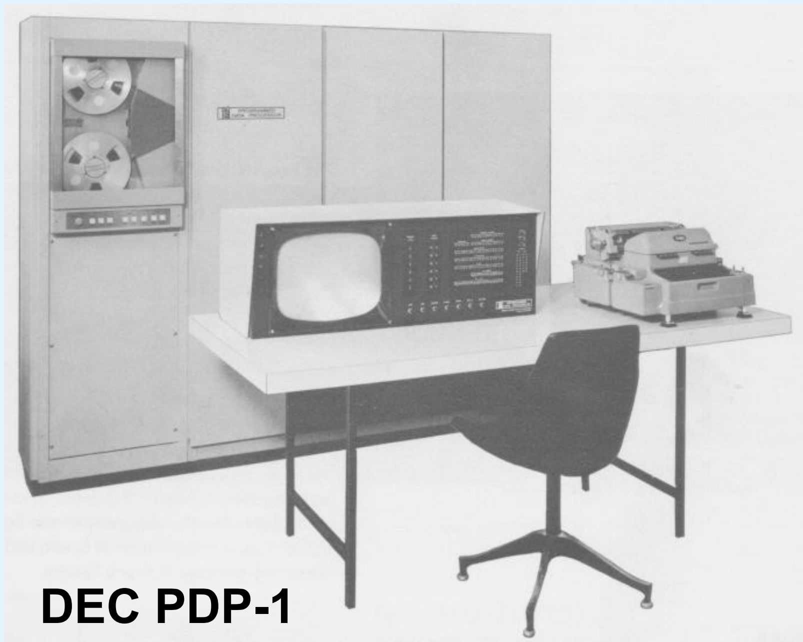
The first *mass-produced* computer.

1.5 Historical Development

- The Second Generation: Transistorized Computers (1954 - 1965)



  - IBM 7094 (scientific) and 1401 (business)
  - Digital Equipment Corporation (DEC) PDP-1
  - Univac 1100
  - . . . and many others.



DEC PDP-1

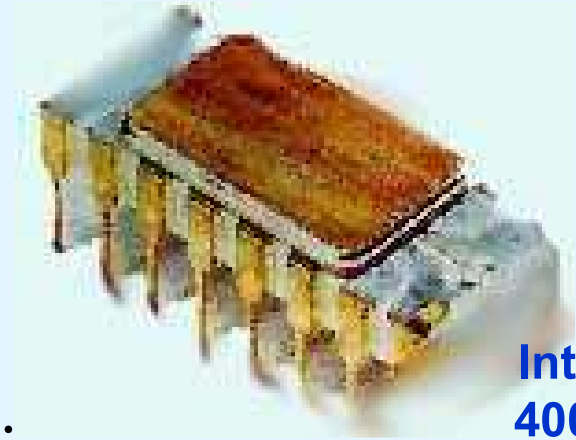
1.5 Historical Development

- The Third Generation: Integrated Circuit Computers (1965 - 1980)
  - IBM 360
  - DEC PDP-8 and PDP-11
  - Cray-1 supercomputer
  - . . . and many others.

1.5 Historical Development

- The Fourth Generation: VLSI Computers (1980 - ????)
  - Very large scale integrated circuits (VLSI) have more than 10,000 components per chip.
  - Enabled the creation of microprocessors.
  - The first was the 4-bit Intel 4004.

    Later versions, such as the 8080, 8086, and 8088 spawned the idea of "personal computing."



Intel 4004

1.5 Historical Development

- Moore's Law (1965)
  - Gordon Moore, Intel founder
  - "The density of transistors in an integrated circuit will double every year."
- Contemporary version:
  - "The density of silicon chips doubles every 18 months."

But this "law" cannot hold forever ...

1.5 Historical Development

- Rock's Law
  - Arthur Rock, Intel financier
  - "The cost of capital equipment to build semiconductors will double every four years."
  - In 1968, a new chip plant cost about $12,000.

At the time, $12,000 would buy a nice home in the suburbs.

An executive earning $12,000 per year was "making a very comfortable living."

1.5 Historical Development

- Rock's Law
  - In 2003, a chip plants under construction will cost over $2.5 billion.

    $2.5 billion is more than the gross domestic product of some small countries, including Belize, Bhutan, and the Republic of Sierra Leone.

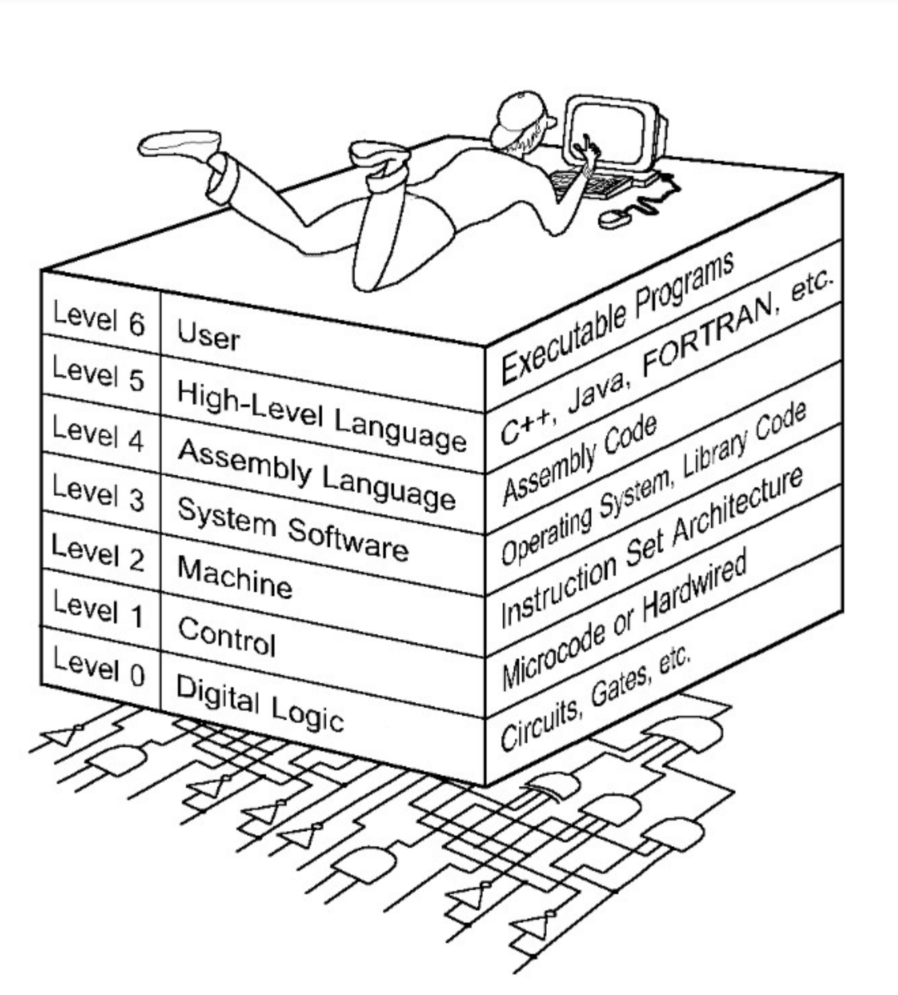
  - For Moore's Law to hold, Rock's Law must fall, or vice versa. But no one can say which will give out first.

1.6 The Computer Level Hierarchy

- Before a computer can do anything worthwhile, it must also use software.
- Writing complex programs requires a “divide and conquer” approach, where each program module solves a smaller problem.
- Complex computer systems employ a similar technique through a series of virtual machine layers

1.6 The Computer Level Hierarchy

- Each virtual machine layer is an abstraction of the level below it.
- The machines at each level execute their own particular instructions, **calling upon** machines at lower levels to perform tasks as required.
- Computer circuits ultimately carry out the work.



1.6 The Computer Level Hierarchy

- Level 6: The User Level
  - Program execution and user interface level.
  - The level with which we are most familiar.
  - The lower levels are invisible from the user level
- Level 5: High-Level Language Level
  - The level with which we interact when we write programs in languages such as C, Pascal, Lisp, and Java.
  - These languages are translated (compiler or interpreter) to a language the machine can understand
  - User in this level sees very little of the lower level
  - Codes can be executed on the different architecture computer

1.6 The Computer Level Hierarchy

- Level 4: Assembly Language Level
  - Acts upon assembly language produced from Level 5, as well as instructions programmed directly at this level.
  - One to one translation to machine language instruction
  - Codes mostly can be executed on the same architecture computer
- Level 3: System Software Level
  - Controls executing processes on the system.
  - Multiprogramming, protecting memory, synchronizing process…
  - Assembly language instructions often pass through Level 3 without modification.

1.6 The Computer Level Hierarchy

- Level 2: Machine Level
  - Also known as the Instruction Set Architecture (ISA) Level.
  - Consists of instructions that are particular to the architecture of the machine.
  - Programs written in machine language need no compilers, interpreters, or assemblers.
  - Codes may be executed on a particular computer

1.6 The Computer Level Hierarchy

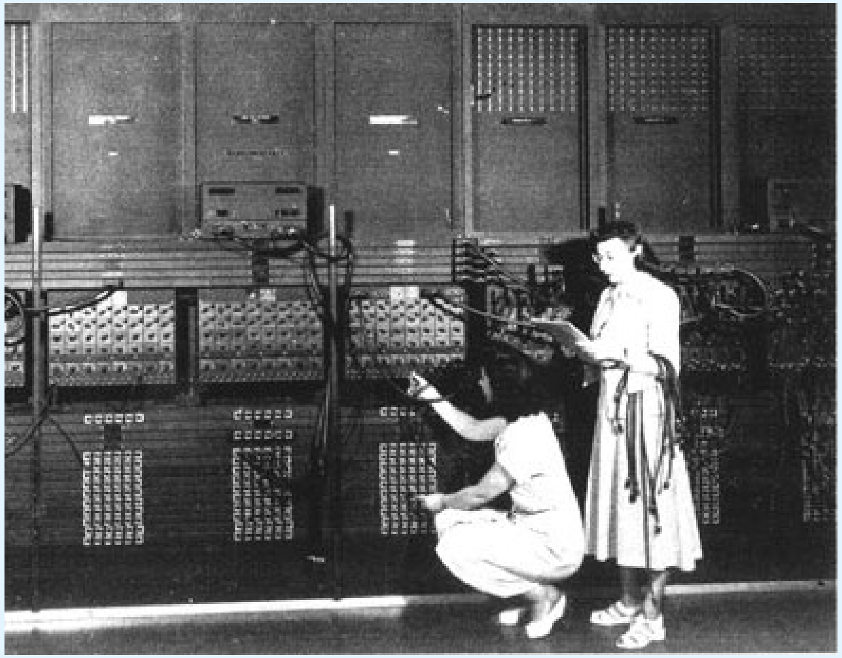
- Level 1: Control Level
  - A *control unit* decodes and executes instructions and moves data through the system.
  - Control units can be *microprogrammed* or *hardwired*.
  - A microprogram is a program written in a low-level language that is implemented by the hardware.
  - Hardwired control units consist of hardware that directly executes machine instructions.

1.6 The Computer Level Hierarchy

- Level 0: Digital Logic Level
  - This level is where we find digital circuits (the chips).
  - Digital circuits consist of gates and wires.
  - These components implement the mathematical logic of all other levels.

1.7 The von Neumann Model

- **On the ENIAC, all programming was done at the digital logic level. No layered architecture existed**
- **Programming the computer involved moving plugs and wires.**



1.7 The von Neumann Model

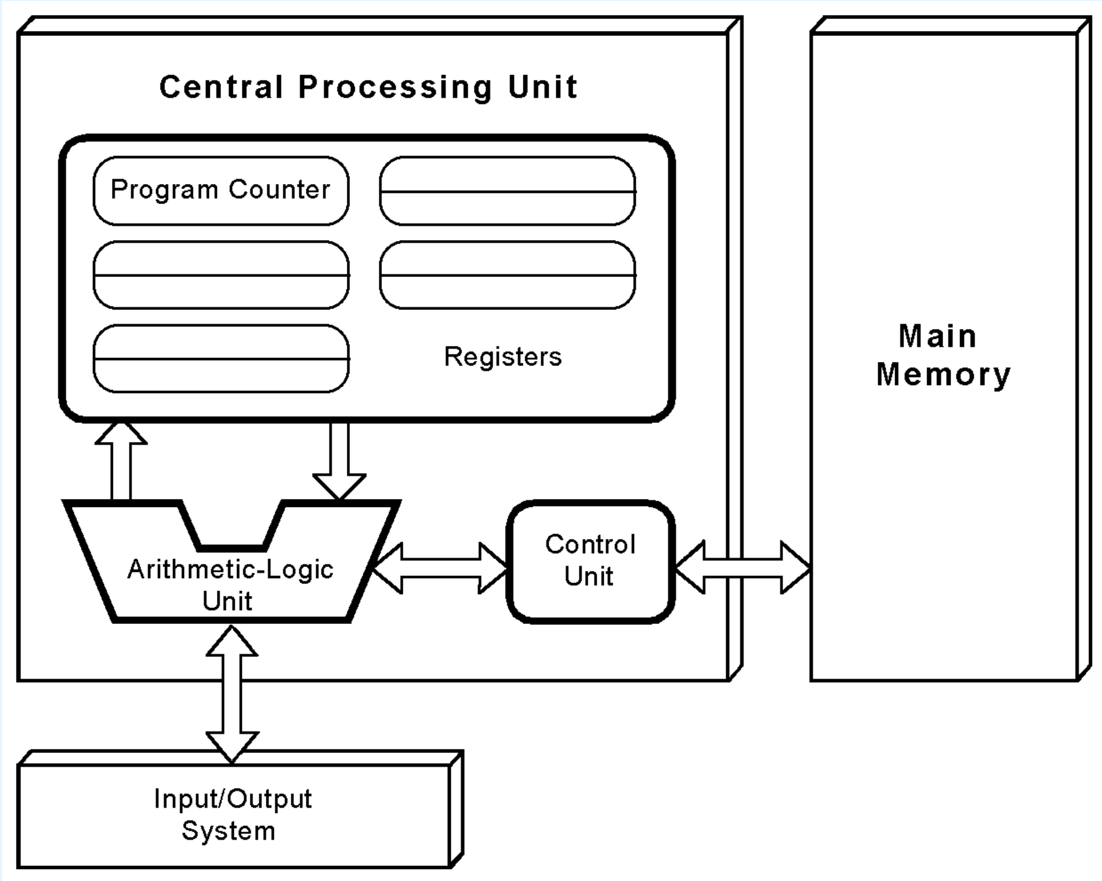
- Inventors of the ENIAC, John Mauchley and J. Presper Eckert, conceived of a computer that could store instructions in memory in their next computer, the EDVAC.
- After reading proposal for EDVAC, John von Neumann published and publicized the idea.(*First Draft of a Report on the EDVAC*)
- Stored-program computers have become known as von Neumann Architecture systems.

1.7 The von Neumann Model

- Today's stored-program computers have the following characteristics:
  - Three hardware systems:
    - A central processing unit (CPU)
    - A main memory system
    - An I/O system
  - The capacity to carry out sequential instruction processing.
  - A single data path between the CPU and main memory.
    - This single path is known as the *von Neumann bottleneck*.

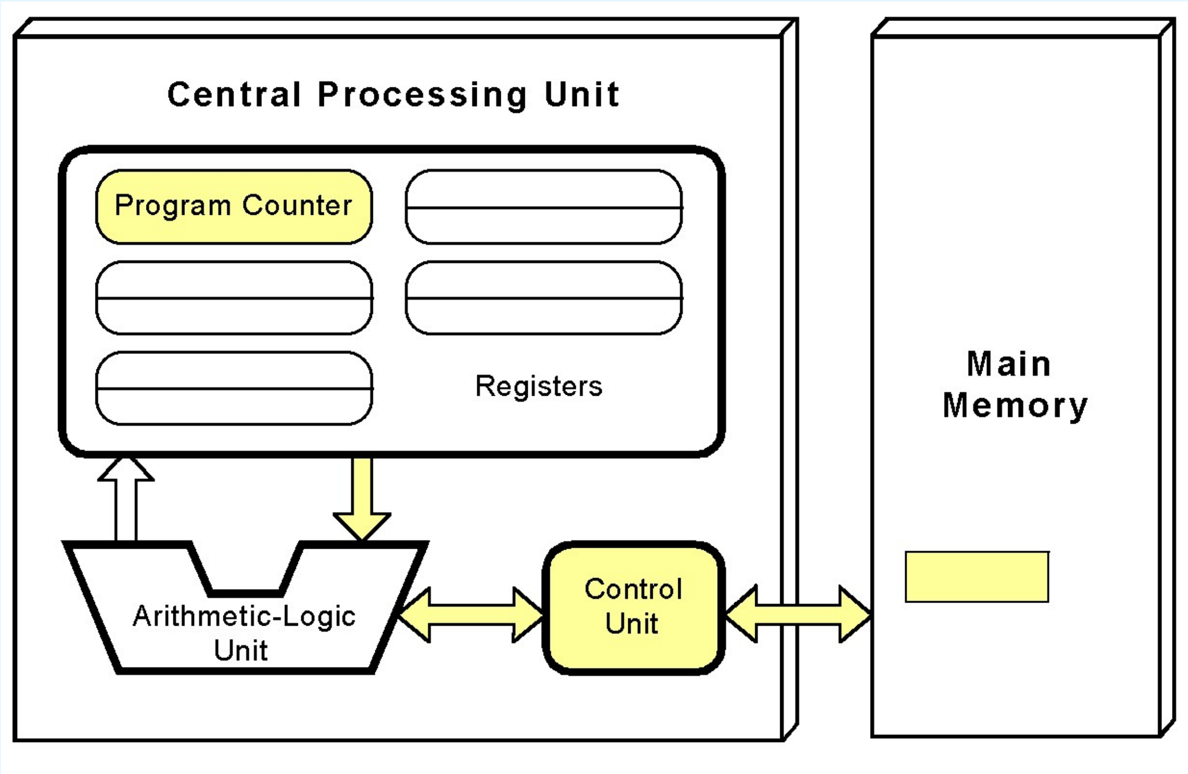
1.7 The von Neumann Model

- This is a general depiction of a von Neumann system:
- These computers employ a fetch-decode-execute cycle to run programs as follows . . .



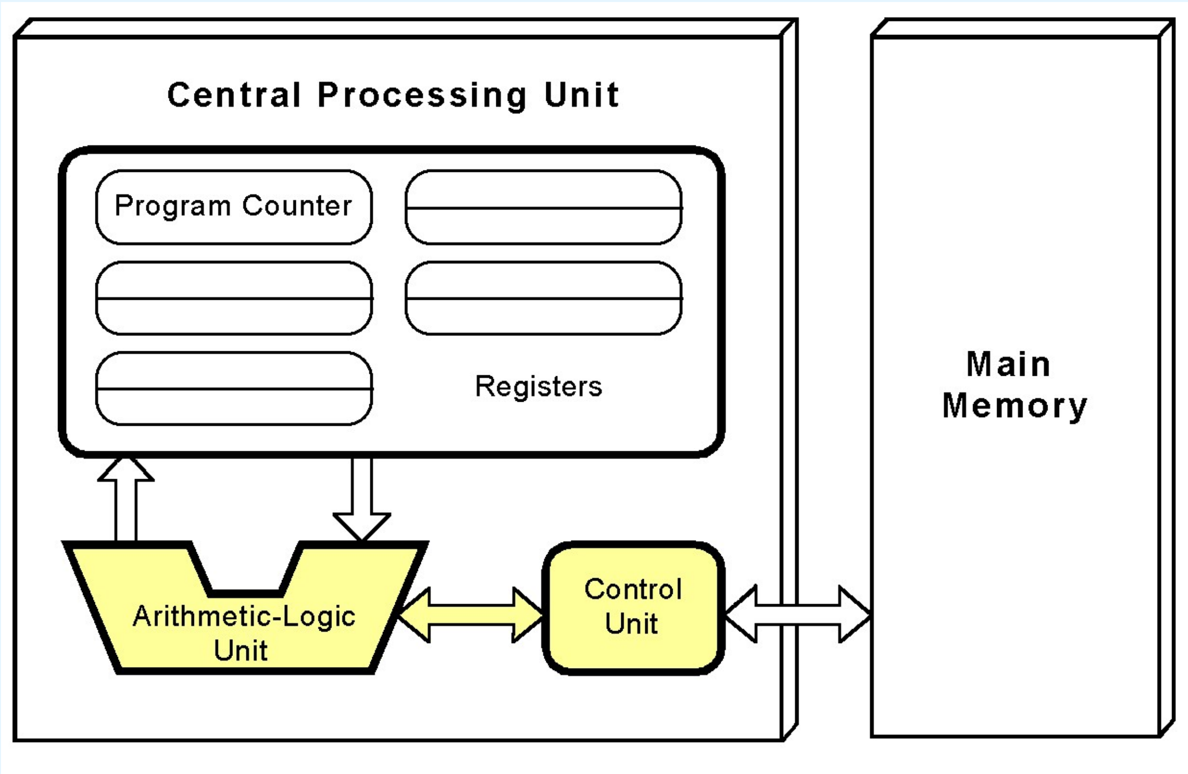
1.7 The von Neumann Model

- The control unit fetches the next instruction from memory using the program counter to determine where the instruction is located.



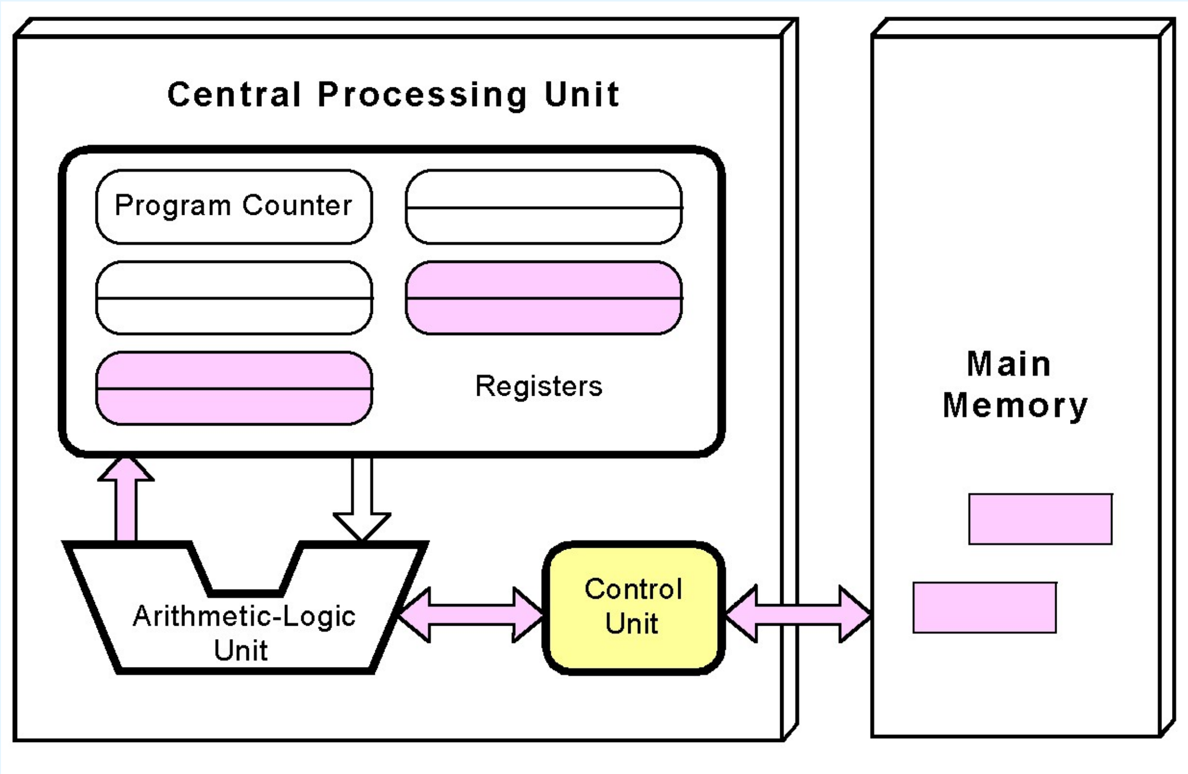
1.7 The von Neumann Model

- **The instruction is decoded into a language that the ALU can understand.**



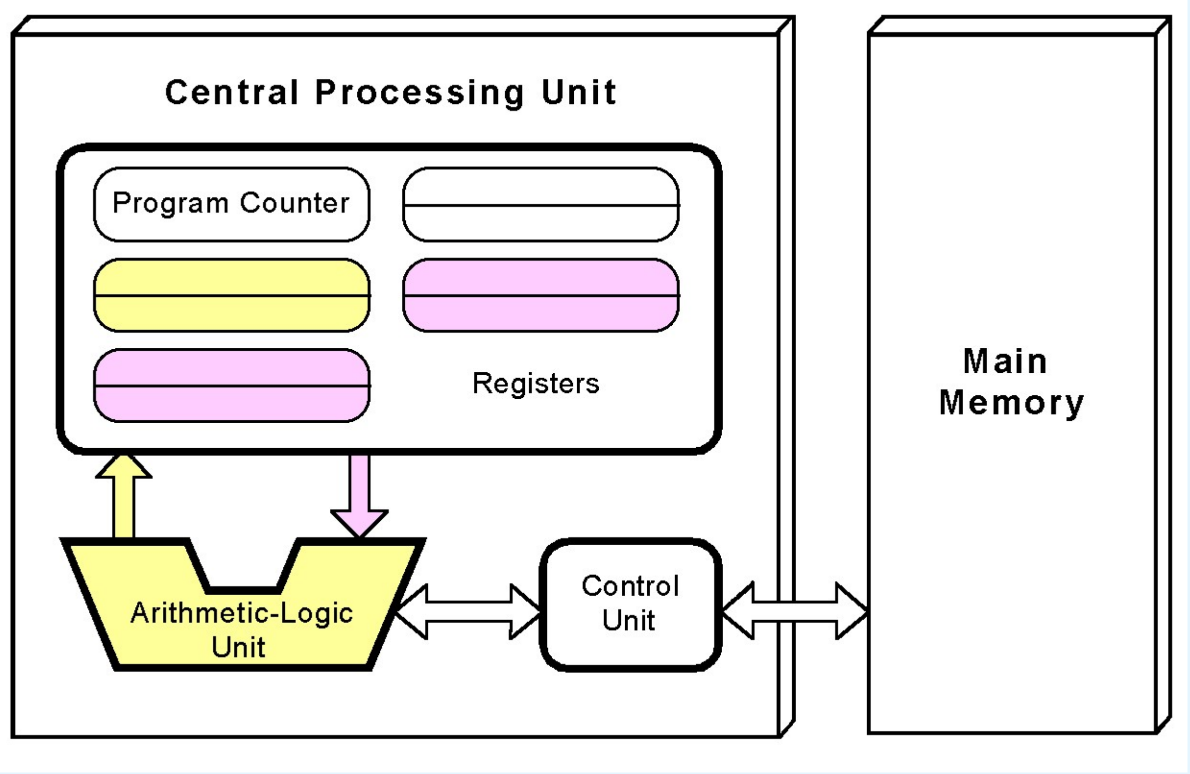
1.7 The von Neumann Model

- **Any data operands required to execute the instruction are fetched from memory and placed into registers within the CPU. (explain how to execute x+y)**



1.7 The von Neumann Model

- **The ALU executes the instruction and places results in registers or memory.**



1.8 Non-von Neumann Models

- Under the von Neumann architecture, a program and its data are both stored in memory. It is therefore possible for a program, to accidentally (or on purpose) modify itself. What implications does this present to you as a programmer??

1.8 Non-von Neumann Models

- Conventional stored-program computers have undergone many incremental improvements over the years.
- These improvements include adding specialized buses, floating-point units, and cache memories, to name only a few.
- But enormous improvements in computational power require departure from the classic von Neumann architecture.
- Adding processors is one approach.

1.8 Non-von Neumann Models

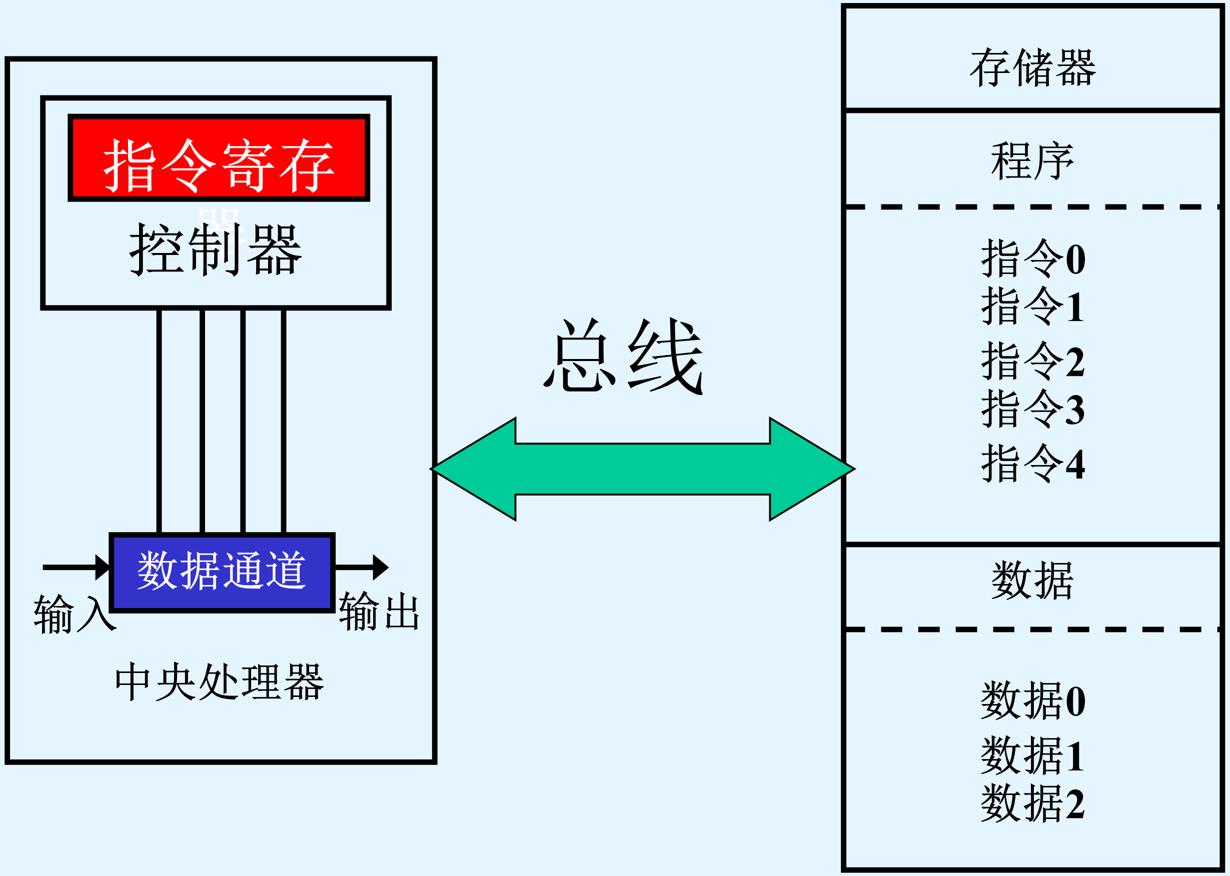
- In the late 1960s, high-performance computer systems were equipped with dual processors to increase computational throughput.
- In the 1970s supercomputer systems were introduced with 32 processors.
- Supercomputers with 1,000 processors were built in the 1980s.
- In 1999, IBM announced its Blue Gene system containing over 1 million processors.

1.8 Non-von Neumann Models

- Parallel processing is only one method of providing increased computational power.
- More radical systems have reinvented the fundamental concepts of computation.
- These advanced systems include genetic computers, quantum computers, and dataflow systems.
- At this point, it is unclear whether any of these systems will provide the basis for the next generation of computers.

几个重要概念

- 冯·诺依曼体系结构模型



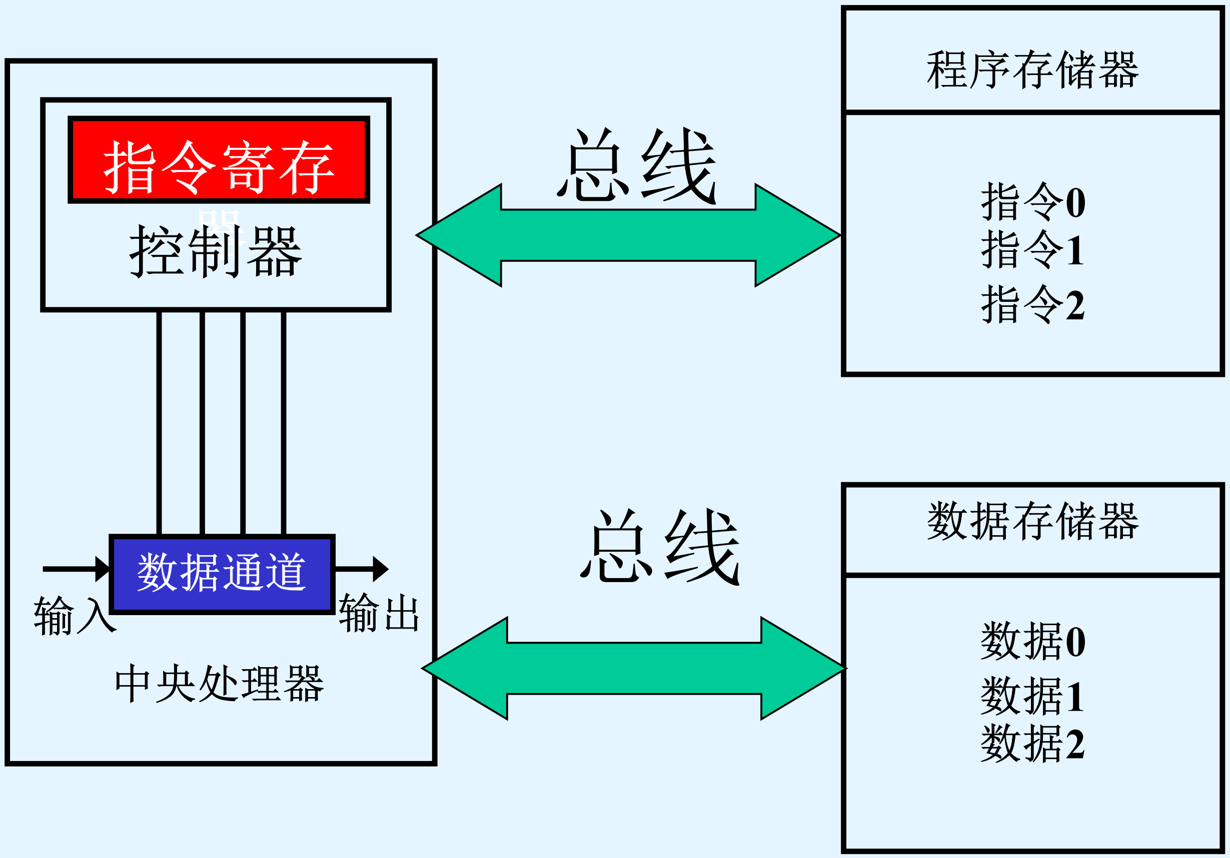
几个重要概念

- 冯·诺依曼体系的特点
  - 数据与指令都存储在同一存储区中，取指令与取数据利用同一数据总线。
  - 早期的微处理器大多采用冯·诺依曼结构，典型代表是Intel公司的X86微处理器。取指和取操作数都在同一总线上，通过分时复用的方式进行的
  - ARM7——冯诺依曼体系

结构简单，但速度较慢。取指不能同时取数据

几个重要概念

- 哈佛体系结构模型



几个重要概念

- 哈佛体系结构的特点
  - 程序存储器与数据存储器分开.
  - 提供了较大的存储器带宽，各自有自己的总线。
  - 适合于数字信号处理.
  - 大多数DSP都是哈佛结构.
  - ARM9是哈佛结构
  - 取指和取数在同一周期进行，提高速度

几个重要概念

- 改进的哈佛结构，其结构特点为：
  - 使用两个独立的存储器模块，分别存储指令和数据，每个存储模块都不允许指令和数据并存；
  - 具有一条独立的地址总线和一条独立的数据总线，利用公用地址总线访问两个存储模块（程序存储模块和数据存储模块），公用数据总线则被用来完成程序存储模块或数据存储模块与CPU之间的数据传输；
  - 两条总线由程序存储器和数据存储器分时共用
  - 如51单片机，虽然数据指令存储区是分开的，但总线是分时复用的，所以属于改进型的哈佛结构
- 现在的处理器虽然外部总线上看是诺依曼结构的，但是由于内部CACHE的存在，因此实际上内部来看已经类似改进型哈佛结构的了

Conclusion

- This chapter has given you an overview of the subject of computer architecture.
- You should now be sufficiently familiar with general system structure to guide your studies throughout the remainder of this course.
- Subsequent chapters will explore many of these topics in great detail.

Home Work

- Homework
  - 查阅资料了解计算机的发展历史，各代计算机的代表机型、优点和缺陷……。
- Review:
  - P33: 1, 3, 5,7,23
  - P34: 4,5,9