THE ESSENTIALS OF

Computer Organization *and* Architecture

THIRD EDITION

Linda Null
Julia Lobur

Chapter 1
Introduction

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

Why study computer organization and architecture?

- Design better programs, including system software such as compilers, operating systems, and device drivers.
- Optimize program behavior.
- Evaluate (benchmark) computer system performance.
- Understand time, space, and price tradeoffs.

1.1 Overview

- Computer organization
  - Encompasses all physical aspects of computer systems.
  - E.g., circuit design, control signals, memory types.

- Computer architecture
  - Logical aspects of system implementation as seen by the programmer.
  - E.g., instruction sets, instruction formats, data types, addressing modes.

1.2 Computer Components

- There is no clear distinction between matters related to computer organization and matters relevant to computer architecture.
- Principle of Equivalence of Hardware and Software:
  - ***Any task done by software can also be done using hardware, and any operation performed directly by hardware can be done using software.****

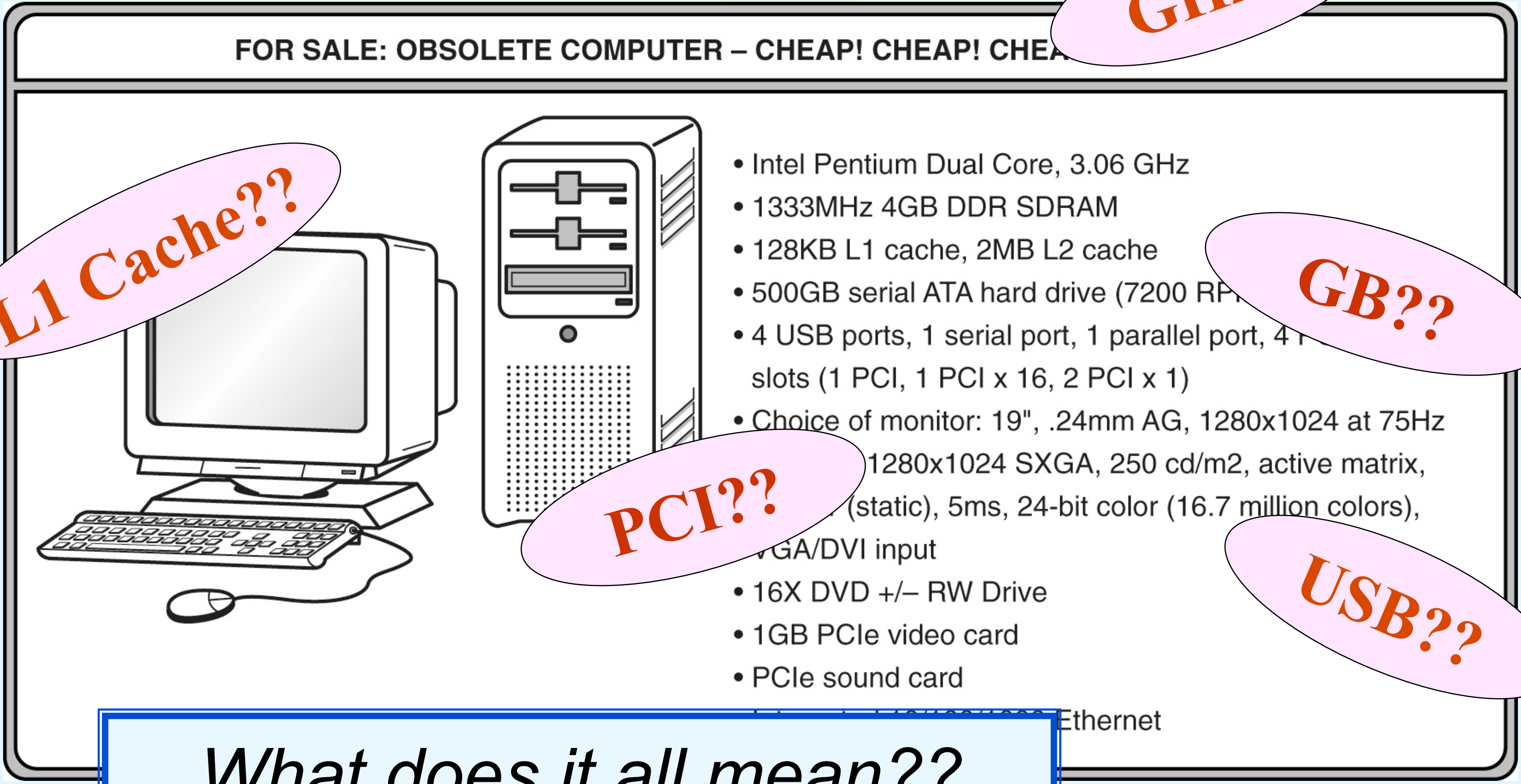
* Assuming speed is not a concern.

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 (I/O system).

1.3 An Example System

Consider this advertisement:



FOR SALE: OBSOLETE COMPUTER – CHEAP! CHEAP! CHEA

- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RP
- 4 USB ports, 1 serial port, 1 parallel port, 4
  slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz
  1280x1024 SXGA, 250 cd/m2, active matrix,
  (static), 5ms, 24-bit color (16.7 million colors),
  VGA/DVI input
- 16X DVD +/– RW Drive
- 1GB PCIe video card
- PCIe sound card

Ethernet

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}
- Exa- (E) = 1 quintillion = 10^{18} and 2^{60}
- Zetta- (Z) = 1 sextillion = 10^{21} and 2^{70}
- Yotta- (Y) = 1 septillion = 10^{24} and 2^{80}

Whether a metric refers to a power of ten or a power of two typically 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 (8 bits)
  - 1KB = 2^{10} = 1024 Bytes
  - 1MB = 2^{20} = 1,048,576 Bytes
  - 1GB = 2^{30} = 1,073,741,824 Bytes
  - Main memory (RAM) is measured in MB or GB
  - 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}
- Atto- (a) = 1 quintillionth = 10^{-18}
- Zepto- (z) = 1 sextillionth = 10^{-21}
- Yocto- (y) = 1 septillionth = 10^{-24}

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

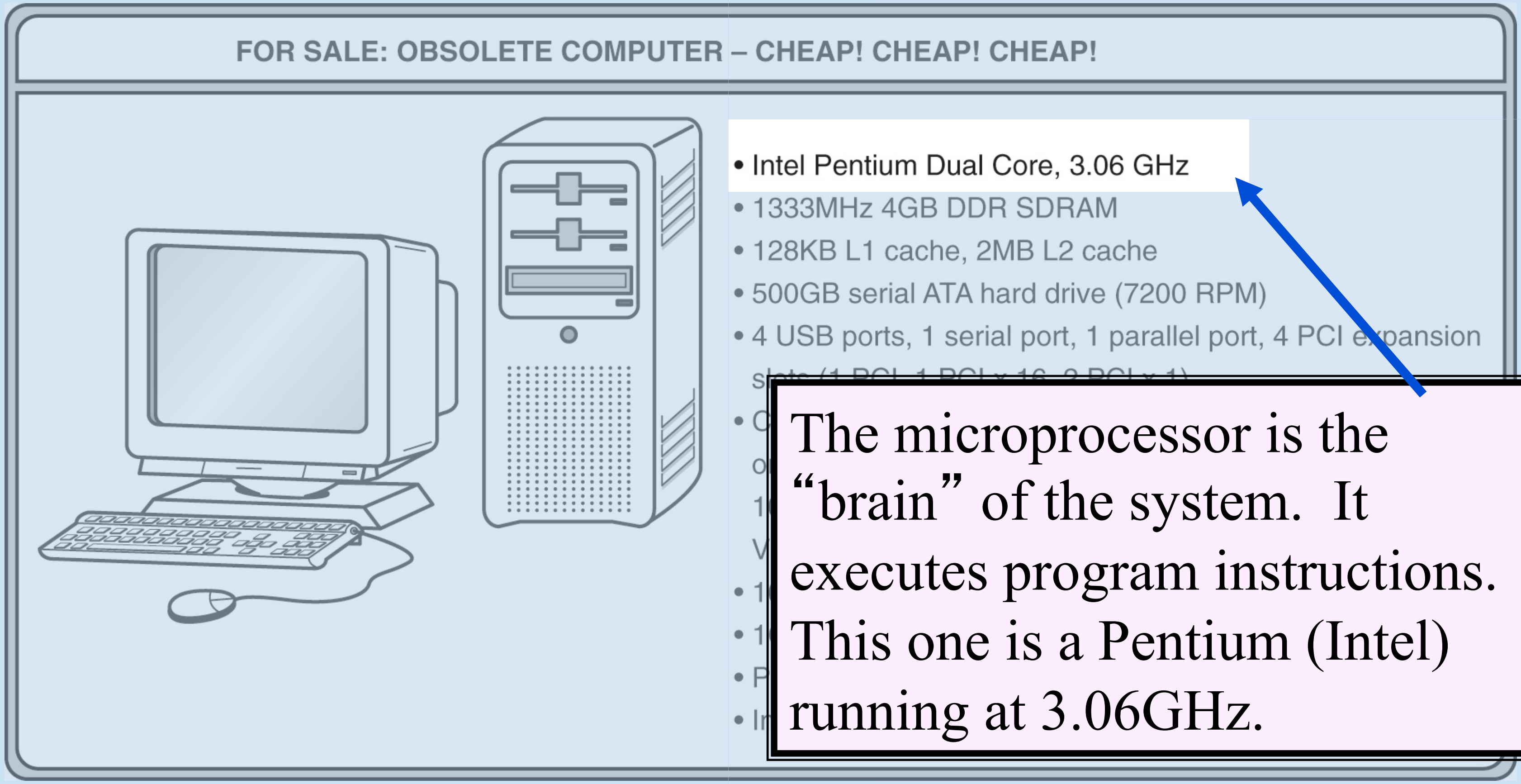
- We note that cycle time is the reciprocal of clock frequency.
- A bus operating at 133MHz has a cycle time of 7.52 nanoseconds:

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

Now back to the advertisement ...

A bus is a subsystem that transfers data between components inside a computer

1.3 An Example System



FOR SALE: OBSOLETE COMPUTER – CHEAP! CHEAP! CHEAP!

- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- C
  o
  1
  V
- 1
- 1
- P
- In

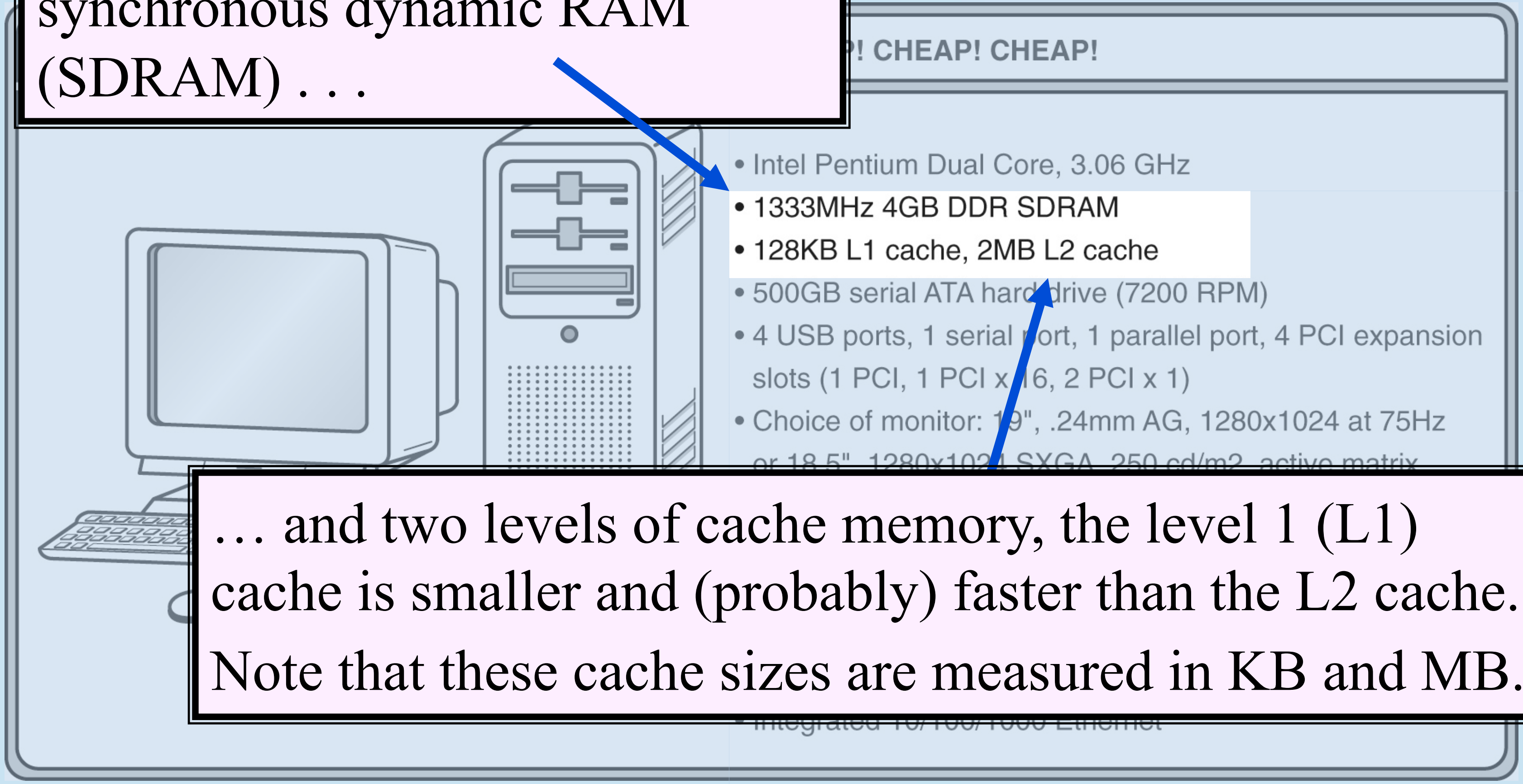
The microprocessor is the "brain" of the system. It executes program instructions. This one is a Pentium (Intel) running at 3.06GHz.

1.3 An Example System

- Computers with large main memory capacity can run larger programs with greater speed than computers having small memories.
- RAM is an acronym for *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 4GB of (fast) synchronous dynamic RAM (SDRAM) . . .



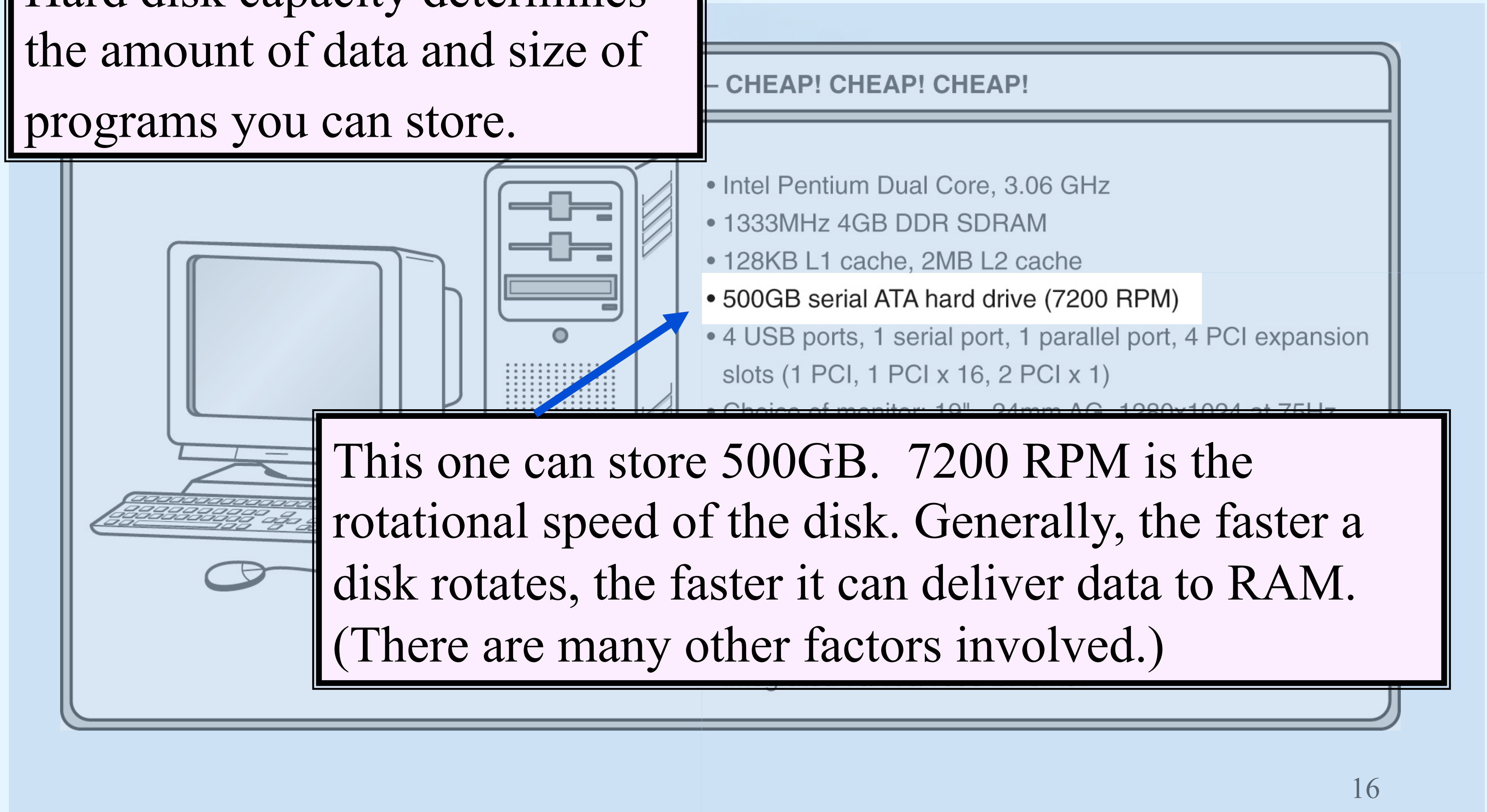
!! CHEAP! CHEAP!

- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz or 18.5", 1280x1024 SXGA, 250 cd/m2, active matrix
- Integrated 10/100/1000 Ethernet

… 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 and MB.

1.3 An Example System

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



CHEAP! CHEAP! CHEAP!

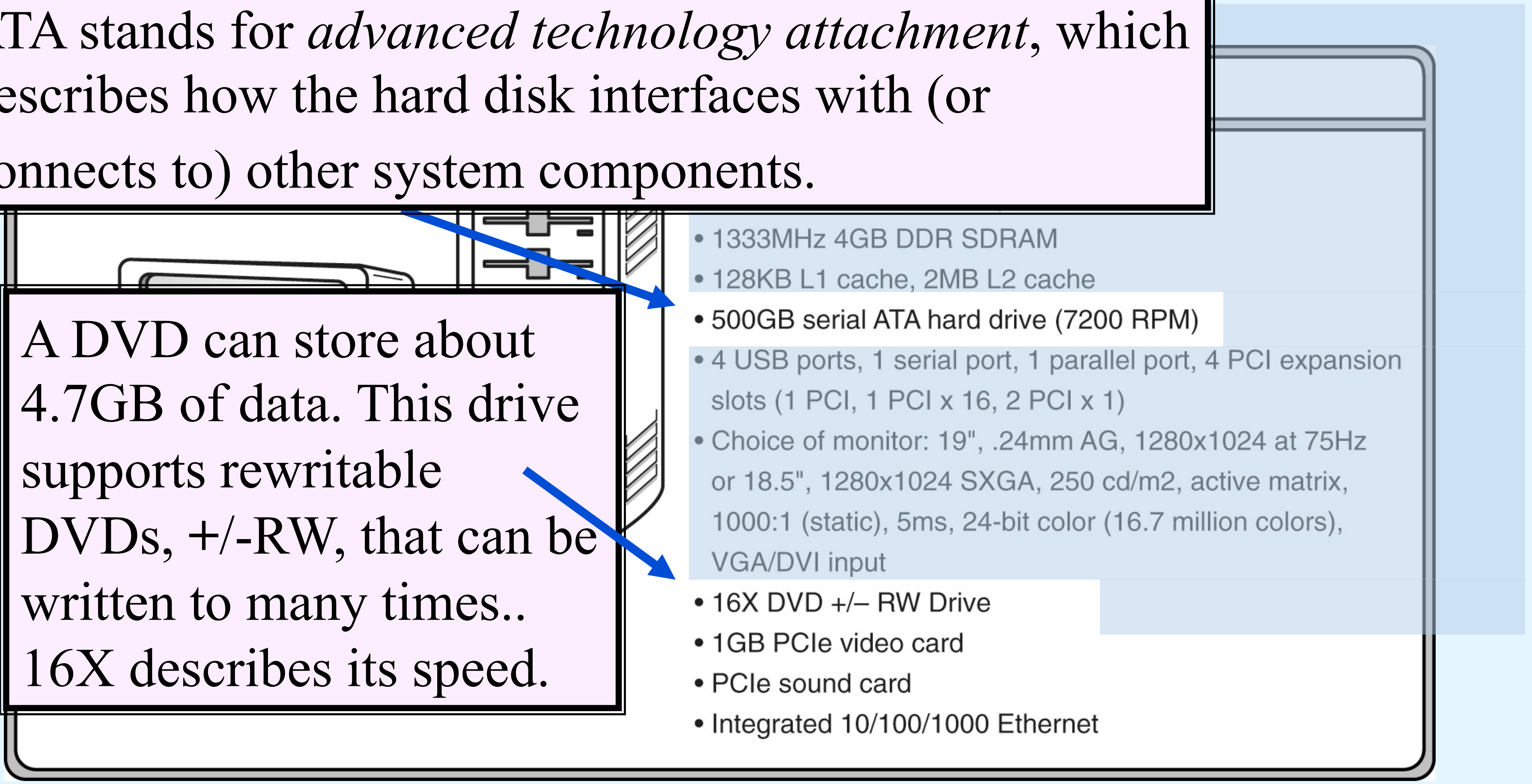
- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)

This one can store 500GB. 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

ATA stands for *advanced technology attachment*, which describes how the hard disk interfaces with (or connects to) other system components.

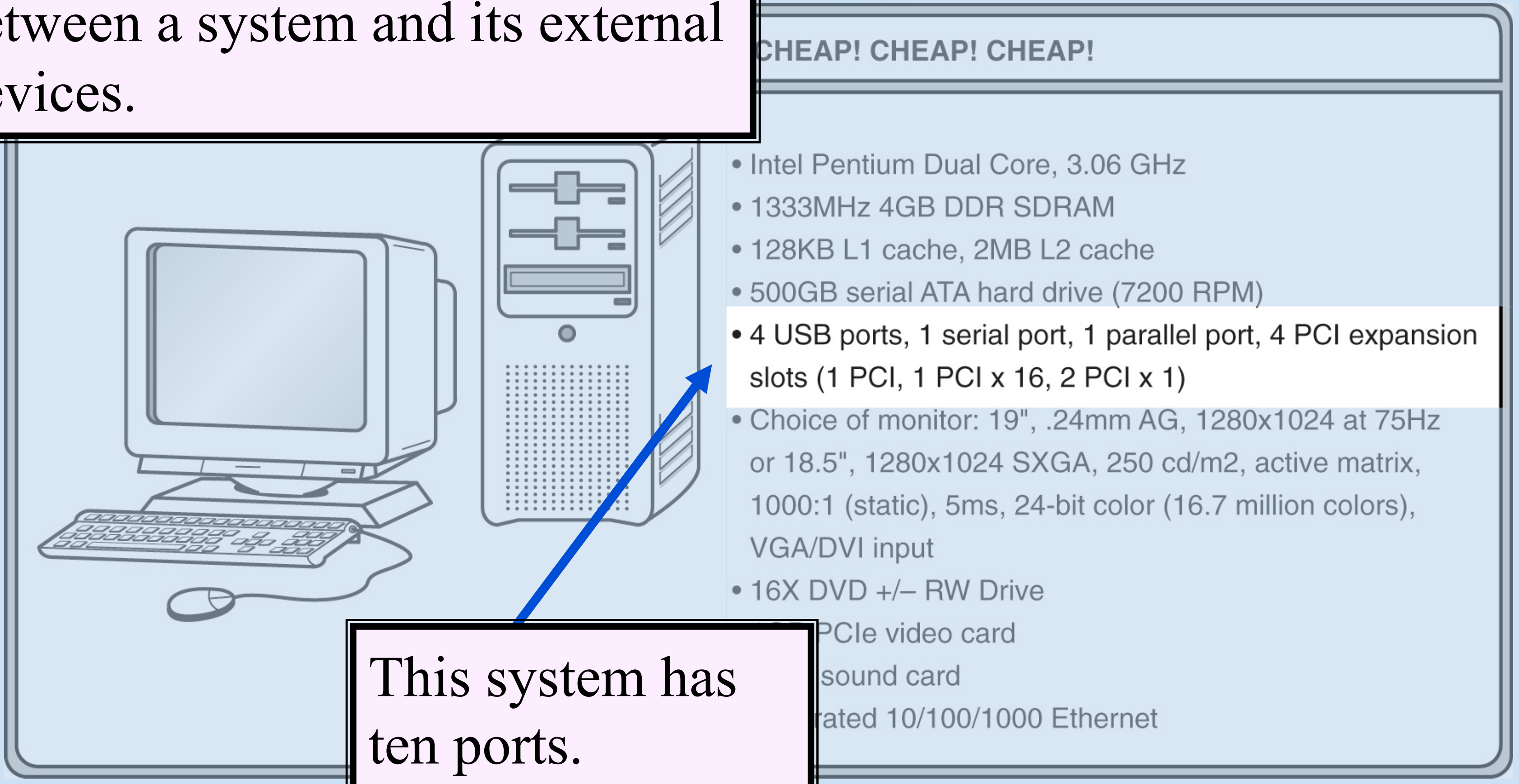
A DVD can store about 4.7GB of data. This drive supports rewritable DVDs, +/-RW, that can be written to many times.. 16X describes its speed.



- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz or 18.5", 1280x1024 SXGA, 250 cd/m2, active matrix, 1000:1 (static), 5ms, 24-bit color (16.7 million colors), VGA/DVI input
- 16X DVD +/– RW Drive
- 1GB PCIe video card
- PCIe sound card
- Integrated 10/100/1000 Ethernet

1.3 An Example System

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



CHEAP! CHEAP! CHEAP!

- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz or 18.5", 1280x1024 SXGA, 250 cd/m2, active matrix, 1000:1 (static), 5ms, 24-bit color (16.7 million colors), VGA/DVI input
- 16X DVD +/– RW Drive
- PCIe video card
- sound card
- rated 10/100/1000 Ethernet

This system has ten ports.

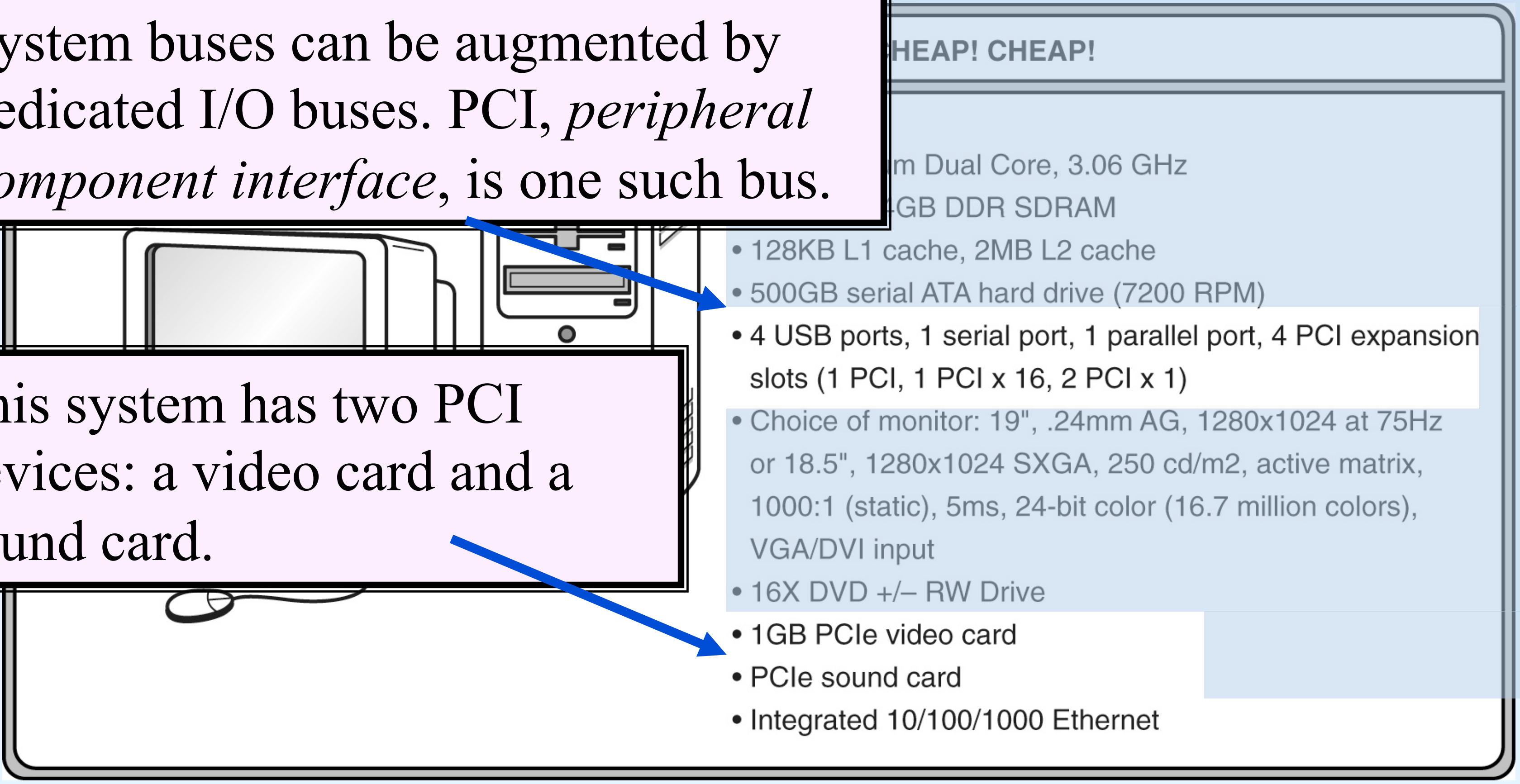
1.3 An Example System

- 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 video card and a sound card.



HEAP! CHEAP!

m Dual Core, 3.06 GHz

GB DDR SDRAM

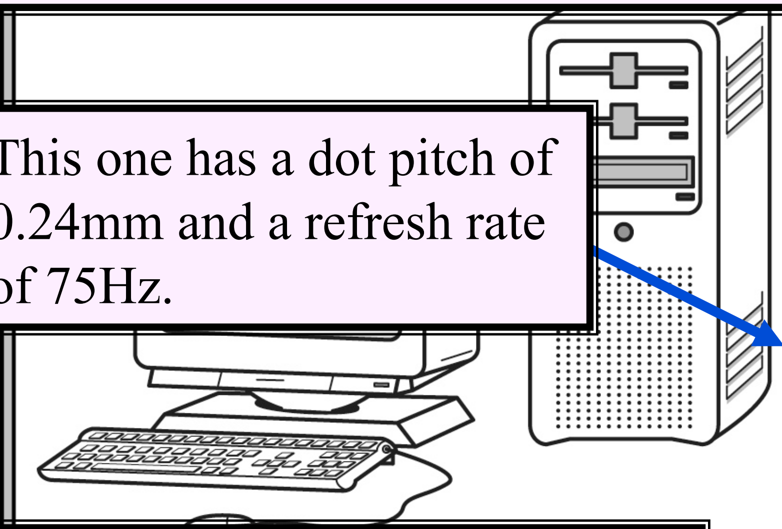
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz or 18.5", 1280x1024 SXGA, 250 cd/m2, active matrix, 1000:1 (static), 5ms, 24-bit color (16.7 million colors), VGA/DVI input
- 16X DVD +/– RW Drive
- 1GB PCIe video card
- PCIe sound card
- Integrated 10/100/1000 Ethernet

1.3 An Example System

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

This one has a dot pitch of 0.24mm and a refresh rate of 75Hz.

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



- Intel Pentium Dual Core, 3.06 GHz
- 1333MHz 4GB DDR SDRAM
- 128KB L1 cache, 2MB L2 cache
- 500GB serial ATA hard drive (7200 RPM)
- 4 USB ports, 1 serial port, 1 parallel port, 4 PCI expansion slots (1 PCI, 1 PCI x 16, 2 PCI x 1)
- Choice of monitor: 19", .24mm AG, 1280x1024 at 75Hz or 18.5", 1280x1024 SXGA, 250 cd/m2, active matrix, 1000:1 (static), 5ms, 24-bit color (16.7 million colors), VGA/DVI input
- 16X DVD +/– RW Drive
- 1GB PCIe video card
- PCIe sound card
- Integrated 10/100/1000 Ethernet

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.

This statement raises two important questions:

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

Example: The Pentium FDIV bug, 1994.

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.

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.

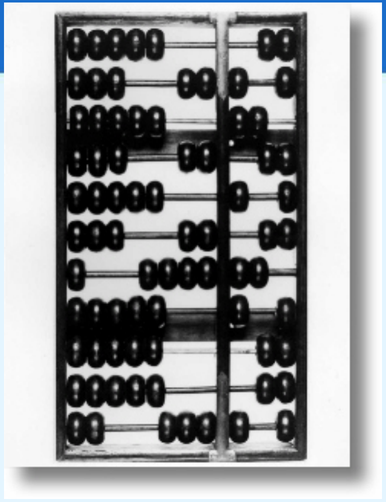
1.5 Historical Development

- 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.

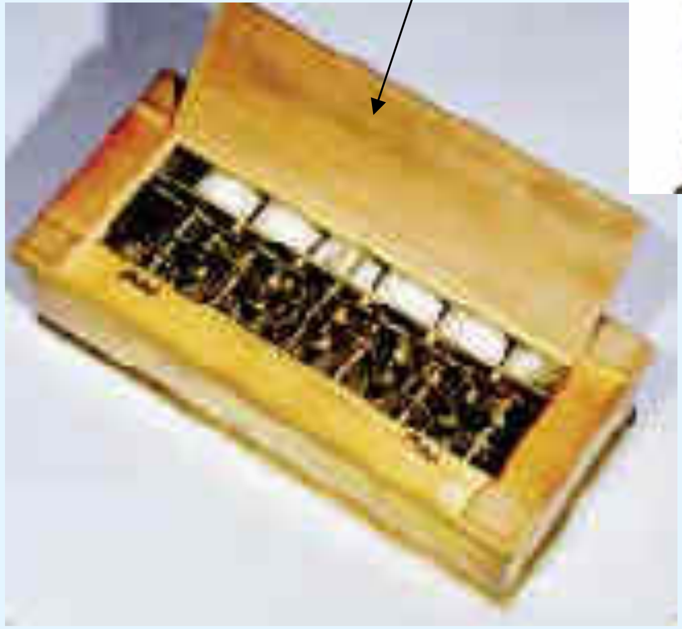
1.5 Historical Development

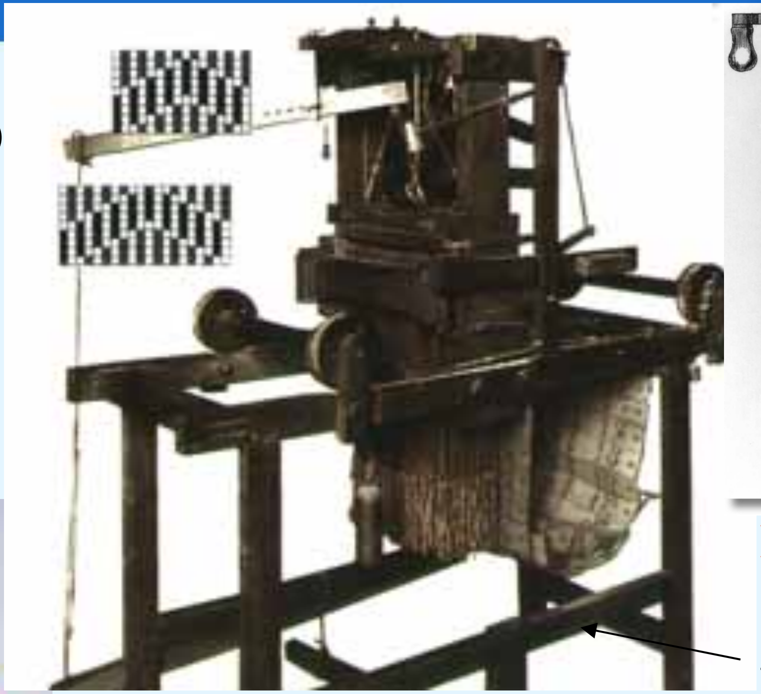
Early mechanical computational devices

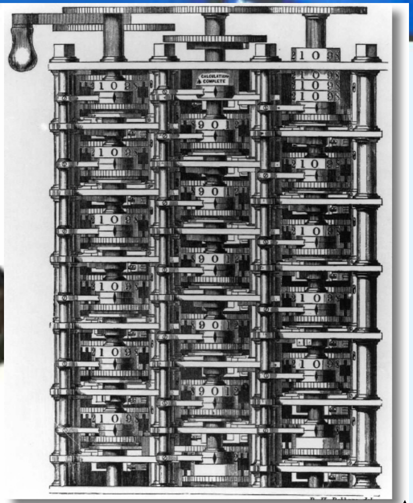


Abacus (3000 BC)

Pascal´s Calculator (1600s)







Early programmable devices:

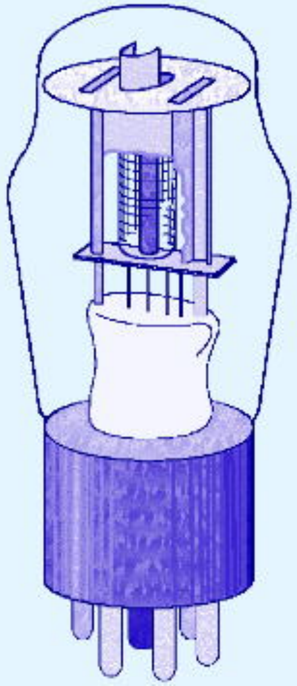
Jacquard´s Loom (1800)

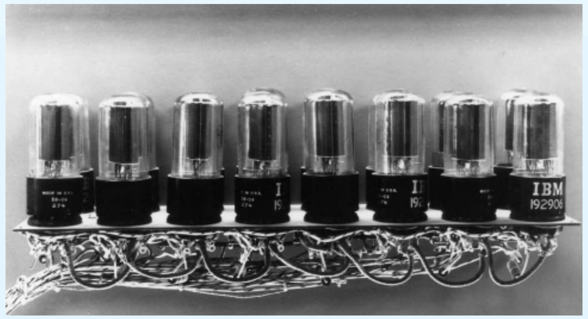
Babbage´s Analytical Engine (1832)

Tabulating machine for 1890 census Hollerith cards

1.5 Historical Development

- The First Generation: Vacuum Tube Computers (1945 - 1953)
  - Atanasoff Berry Computer (1937 - 1938) solved systems of linear equations. Not a general-purpose computer.
  - John Atanasoff and Clifford Berry of Iowa State University.





A vacuum-tube circuit storing 1 byte

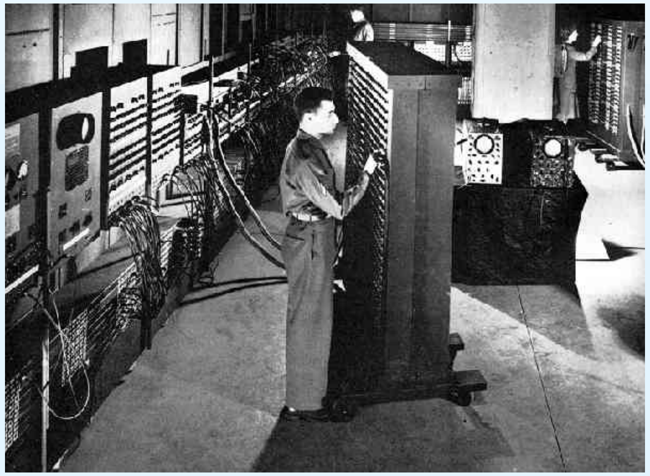
1.5 Historical Development

- 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 ENIAC was the first *general-purpose* computer.

1.5 Historical Development

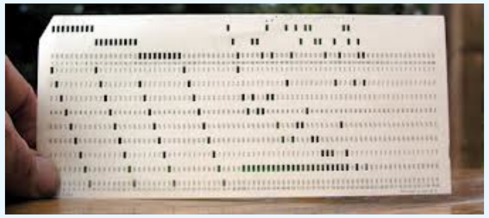
- ENIAC (1943 - 1946)



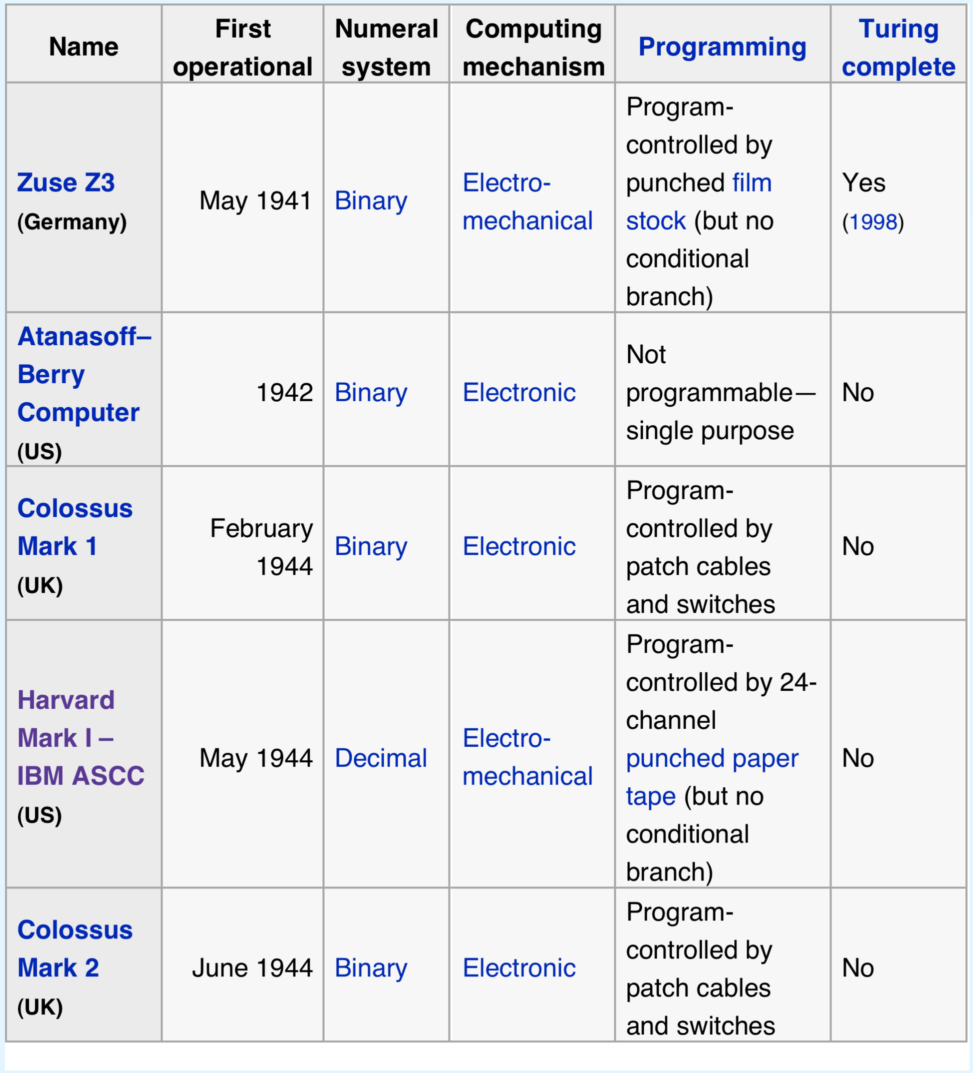
Built to calculate trajectories for ballistic shells during WWII, programmed by setting switches and plugging and unplugging cables. It used 18,000 tubes and weighted 30 tons. The size of its numerical word was 10 decimal digits, and it could perform 5000 additions and 357 multiplications per second.

1.5 Historical Development

- The First Generation: Vacuum Tube Computers (1945 – 1953)
  - Machine code, assembly language
  - Central processor that was unique to that machine
  - Few machines could be considered "general-purpose"
  - Use of drum memory and magnetic core memory
  - Program and data are loaded using punched cards or paper tape
  - 2 Kb memory, 10 KIPS



1.5 Historical Development



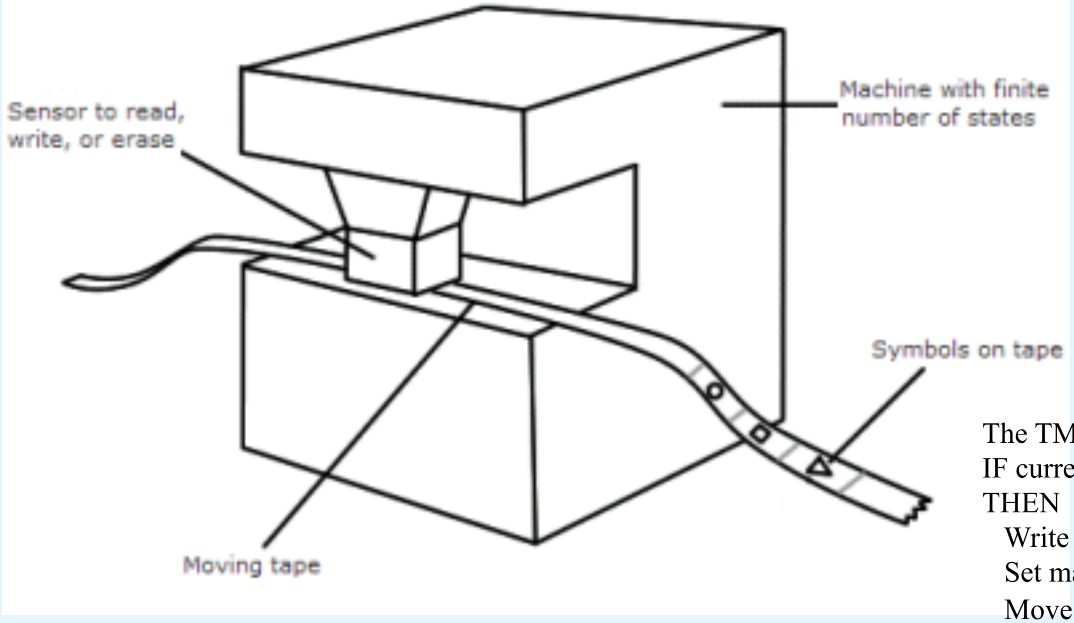
Name	First operational	Numeral system	Computing mechanism	Programming	Turing complete
Zuse Z3 (Germany)	May 1941	Binary	Electro-mechanical	Program-controlled by punched film stock (but no conditional branch)	Yes (1998)
Atanasoff–Berry Computer (US)	1942	Binary	Electronic	Not programmable—single purpose	No
Colossus Mark 1 (UK)	February 1944	Binary	Electronic	Program-controlled by patch cables and switches	No
Harvard Mark I – IBM ASCC (US)	May 1944	Decimal	Electro-mechanical	Program-controlled by 24-channel punched paper tape (but no conditional branch)	No
Colossus Mark 2 (UK)	June 1944	Binary	Electronic	Program-controlled by patch cables and switches	No

1.5 Historical Development

Name	First operational	Numeral system	Computing mechanism	Programming	Turing complete
ENIAC **(US)**	July 1946	Decimal	Electronic	Program-controlled by patch cables and switches	Yes
Manchester Small-Scale Experimental Machine **(UK)**	June 1948	Binary	Electronic	Stored-program in Williams cathode ray tube memory	Yes
Modified ENIAC **(US)**	September 1948	Decimal	Electronic	Program-controlled by patch cables and switches plus a primitive read-only stored programming mechanism using the Function Tables as program ROM	Yes
EDSAC **(UK)**	May 1949	Binary	Electronic	Stored-program in mercury delay line memory	Yes
Manchester Mark 1 **(UK)**	October 1949	Binary	Electronic	Stored-program in Williams cathode ray tube memory	Yes

1.5 Historical Development

A Turing machine is a theoretical generalized computer, composed of a tape on which symbols representing instructions are imprinted. The tape can move backwards and forwards in the machine, which can read the intructions and write the result-ant output back onto the tape.


Sensor to read, write, or erase
Machine with finite number of states
Symbols on tape
Moving tape

Alan Turing, 1936

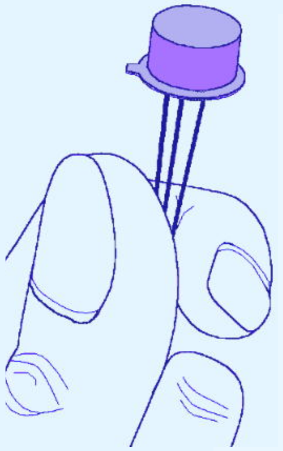
	i
j	**k,s,d**

State transition table

```
The TM interprets instruction (i, j, k, s, d) as:
IF current state is i AND current symbol is j
THEN
   Write symbol k on tape (replacing j)
   Set machine state to s
   Move read/write head one cell in direction d
ENDIF
```

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
  - Control Data Corporation 1604.
  - . . . and many others.

These systems had few architectural similarities.

1.5 Historical Development

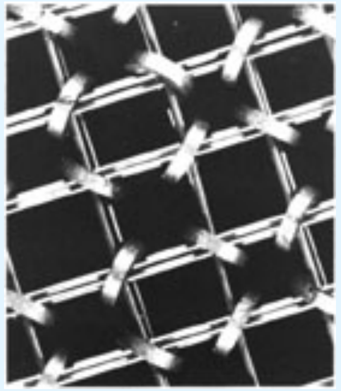
- IBM 7094

1.5 Historical Development

- The Second Generation: Transistorized Computers (1954 - 1965)
  - Transistors – small, low-power, low-cost, more reliable than vacuum tubes
  - Magnetic core memory
  - Two's complement, floating point arithmetic
  - Reduced the computational time from milliseconds to microseconds
  - High level languages
  - First operating systems: handled one program at a time

1.5 Historical Development

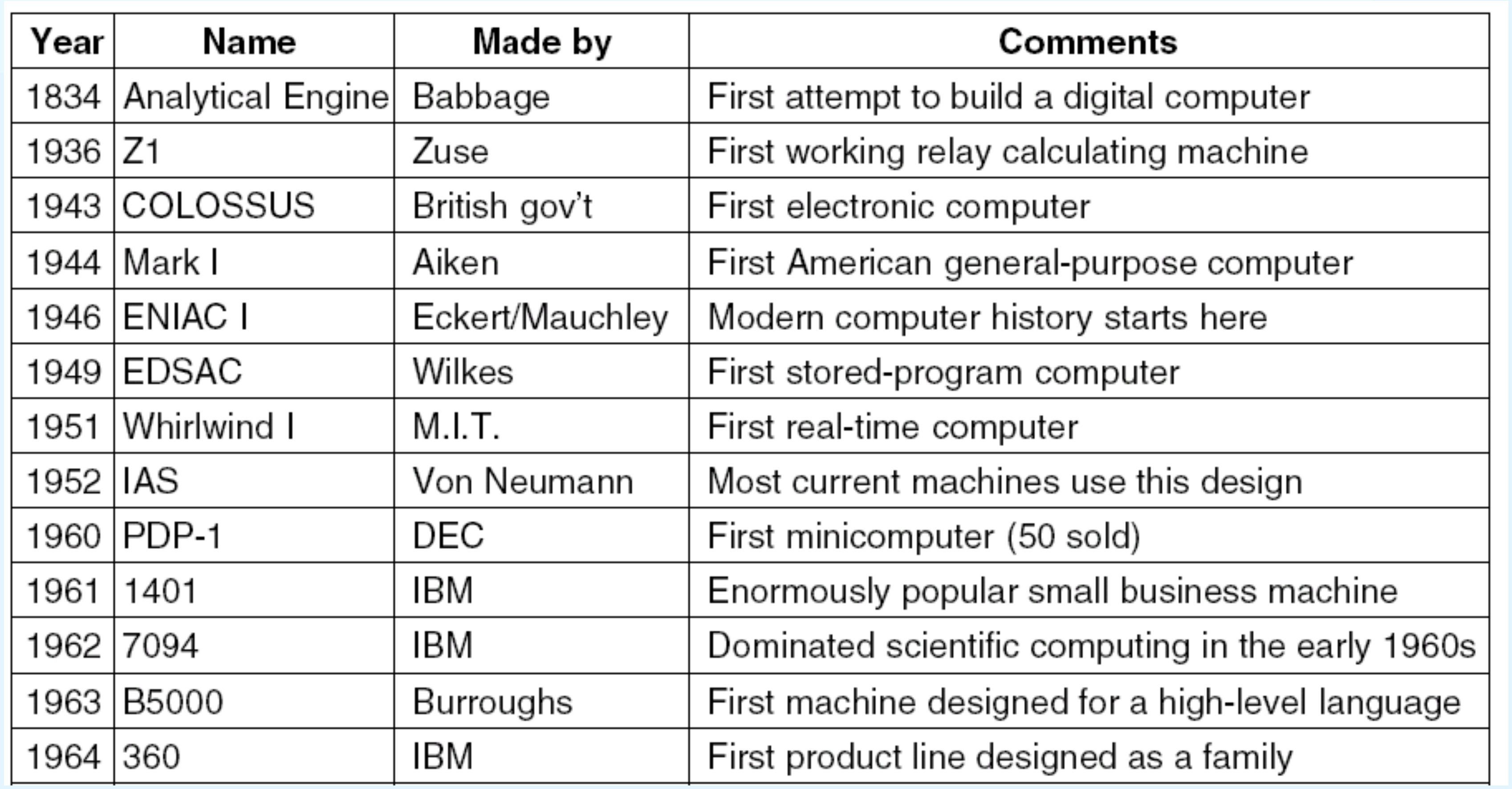
- The Second Generation: Transistorized Computers (1954 - 1965)



An array of magnetic core memory – very expensive – $1 million for 1 Mbyte!

1.5 Historical Development

- Milestones in computer architecture



Year	Name	Made by	Comments
1834	Analytical Engine	Babbage	First attempt to build a digital computer
1936	Z1	Zuse	First working relay calculating machine
1943	COLOSSUS	British gov't	First electronic computer
1944	Mark I	Aiken	First American general-purpose computer
1946	ENIAC I	Eckert/Mauchley	Modern computer history starts here
1949	EDSAC	Wilkes	First stored-program computer
1951	Whirlwind I	M.I.T.	First real-time computer
1952	IAS	Von Neumann	Most current machines use this design
1960	PDP-1	DEC	First minicomputer (50 sold)
1961	1401	IBM	Enormously popular small business machine
1962	7094	IBM	Dominated scientific computing in the early 1960s
1963	B5000	Burroughs	First machine designed for a high-level language
1964	360	IBM	First product line designed as a family

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.
- By this time, IBM had gained overwhelming dominance in the industry.
  - Computer manufacturers of this era were characterized as IBM and the BUNCH (Burroughs, Unisys, NCR, Control Data, and Honeywell).

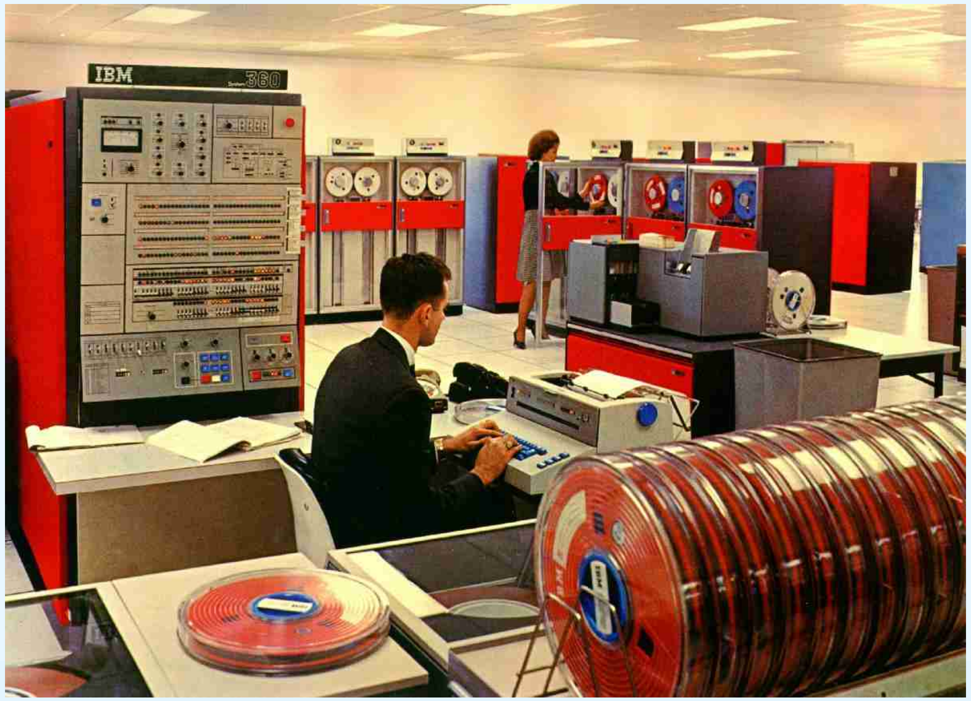
1.5 Historical Development

- The Third Generation: Integrated Circuit Computers (1965 – 1980)
  - Thousands of transistors on a single chip
  - Semiconductor memory
  - 2 MB memory, 5 MIPS
  - Use of cache memory
  - Timesharing, graphics, structured programming

Silicon chips now contained both logic (CPU) and memory

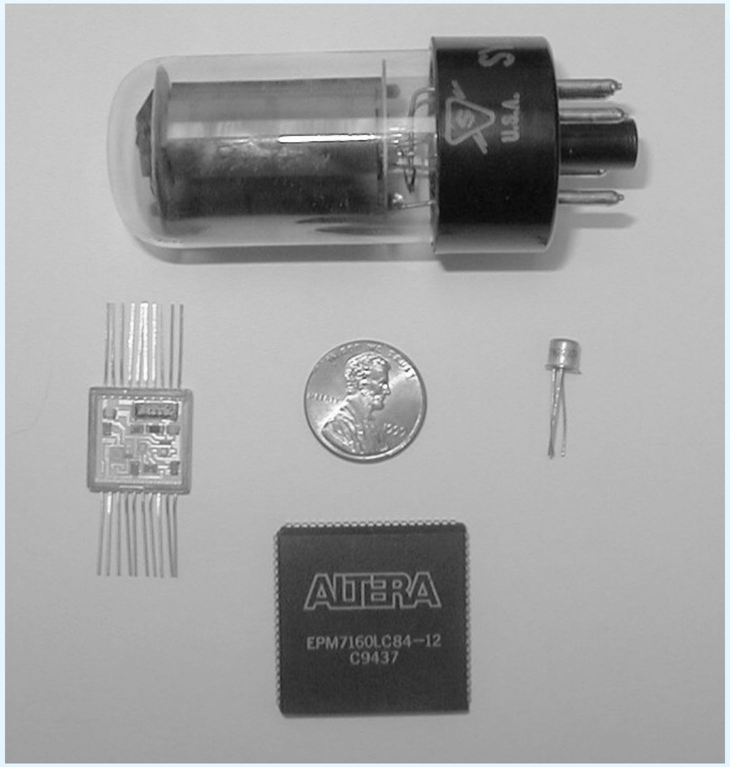
1.5 Historical Development

- IBM 360



1.5 Historical Development

- Comparison of computer components



Vacuum tube

Integrated circuit package

Transistor

Chip
(3200 NAND gates)

1.5 Historical Development

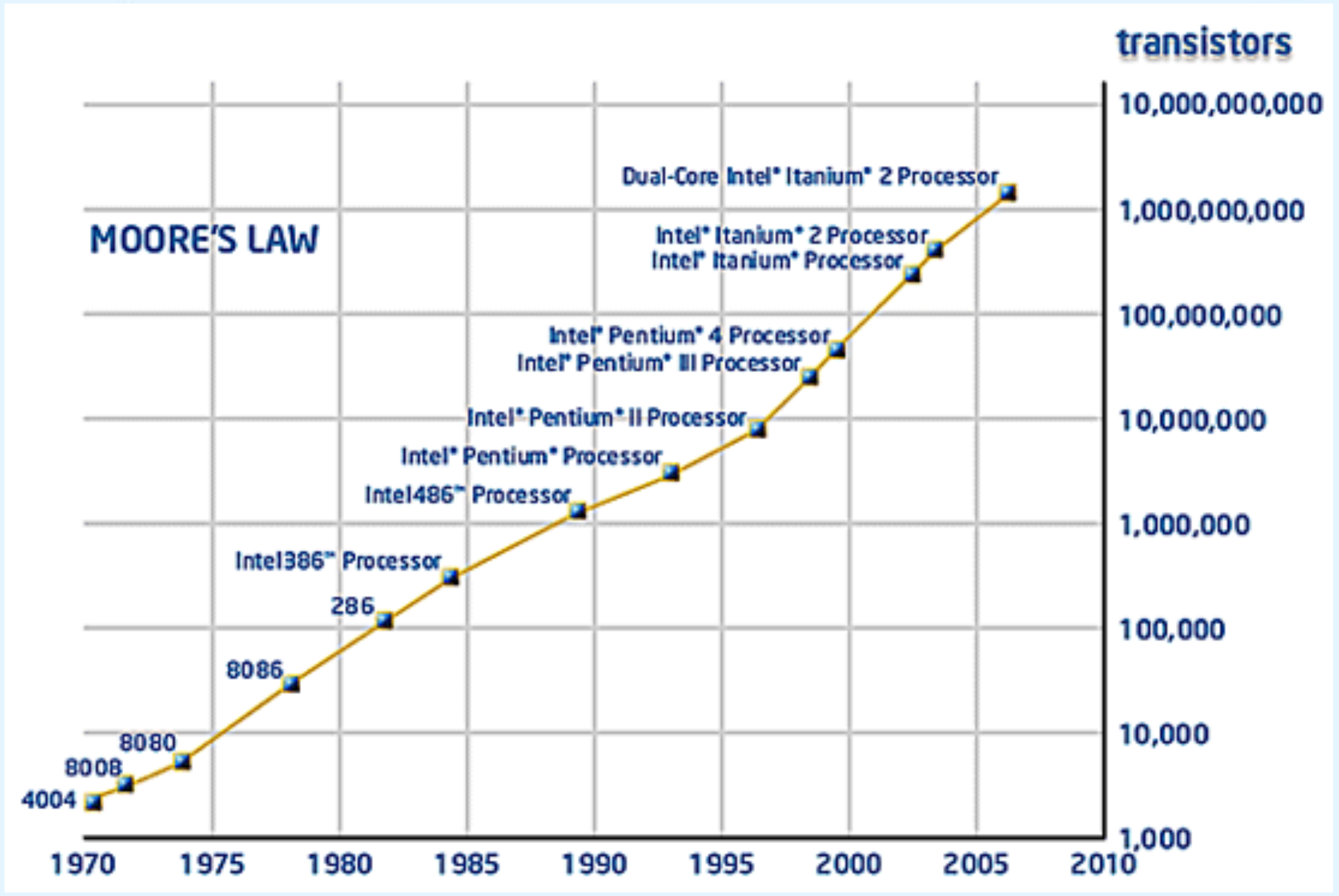
- The Fourth Generation: VLSI Computers (1980 - ????)
  - Very large scale integrated circuits (VLSI) have more than 10,000 components per chip.
  - Enabled the creation of microprocessors.
  - The first was the 4-bit Intel 4004.
  - Later versions, such as the 8080, 8086, and 8088 spawned the idea of "personal computing."

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



The growth has meant an increase in transistor count (and therefore memory capacity and CPU capability) of about 2^{20} since 1965, or computers 1 million times more capable!

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 2005, a chip plant under construction 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.

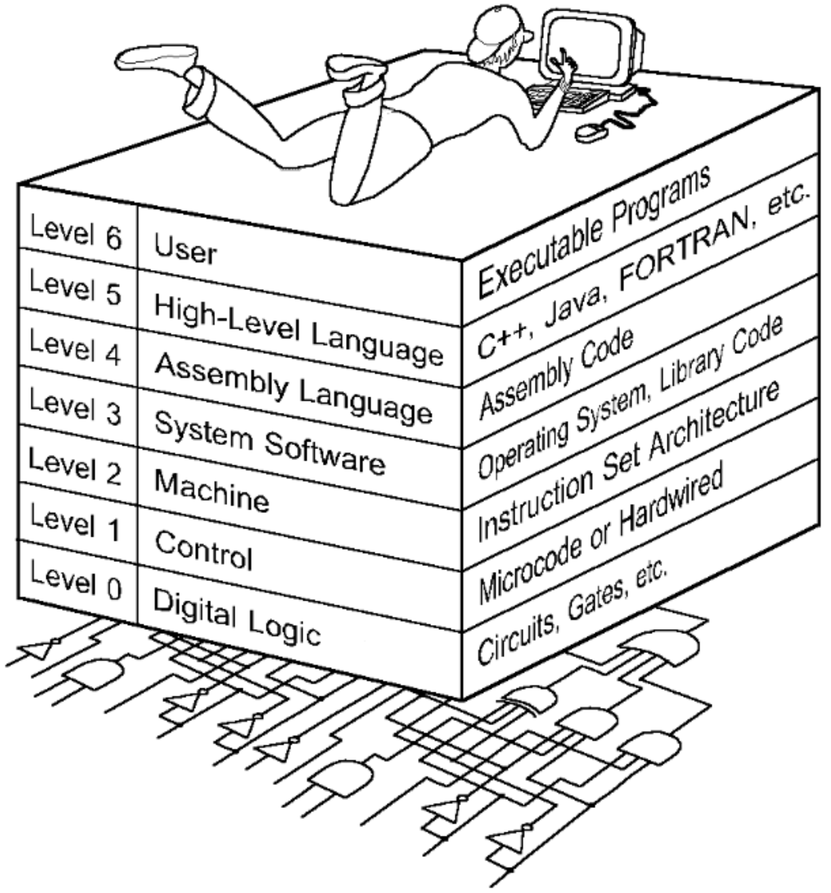
  - For Moore´s Law to hold, Rock´s Law must fall, or vice versa. But no one can say which will give out first.

1.6 The Computer Level Hierarchy

- Computers consist of many things besides chips.
- 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.**



Level 6 User Executable Programs
Level 5 High-Level Language C++, Java, FORTRAN, etc.
Level 4 Assembly Language Assembly Code
Level 3 System Software Operating System, Library Code
Level 2 Machine Instruction Set Architecture
Level 1 Control Microcode or Hardwired
Level 0 Digital Logic Circuits, Gates, etc.

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.
- 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.

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.
- Level 3: System Software Level
  - Controls executing processes on the system.
  - Protects system resources.
  - 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.

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.
- Programming the computer involved moving plugs and wires.
- A different hardware configuration was needed to solve every unique problem type.

Configuring the ENIAC to solve a “simple” problem required many days labor by skilled technicians.

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.
- The invention of this idea has since been ascribed to a mathematician, John von Neumann, who was a contemporary of Mauchley and Eckert.
- Stored-program computers have become known as von Neumann Architecture systems.

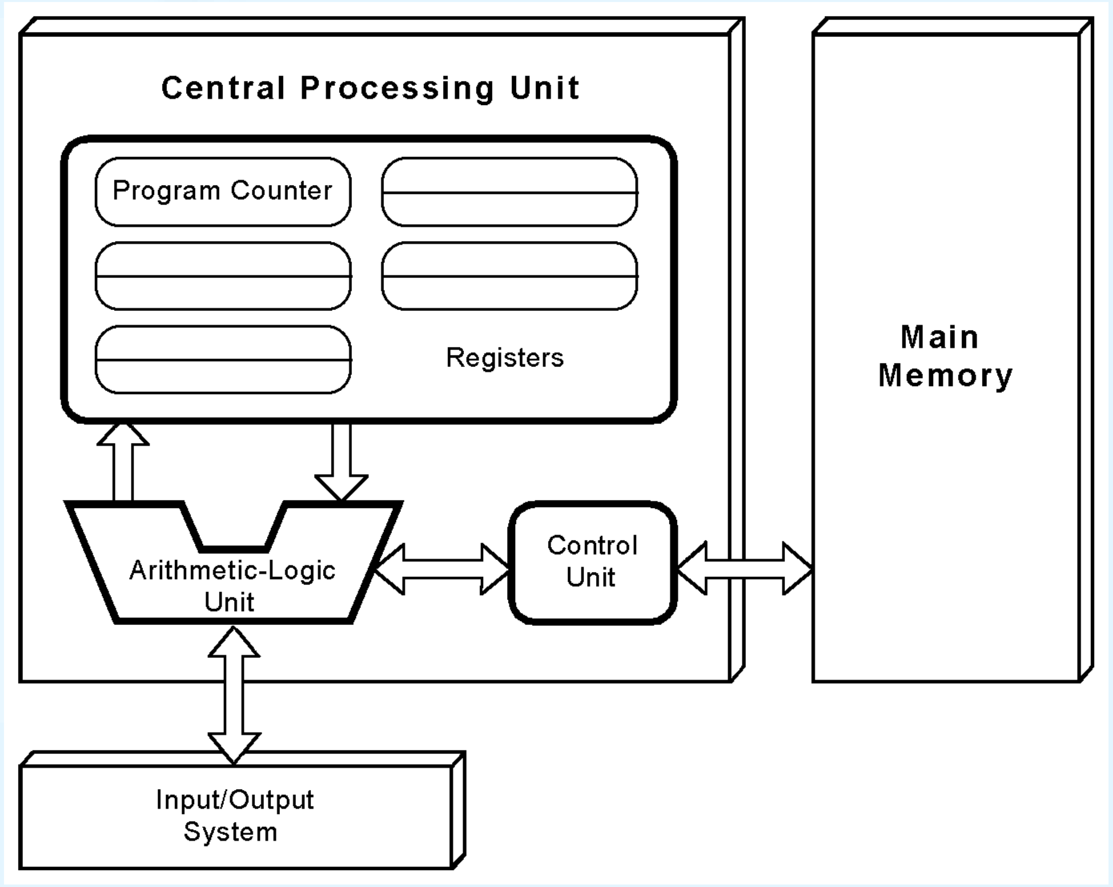
1.7 The von Neumann Model

- 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

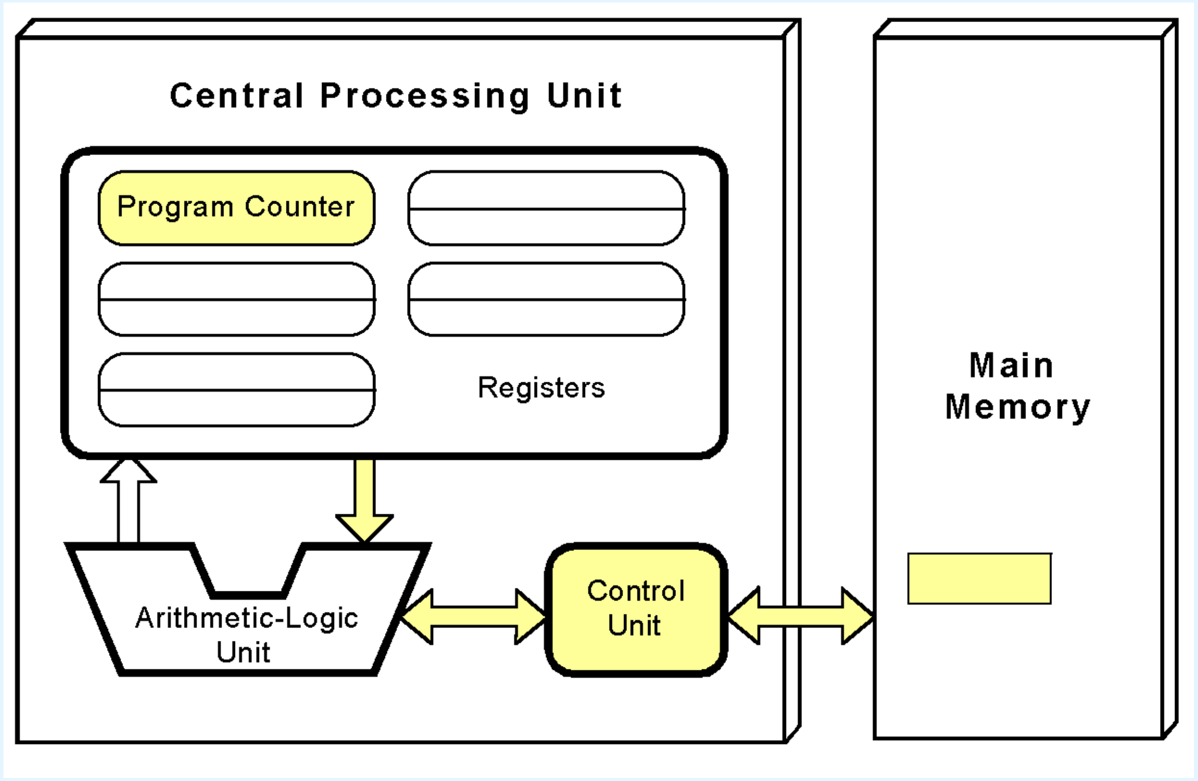
- This is a general depiction of a von Neumann system:

- These computers employ a **fetch-decode-execute cycle** to run programs as follows . . .



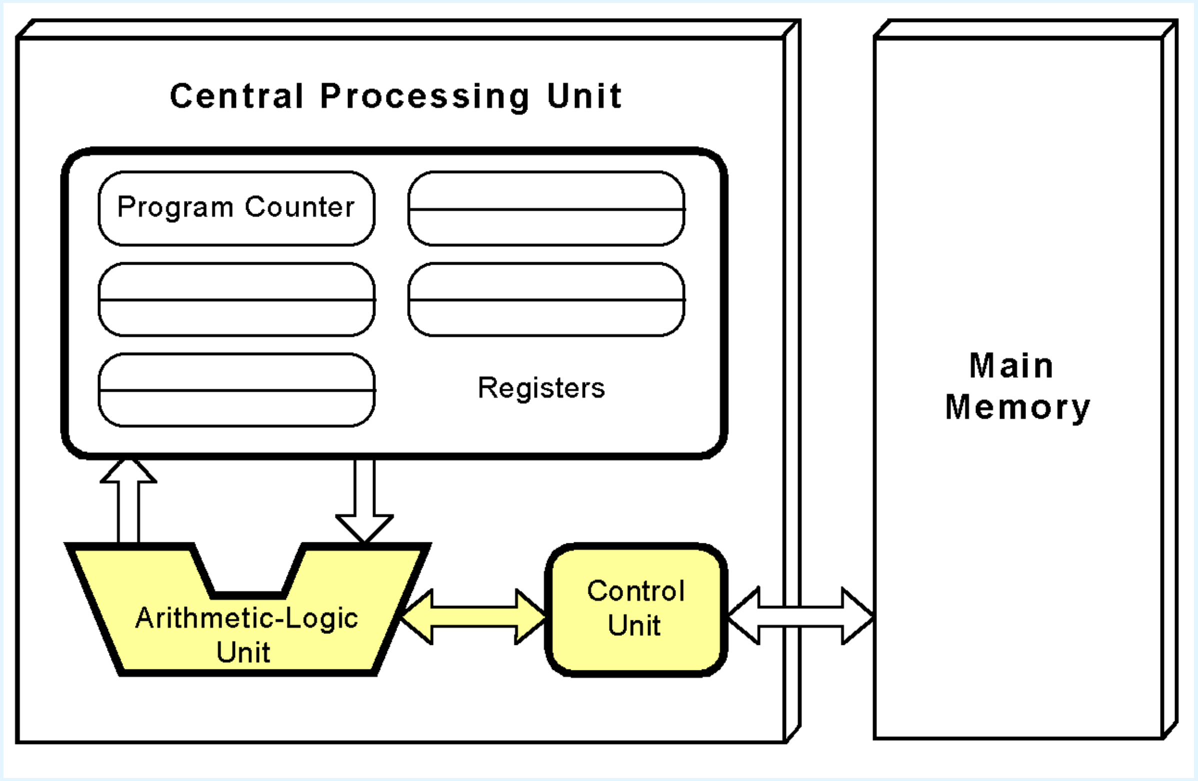
1.7 The von Neumann Model

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



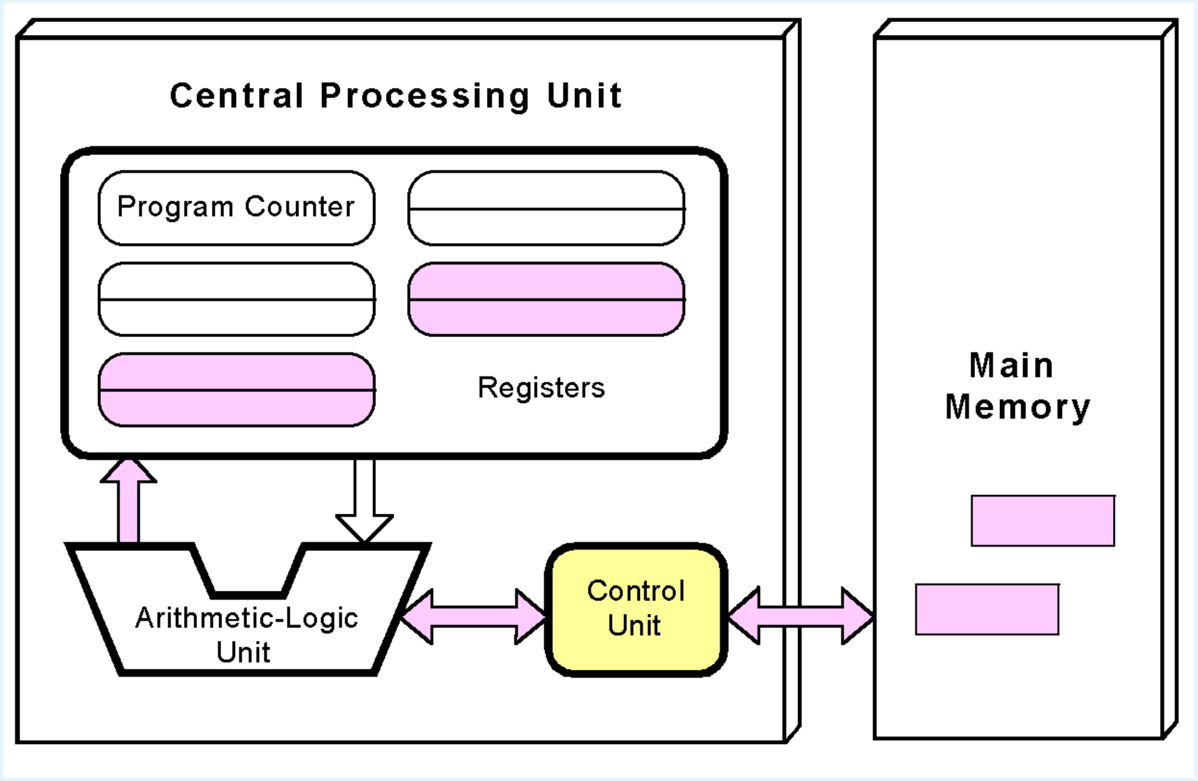
1.7 The von Neumann Model

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



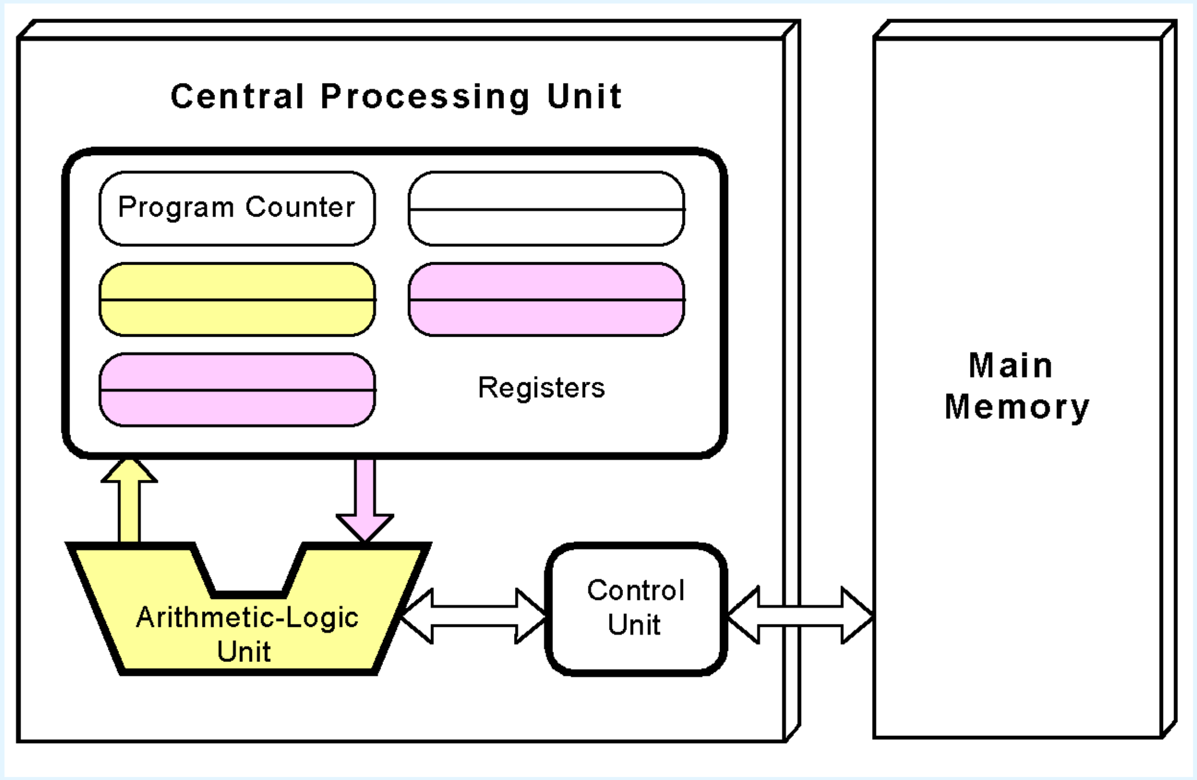
1.7 The von Neumann Model

- **Any data operands required to execute the instruction are fetched from memory and placed into registers within the CPU.**



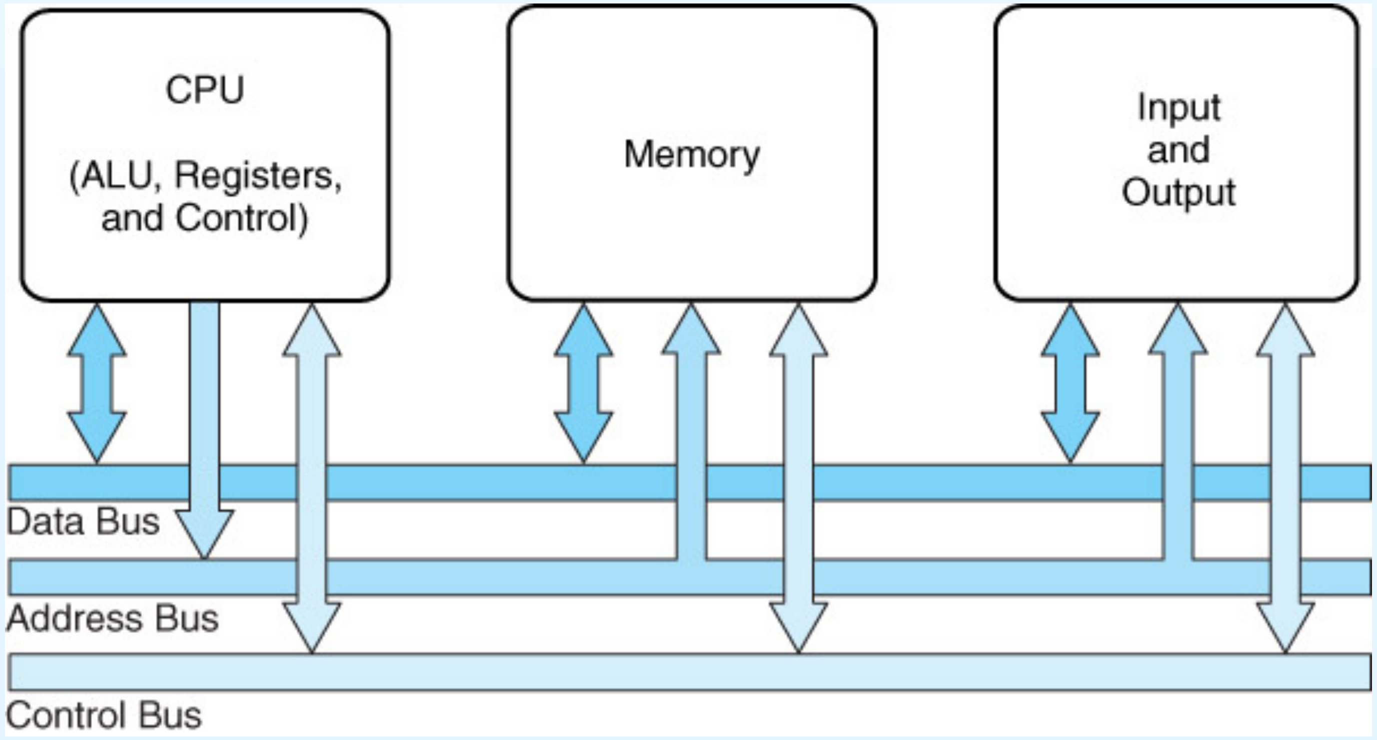
1.7 The von Neumann Model

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



1.7 The von Neumann Model

- **The Modified von Neumann Model. Adding a system bus.**



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

- Multicore architectures have multiple CPUs on a single chip.
- Dual-core and quad-core chips are commonplace in desktop systems.
- Multi-core systems provide the ability to multitask
  - E.g., browse the Web while burning a CD
- Multithreaded applications spread mini-processes, *threads*, across one or more processors for increased throughput.

1.8 Non-von Neumann Models

Amdahl's Law states that overall performance enhancement is limited by the slower parts of the system.

Premise: Every algorithm has a sequential part that ultimately limits the speedup that can be achieved by multiprocessor implementations.

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.

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.

End of Chapter 1