

**Bio 595**

**Computers in  
Biomedical Research  
Class notes set 1  
Fall 2003**

# **Computer impact on science, medicine**

**Question: will physicians and scientists be replaced by computers? probably not**

**Will medical technologists be replaced entirely by computers? probably yes**

**Medical technicians are now operators of computerized, automated analytic instruments.**

**Lab techs are evolving the same way.**

**But the scientist and physician require creative thinking, insight, and intuition - computers can't provide this.**

**Computers have created new problems:**

- 1. Information overload: abundance of data. Requires data reduction.**
- 2. Too much precision, accuracy, and detail.**

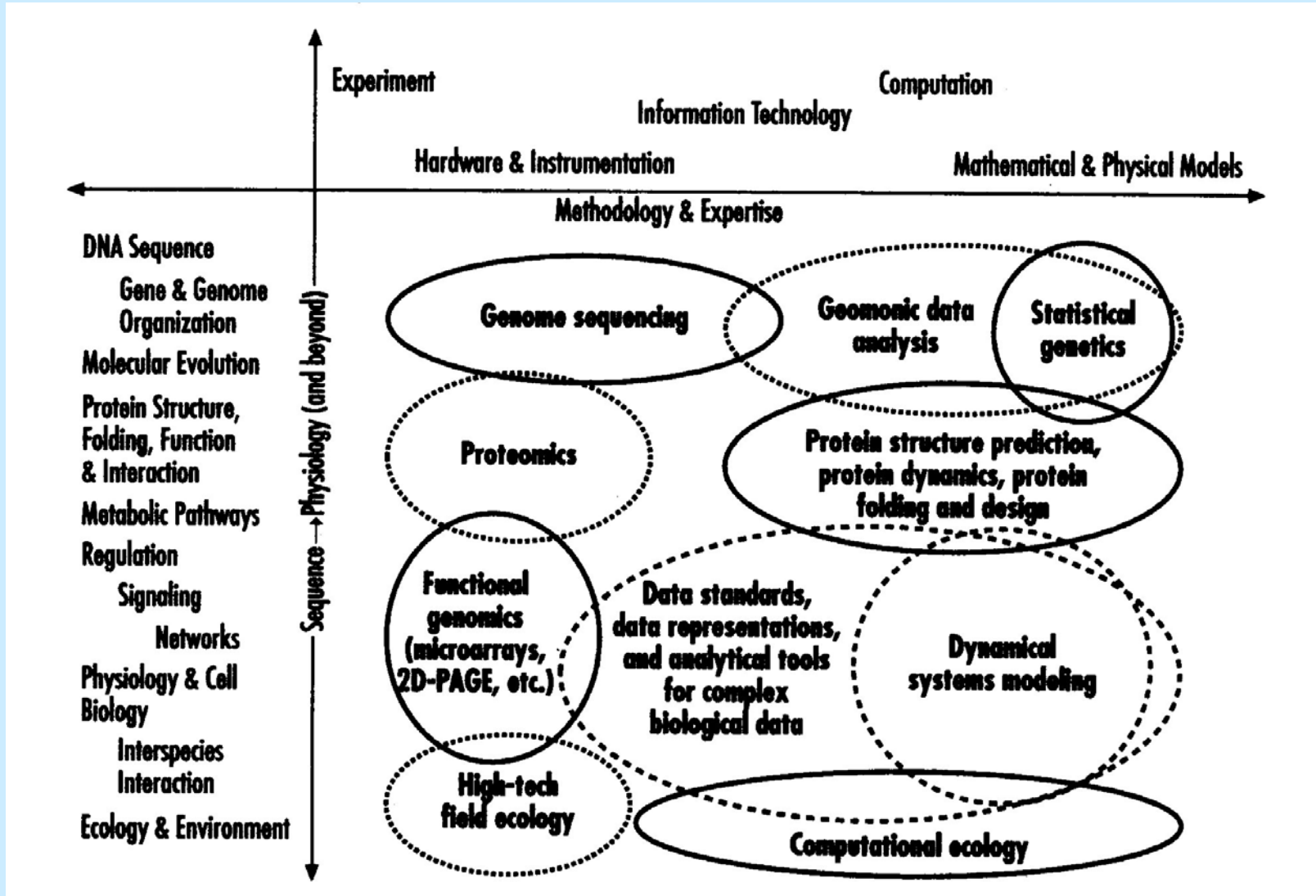
# **General uses of computers in professional applications**

- 1. numerical problem-solving: number-crunching; software for statistics, curve-fitting, differential, simultaneous equations, polynomials, formulae.**
- 2. Data storage, retrieval: inventory, billing, accounts receivable, medical records, bibliographic searches, medical diagnosis symptom libraries.**
- 3. teaching: computer aided (self) instruction, CAI; question-and-answer dialogs.**
- 4. simulation, modeling: use mathematical analog (model) to mimic natural phenomena. Why? Phenomenon too slow, expensive, dangerous, or difficult. Model an imagined world.**

## Uses, continued

- 5. real-time on-line control of processes. Regulate the timing, sequence of steps in a process; closed loop control.**
- 6. telecommunications: data transfer worldwide via InterNet, World Wide Web; local area and broader networks, wireless, video conferencing, PDAs.**
- 7. graphics and picture processing: use display to simplify reduced data.**
- 8. artificial intelligence, decision-making, problem-solving of "word problems". Natural language processing (NLP), expert systems, pattern recognition.**

# Overlap between Biology and Technology



# **Informatics, defined**

**Informatics (information technology, information systems): representation, organization, manipulation, distribution, maintenance, use of recorded information - storage, retrieval, classification, processing.**

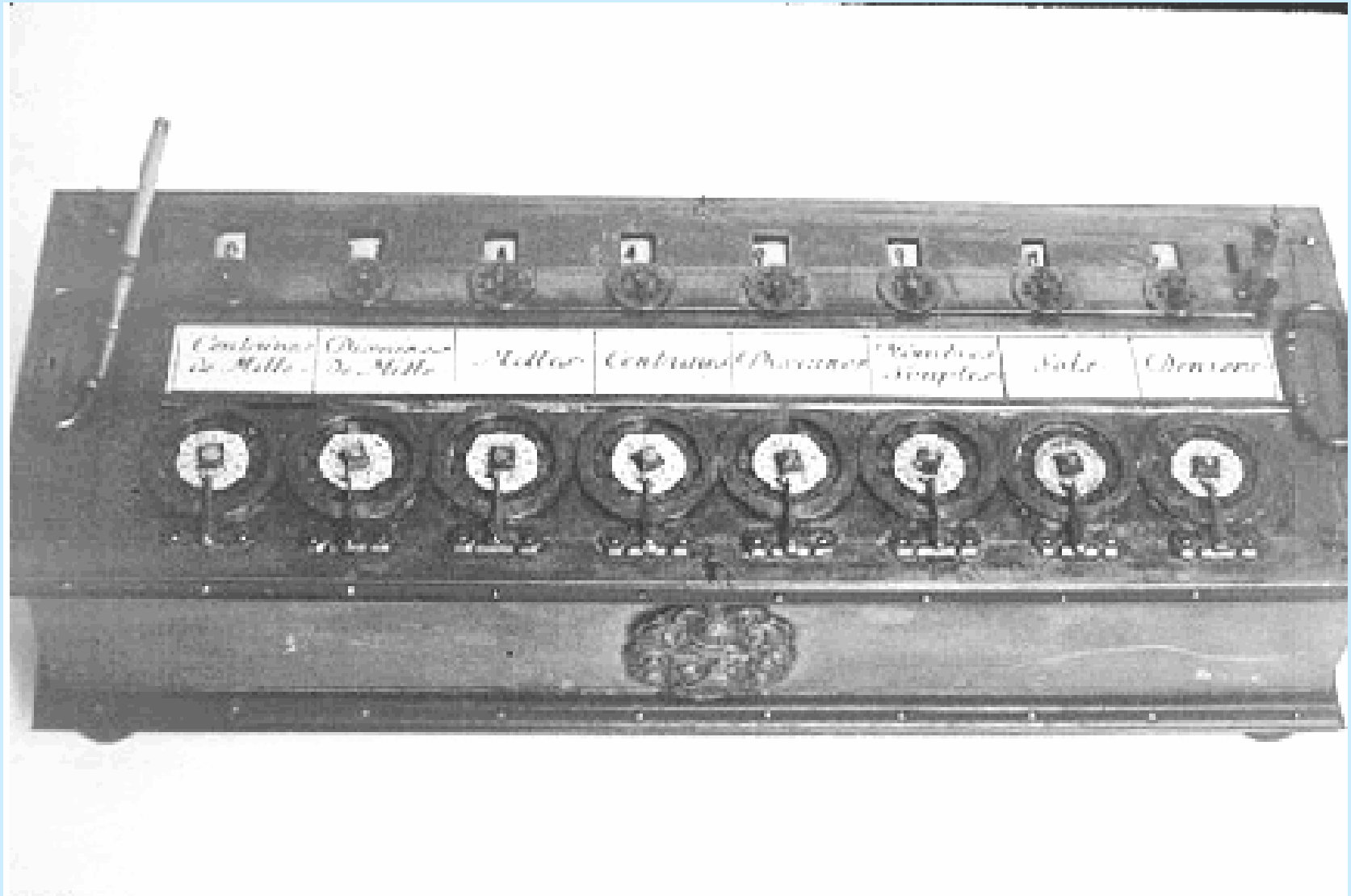
**Medical informatics: using computers to save, retrieve, analyze, extract information from medical databases such as patient hospital records, clinical laboratory reports, billing, and image (e.g. x-ray , MRI) records.**

**Laboratory Information Management Systems (LIMS): manage research laboratory data, control experiments, collect data from lab experiments, reduce, visualize data. (Related: patient care control in CCU, OR).**

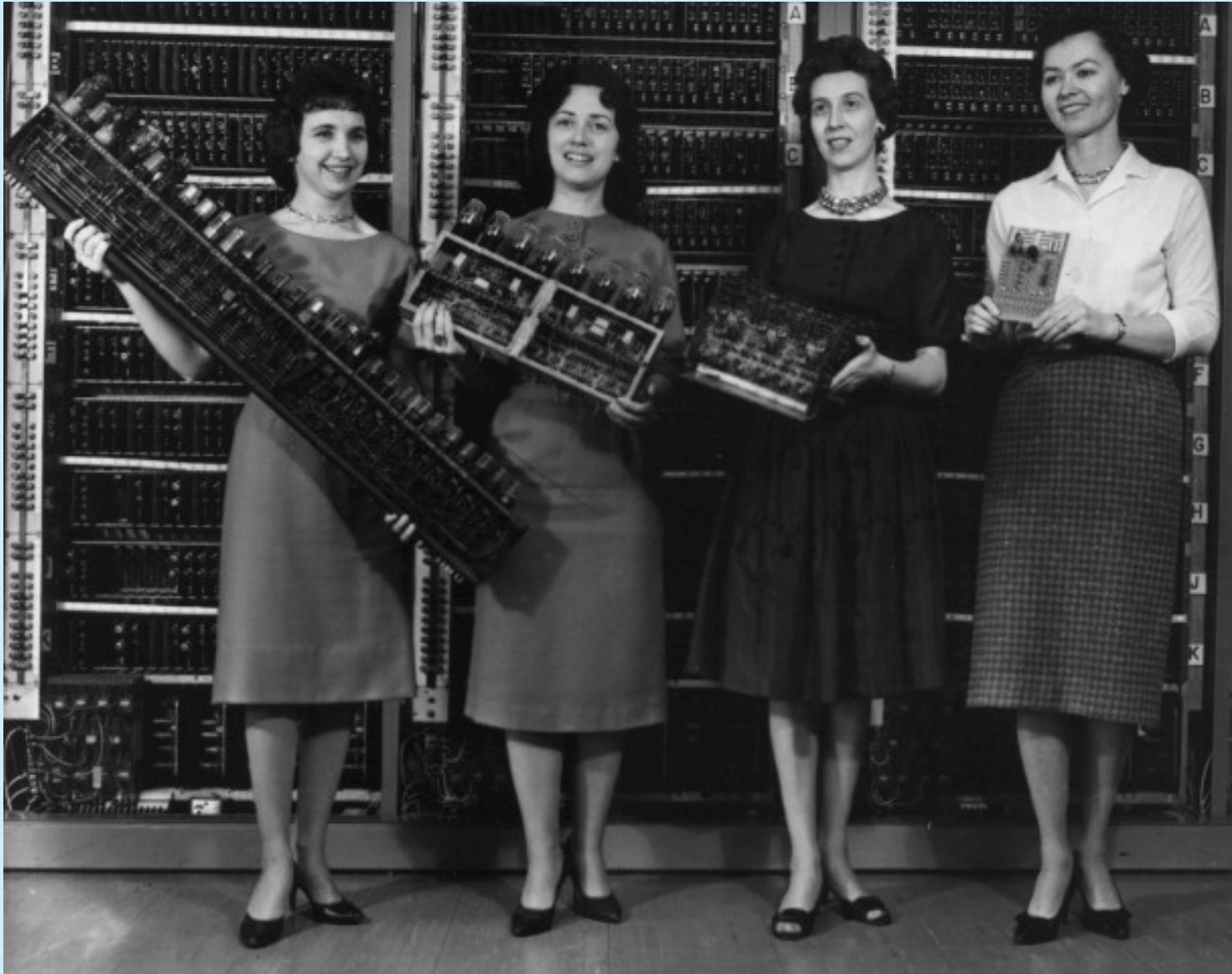
**Computational biology: application of quantitative analytical techniques to model biological systems. Bioinformatics is considered to be a subset of computational biology.**

**Bioinformatics: application of information technology to management of biological data. Most often, finding information in biological sequence or molecular structure databases. Employs algorithms, databases, user interfaces, statistical methods, pattern recognition.**

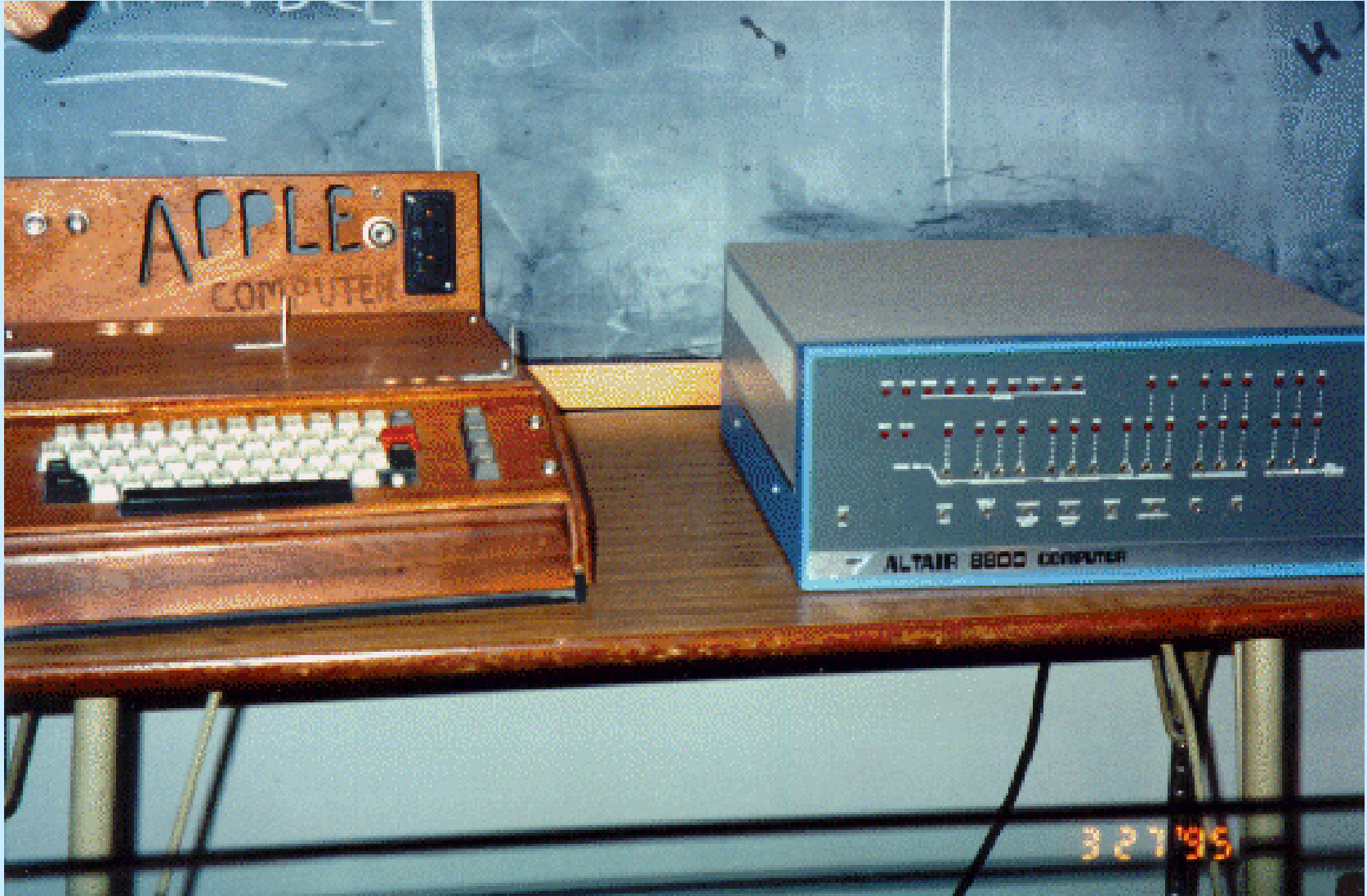
# The Beginning: Pascal's "Computer"



# Eniac, around 1945



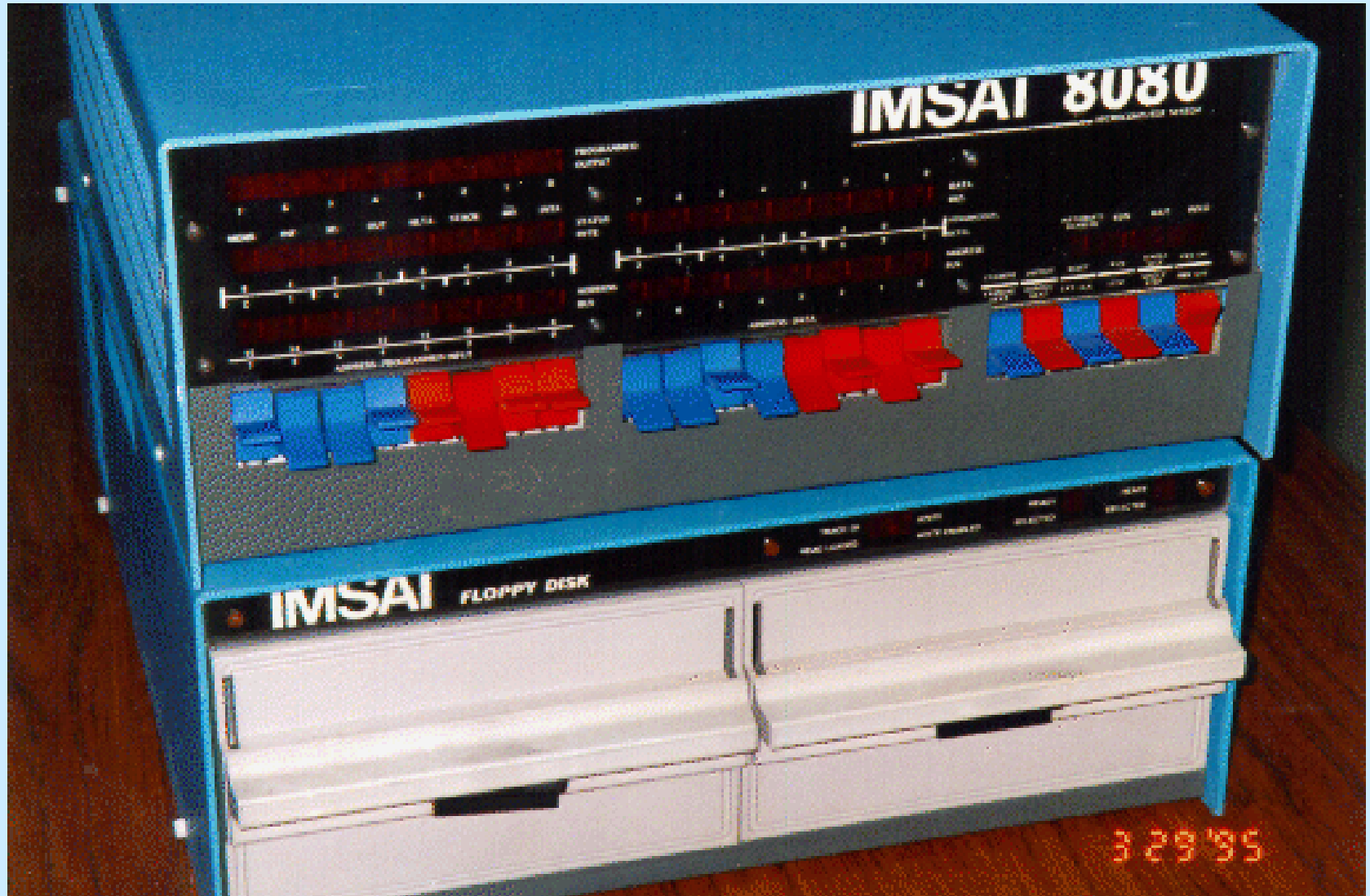
# Apple I and Altair 8800



# The Osborne 1



# IMSAI 8080, an S-100 machine



# Basic Computer Design

**Numbers are represented by discrete numerals coded in binary arithmetic: where 0s and 1s encode data, locations of data, and control codes (instructions that make up the computer program).**

**The circuitry in the computer act like switches rapidly turned on and off ( = 1 or 0, voltages high or low).**

# Terminology

**Memory, bit, byte, word**

**Binary numbers: a number system to base 2:  
digits are 0,1.**

**Word: a packet of binary digits**

**Byte: 8 bit length.**

**Address: the word(s) specifying numeric  
location in memory**

# Meaning of stored numbers?

What does a word value stored in memory "mean"? it is context-dependent; either

- (1) numeric data (perhaps the numeric code representing an alphanumeric character like "A" or "\$") or
- (2) a computer instruction to be interpreted by the machine, commanding it to take some action like "ADD" or
- (3) the address of some other memory location being referenced.

# Terminology...

**What is a computer “instruction? numerically coded command interpreted by computer's circuits**

**CPU: the Central Processor Unit**

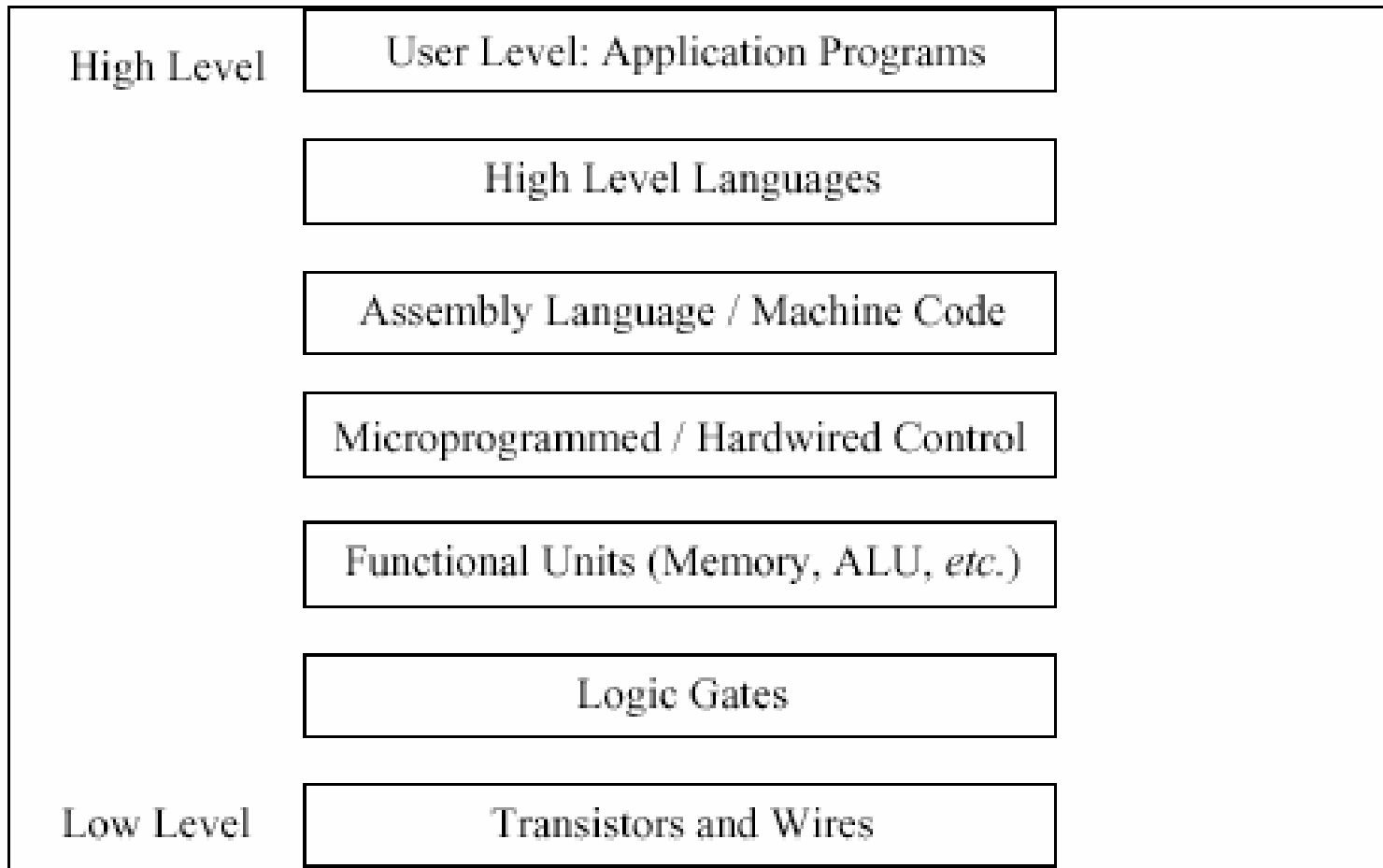
**Register: set of switches in the CPU, 1 or 2 word length, for intermediate results**

**Program: sequence of instructions, coded in binary, to be executed by the computer**

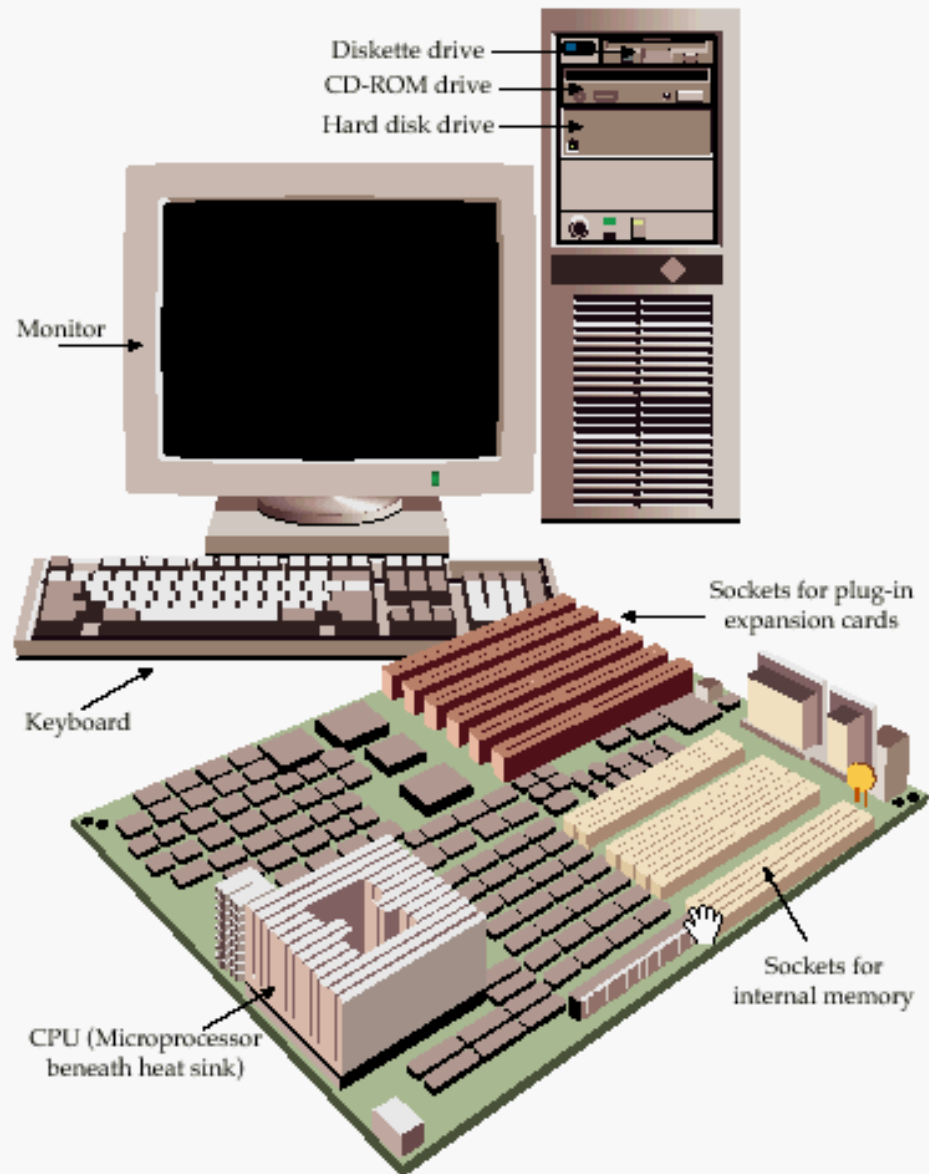
**Cycle time: the interval required by computer to perform one defined CPU operation**

**I/O bus: wiring for transfer to, from devices**

# Levels of Computer Structure



# What's inside?



# Components of the modern computer

- **Central Processing Unit (CPU), the microprocessor “brain” of the computer**
- **The “box” - the system unit, or “tower”**
- **The keyboard, mouse: input for the computer**
- **Circuit boards - daughter-cards that plug into the motherboard**
- **Internal cache memory, built into the CPU**
- **Random Access Memory (RAM) or external cache memory**
- **ROM: Read Only Memory**

# **Parts of the computer, continued**

**Buses: ISA, VESA, PCI**

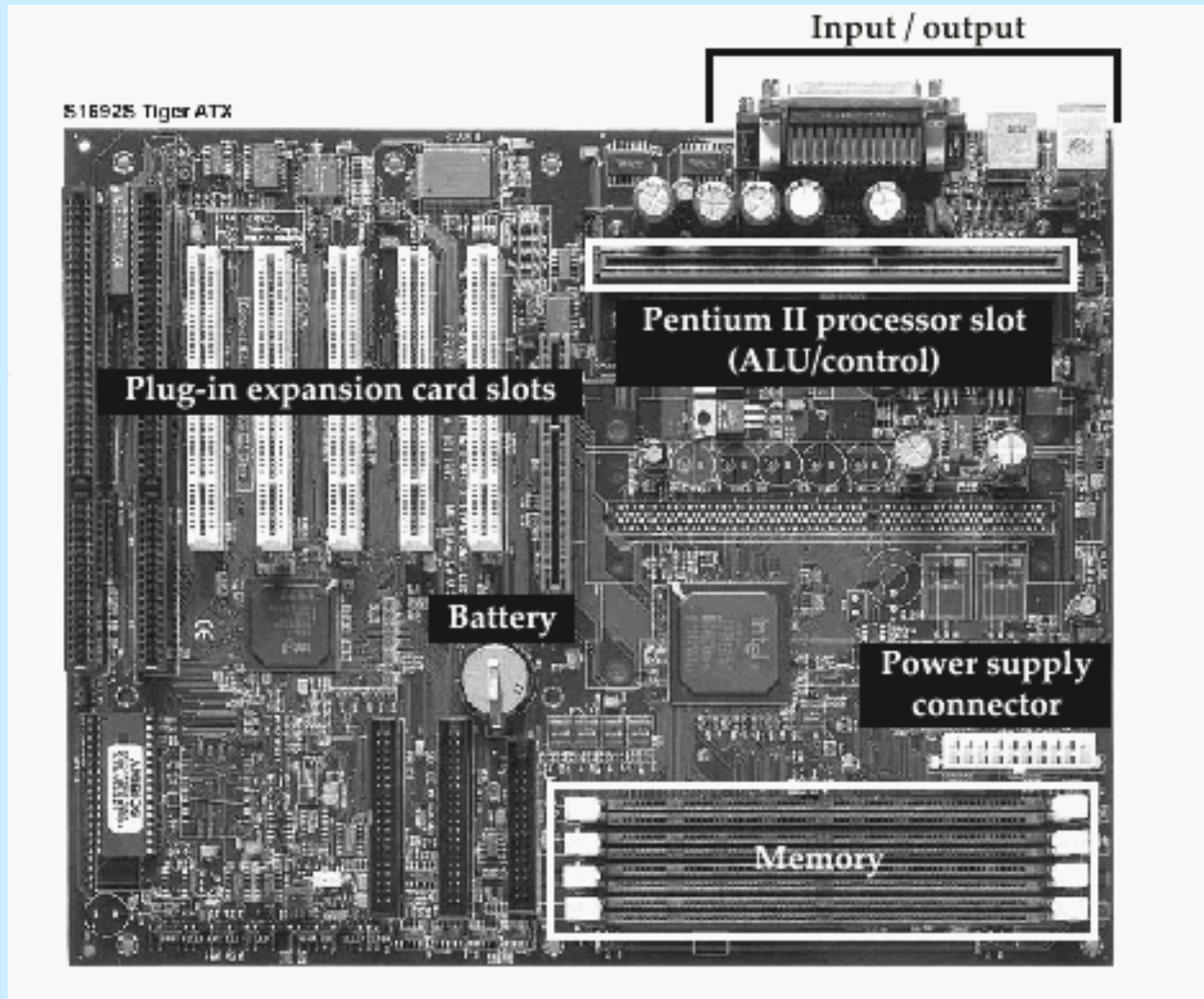
**The Display: desktop monitor (CRT? LCD Flat Screen?)**

**The Disk Drive**

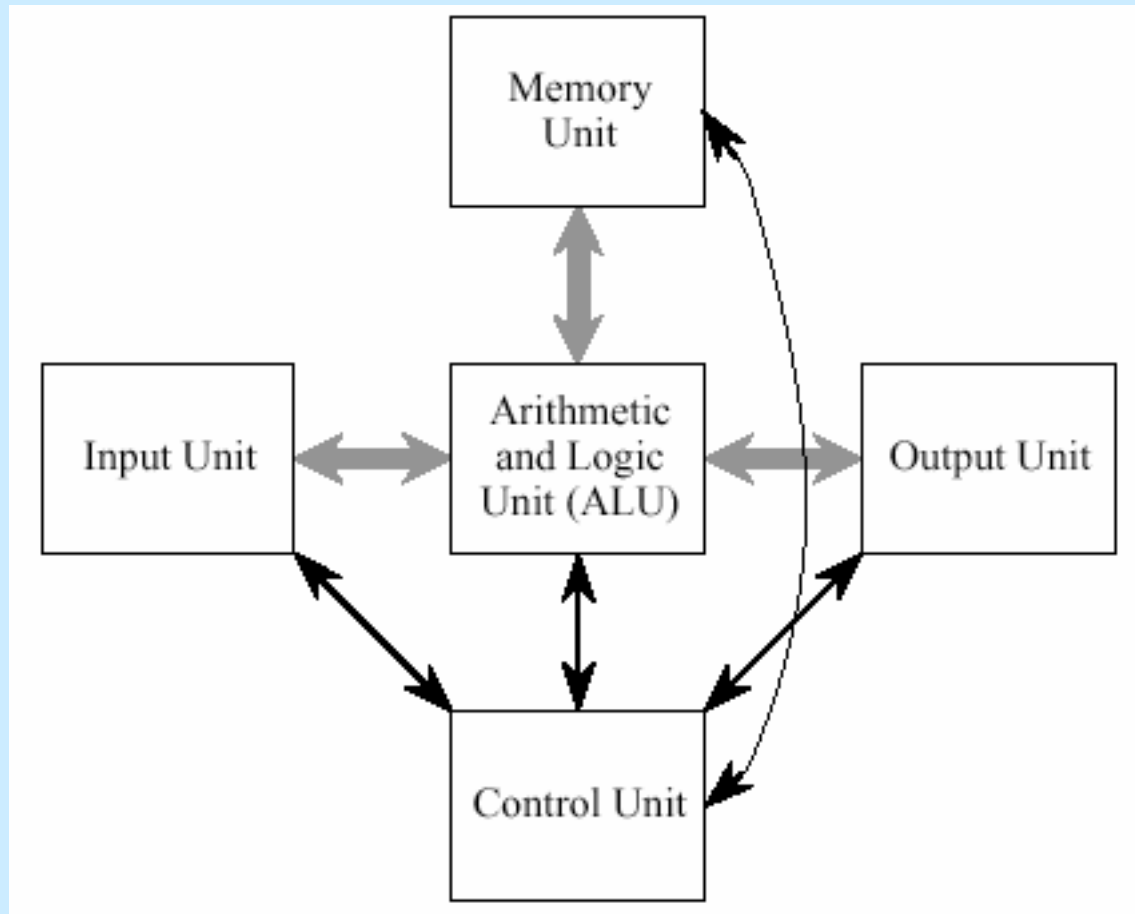
**Removable drives: floppy, cartridge**

**Ports: multi-pin connectors wired to external devices, including keyboards and monitors; serial port, parallel, SCSI, etc.**

# Bare motherboard

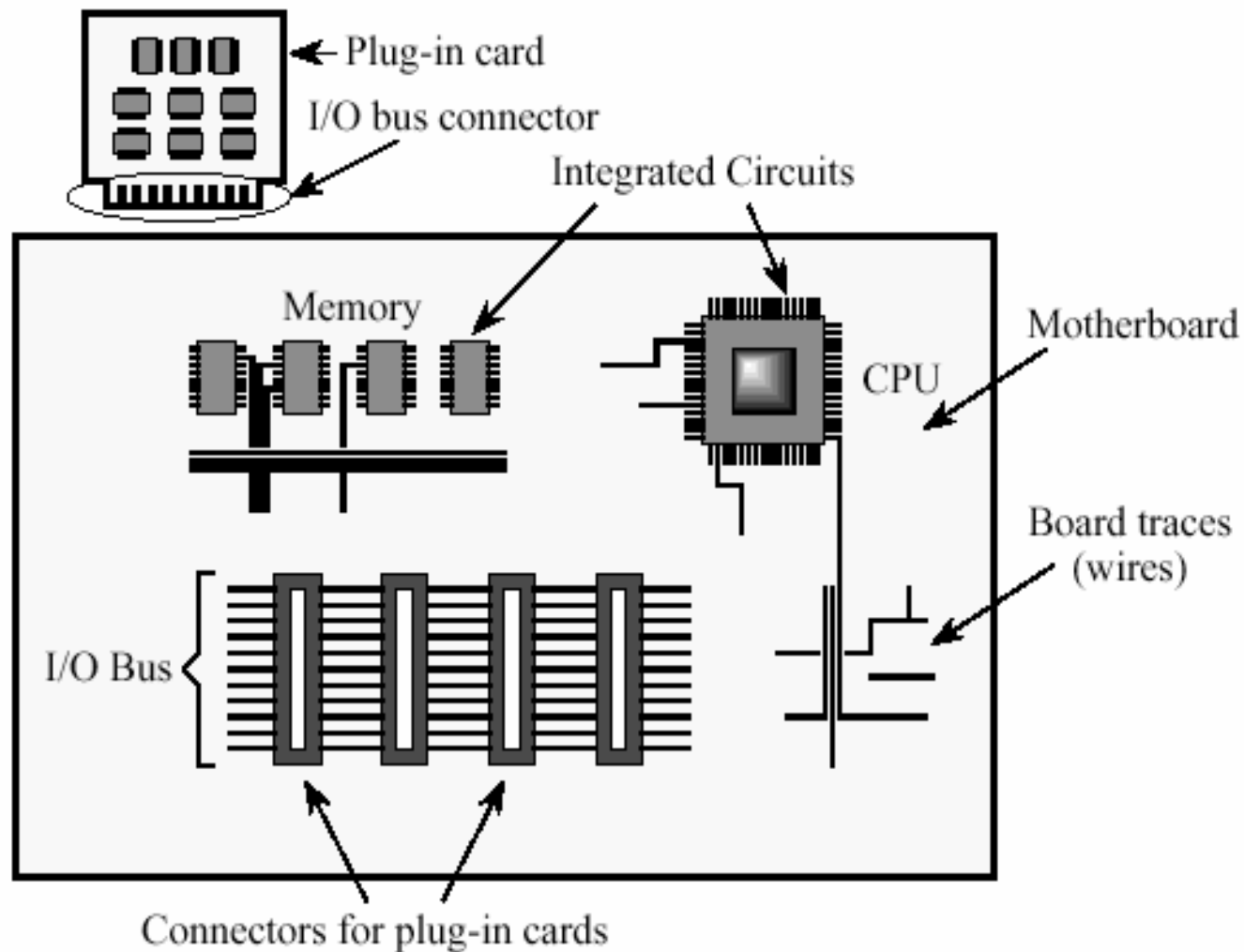


# Basic Elements of the Computer



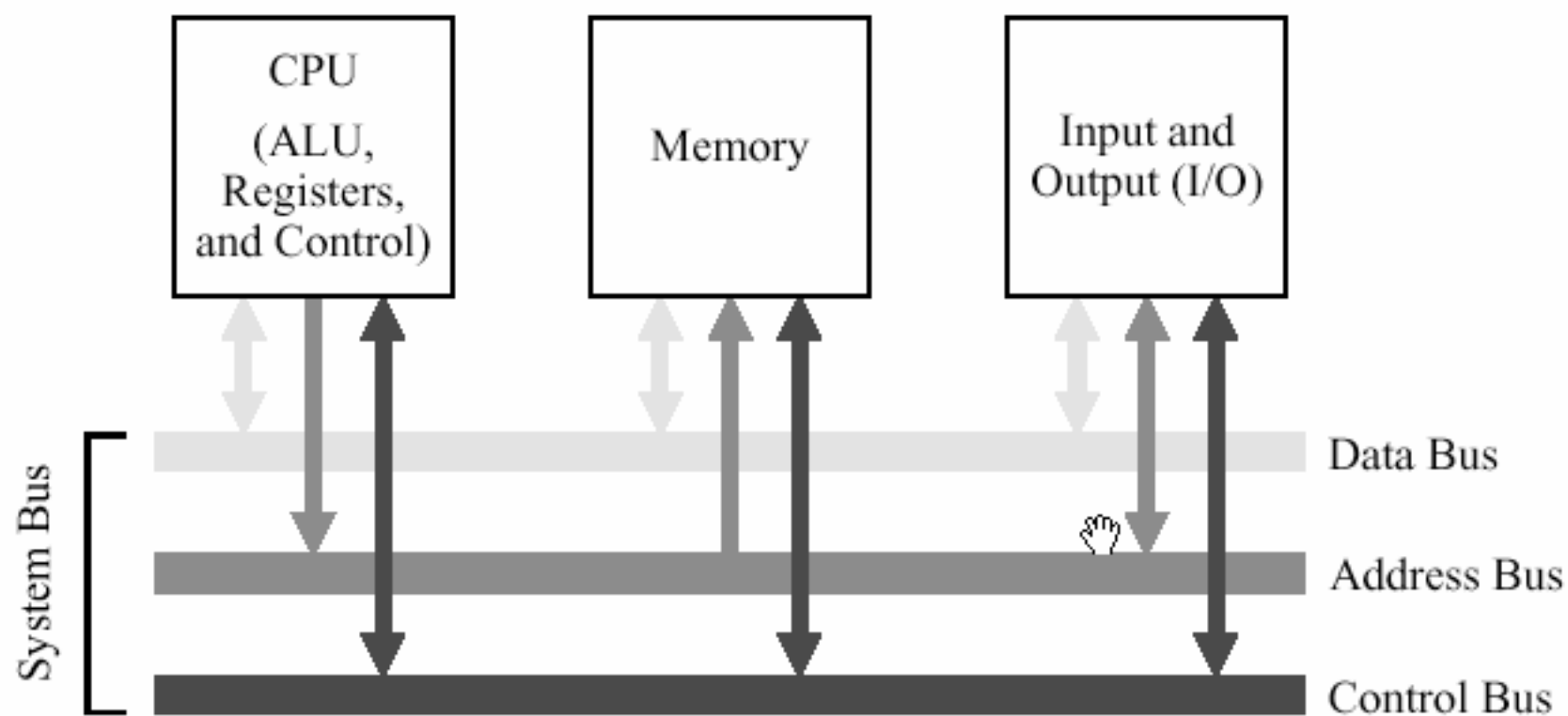
# Simple Bus Architecture

- A simplified motherboard of a personal computer (top view):



# The System Bus Model of a Computer

- A compiled program is copied from a hard disk to the memory. The CPU reads instructions and data from the memory, executes the instructions, and stores the results back into the memory.



# Intel Memory and I/O Address Spaces

Address  
FFFFFFFF

Memory  
Space

00000000

Address  
FFFF

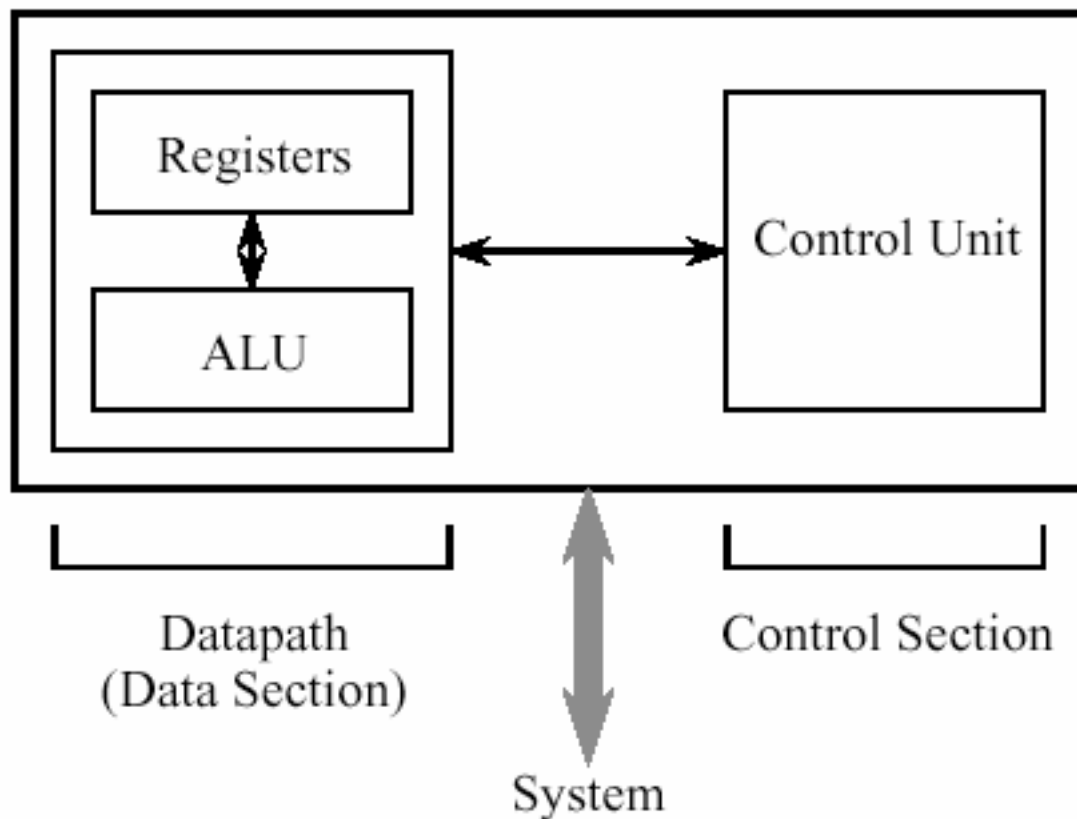
I/O  
Space

0000



# Abstract View of a CPU

- The CPU consists of a data section containing registers and an ALU, and a control section, which interprets instructions and effects register transfers. The data section is also known as the *datapath*.



# Base 2, 8, 10, 16 Number Systems

Binary (base 2)	Octal (base 8)	Decimal (base 10)	Hexadecimal (base 16)
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F

# Binary Addition

Carry bit →	1	1	1		1	1	1		in hexadecimal (base 16)	
First argument →	0	0	1	1	0	0	1	1	3	3
Second argument →	0	1	0	1	0	1	0	1	9	9
	<hr/>								<hr/>	
Answer →	1	0	0	0	1	0	0	0	D	C

# Common Sizes for Data Types

- A byte is composed of 8 bits. Two nibbles make up a byte.
- Halfwords, words, doublewords, and quadwords are composed of bytes as shown below:

Bit	<table border="1"><tr><td>0</td></tr></table>	0			
0					
Nibble	<table border="1"><tr><td>0110</td></tr></table>	0110			
0110					
Byte	<table border="1"><tr><td>10110000</td></tr></table>	10110000			
10110000					
16-bit word (halfword)	<table border="1"><tr><td>11001001 01000110</td></tr></table>	11001001 01000110			
11001001 01000110					
32-bit word	<table border="1"><tr><td>10110100 00110101 10011001 01011000</td></tr></table>	10110100 00110101 10011001 01011000			
10110100 00110101 10011001 01011000					
64-bit word (double)	<table border="1"><tr><td>01011000 01010101 10110000 11110011</td></tr><tr><td>11001110 11101110 01111000 00110101</td></tr></table>	01011000 01010101 10110000 11110011	11001110 11101110 01111000 00110101		
01011000 01010101 10110000 11110011					
11001110 11101110 01111000 00110101					
128-bit word (quad)	<table border="1"><tr><td>01011000 01010101 10110000 11110011</td></tr><tr><td>11001110 11101110 01111000 00110101</td></tr><tr><td>00001011 10100110 11110010 11100110</td></tr><tr><td>10100100 01000100 10100101 01010001</td></tr></table>	01011000 01010101 10110000 11110011	11001110 11101110 01111000 00110101	00001011 10100110 11110010 11100110	10100100 01000100 10100101 01010001
01011000 01010101 10110000 11110011					
11001110 11101110 01111000 00110101					
00001011 10100110 11110010 11100110					
10100100 01000100 10100101 01010001					

# ASCII Character Code

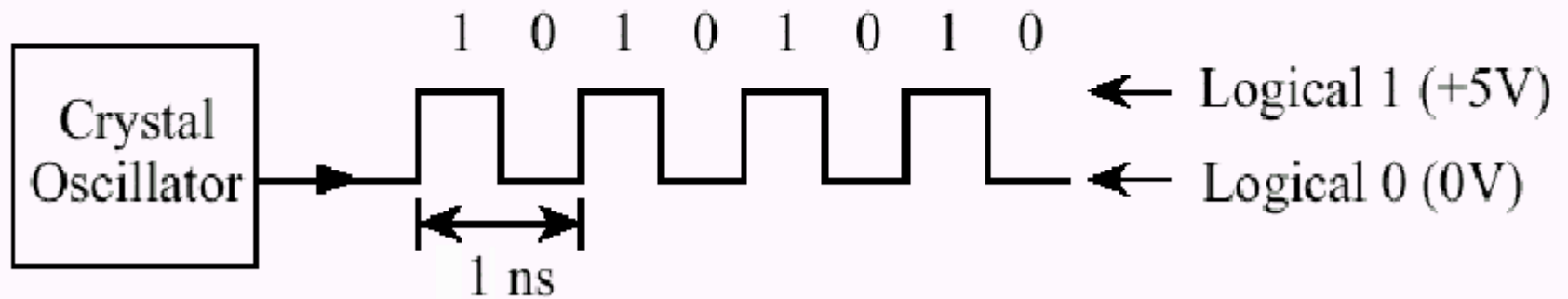
- ASCII is a 7-bit code, commonly stored in 8-bit bytes.
- “A” is at  $41_{16}$ . To convert upper case letters to lower case letters, add  $20_{16}$ . Thus “a” is at  $41_{16} + 20_{16} = 61_{16}$ .
- The character “5” at position  $35_{16}$  is different than the number 5. To convert character-numbers into number-numbers, subtract  $30_{16}$ :  $35_{16} - 30_{16} = 5$ .

00 NUL	10 DLE	20 SP	30 0	40 @	50 P	60 `	70 p
01 SOH	11 DC1	21 !	31 1	41 A	51 Q	61 a	71 q
02 STX	12 DC2	22 "	32 2	42 B	52 R	62 b	72 r
03 ETX	13 DC3	23 #	33 3	43 C	53 S	63 c	73 s
04 EOT	14 DC4	24 \$	34 4	44 D	54 T	64 d	74 t
05 ENQ	15 NAK	25 %	35 5	45 E	55 U	65 e	75 u
06 ACK	16 SYN	26 &	36 6	46 F	56 V	66 f	76 v
07 BEL	17 ETB	27 '	37 7	47 G	57 W	67 g	77 w
08 BS	18 CAN	28 (	38 8	48 H	58 X	68 h	78 x
09 HT	19 EM	29 )	39 9	49 I	59 Y	69 i	79 y
0A LF	1A SUB	2A *	3A :	4A J	5A Z	6A j	7A z
0B VT	1B ESC	2B +	3B ;	4B K	5B [	6B k	7B {
0C FF	1C FS	2C ,	3C <	4C L	5C \	6C l	7C
0D CR	1D GS	2D -	3D =	4D M	5D ]	6D m	7D }
0E SO	1E RS	2E .	3E >	4E N	5E ^	6E n	7E ~
0F SI	1F US	2F /	3F ?	4F O	5F _	6F o	7F DEL

NUL	Null	FF	Form feed	CAN	Cancel
SOH	Start of heading	CR	Carriage return	EM	End of medium
STX	Start of text	SO	Shift out	SUB	Substitute
ETX	End of text	SI	Shift in	ESC	Escape
EOT	End of transmission	DLE	Data link escape	FS	File separator
ENQ	Enquiry	DC1	Device control 1	GS	Group separator
ACK	Acknowledge	DC2	Device control 2	RS	Record separator
BEL	Bell	DC3	Device control 3	US	Unit separator
BS	Backspace	DC4	Device control 4	SP	Space
HT	Horizontal tab	NAK	Negative acknowledge	DEL	Delete
LF	Line feed	SYN	Synchronous idle		
VT	Vertical tab	ETB	End of transmission block		

# Clock Speed

## 1 GHz Bus Clock

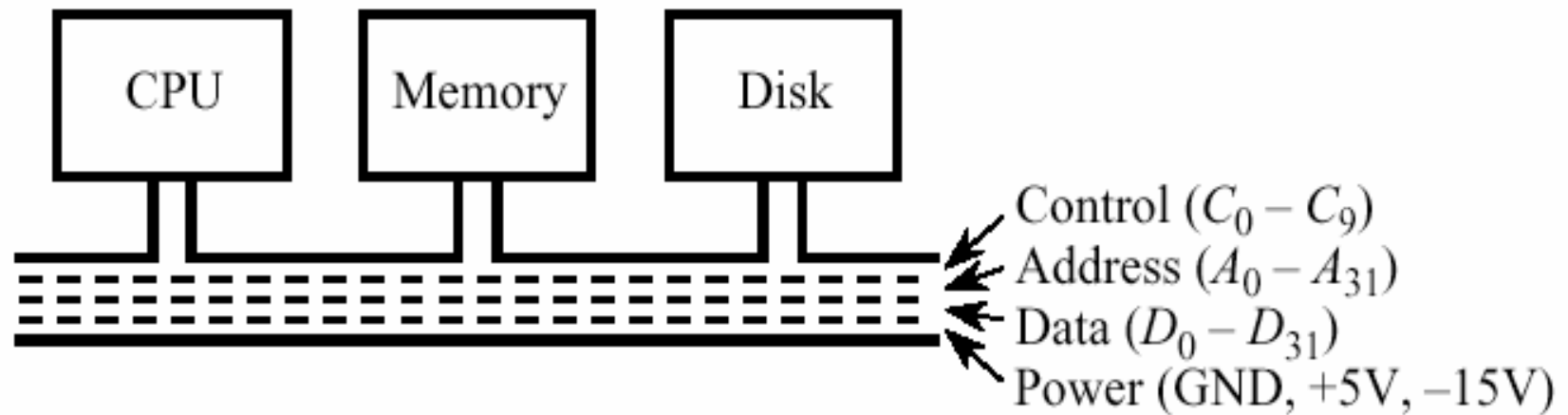


# The Fetch-Execute Cycle

- The steps that the control unit carries out in executing a program are:
  - (1) Fetch the next instruction to be executed from memory.
  - (2) Decode the opcode.
  - (3) Read operand(s) from main memory, if any.
  - (4) Execute the instruction and store results.
  - (5) Go to step 1.

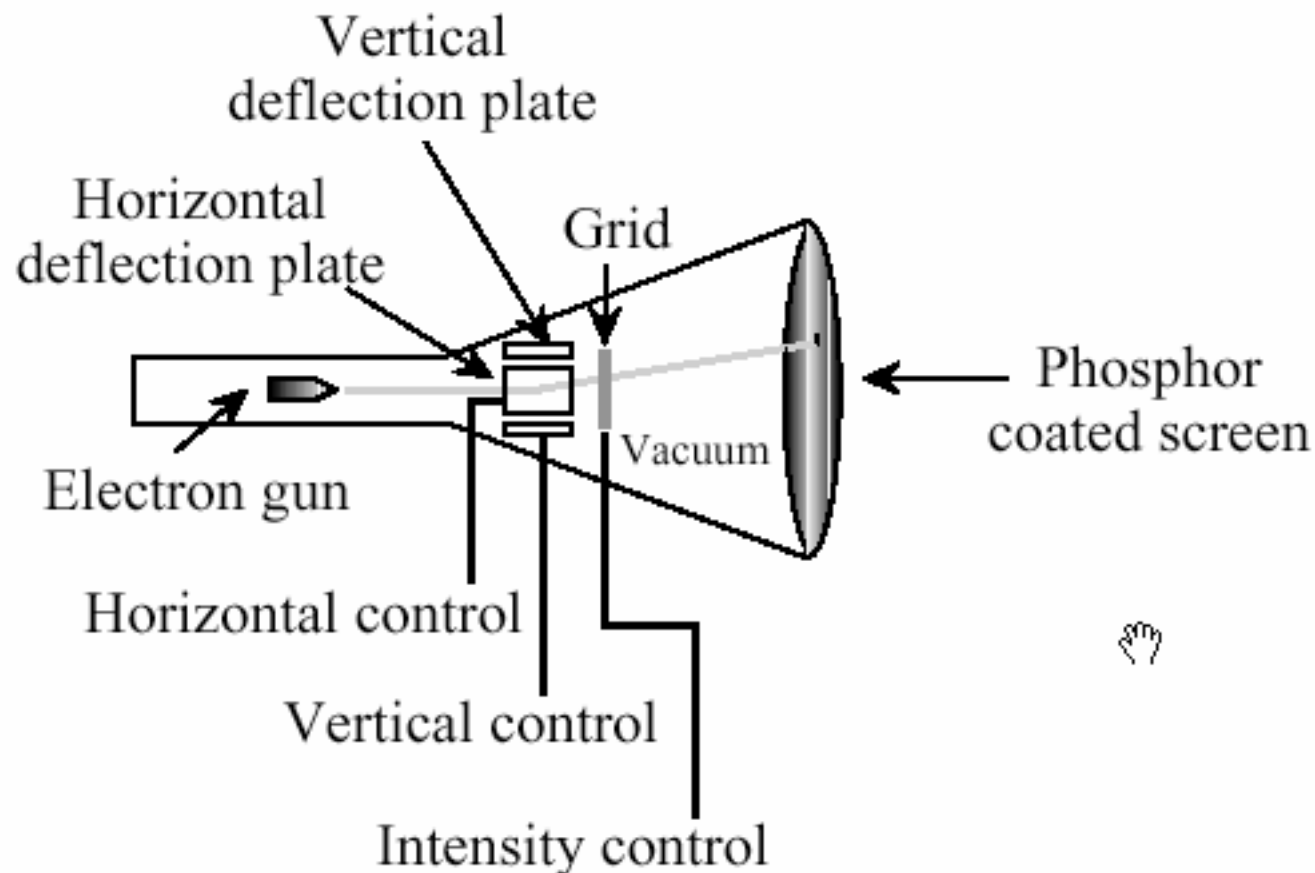
This is known as the *fetch-execute cycle*.

# Simplified Illustration of a Bus



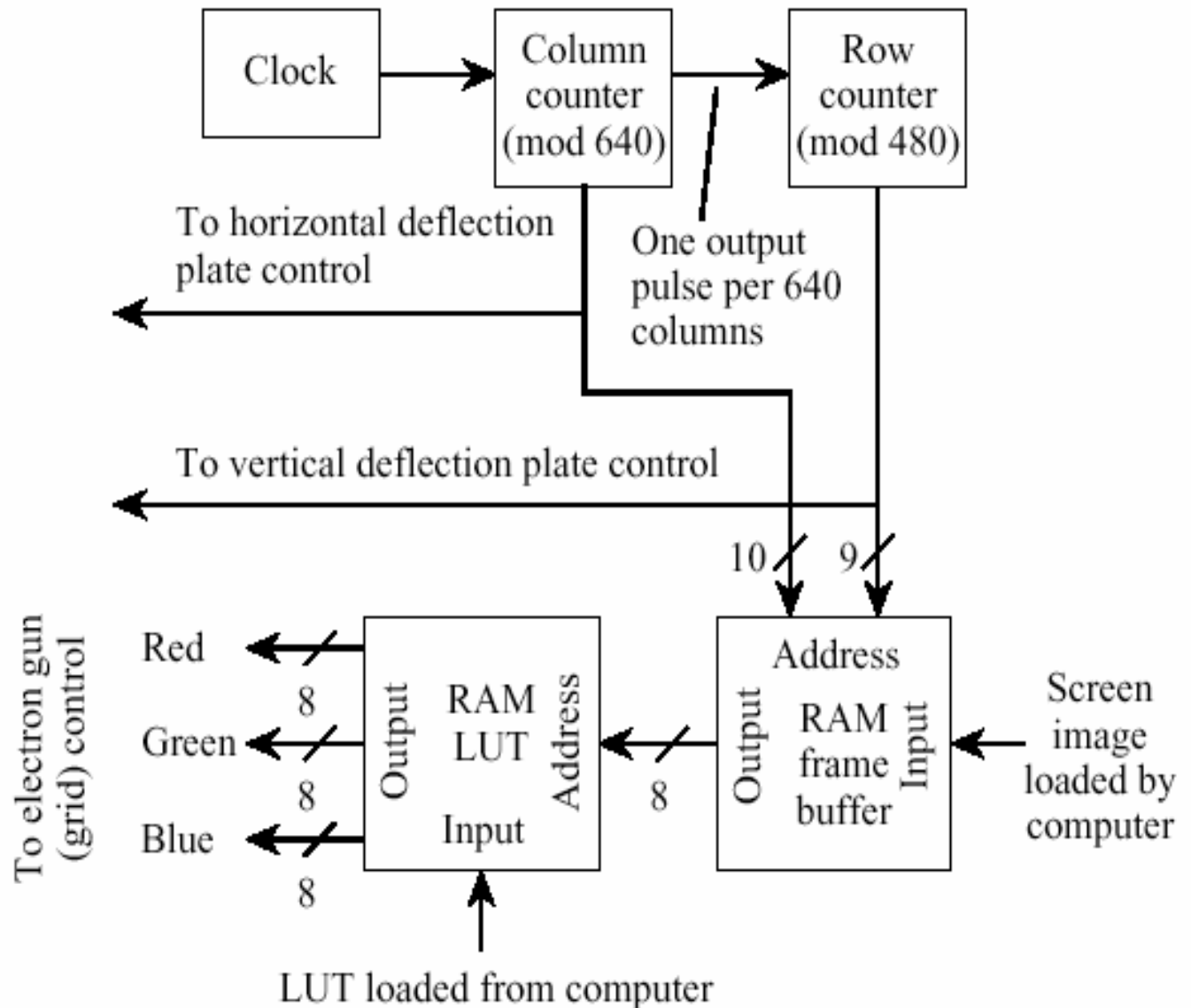
# Cathode Ray Tube

- A CRT with a single electron gun:



# Display Controller

- Display controller for a 640×480 color monitor



# Laser Printer

