Machine Architecture and Number Systems

Some material in this presentation is borrowed form Adrian Ilie
From The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Topics

- Major Computer Components
- Bits, Bytes, and Words
- The Decimal Number System
- The Binary Number System
- Converting from Binary to Decimal
- Converting from Decimal to Binary
- The Hexidecimal Number System

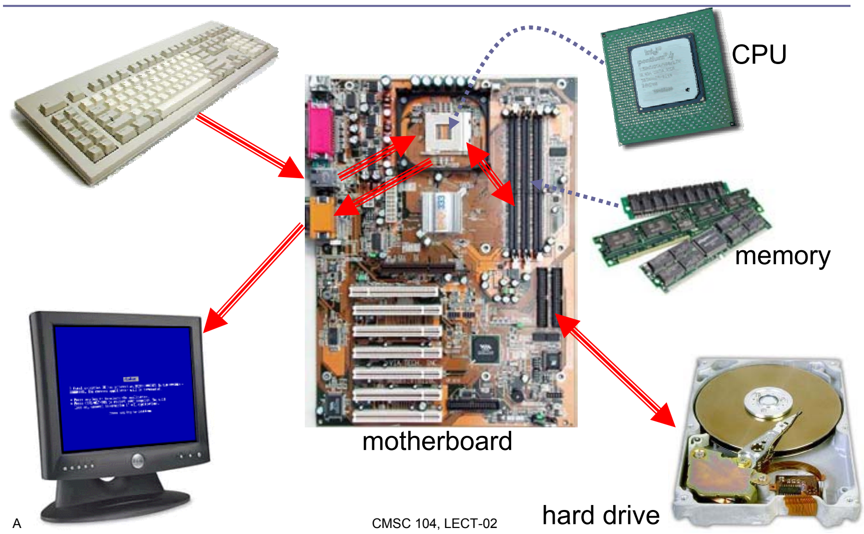
Reading

- Sections 1.1 - 1.3
- Appendix E (Sections E.1, E.4, E.5)

Major Computer Components

- Central Processing Unit (CPU)
- Bus
- Main Memory (RAM)
- Secondary Storage Media
- I / O Devices

Major Computer Components



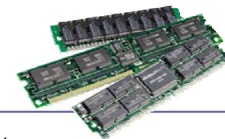
The CPU

- **Central Processing Unit (CPU)**
- The “brain” of the computer. This is the component that actually executes instructions
- Controls all other computer functions
- In **PCs** (**personal computers**) also called the **microprocessor** or simply **processor**.

The Bus

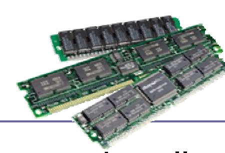
- Computer components are connected by a **bus.**
- A bus is a group of parallel wires that carry **control signals** and **data** between components.

Main Memory



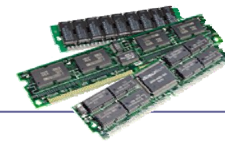
- Main memory holds information such as computer programs, numeric data, or documents created by a **word processor**.
- All programs must be brought into main memory before execution.
- When power is turned off, everything in main memory is lost
- Main memory is made up of **capacitors.**
- If a capacitor is charged, then its state is said to be **1,** or **ON**.
- We could also say the **bit is set**.
- If a capacitor does not have a charge, then its state is said to be **0,** or **OFF**.
- We could also say that **the bit is reset** or **cleared**

Main Memory (con't)



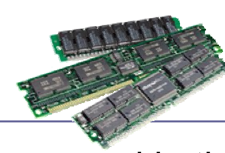
- Memory is divided into **cells**, where each cell contains 8 **bits** (a 1 or a 0). Eight bits is called a **byte**.
- Each of these cells is uniquely numbered.
- The number associated with a cell is known as its **address**.
- Main memory is **volatile** storage. That is, if power is lost, the information in main memory is lost.

Main Memory (con't)



- Other computer components can
  - get the information held at a particular address in memory, known as a **READ,**
  - or store information at a particular address in memory, known as a **WRITE**.
- Writing to a memory location alters its contents.
- Reading from a memory location does not alter its contents.

Main Memory (con't)



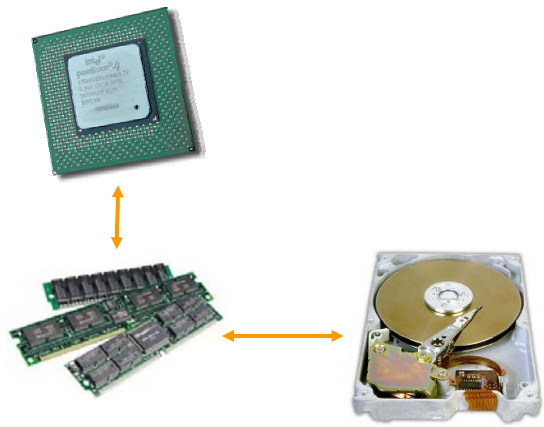
- All addresses in memory can be accessed in the same amount of time.
- We do not have to start at address 0 and read everything until we get to the address we really want (**sequential access**).
- We can go directly to the address we want and access the data (**direct** or **random access**).
- That is why we call main memory **RAM** (**Random Access Memory**).

CPU and Main Memory

Secondary Storage Media

- Provides permanent storage for information
- Retains information even when power is off
- Examples of secondary storage:
  - Hard Disks (sequential access)
  - Floppy Disks (sequential access)
  - Tapes (sequential access)
  - CD-ROMs (random access)
  - DVDs (random access)
- **Secondary storage media** store **files** that contain
  - computer programs
  - data
  - other types of information
- This type of storage is called **persistent (permanent) storage** because it is **non-volatile**.

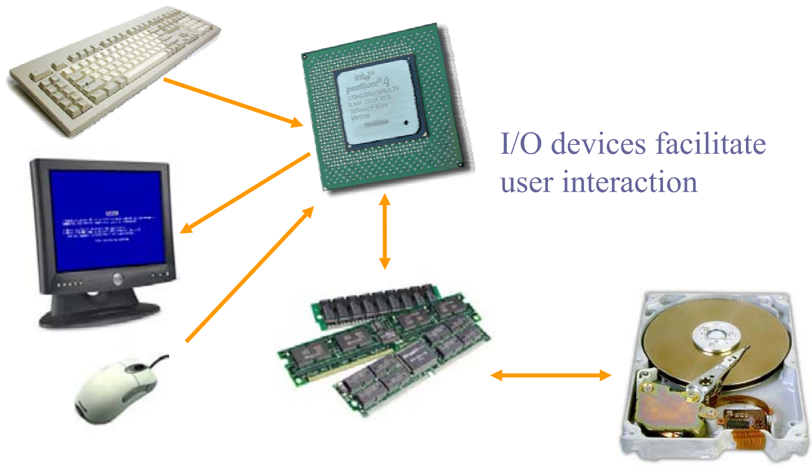
Secondary Storage



I/O (Input/Output) Devices

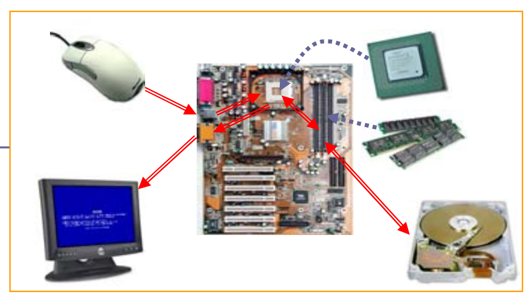
- Information input and output is handled by **I/O** (**input/output**) **devices**.
- More generally, these devices are known as **peripheral devices**.
- Examples:
  - Monitor
  - Keyboard
  - Mouse
  - Disk Drive (Floppy, Hard, Removable)
  - CD or DVD Drive
  - Printer
  - Scanner

Input/Output Devices



I/O devices facilitate user interaction

Opening MS Word



- Use the mouse to select MS Word
- The CPU requests the MS Word application
- MS Word is loaded from the hard drive to main memory
- The CPU reads instructions from main memory and executes them one at a time
- MS Word is displayed on your monitor

Bits, Bytes, and Words

- A **bit** is a single **binary digit** (a 1 or 0).
- A **byte** is 8 bits
- A **word** is 32 bits or 4 bytes
- **Long word** = 8 bytes = 64 bits
- **Quad word** = 16 bytes = 128 bits
- Programming languages use these standard number of bits when organizing data storage and access.

Bits, Bytes

Unit	Symbol	Number of Bytes
kilobyte	KB	2^{10} = 1024
megabyte	MB	2^{20} (over 1 million)
gigabyte	GB	2^{30} (over 1 billion)
terabyte	TB	2^{40} (over 1 trillion)

Bit Permutations

1 bit	2 bits	3 bits	4 bits	
0	00	000	0000	1000
1	01	001	0001	1001
	10	010	0010	1010
	11	011	0011	1011
		100	0100	1100
		101	0101	1101
		110	0110	1110
		111	0111	1111

Each additional bit doubles the number of possible permutations

Number Systems

- The on and off states of the capacitors in RAM can be thought of as the values 1 and 0, respectively.
- Therefore, thinking about how information is stored in RAM requires knowledge of the **binary (base 2) number system.**
- Let's review the **decimal (base 10) number system** first.

The Decimal Number System

- The decimal number system is a positional number system.
- Example:

5	6	2	1
10^3	10^2	10^1	10^0

$1 \times 10^0 = 1$

$2 \times 10^1 = 20$

$6 \times 10^2 = 600$

$5 \times 10^3 = 5000$

The Decimal Number System (con't)

- The decimal number system is also known as **base 10**.
- The values of the positions are calculated by taking 10 to some power.
- Why is the base 10 for decimal numbers?
  - Because we use 10 digits, the digits 0 through 9.

The Binary Number System

- The binary number system is also known as **base 2**. The values of the positions are calculated by taking 2 to some power.
- Why is the base 2 for binary numbers?
  - Because we use 2 digits, the digits 0 and 1.

The Binary Number System (con't)

- The binary number system is also a positional numbering system.
- Instead of using ten digits, 0 - 9, the binary system uses only two digits, 0 and 1.
- Example of a binary number and the values of the positions:

1	0	0	1	1	0	1
2^6	2^5	2^4	2^3	2^2	2^1	2^0

Converting from Binary to Decimal

1	0	0	1	1	0	1
2^6	2^5	2^4	2^3	2^2	2^1	2^0

$2^0 = 1$ $2^4 = 16$

$2^1 = 2$ $2^5 = 32$

$2^2 = 4$ $2^6 = 64$

$2^3 = 8$

$1 \times 2^0 = 1$

$0 \times 2^1 = 0$

$1 \times 2^2 = 4$

$1 \times 2^3 = 8$

$0 \times 2^4 = 0$

$0 \times 2^5 = 0$

$1 \times 2^6 = \underline{64}$

77_{10}

Converting from Binary to Decimal

Practice conversions:

Binary	Decimal
11101	
1010101	
100111	

Converting From Decimal to Binary

- Make a list of the binary place values up to the number being converted.
- Perform successive divisions by 2, placing the remainder of 0 or 1 in each of the positions from right to left.
- Continue until the quotient is zero.

Example: 42_{10}

2^5	2^4	2^3	2^2	2^1	2^0
32	16	8	4	2	1
<u>1</u>	<u>0</u>	<u>1</u>	<u>0</u>	<u>1</u>	<u>0</u>

Converting From Decimal to Binary

Practice conversions:

<u>Decimal</u>	<u>Binary</u>
59	
82	
175	

Working with Large Numbers

0 1 0 1 0 0 0 0 1 0 1 0 0 1 1 1 = ?

- Humans can't work well with binary numbers; there are too many digits to deal with.
- Memory addresses and other data can be quite large. Therefore, we sometimes use the **hexadecimal number system**.

The Hexadecimal Number System

- The hexadecimal number system is also known as **base 16**. The values of the positions are calculated by taking 16 to some power.
- Why is the base 16 for hexadecimal numbers ?
  - Because we use 16 symbols, the digits 0 through 9 and the letters A through F.

The Hexadecimal Number System (con't)

Binary	Decimal	Hexadecimal
0	0	0
1	1	1
10	2	2
11	3	3
100	4	4
101	5	5
110	6	6
111	7	7
1000	8	8
1001	9	9

Binary	Decimal	Hexadecimal
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

Hexidecimal Multiplication Table

×	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E	20
3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D	30
4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C	40
5	A	F	14	19	2E	23	28	2D	32	37	3C	41	46	4B	50
6	C	12	18	2E	24	2A	30	36	3C	42	48	4E	54	5A	60
7	E	15	1C	23	2A	31	38	3F	46	4D	54	4B	62	69	70
8	10	18	20	28	30	38	40	48	50	58	60	68	70	78	80
9	12	1B	24	2D	36	3F	48	51	5A	63	6C	75	7E	87	90
A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96	A0
B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5	B0
C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4	C0
D	1A	27	34	41	4E	4B	68	75	82	8F	9C	A9	B6	C3	D0
E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2	E0
F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1	F0
10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0	100

The Hexadecimal Number System

- Example of a hexadecimal number and the values of the positions:

<u>3</u> <u>C</u> <u>8</u> <u>B</u> <u>0</u> <u>5</u> <u>1</u>

16^6 16^5 16^4 16^3 16^2 16^1 16^0

Example of Equivalent Numbers

- Binary: 1 0 1 0 0 0 0 1 0 1 0 0 1 1 12
- Decimal: 2064710
- Hexadecimal: 50A716
- Notice how the number of digits gets smaller as the base increases.