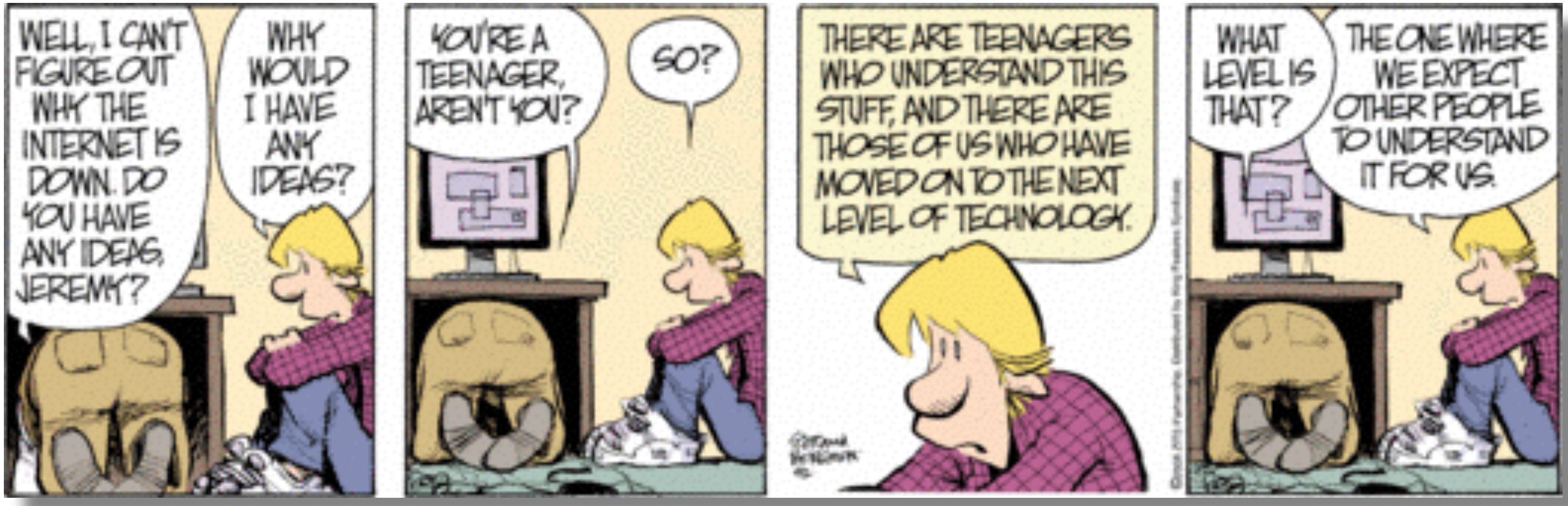
Machine Architecture and Number Systems

CMSC 104, Spring 2014

Christopher S. Marron

(thanks to John Park for slides)

Why Are We in this Course?



Remember:

"There are no stupid questions: just stupid people who don't know they should be asking something."

Machine Architecture and Number Systems

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 Hexadecimal Number System

Some material in this lecture borrowed from Olga Ratsimor

Some People Think A Computer is…

Some People Think A Computer is…

Major Computer Components

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

Schematic Diagram of a Computer

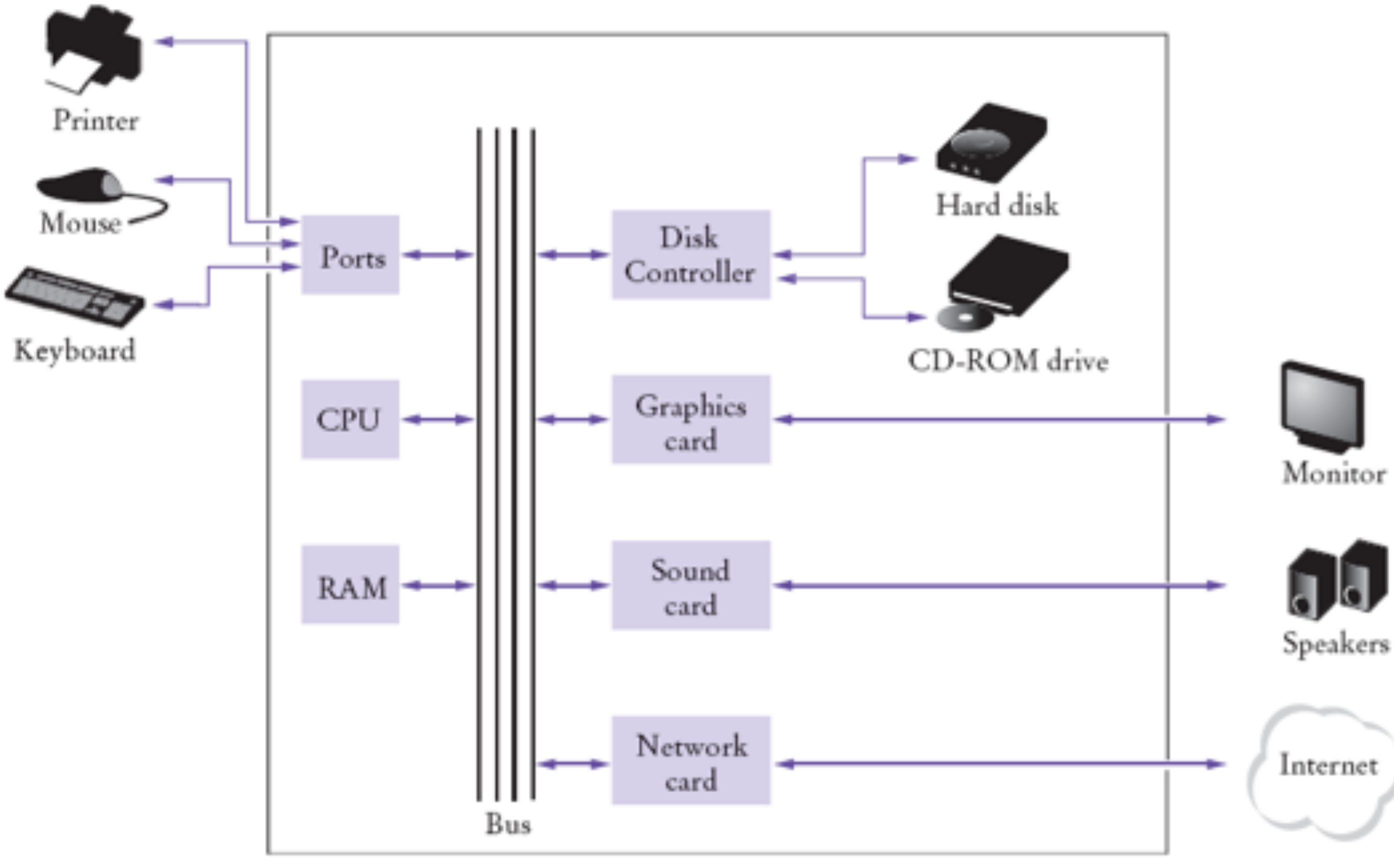
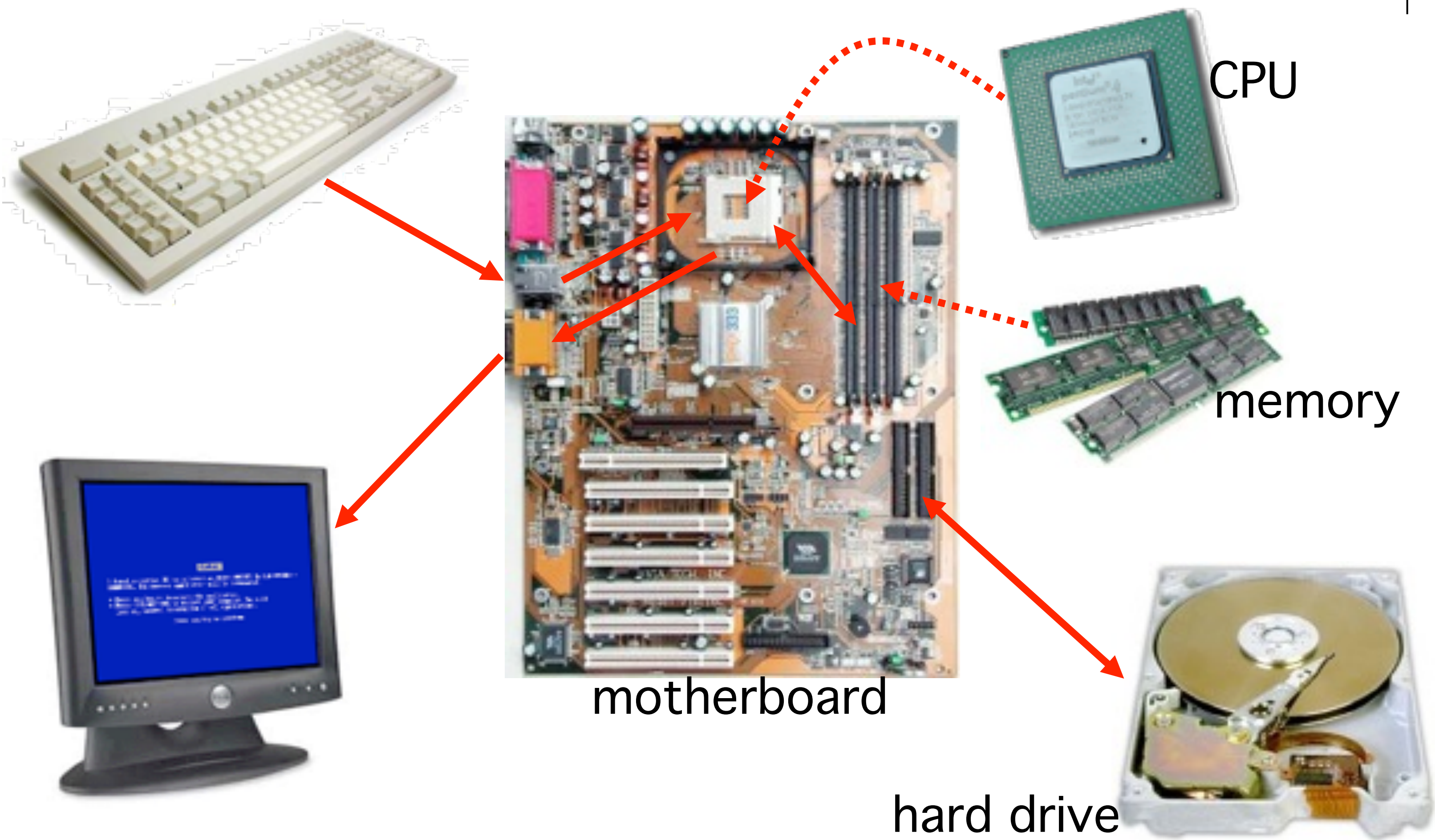


Figure 5 Schematic Diagram of a Computer

Diagram taken from Java Concepts, Fourth Edition

Realistic Diagram of Computer Components



The CPU

- Central Processing Unit
- The “brain” of the computer
- Controls all other computer functions
- In PCs 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**.

Main Memory (con’t)

- 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**).

Main Memory (con’t)

- “Stupid Question” #1:

  Why does adding more RAM make computers faster (sometimes)?
- Answer is much more complicated than you think: has to do with swapping/paging, multiprocessing

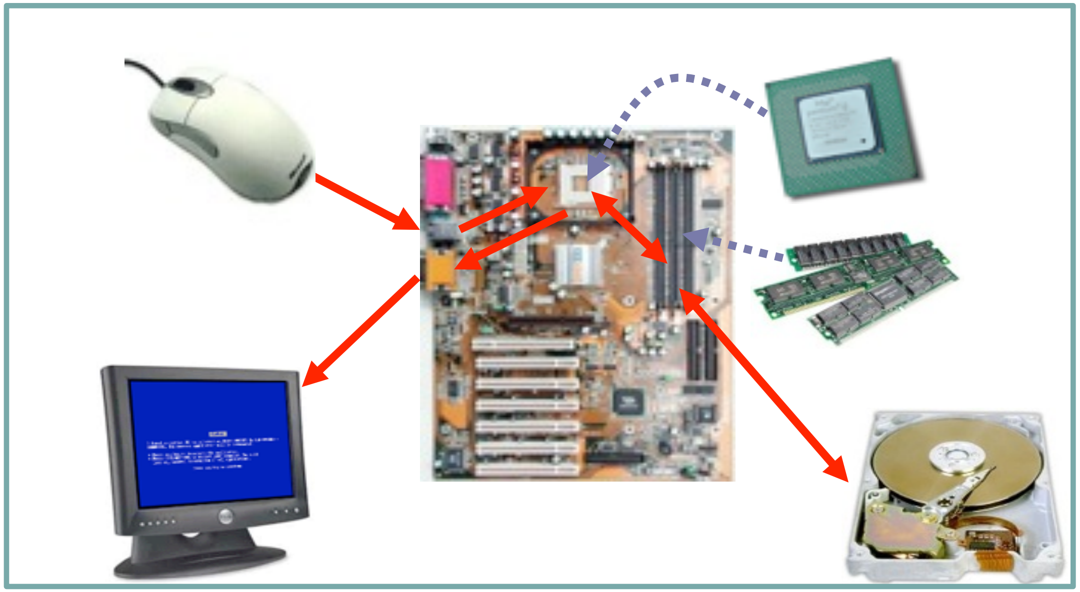
Secondary Storage Media

- Disks -- floppy, hard, removable (random access)
- Tapes (sequential access)
- CDs (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.

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

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

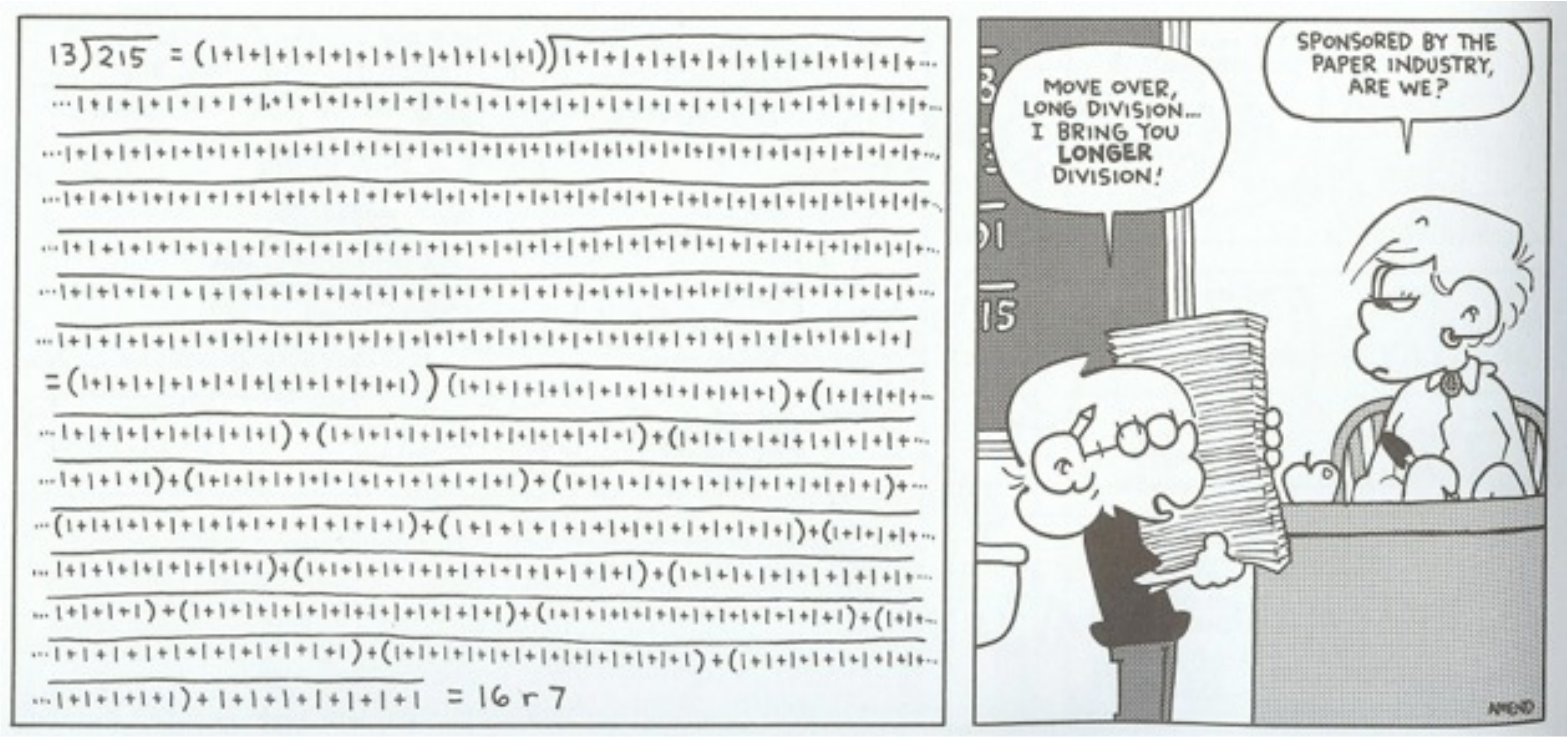
Bits, Bytes, and Words

- A bit is a single binary digit (a 1 or 0).
- A byte is 8 bits (usually… but not always!)
- 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.
- What do you call 4 bits? 2 bits?
  (hint: it is a small byte)

Number Systems

- The most elementary "number system" is unary:
  "I have *this* many things."
- An interesting problem:
  If you had 1 + 1 + 1 things, and you gave away 1 + 1 + 1 of them, how would you answer the question:
  "How many do you have left?"
- Unary counting is not a symbolic number system.

Unary Numbers are Not Practical



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:

$\underline{5}\ \underline{6}\ \underline{2}\ \underline{1}$

$10^3\ 10^2\ 10^1\ 10^0$

$1 \times 10^0 = 1$

$2 \times 10^1 = 20$

$6 \times 10^2 = 600$

$5 \times 10^3 = 5000$

The Decimal Number System

- 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 decimal number system, and other number systems, are *symbolic representations* of concrete quantities

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

- 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

$\underline{1}$	$\underline{0}$	$\underline{0}$	$\underline{1}$	$\underline{1}$	$\underline{0}$	$\boxed{\underline{1}}$
2^6	2^5	2^4	2^3	2^2	2^1	$\boxed{2^0}$

$1 \text{ X } 2^0 = 1$

$0 \text{ X } 2^1 = 0$

$1 \text{ X } 2^2 = 4$

$1 \text{ X } 2^3 = 8$

$0 \text{ X } 2^4 = 0$

$0 \text{ X } 2^5 = 0$

$1 \text{ X } 2^6 = \underline{64}$

77_{10}

$2^0 = 1$	$2^4 = 16$
$2^1 = 2$	$2^5 = 32$
$2^2 = 4$	$2^6 = 64$
$2^3 = 8$	$2^7 = 128$

Converting from Binary to Decimal

Practice conversions:

Binary	Decimal
11101	
1010101	
100111	

Geek Joke #1

- Seen on a random T-shirt:

  There are 10 kinds of people in the world:
  Those who understand binary
  …and those who don't

Converting from Decimal to Binary

- Make a list of the binary place values up to the number being converted. (In the example below, 2^5 is the largest possible leftmost position)
- Perform successive divisions by 2, placing the remainder of 0 or 1 in each of the positions from <u>right to left</u>.
- Continue until the quotient is zero.
- Example: 42_{10}

too large!

~~64~~	32	16	8	4	2	1
~~_~~	<u>1</u>	<u>0</u>	<u>1</u>	<u>0</u>	<u>1</u>	<u>0</u>
$\cancel{2^6}$	2^5	2^4	2^3	2^2	2^1	2^0

Converting from Decimal to Binary

Practice conversions:

Decimal	Binary
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 and octal number systems**.

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 it 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

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

The Hexadecimal Number System

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

$$\underline{3}\quad \underline{C}\quad \underline{8}\quad \underline{B}\quad \underline{0}\quad \underline{5}\quad \underline{1}$$

$$16^6\quad 16^5\quad 16^4\quad 16^3\quad 16^2\quad 16^1\quad 16^0$$

The Octal Number System

- Example of an octal number and the values of the positions:

1	3	0	0	2	4
8^5	8^4	8^3	8^2	8^1	8^0

- Binary equivalent:
  1 011 000 000 010 100 =
  1011000000010100

Example of Equivalent Numbers

Binary: 1 1 0 1 0 0 0 0 1 0 1 0 0 1 1 1_2

Octal: 150247_8

Decimal: 53415_{10}

Hexadecimal: $D0A7_{16}$

Notice how the number of digits gets smaller as the base increases.

But Why Use Hex or Octal?

- Simple: can divide binary numbers into equal-sized sets of bits, then convert directly
- This is *not* true of decimal-to-{binary,hex,octal}