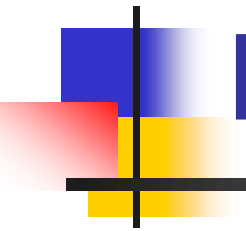
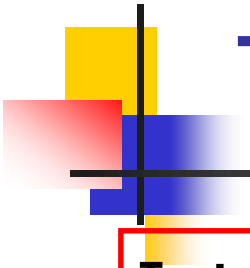


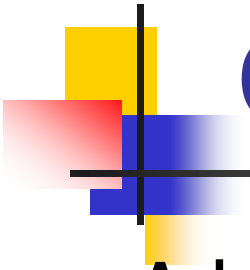
Microprocessor Systems

Dr. Gökhan İnce



Topics

- Introduction: Computer History
- Microprocessor based Systems
- Number Systems



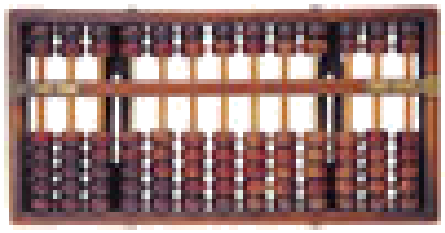
Computer History

- Advances in computer technology
- Pre-electronics era
  - Gears and mechanical systems
- Electronics era (1945-1975)
  - Vacuum tubes, transistors
- Microprocessor age (1976-)
  - Integrated circuits and single-chip microprocessors

Computer History

- The first computers were people (predominantly women). "Computer" was originally a job title to perform the repetitive calculations required to compute such things as **navigational tables, tide charts, and planetary positions** for astronomical almanacs.
- The abacus, also called a counting frame, is a calculating tool used primarily in parts of Asia as an aid for mathematical computations

Ref: http://www.computersciencelab.com/ComputerHistory/History.htm



Computer History – Pre Electronics

- In 1642, first Calculator: Blaise Pascal invented the mechanical calculator.

- In 1673 Gottfried Leibniz developed a machine to perform multiplication, division and square roots.

- In 1804, Joseph-Marie Jacquard designed a loom capable of following instructions on a punched paper tape.

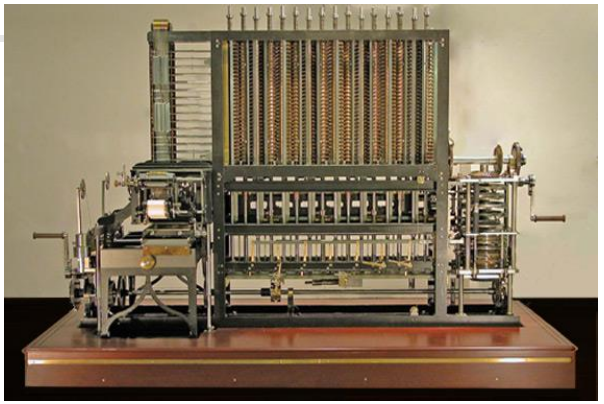
Computer History – Pre Electronics

Charles Babbage – Difference Engine

- 1822 the English mathematician ***Charles Babbage*** proposed a steam driven calculating machine the ***Difference Engine***.
- This machine would be able to compute tables of numbers (polynomials), such as logarithm tables.
- He obtained government funding for this project due to the importance of numeric tables in ocean navigation.
- In that time frame the British government was publishing a seven volume set of navigation tables which came with a companion volume of corrections.
- It was hoped that Babbage's machine could eliminate errors in these types of tables.
- Construction of Babbage's Difference Engine proved exceedingly difficult and the project soon became the most expensive government funded project up to that point in English history.

Computer History – Pre Electronics

Charles Babbage – Analytic Engine



- This large device, powered by 6 steam engines, would be more general purpose in nature because it would be **programmable**, thanks to the punched card technology of Jacquard.
- Babbage saw that the pattern of holes on a punch card could be used to represent an abstract idea such as a problem statement or the raw data required for that problem's solution. Babbage realized that **punched paper** could be employed as a **storage mechanism**, holding computed numbers for future reference.
- Because of the connection to the Jacquard loom, Babbage called the two main parts of his Analytic Engine the "Store" and the "Mill", as both terms are used in the weaving industry. The Store was where numbers were held and the Mill was where they were "woven" into new results. In a modern computer these same parts are called the ***memory unit*** and the ***central processing unit*** (CPU).
- The Analytic Engine also had a key function that distinguishes computers from calculators: **the conditional statement**. Based on the conditional statement, the path of the program (that is, what statements are executed next) can be determined based upon a condition or situation that is detected at the very moment the program is running.
- Ref: http://www.computersciencelab.com/ComputerHistory/History.htm

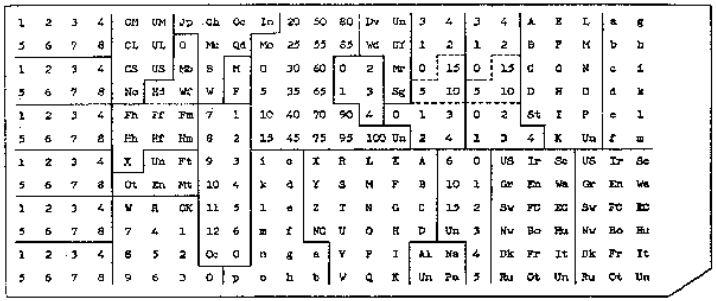
Computer History – Pre Electronics

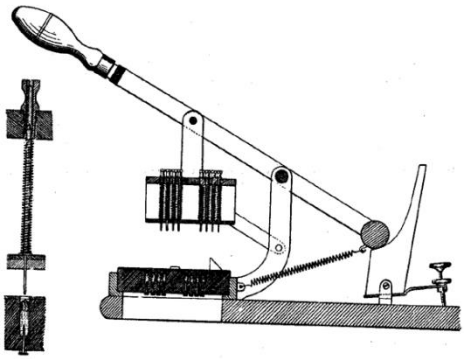
Ada Byron, the first programmer

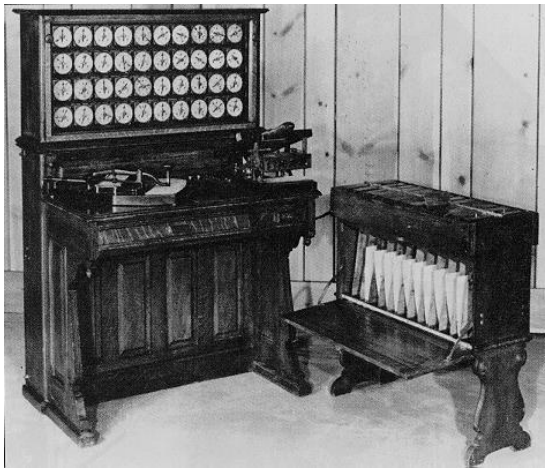
- Babbage befriended ***Ada Byron***, the daughter of the famous poet Lord Byron
- She was only 19, she was fascinated by Babbage's ideas and thru letters and meetings with Babbage she learned enough about the design of the Analytic Engine to begin fashioning programs for the still unbuilt machine.
- While Babbage refused to publish his knowledge for another 30 years, Ada wrote a series of "Notes" wherein she detailed sequences of instructions she had prepared for the Analytic Engine.
- The Analytic Engine remained unbuilt (the British government refused to get involved with this one) but Ada earned her spot in history as the first computer programmer.
- **Ada invented the subroutine and was the first to recognize the importance of looping.**

Computer History – Pre Electronics

- In 1890 Herman Hollerith built machines under contract for the Census Office, which used them to tabulate the 1890 **census** in only one year.
- ->1924:IBM



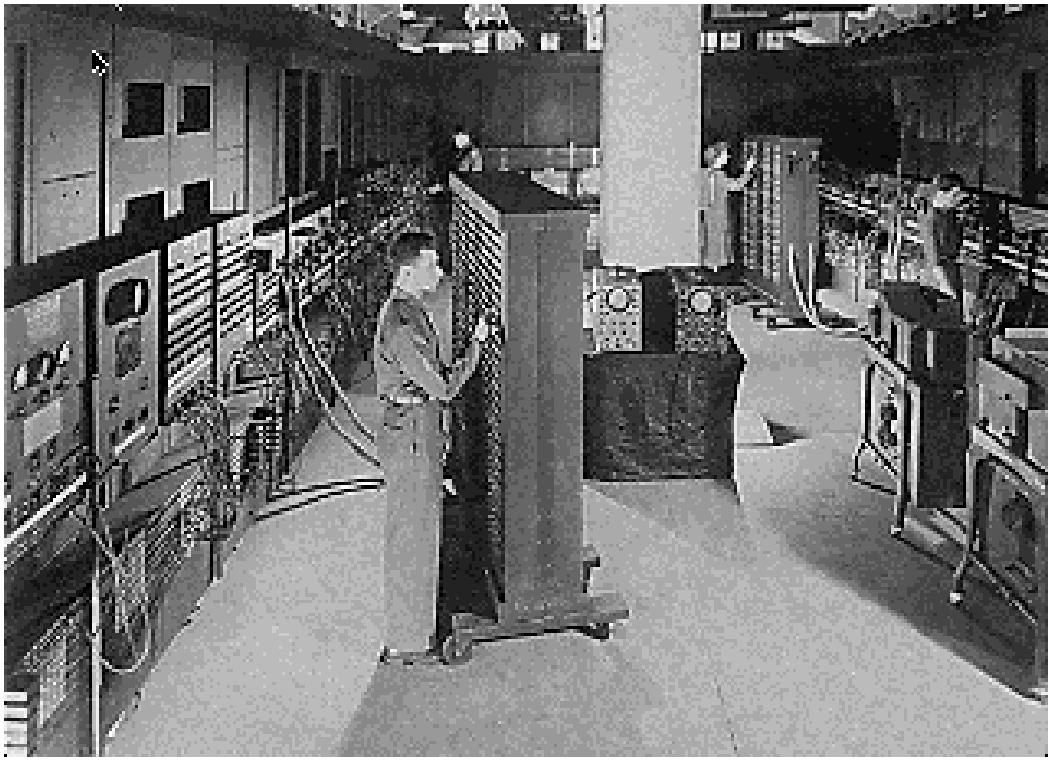




- In 1935, at Cambridge University, **Turing** conceived the principle of the modern computer. He described an abstract digital computing machine consisting of a **limitless memory** and a **scanner** that moves back and forth through the memory. The actions of the scanner are dictated by a program of instructions that is stored in the memory in the form of symbols.

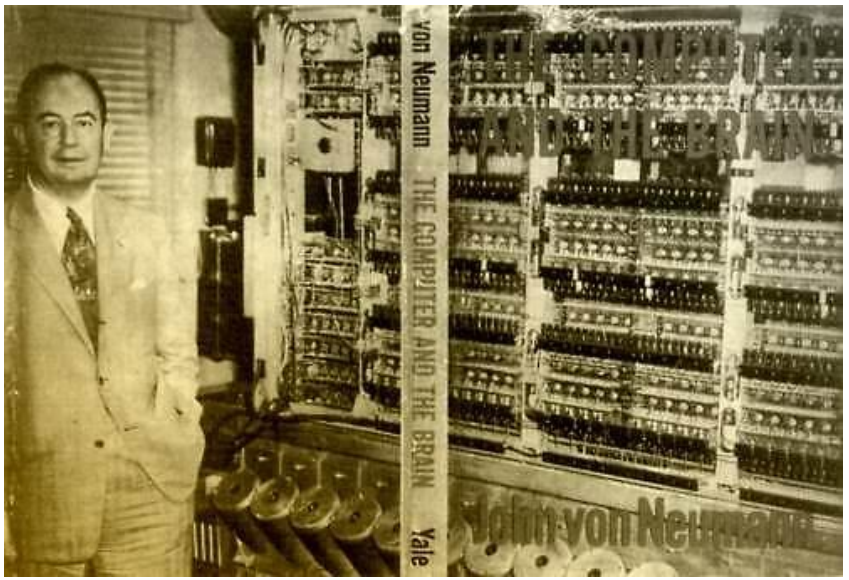
Electronics Age

- The Electronic Numerical Integrator and Computer (ENIAC), was the first general-purpose electronic computer (1945).
- 100kW power consumption
- 30x10x3m dimensions, 18.000 tubes, 30 tonnes
- Used decimal system,
- no internal memory
- One operation in 200 μs. -> 5000 cycles per second -> 5kHz



Electronics Age

- In 1945, in his first draft of a report on the EDVAC, von Neumann proposed the stored program concept.
  - instructions and data are both stored in the same medium.
  - should be able to hold any programme in memory
  - same hardware can be loaded with different software to make a computer perform different types of jobs
    - a multipurpose computer
  - EDVAC was built in 1952.

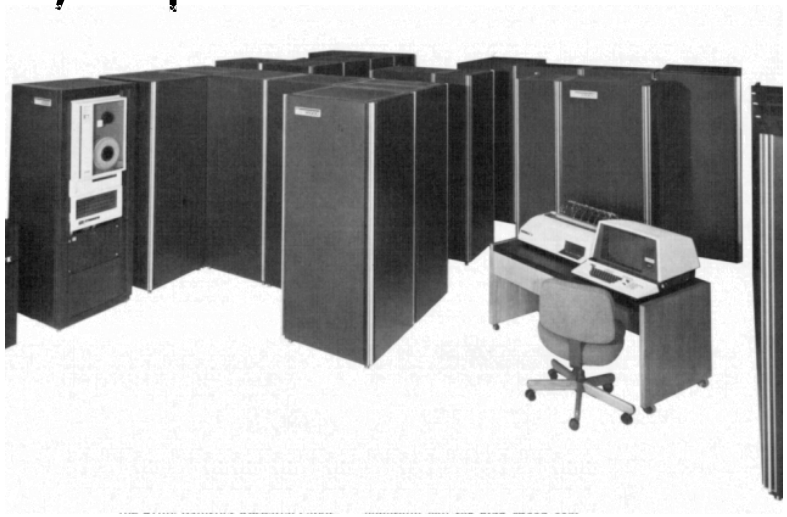


Computer History

- These computers would be miniaturized in the future!
* I hope they shrink the ELEPHANTS, too.

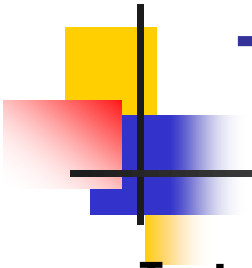
Electronics Age

- First generation computers (1945-1955): **vacuum tubes for circuitry and magnetic drums for memory**, and were often enormous, taking up entire rooms
- Second generation computers (1956-1965): **Transistors**, invented in 1948, replaced vacuum tubes; **assembly language**
- Third generation computers (1966-1975): Transistors were miniaturized and placed on **silicon chips**, called semiconductors.



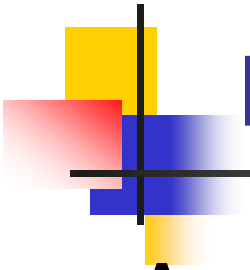
The Age of Microprocessors

- Fourth generation computers (1971-): Thousands of integrated circuits were built onto a single silicon chip. (Intel 4004)
- Fifth generation computers based on parallel processing hardware and artificial intelligence, are still in development.



Topics

- Introduction: Computer History
- Microprocessor based Systems
- Number Systems

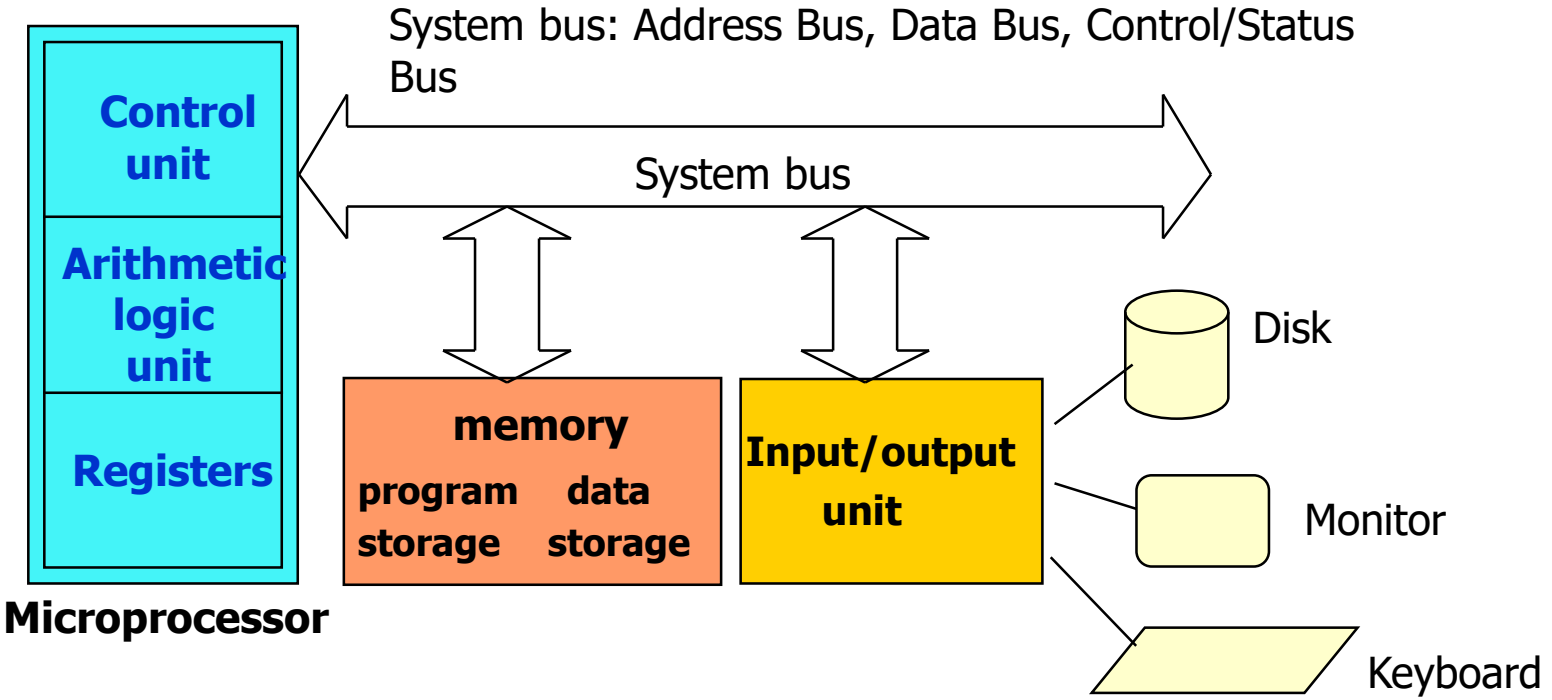


Microprocessor based Systems

- A computer is a programmable machine designed to automatically carry out a sequence of arithmetic or logical operations.
  - **Embedded System** is a special-purpose computer system designed to perform one or a few dedicated functions
  - **General purpose computer systems** provide programmability to users to carry out a finite set of arithmetic or logical operations.

Computer Organization

- The block diagram of a micro-computer system is shown below. In this course we will discuss about such systems.



What is a microprocessor?

- The microprocessor is a programmable device that takes in numbers, performs on them arithmetic or logical operations according to the program stored in memory and then produces other numbers as a result

- to be explained in detail in the next slides...

What is a microprocessor?

- Programmable device: The microprocessor can perform different sets of operations on the received data depending on the sequence of instructions supplied in the given program. By changing the program, the microprocessor manipulates the data in different ways.

- Instructions: Each microprocessor is designed to work with its own instruction set. This instruction set defines what the microprocessor can and cannot do.

What is a microprocessor?

- Takes in: Input devices (such as keyboard, mouse...) feed data into computer
- Numbers: Microprocessors operate on numbers in the form of binary. The number of bits it can run simultaneously defines its word length. Microprocessor input and output are digital numbers
- Arithmetic and Logic Operations: Every microprocessor has arithmetic operations such as add and subtract and logic operations such as AND, OR, XOR, shift left, shift right, etc.

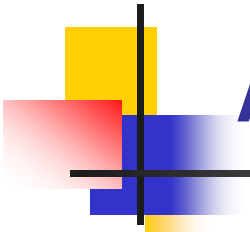
What is a microprocessor?

- Program: A program is a sequence of instructions to operate the computer
  - Machine language: Low level instructions in binary form specific for the microprocessor. In this language, every instruction is described by binary patterns.
    - ex. 11001101 may mean 1 + 2
  - Assembly Language: Symbolic language to represent low level machine code
  - Languages such as C, Fortran, Pascal etc… are high level languages, which are compiled (translated into the machine language)

Machine Language

- The number of bits that form the **word** of a microprocessor is fixed for that particular processor.
  - These bits define a maximum number of combinations.
  - For example an 8-bit microprocessor can have at most 2^8 = 256 different combinations.

- However, in most microprocessors, not all of these combinations are used.
  - Certain patterns are chosen and assigned specific meanings.
  - Each of these patterns forms an instruction for the microprocessor.
  - The complete set of patterns makes up the microprocessor's machine language.



Assembly Language

- Entering the instructions using **hexadecimal** is quite easier than entering the binary combinations.
  - However, it still is **difficult** to understand what a program written in hexadecimal does.
  - So, each company defines a symbolic code for the instructions.
  - These codes are called **mnemonics**.
  - The mnemonic for each instruction is usually a group of letters that suggest the operation performed.

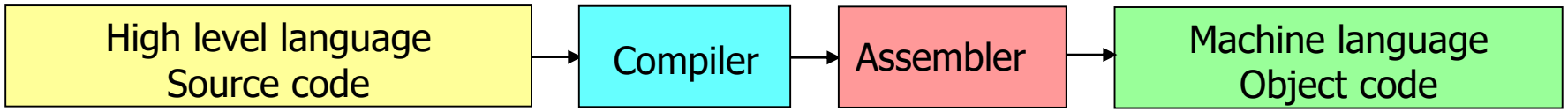
Assembly Language

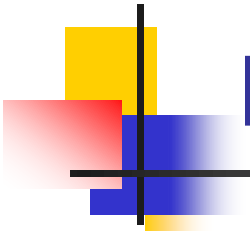
- It is important to remember that a **machine language** and its associated **assembly language** are completely **machine dependent**.
  - In other words, they are not transferable from one microprocessor to a different one.

- For example, Motorola has an 8-bit microprocessor called the 6800.
  - The Intel 8085 machine language is very different from that of the 6800. So is its assembly language.
  - A program written for the 8085 cannot be executed on the 6800 and vice versa.

High Level Languages

- We said earlier that assembly and machine language are completely dependent on the microprocessor. They can not be easily moved from one to the other.
- To allow **programs** to be developed **for multiple machines** high level languages were developed.
  - These languages describe the operation of the program in general terms.
  - These programs are translated into microprocessor specific assembly language using a **compiler** or **interpreter** program.
  - These programs take as an input **high level statements** such as " i= j + k;" and translate them to **machine language** compatible with the microprocessor being used.





Microprocessor vs. Microcontroller

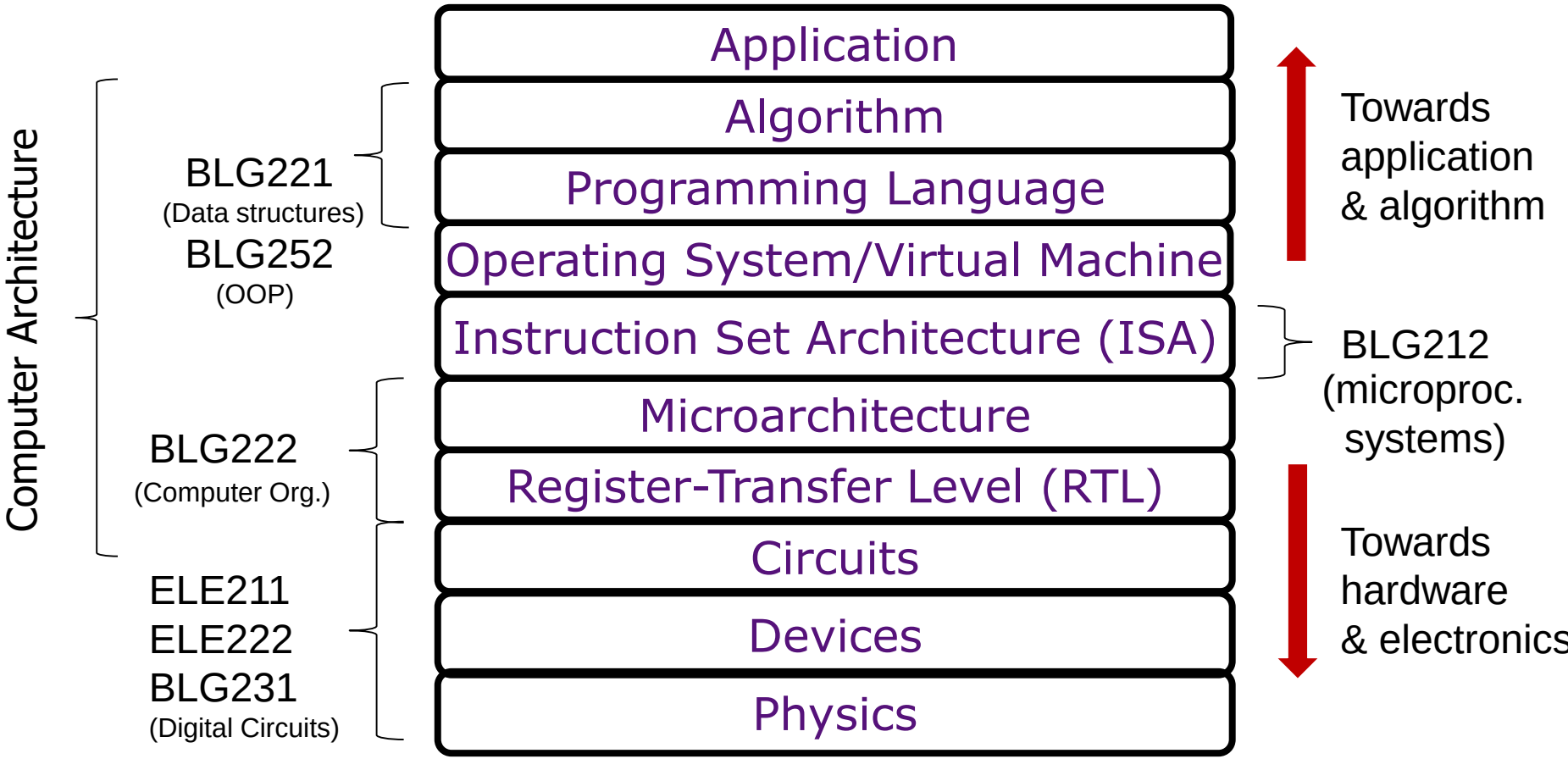
- The processor (Central Processing Unit - CPU)
  - Registers -- storage locations in the processor
  - Arithmetic logic unit
  - Control unit
- The microprocessor
  - A processor implemented on a Very Large Scale Integration (VLSI) chip
  - Peripheral chips are needed to construct a product
- The microcontroller
  - The processor and peripheral functions implemented on one VLSI chip

Microcomputer vs. Microcontroller

- The differences between MicroController (MCtrl) and MicroComputer (MCmp) are
  - MCtrl has timers and counters inbuilt but MCmp does not have inbuilt timers and counters.
  - MCtrl is used for specific task purpose. MCmp is used for general purpose.
  - MCtrl is having less flexibility in design. MCmp is having flexibility in design.

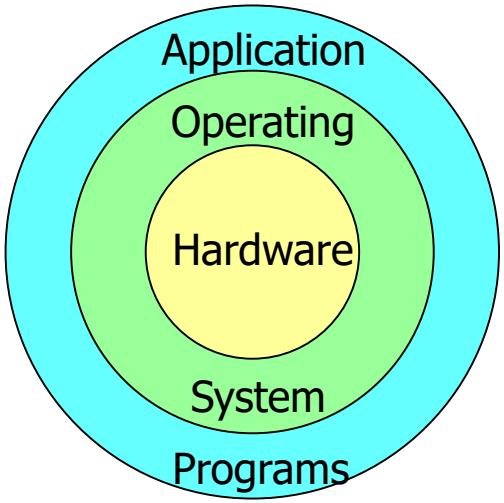
Operation of a microcomputer

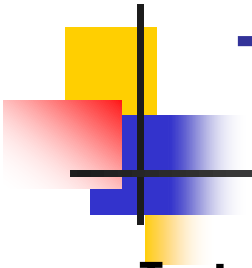
LAYERS OF ABSTRACTION



The Hardware/Software interaction

- The **hardware** of a computer system is **the collection of chips** that make up its different pieces, including the microprocessor.
- **Software** refers to any **program** that is executed on the hardware.
- The interaction between the two systems (hardware and software) is managed by a group of programs known collectively as Operating system.





Topics

- Introduction: Computer History
- Micriprocessor based Systems
- Number Systems

Number Systems

- Decimal (DEC)

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Human has 10 fingers, Base 10 (Decimal seems intuitive)

- Digital systems have two states 0 and 1 (ON / OFF)

Binary system 0, 1 (electronic computers use binary system)

Word length in computers can be 8, 16, 32, 64 bits etc…

It is hard to define such long strings in binary format like:

0010101101101100 b = 11116

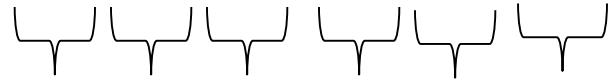
$ 2 B 6 C

Group them into four bits and define the Hexadecimal (HEX) system

Number Systems

Group them into three bits and define the Octal (OCT) system

000 010 101 101 101 100 b = 11116 = $2B6C



octal 0 2 5 5 5 4

Binary Coded Decimal (BCD) system

Example: $13_{10} = 1101_2 = 0001\ \ 0011$ BCD

Number Systems

Decimal	Binary (% or b)	Octal	Hexadecimal ($ or h)	Binary Coded Decimal (BCD)
0	0000 0000	000	00	0000
1	0000 0001	001	01	0001
2	0000 0010	002	02	0010
3	0000 0011	003	03	0011
4	0000 0100	004	04	0100
5	0000 0101	005	05	0101
6	0000 0110	006	06	0110
7	0000 0111	007	07	0111
8	0000 1000	010	08	1000
9	0000 1001	011	09	1001
10	0000 1010	012	0A	0001 0000
11	0000 1011	013	0B	0001 0001
12	0000 1100	014	0C	0001 0010
13	0000 1101	015	0D	0001 0011
14	0000 1110	016	0E	0001 0100
15	0000 1111	017	0F	0001 0101
16	0001 0000	020	10	0001 0110
20	0001 0100	024	14	0010 0000
160	1010 0000	240	A0	0001 0110 0000
248	1111 1000	370	F8	0010 0100 1000

Number Conversions

Decimal $953,78 = 9*10^2+5*10^1+3*10^0+7*10^{-1}+8*10^{-2}$
$= 900+50+3+0,7+0,08 = 953,78$

BIN to DEC $\%1011 = 1*2^3+0*2^2+1*2^1+1*2^0$
$= 8+0+2+1 = 11$

Hex to DEC $\$A2F = 10*16^2+2*16^1+15*16^0$
$= 2560+32+15 = 2607$

BIN to OCT $10101001_2 = (010\ \ 101\ \ 001)_2 = 251_8$

BIN to HEX $10101001_2 = (1010\ \ 1001)_2 = A9h$

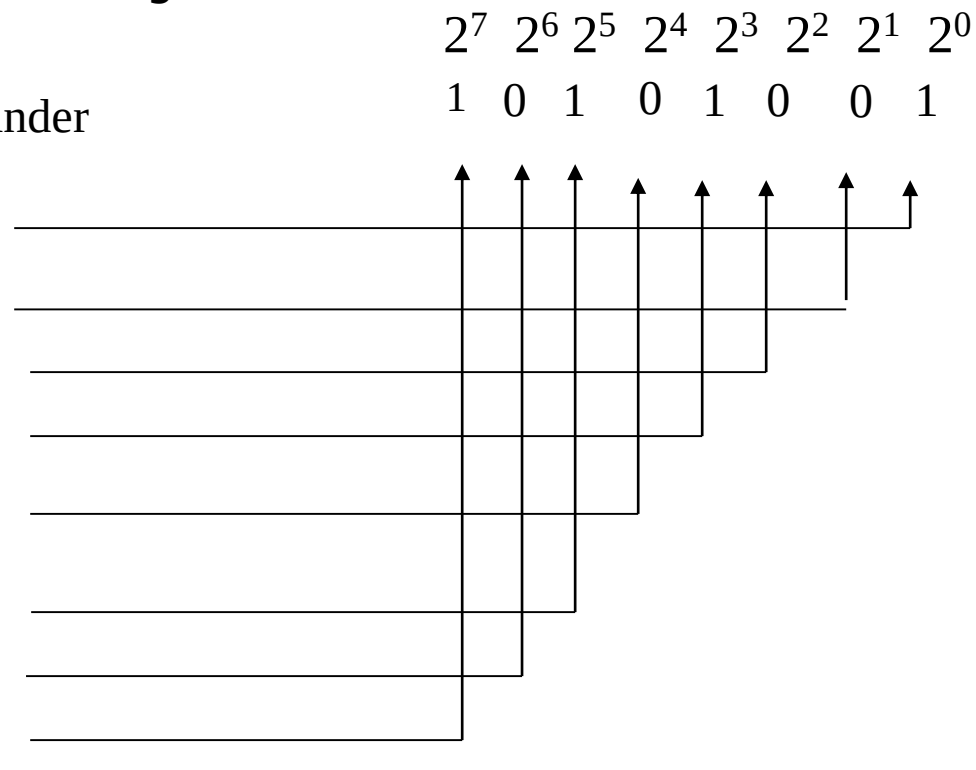
DEC to BCD $13_{10} = 1101_2 = 0001\ \ 0011$

Number Conversions

DEC to BIN

Method 1: Divide to 2's until reaching 1

DECIMAL		Remainder
169	169/2=84	1
84	84/2=42	0
42	42/2=21	0
21	21/2=10	1
10	10/2= 5	0
5	5/2=2	1
2	2/2=1	0
1	1/2=0	1



Number Conversions

DEC to BIN

Method 2: Look for the highest 2^n

DECIMAL	Largest 2^n
169	169-128=41
41	41- 32=9
9	9- 8=1
1	1- 1=0

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	0	1	0	0	1

Representing Data (Number) in Computers

Computers store numbers in binary form

One Byte = 8 bits is the unit for defining bit lengths

Assume 1 Byte of storage:

It can store unsigned numbers from

%0000 0000 to %1111 1111 or
$00 to $FF or
0 to 255

Representing Data (Number) in Computers

1 Byte of storage:

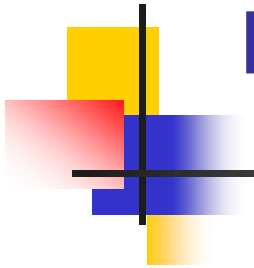
It can store signed numbers from

%0111 1111 to %1111 1111
Most Significant Bit, MSB is the sign bit,
0 is positive, 1 is negative

127 to -127

%1111 1111 -127
%1000 0001 -1
%0000 0000 0
%0000 0001 1
%0111 1111 127

Representing Data (Number) in Computers



2's complement is an efficient method to represent signed numbers. For negative numbers, complement the number (replace 0s with 1s, 1s with 0s, then add 1)

1 Byte of storage can store signed numbers from

%0111 1111 to %1000 0000
Most Significant Bit, MSB is the sign bit,
0 is positive, 1 is negative

Representing Data (Number) in Computers

Example: +7 0000 0111

Signed numbers : -7 -> 1000 0111

Complement numbers :

1's complement : 1111 1000

2's complement : 1111 1001 -> -7

Taking the 2s complement of the 2s complement restores the number

Subtraction in Unsigned numbers

Adding 2s complement to another unsigned number is subtraction of unsigned numbers

5-2=3

```
5:    0000 0101                      0000 0101
2:    0000 0010   2's complement   +  1111 1110
3:                                  1 0000 0011
```

End carry: discard it, the result is positive

2-5=-3

```
2:    0000 0010                      0000 0010
5:    0000 0101 2's complement     +  1111 1011
-3:                                  1111 1101
```

No end carry: result is negative-> take 2's complement to find the magnitude

Addition and subtraction in signed numbers

Add two numbers including their sign bits, then discard the carry bit. Negative numbers are represented in 2s complement.

```
+6 :     0000 0110
+13: + 0000 1101
+19:    0001 0011
```

```
-6 :     1111  1010
+13: + 0000 1101
+7:      0000 0111
```

```
+6 :     0000  0110
-13: +   1111  0011
-7  :    1111  1001
```

```
-6 :     1111  1010
-13: +   1111  0011
-19:     1110  1101
```

Overflow condition and detection

Overflow occurs if the resulting number exceeds the length of the register word. (For example, addition result of two 8-bit numbers do not fit into 8-bits)

While adding two unsigned numbers: Overflow is detected from the end carry-out bit of the MSB

While adding two signed numbers: When two signed numbers are added, the sign bit is treated as part of the number. End carry does not indicate overflow. **Overflow cannot occur if one number is positive and the other is negative**. Overflow occurs if the sign bit changes for the addition of two positive, or two negative number.

```
carries:     0 -> 1
  70:    0 100 0110
  80: + 0 101 0000
 150:    1 001 0110
```

```
carries:     1 -> 0
 -70:    1 011 1010
-80: +   1 011 0000
-150:    0 110 1010
```

Overflow cases

Computer Arithmetics

- **Carry**: In the case of unsigned addition carry occurs when the correct result is too large to be represented.

- **Borrow**: In the case of unsigned subtraction borrow occurs when the value being subtracted is larger than the value it is subtracted from. If subtraction is performed by negating and adding , there is a borrow if the addition does not produce a carry.

- **Overflow**: In the case of signed addition and subtraction, overflow occurs when the correct result is either more positive than the largest possible positive value ore more negative than the smallest possible negative one. It is indicated by a result having incorrect sign.

  There is overflow if:

  pos + pos $\rightarrow$ neg pos – neg $\rightarrow$ neg

  neg + neg $\rightarrow$ pos neg – pos $\rightarrow$ pos

Floating Point number representation

Mantissa and Exponent defines a very large scale of scientific numbers

Example: 6132.789 DEC can be represented as +0.6132789*10^+04

Similarly, %1001.11 -> $(0.1001110)_2 * 2^{000100}$

Mantissa: 01001110, Radix: 2, Exponent: 000100

The Radix and Radix point position of the mantissa is known

The mantissa and exponent can be signed numbers resulting in a very large positive and negative range and resolution

Arithmetic operations with floating point numbers require more complicated hardware, but it is necessary for scientific computation

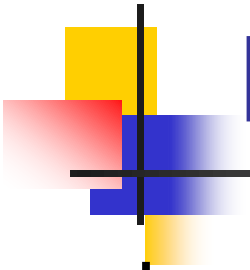
Capable microprocessors have floating point operation feature

Other representation systems in computers

ASCII

a method to represent alphanumeric characters

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□



References

- Lecture Slides: Dr. Şule Gündüz Öğüdücü
- Lecture Slides: Dr. Erdem Matoğlu
- Lecture Slides: Dr. Feza Buzluca
- Lecture Slides: Dr. Bassel Soudan
- Lecture Slides: Dr. Gökhan İnce