Islamic University Of Gaza
Faculty of Engineering
Computer Department

Assembly Language
Discussion
Chapter 1

Created By: Eng. Ahmed M. Ayash **Date: 15/02/2015**

Modified and Presented by: ***Eng. Eihab S. El-Radie***

CHAPTER 1

Basic Concepts

- **Machine code** or **machine language** is a system of impartible instructions executed directly by a computer's central processing unit (CPU).
    - Instructions consist of binary code: 1s and 0s

- An **assembly language:** is a low-level programming language that is very similar to machine language, but uses symbols instead of binary numbers.
    - Slightly higher-level language

- An **assembler** is a program that converts **source-code** programs written in **assembly language** into **object files** in machine language.

  Assembler Examples:
    - MASM (Macro Assembler from Microsoft),
    - TASM (Turbo Assembler from Borland)
    - NASM (Netwide Assembler for both Windows and Linux),

  We will use MASM6.15 (Macro Assembler from Microsoft)
    - Masm.exe creates an .obj file from an .asm file.

- A **linker** is a program that combines individual files created by an assembler into a single executable program.

    - Two linkers: **LINK.EXE** and **LINK32.EXE** are provided with the MASM 6.15 distribution to link **16-bit real-address mode** and **32-bit protected-address mode** programs respectively.

16-Bit Real-Address Mode: 16-bit real-address mode programs run under MS-DOS and in the console window under MS-Windows. Also known as real mode programs, they use a segmented memory model required of programs written for the Intel 8086 and 8088 processors.

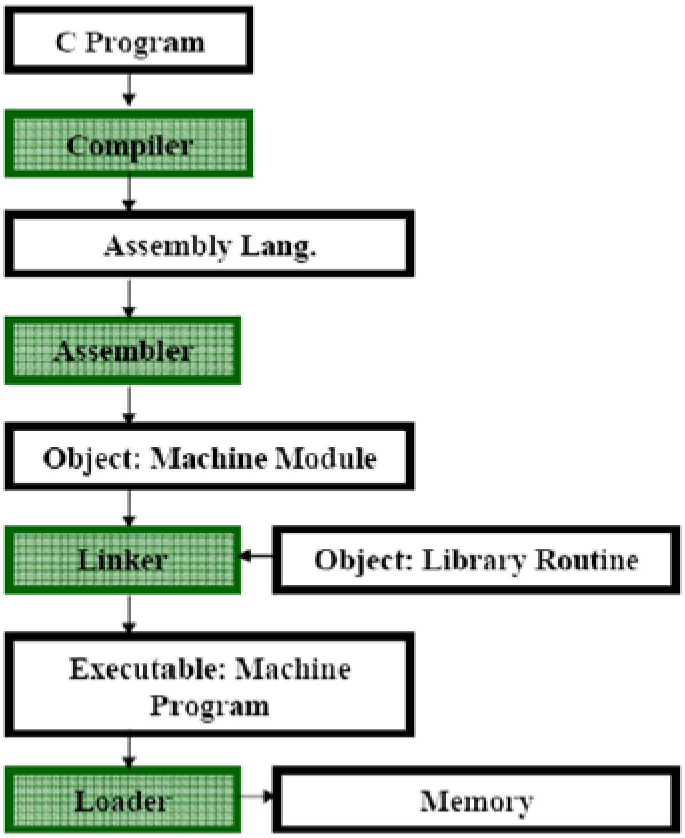
32-Bit Protected Mode:
32-bit protected mode programs run under all 32-bit versions of Microsoft Windows. They are usually easier to write and understand than real mode programs.

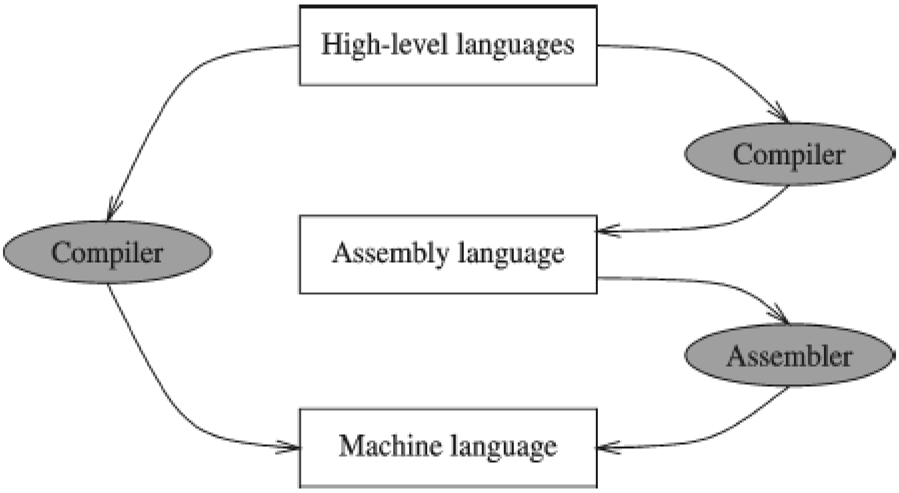
- Link.exe creates an .exe file from a .obj file.
- Use make16.bat to assemble and link a 16-bit format assembly program.
- Use make32.bat to assemble and link a 32-bit format assembly program.

➢ A **debugger** is a program that allows you to trace the execution of a program and examine the content of registers and memory.

- For 16-bit programs, MASM supplies a 16-bit debugger named CodeView. CodeView can be used to debug only 16-bit programs and is already provided with the MASM 6.15 distribution.
- For 32-bit protected-mode programs, you need a 32-bit debugger. The latest version of the 32-bit Windows debugger is available for download for free from Microsoft.

➢ **Compilers** translate high-level programs to machine code either directly or indirectly via an assembler.





- Programs in high level language are portable; However, Assembly language programs are not portable.
  - A portable application (portable app), sometimes also called standalone, is a computer program designed to run independently from an operating system.

  - Portable versions of applications can usually also be used instead of the normal installable version, which some users find useful.

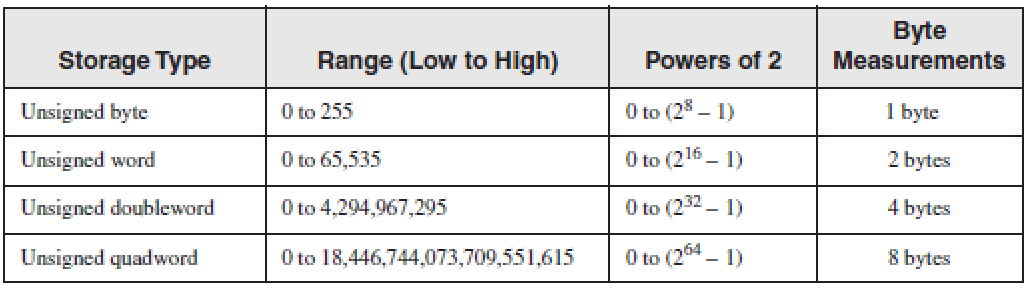
- The Microsoft Macro Assembler (**MASM**) is an x86 assembler that uses the Intel syntax for MS-DOS and Microsoft Windows.
  - x86-64 (or x64) is often used to denote 64-bit software, while the term x86 implying only 32-bit.

- **Integer Storage Sizes**
  The basic storage unit for all data in an x86 computer is a *byte*, containing 8 bits. Other storage sizes are *word* (2 bytes), *doubleword* (4 bytes), and *quadword* (8 bytes).

  Large Measurements: A number of large measurements are used when referring to both memory and disk space:

  - One *kilobyte* is equal to 2^{10}, or 1024 bytes.
  - One *megabyte* (1 MByte) is equal to 2^{20}, or 1,048,576 bytes.
  - One *gigabyte* (1 GByte) is equal to 2^{30}, or 1024^3, or 1,073,741,824 bytes.
  - One *terabyte* (1 TByte) is equal to 2^{40}, or 1024^4, or 1,099,511,627,776 bytes.
  - One petabyte is equal to 2^{50}, or 1,125,899,906,842,624 bytes.
  - One exabyte is equal to 2^{60}, or 1,152,921,504,606,846,976 bytes.
  - One zettabyte is equal to 2^{70} bytes.
  - One yottabyte is equal to 2^{80} bytes.

The unsigned range:

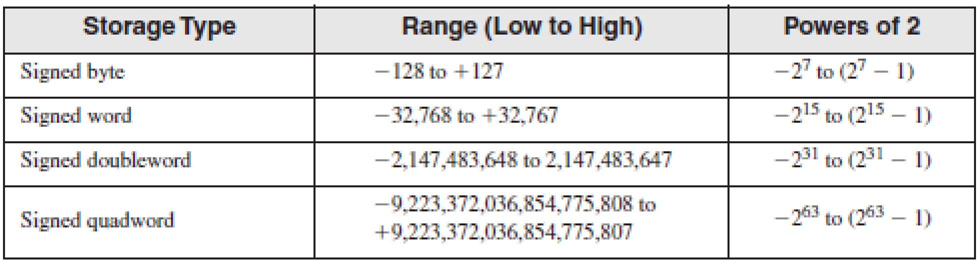


Storage Type	Range (Low to High)	Powers of 2	Byte Measurements
Unsigned byte	0 to 255	0 to $(2^8 – 1)$	1 byte
Unsigned word	0 to 65,535	0 to $(2^{16} – 1)$	2 bytes
Unsigned doubleword	0 to 4,294,967,295	0 to $(2^{32} – 1)$	4 bytes
Unsigned quadword	0 to 18,446,744,073,709,551,615	0 to $(2^{64} – 1)$	8 bytes

(1.3.7 P28) What is the largest value you can represent using a 256-bit *unsigned* integer?

Sol: $\boxed{\mathbf{2^{256}-1}}$

The signed range:



Storage Type	Range (Low to High)	Powers of 2
Signed byte	−128 to +127	-2^7 to $(2^7 - 1)$
Signed word	−32,768 to +32,767	-2^{15} to $(2^{15} - 1)$
Signed doubleword	−2,147,483,648 to 2,147,483,647	-2^{31} to $(2^{31} - 1)$
Signed quadword	−9,223,372,036,854,775,808 to +9,223,372,036,854,775,807	-2^{63} to $(2^{63} - 1)$

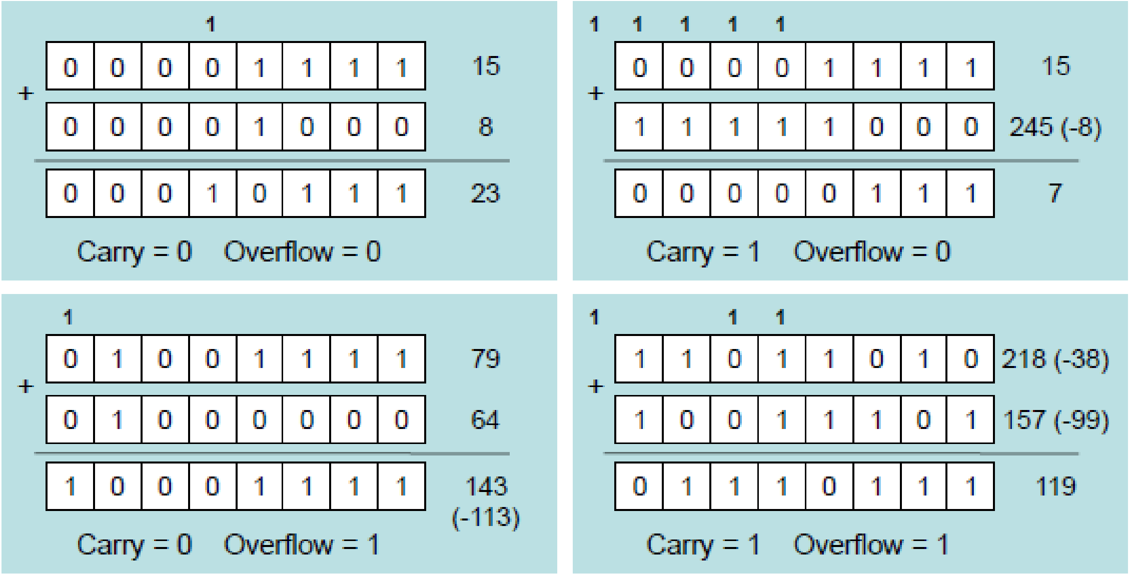
(1.3.7 P29) What is the largest positive value you can represent using a 256-bit *signed* integer?

Sol: $\boxed{\mathbf{+\ 2^{255}-1}}$

➢ Overflow

Overflow occurs when

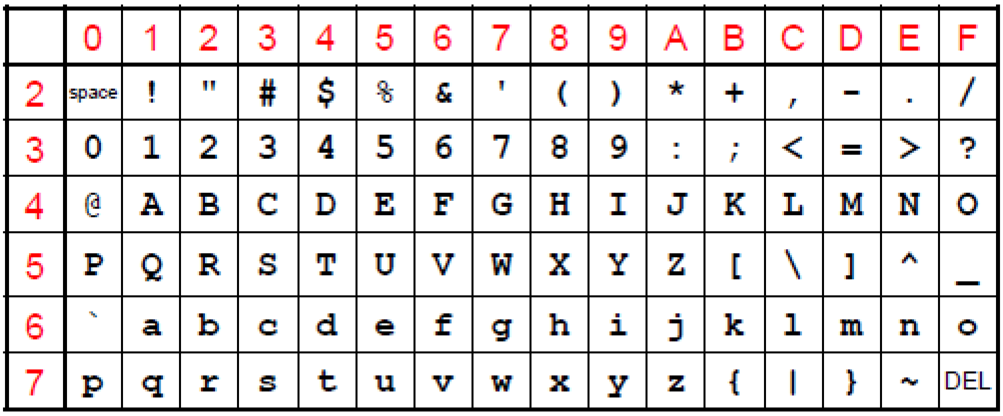
- Adding two positive numbers and the sum is negative
- Adding two negative numbers and the sum is positive



- **Character Storage**

- Standard ASCII: 7-bit character codes (0 – 127)
- Extended ASCII: 8-bit character codes (0 – 255)

Printable ASCII Codes



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	space	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Examples:

- ASCII code for space character = 20 (hex) = 32 (decimal)
- ASCII code for 'L' = 4C (hex) = 76 (decimal)
- ASCII code for 'a' = 61 (hex) = 97 (decimal)

(1.3.7 P26) What are the hexadecimal and decimal representations of the ASCII character capital M?

Sol: **4Dh and 77d**

Control Characters

- Control character codes = 00 to 1F (hex)

Examples of Control Characters

- Character 0 is the NULL character, used to terminate a string
- Character 9 is the Horizontal Tab (HT) character
- Character 0A (hex) = 10 (decimal) is the Line Feed (LF)
- Character 0D (hex) = 13 (decimal) is the Carriage Return (CR)
- The LF and CR characters are used together , they advance the cursor to the beginning of next line