WEEK03 – COMPUTER SOFTWARE

WEN-BIN JIAN

DEPARTMENT OF ELECTROPHYSICS, NATIONAL CHIAO TUNG UNIVERSITY

OUTLINE

1. Machine Language
2. Operating System – Computer
3. Operation System – Hand Held
4. Functions of OS
5. MS-DOS
6. ASCII Code
7. File System
8. Editor for Text & Hexadecimal Numbers

MACHINE LANGUAGE

The 8086 Instruction Set

One instruction consists of an operation code (OP Code) and an Operand.

An example of the machine code:

00100 B4 09 BA 0B 01 CD 21 B4 4C CD 21 68 65 6C 6C 6F You can use DosBox emulator to check it.
00110 20 77 6F 72 6C 64 20 21 24 00

relative address code & data

B4 09:	mov ah, 09h; ⟸ Assembly Language
BA 0B 01:	mov dx, message; // put the address of our message in the dx resgiter
BA 0B 01:	0B 01 means the relative address at 010B -> 68 65 … 24 -> hello world !$
CD 21:	int 21h; // call the function, provided by the DOS operating system, to print
B4 4C:	mov ah, 4Ch;
CD 21:	int 21h; // call the DOS function to exit

Ref1: https://www.youtube.com/watch?v=L1_SLBLdJVY

BINARY NUMBER CALCULATION & OPERATION

- Addition

```
  0000 1011
+ 0011 0101
-----------
  0100 0000
```

- Unsigned and **Signed Short Integers** 1 = 0000 0001 -> -1 = 1111 1110 + 1 = 1111 1111

  -1 = 1111 1111 -> 0000 0000 + 1 = 0000 0001 = 1

  The highest bit is used to mark the sign of the number.

- Subtraction

```
  0011 0101          0011 0101
- 0000 1011        + 1111 0101
-----------        -----------
  0010 1010          0010 1010
```

- Multiplication / Shift Operation

```
  0001 0101          0101 0100
X 0000 0101        + 0001 0101
-----------        -----------
                     0110 1001
```

- Division / Shift Operation 0001 0101 / 0000 0010 = 0000 1010

- And & Or Operations to Set One Bit On/Off in a Byte (Set The Flag)

0001 0101 Or 0000 1000 = 0001 1101 0001 1101 And 1111 0111 = 0001 0101

BINARY NUMBER CALCULATION & OPERATION

- From unsigned integer to signed integer
  - One byte (char), one byte unsigned int
  - Two bytes (word), two byte unsigned int
- How do you change the bin to signed int?
  - The MSB (most significant) bit gives +-
  - If it is negative, do NOT operation and add 1.
- Why do you use this kind of encoding for signed integers?
  - Easy for subtraction operation

binary from 0000 0000 to 1111 1111
hex from 0 to F F

Encode it to become a signed integer
the binary number is the same
from 0000 0000 to 1111 1111

binary from 0000 0000 0000 0000 to 1111 1111 1111 1111
hex from 0 to F F F F

Encode it to become a signed integer
the binary code is the same but the interpretation changes
0000 0000 0000 0000 → 0,
1000 0000 0000 0000 → -32768 (dec)

10 – 8 = 2 → 0000 1010 + 1111 1000 = (1) 0000 0010
2 – 8 = -6 → 0000 0010 + 1111 1000 = 1111 1010 → -6

BINARY NUMBER CALCULATION & OPERATION

- Multiplication
  - Use shift to left operation
- Divide by 2^n
  - Use shift to right operation
- Bit operation
  - Set bit to 0
  - Set bit to 1
  - Read bit

3 x 2 = 6 (dec) → 0011 shift to left 0110

0000

3 x 5 = 3 x (1 + 4) = 3 x 1 + 3 x 4 (dec)

7 x 11 = 77 → 7 x (8 + 2 + 1)

0000 0111. shift left 3: 0011 1000. shift left 1: 0000 1110.

001

77 (dec)

3 / 2 = 1 (dec) → 0011 shift to right 0001

12 / 4 = 3 (dec) → 0000 1100 shift right 2: 0000 0011

Data: 1010 0100

Set bit 2 false (off): 1010 0100 and (&) 1111 1011

Set bit 0 true (on): 1010 0100 or (|) 0000 0001

Check bit 1: results of 1010 0100 and 0000 0010 → false

Check bit 2: results of 1010 0100 and 0000 0100 → true

TOOLS AND TRICKS USED IN CPU & OS

- Different Meaning: Operation Code & Operands / Commands & Parameters / Functions or Procedures & Input / Output Parameters
- Queue of Operations / Process – First In First Out
- Stacking of Operations / Data / Process – First In Last Out
- Interrupt – Real Time Response to User (Keyboard / Mouse) → event-driven programming
- System Function Calls – Do More Complicated Work
- Flag – Register The Current Condition
- Idle – Waiting for The Completion of I/O

OPERATING SYSTEMS (OS) - COMPUTER

- UNIX:"In 1969, Ken Thompson, Dennis Ritchie and others started working on the "little-used PDP-7 in a corner" at Bell Labs and what was to become UNIX." GNU project
- DOS: Released in 1981 by IBM while developed by Microsoft. PC-DOS / MS-DOS
- MAC iOS: Introduced in 1984 with "graphical user interface".
- WINDOWS: In 1983, Bill Gates announced the using of GUI for DOS.
  - The early versions are Windows 1.0-3.11.
  - Win 3.1->Win NT 3.1->Win 95->Win NT 4.0->Win 98->Win 2000->Win ME->Win XP->Win Vista->Win 7->Win 8->Win 10
- LINUX: In 1991, Linus announced the first version (0.02) of Linux. NetBSD, FreeBSD, Red Hat, Ubuntu

Ref1: http://www.unix.org/what_is_unix/history_timeline.html

OPERATING SYSTEMS (OS) – HAND HELD SYSTEM

- Windows Phone: In 1996, the Microsoft company released Windows CE 1.0. 2000: Pocket PC 2000, 2008: Windows Mobile 6.1 – Sony Ericsson Xperia X1
- PalmOS: 1996: first Palm Pilot, 2003: PalmOne Treo 600 (PalmOS 5.2.1, 144MHz ARM-based CPU, 32 MB of RAM, and a 160x160 resolution color touchscreen)
- Symbian OS: 1998 used in PDA (personal digital assistant)
- BlackBerry: 2002 The first BlackBerry OS (BlackBerry 3.0) is used on a phone.
- iOS: 2007 Version 1.X, 2008 Version 2.X for iPhone 3G (Samsung S5PC100 32-bit RISC ARM11 620 MHz, ARM - Advanced RISC Machine) & iPod Touch (ARM), 2010 Version 3.13 for 32-bit ARM
- Android: In 2008, the first Android smartphone was announced, the T-Mobile-G1 (also known as HTC Dream)
- 2010 HTC-Desire, CPU: ARM(1 GHz Qualcomm QSD8250 Snapdragon), OS: Android 2.1

Ref1: http://www.unix.org/what_is_unix/history_timeline.html

FUNCTIONS OF OPERATING SYSTEMS (OS)

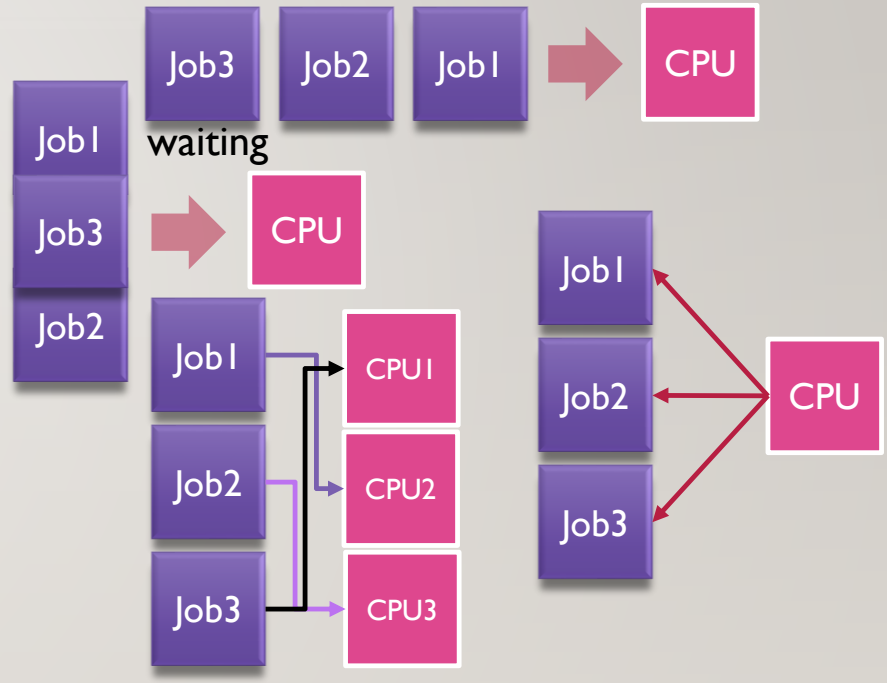
System Programs & Application Programs

- **Memory Management** – Keep Track of Memory Used for Application Programs, Allocate Memory on Request
- **Process Management** – Arrange Schedule for Processes
- **File System and File Management** – NTFS, FAT32, exFAT
- **Device Management** – I/O Control, Usually Accompanied With Driver
- **Security Control** – Password for Accessing The Computer or Each Files
- **Error Handling** – Report Error, Prevent Unexpectedly Shut Down
- **Checking System Performance** – Clean Cache Files, Temporarily Saved Data
- Provide **System Function Calls** (System Calls) for The Application Programs – WinAPI, DOS API – int 21h

Ref1: https://www.tutorialspoint.com/operating_system/os_overview.htm

TYPES OF OPERATING SYSTEMS

- **Batch** operating system – similar jobs are batched together, CPU is often in an idle state, no priority arrangement, easily get crashed
- **Multiprogramming OS** – executing another job when idle, executing jobs with different priorities
- **Time-sharing OS** – CPU time shared by all programs, quick response but poor reliability
- **Distributed OS** – multiple CPUs, jobs are distributed to one of the CPUs accordingly (parallel computing)
- **Network OS** – server to client operations, share files and hardware
- **Real-time OS** – for real-time machine operation



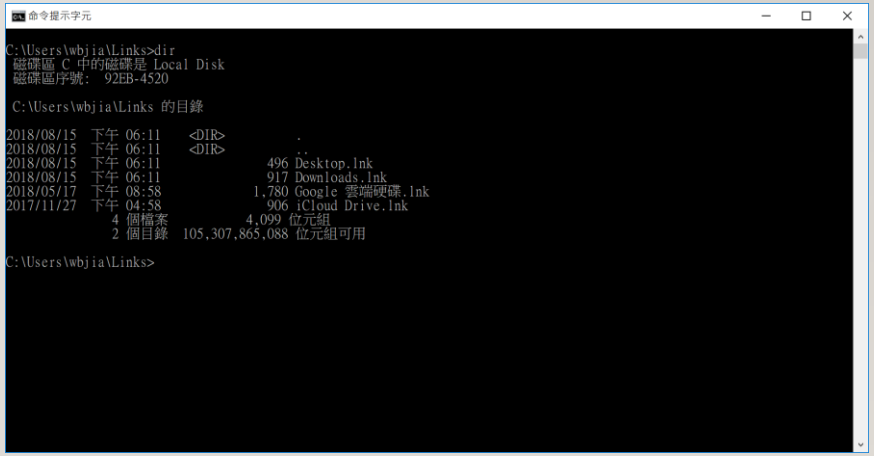
Ref1: https://www.tutorialspoint.com/operating_system/os_types.htm
Ref2: http://www.it.uu.se/education/course/homepage/os/vt18/module-1/multiprogramming/

Exercise

OPERATING SYSTEMS (OS) - DOS

- DOS – Disk Operation System, Commands & its batch file
  - **dir** or dir /s (with /s options) – show directory & files, help dir – show how to use the command dir
  - **cls** – clear screen, **cd** – change directory, **mkdir** – make a new directory
  - **format** – format e:, initialize and empty the disk e
  - **date** (**time**) – show current date and prompt to input new date
  - **echo** – print our text, **type** – print out text in a file (text are ASCII coded symbols)
  - **ping** 140.113.6.2 – use the domain name server to check your internet connection
  - **tree** – directory structures of the current disk
  - tree > ex02a.txt – redirect output to the file of the name "ex02a.txt"
  - type ex02a.txt – show text contents
  - ex02.txt – Windows will identify the extension "txt" and call the relating application program

The MS-DOS Emulator in Windows 10



Ref1: https://www.computerhope.com/msdos.htm

OPERATING SYSTEMS (OS) – FILE CONTENTS & ASCII CODES

- Files always consist of binary codes. The binary condes are used to represent "characters" as well, thus you can express the binary codes as text.
- The binary coding of the file is the same. The play role of the file content is only dependent on its extension name (file suffix).
- If the filename extension is "**exe**", the binary codes are operation codes and operands.
- If the filename extension is "**txt**", the binary codes are characters and symbols.
- If the filename extension is like "**dat**", the binary codes are 16bit or 32 bit data or even image data.
- The file contents are composed of bytes or words (two bites) so we define the symbols by either one byte (0-255, **ASCII** codes) or one word (0-65535, **UNICODE**).

Ref1: https://commons.wikimedia.org/wiki/File:Ascii-codes-table.png

The ASCII code

American Standard Code for Information Interchange

ASCII control characters

DEC	HEX	Simbolo ASCII	
00	00h	NULL	(carácter nulo)
01	01h	SOH	(inicio encabezado)
02	02h	STX	(inicio texto)
03	03h	ETX	(fin de texto)
04	04h	EOT	(fin transmisión)
05	05h	ENQ	(enquiry)
06	06h	ACK	(acknowledgement)
07	07h	BEL	(timbre)
08	08h	BS	(retroceso)
09	09h	HT	(tab horizontal)
10	0Ah	LF	(salto de linea)
11	0Bh	VT	(tab vertical)
12	0Ch	FF	(form feed)
13	0Dh	CR	(retorno de carro)
14	0Eh	SO	(shift Out)
15	0Fh	SI	(shift In)
16	10h	DLE	(data link escape)
17	11h	DC1	(device control 1)
18	12h	DC2	(device control 2)
19	13h	DC3	(device control 3)
20	14h	DC4	(device control 4)
21	15h	NAK	(negative acknowle.)
22	16h	SYN	(synchronous idle)
23	17h	ETB	(end of trans. block)
24	18h	CAN	(cancel)
25	19h	EM	(end of medium)
26	1Ah	SUB	(substitute)
27	1Bh	ESC	(escape)
28	1Ch	FS	(file separator)
29	1Dh	GS	(group separator)
30	1Eh	RS	(record separator)
31	1Fh	US	(unit separator)
127	20h	DEL	(delete)

ASCII printable characters

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_			

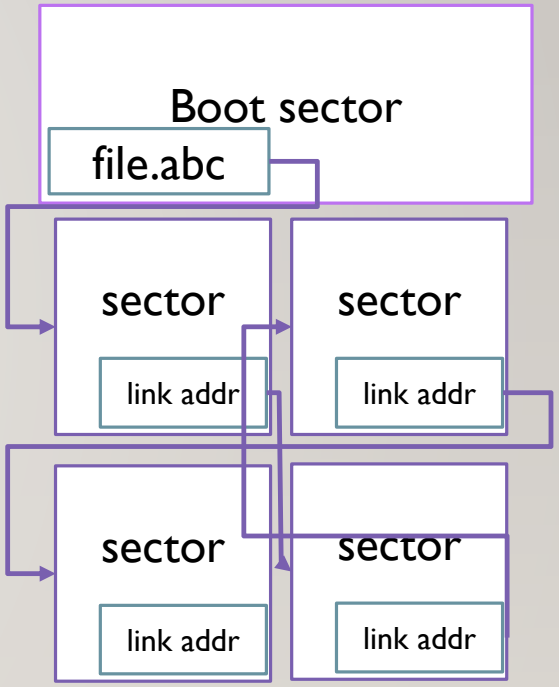
theASCIIcode.com.ar

Extended ASCII characters

DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo	DEC	HEX	Simbolo
128	80h	Ç	160	A0h	á	192	C0h	└	224	E0h	Ó
129	81h	ü	161	A1h	í	193	C1h	┴	225	E1h	ß
130	82h	é	162	A2h	ó	194	C2h	┬	226	E2h	Ô
131	83h	â	163	A3h	ú	195	C3h	├	227	E3h	Ò
132	84h	ä	164	A4h	ñ	196	C4h	─	228	E4h	õ
133	85h	à	165	A5h	Ñ	197	C5h	┼	229	E5h	Õ
134	86h	á	166	A6h	ª	198	C6h	ã	230	E6h	µ
135	87h	ç	167	A7h	º	199	C7h	Ã	231	E7h	þ
136	88h	ê	168	A8h	¿	200	C8h	╚	232	E8h	Þ
137	89h	ë	169	A9h	®	201	C9h	╔	233	E9h	Ú
138	8Ah	è	170	AAh	¬	202	CAh	╩	234	EAh	Û
139	8Bh	ï	171	ABh	½	203	CBh	╦	235	EBh	Ù
140	8Ch	î	172	ACh	¼	204	CCh	╠	236	ECh	ý
141	8Dh	ì	173	ADh	¡	205	CDh	═	237	EDh	Ý
142	8Eh	Ä	174	AEh	«	206	CEh	╬	238	EEh	¯
143	8Fh	Å	175	AFh	»	207	CFh	¤	239	EFh	´
144	90h	É	176	B0h	░	208	D0h	ð	240	F0h	
145	91h	æ	177	B1h	▒	209	D1h	Ð	241	F1h	±
146	92h	Æ	178	B2h	▓	210	D2h	Ê	242	F2h	‗
147	93h	ô	179	B3h	│	211	D3h	Ë	243	F3h	¾
148	94h	ò	180	B4h	┤	212	D4h	È	244	F4h	¶
149	95h	ò	181	B5h	Á	213	D5h	ı	245	F5h	§
150	96h	û	182	B6h	Â	214	D6h	Í	246	F6h	÷
151	97h	ù	183	B7h	À	215	D7h	Î	247	F7h	¸
152	98h	ÿ	184	B8h	©	216	D8h	Ï	248	F8h	°
153	99h	Ö	185	B9h	╣	217	D9h	┘	249	F9h	¨
154	9Ah	Ü	186	BAh	║	218	DAh	┌	250	FAh	·
155	9Bh	ø	187	BBh	╗	219	DBh	█	251	FBh	¹
156	9Ch	£	188	BCh	╝	220	DCh	▄	252	FCh	³
157	9Dh	Ø	189	BDh	¢	221	DDh	¦	253	FDh	²
158	9Eh	×	190	BEh	¥	222	DEh	▌	254	FEh	■
159	9Fh	ƒ	191	BFh	┐	223	DFh	▀	255	FFh	

OPERATING SYSTEMS (OS) – FILE SYSTEM

- File Allocation Table (FAT):
  - started since the development of DOS, several different types – FAT12, FAT16, FAT32, NTFS, use **link list** data structure
  - 5.25" Disk: 1 side, 40 tracks on each side, 8 sectors on each track, 512 bytes on each sector, totally 160 kBytes; boot sector on the 1st track (boot & file information)
  - FAT32: 32 bits for restoring sector address, 512 bytes or 8 clusters (8 * 512 bytes) on each sector, totally $2^{32} \times 512 \cong 2 \times 10^9$ bytes.
- Disks, Directories & Files:
  - Disk Label & Title: A & B for floppy, C, D, … for hard disk, CD ROM, and other plug and play drives
  - Disk -> Root directory, consisting of files and directories – tree data structure



Ref1: https://www.minitool.com/lib/boot-sector.html

Exercise

APPPLICATION PROGRAM – EDITOR

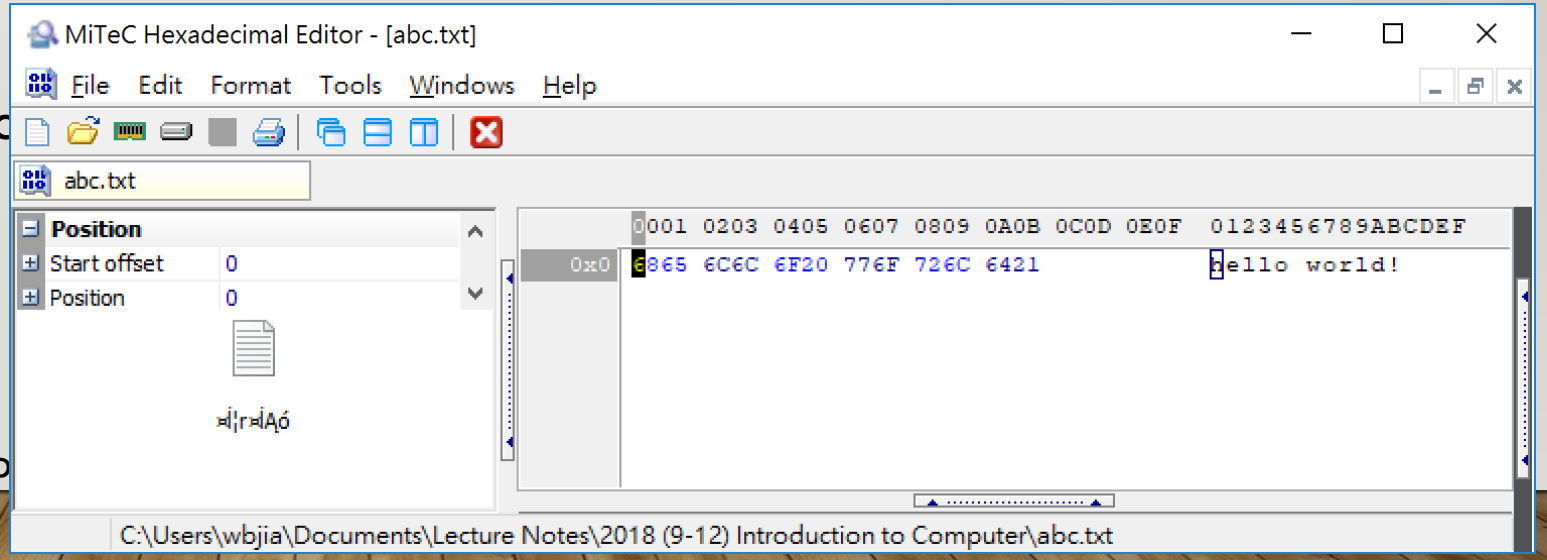
- Install PSPad (or Notepad++) – A text editor used for "programming" – a sequence of codes that can be executed once
- Google search: pspad
- Download the latest version, only 32-bit version
- Check & doubt whether you are targeted by virus in the execution files
- Open the application program – PSPad (notepad++) and take a look of its menu & functions
- Where is your installed program files in the disk?
- Start PSPad to edit a text file – create abc.txt and put a line of hello world in it

Ref1: http://www.pspad.com/

Exercise

APPLICATION PROGRAM – EDITOR

- Install a hexadecimal editor to see what's in the ex02a.txt
- Google search HEXEdit
- Download and open it by WinRAR
- Use HEXEdit.exe program to open the text file ex02a.txt
- What do you see?
- The files are always in binary c

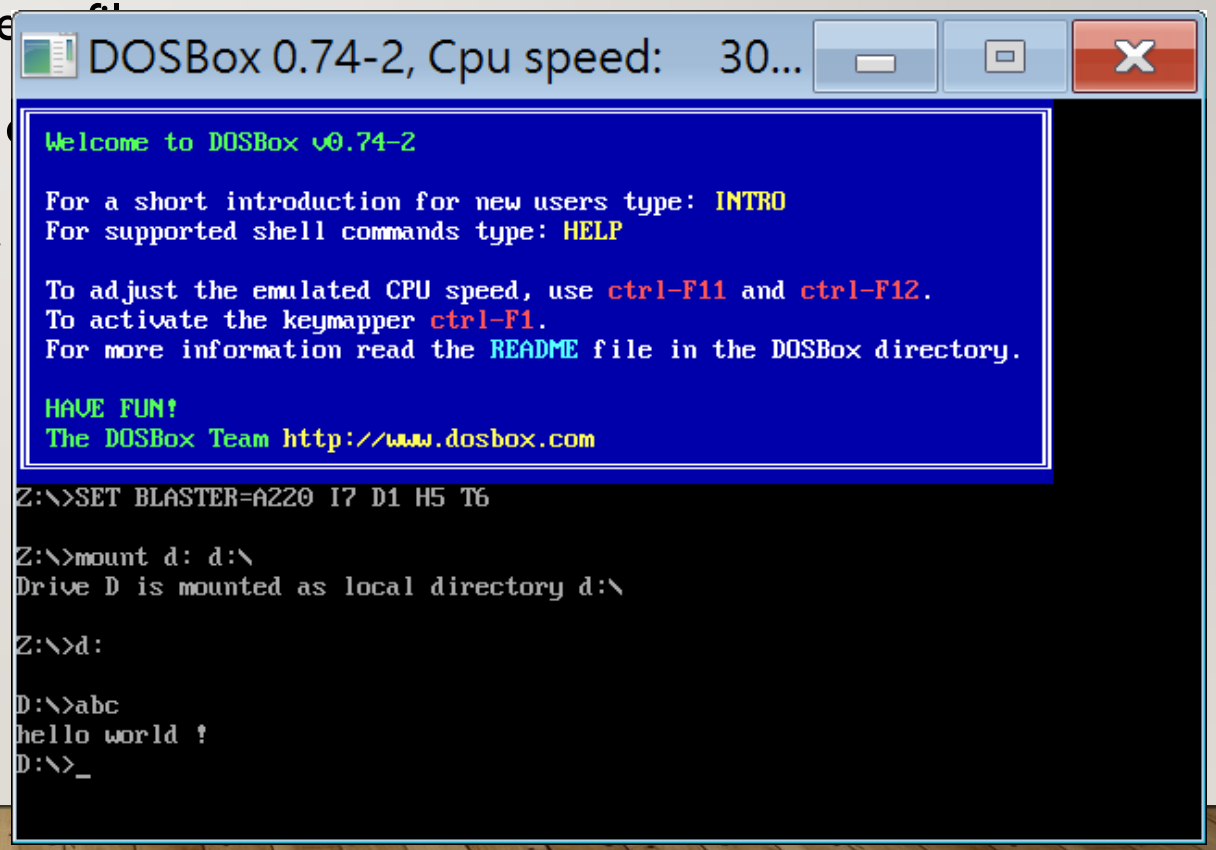


Ref1: https://www.techworld.com/download/developer-p

Exercise

MACHINE CODE EDITING

- Use PSPad to create an exe file with a specified file size.
- Use PSPad to edit the machine code of the e
- Use DOSBox (DOS emulator), mount your
- Run your edited program of machine codes.



EXERCISE

1. For a 8-bit, signed integer, please show the decimal numbers 20, -40, 110, -120 in the binary form.
2. For a 16-bit and signed integer, please show the decimal numbers of -3268, 32767, -1, 1 in binary form.
3. Please convert the decimal numbers 50, 4 to binary numbers (use unsigned 8-bit format) and calculate results of multiplication (50*4) and division (50/4) using the shift operation.
4. Please use 8-bit signed binary numbers to show how computer calculate the decimal number operation of 7 x 3 = 21.
5. Please use 8-bit signed binary numbers to show how computer calculate the decimal number operation of 3 – 4 = -1.