Laboratory 04
INT for IO

CO 2103 Assembly Language

Objective

AL programming using software interrupt

-INT (21h and 10h) instructions for IO

-basic keyboard, screen programming

INT 21h: Few Useful Ones - 1

--------D-20---------------------------------

INT 20 - TERMINATE PROGRAM

--------D-2101-------------------------------

INT 21 - DOS 1+ - READ CHARACTER FROM STANDARD INPUT, WITH ECHO

AH = 01h

Return: AL = character read

Notes: ^C/^Break are checked

--------D-2102-------------------------------

INT 21 - DOS 1+ - WRITE CHARACTER TO STANDARD OUTPUT

AH = 02h

DL = character to write

Return: AL = last character output (despite the official docs which state nothing is returned) (at least DOS 3.3-5.0)

Notes: ^C/^Break are checked

INT 21h: Few Useful Ones - 2

--------D-2107------------------------------
INT 21 - DOS 1+ - DIRECT CHARACTER INPUT, WITHOUT ECHO
    AH = 07h
Return: AL = character read from standard input
Notes: does not check ^C/^Break

--------D-2108------------------------------
INT 21 - DOS 1+ - CHARACTER INPUT WITHOUT ECHO
    AH = 08h
Return: AL = character read from standard input
Notes: ^C/^Break are checked

--------D-2109------------------------------
INT 21 - DOS 1+ - WRITE STRING TO STANDARD OUTPUT
    AH = 09h
    DS:DX -> '$'-terminated string
Return: AL = 24h (the '$' terminating the string, despite official docs which state that nothing is returned) (at least DOS 3.3-5.0)
Notes: ^C/^Break are checked

INT 21h: Few Useful Ones - 3

--------D-210A---------------------------------

INT 21 - DOS 1+ - BUFFERED INPUT

AH = 0Ah

DS:DX -> buffer (see Format of DOS input buffer below)

Return: buffer filled with user input

Notes: ^C/^Break are checked. Input starts at buffer+2 (see Format of input buffer below)

Format of DOS input buffer:

Offset	Size	Description
00h	BYTE	maximum characters buffer can hold
01h	BYTE	(call) number of chars from last input which may be recalled (return) number of characters actually read, excluding CR
02h	N BYTEs	actual characters read, including the final carriage return

How to use INT 21h?

- INT 21h is an OS function call which can be configured for different functions. Examples:
  - set AH=01h and call INT 21h will read a character from standard input (keyboard) and echo it on the screen; ASCII code of the character read is stored in register AL
    - mov ah,1 ;select function 01
    - INT 21h ;call the function, i.e. read character
  - set AH=02h and call INT 21 will write a character to the standard output (screen); the ASCII code of the character to be written is retrieved from register DL
    - mov dl,41h ;store ASCII code ("A") in DL first
    - mov ah,2 ;select function 02
    - INT 21h ;call the function, i.e. write character

INT 21h Exercise

Use MASM and LINK to create the programs and use DEBUG to test (gain more understand) the programs

- **Task 1:** Write a simple program that reads the ASCII code of a character and display the character to the screen on next line (you are not allowed to use the INT function that echo the input): save as echo1.asm
- **Task 2:** Write a simple program that reads two characters and display them on screen on next line: save as echo2_1.asm
  - Try to use buffered input: save as echo2_2.asm
- **Task 3:** Write a simple program that reads two characters (numbers) of one digit each, add them and display the result on screen: save as cal1_1.asm. Hint: Adjust ASCII to number

INT 21h Exercise:
Simple Calculator

- **Task 4:** Improve the program in Task 3 to include the followings: save as cal1_2.asm
  - proper message prompts (user friendliness)
  - ability to check for valid inputs (0 to 9) and give error message and re-ask for input if invalid input received
  - ability to print result up to 2 digits (0 to 18)

Cursor Positioning - 1

--------B-1001------------------------------
INT 10 - SET CURSOR SIZE
AH = 01
CH = cursor starting scan line (cursor top) (low order 5 bits)
CL = cursor ending scan line (cursor bottom) (low order 5 bits)
Returns nothing

--------B-1002------------------------------
INT 10 - SET CURSOR POSITION
AH = 02
BH = page number (0 for graphics modes)
DH = row
DL = column
Returns nothing
Positions relative to 0,0 origin

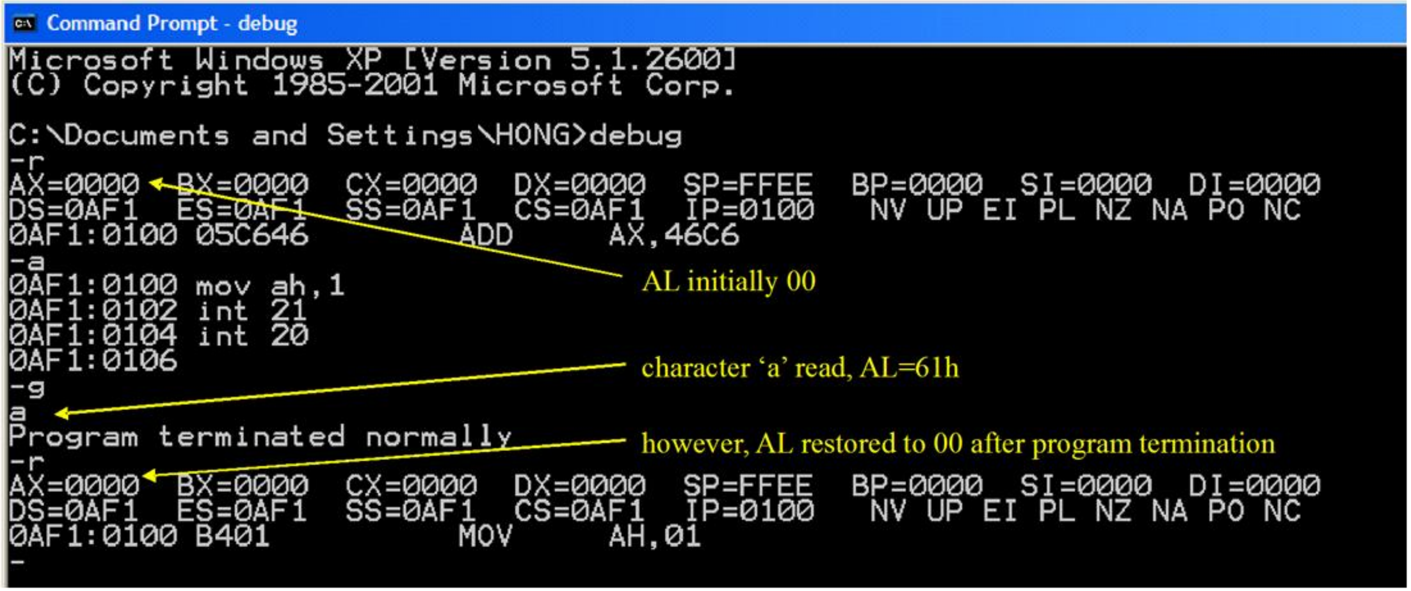
Cursor Positioning - 2

```
--------B-1003------------------------------
INT 10 - READ CURSOR POSITION AND SIZE
        AH = 03
        BH = video page
Return:
       CH = cursor starting scan line (low order 5 bits)
       CL = cursor ending scan line (low order 5 bits)
       DH = row
       DL = column
```

- **Task 5:** Write a program to print ‘X’ at the centre and four corners of the screen (CMD Window). Save this file as cursor.asm.

Brief note on testing INT - 1

- It is useful to save the content of relevant register(s), e.g. AL that contains the ASCII code of the character read from keyboard, into the memory (for checking)
  - in debug, the content of the registers will be restored to its original value after running the program



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\HONG>debug
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0AF1  ES=0AF1  SS=0AF1  CS=0AF1  IP=0100   NV UP EI PL NZ NA PO NC
0AF1:0100 05C646        ADD     AX,46C6
-a
0AF1:0100 mov ah,1
0AF1:0102 int 21
0AF1:0104 int 20
0AF1:0106
-g
a
Program terminated normally
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0AF1  ES=0AF1  SS=0AF1  CS=0AF1  IP=0100   NV UP EI PL NZ NA PO NC
0AF1:0100 B401          MOV     AH,01
-
```

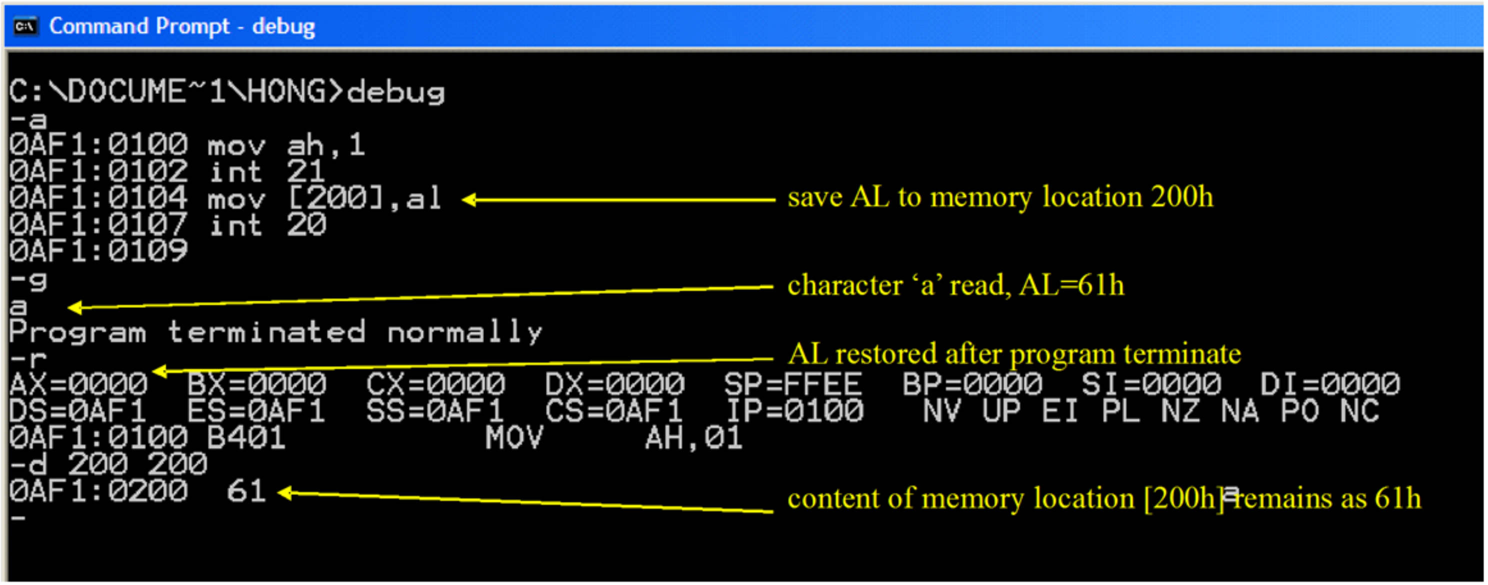
AL initially 00

character ‘a’ read, AL=61h

however, AL restored to 00 after program termination

Brief note on testing INT - 2

– saving the register(s) content into memory allows us to check using dump



```
C:\DOCUME~1\HONG>debug
-a
0AF1:0100 mov ah,1
0AF1:0102 int 21
0AF1:0104 mov [200],al
0AF1:0107 int 20
0AF1:0109
-g
a
Program terminated normally
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0AF1  ES=0AF1  SS=0AF1  CS=0AF1  IP=0100   NV UP EI PL NZ NA PO NC
0AF1:0100 B401          MOV     AH,01
-d 200 200
0AF1:0200  61
-
```

save AL to memory location 200h

character ‘a’ read, AL=61h

AL restored after program terminate

content of memory location [200h] remains as 61h

– you may use Proceed or Trace to monitor register content, though