CSE2006 Microprocessor & Interfacing

Module – 2 & 3

Introduction to ALP Advanced ALP

Dr. E. Konguvel
Assistant Professor (Sr. Gr. 1),
Dept. of Embedded Technology,
School of Electronics Engineering (SENSE),
konguvel.e@vit.ac.in
9597812810

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Module 3: Advanced ALP

- Interrupts
- Interrupt Programming
  - DOS
  - BIOS
- **File Management**

INT 21H

Create (or Truncate File: AH = 3CH

Entry:

CX = file attributes:

mov cx, 0 ; normal - no attributes.

mov cx, 1 ; read-only.

mov cx, 2 ; hidden.

mov cx, 4 ; system

mov cx, 7 ; hidden, system and read-only!

mov cx, 16 ; archive

DS:DX -> ASCIZ filename.

Return:

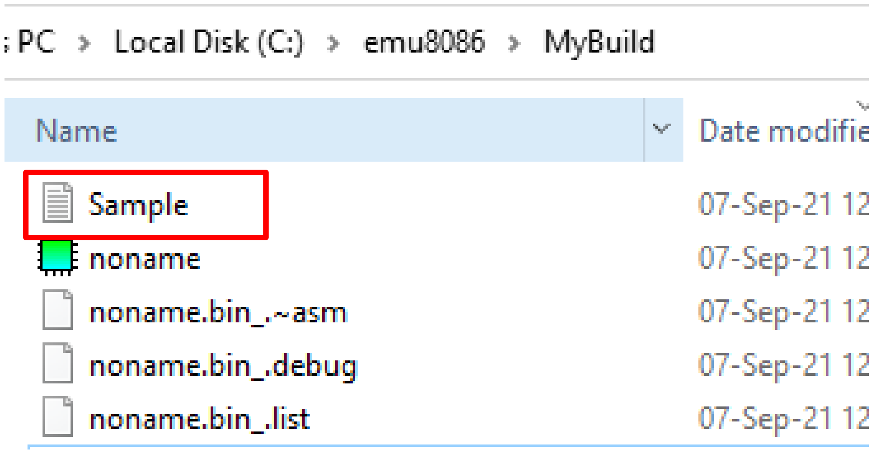
CF clear if successful, AX = file handle.

CF set on error AX = error code.

MyBuild Folder -> Sample.txt

```
;CREATE FILE
MOV CX, 0
MOV DX, OFFSET FNAME
MOV AH, 3CH
INT 21H
MOV FHANDLE, AX
```

```
FNAME DB "Sample.txt", 0
FHANDLE DW ?
```



```
PC > Local Disk (C:) > emu8086 > MyBuild
Name                         Date modifie
Sample                       07-Sep-21 12
noname                       07-Sep-21 12
noname.bin_.~asm             07-Sep-21 12
noname.bin_.debug            07-Sep-21 12
noname.bin_.list             07-Sep-21 12
```

INT 21H

Write into File: AH = 40H

Entry:

BX = file handle.

CX = number of bytes to write.

DS:DX -> data to write.

Return:

CF clear if successful;

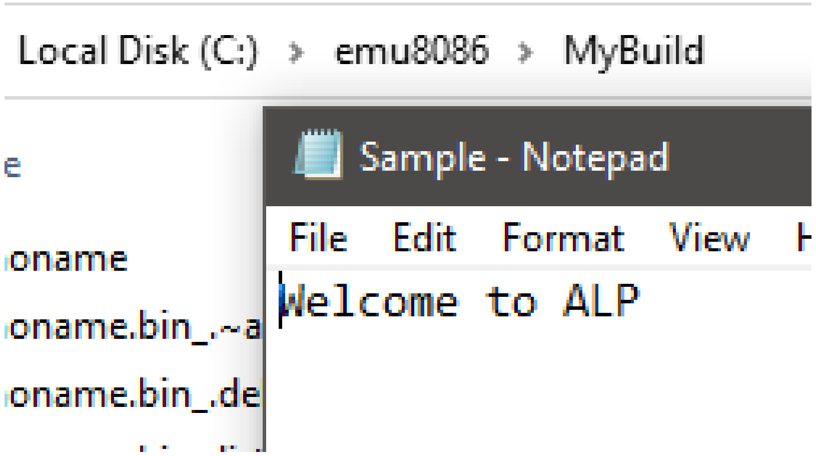
AX = number of bytes written.

CF set on error;

AX = error code.

```
;WRITE INTO FILE
MOV BX, FHANDLE
MOV DX, OFFSET DATA
MOV CX, DATA_SIZE
MOV AH, 40H
INT 21H
RET

FNAME DB "Sample.txt", 0
FHANDLE DW ?
DATA DB "Welcome to ALP"
DATA_SIZE=$-OFFSET DATA
```



```
Welcome to ALP
```

INT 21H

Read from File: AH = 3FH

Entry:

BX = file handle.

CX = number of bytes to read.

DS:DX -> buffer for data.

Return:

CF is clear if successful

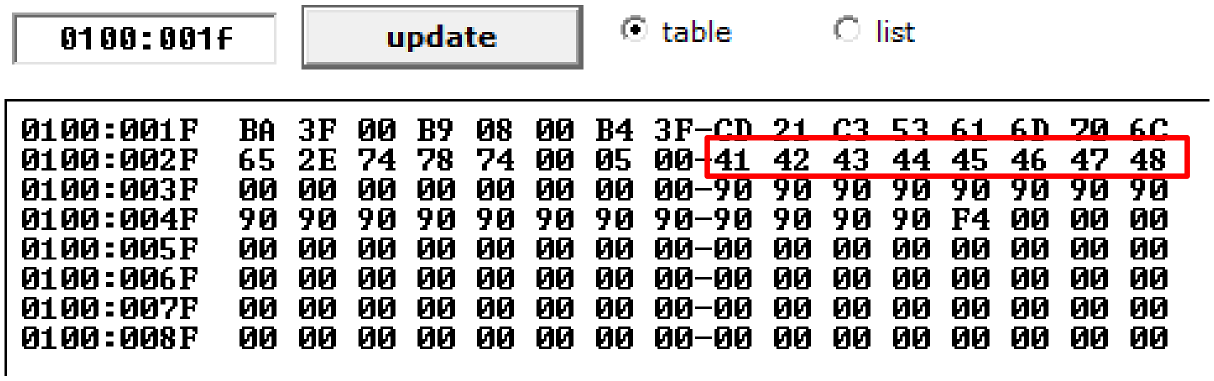
AX = number of bytes actually read;

CF is set on error

AX = error code.

```
;READ FROM FILE
MOV BX, FHANDLE
MOV DX, OFFSET BUFFER
MOV CX, 8
MOV AH, 3FH
INT 21H
RET

FNAME DB "Sample.txt", 0
FHANDLE DW ?
DATA DB "ABCDEFGH"
DATA_SIZE=$-OFFSET DATA
BUFFER DB 8 DUP (?)
```



```
0100:001F   BA 3F 00 B9 08 00 B4 3F-CD 21 C3 53 61 6D 70 6C
0100:002F   65 2E 74 78 74 00 05 00-41 42 43 44 45 46 47 48
0100:003F   00 00 00 00 00 00 00 00-90 90 90 90 90 90 90 90
0100:004F   90 90 90 90 90 90 90 90-90 90 90 90 F4 00 00 00
0100:005F   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0100:006F   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0100:007F   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
0100:008F   00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

INT 21H

Close File: AH = 3EH

Entry:

BX = file handle

Return:

CF clear if successful

AX destroyed.

CF set on error

AX = error code

```
;CLOSE FILE
MOV BX, FHANDLE
MOV AH, 3EH
INT 21H
RET
```

INT 21H

Open Existing File: AH = 3DH

Entry:

AL = access and sharing modes:

mov al, 0 ; read

mov al, 1 ; write

mov al, 2 ; read/write

DS:DX -> ASCIZ filename.

Return:

CF clear if successful, AX = file handle.

CF set on error AX = error code.

Note 1: file pointer is set to start of file.

Note 2: file must exist.

```
;OPEN EXISTING FILE
MOV AL, 2
MOV DX, OFFSET FNAME
MOV AH, 3DH
INT 21H
```

INT 21H

Seek: AH = 42H

Entry:

AL = origin of move:

0 - start of file

1 - current file position

2 - end of file

BX = file handle

CX:DX = offset from origin of new file position

Return:

CF clear if successful

DX:AX = new file position in bytes from start of file.

CF set on error

AX = error code

```
;SEEK
MOV AL, 0
MOV BX, FHANDLE
MOV CX, 0
MOV DX, 7
MOV AH, 42H
INT 21H
```

INT 21H

Delete (Unlink): AH = 41H

Entry:

DS:DX -> ASCIZ filename

Return:

CF clear if successful

AX destroyed

CF set on error

AX = error code

Note:

DOS does not erase the file's data; it merely becomes inaccessible because the FAT chain for the file is cleared deleting a file which is currently open may lead to file system corruption.

INT 21H

Get Current Directory: AH = 47H

Entry:

DL = drive number (00h = default, 01h = A:, etc,..)

DS:SI -> 64-byte buffer for ASCIZ pathname.

Return:

Carry is clear if successful

Carry is set on error, AX = error code (0Fh)

Note: The returned path does not include a drive and the initial backslash

INT 21H

Rename/Move File: AH = 56H

Entry:

DS:DX -> ASCIZ filename of existing file

ES:DI -> ASCIZ new filename

Return:

CF clear if successful

CF set on error, AX = error code

Note: Allows move between directories on same logical drive only; open files should not be renamed!