ภาคผนวก ค

BIOS and DOS Interrupts

C.1 Introduction

In this appendix, WC show some of the common BIOS and DOS interrupt calls. We begin with interrupt 10h; interrupts 0 to Fh are not normally used by application programs, their names are given in Table C. 1.

c.2 BIOS Interrupts

Interrupt 10: Video

Function Oh:
Select Display Mode
Selects video display mode
Input: AH = Oh
AL = video mode
output: none

Function 1 h:
Change Cursor Size
Selects the start and ending lines for the cursor.
Input: AH = 1h
Cl I (bits O-4) = starting line for cursor
Cl, (bits 0–4) = ending line for cursor
output: none

table C.I Interrupts 0 to 0Fh

Interrupt	*Type*	*usage*
0h		Divide by zero
Ih		Single step
2h		NM
3h		Breakpoint
4h		Overflow
5h		PrintScreen
6h		Reserved
7h		Reserved
8h		Timer tick
9h		Keyboard
0Ah		Reserved
0Bh		Serial communications (COM2)
0Ch		Serial communications (COM1)
0Dh		Fixed disk
0Eh		floppy disk
0Fh		Parallel minter

Function **2h:**
Move Cursor
Positions the cursor.
Input: AH = 2h
BH = page
DH = row
DL = column
output: none

Function 3h:
Get Cursor Position and Size
Obtains the current position and size of the cursor.
Input: AH = 3h
BH = page
Output: CH = starting line for cursor
CL = ending line for cursor
DH = row
DL = column

Function Sh:
Select Active Display Page
Input: AH = 5h
AL = page
DH = row
DL = column
output: none

Function 6h:
Scroll Window Up
scrolls the entire screen or a window up by a specified number of lines.
Input: AH = 6h
AL = number of lines to scroll
(if zero, entire window is blanked)
BH = attribute for blanked lines

CH,CL = row, column of upper left corner of windows
DH,DL = row, column of lower right corner of windows

output: none

Function 7h:
Scroll Window Down

Scrolls the entire screen or a window down by a specified number of lines

Input: AH = 7h
AL = number of liner to scroll
(If zero, entire window is blanked)
BH = attribute for blanked lines
CH,CL = row, column of upper left corner of window
DH,DL = row, column of lower right corner of window

output: none

Function **8h:**
Read Character and Attribute at Cursor

Obtains the ASCII character and its attribute at the cursor position.

Input: AH = 8h
BH = page

output: AH = attribute
AL = character

Function **9h:**
Write Character and Attribute at Cursor

Writes an ASCII character and its attribute at the cursor position.

Input: AH = 9h
AL = character
BH = page
BL = attribute (text mode) or color (graphics mode)
CX = count of characters to write

output: none

Function **0Ah:**
Write Character at Cursor

Writes an ASCII character at the cursor position. The character receives the attribute of the previous character at that position.

Input: AH = 0Ah
AL = character
BH = page
CX = count of characters to write

output: none

Function **0Bh:**
Set Palette, Background, or Border

Selects a palette, background color, or border color.

Input: To select the background color and border color
AH = 0Bh
BH = 0
BL = color
To select palette (320 x 200 four-color mode)
AH = 0Bh
BH = 1
BL = palette

Output: none

Function **0Ch:**
Write Graphics Pixel
Input: AH = 0Ch
AL = pixel value
BH = page
CX = column
DX = row
output: none

Function **0Dh:**
Read Graphics Pixel
Obtains a pixel value.
Input: AH = 0Dh
BH = page
CX = column
DX = row
output: AL = pixel value

Function **0Eh:**
Write Character in Teletype Mode
Writes an ASCII character at the cursor position, then increments cursor position.
Input: AH = 0Eh
AL = character
BH = page
BL = color (graphics mode)
Output: none
Note: the attribute of the character cannot be specified.

Function **0Fh:**
Get Video Mode
Obtains current display mode.
Input: AH = 0Fh
output: AH = number of character columns
AL = display mode
BH = active display page

Function **10h,** Subfunction **10h:**
Set Color Register
Sets individual VGA color register.
Input: AH = 10h
AL = 10h
BX = color register
CH = green value
CL = blue value
DH = red value
Output: none

Function **10h,** Subfunction **12h:**
Set Block of Color Registers
Sets a group of VGA color registers.
Input: AH = 10h
AL = 12h
BX = firstcolor register
CX = number of color registers
ES:DX = segment:offset of color table
Output: none

Note: the table consists of a group of three-byte entries corresponding to red, green, and blue values for each color register.

Function **10h,** Subfunction **15h:**
Get Color Register
Obtains the red, green, and blue values of a VGA color register.
Input: AH = 10h
AL = 15h
RX = color register
output: CH = green value
CL = blue value
DH = red value

Function **10h,** Subfunction **17h:**
Get Block of Color Registers
Obtains the red, green, and blue values of a group of VGA color registers.
Input: AH = 10h
AL = 17h
RX = first color register
CX = number of color registers
ES:DX = segment:offset of buffer to receive color list
output: ES:DX = segment:offset of buffer
Note: the color list consists of a group of three-byte entries corresponding to red, green, and blue values for each color register.

Interrupt 11h: Get Equipment Configuration

Obtains the equipment list code word.
Input: none
output: AX = equipment list code word
(bits 14–15 = number of printers installed,
13 = internal modem,
12 = game adapter,
9–11 = number of serial ports,
8 is reserved,
6-7 = number of floppy disk drives,
4-S = Initial video mode,
2-3 = system board RAM size, original PC
2 used by PS/2,
1 = math coprocessor,
0 = floppy disk installed)

Interrupt 12h: Get Conventional Memory Size

Returns the amount of conventional memory.
Input: none
Output: AX = memory size (In KB)

Interrupt 13h: Disk I/O

Function **2h:**
Read Sector
Reads one or more sectors.
Input: AH = 2h
AL = number of sectors

CH = cylinder
CL = sector
DH = head
DL = drive (0-7Fh = floppy disk, 80h-FFh = fixed disk)
ES:BX = segment:offset of buffer

output:

If function successful
CF = clear
AH =0
AL = number of sectors transferred
If function unsuccessful
CF = set
AH = error status

Function 3h:
Write Sector
Writes one or more sectors.

Input: AH = 3h
AL = number of sectors
BX = firstcolor register
CH = cylinder
CL = sector
DH = head
DL = drive (0-7Fh = floppy disk, 80h-FFh = fixed disk)
ES:BX = segment:offset of buffer

Output: If function successful
CF = clear
All = 0
AL = number of sectors transferred
If function unsuccessful
CF = set
AH = error status

Interrupt 15h; Cassette *I/O and Advanced* features *for AT, PS/2*

Function 87h:
Move Extended *Memory* Block
Transfers data between conventional memory and extended memory.

Input: AH = 87h
CX = number of words to move
ES:SI = segment:offset of Global Descriptor Table

output: If function successful
CF = clear
AH = 0
AL = number of sectors transferred
If function unsuccessful
CF = set
AH = error status

Function 88h:
Get Extended Memory Size
Obtains amount of extended memory

Input: All = 88h
Output: AX = extended memory size (in KB)

Interrupt 16h: Keyboard

Function Oh:
Read Character from Keyboard
Input: AH = Oh
output: AH = keyboard scan code
AL = ASCII character

Function 2h:
Get Keyboard Flags
Obtains key flags that describe the status of the function keys.
Input: AH = 2h
output: AL = flags

Bit	If Set
7	Insert on
6	Caps Lock on
5	Num Lock on
4	Scroll Lock on
3	Alt key is down
2	Ctrl key is down
1	left shift key is down
0	right shift key is down

Function 10h:
Read Character from Enhanced Keyboard
Input: AH = Oh
output: AH = keyboard scan code
AL = ASCII character
Note: this function can be used to return scan codes for control keys such as II I and F12.

Interrupt 17h: Printer

Function Oh:
Write Character to Printer
Input: AH = 0
AL = character
DX = printer number
output: AH = status

Bit	If Set
7	printer not busy
6	printer acknowledge
5	out of paper
4	printer selected
3	I/O error
2	unused
1	unused
0	printer timed out

Interrupt 21h

Function Oh:
Program Terminate
Terminates the execution of a program.
Input: AH=Oh
CS = segment of PSP
output: none

Function **1h:**
Keyboard Input
Waits for a character lo be read at the standard input device (unless one is ready), then echoes the character to the standard output device and returns the ASCII code in AL.
Input: AH = Olh
Output: AL = character from the standard input device

Function **2h:**
Display Output
Outputs the character in DL to the standard output device.
Input: AH = 02h
DL = character
Output: none

Function 5h:
Printer Output
Outputs the character in DL to the standard printer device.
Input: AH = 05h
DL = character
Output: none

Function **09h:**
Print String
Outputs the characters in the print string to the standard output device.
Input: AH = 09h
DS:DX = pointer to the character string ending with '$
output: none

Function **2Ah:**
Get Date
Returns the day of the week, year, month and date.
Input: AH = 2ah
Output: AL = Day of the week (0=SUN, 6=SAT)
CX = Year (1980-2099)
DH = Month (l-12)
DL = Day (1-31)

Function **2Bh:**
Set Date
Sets the date.
Input: AH = 2Bh
CX = year (1980-2099)
DH = month (1-12)
DL = day (l-31)

Output: AL = 00h, if the date is valid
FFh, If the date is not valid

Function 2Ch:
Get lime
Returns the time: hours, minutes, seconds and hundredths of seconds.
Input: AH = 2Ch
output: CH = hours (O-23)
CL = minutes (0-59)
DH = seconds (O-59)
DL = hundredths (O-99)

Function 2Dh:
Se: Time
Sets the time.
Input: AH = 2Dh
CH = Hours (O-23)
DH = Seconds (0-59)
CL = Minutes (O-59)
DL = Hundredths (O-99)
Output: AL = 00h if the time is valid
FFh if the time is not valid

Function 30h:
Get DOS Version Number
Returns the DOS version number.
Input: AH = 30h
output: BX = 0000H
cx = 0000H
AL = major version number
Al I = minor version number

Function 31 h:
Terminate Process and Remain Resident
Terminates the current process and attempts to set the initial allocation block to the memory size in paragraphs.
Input: AH = 31h
AL = return code
DX = memory size in paragraphs
output: none

Function 33h:
Ctrl-break Check
Set or get the state of BREAK (Ctrl-break checking).
Input: AH = 33h
AL = 00h, to request current state
01h, to set the current state
DL = 00h, to set current state OFF
Olh, to set current state ON
Output: DL = The current state (00h=OFF, 01h=ON)

Function 35h:
Get Vector
Obtains the address in an interrupt vector.
Input: AH = 35h

AL = interrupt number
output: ES:BX = pointer to the interrupt handling routine.

Function 36h:
Get Disk Free Space
Returns the disk free space (available clusters, clusters/drive, bytes/sector).
Input: AH = 36h
DL = drive @default, 1=A)
Output: BX = Available clusters
DX = clusters/drive
CX = bytes/sector
AX = FFFFh if the drive in DL is invalid,
otherwise the number of sectors per cluster

Function 39h:
Create Subdirectory (MKDIR)
Creates the specified directory.
Input: AH = 39h
DS:DX = pointer to an ASCIIZ string
Output: AX = error codes if carry flag Is set

Function 3Ah:
Remove Subdirectory (RMDIR)
Removes the specified directory.
Input: AH = 3Ah
DS:DX = pointer to an ASCIIZ string
output: AX = error codes if carry flag Is set

Function 3Bh:
Change the Current Directory(CHDIR)
Changes the current directory to the specified directory.
Input: AH = 3Bh
DS:DX = pointer to an ASCIIZ string
output: AX = error codes if carry flag is set

Function 3Ch:
Create a File (CREAT)
Creates a new file or truncates an old file to zero length in preparation for writing.
Input: AH = 3Ch
DS:DX = pointer to an ASCIIZ string
CX = attribute of the file
output: AX = error codes if carry flag is set
16-bit handle if carry flag not set

Function 3Dh:
Open a File
Opens the specified file.
Input: AH = 3Dh
DS:DX = pointer to an ASCIIZ path name
AL = access Code
output: AX = error codes if carry flag is set
16-bit handle if carry flag not set

Function 3Eh:
Close a File Handle
Closes the specified file handle.
Input: AH = 3Eh
BX = file handle returned by open or create
Output: AX = error codes if carry flag is set
none If carry flag not ret

Function 3Fh:
Read from a File or Device
Transfers the specified number of bytes from a file into a buffer location.
Input: AH = 3Fh
BX = file handle
DS:DX = buffer address
cx = number of bytes to be read
Output: AX = number of bytes read
error codes if carry flag set

Function 40h:
Write to a File or Device
Transfers the specified number of bytes from a buffer into a specified file.
Input: AH = 40h
BX = file handle
DS:DX = address of the ddta to write
CX = number of bytes to be write
Output: AX = number of bytes written
error codes if carry flag set

Function **41h:**
Delete a File from a Specified Directory (UNLINK)
Removes a directory entry associated with a file name.
Input: AH = 41h
DS:DX = address of an ASCIIZ string
Output: AX = error codes if carry flag set
none if carry flag not set

Function 42h:
Move File Read Write Pointer (LSEEK)
Moves the rend/write pointer according to the method specified.
Input: AH = **4211**
CS:DX = distance (offset) to move in bytes
Al. = method of moving (0,1,2)
RX = file handle
Output: AX = error codes if carry flag ret
DX:AX = new pointer location if carry flag not set

Function 47h:
Get Current Directory
Places the full path name (starting from the root director),) of the current directory for the spcciiicd drive in the area pointed to by DS:SI.
Input: AH = 47h
DS:SI = pointer to a 64-byte user memory area
DL = drive number (0=default, 1=A, etc.)
error codes if carry flag set

Output: DS:SI = filled out with full path name from the root if carry is not set
AX = error codes if carry flag is set

Function 48h:
Allocate Memory
Allocates the requested number of paragraphs of memory.
Input: All = 48h
RX = number of paragraphs of memory requested
output: AX:0 = points to the allocated memory block
AX = error codes it carry flag set
BX = size of the largest block of memory available (in paragraphs) if the allocation fails

Function 49h:
Free Allocated Memory
Frees the specified allocated memory.
Input: All = 49h
ES = segment of the block to be returned
output: AX = error codes if carry flag set
none if carry flag not set

Function 4Ch:
Terminate a Process (EXIT)
Terminates the current process and transfers control to the invoking process.
Input: AH = 4Ch
AL = return code
Output: none

Interrupt ***25h: Absolute*** Disk Read

Input: Al. = drive number
CX = number of sectors to read
DX = beginning logical sector number
DS:BX = transfer address
output: If successful CF = 0
If unsuccessful CF = 1 and AX contains error code

Interrupt 26h: Absolute Disk Write

Input: Al. = drive number
CX = number of sectors to read
DX = beginning logical sector number
DS:BX = transfer address
Output: If successful CF = 0
If unsuccessful CF = 1 and AX contains error code

Interrupt 27h: Terminate but Stay ***Resident***

Input: DX = offset of beginning of free space, segment is with respect to PSI?
output: none