Interrupt 21H

Function Number:	**5AH**
Function Name:	Create Temporary File
Purpose:	Creates and opens a new file on disk, assigning a unique name to the file

Data Passed:

AH =	5AH
CX =	Attribute bits
(Native Mode) DS:EDX =	Address of directory specification, followed by a backslash (\)
(8086 Mode) DS:DX =	Address of directory specification, followed by a backslash (\)

Data Returned:

AX =	Error code if carry is set; otherwise a file handle
(Native Mode) DS:EDX =	Unchanged address of directory name, which has the new filename appended
(8086 Mode) DS:DX =	Unchanged address of directory name, which has the new filename appended

Error Codes:

2 -	Invalid filename
3 -	Invalid path
4 -	Too many files open
5 -	Access denied (file or directory can't be read/written to)

Extended Error Codes:

Error Code	Type	Action	Location
2	8	3	2
3	8	3	2
4	1	4	1
5	3	4	1
Critical Error	11	7	2
	11	5	2
	11	1	2
	11	1	1
	9	4	1
	7	4	1
	5	1	1

(See function 59H for Extended Error Code meanings)

Comments: Normal use of this function is to create a temporary scratch file, which is deleted before the application terminates; however, the operating system does not require that the file be deleted.

See function 39H for the format of the name string. Note that the user-supplied string is modified by the call to complete the filename specification.

The new file is opened in Compatibility Mode for read/write access (see function 3DH).

Interrupt 21H

Function Number:	**5BH**
Function Name:	Create New File
Purpose:	Creates and opens a new file on disk, provided that the file doesn't already exist

Data Passed:

AH =	5BH
CX =	Attribute bits
(Native Mode) DS:EDX =	Address of filename
(8086 Mode) DS:DX =	Address of filename

Data Returned:

AX = Error code if carry is set; otherwise a file handle

Error Codes:

2 -	Invalid filename
3 -	Invalid path
4 -	Too many files open
5 -	Access denied (file or directory can't be read/written to)
80 -	File already exists

Extended Error Codes:

Error Code	Type	Action	Location
2	8	3	2
3	8	3	2
4	1	4	1
5	3	4	1
80	12	2	2
Critical Error	11	7	2
	11	5	2
	11	1	2
	11	1	1
	9	4	1
	7	4	1
	5	1	1

(See function 59H for Extended Error Code meanings)

Comments: This call is identical to function 3CH, except that it will fail if the specified filename already exists.

Interrupt 21H

Function Number:	**5CH**
Function Name:	Lock or Unlock Records
Purpose:	Obtain or release exclusive access to a portion of an open file

Data Passed:

```
                    AH = 5CH
                    AL = 0 to lock, 1 to unlock
                    BX = File handle
   (Native Mode)   ECX = Offset of first byte to lock or unlock
   (Native Mode)   EDX = Number of bytes to lock or unlock
   (8086 Mode)   CX:DX = Offset of first byte to lock or unlock
   (8086 Mode)   SI:DI = Number of bytes to lock or unlock
```

Data Returned:

```
                    AX = Error code if carry is set
```

Error Codes:

```
                     1 - Invalid function number, AL not 0 or 1
                     6 - Invalid handle
                    33 - Area locked by someone else
```

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
6	7	4	1
33	10	2	1
Critical Error	11	7	2
	11	5	2
	11	1	2
	11	1	1
	9	4	1
	7	4	1
	5	1	1

(See function 59H for Extended Error Code meanings)

Comments: Locking obtains exclusive access to the specified portion of the file. It makes sense only if the file is opened in a sharing mode that permits write access to this and other tasks.

Interrupt 21H

Function Number:	**62H**
Function Name:	Get PSP Address
Purpose:	To get the PSP address of the currently executing program.
Data Passed:	
	AH = 62H
Data Returned:	
	BX = PSP segment address
Error Codes:	None
Extended Errors:	None

Interrupt 21H

Function Number:	**67H**
Function Name:	Set Handle Count
Purpose:	Allows an application to increase the number of file handles above the 20-handle limit to as much as 255 handles.

Data Passed:

AH = 67H
BX = Number of handles to allow.

Data Returned:

AX = Error code if carry set.

Error Codes:

8 - Insufficient memory

Extended Error Codes:

Error Code	Type	Action	Location
8	1	4	5

Interrupt 21H

Function Number:	**68H**
Function Name:	Commit File to Disk
Purpose:	Flushes any file sectors to disk that are presently retained in cache.

Data Passed:

AH = 68H
BX = File Handle

Data Returned:

AX = Error code if Carry Set

Error Codes:

6 -	Invalid handle
34 -	Wrong disk

Extended Error Codes:

Error Code	Type	Action	Location
6	7	4	1
34	11	7	2

The Extended Services Interrupt

Calls to MOS's Extended Services handler were formerly made through the use of interrupt vector 38H. Although the use of this interrupt is still supported, the recommended vector to use has been changed to D4H. The use of interrupt vector 38H caused compatibility problems in certain environments.

There can sometimes be stiff competition for the use of interrupt vectors by language interpreters, applications programs, device drivers, and TSR utilities. Rather than relying on the contents of the D4H vector as being accurate, the method illustrated in Chapter 13 should be used to obtain a vector to MOS's Extended Services function handler.

Extended Services

Function Number:	**02H**
Function Name:	Get SCB Address
Purpose:	Obtain the address of the MOS System Control Block

Data Passed:

AH = 02H

Data Returned:

```
(Native Mode) ES:EBX = Address of SCB
   (8086 Mode) ES:BX = Address of SCB
```

Error Codes:	None
Extended Errors:	None

Comments: This call is used by system utilities and is not normally needed by applications.

NOTE: Beginning with version 4.10, this call should no longer be used. Instead, use Extended Services functions 26h, 28h, 29h and 2ah to read and write SCB fields. Function 2 is only documented for the sake of utilities and drivers which must operate with versions of MOS prior to 4.10.

Extended Services

Function Number:	**03H**
Function Name:	Get or Change Extended Directory Information
Purpose:	Interrogate or modify bytes 12-21 of a file's directory entry

Data Passed:

	AH =	03H
	AL =	0 to interrogate, 1 to modify
(Native Mode)	DS:EDX =	Address of path and filename
(Native Mode)	ES:EBX =	Pointer to 10-byte buffer area
(8086 Mode)	DS:DX =	Address of path and filename
(8086 Mode)	ES:BX =	Pointer to 10-byte buffer area

Data Returned:

	AX =	Error code if carry is set
	AL =	Permitted access level (0-3) to this file
(Native Mode)	ES:EBX =	Pointer to modified buffer (if AL was 1)
(8086 Mode)	ES:BX =	Pointer to modified buffer (if AL was 1)

Error Codes:

1 -	Invalid function number, AL not 0 or 1
2 -	Invalid filename
3 -	Invalid path
5 -	Access denied (file or directory can't be read/written to)

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
2	8	3	2
3	8	3	2
5	3	4	1
Critical Error	11	7	2
	11	5	2
	11	1	2
	11	1	1
	9	4	1
	7	4	1
	5	1	1

(See function 59H for Extended Error Code meanings)

Comments: The format of the user-supplied buffer is:

0 -	Reserved (set to 00H)
1 -	File class (A-Z or null)
2-5 -	User ID of original file creation
6-7 -	Time of original file creation
8-9 -	Date of original file creation

This function cannot be used to change a file's class, because class affects the enciphering method; directory classes, however, may be changed.

Date and time are in the same format as with function 57H, with byte 6 corresponding to CH and byte 9 to DL. The creation date/time are not the same as the "last-changed" date/time associated with function 57H.

This function is useful for applications that wish to determine the permitted access level to a file.

Extended Services

Function Number: **04H**

Function Name: Get TCB Address

Purpose: Obtain the address of the Task Control Block for a particular task.

Data Passed:

AH = 04H
BX = Task ID (or -1 for current task)

Data Returned:

AX = Error code if carry is set
ES = Pointer to TCB

Error Codes:

87 - Invalid parameter (task ID not in use)

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call is used by system utilities and is not normally needed by applications.

NOTE: Beginning with version 4.10, this call should no longer be used. Instead, use Extended Services functions 27h, 28h, 29h and 2ah to read and write TCB fields. Function 4 is only documented for the sake of utilities and drivers which must operate with versions of MOS prior to 4.10.

Extended Services

Function Number:	**07H**
Function Name:	Wait for Event
Purpose:	Suspend the current task until any one of the specified events occurs

Data Passed:

AH =	07H
AL =	Event type(s) to be monitored
BX =	Number of time ticks if AL bit 1 is set
CX =	Bit map identifying which interrupt(s) if AL bit 2 is set
DH =	Number of ports if AL bit 3 is set
DL =	First serial port in range of DH ports if AL bit 3 is set (0-255)
ES:BX =	pointer to a routine to call to test for wake up - only for the case where AL = 80H

Data Returned:

AX =	Error code if carry is set; otherwise:
AL =	Identifies the event type that occurred
CX =	Identifies which hardware interrupt, if any, awakened the task
DL =	Identifies which serial port (0-255), if any, awakened the task

Error Codes:

8 -	Insufficient memory (to store information for wait)
87 -	Invalid parameter (0 ticks wait time, 0 ports to check, or no bits in interrupt)

Extended Error Codes:

Error Code	Type	Action	Location
8	1	4	5
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: AL contains a "bit map" that identifies what types of events are allowed to re-awaken the task. With the exception of the case where AL = 80H, AL may contain any combination of the following:

01H -	Awaken on the next keystroke
02H -	Awaken after the number of timer ticks given in BX has elapsed (there are 18.2 timer ticks per second)
04H -	Awaken on the hardware interrupt(s) specified in CX
08H -	Awaken if any of the specified serial ports receives an incoming character, or if its status changes

The case where AL = 80H is special in that all other bits within AL are ignored when the high bit is set:

80H -	Awaken based on the return status of a user written poll routine

X may be any combination of the following:

0001H - IRQ0
0002H - IRQ1
0004H - IRQ2
.
.
.
8000H - IRQ15

Upon return to the application, one of the bits in AL will be set to indicate which event type caused the task to wake up. If it is bit 2 or 3, then CX or DL will be set accordingly.

A device driver may execute a WAIT function with a value of 00H in AL. This indicates that the driver itself will take responsibility for establishing and clearing the wait condition by appropriate manipulation of the field TCBWAIT in the task's TCB (see the assembler INCLUDE file MOSTCB.INC).

"Appropriate manipulation" is as follows:

1. To put the task to sleep, set TCBWAIT bit 0 to 1, and bits 1 and 2 to 0; then execute the "wait for event" call with AL=00H;

2. To re-awaken the task from within an IRQ interrupt handler, just set TCBWAIT bit 1 to 1 (do not clear any bits!), and exit from the interrupt.

NOTE: Do not install hardware interrupt handlers in applications software, where they may be "swapped out" of addressable memory. Put them in device drivers instead.

The TCBID field in the TCB is useful for uniquely identifying a task.

Regarding the use of this function when AL = 80H, the only other registers which must be set on entry are ES and BX. A pointer to a user written routine is passed in ES:BX. No data is returned.

This routine is called by MOS, using a far call, to see if the task is ready to be awakened. If so, the called routine must return a zero in the AL register. When this occurs, MOS will re-dispatch the task. If not, the routine must return a non-zero value in AL.

In either case, this routine must return immediately. Since task switching decisions are made based on this routine, it becomes part of MOS's task switching overhead and therefore should be as fast as possible.

A user written poll routine may change any registers except DS and SS. It must also be located within non-switched memory. Therefore, it is best loaded as a device driver during CONFIG.SYS processing.

Extended Services

Function Number:	**10H**
Function Name:	Mode Change
Purpose:	Switch an application to 80386 Native Mode or back into Virtual 8086 Mode

Data Passed:

```
               AH = 10H
               AL = 0 for Virtual Mode, 1 for Native Mode
  (8086 Mode)  CX = Length (in bytes) of NCA buffer (must be
                    at least 1,024 bytes)
  (8086 Mode)  DX = Segment address of NCA buffer
```

Data Returned:

```
               AX = Error code if carry is set
CS,SS,DS,ES,FS,GS = Translated selectors
```

Error Codes:

1 - Invalid function number (no 386 driver)

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1

(See function 59H for Extended Error Code meanings)

Comments: In addition to the mode change, the values in all segment registers are translated (if possible) to address the same memory segments in the new mode. Execution resumes with the instruction following the call.

The NCA address passed in DX is a "segment address" (e.g. address bits 4-19 of a paragraph-aligned scratch area within the first megabyte of RAM). It is only needed when going from VM86 mode to native mode.

Extended Services

Function Number:	**11H (Native Mode Only)**
Function Name:	Allocate Extended Memory
Purpose:	Obtain a block of extended memory

Data Passed:

```
 AH = 11H
EBX = Number of bytes desired
```

Data Returned:

```
 AX = Error code if carry is set
EBX = Number of bytes actually allocated
 ES = Selector for allocated memory
```

Error Codes:

1 - Invalid function number (no 386 driver)
8 - Insufficient memory (to store information for wait)

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
8	1	4	5

(See function 59H for Extended Error Code meanings)

Comments: This function returns a selector which permits read/write access to the allocated memory block. If the block is to be used for code, it will be necessary to use function 13H after loading the code into the memory block.

Memory blocks allocated by this call are not automatically released when the application terminates; you must be careful to do appropriate cleanup yourself.

Extended Services

Function Number: **12H (Native Mode Only)**

Function Name: Deallocate Extended Memory

Purpose: Release a memory block obtained via function 91H

Data Passed:

AH = 12H
ES = Selector

Data Returned:

AX = Error code if carry is set

Error Codes:

1 - Invalid function number (no 386 driver)
6 - Invalid handle (memory handle in ES)

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
6	7	4	1

(See function 59H for Extended Error Code meanings)

Comments: This function must be used for cleanup; allocated memory blocks are not automatically released when an application terminates.

Extended Services

Function Number:	**13H (Native Mode Only)**
Function Name:	Get Alias
Purpose:	Duplicate an extended memory selector to obtain a selector of a different type

Data Passed:

```
AH = 13H
AL = 00H to get a data selector;
     01H to get a stack selector;
     02H to get a code selector
BX = Selector
```

Data Returned:

```
AX = Error code if carry is set, otherwise;
AX = New selector (the AX register will be 0 if
     selector in BX could not be found)
```

Error Codes:

1 - Invalid function number (no 386 driver)

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1

(See function 59H for Extended Error Code meanings)

Comments: This function is normally used to make an allocated segment addressable as a code segment.

The original selector is still valid after this call.

Extended Services

Function Number:	**16H**
Function Name:	Set/Reset IRQ Reservation
Purpose:	Allows a serial driver to control IRQ reservation status. This is useful in implementing function 11 of INT14.

Data Passed:

AH = 16H
AL = 0 to clear the IRQ reservation
AL = 1 to set the IRQ reservation
CX = the RQ number

Data Returned:

AX = 0 if successful
AX = 1 if the IRQ is currently reserved by some task

Errors codes: (returned in AX)

Comments: This function is provided for use within function 11 of INT 14 (Disable Port). Once this INT 14 logic has determined that a port may be disabled, it should call Extended Services function 16H to clear the IRQ reservation.

When a serial driver is counteracting a port disable, it should use this call again to re-establish the IRQ reservation. Calls to INT 14 functions 0, 4 and 13 for a disabled port should automatically re-enable that port.

Extended Services

Function Number:	**19H**
Function Name:	Return Task ID
Purpose:	Returns the Task ID of the current task

Data Passed:

AH = 19H

Data Returned:

BX = Task ID value

Error Codes:	None
Extended Error:	None

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

Extended Services

Function Number:	**1AH**
Function Name:	Set/Read/Exchange Priority
Purpose:	Allows a task's priority to be read, set or exchanged with a new value

Data Passed:

```
AH = 1AH
AL = 00  - read current priority
AL = 01  - set a new priority
AL = 02  - exchange priority
BX = Task ID (-1 for current task)
CL = Priority value
```

Data Returned:

```
AX = Error Code if carry is set
CL = Returned priority value
```

Error Codes:

```
 1 - Invalid function
87 - Invalid parameter
```

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

Extended Services

Function Number: **1BH**

Function Name:	Read/Set/Exchange Time Slice
Purpose:	Allows a task's time slice to be read, set, or exchanged with a new value.

Data Passed:

```
AH = 1BH
AL = 00  - read current slice
AL = 01  - set a new slice
AL = 02  - exchange slice
BX = Task ID (-1 for current task)
CL = Slice value
```

Data Returned:

```
AX = Error Code if carry is set
CL = Returned slice value
```

Error Codes:

```
 1 - Invalid function
87 - Invalid parameter
```

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

Extended Services

Function Number:	**1CH**
Function Name:	Clear/Set/Read Keyboard Mode
Purpose:	Allows a task's keyboard mode to be read or changed.

Data Passed:

```
AH = 1CH
AL = 0 to set NODIS mode
AL = 1 to set DIS mode
AL = 2 to return current state in CL
BX = Task ID (-1 for current task)
```

Data Returned:

```
AX = Error Code if carry is set
CL = Current keyboard state
```

Error Codes:

87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

Extended Services

Function Number:	**1DH**
Function Name:	Return Current Program Name
Purpose:	Allows the name of the currently executing program within a task to be read.

Data Passed:

AH =	1DH
BX =	Task ID (-1 for current task)
ES:DI =	Pointer to an 11 byte buffer into which the name shall be placed.

Data Returned:

AX = Error Code if carry is set

The program name is copied into the buffer provided.

Error Codes: 87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs. The format of the program name within the buffer is as follows:

bytes 0 - 7	the program's file name
bytes 8 - 11	the program's extension

Extended Services

Function Number:	**1EH**
Function Name:	Return Current User Name and Security Class
Purpose:	Allows the name of the current user and their security class to be read.

Data Passed:

AH = 1EH
BX = Task ID (-1 for current task)
ES:DI = Pointer to a 4 byte buffer into which the name shall be placed.

Data Returned:

AX = Error Code if carry is set
CL = The security class

The program name is copied into the buffer provided.

Error Codes: 87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

Extended Services

Function Number:	**1FH**
Function Name:	Return Task Partition Information
Purpose:	Allows the beginning and ending address of a task to be read.

Data Passed:

AH = 1FH
BX = Task ID (-1 for current task)

Data Returned:

AX = Error Code if carry is set
CX = The beginning address
DX = The ending address

Error Codes: 87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

The addresses provided are segment addresses. The ending address, returned in DX, is the first segment beyond the task. For example, if the beginning address is returned as 0800 and the ending as A000, the task is occupying addresses 08000 through 9FFFF inclusive.

Extended Services

Function Number:	**20H**
Function Name:	Return Port and Baud Rate Information
Purpose:	Allows a task's port and baud rate settings to be read

Data Passed:

AH = 20H
BX = Task ID (-1 for current task)

Data Returned:

AX = Error Code if carry is set
CX = Port number (0 if none)
DI:SI = Baud rate (n/a if no port)

Error Codes:

87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

Extended Services

Function Number: **21H**

Function Name: Remove a Task

Purpose: Removes the task whose ID is in BX.

Data Passed:

AH = 21H
BX = Task ID (-1 for current task)

Data Returned:

AX = Error Code if carry is set
AX = Percentage of SMP used. See below

Error Codes:

5 - Access denied
87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
5	3	4	1
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

AX is interpreted as: AL = ASCII character for the tens digit of the percentage of SMP used. AH = the ASCII character for the ones digit of the percentage used. DS and SI are changed.

Extended Services

Function Number:	**22H**
Function Name:	Add a Task
Purpose:	Adds a new task to the system

Data Passed:

```
   AH = 22H
DS:SI = Pointer to a data structure as outlined
        below
```

Data Returned:

```
   AX = Error Code if carry is set
   ES = Segment address of the new task's TCB
        data structure

        Additional data is returned within the
        data structure as outlined below
```

Error Codes:

```
 1 - Invalid function number
 8 - Insufficient memory
11 - Invalid format
18 - Insufficient SMP
31 - General failure
85 - Already allocated
87 - Invalid parameter
```

Extended Error Codes:

Error Code	Type	Action	Location
1	7	4	1
8	1	4	5
11	9	4	1
18	8	3	2
31	5	1	1
85	12	6	3
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs.

When calling this function to add a task, DS:SI must point to a structure of the type shown below. The first seven fields must be filled in with the appropriate data before making the call. Upon return from a successful call, the remaining four fields will hold status information about the new task.

```
add_data struc
tsize         dw 0            ; task size
tid           dw 0            ; task id
tclass        db ' '          ; task class
tbatch        dd 0            ; task startup batchfile
tdriver       dd 0            ; task terminal driver
tport         dw 0            ; task port
tbaud         dd 0            ; task baud rate
tmemtot       dd 0            ; total ext mem  (RETURN)
tmemalc       dd 0            ; ext mem allocated  (RETURN)
tsmpal        dw 0            ; task smp allocate  (RETURN)
tsmpsiz       dw 0            ; task smp size  (RETURN)
tpercent      dw 0            ; task percent heap  (RETURN)
tres          db 3 dup (0)    ; reserved
add_data ends
```

ENTRY DATA

TSIZE - the number of K of memory for the new task.

TID - The ID number for new task. If a specific value is supplied, it must be within the range 1-99. Error 87 will result if the chosen value is the same as the ID number of an existing task. When this field is 0, MOS will assign next free number. The number assigned may then be found within the TCBID field of the new task's TCB.

TCLASS - Use a blank (20h) if no security is to be applied. Otherwise, use a letter in the range "A" to "Z" for the desired partition class.

TBATCH - This field must hold a far pointer to an asciiz string which is the name of the startup batch file for the new task. Only the batch file's name should be specified. Do not include an drive letter, path, or file extension. The file specified must exist within the root directory of the current drive. If no startup batch file is to be used, a pointer which points to a binary zero must be supplied.

TDRIVER - This field must hold a far pointer to the function handler within the terminal driver. This pointer is obtained by opening the terminal driver, setting it to raw mode and reading 4 bytes. To add a background task, set this field to 0. Refer to Chapter 13, Programming Techniques, for a complete explanation with sample source code.

TPORT - To add a background task, this value should be 0. When adding a workstation task, the treatment of this field involves a set of tests which are beyond the scope of this section. Refer to Chapter 13, Programming Techniques, for a complete explanation with sample source code.

TBAUD - To add a background task, set this field to 0. When adding a serial terminal workstation type of task, a valid serial baud rate must be specified. The transmission speed is the actual speed stored low-order/high-order. In the case of a workstation such as VNA or SunRiver®, where a baud rate is not relevant, set this field to 0.

RETURN DATA

TMEMTOT - When paging-capable memory management support exists, this field holds the total number 4K pages of extended memory. A zero in this field means that no paging-capable memory management support exists. Note that this is a doubleword field. Refer to Chapter 13, Programming Techniques, for a complete explanation with sample source code.

TMEMALC - When paging-capable memory management support exists, this field holds the number 4K pages of extended memory which are currently in use. A zero in this field means that no paging-capable memory management support exists. Note that this is a doubleword field. Refer to Chapter 13, Programming Techniques, for a complete explanation with sample source code.

TSMPAL - This field holds the number of paragraphs of the SMP which are presently allocated.

TSMPSIZ - This field holds the total number of paragraphs in the SMP.

TPERCENT - This field holds the percentage of the SMP which is currently in use. It is interpreted as: lsb = ASCII character for the tens digit of the percentage of SMP used and msb = the ASCII character for the ones digit of the percentage used.

Extended Services

Function Number:	**23H**
Function Name:	Change Terminal Driver
Purpose:	Allows a serial terminal workstation driver to be changed.

Data Passed:

```
   AH = 23H
   BX = Task ID (-1 for current task)
DS:SI = Pointer to the entry point of the new DDT
```

Data Returned:

```
   AX = Error Code if carry is set
```

Error Codes:

87 - Invalid parameter

Extended Error Codes:

Error Code	Type	Action	Location
87	9	3	3

(See function 59H for Extended Error Code meanings)

Comments: This call will be available in version 4.00 of PC-MOS. It is provided to simplify the interface between MOS and user written application programs. See Chapter 13 for details on obtaining the pointer to the entry point of a DDT driver.

Extended Services

Function Number:	**25H**
Function Name:	Identify Device Driver Location
Purpose:	Enables a device driver to detect whether it has been loaded into the SMP or into task memory.

Data Passed:

AH = 25H
DX = the driver's CS segment value

Data Returned:

AX = 0 if driver is not within the SMP
AX = segment address of SMP if driver is within the SMP

Errors codes: None

Comments: New to 4.10, this call enables a device driver to determine where it has been loaded. If a driver cannot function properly if loaded into task memory (using ADDDEV's task specific load option), the driver could use this function call to detect this condition and fail the load process.

Extended Services

Function Number: **26H**

Function Name: Get SCB Address Segment/Selector

Purpose: Obtains a segment/selector that corresponds to the System Control Block. This segment/selector must be used in all transactions with the SCB.

Data Passed:

AH = 26H

Data Returned:

DX = Segment/Selector to the SCB.

Errors codes: None

Comments: For PC-MOS versions 4.10 and above, this is now the approved method of obtaining the data for the System Control Block. As MOS is improved, the SCB is now becoming a moving target. This call in conjunction with functions 28, 29 and 2A will be the only guaranteed method of reading and writing the SCB.

Extended Services

Function Number:	**27H**
Function Name:	Get TCB Address Segment/Selector
Purpose:	Obtains a segment/selector to the desired Task Control block. This segment/selector must be used in all transactions with the selected TCB.

Data Passed:

AH = 27H
BX = Task ID. (-1 = Current task)

Data Returned:

AX = Error Code if carry is set.
DX = Segment/Selector corresponding to TCB.

Error codes:

87 - Invalid Parameter. (Task ID)

Comments: For PC-MOS versions 4.10 and above, this is now the approved method of obtaining the data for the TCB. As MOS is improved, the TCB is now becoming a moving target. This call in conjunction with functions 28, 29 and 2A will be the only guaranteed method of reading and writing the TCB.

Extended Services

Function Number:	**28H**
Function Name:	Read Control Block Data
Purpose:	Gets a copy of the control block corresponding to the supplied segment/selector. This segment/selector must have been obtained through function 26H or 27H. (Get SCB Segment/Selector or Get TCB Segment/Selector)

Data Passed:

```
   AH = 28H
   BX = Offset into control block to start reading.
   CX = Number of bytes to read.
   DX = Segment/Selector of control block. (Ob-
        tained from function 26H or 27H)
ES:DI = Location in which to place the data read
        from the control block.
```

Data Returned:

```
   AX = Error Code if carry is set.
```

Error Codes:

05 - Access Denied. (Invalid Segment/Selector)

Comments: New to 4.10, this call should be used to read data values from MOS's SCB and TCB data structures. For MOS versions 4.10 and above, Extended Services 02H and 04H should not be used.

Extended Services

Function Number:	**29H**
Function Name:	Write Control Block Data
Purpose:	Copies data to the control block corresponding to the supplied segment/selector. This segment/selector must have been obtained through function 26H or 27H. (Get SCB Segment/Selector or Get TCB Segment/Selector)

Data Passed:

AH =	29H
BX =	Offset into control block to start writing.
CX =	Number of bytes to write.
DX =	Segment/Selector of control block. (Obtained from function 26H or 27H)
DS:SI =	Location in which to get the data to be written to the control block.

Data Returned:

AX =	Error Code if carry is set.

Error Codes:

05 -	Access Denied. (Invalid Segment/Selector)

Comments: New to 4.10, this call should be used to write data values into MOS's SCB and TCB data structures. For MOS versions 4.10 and above, Extended Services 02H and 04H should not be used.

Extended Services

Function Number:	**2AH**
Function Name:	Swap Control Block Data
Purpose:	Swaps data in a local buffer with the Control Block selected. The segment/selector used must have been obtained through function 26H or 27H. (Get SCB Segment/Selector or Get TCB Segment/Selector)

Data Passed:

AH =	2AH
BX =	Offset into control block to start the swap.
CX =	Number of bytes to be swapped.
DX =	Segment/Selector of control block. (Obtained from function 26H or 27H)
DS:SI =	Location in which to get the data to be written to the control block and where the control block data will be written.

Data Returned:

AX = Error Code if carry is set.

Error Codes:

05 - Access Denied. (Invalid Segment/Selector)

Comments: New to 4.10, this function is provided so that items such as interrupt handler addresses (vectors), memory addresses and other critical data may be swapped in a safe manner. Interrupts are disabled during the swap to ensure that the value placed does not inadvertently get changed by another task before the swap is complete.

Extended Services

Function Number:	**2CH**
Function Name:	Get/Set Spooler Parameters
Purpose:	Enables manipulation of the print spooler's operating parameters.

Data Passed:

```
AH = 2CH
AL = 00H - set spooler time out
     CX = time in seconds
AL = 01H - get spooler time out
     CX = time in seconds
AL = 02H - set spooler parameters
     CL = disposition (d,s,h,i,n)
     CH = priority   (0 - 9)
     SI = class      (a - z)
AL = 03H - set spooler parameters
     CL = disposition (d,s,h,i,n)
     CH = priority   (0 - 9)
     SI = class      (a - z)
BX = task id or -1 for current task
DX = lpt number
```

Data Returned:

AX = Error Code if carry is set.

Error Codes:

1 - Invalid function number

Comments: This function is new to version 4.10 of MOS.

Extended Services

Function Number:	**2DH**
Function Name:	Return Maximum Task Size
Purpose:	To determine the largest possible task size.

Data Passed:

AH = 2DH

Data Returned:

DX = Maximum task size in paragraphs
BX = Starting address of task space

Error Codes: None

Comments: New to 4.10, this function returns the maximum possible task size in paragraphs. Note that this function does not verify that there is enough extended memory available to allocate to such a task - it only indicates the largest potential task size. To convert the DX return value to kilobytes, divide by 64.

This statistic does not apply in the non-memory managed configuration.

Interrupt 17H

Function Number:	**3H**
Function Name:	Print String
Purpose:	Allows a string of data to be printed with one 17H call.

Data Passed:

```
   AH = 3H
   DX = printer port number
   CX = characters in string
DS:SI = Pointer to the string
```

Data Returned:

```
   AH = printer status as defined by the BIOS for
        function 1
   CX = number of characters printed
```

Error Codes:	As defined by the BIOS
Extended Errors:	None

Comments: This call provides an efficient way to output data to a printer. Any redirection set up with the MOS ROUTE command or the SPOOL.COM utility will be effective through MOS's INT 17H services.

With regards to the printer status returned in the AH register, MOS will provide the most accurate information available. Note that certain serial workstation terminals do not provide printer status feedback.