Last update: april 20th 2006

Level 3 Subprograms

These page contains a description of the various level 3 subprograms as supported by different mass storage devices like floppy disk controllers, harddisk controllers, SCSI controllers and IDE controllers. Level 3 subprograms can be accessed from an assembler programs or a TI-Basic or Extanded-Basic program.

back to documentation index

Subprogram >0 - Open
Subprogram >1 - Close
Subprogram >2 - Read
Subprogram >3 - Write
Subprogram >4 - Rewind
Subprogram >5 - Load
Subprogram >6 - Save
Subprogram >7 - Delete
Subprogram >8 - Scratch record
Subprogram >9 - Status

Before a level 3 subprogram can be used an peripheral acces block must be build. If called from basic, the basic interpreter does this for you.

A peripheral acces block (PAB) is defined as follows: (most significant bit is bit 0, least significant bit is bit 7)

```
  Bit    0          7 0          7
  Byte  +--------------------------+
     0  | I/O Opcode | Flag/Status|
        +--------------------------+
     2  | Data buffer address      |
        +------------+-------------+
     4  | Rec.length | Char Count  |
        +------------+-------------+
     6  | Record number            |
        +------------+-------------+
     8  | Scr Offset | Name length |
        +------------+-------------+
    10+ | File descriptor          |
        +------------+-------------+
```

Byte 0 - I/O Opcode:
The opcode for the current I/O call (0 through 9)

Byte 1 - Flag/Status:
All information the system needs about the file like type, mode of operation and data type:

Bit 0-2: Returned error code:

Error Code	Meaning
>00 (0)	No error
>20 (1)	Device is write protected
>40 (2)	Bad open attribute
>60 (3)	Illegal operation

```
             >80 (4)    Out of buffer space
             >A0 (5)    Read past end of file
             >C0 (6)    Device error
             >E0 (7)    File error

          (If error code is 0 and bit 2 of >837C is set, this means bad device
          name)

    Bit 3  : Record type: 0=Fixed,   1=Variable
    Bit 4  : Data type  : 0=Display, 1=Internal
    Bit 5-6: Mode of operation: 00=Update, 01=Output,
                                10=Input,  11=Append

    Bit 7  : File type  : 0=Sequential, 1=Relative

Byte 2-3 - Data buffer address:
    The address of the data buffer that the data must be written to or read from
    in VDP memory.

Byte 4 - Logical record length:
    The logical record length for fixed length records or the maximum record
    length for a variable length record.

Byte 5 - Character count:
    The number of bytes to be transferred for a write op-code or the number of
    bytes actually read after a read op-code.

Byte 6-7 - Record number:
    (Only required for a relative record type file) The record number on witch
    the current I/O operation is performed. The most significant bit is ignored,
    so the maximum record number can be from o through 32767.

Byte 8 -  Screen offset:
    The offset of the screen characters with respect to their normal ASCII
    value. This is only used by the cassette interface, wich must put prompts on
    the screen.

Byte 9 - Name length:
    The length of the file descriptor which starts at byte 10.

Byte 10+ - The devicename and if required the directoryname and/or
    filename (and options). The maximum devicename length is 7 but is usually 4
    characters, the maximum directoryname and filename length is 10 characters.
    The overall length of a devicename has a maximum of 40 characters (on most
    devices).




Subprogram 0 - Open


(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The OPEN operation must be performed before any data-transfer operation (except
with LOAD or SAVE) can be performed. The file remains open until a CLOSE
operation is performed. The mode of operation must be given in byte 1
```

(Flag/Status) of the PAB. Changing the mode of operation after an OPEN can cause unpredictable results.

If a record length of 0 is given in byte 4 (logical record length) of the PAB, the default record length (which depends on the peripheral) is returned and used to create a non-existend file or the actual record length of an existend file is returned in byte 4. If a non-zero record length is given, it is used after being checked for correctness with the given peripheral, file type and data type:

Basic example(s):

```
OPEN #1:"DSK1.FILENAAM", INPUT, DISPLAY, VARIABLE

OPEN #2:"SCS1.DIRECTORY.FILENAAM", OUTPUT, INTERNAL, FIXED 46

OPEN #3:"IDE2.DIRECTORY.DIRECTORY.", INPUT, INTERNAL, FIXED

OPEN #4:"RS232/2.BA=9600.DA=8.PA=N", OUTPUT, DISPLAY, VARIABLE 80
```

Subprogram 1 - Close

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The close operation closes a file. If the file was opened in OUTPUT or APPEND mode, an End of File (EOF) record is written to the device or file before closing.

Basic example(s):

```
CLOSE #1
```

Subprogram 2 - Read

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The READ operation reads a record from the selected device and copies the bytes into the buffer specified in byte 2-3 (data buffer address) of the PAB. The size of the buffer is specified in byte 4 (logical record length) of the PB and the actual number of bytes stored in the buffer is returned in byte 5 (character count) of the PAB. If the length of the input record exceeds the buffer size, the remaining characters are discarded.

Basic example(s):

```
    INPUT #1:A$,A,B,C
```

Subprogram 3 - Write

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The write operation writes a record from the buffer specified in bytes 2-3 (data buffer address) of the PAB. The number of bytes to be written is specified in byte 5 (character count) of the PAB.

Basic example(s):

```
    WRITE #1:"Writing a record with three numbers",1,2,3
```

Subprogram 4 - Rewind

(DSK, WDS, SCS, IDE, HDX)

The RESTORE/REWIND operation repositions the file read/write pointer to the beginning of the file or, in the case of a relative record file, to the record specified in bytes 6-7 (record number of the PAB.

The RESTORE/REWIND operation can only be used if the file was opened in INPUT or UPDATE mode. For relative record files, a RESTORE can be simulated by specifying the record at which the file is to be positioned in bytes 6-7 (record number) of the PAB. The next operation (READ, WRITE or SCRATCH RECORD) then uses the indicated record.

Basic example(s):

```
    OPEN #1:"HDS1.DATAFILES.FILE1",UPDATE,VARIABLE,FIXED 128,RELATIVE
    RESTORE #1,REC(12)
    PRINT #1:"ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    RESTORE #1,REC(12)
    INPUT #1:A$
    PRINT A$
    PRINT #1,REC(12):"abcdefghijklmnopqrstuvwxyz"
    INPUT #1,REC(12):A$
    PRINT A$
    CLOSE #1
```

Subprogram 5 - Load

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The LOAD operation loads a memory image of a file from an external device or file into VDP RAM. The LOAD operation is used without a previous OPEN operation. Note that the LOAD operation requires as much buffer space in VDP RAM as the file occupies on the device.

Basic example(s):

```
OLD HDS1.BASICPROG.THISONE
```

Subprogram 6 - Save

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The SAVE operation writes a file from VDP RAM to a device. The SAVE operation is used without a previous OPEN operation. Note that the SAVE oparation copies the entire memory image from the buffer in VDP RAM to the device.

Basic example(s):

```
SAVE IDE1.BASICPROG.NEWPROG
```

Subprogram 7 - Delete

(DSK, WDS, SCS, IDE, HDX)

The DELETE operation deletes a file from the peripheral. The operation also performs a CLOSE (is this true for all devices?)

Basic example(s):

```
DELETE "SCS2.BASICPROG.OLDPROG"
```

Subprogram 8 - Scratch record

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The SCRATCH RECORD operation removes a record specified in bytes 6-7 (record number) of the PAB from the specified relative record file. This operation causes an error for peripherals opened as sequential files.

Basic example(s):

```
(does not exist)
```

Subprogram 9 - Status

(DSK, WDS, SCS, IDE, HDX, RS232, PIO)

The STATUS is returned in byte 8 (screen offset) of the PAB. The status byte returns the status of a peripheral and can be examined at any time. All of the bits have meaning if the file is currently open. Bits 6 and 7 have meaning for files that are currently open, otherwise they are reset.

Bit	Description
0	File existance: 0=File exists, 1=File does not exist On some devices like PIO and RS232 this bit is never set since any file could exist
1	File protection: 0=File is not protected, 1=File is protected
2	Reserved
3	File data type: 0=DISPLAY, 1=INTERNAL
4	File type: 0=data file, 1=program file
5	Record type: 0=FIXED, 1=VARIABLE
6	1=The file is at the physical end of the peripheral and no more data can be written (disk full).
7	1=The file is at the end of its previously created contents (End of File). You can still write to the file if opened in APPEND, OUTPUT or UPDATE mode, but any attempt to read data from the file will causes an error.

Basic example(s):

```
100 OPEN #1: "DSK1.DATAFILE",INPUT,DISPLAY,VARIABLE 80
110 IF EOF(1) THEN 120
120 READ #1:A$
130 GOTO 110
```

```
    120 CLOSE #1
```

Ti99-geek