GPL INTERFACE SPECIFICATION

FOR THE

99/4 DISK PERIPHERAL

Consumer Group
Mail Station 5890
2301 N. University
Lubbock, Texas 79414

TEXAS INSTRUMENTS
INCORPORATED

Date: March 28, 1983
Version 2.0

TABLE of CONTENTS

Paragraph Title

LIST of TABLES

SECTION 1

INTRODUCTION

The information contained in this document gives a complete specification of the interface between the 99/4 Disk Peripheral and the GPL interpreter.

NOTE

Throughout this document hexadecimal numbers are indicated by either a preceding 0 or a preceding >. Therefore the numbers 010 and >10 are the same as 16 decimal.

The items in transfer blocks which are enclosed in brackets {} are items that are returned by the subprogram.

SECTION 2

APPLICABLE DOCUMENTS

File Management Specification for the TI-99/4 Home Computer
(Version 2.5, 25 February 1983)

Home Computer BASIC Language Specification
(Revision 4.1, 12 April 1979)

Home Computer Disk Peripheral Hardware Specification

Functional Specification for the 99/4 Disk Peripheral
(Version 3.0, 28 March 1983)

Software Specification for the 99/4 Disk Peripheral
(Version 2.0, Revised 28 March 1983)

SECTION 3

DISK DSR LEVEL CONCEPT

The disk DSR has been developed as a three level software package, each level defining distinct options that can be used on higher levels. This section will give a brief overview of the levels used and of the features built-in to each level.

The three levels used are:

Level 1 - Definition of the basic disk functions like sector read/write, head control, drive selection, and track formatting.

Level 2 - Definition of the "file" concept. Each file is addressable by its name and an offset of a 256-byte block relative to the beginning of the file.

Level 3 - Extension of the file concept to the level given in the file management specifications. Introduction of the logical fixed or variable length records, relative record or sequential files.

The following sections will each describe a level and the related subprogram calls to it.

3.1 Level 1 Subroutines

The lowest routines in the disk DSR are called Level 1 Subroutines. These routines make the higher levels independent of the physical disk medium, e.g. changing the disk software for a double density disk would only involve changing the routines on this level, as long as the physical sector size remains 256 bytes.

The following routines are available on this level:

Sector read/write

Format Disk

The following sections will contain a description of these routines and their call requirements. All parameters will be transferred through the FAC block in CPU RAM. This block is located in CPU RAM starting at relative location 04A (currently >834A).

3.1.1 Sector READ/WRITE - SUBPROGRAM 010.

The transfer block for this subprogram is:

004A	{Sector Number}	004B
004C	Unit # | READ/WRITE	004D
004E	VDP Buffer start address	004F
0050	Sector Number	0051

The meaning of each entry is:

Sector Number - Number of the sector to be written or read. Sectors are addressed as logical sectors (0 - 359 for a standard single density mini-floppy) rather than as a track and sector number, which would require a knowledge of the physical layout of the floppy disk. The sector number has to be given in CPU RAM locations 050-051, and will be returned in CPU RAM locations 04A-04B.

Unit # - Indicates the disk drive on which the operation is to be performed. This entry has to be either a 1, 2, or 3.

READ/WRITE - Indicates the direction of data-flow.
0 = WRITE
<> 0 = READ

VDP Buffer start address - Indicates start of VDP buffer for data-transfer. The number of bytes transferred will always be 256.

Error codes will be returned in CPU location 050.

3.1.2 Disk Formatting - SUBPROGRAM 011.

The transfer block for this subprogram is:

004A	# of sectors/disk	004B
004C	DSR Ver | Unit # | # of tracks	004D
004E	VDP Buffer start address	004F
0050	Density | # of Sides	0051

The meaning of each entry is:

# of sectors/disk - Is returned by the routine to provide compatibility between the current controller version and future (double density or SA200) versions.

DSR Version - This is the MSNibble.
0 indicates the format requires nothing special and can be done on any version of the DSR.
1 indicates the format requires the 2nd version of the DSR for 1 of 2 reasons. It may be because a double sided format is requested or it may be because a # of tracks other than 35 or 40 is requested.
2 indicates the format requires features that are not available on the 1st or 2nd DSR. (Density and perhaps double tracking if it is available on the next DSR.)

Unit # - Indicates the disk drive on which the operation is to be performed. This entry has to be either a 1, 2, or 3.
This is the LSNibble.

# of tracks - Indicates the number of tracks to be formatted. In the current version this entry has to be either 35 or 40!!! Upon return, this entry contains the number of sectors/track.

VDP Buffer start address - Indicates start address of the VDP buffer that can be used by the disk controller to write tracks.

Density -

# of Sides - Indicates the number of sides to format.

This routine will format the entire disk on the given unit unless the disk in the unit has been hardware write protected. It can use any VDP memory, starting at the location given in the transfer block. The amount of memory used depends on the disk format. For the current single density format, the buffer memory used is a nominal 3125 bytes. This can vary with the disk motor speed to a maximum of 3300 bytes. To be compatible with double density versions of the disk controller, the minimum buffer size must be 8K bytes.

Error codes are returned in CPU location 050.

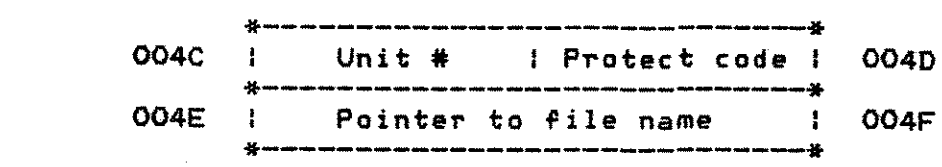
3.2 Level 2 Subroutines

The Level 2 Subroutines are those routines that use the concept "file" rather than "logical sector number". Notice that the file concept on this level is limited to an abstract type of file which has no properties such as "program file" or "data file". A file on this level is merely a collection of data, stored in logical blocks of 256 bytes each.

The logical blocks on this level are accessed by filename and logical block offset. This offset starts with block 0 and ends with block N-1 for a file with a length of N blocks.

3.2.1 Modify File Protection - SUBPROGRAM 012.

The transfer block for this subprogram is:



The protect bit for the indicated file will be set or reset according to the information given in CPU location 04D:

0 - Reset the file protect bit. The file is no longer protected against modification/deletion.

OFF - Set the file protect bit. Disallow SAVE and OPEN for OUTPUT, APPEND, or UPDATE mode.

3.2.2 File Rename Routine - SUBPROGRAM 013.

The transfer block for this subprogram is:

004C	Unit # | unused	004D
004E	Pointer to new name	004F
0050	Pointer to old name	0051

Both pointers, located at 04E and 050 in CPU RAM, point to the VDP location of the first character of a file-name. The first pointer points to the new name, the second one to the original filename. Each name is left adjusted in a 10-character field, filled with spaces. Each name is located in VDP RAM and has to be a legal filename. No checks are being made to ensure legality of the name.

Since the rename has to be done on the same disk, only one unit # entry is required. This unit # is located in CPU RAM location 04E.

Error codes are returned in the standard error byte at CPU location 050. The error codes returned are identical to the standard file management error codes, i.e. only the upper thr bits of the error byte are significant.

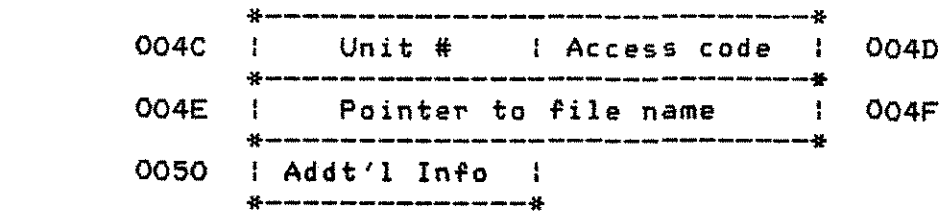
3.3 Direct File Access Routines

The direct file access routines can be used for accessing disk files without paying attention to the type of disk file (PROGRAM or DATA). The level of access is equivalent to the Level 2 disk software, which means that access is performed on the basis of straight AUs. However, Level 3 information can be passed at file open time.

Since the input and output direct access subprograms can be used together to copy files, the user has to be very careful with the information returned by the input file subprogram, since some of this information may be used by the output file subprogram.

3.3.1 Access Direct Input File - SUBPROGRAM 014.

The transfer block for this subprogram is:



The meaning of each entry is:

Unit # - Indicates the disk drive on which the operation is to be performed. This entry has to be either a 1, 2, or 3.

Access code - An access code is used to indicate which function is to be performed, since this subprogram combines multiple functions. The following codes are used:

0 Transfer file parameters. This will transfer Level 2 parameters to the additional information area (six bytes). It also passes the number of AUs allocated for the file.

N When N is not equal to 0, this indicates the number of AUs to be read from the given file, starting at the AU indicated in the additional information block.
After the READ is complete, this entry contains the actual number of AUs read. If all AUs have been read, this entry will be 0.

Pointer to file name - Contains a pointer to the first character of a 10-character filename, possibly padded to the right with spaces. This filename is NOT checked by the disk software.

Additional Info - Points to a 10-byte location in CPU RAM containing additional information for direct disk access:

Table 3-1 Additional Information Block

X	VDP Buffer Start Address	
X+2	# of first AU	
X+4	Status Flags	# records/AU
X+6	EOF offset	Log. Rec. Size
X+8	# of Level 3 records allocated	

The VDP Buffer start address indicates where the information read from the disk can be stored. The buffer has to be able to store at least N * 256 bytes, in which N is the access code.

The # of first AU entry indicates the AU number at which the read should begin. If the access code = 0 (parameter passing), the total number of AUs allocated for the file will be returned.

The remaining 6 bytes are explained in the Software Specification for the 99/4 Disk Peripheral. The user should be very careful when changing these bytes, since they directly affect Level 3 operation. If the information in these 6 bytes is not modified consistently, unpredictable results may occur.

Error codes are returned at location 050 in CPU RAM.

3.3.2 Access Direct Output File - SUBPROGRAM 015.

The transfer block for this subprogram is:

004C	Unit #	Access code	004D
004E	Pointer to file name		004F
0050	Addt'l Info		

The meaning of each entry is:

Unit # - Indicates the disk drive on which the operation is to be performed. This entry has to be either a 1, 2, or 3.

Access code - An access code is used to indicate which function is to be performed, since this subprogram combines multiple functions. The following codes are used:

0 Create file and copy Level 3 parameters from additional information area.

N When N is not equal to 0, indicates the number of AUs to be written to the given file, starting at the AU indicated in the additional information block.

Pointer to file name - Contains a pointer to the first character of a 10-character filename, possibly padded to the right with spaces. This filename is NOT checked by the disk software.

Additional Info - Points to a 10-byte location in CPU RAM containing additional information for direct disk access:

Table 3-2 Additional Information Block

X	VDP Buffer Start Address	
X+2	# of first AU	
X+4	Status Flags	# records/AU
X+6	EOF offset	Log. Rec. Size
X+8	# of Level 3 records allocated	

The VDP Buffer start address indicates where the information read from the disk can be stored. The buffer has to be able to store at least N * 256 bytes, in which N is the access code.

The # of first AU entry indicates the AU number at which the read should begin. If the access code = 0 (parameter

passing), the total number of AUs to be allocated for the file has to be indicated.

The remaining 6 bytes are explained in the Software Specification for the 99/4 Disk Peripheral. The user should be very careful when changing these bytes, since they directly affect Level 3 operation. If the information in these 6 bytes is not modified consistently, unpredictable results may occur.

Error codes are returned at location 050 in CPU RAM.

3.4 Buffer Allocation Routine - SUBPROGRAM 016

The argument for this subprogram is the number of file buffers to be allocated. This argument is given in FAC+2 (CPU location 04C).

The effect of this routine is that an attempt is made to allocate enough VDP space for disk usage to facilitate the simultaneous opening of the given number of files. This number has to be between 1 and 16.

The disk software automatically relocates all buffer are that have been linked in the following manner:

Byte 1 - Validation code

Byte 2/3 - Top of memory before allocation of this buffer

Byte 4 - High byte of CRU address for given buffer area. For programs this byte is 0.

The linkage to the first buffer area is made through the current top of memory, given in CPU location 070 (currently >8370).

The top of memory is also automatically updated after successful completion of this subprogram.

A check is made that the current request leaves at least 0800 bytes of VDP space for screen and data storage. If this is not the case, or if the total number of buffers requested is 0 or >16, the request is ignored and an error code will be indicated in CPU location 050 (currently >8350).

Successful completion is indicated by a 0 byte in CPU location 050. A nonzero byte in CPU location 050 indicates unsuccessful completion.