Global File Converters Manual Version 8.1

MFCMV8.1/60

TABLE OF CONTENTS

APPENDICES

1. Introduction

This manual describes the various programs that allow data and text files to be transferred between Global System Manager and various host operating systems. It also describes the interfaces available within Global System Manager to perform functions under the host operating system.

In addition to the data transfer techniques described in this manual, other methods are available: The Global Integrator product allows data to be exported from and, in some cases, imported into Global System Manager. Global Integrator supplements Global File Converters by formatting the data into industry standard file formats (e.g. Comma Separated Variable file format). The FTRAN utility distributed with Global PC Workstation allows files to be transferred between MS-DOS on the "workstation" and Global System Manager on the "host computer". Furthermore, the interfaces described in this manual (e.g. SVC-61) and some of those described in the Global Development File Management Manual (e.g. the DOS Access Method, the Unix Access Method and the Native ISAM Access Method) can be used by developers to write their own file converters and transfer utilities.

Two types of file converters are available. The first type (Physical Sector File Converters) do not use any of the functions within the host operating system and thus can be run on any Global System Manager configuration. As the name suggests they use the Physical Sector Access Method (see Chapter 8 of the Global Development File Management Manual) to access files within a non-Global directory. The second type of file converters (SVC-61 File Converters) use SVC-61 (see Chapters 6 and 7) to execute functions and system calls available within the host operating system in order to access files within a non-Global directory. Because of their dependence on a host operating system, the SVC-61 File Converters can only run on specific Global System Manager configurations.

In addition to the file converters, the Global File Converters product includes the RCBUILD utility. This utility is provided to build a conversion table to control the conversion of record structures when importing and exporting Global format ISAM files to C-ISAM or Btrieve files.

1.1 Physical Sector File Converters

Only one Physical Sector File Converter is distributed with Global File Converters V8.1. It allows data and text files on MS-DOS diskettes (and for some specialised Global System Manager (BOS) configurations, the root directory on a partitioned MS-DOS hard-disk) to be transferred to and from Global System Manager. This MS-DOS file converter can be used with all Global System Manager configurations that support industry standard diskettes, and is described in Chapter 2.

In addition to copying data and text files, the MS-DOS file converter also allows assembler programs, developed using a Microsoft assembler, to be transferred to Global System Manager so that they can be invoked by Global Cobol programs as described in the Global Assembler Interface Manual.

Note that versions of Global File Converters prior to V8.1, included Physical Sector File Converters that allowed Global System Manager to access files on RT-11, VMS and CP/M diskettes. Such file converters are now considered to be obsolete.

1.2 SVC-61 File Converters

Two SVC-61 File Converters are distributed with Global File Converters V8.1. The first SVC-61 File Converter, described in Chapter 3, allows files in any MS-DOS directory to be accessed on Global System Manager (MS-DOS and Windows) and Global System Manager (Novell NetWare) configurations (i.e. if the Machine Family Code, as displayed by $S, is "JW"). The second SVC-61 File Converter, described in Chapter 4, allows files anywhere within the Unix filing system to be accessed on Global System Manager (Unix) configurations (i.e. if the Machine Family Code, as displayed by $S, is "C2").

Because of the requirement for functions supplied by the host operating system, the SVC-61 File Converters can only be used on a particular host operating system. To enforce this restriction only the SVC-61 File Converter that is apposite for the host operating system is installed so that, for example, no SVC-61 file converters are installed on Global System Manager (BOS) configurations.

Note that versions of Global File Converters prior to V8.1, included an SVC-61 File Converter that allowed Global System Manager to access files on Global System Manager (VMS) configurations. This file converter is now considered to be obsolete.

1.3 Installation

The installation program installs all the possible Physical Sector File Converters (for Global File Converters V8.1, only the MS-DOS file converter is installed) and, if one is available, the appropriate SVC-61 File Converter for the host operating system.

Global File Converters is distributed on a single diskette, with a volume name of FCA. The installation program can be run from the "Install Global Software" function of a menu, or by running program FCINS directly from the distribution diskette. It first prompts for the hard disk subunit to be used for the FCPROG volume. The optional default given in brackets will be the first FCPROG unit allocated or, if there is no existing FCPROG unit, the first unused subvolume on the hard disk of suitable capacity (i.e. 200Kb minimum). For example:

```
GSM READY:FCINS<CTRL A>
PLEASE ASSIGN $P:140

Global file converters will be installed onto the FCPROG
unit unless otherwise specified.

The FCPROG unit must be at least 200K in size.

Overwrite existing FCPROG on unit 205? (Y):
```

Key <CR> to accept the default, or specify another unit that you wish to use. For example:

```
Overwrite existing FCPROG on unit 205? (Y):N

Specify FCPROG unit (299):204
```

You are then prompted to confirm that you wish to destroy the volume currently occupying the subunit specified, if this unit is already allocated:

```
      Destroy XXPROG on unit 204? (N):
```

Key Y to proceed, or N or <CR> to return to the "Specify FCPROG unit:" prompt in order to use a different subunit.

At any point in the dialogue you may key Q to quit the installation or <ESCAPE> to return to the starting point.

The installation procedure determines whether an SVC-61 File Converter is available for the host operating system. If an SVC-61 File Converter is not available (e.g. if Global File Converters is being installed on a Global System Manager (BOS) configuration) the following message will be displayed:

```
      No SVC-61 file converter is available for this
      Global System Manager configuration.
```

1.4 Difference Between the V8.0 and V8.1 Manuals

The equivalent of this manual supplied with Global File Converters V8.0, and earlier, included several sections and appendices that are no longer relevant: The description of the Global System Manager directory structure; the information required by developers writing their own file converters and operating notes for the VMS file converter. Such obsolescent topics and techniques are not documented in this version of the Global File Converters Manual.

2. Physical Sector File Converters

Physical Sector File Converters can be used on any Global System Manager configuration. They use the Physical Sector Access Method to access files within a non-Global directory structure. The file converter consists of 2 parts: FCONV is the command used to transfer files between Global System Manager and the non-Global format diskette (or hard-disk). It is responsible for all the messages and prompts which start $56 (see below). This utility acquires the capability to access the files of a particular operating system by loading the relevant Native Interface Program (NIP), the name of which is supplied in response to its first prompt. The NIP is a Global Cobol program which is linkage edited to begin at location #2000. It is normally named after the operating system with which it interfaces (e.g. MSDOS).

The only Physical Sector File Converter supplied with Global File Converters V8.1 is the MS-DOS file converter.

2.1 The MS-DOS Physical Sector File Converter

The MS-DOS Physical Sector File Converter supports the transfer of files from an MS-DOS diskette (or hard-disk) to Global System Manager, and vice versa. Table 2.1 lists the types of conversion supported. The MS-DOS Physical Sector File Converter **MUST** be customised for a particular disk format. This should be done by first copying the MSDOS program to a different name (e.g. MSDOSO2 to support O2A format) and then running MSDOSO2 and customising the O2A format as described in section 2.2.

Important note: The MS-DOS Physical Sector File Converter only recognises the root directory on an MS-DOS diskette or hard-disk. It is incapable of accessing files in sub-directories.

2.1.1 Using the MS-DOS Physical Sector File Converter

To use an MS-DOS file converter you must run FCONV and reply to its file converter prompt with the name of the converter to be used. For example, to run an MS-DOS converter customised and renamed as MSDOSO2 key:

```
GSM READY:FCONV
$56 FILE CONVERTER:MSDOSO2
$56 INPUT UNIT:
```

2.1.2 The Device Prompts

Next you are prompted for the unit-ids of the input volume and the output volume. For example to make the volume on 140 the input volume, and that on 210 the output volume:

```
$56 INPUT UNIT:140
$56 OUTPUT UNIT:210
```

The output device that you specify must be different from the input device, otherwise the following warning message is displayed, and the input device prompt is repeated to allow you to correct your error:

```
$56 INPUT AND OUTPUT UNITS CANNOT BE THE SAME
```

Conversion Type	Input file	Output file
D	An MS-DOS file considered as a set of blocks of size equal to the allocation size.	Global Relative Sequential (RS) file containing records whose length is the allocation size of the MS-DOS disk.
D	Global Relative Sequential (RS) file	MS-DOS data file, with the Global records stored contiguously and spanning allocation blocks.
T	MS-DOS ASCII file terminated with <CTRL Z>.	Global text file
T	Global text file identified as type TF in the directory listing.	MS-DOS ASCII file terminated with <CTRL Z>.
A	MS-DOS .EXE file produced by LINK.	Global absolute code file. If file name ends with B or 1 then the first 256 bytes of the .EXE file are ignored.

Table 2.1 - MS-DOS Interface Summary

Once different devices have been established the program asks you to identify the device to be used for the MS-DOS volume:

```
$56 WHICH DEVICE CONTAINS MSDOSO2 FORMAT FILES (I OR O)?:
```

You must key I if the MS-DOS device is to be used for input, O if It is to be employed for output. A reply of <CR> allows you to return to the input device prompt. An invalid reply causes the prompt to be repeated, over and over again as necessary, until correct information is supplied.

2.1.3 To Quit

You may exit the file converter at any time by simply keying <ESCAPE> in response to any prompt.

2.1.4 The Instruction Prompt

Once the device prompts have been satisfied the instruction prompt is displayed:

```
$56 FILE CONVERSION
:
```

You may reply:

<CR> to return to the input device prompt to change device assignments;

LIS to display the directory of either device on the screen;

COP to copy a named file from the input volume to the output volume.

If you supply an instruction the file converter cannot recognise, the following warning message is displayed and the instruction prompt is re-displayed to allow you to correct your keying error:

```
INVALID - REINPUT
```

Note that lower-case replies are allowed (e.g. "lis" is equivalent to "LIS").

2.1.5 LIS - List a Volume's Directory

The LIS instruction lists the directory of either the input or output volume:

```
:LIS INPUT OR OUTPUT?:
```

You must respond to this prompt by keying I (or i) if you wish to list the directory of the input volume; or O (or o) if a list of the output volume directory is required.

The directory listing produced is much simpler than that displayed by the Global System Manager $F file utility. For a Global System Manager volume the listing consists simply of the name and type of each file present. Similar skeletal information is displayed when the MS-DOS volume is listed.

If there are too many files present for the directory to be displayed as a single screen of information the following prompt is displayed on the base-line:

```
$56 NEXT PAGE?:
```

You must key Y, <CR> (or any single character apart from N) to obtain the next page of output. If you reply N, no more information will be output, and the instruction prompt will be re-displayed.

2.1.6 COP - Copy a Named File

You use the COP instruction to transfer a named file from an MS-DOS volume to a Global System Manager volume, or vice versa. The direction of the transfer depends on the initial input and output device assignments. The copying process involves a limited amount of file conversion, and the actual conversion that takes place depends on a type code which you supply.

You specify the input file name, the output file name, and the type, in response to the copy prompt:

```
:COP :input file name TO:output file name TYPE:type
```

Providing the inputs are supplied correctly, file transfer then takes place. The confirmatory message COPIED is output once the operation is complete, and then the instruction prompt is re-displayed.

For a Global file, the file name is the full file-id, including the prefix if one is present. The MS-DOS file name must include the extension, if any. The type is a single character code indicating which sort of conversion is to take place, as follows:

D	Conversion type D can be used to transfer any MS-DOS file to Global System Manager. Sectors are copied unchanged to a relative sequential file whose record length is the sector size. Transfer in the other direction causes the records of a Global relative sequential file to be copied, unaltered, to contiguous MS-DOS file space, spanning sectors if necessary.
T	Conversion type T is used to transfer MS-DOS ASCII source to Global text files, or vice versa. A terminating <CTRL Z> character is appended when creating an MS-DOS file, and removed when creating a Global file.
A	Conversion type A is used to convert a .EXE format load file as created by LINK. This file must consist of a single segment. Because LINK can only produce modules linked at zero, and the Global $RELOC command requires two copies of each module linked 256 bytes apart, some special conversion processing is necessary. The first copy of the module is linked normally at zero, and given a name ending "0" or "A". The second copy is linked with a 256 byte data area at the front and given a name ending "1" or "B" which causes the converter to ignore the first 256 bytes. Note that the segment names of the data module and the program must be the same to avoid there being two segments in the .EXE file.

2.1.7 Example 1 - Copying a Global System Manager Text File

A Global System Manager text file, S.SAMPLE, is to be transferred from the Global volume on unit 210 to an MS-DOS diskette on unit 140:

```
GSM READY:FCONV
$56 FILE CONVERTER:MSDOS
$56 INPUT UNIT:210
$56 OUTPUT UNIT:140
$56 WHICH DEVICE CONTAINS MSDOS FORMAT FILES (I OR O):O
$56 FILE CONVERSION
:LIS INPUT OR OUTPUT?:O
...........
........... (the files on the MS-DOS volume are listed)
...........
$56 FILE CONVERSION
:COP :S.SAMPLE TO:SAMPLE.TXT TYPE:T COPIED
$56 FILE CONVERSION
:<ESCAPE>
GSM READY:
```

2.1.8 Example 2 - Copying an MS-DOS Program File

A program has been assembled using ASM86 and linked as file ASMC1A.EXE, and has also been linked with a 256 byte data module as ASMCIB.EXE. These are to be transferred from a MS-DOS diskette on unit 140 to the Global volume on unit 210:

```
GSM READY:FCONV
```

```
$56 FILE CONVERTER:MSDOS
$56 INPUT UNIT:140
$56 OUTPUT UNIT:210
$56 WHICH DEVICE CONTAINS MSDOS FORMAT FILES (I OR O):I
$56 FILE CONVERSION
:LIS INPUT OR OUTPUT?:I
...........
........... (the files on the MS-DOS volume are listed)
...........
$56 FILE CONVERSION
:COP :ASMC1A.EXE TO:ASMC1A TYPE:A COPIED
$56 FILE CONVERSION
:COP :ASMC1B.EXE TO:ASMC1B TYPE:A COPIED
$56 FILE CONVERSION
:<ESCAPE>
GSM READY:
```

2.2 Customising the MS-DOS Physical Sector File Converter

The MS-DOS Physical Sector File Converter is a general purpose converter which must be parameterized with details of a specific disk or diskette format before it can be used. The customisation is performed by running the converter directly from the ready prompt, rather than via FCONV, which causes it to prompt you for the details it requires and then customize itself. Typically you will need to define where the directory starts on the disk, how many entries it contains, how much space is available for files, and whether any interleaving is to be performed.

MSDOS can be customised by running it as a stand alone program and responding to the prompts it produces. Using a job to perform the customization ensures it can be reapplied to any new version of converter that may be produced. If you produce your own version of a converter you are recommended to create such a job so that you have a permanent record of the customization needed. Note that the example customisation jobs provided with earlier versions of Global File Converters are not supplied with Global File Converters V8.1.

2.2.1 Diskette Customisation Parameters

The following customisation parameters are required for the standard 3½" and 5¼" diskette formats:

Parameter	C10	C24	G1	B3	O2
12-bit entries in FAT	Y	Y	Y	Y	Y
Starting sector of FAT #1	2	2	2	2	2
Starting sector of FAT #2	3	4	9	5	11
Length of the FAT in sectors	1	2	7	3	9
Starting sector of the directory	4	6	16	8	20

Starting sector of the data	11	13	30	15	34
Sectors per cluster	2	2	1	2	1
Last cluster on the disk	317	356	2373	715	2849
Interleaving table	N	N	N	N	N

Important note: The values for the "Last cluster on the disk" parameter in earlier versions of this manual were incorrect.

To customise MS-DOS to access standard MS-DOS 360Kb diskettes (i.e. format C24) only you need only key Y to the initial prompt as follows:

```
Do you wish to customise MSDOS permanently to allow access  to standard IBM PC
diskettes only?:Y
```

This option is not generally useful, now that 360Kb 5¼" diskettes are effectively obsolete, and will be removed in a future version of Global File Converters.

The following example illustrates the renaming and customisation of MSDOS to access 1.44Mb (format O2) diskettes:

```
GSM READY:$F
$66 INPUT DEVICE:207
$66 OUTPUT DEVICE:<CR>
$66 FILE MAINTENANCE
:COP :MSDOS TO:MSDOSO2 SIZE:<CR> COPIED
$66 FILE MAINTENANCE
:<ESCAPE>
GSM READY:MSDOSO2<CTRL A>

By running MSDOS as a stand-alone program you
may customise it to deal with the format of MS-DOS
diskette you are dealing with. FCONV must still be
used to transfer files.

Do you wish to customise MSDOSO2 permanently to allow
access to standard IBM PC diskettes only? (N):N

Does the FAT table have 12 bit entries? (Y):Y

What is the starting sector of FAT #1?
(the first sector on the disk is numbered 1) ( 2):2

What is the starting sector of FAT #2? ( 4):11

What is length of the FAT in sectors? ( 2):9

What is the starting sector of the directory? ( 6):20

Starting sector of the data, i.e. cluster #2? ( 13):34

How many sectors are there per cluster? ( 2):1
```

```
What is the number of the last cluster on
the disk. The first cluster of data is #2 ( 355):2849

Do you wish to specify a sector interleaving table? (N):N

Do you wish to customise MSDOSO2 permanently with the
parameters you have entered? (N):Y
```

2.2.2 Hard-Disk Customisation Parameters

Using the MSDOS Physical Sector File Converter to access files on an MS-DOS hard-disk is a highly specialised technique which can only be used in Global System Manager (BOS) "partitioned hard-disk" configurations. A suitable MS-DOS utility must be used to determine the parameters required to customise the MS-DOS file converter. Note that whereas all the currently supported MD-DOS diskettes (see section 2.2.1) include a 12-bit File Allocation Table (FAT) an MS-DOS hard-disk will invariably contain a 16-bit FAT.

3. The MS-DOS SVC-61 File Converter

The MS-DOS SVC-61 File Converter will be only be installed, at File Converter installation time (see section 1.3), if the host operating system is MS-DOS (i.e. if the Machine Code, displayed by $S as the Machine Family Code, is "JW"). This dependency on the host operating system is reflected in the name of the file converter which is distributed on the FCA installation diskette as "FCONV-JW". Consequently, this file converter can only be used on Global System Manager (MS-DOS and Windows) or Global System Manager (Novell NetWare) configurations.

If an attempt is made to run the MS-DOS SVC-61 File Converter on an inappropriate configuration (e.g. a Global System Manager (BOS) configuration) the following error message will be
displayed:

```
This file converter can only operate on
Global System Manager (MS-DOS and Windows) or
Global System Manager (Novell NetWare)
```

Unlike the Physical Sector File Converters, described in Chapter 2, the MS-DOS SVC-61 File Converter uses standard MS-DOS functions (see Chapter 6) to access MS-DOS files and is not restricted to just the MS-DOS root directory (a full MS-DOS pathname can be specified).

3.1 Using the MS-DOS SVC-61 File Converter

The MS-DOS SVC-61 File Converter is installed with a filename of FILECONV (i.e. during the installation, FCONV-JW on the FCA distribution diskette is copied to FILECONV on the FCPROG volume). The file converter commences by prompting for the type of operation to be performed:

```
GSM READY:FILECONV
Key List, Export, Import, <ESC> to exit (L):
```

The valid operations are:

- List an MS-DOS directory;
- Export a file from Global System Manager to MS-DOS;
- Import a file to Global System Manager from MS-DOS.

Key the first letter of the option to select an operation. For example, use the following dialogue to import a file from MS-DOS to Global System Manager:

```
Key List, Export, Import, <ESC> to exit (L):I
```

3.1.1 MS-DOS Pathnames

All the options described below require you to specify an MS-DOS directory or file pathname. If you do not specify a drive letter or sub-directory in the MS-DOS pathname, the file converter will use the currently logged in drive and directory. This is normally the drive and directory from which you loaded Global System Manager (e.g. C:\GSM).

For example:

C:\GSM\GLOBAL.BAT	full drive and absolute pathname;
\GSM\GLOBAL.BAT	absolute pathname from the currently logged-in drive;
GLOBAL.BAT	relative pathname from the currently logged-in drive and directory.

3.2 Listing an MS-DOS Directory

Key L to the command prompt to list an MS-DOS directory. You will be prompted for an MS-DOS pathname. For example, to list all the files on the GSM directory of the C: drive:

```
Key List, Export, Import, <ESC> to exit:L
Specify MS-DOS path and wildcard filename (e.g. C:\*.*)
:C:\GSM\*.*
```

Important note: The "directory" specification is actually a filename specification.

For example, to list all files in the GSMTEST directory that start with the string "GL" use the following dialogue:

```
Key List, Export, Import, <ESC> to exit:L
Specify MS-DOS path and wildcard filename (e.g. C:\*.*)
:C:GSMTEST\GL*.*
```

The List option displays the names, sizes and creation dates of all the "normal" MS-DOS files that match the selection criteria. MS-DOS hidden files, system files and sub-directories are NOT displayed by the List option.

If you do not specify a drive letter or sub-directory in the MS-DOS pathname, the file converter will use the currently logged in drive and directory (see section 3.1.1).

Important note: When specifying a directory for the List option you must normally end the sub-directory path with "*.*", or some other filename containing wild card characters. If you forget to append a wildcard filename to the directory pathname the message "Directory empty" will normally be displayed (unless a file that matches the directory specification is present on the MS-DOS disk).

3.3 Exporting a Single File From Global System Manager

Key E to the command prompt to export a single file from Global System Manager to MS-DOS. You will be prompted for a file type:

```
Key List, Export, Import, <ESC> to exit:E
Specify file type - Comms, Data, Report, System, Text or Zap
```

The file type must be one of:

C   Convert any type of Global System Manager file into an MS-DOS text file that is suitable for transmission by an MS-DOS Communications utility or Bulletin Board System. See section 3.3.4;

D	Copy any type of Global System Manager file to an MS-DOS file without any conversion (i.e. treat the file as raw data). See section 3.3.1;
R	Convert a Global System Manager report file (type RS) to MS-DOS text file format. See section 3.3.5;
S	Convert any type of Global System Manager file into an MS-DOS file preserving system information. See section 3.3.3;
T	Convert a Global System Manager text file (type TF) to MS-DOS text file format. See section 3.3.2;
U	Convert a Global System Manager text file (type TF) to MS-DOS text file format, replacing all hex 23 characters by hex 9C. See section 3.3.7;
Z	Convert a Global System Manager zap file (type RS) to MS-DOS text file format. See section 3.3.6.

When a valid file type has been specified, you are prompted for the name and unit of the Global System Manager file and the pathname for the MS-DOS file. For example, to copy the Global System Manager text file S.SAMPLE on unit 248 to an MS-DOS text file SAMPLE in directory C:\GLOBAL use the following dialogue:

```
Key List, Export, Import, <ESC> to exit:E
Specify file type - Comms, Data, Report, System, Text or Zap:T
Specify Global file:S.SAMPLE  Unit:248
Specify MS-DOS path and filename
:C:\GLOBAL\SAMPLE Copied
```

If an MS-DOS file of the same name already exists in the specified directory the file converter will ask you if you wish to delete the existing copy and proceed with the export.

When the transfer completes, the message "Copied" will appear and the main prompt described in section 3.1 will be redisplayed. If the export fails for any reason, an appropriate error message will be displayed.

If you do not specify a drive letter or sub-directory in the MS-DOS pathname, the file converter will use the currently logged in drive and directory (see section 3.1.1).

Important note: The file converter only performs cursory validation of the MS-DOS file name. It only checks for the illegal characters "*" and "?". If an attempt is made to export a Global System Manager file to an MS-DOS file with a strictly illegal filename (e.g. if it contains a "+" character) the transfer will fail but sometimes the error message will be inappropriate. You are strongly advised to only include alphanumeric characters, and full-stop, in the MS-DOS file name.

3.3.1 Data Export Mode (D)

Key D to the file-type prompt to export the Global System Manager file as a Data file. This mode can be used on any type of Global System Manager file. The data is copied unchanged to the MS-DOS file.

3.3.2 Text Export Mode (T)

Key T to the file type prompt to export a Global System Manager Text file (i.e. file-type TF) to an MS-DOS text file. This mode can only be used on TF files. If you attempt to copy any other type of Global System Manager file using the T option, an "Invalid format file" error message will appear.

To allow for the slight difference between Global System Manager and MS-DOS text file formats, the file converter appends <CR><LF><CTRL Z> (hex 0D 0A 1A) to the end of the MS-DOS file.

3.3.3 System Export Mode (S)

Key S to the file-type prompt to export the Global System Manager as a data file but preserving the System information within the file label. This mode can be used on any type of Global System Manager file. The data is copied unchanged to the MS-DOS "System image" file with the Global System Manager label information appended to the start of the file. When a file is imported from MS-DOS to Global System Manager (see section 3.5.3) the System information is written back to the directory thus restoring the original file type.

This option allows a Global System Manager program file to be exported to MS-DOS on one computer, transferred to another MS-DOS computer (e.g. using an MS-DOS Comms utility or Bulletin Board System) then imported back into Global System Manager preserving the original file type. If the same transfer was attempted using the D option (see sections 3.3.1 and 3.5.1), the resulting Global System Manager file would be a non-executable RS file.

3.3.4 Comms Export Mode (C)

Key C to the file-type prompt to export the Global System Manager file as a data file, preserving the System information within the file label, and converting the file to a format suitable for all MS-DOS Comms utilities and Bulletin Board Systems. This mode can be used on any type of Global System Manager file. The Global System Manager label information is appended to the start of the MS-DOS "Comms image" file. Furthermore, the file is converted to MS-DOS text file format containing only characters in the range #20 (hex) to #7F (hex), #0D (<CR>) and #0A (<LF>). A <CTRL Z> is appended to the end of the file. The resultant file is a valid MS-DOS text-file.

When a file is imported from MS-DOS to Global System Manager using the Comms option (see section 3.5.4) the "text" is converted back to the original byte values and the System information is written back to the directory thus restoring the original file type.

This option allows a Global System Manager program file to be exported to MS-DOS (as a text file) on one computer, transferred to another MS-DOS computer (e.g. using an MS-DOS Comms product or Bulletin Board System) then imported back into Global System Manager preserving the original file type. If the same transfer was attempted using the D option (see sections 3.3.1 and 3.5.1), the resulting Global System Manager file would be a non-executable RS file.

Note the difference between the C option and the S option (see section 3.3.3). Although both options allow Global System Manager program files, for example, to be exported to and re-imported from MS-DOS the intermediate MS-DOS file is vastly different. The MS-DOS file produced by the S option is a "pure data" file containing non-ASCII characters, whereas the MS-DOS file produced by the C option is a valid text file containing only 7-bit, ASCII characters (including <CR>, <LF> and <CTRL Z>).

The data-to-text conversion algorithm is as follows:

Byte value (hex)	Converted byte(s) (hex)
00 - 1F	7E20 - 7E3F
20 - 7C	20 - 7C
7D	7E60
7E	7E61
7F	7E62
80 - 9F	7E40 - 7E5F
A0 - FC	7D20 - 7D7C
FD	7E70
FE	7E71
FF	7E72

3.3.5 Report Export Mode (R)

Key R to the file type prompt to export a Global System Manager Report file (i.e. Print file, file-type RS) to an MS-DOS text file. This mode can only be used on RS files. If you attempt to copy any other type of Global System Manager file using the R option, an "Invalid format file" error message will appear.

The file converter recognises Print Control Bytes in the Global System Manager report file and replaces them by an equivalent character, or characters, in the MS-DOS text file. Print Control Bytes that indicate a "form-feed" are replaced by a byte of #0C (hex). Print Control Bytes that indicate a number of "new lines" are replaced by the requisite number of <CR><LF> characters. All other Print Control Bytes that have no equivalent in an MS-DOS text file (e.g. stationery format lines) are ignored completely.

Note that the file converter recognises Global System Manager spool files in a spool directory.

3.3.6 Zap Export Mode (Z)

Key Z to the file type prompt to export a Global System Manager Zap file (i.e. file-type RS) to an MS-DOS text file. This mode can only be used on RS files. If you attempt to copy any other type of Global System Manager file using the Z option, an "Invalid format file" error message will appear.

This option is functionally very similar to the R option (see section 3.3.5) except that all Print Control Bytes in the zap file are ignored. This produces a very condensed MS-DOS text file that excludes the page breaks and blank lines normally present in Global System Manager zap files.

Note that the file converter recognises Global System Manager spool files in a spool directory.

Note also, that unlike all the other export options, which have an equivalent import option (see section 3.5), the Zap option is unidirectional.

3.3.7 Special Text Export Mode (U)

Key U to the file type prompt to export a Global System Manager Text file (i.e. file-type TF) to an MS-DOS text file. This mode can only be used on TF files and is similar to the "T" mode (see section 3.3.2). In addition to the functionality of the "T" mode, the "U" mode converts all hex 23 characters (i.e. "hash" or "pound", depending on the Terminal Attribute Program) in the Global System Manager text file to hex 9C characters (i.e. "pound") in the MS-DOS text file.

3.4 Exporting Multiple Files From Global System Manager

To export multiple Global System Manager files from a single unit to a specific MS-DOS directory, terminate the reply of E to the command prompt by <CTRL B>. For example:

```
Key List, Export, Import, <ESC> to exit:E<CTRL B>
```

You are prompted for both a Global System Manager unit and an MS-DOS directory. For example, use the following dialogue to export multiple files from unit 210 to the MS-DOS directory C:\XFER:

```
Key List, Export, Import, <ESC> to exit:E<CTRL B>
Specify Global Unit:210
Specify MS-DOS directory
:C:\XFER
```

If you do not specify a drive letter or sub-directory in the MS-DOS pathname, the file converter will use the currently logged in drive and directory (see section 3.1.1).

Important note: The reply to the MS-DOS directory prompt MUST be a valid directory pathname (unlike the format of the MS-DOS pathname expected by the List Directory and Multiple File Import options - see sections 3.2 and 3.6).

You are prompted, in turn, for each file on the Global System Manager unit. The following replies are valid:

Y	Export the file (see below) and prompt for the next file in the Global System Manager directory;
N	Do not export the current file. Prompt for the next file in the Global System Manager directory. This response also applies for any single-character reply except for Y;
<CR>	Do not export the current file. Prompt for the next file in the Global System Manager directory;
<ESC>	Abandon the multiple export and return to the main prompt (see section 3.1);
<CTRL A>	Abandon the multiple export and return to the main prompt (see section 3.1);
<CTRL B>	Export the current file, and all the following files in the Global System Manager directory, using the default MS-DOS file name and export type (see below);
<CTRL C>	Treat as <CR>. Note that a reply of <CTRL C> does NOT redisplay the previous Global System Manager file.

3.4.1 The Default MS-DOS Filename

The default MS-DOS filename is derived from the Global System Manager filename by removing the Global System Manager file prefix, if any, and appending an MS-DOS file-suffix according to the following rules:

- If the Global System Manager file-prefix is one of the following the default MS-DOS file-suffix is set to ".TXT":

  B. F. D. L. M. Q. S. T. X. Z.

- If the first character of the Global System Manager file prefix is a lower-case letter, AND the multiple export is executing in "zap mode" (see section 3.4.2) the default MS-DOS file-suffix is set to ".XXX", where XXX is the upper-case equivalent of the lower-case Global System Manager file-prefix. This option has been included to facilitate the export of multi-part zaps;

- For all other Global System Manager filenames, the default MS-DOS file suffix is ".DAT".

Important note: When the file converter constructs the default MS-DOS file name it does not remove illegal characters (e.g. "*" and "?"). It is possible for the default MS-DOS file name to be invalid. You are strongly advised to only include alphanumeric characters, and full-stop, in the MS-DOS file name.

3.4.2 The Default Export Type

The default export type is derived from the Global System Manager filename according to the following rules:

- If the Global System Manager file-prefix is one of the following the default export type is "T" (for text file conversion):

  B. F. Q. S. T.

- If the Global System Manager file-prefix is one of the following the default export type is "R" (for report file conversion) if the multiple file export is executing in "normal mode"; or "Z" (for zap file conversion) if the multiple file export is executing in "zap mode". Note that "zap mode" is enabled when the export type is "Z", disabled when the export type is "R" and left unaltered for all other conversion types:

  D. L. M. X. Z.

- If the first character of the Global System Manager file prefix is a lower-case letter, AND the multiple file export is executing in "zap mode" the default export type is "Z" (for zap file conversion). This option has been included to facilitate the export of multi-part zaps, Note that "zap mode" is enabled when the export type is "Z", disabled when the export type is "R" and left unaltered for all other conversion types;

- For all other Global System Manager filenames, the default export type is "D" (for data file conversion).

3.4.3 An Example Multiple File Export

The following example illustrates the functionality described in sections 3.4.1 and 3.4.2:

```
Key List, Export, Import, <ESC> to exit:E<CTRL B>
Specify Global Unit:210
Specify MS-DOS directory
:C:\XFER
Copy TEST1   ?:Y to:TEST1.DAT Type:D Copied
Copy S.TEST2 ?:Y to:TEST2.TXT Type:T Copied
Copy D.TEST3 ?:Y to:TEST3.TXT Type:R Copied
Copy Z.ZAP1  ?:Y to:ZAP1.TXT  Type:Z Copied
Copy a.ZAP2  ?:Y to:ZAP2.AAA  Type:Z Copied
Copy b.ZAP3  ?:Y to:ZAP3.BBB  Type:Z Copied
```

Important note: The default file type is determined solely by the name of the Global System Manager file (i.e. by the file prefix). However, the file type validation, performed at the start of the conversion, is based on the Global System Manager file type. Consequently, it may be possible for the default file type to be invalid for an inappropriately named Global System Manager file (e.g. for an RS file with an S. file prefix).

3.5 Importing a Single File to Global System Manager

Key I to the command prompt to import a single file from MS-DOS to Global System Manager. You will be prompted for a file type:

```
Key List, Export, Import, <ESC> to exit:I
Specify file type - Comms, Data, Program, Report, System or Text
```

The file type must be one of:

C	Reconvert an MS-DOS text file, created using the Comms export mode (see section 3.3.4), back into the original Global System Manager file. See section 3.5.4;
D	Copy any type of MS-DOS file to a Global System Manager Relative Sequential file (type RS) without any conversion (i.e. treat the file as raw data). See section 3.5.1;
P	Convert an MS-DOS program file to a Global System Manager program file format. See section 3.5.6;
R	Convert an MS-DOS text file to Global System Manager report file format (type RS). See section 3.5.5;
S	Reconvert an MS-DOS file, created using the System export mode (see section 3.3.3), back into the original Global System Manager file. See section 3.5.3;
T	Convert an MS-DOS text file to Global System Manager text file format (type TF). See section 3.5.2.

Note that there is no "import" equivalent to the export zap file option (see section 3.3.6)

When a valid file type has been specified, you are prompted for the pathname of the MS-DOS file and the name and unit of the Global System Manager file. For example, to copy the MS-DOS text file SAMPLE in directory C:\GLOBAL to a Global System Manager text file S.SAMPLE on unit 248 use the following dialogue:

```
Key List, Export, Import, <ESC> to exit:I
```

```
Specify file type - Comms,Data,Program,Report,System or Text:T
Specify MS-DOS path and filename
:C:\GLOBAL\SAMPLE
Specify Global file:S.SAMPLE  Unit:248  Copied
```

If a Global System Manager file of the same name already exists on the specified unit the file converter will ask you if you wish to delete the existing copy and proceed with the import.

When the transfer completes, the message "Copied" will appear and the main prompt described in section 3.1 will be redisplayed. If the import fails for any reason, an appropriate error message will be displayed.

If you do not specify a drive letter or sub-directory in the MS-DOS pathname, the file converter will use the currently logged in drive and directory (see section 3.1.1).

3.5.1 Data Import Mode (D)

Key D to the file type prompt to import an MS-DOS data file to Global System Manager Relative Sequential file format (i.e. file-type RS). The contents of the MS-DOS file are copied to the Global System Manager file unchanged.

You will be prompted for the Record Length to be associated with the Global System Manager RS-file. The RS-file is created with the record length specified even if the length of the MS-DOS file is not an exact multiple of the record length.

This option may be used to import any MS-DOS file (i.e. not just those MS-DOS files created using the Export data-file mode of this file converter - see section 3.3.1).

3.5.2 Text Import Mode (T)

Key T to the file type prompt to import an MS-DOS text file to Global System Manager text file format (i.e. file-type TF). All the characters in the MS-DOS file are copied to the Global System Manager text file except for the <CTRL Z> that terminates the file. Note that any <CTRL Z> character, or any other control character, that is not the last character in the file, will be copied into the Global System Manager file unchanged.

This option may be used to import any MS-DOS text file (i.e. not just those MS-DOS text files created using the Export text-file mode of this file converter - see section 3.3.2).

3.5.3 System Import Mode (S)

Key S to the file-type prompt to re-constitute the Global System Manager file that was exported to create the MS-DOS "System image" file (see section 3.3.3).

Important note: This option should only be used on MS-DOS "System image" files (which should not be confused with MS-DOS System Files e.g. CONFIG.SYS). If an attempt is made to use this conversion option on an MS-DOS file that was not created using the export System mode of this file converter (see section 3.3.3), the results will be unpredictable.

3.5.4 Comms Import Mode (C)

Key C to the file-type prompt to re-constitute the Global System Manager file that was exported to create the MS-DOS "Comms image" file (see section 3.3.4).

Important note: This option should only be used on MS-DOS "Comms image" files. If an attempt is made to use this conversion option on an MS-DOS file that was not created using the export Comms mode of this file converter (see section 3.3.4), the results will be unpredictable.

3.5.5 Report File Import Mode (R)

Key R to the file type prompt to import an MS-DOS text file to a Global System Manager report file (i.e. Print file, file-type RS). The Global System Manager print file is always created with a record length of 133.

All <CR><LF> character combinations in the MS-DOS text file are replaced by the appropriate Print Control Byte in the Global System Manager print file. If the file converter detects a form-feed character (i.e. a byte of #0C (hex)) in the MS-DOS text-file, a Print Control Byte indicating a new line is appended to the start on the next line in the Global System Manager print file. No special stationery format lines are included in the print file.

This option may be used to import any MS-DOS text file (i.e. not just those MS-DOS text files created using the Export report-file mode of this file converter - see section 3.3.5).

3.5.6 Program File Import Mode (P)

Key P to the file-type prompt to import an assembler file developed using an MS-DOS based Software Development Kit to the Global System Manager environment. The file converter will copy MS-DOS .EXE files providing they have been suitably linked. Two copies of the same program, linked 100 (hex) bytes apart, must be produced using the Microsoft Assembler (MASM) and Linker (LINK). The two files must be specially named: The name of the file linked at address 0 must end in 0 or A. The name of the file linked at address hex 100 must end in 1 or B. The pair of files imported using this file converter option are suitable for subsequent conversion using $RELOC. For further information regarding the creation of assembler code for use in the Global System Manager environment please contact TIS Software Ltd.

3.6 Importing Multiple Files to Global System Manager

To import multiple files from a MS-DOS directory, terminate the reply of I to the command prompt by <CTRL B>. For example:

```
Key List, Export, Import, <ESC> to exit:I<CTRL B>
```

You are prompted for both an MS-DOS pathname and Global System Manager unit. Note that the MS-DOS "directory" specification is actually a filename specification. For example, use the following dialogue to import multiple files from the MS-DOS directory C:\XFER to unit 210:

```
Key List, Export, Import, <ESC> to exit:I<CTRL B>
Specify MS-DOS path and wildcard filename (e.g. C:\*.*)
:C:\XFER\*.*
Specify Global Unit:210
```

If you do not specify a drive letter or sub-directory in the MS-DOS pathname, the file converter will use the currently logged in drive and directory (see section 3.1.1).

Important note: The reply to the MS-DOS directory prompt MUST be a valid file pathname (unlike the format of the MS-DOS directory pathname expected by the Multiple File Export option - see section 3.4).

For example, use the following dialogue to import all the files in the MS-DOS directory C:\XFER, that start with the characters "GL", to unit 210:

```
Key List, Export, Import, <ESC> to exit:I<CTRL B>
Specify MS-DOS path and wildcard filename (e.g. C:\*.*)
:C:\XFER\GL*.*
Specify Global Unit:210
```

You are prompted, in turn, for each file in the MS-DOS directory that matches the pathname specification. The following replies are valid:

Y	Import the file (see below) and prompt for the next file in the MS-DOS directory;
N	Do not import the current file. Prompt for the next file in the MS-DOS directory. This response also applies for any single-character reply except for Y;
<CR>	Do not import the current file. Prompt for the next file in the MS-DOS directory;
<ESC>	Abandon the multiple import and return to the main prompt (see section 3.1);
<CTRL A>	Abandon the multiple import and return to the main prompt (see section 3.1);
<CTRL B>	Treat as <CR>. Note that a reply of <CTRL B> does NOT import the current, and all subsequent, MS-DOS files;
<CTRL C>	Treat as <CR>. Note that a reply of <CTRL C> does NOT redisplay the previous MS-DOS file.

3.6.1 The Default Global System Manager Filename

The default Global System Manager filename is derived from the MS-DOS filename by removing the MS-DOS file extension, if any, and appending a Global System Manager file-prefix according to the following rules:

- If the MS-DOS file extension is one of the following the default Global System Manager file-prefix is set to "S." and only the first 6 characters of the 8-character MS-DOS file-prefix are significant:

  .TXT .BAT

- For all other MS-DOS filenames, no Global System Manager file-prefix is appended to the default file name and all 8 characters of the MS-DOS file-prefix are significant.

3.6.2 The Default Import Type

The default import type is derived from the MS-DOS filename according to the following rules:

- If the MS-DOS file extension is one of the following the default import type is "T" (for text file conversion):

  .TXT .BAT

- If the MS-DOS file extension is ".EXE", the default import type is "P" (for program file conversion);

- For all other MS-DOS filenames, the default import type is "D" (for data file conversion).

3.6.3 An Example Multiple File Import

The following example illustrates the functionality described in sections 3.6.1 and 3.6.2:

```
Key List, Export, Import, <ESC> to exit:I<CTRL B>
Specify MS-DOS path and wildcard filename (e.g. C:\*.*)
:C:\XFER\*.*
Specify Global Unit:210
Copy TEST.DAT ?:Y to:TEST   Type:D Copied
Copy TEST.TXT ?:Y to:S.TEST Type:T Copied
```

4. The Unix SVC-61 File Converter

The Unix SVC-61 File Converter will be only be installed, at File Converter installation time (see section 1.3), if the host operating system is Unix (i.e. if the Machine Code, displayed by $S as the Machine Family Code, is "C2"). This dependency on the host operating system is reflected in the name of the file converter which is distributed on the FCA installation diskette as "FCONV-C2". Consequently, this file converter can only be used on Global System Manager (Unix) configurations.

If an attempt is made to run the Unix SVC-61 File Converter on an inappropriate configuration (e.g. a Global System Manager (BOS) configuration) the following error message will be displayed:

```
This file converter can only operate on Global System Manager (Unix)
```

Important note: Under some circumstances Global System Manager (Unix) obtains the privileges of a Unix super-user. Consequently, when using the Unix SVC-61 File Converter on such configurations it is possible to by-pass the normal Unix security. The Unix System Administrator should be aware of the two conditions under which Global System Manager (Unix) operates with super-user privileges:

- On those screens where the DIRECT DISPLAY option in the USER DISPLAY ATTRIBUTES section of the configuration file is set to N;
- On the second, or subsequent, USER for a particular SYSTEM (as specified in the Systems file).

Please consult the Global Operating Manual (Unix) for further details.

4.1 Using the Unix SVC-61 File Converter

The Unix SVC-61 File Converter is installed with a filename of FILECONV (i.e. during the installation, FCONV-C2 on the FCA distribution diskette is copied to FILECONV on the FCPROG volume). The file converter commences by prompting for the type of operation to be performed:

```
GSM READY:FILECONV
Key Directory, List, Export, Import, <ESC> to exit (L):
```

If the Global System Manager (Unix) configuration includes the Informix C-ISAM routines, the following message will appear before the main selection prompt:

```
Using C-ISAM from Informix Software Inc.
```

Please refer to your Global Configuration Notes for further details regarding the availability of Informix C-ISAM.

The valid operations are:

- Establish a default Unix directory;
- List a Unix directory;
- Export a file from Global System Manager to Unix;

- Import a file to Global System Manager from Unix.

Key the first letter of the option to select an operation. For example, use the following dialogue to import a file from Unix to Global System Manager:

```
      Key Directory, List, Export, Import, <ESC> to exit (L):I
```

In addition to the 4 options listed above, the Unix file converter includes a diagnostic mode which is toggled on/off by keying X to the main selection prompt. See section 4.6 for further details.

4.2 Establishing a Default Directory

All the commands (i.e. List, Export and Import) prompt for a Unix directory path. You can establish a default directory path using the D command as follows:

```
      Key Directory, List, Export, Import, <ESC> to exit (L):D
      Enter default Unix directory path or <CR> to accept current path
      :/u5/gsm/global
```

This directory path will now be used as the default path in any further commands. Note that this operation does not invoke the Unix "change directory" function, it merely initialises a data area within the Unix file converter.

4.3 Listing a Unix Directory

Key L to the command prompt to list a Unix directory. You will be prompted for a Unix pathname with the default directory (see section 4.2), if any, as the default reply. For example, to list all the files in the Unix /u5/gsm/global directory:

```
      Key Directory, List, Export, Import, <ESC> to exit (L):L
      Enter Unix directory path or <CR> to accept current path
      :/u5/gsm/global
```

The List option merely displays the names of all the "normal" Unix files and sub-directories in the specified directory. The results are similar to those produced by the following Unix command:

```
      # ls
```

No file-size, creation-date, ownership or permission information is displayed.

If no directory is specified the current Unix home directory is used.

4.4 Exporting a Single File From Global System Manager

Key E to the command prompt to export a single file from Global System Manager to Unix. You will be prompted for a file type:

```
      Key Directory, List, Export, Import, <ESC> to exit:E
      Specify file type - Comms, Text, System or Data (T):
```

If the Global System Manager (Unix) configuration includes the Informix C-ISAM routines, the file-type prompt will appear as follows:

```
      Key Directory, List, Export, Import, <ESC> to exit:E
      Specify file type - Comms, Text, System, Data or ISAM (T):
```

The file type must be one of:

C	Convert any type of Global System Manager file to a Unix text file that is suitable for transmission by a Unix communications utility. See section 4.4.5;
D	Copy any type of Global System Manager file to a Unix file without any conversion (i.e. treat the file as raw data). See section 4.4.1;
I	Convert a Global System Manager Relative Sequential or Index Sequential file to a C-ISAM database. See section 4.4.3. Note that this option is only available if the Global System Manager (Unix) configuration includes the Informix C-ISAM routines;
S	Convert any type of Global System Manager file to a Unix file preserving system information. See section 4.4.4;
T	Convert a Global System Manager text file (type TF) to Unix text file format. See section 4.4.2.

When a valid file type has been specified, you are prompted for the name and unit of the Global System Manager file and the pathname for the Unix file. For example, to copy the Global System Manager text file S.SAMPLE on unit 248 to a Unix text file /usr/global/sample (assuming the default directory string is set to /usr/global) use the following dialogue:

```
Key Directory, List, Export, Import, <ESC> to exit:E
Specify file type - Text, Data or ISAM (T):T
Specify Global file:S.SAMPLE Unit:248
Specify Unix path and filename
:/usr/global/sample Copied
```

When the transfer completes, the message "Copied" will appear and the main prompt described in section 4.1 will be redisplayed. If the export fails for any reason, an appropriate error message will be displayed.

Important note: The file converter performs no validation of the Unix file name. If an attempt is made to export a Global System Manager file to a Unix file with a strictly illegal filename (e.g. if it contains a "*" character) the transfer will fail but sometimes the error message will be inappropriate. You are strongly advised to only include alphanumeric characters in the Unix file name.

4.4.1 Data Export Mode (D)

Key D to the file-type prompt to export the Global System Manager file as a Data file. This mode can be used on any type of Global System Manager file. The data is copied unchanged to the Unix file. If the Unix file already exists in the directory specified, the file converter will prompt you to delete the existing file and proceed with the export.

4.4.2 Text Export Mode (T)

Key T to the file type prompt to export a Global System Manager Text file (i.e. file-type TF) to a Unix text file. This mode can only be used on TF files. If you attempt to copy any other type of Global

System Manager file using the T option, an "Invalid format file" error message will appear.

To allow for the slight difference between Global System Manager and Unix text file formats, the file converter removes <LF> characters (0x0a) from the text file.

If the Unix file already exists in the directory specified, the file converter will prompt you to delete the existing file and proceed with the export.

4.4.3 ISAM Export Mode (I)

This option is only available if the Global System Manager (Unix) configuration includes the Informix C-ISAM routines.

Key I to the file type prompt to export a Global System Manager Relative Sequential (RS) or Indexed Sequential (IS) file to a Unix C-ISAM database. This mode can only be used on RS or IS files. If you attempt to copy any other type of Global System Manager file using the I option, an "Invalid format file" error message will appear.

A Record Conversion Table, built by the RCBUILD utility (see Chapter 5), describing the record structures of both the Global and Unix files, must be specified. If the Record Conversion Table file is a data library you will be prompted for the conversion table entry. For example:

```
Name of conversion table file:PRDLIB Unit:207
Conversion table name, ? to list:PRDCON
```

Key ? to display a complete list of conversion tables in the data library.

When the Record Conversion Table has been specified you are prompted for the file name and unit of the Global file and the pathname of the C-ISAM database. For example:

```
Key Directory, List, Export, Import, <ESC> to exit:E
Specify file type - Text, Data or ISAM (T):I
Name of conversion table file:PRDT Unit:207
Specify Global file:PRODUCT Unit:248
Specify Unix path and filename
:/usr/global/product Copied
```

If the Unix C-ISAM database already exists, the following prompt will appear:

```
ISAM file not empty - Delete records, Add to file or <CR> to continue:
```

Key D to this prompt to delete all the existing records in the Unix C-ISAM database and add the new records exported from the Global file. Key A to add the new records exported from the Global file to the Unix C-ISAM database. Key <CR> to return to the output file prompt without exporting any records.

Note that the export of a Global System Manager RS or IS file to a C-ISAM database normally results in the creation of two Unix files:

xxxxxxxx.idx	C-ISAM index file
xxxxxxxx.dat	C-ISAM data file

4.4.4 System Export Mode (S)

Key S to the file-type prompt to export the Global System Manager as a data file but preserving the System information within the file label. This mode can be used on any type of Global System Manager file. The data is copied unchanged to the Unix "System image" file with the Global System Manager label information appended to the start of the file. When a file is imported from Unix to Global System Manager (see section 4.5.4) the System information is written back to the directory thus restoring the original file type.

This option allows a Global System Manager program file to be exported to Unix on one computer, transferred to another Unix computer (e.g. using a Unix Comms utility) then imported back into Global System Manager preserving the original file type. If the same transfer was attempted using the D option (see sections 4.4.1 and 4.5.1), the resulting Global System Manager file would be a non-executable RS file.

4.4.5 Comms Export Mode (C)

Key C to the file-type prompt to export the Global System Manager file as a data file, preserving the System information within the file label, and converting the file to a format suitable for all Unix Comms utilities (e.g. uucp). This mode can be used on any type of Global System Manager file. The Global System Manager label information is appended to the start of the Unix "Comms image" file. Furthermore, the file is converted to Unix text file format containing only characters in the range #20 (hex) to #7F (hex), #0D (<CR>) and #0A (<LF>). The resultant file is a valid Unix text-file.

When a file is imported from Unix to Global System Manager using the Comms option (see section 4.5.5) the "text" is converted back to the original byte values and the System information is written back to the directory thus restoring the original file type.

This option allows a Global System Manager program file to be exported to Unix (as a text file) on one computer, transferred to another Unix computer (e.g. using a Unix Comms product) then imported back into Global System Manager preserving the original file type. If the same transfer was attempted using the D option (see sections 4.4.1 and 4.5.1), the resulting Global System Manager file would be a non-executable RS file.

Note the difference between the C option and the S option (see section 4.4.4). Although both options allow Global System Manager program files, for example, to be exported to and re-imported from Unix the intermediate Unix file is vastly different. The Unix file produced by the S option is a "pure data" file containing non-ASCII characters, whereas the Unix file produced by the C option is a valid text file containing only 7-bit, ASCII characters (including <CR> and <LF>).

The data-to-text conversion algorithm is as follows:

Byte value (hex)	Converted byte(s) (hex)
00 - 1F	7E20 - 7E3F
20 - 7C	20 - 7C
7D	7E60
7E	7E61
7F	7E62

80 - 9F	7E40 - 7E5F
A0 - FC	7D20 - 7D7C
FD	7E70
FE	7E71
FF	7E72

4.5 Importing a Single File to Global System Manager

Key I to the command prompt to import a single file from Unix to Global System Manager. You will be prompted for a file type:

```
Key Directory, List, Export, Import, <ESC> to exit:I
Specify file type - Comms, Text, System or Data (T):
```

If the Global System Manager (Unix) configuration includes the Informix C-ISAM routines, the file-type prompt will appear as follows:

```
Key Directory, List, Export, Import, <ESC> to exit:E
Specify file type - Comms, Text, System, Data or ISAM (T):
```

The file type must be one of:

C	Reconvert a Unix text file, created using the Comms export mode (see section 4.4.5), back to the original Global System Manager file. See section 4.5.5;
D	Copy any type of Unix file to a Global System Manager Relative Sequential file (type RS) without any conversion (i.e. treat the file as raw data). See section 4.5.1;
I	Convert a Unix C-ISAM file to a Global System Manager Relative Sequential or Index Sequential file. See section 4.5.3. Note that this option is only available if the Global System Manager (Unix) configuration includes the Informix C-ISAM routines;
S	Reconvert a Unix file, created using the System export mode (see section 4.4.4), back to the original Global System Manager file. See section 4.5.4;
T	Convert a Unix text file to Global System Manager text file format (type TF). See section 4.5.2.

When a valid file type has been specified, you are prompted for the pathname of the Unix file and the name and unit of the Global System Manager file. For example, to copy the Unix text file /usr/global/sample (assuming the default directory string is set to /usr/global) to a Global System Manager text file S.SAMPLE on unit 248 use the following dialogue:

```
Key Directory, List, Export, Import, <ESC> to exit:I
Specify file type - Text, Data or ISAM (T):T
Specify Unix path and filename
:/usr/global/sample
Specify Global file:S.SAMPLE Unit:248 Copied
```

If a Global System Manager file of the same name already exists on the specified unit the file converter will ask you if you wish to delete the existing copy and proceed with the import.

When the transfer completes, the message "Copied" will appear and the main prompt described in section 4.1 will be redisplayed. If the import fails for any reason, an appropriate error message will be displayed.

4.5.1 Data Import Mode (D)

Key D to the file type prompt to import a Unix data file to Global System Manager Relative Sequential file format (i.e. file-type RS). The contents of the Unix file are copied to the Global System Manager file unchanged.

You will be prompted for the Record Length to be associated with the Global System Manager RS-file. The RS-file is created with the record length specified even if the length of the Unix file is not an exact multiple of the record length.

This option may be used to import any Unix file (i.e. not just those Unix files created using the Export data-file mode of this file converter - see section 4.4.1).

4.5.2 Text Import Mode (T)

Key T to the file type prompt to import a Unix text file to Global System Manager Text file format (i.e. file-type TF). All the characters in the Unix file are copied to the Global System Manager text file except for <LF> characters (0x0a) which are converted to a <CR><LF> character pair to satisfy the requirements of the Global System Manager text file format.

This option may be used to import any Unix text file (i.e. not just those Unix text files created using the Export text-file mode of this file converter - see section 4.4.2).

4.5.3 ISAM Import Mode (I)

This option is only available if the Global System Manager (Unix) configuration includes the Informix C-ISAM routines.

Key I to the file type prompt to import a Global System Manager Relative Sequential (RS) or Indexed Sequential (IS) file from a Unix C-ISAM database.

A Record Conversion Table, built by the RCBUILD utility (see Chapter 5), describing the record structures of both the Global and Unix files, must be specified. If the Record Conversion Table file is a data library you will be prompted for the conversion table entry. For example:

```
      Name of conversion table file:PRDLIB Unit:207
      Conversion table name, ? to list:PRDCON
```

Key ? to display a complete list of conversion tables in the data library.

When the Record Conversion Table has been specified you are prompted for the pathname of the C-ISAM database and file name and unit of the Global file. You are also prompted for the Global file type (i.e. IS or RS). For example:

```
      Key Directory, List, Export, Import, <ESC> to exit:I
      Specify file type - Text, Data or ISAM (T):I
      Name of conversion table file:PRDT Unit:207
      Specify Unix path and filename
      :/usr/global/product
      Specify Global file:PRODUCT Unit:248
```

```
Index sequential or Relative sequential (I):I Copied
```

4.5.4 System Import Mode (S)

Key S to the file-type prompt to re-constitute the Global System Manager file that was exported to create the Unix "System image" file (see section 4.4.4).

Important note: This option should only be used on Unix "System image" files. If an attempt is made to use this conversion option on a Unix file that was not created using the export System mode of this file converter (see section 4.4.4), the results will be unpredictable.

4.5.5 Comms Import Mode (C)

Key C to the file-type prompt to re-constitute the Global System Manager file that was exported to create the Unix "Comms image" file (see section 4.4.5).

Important note: This option should only be used on Unix "Comms image" files. If an attempt is made to use this conversion option on a Unix file that was not created using the export Comms mode of this file converter (see section 4.4.5), the results will be unpredictable.

4.6 Diagnostics Mode

The Unix SVC-61 File Converter includes a special diagnostic mode which can be used to determine the exact cause of any error. When the diagnostics mode is enabled, the List, Export and Import commands display the various SVC-61 operations (see section 7.2) and error codes (see section 7.7.3). To enable diagnostics mode, key X to the options prompt before attempting a List, Export or Import operation. For example:

```
Key Directory, List, Export, Import, <ESC> to exit (L):X
Diagnostics now enabled
```

To disable diagnostics mode, key X to the options prompt. For example:

```
Key Directory, List, Export, Import, <ESC> to exit (L):X
Diagnostics now disabled
```

5. RCBUILD - Record Conversion Table Build Utility

The RCBUILD utility is provided to build a conversion table to control the conversion of record structures when importing and exporting Global ISAM files to C-ISAM or Btrieve files. RCBUILD is also required to build a conversion table when DBMAIN is used to create a C-ISAM format or Btrieve format DMAM database (see section 5.1.1 of the Global Cobol Data Management Manual). The conversion tables are used by the Unix file converter (see Chapter 4), some file access methods and both the Unix C-ISAM (see Chapter 6) and MS-DOS Btrieve (see Chapter 7) Universal Channel Interfaces.

RCBUILD compiles the conversion table from a given source file. Note that conversion tables are built automatically by Speedbase when Speedbase databases are held in C-ISAM format.

5.1 Running the Conversion Table Build Program

To use the conversion table build program, run RCBUILD from the unit onto which you have installed the Global File Converters (usually FCPROG). RCBUILD will then prompt you for the input source file and unit:

```
GSM READY:RCBUILD
SOURCE FILE:S.PRTAB UNIT:207
```

You will then be asked if you want to create a Relative Sequential output conversion table or if you want to include the conversion table in a data library. RCBUILD will prompt for the name and unit of the relative sequential file or data library.

If you have chosen a Relative Sequential conversion table, and the output file already exists, you will be asked if you want to delete it. If you have chosen a data library output file, and the file does not exist, RCBUILD will create one of that name. RCBUILD will then ask for the data library record name to assign to the conversion table. If the record exists you are given the option of deleting it. For example:

```
OUTPUT FILE TYPE - DLAM/RSAM (D/R):D
OUTPUT LIBRARY( ):PRLIB UNIT( ):207
DLAM RECORD NAME:PRTAB
```

RCBUILD will then prompt for the listing file unit. If you key <CTRL A> to this prompt the listing will be displayed on the screen.

5.2 The Conversion Table Source File

The conversion table source file provides a description of the salient features required for a Global Relative Sequential, Indexed Sequential, DMAM or (potentially) Speedbase conversion.

Only one conversion source file and table is required to perform the conversion in either direction. The input and output fields depend on whether you are importing to, or exporting from, Global System Manager.

5.2.1 Source File Structure

The following initial header lines describe the records which form the two sides of the conversion:

```
     FILE filename GLOBAL TO [UNIX/BTRIEVE]
     GLOBAL RECORD [LENGTH] glen KEY [LENGTH] gklen OFFSET 4
     [UNIX/BTRIEVE]  RECORD  [LENGTH]  ulen  KEY  [LENGTH]  uklen  OFFSET
uoff
```

where filename is the name of the Global file which is included for descriptive purposes only. The type of conversion, UNIX (to indicate a Unix C-ISAM file) or BTRIEVE (to indicate a Btrieve file) must be indicated.

The values of glen and gklen define the record length and key length of the Global ISAM file or the Global DMAM record set. The offset of the key from the start of the Global record is 4 as required for Global ISAM files. If DMAM records are being converted the offset must still be keyed in as 4 although this has no real relevance. The values of ulen, uklen and uoff describe the record length key length and key offset of the output file. The offset of the key from the start of the record counts from 0 as in Global files and although a value must be entered it is only of relevance for ISAM conversion.

Following the header lines there will be a number of field conversion lines, each specifying a conversion to be performed (see section 5.2.2). Finally there will be a conversion termination line as follows:

```
     END CONVERSION
```

All lines may contain comments (introduced by a "*" character). Any characters after the comment character are not processed.

5.2.2 Field Conversion Lines

The general syntax of a field conversion line is:

```
     [gfield]  [TRANS]  [DESC]  gtype-qual  goff  =  [ufield]  utype-qual
uoff
```

where gfield and ufield are the names of the Global and C-ISAM (or Btrieve) fields which are included primarily as comments. The type and size of the Global and C-ISAM (or Btrieve) fields are indicated by gtype-qual and otype-qual (see section 5.3) and the location of the field within the two records by goff and uoff. The offsets defined count from 0 and may either be decimal numbers or 2 byte hexadecimal numbers preceded by a "#". RCBUILD will build a maximum of 179 field conversion lines. The TRANS and DESC key words are for use with DMAM translation and descending keys and are described in section 5.4).

In addition to field conversion description lines, conditional lines may be used to control the conversion of multiple record types. These have the form:

```
     IF [offset] [NOT] comparison
```

where offset is the decimal offset within the Global record of the field you wish to compare. If the offset is omitted then a value of 0 will be assumed. The comparison field must be either a character string enclosed in double quotes or a hexadecimal value, and is always a one or two byte quantity. If the comparison is true then the following conversion lines are processed. If the comparison is not true then subsequent conversion lines are ignored until the end of the conversion or until another conditional line is encountered.

There is a special case of the conditional line which is written:

```
     ALWAYS
```

This causes all subsequent conversion lines to be valid.

Conditional instructions are of most use when the Global file contains two slightly different record structures. For example, for Global ISAM files the different record types are usually indicated by the value in the "type" field. Global System Manager ISAM files that contains two record types would usually need to be converted to two separate C-ISAM, or Btrieve, files.

There are two other conditional modes which are written as:

```
     RECORD AREA1
 and:
     RECORD AREA2
```

These record areas are used in conjunction with the UCI interface "conversion modes" described in sections 6.8.1.3 and 7.8.1.3. If the conversion mode specifies a record area then only those fields within that record area will be converted. All other conditionals still apply and are not affected by the record area statements. The conversion will assume that fields are in neither record area until a record area statement occurs. All subsequent fields are considered to be in that record area until the end of the conversion table has been reached or until a different record area is specified.

In addition to the conditional lines there is a skip line as follows:

```
     SKIP records
```

where records is the number of following conversion records (including any conditional lines, skip lines, occurs lines as well as normal conversion lines) which are to be skipped.

The skip line may be useful as part of a conditional. For example:

```
     IF #00
          conversion line 1
     SKIP 2
     ALWAYS
          conversion line 2
     ALWAYS
          conversion line 3
```

where conversion line 2 will be omitted if offset zero is #00 but will be executed otherwise thus providing an "IF/THEN/ELSE" construct.

Note that the number of conversion lines (including conditional lines) is limited to 179.

5.3 Conversion Types and Qualifiers

This section describes the conversion types allowed and the formats of the type-qual parameters.

Important note: The conversion of Global floating point numbers and pointer fields are not supported. Furthermore, Global display numeric fields do not have a direct C-ISAM or Btrieve equivalent so are also not supported.

5.3.1 Character Fields

Global character fields can only have C-ISAM or Btrieve character fields or nothing fields on the other side of the conversion. They are indicated on both sides of the conversion line by gtype-qual and utype-qual of the form:

```
     [PIC] X(n)
```

where n is the length of the field. The output field is truncated if shorter than the input field or padded with spaces if longer.

5.3.2 Global Computational Fields

Global computational fields are indicated by gtype-qual of the form:

```
     [PIC] 9(p[,q])
```

with p+q total digits and q decimal digits.

Global computational fields can be equated in the conversion with several C-ISAM or Btrieve formats.

5.3.2.1 Unix Short Integer Fields

Unix short integer fields (2 bytes) are indicated by a utype-qual qualifier of the form:

```
     [PIC] SHORT
```

It is advisable that short fields are aligned on a two byte boundary on the Unix side of the conversion to facilitate the access of the fields from Unix applications.

5.3.2.2 Unix Long Integer Fields

Unix long integer fields (4 bytes) are indicated by a utype-qual qualifier of the form:

```
     [PIC] LONG
```

It is advisable that long fields are aligned on a four byte boundary on the Unix side of the conversion to facilitate the access of the fields from Unix applications.

5.3.2.3 Single Precision Floating Point Number

Single precision IEEE floating point numbers (4 bytes) are indicated by a utype-qual qualifier of the form:

```
     [PIC] FLOAT
```

It is advisable that float fields are aligned on a four byte boundary on the Unix side of the conversion to facilitate the access of the fields from Unix applications.

If the value of an IEEE floating point number exceeds the value that can be held in the Global computational field on the opposite side of the conversion, the computational field will be set to the most negative number it can be set to and an overflow error will be indicated by the conversion routine.

5.3.2.4 Double Precision Floating Point Number

Double precision IEEE floating point numbers (8 bytes) are indicated by a utype-qual qualifier of the form:

```
     [PIC] DOUBLE
```

It is advisable that double fields are aligned on an eight byte boundary on the Unix side of the conversion to facilitate the access of the fields from Unix applications.

If the value of an IEEE floating point number exceeds the value that can be held in the Global computational field on the opposite side of the conversion, the computational field will be set to the most negative number it can be set to and an overflow error will be indicated by the conversion routine.

5.3.2.5 C-ISAM or Btrieve Decimal Integer

C-ISAM or Btrieve decimal integers are indicated by a utype-qual qualifier of the form:

```
     [PIC] D(x,y)
```

with x **total** digits and y decimal digits.

Important note: This representation is different from the representation of Global computational fields (see section 5.3.2).

If the value of the Global computational field is too large to fit in a Unix decimal field then the field will be rounded by the conversion routine.

The internal representation of these numbers is described in the C-ISAM Programmers Manual (for Unix C-ISAM decimal integers) and the Btrieve Programmers Manual (for Btrieve decimal integers). The length of these fields is documented in the appropriate C-ISAM or Btrieve manual.

5.3.2.6 Btrieve Integer

Btrieve integers are indicated by a utype-qual qualifier of the form:

```
     [PIC] N(n)
```

where n is the number of bytes and MUST be an even number.

If the value of the Global computational field is too large to fit in a Btrieve integer field then the field will be rounded by the conversion routine.

5.3.2.7 Nothing Fields

Global computational fields can be converted to NOTHING fields (see section 5.3.6).

5.3.3 Date Fields

Global date fields are indicated by a gtype-qual of the form:

```
     [PIC] DATE
```

Global date fields can be converted to Unix date fields (4 bytes) containing the number of days since 31st December 1899. Global date fields can also be converted to Btrieve date fields. Both C-ISAM and Btrieve date fields are indicated by a utype-qual of the form:

```
     [PIC] NATDATE
```

If the value of a C-ISAM or Btrieve date is out of range for a valid Global date, the Global date field will be set to -1 during the file conversion. If the value of a Global date is out of range for a valid C-ISAM or Btrieve date, the C-ISAM or Btrieve date field will be set to #80000000 during the file conversion.

Important note: A Global date field containing binary-zeroes will be converted to a C-ISAM "NULL" date; and vice-versa. A global date field containing 8,000,000 (i.e. 8 million) will be converted to a large, but strictly valid, C-ISAM date; and vice-versa.

5.3.4 Time Fields

Global internal format time fields are indicated by a gtype-qual of the form:

```
     [PIC] TIME
```

Global time fields may only be used in Btrieve conversion tables to produce either Btrieve internal time or NOTHING fields (see section 5.3.6). Btrieve internal time fields are indicated by a utype-qual of the form:

```
     [PIC] NATTIME
```

5.3.5 Fixed Fields

Fixed fields are indicated by a gtype-qual or utype-qual of the form:

```
     FIXED val  len
```

where len is the length of the field containing the value val (expressed as "x" or #hh). Note that a length field MUST be supplied.

Fixed fields may only appear with a NOTHING field (see section 5.3.6) on the opposite side of the conversion line.

5.3.6 Nothing Fields

Nothing fields can appear on either side of the conversion and are indicated by a gtype-qual or utype-qual of the form:

```
     NOTHING
```

The principle use of NOTHING fields is to act as a counterpart to FIXED fields (see section 5.3.5) on the other side of the conversion. NOTHING fields may also be used when a field present on one side of the conversion has no meaning on the other side.

If the NOTHING field is present as conversion input, an empty output field of the type specified on the output side of the conversion is established, except for FIXED output fields, which are treated as normal. This means a character field containing spaces, and a date, time, computational, floating point or decimal field containing 0. If the NOTHING field is present as the conversion output then no output field is produced. For example, a Global ISAM link field (bytes 3 and 4 of the record) would have no meaning in a C-ISAM record and would be converted to a NOTHING field. On converting from C-ISAM to Global ISAM however, an empty link field as bytes (3 and 4) will be needed for use by Global ISAM.

5.3.7 Occurring Fields

It is possible to define an occurring group of fields using the following statement:

```
     NEXT no of records OCCURS n
```

where no of records refers to the number of records following the OCCURS statement and n refers to the number of iterations. The conversion routine will apply the indicated conversion records starting at the offset of the first field continually n times.

ALL THE FIELDS WITHIN THE OCCURS GROUP, IN BOTH THE INPUT AND OUTPUT RECORDS, MUST BE CONTIGUOUS.

5.4 Translation and Descending Key Fields

Translation and descending order key fields are mainly of use when producing conversion tables for DMAM database files.

5.4.1 Translation Fields

Translation fields are converted to C-ISAM or Btrieve format using the DMAM translation table passed to the Universal Channel Interface (UCI) via the "Intialise Channel" function (see Chapters 6 and 7).

Translation fields must always be converted as character fields.

Important note: The conversion record for a Global "translated" field MUST always be followed by another conversion record to convert the same Global field to second C-ISAM or Btrieve field. The second occurrence of the C-ISAM or Btrieve field will contain the 'original' value. For example, field FLD is a 5 byte translation key field which will occupy a maximum of 3 bytes once converted. A pair of conversions records are required to convert this field:

```
     FLD TRANS PIC X(5) 6 = FLD1 PIC X(3) 6
     FLD   PIC X(5) 6 = FLD2 PIC X(5) 9
```

DMAM translation tables are fully documented in the Global Cobol Data Management Manual.

5.4.2 Descending Key Fields

Descending order key fields are converted in such a way so that the records are retrieved in descending order. For the Btrieve UCI this involves no extra effort as Btrieve supports descending key segments. For the C-ISAM UCI, special conversion is necessary as descending key segments are not supported by Informix C-ISAM.

Descending key fields can be of any reasonable type.

Important note: In order to overcome a deficiency in C-ISAM, the conversion record for a Global "descending" field MUST always be followed by another conversion record to convert the same Global field to second C-ISAM field. The second occurrence of the C-ISAM field will contain the 'original' value. For example, field FLD is a 5 byte descending key field which will occupy a maximum of 3 bytes once converted. A pair of conversion records are required to convert this field:

```
     FLD DESC  PIC X(5) 6 = FLD1 PIC X(3) 6
     FLD       PIC X(5) 6 = FLD2 PIC X(5) 9
```

DMAM descending keys are fully documented in the Global Cobol Data Management Manual.

6. Interfacing to the MS-DOS Operating System

This chapter describes SVC-61 and the Universal Channel Interface (UCI). These interfaces allow MS-DOS files and Btrieve databases to be accessed directly from within Global System Manager and are only available on Global System Manager (MS-DOS and Windows) and Global System Manager (Novell NetWare) configurations.

The basic SVC-61 interface is used by both the MS-DOS SVC-61 File Converter (see Chapter 3) and the MS-DOS File Access Method (see Chapter 12 of the Global Development File Management Manual).

The arguments required by MS-DOS functions (see section 6.2) are described in any good MS-DOS Programming guide. We recommend the "DOS Programmer's Reference" by Dettmann and Johnson published by the Que Corp. (ISBN: 0-88022-790-7). **PLEASE CONSULT AN MS-DOS PROGRAMMER'S GUIDE FOR FURTHER INFORMATION REGARDING THE MS-DOS SYSTEM CALLS PROVIDED BY SVC-61.**

In addition to the MS-DOS functions described in section 6.2, SVC-61 also includes the Btrieve Universal Channel Interface. The Universal Channel Interface (UCI) can be considered an extension to SVC-61. Function codes 1900 to 2999 are used for UCI functions (see sections 6.3 and 6.4). **PLEASE CONSULT THE BTRIEVE PROGRAMMER'S GUIDE FOR FURTHER INFORMATION REGARDING THE BTRIEVE FUNCTIONS PROVIDED BY THE UCI.**

6.1 Using SVC-61 to Invoke an MS-DOS or Btrieve Function

An MS-DOS or Btrieve UCI function is called from Global System Manager using a Global Cobol statement of the form:

```
     SVC 61 USING ds
```

where ds is a request block.

6.1.1 SVC-61 DS Control Block

The ds request block for the SVC-61 functions listed in section 6.2 is defined below:

```
     01   DS
       02 DSFUNC    PIC X                        * Function code
       02 DSMODE    PIC X                        *   Subfunction   or
mode
       02 DSRES          PIC 9(4) COMP           * MS-DOS return code
       02 DSHAND    PIC 9(4) COMP           * File handle
       02 DSNAME    PIC PTR                 * Pointer to file name
       02 DSBUFF    PIC PTR                 * Pointer to buffer
       02 DSATTR    PIC X(2)                * File attributes
       02 DSNBYT    PIC 9(4) COMP           * Number of bytes moved
       02 DSPAR1    PIC X(2)                * Function specific
       02 DSPAR2    PIC X(2)                * Function specific
       02 DSPAR3    PIC X(2)                * Function specific
       02 DSPAR4    PIC X(2)                * Function specific

     01   FILLER REDEFINES DSPAR1
       02 DSP1H PIC X                            * Function specific
       02 DSP1L PIC X                            * Function specific
     01   FILLER REDEFINES DSPAR2
       02 DSP2H PIC X                            * Function specific
       02 DSP2L PIC X                            * Function specific

     77   DSP32B REDEFINES DSPAR1 PIC 9(9) COMP   * Expect WARNING
```

All fields are in Global Cobol format unless specified below.

The file handle, DSHAND, is returned by MS-DOS when the file is opened and **MUST NOT BE CHANGED IN ANY WAY.** If more than one MS-DOS file is to be opened at one time then a separate DS block should be allocated for each open MS-DOS file.

6.1.2 Btrieve UCI DS Control Block

The ds request block for the Btrieve UCI functions listed in sections 6.3 and 6.4 is defined below:

```
     01   DS
        02     DSFUNC    PIC 9(4) COMP            * Function code
        02     DSRECN    PIC S9(9) COMP           * Record number
        02     DSSTA1    PIC 9(2) COMP            * Returned status 1
        02     DSSTA2    PIC 9(2) COMP            * Returned status 2
        02     DSKPART   PIC 9(4) COMP            *  Max  no  of  key
     parts
        02     DSLNID    PIC X                        * Computer node
     id
        02     DSUSER    PIC 9(2) COMP            * User number
        02     DSEXTR    PIC X(10)                * Specialised data
        02     DSERR     PIC S9(9) COMP           * Result code errno
        02     DSDATA    PIC PTR                  *  Pointer to return
     data
        02     DSSIZE    PIC 9(4) COMP            * Max size of return
     data
        02     DSRET          PIC X(4)                *       Returned
     value
        02     DSPAR OCCURS 6 PIC X(4)            * Up to 6 parameters

     77    DSBTRV REDEFINES DSEXTR PIC 9(4) COMP  * Btrieve error code
     77    DSRETN REDEFINES DSRET PIC S9(9) COMP
     77    DSRETP REDEFINES DSRET PIC PTR

     01   FILLER REDEFINES DSPAR OCCURS 6
       02 DSPARP    PIC PTR
       02 FILLER         PIC X(2)

     01    FILLER REDEFINES DSPAR OCCURS 6
       02 DSPARN    PIC S9(9) COMP
```

All fields are in Global Cobol format.

Important note: The DS control block (copy-book DY), and all the other Global Cobol control blocks described in this chapter, are defined as copy-books within the S.IS copy library. The S.IS copy-library is NOT distributed with the V8.1 Global File Converters product although it is available on request.

6.2 SVC-61 Function Numbers for MS-DOS Functions

This section describes the MS-DOS functions that are available using the SVC-61 interface. **THIS SECTION SHOULD BE READ IN CONJUNCTION WITH AN MS-DOS PROGRAMMER'S GUIDE.**

Disclaimer: All the MS-DOS functions described below are passed directly to the MS-DOS operating system. In general, SVC-61 does not validate the DS request block before invoking the MS-DOS function. Software developers using this interface should be aware that misuse can cause serious problems.

In this section, all numbers suffixed by a "H" are in hexadecimal notation.

DSFUNC	Description
00H	Get version number of SVC-61
0EH	Select disk
19H	Get default disk drive
1BH	Get default drive allocation table information
1CH	Get specific drive allocation table information
2AH	Get system date
2BH	Set system date
2CH	Get system time
2DH	Set system time
30H	Get MS-DOS version number
36H	Get free disk space
39H	Create directory
3AH	Delete directory
3BH	Set default directory
3CH	Create or open file
3DH	Open old file
3EH	Close file
3FH	Read sequential
40H	Write sequential
41H	Delete file
42H	Position file pointer
43H	Change file attributes
47H	Get default directory
4EH	Get first directory entry
4FH	Get next directory entry
56H	Rename file
57H	Get or set file date and time
5AH	Create uniquely named file
5BH	Create new file

6.2.1 Get version number of SVC-61 (function 00H)

This function simply returns the version of SVC-61. It does not invoke any MS-DOS functions.

6.2.1.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	00H

6.2.1.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if this function is supported

	1 if this function not supported
DSPAR1	Portion of version number before the decimal point.
DSPAR2	Portion of version number after the decimal point.

6.2.1.3 Comments
This function returns the version number of SVC-61 (i.e. m.n) as two separate character fields. The number before the decimal point is returned in DSPAR1 as a character field. The number after the decimal point is returned in DSPAR2 as a character field. For example, the values returned by the 4.2 version of SVC-61 will be:

```
DSPAR1 " 4"
DSPAR2 "2 "
```

If this function returns an exception some SVC-61 functions will not be available. Furthermore, if this function returns an exception, the format of the DSATTR parameter for the following 3 functions must be supplied in "low-endian", Intel format rather than in "big-endian", Cobol format (i.e. the 2 bytes of the word field must be swapped):

3CH	Create or open file (section 6.2.15)
4EH	Get first matching directory entry (section 6.2.24)
5BH	Create new file (section 6.2.29)

6.2.2 Select disk (function 0EH)
This function changes the default MS-DOS disk drive.

6.2.2.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	0EH
DSMODE	Drive number (0 = A, 1 = B to 25 = Z)

6.2.2.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSMODE	Number of logical drives

6.2.2.3 Comments
Functions 0EH and 3BH both modify the current directory and will affect the operation of the SSD-FILE and DOS.PRI controllers if the pathname in the GSM.INI file is not an absolute pathname (i.e. if the pathname is a relative pathname). If the pathname in the GSM.INI file is a relative pathname, both the SSD-FILE and DOS.PRI controllers expect MS-DOS to remain in the "Global directory". This potential problem is easily solved by specifying full pathnames, including the drive letter, for the GSM.INI file entries for these controllers. For example, instead of including the following line in GSM.INI which is relative to the "Global directory":

```
SSD-FILE 0 GSM200
```

Use this line which defines an absolute pathname and makes the operation of the SSD-FILE controller impervious to any drive number or directory changes affected using SVC-61:

```
SSD-FILE 0 C:\GSM\GSM200
```

Note that neither function 0EH nor function 3BH will affect the operation of the DOS.PRI controller if the MS-DOS printer device defined in the GSM.INI file is physical printer (e.g. LPT1:) rather than a filename or directory name. Please refer to the Global Operating Manual (MS-DOS and Windows) or the Global Operating Manual (Novell NetWare) for further details.

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.3 Get default disk drive (function 19H)

This function returns the number of the current default drive.

6.2.3.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	19H

6.2.3.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSMODE	Current drive number (0 = A, 1 = B to 25 = Z)

6.2.3.3 Comments

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.4 Get default drive allocation table info. (function 1BH)

This function returns the basic information about the disk allocation for the disk in the default drive.

6.2.4.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	1BH

6.2.4.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSP1H	Media Descriptor Byte
DSP1L	Sectors per cluster
DSPAR2	Bytes per physical sector

DSPAR3	Clusters per disk

6.2.4.3 Comments
Note that the MS-DOS function returns a pointer to the Media Description Byte which would be very difficult for a Global Cobol program to use. SVC-61 returns the **actual value** of the Media Descriptor Byte in the DS block rather than the Intel format segment:offset **pointer**.

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.5 Get specific drive allocation table info. (function 1CH)
This function returns the basic information about the disk allocation for the disk in a specified drive.

6.2.5.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	1CH
DSMODE	Drive number (0 = current, 1 = A, 2 = B to 26 = Z)

6.2.5.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSP1H	Media Descriptor Byte
DSP1L	Sectors per cluster
DSPAR2	Bytes per physical sector
DSPAR3	Clusters per disk

6.2.5.3 Comments
Note that the MS-DOS function returns a pointer to the Media Description Byte which would be very difficult for a Global Cobol program to use. SVC-61 returns the **actual value** of the Media Descriptor Byte in the DS block rather than the Intel format segment:offset **pointer**.

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.6 Get system date (function 2AH)
This function returns the year, month, day, and day of the week from MS-DOS.

6.2.6.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	2AH

6.2.6.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSP1H	Day (1 - 31)
DSP1L	Month (1 - 12)
DSPAR2	Year (1980 - 2099)
DSPAR3	Day of week (0 = Sunday, 1 = Monday etc.)

6.2.6.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.7 Set system date (function 2BH)
This function sets the MS-DOS system date to the specified value without affecting the system time.

6.2.7.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	2BH
DSP1H	Day (1 - 31)
DSP1L	Month (1 - 12)
DSPAR2	Year (1980 - 2099)

6.2.7.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.7.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.8 Get system time (function 2CH)
This function gets the MS-DOS system time in hours, minutes, seconds and hundredths of seconds.

6.2.8.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	2CH

6.2.8.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSP1H	Hour (0 - 23)
DSP1L	Minutes (0 - 59)
DSP2H	Seconds (0 - 59)
DSP2L	Hundredths of seconds (0 - 99)

6.2.8.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.9 Set system time (function 2DH)
This function sets the MS-DOS system time to the specified hour, minute, second and hundredth of a second without affecting the system date.

6.2.9.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	2DH
DSP1H	Hour (0 - 23)
DSP1L	Minutes (0 - 59)
DSP2H	Seconds (0 - 59)
DSP2L	Hundredths of seconds (0 - 99)

6.2.9.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.9.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.10 Get MS-DOS version number (function 30H)
This function returns the MS-DOS version number.

6.2.10.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	30H
DSMODE	0 = Get OEM number 1 = Get MS-DOS version flag

6.2.10.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSP1H	Major version number (2,3 etc.)
DSP1L	Minor version number (2.1 = 01)
DSP2H	OEM number or MS-DOS version flag
DSP2L	24-bit serial number (high 8 bits)
DSPAR3	24-bit serial number (low 16 bits)

6.2.10.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.11 Get free disk space (function 36H)
This function returns the amount of space available on a designated drive along with other selected information about the drive.

6.2.11.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	36H
DSMODE	Drive number (0 = current, 1 = A, 2 = B to 26 = Z)

6.2.11.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSPAR1	Sectors per cluster
DSPAR2	Number of available clusters
DSPAR3	Bytes per sector
DSPAR4	Clusters on the drive

6.2.11.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.12 Create directory (function 39H)
This function creates a sub-directory at the specified drive and path location.

6.2.12.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	39H
DSNAME	Pointer to ASCIIZ path specification

6.2.12.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.12.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.13 Delete directory (function 3AH)
This function removes a sub-directory if it is empty.

6.2.13.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3AH
DSNAME	Pointer to ASCIIZ path specification

6.2.13.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.13.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.14 Set default directory (function 3BH)
This function sets the current or default directory to match the designated string.

6.2.14.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3BH
DSNAME	Pointer to ASCIIZ path specification

6.2.14.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.14.3 Comments
Functions 0EH and 3BH both modify the current directory and will affect the operation of the SSD-FILE and DOS.PRI controllers if the pathname in the GSM.INI file is not an absolute pathname (i.e. if the pathname is a relative pathname). If the pathname in the GSM.INI file is a relative pathname, both the SSD-FILE and DOS.PRI controllers expect MS-DOS to remain in the "Global directory". This potential problem is easily solved by specifying full pathnames, including the drive letter, for the GSM.INI file entries for these controllers. For example, instead of including the following line in GSM.INI which is relative to the "Global directory":

```
     SSD-FILE 0 GSM200
```

Use this line which defines an absolute pathname and makes the operation of the SSD-FILE controller impervious to any drive number or directory changes affected using SVC-61:

```
     SSD-FILE 0 C:\GSM\GSM200
```

Note that neither function 0EH nor function 3BH will affect the operation of the DOS.PRI controller if the MS-DOS printer device defined in the GSM.INI file is physical printer (e.g. LPT1:) rather than a filename or directory name. Please refer to the Global Operating Manual (MS-DOS and Windows) or the Global Operating Manual (Novell NetWare) for further details.

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.15 Create or open file (function 3CH)

This function creates the designated file if it does not exist, or truncates it to zero length if it does exist. If the open succeeds, this function returns a file handle (a 16-bit number) to reference the opened file.

6.2.15.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3CH
DSNAME	Pointer to ASCIIZ path specification
DSATTR	File attribute:
	#00 Normal file
	#02 Hidden file
	#04 System file
	#06 Hidden and system file

6.2.15.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSHAND	Returned file handle

6.2.15.3 Comments

If function 00H (see section 6.2.1) returns an exception, the attribute information, DSATTR, for functions 3CH, 4EH and 5BH, must be supplied in "low-endian", Intel format rather than in "big-endian", Cobol format.

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.16 Open old file (function 3DH)

This function opens the designated file and returns a file handle (a 16-bit number) to reference the opened file.

6.2.16.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3DH
DSNAME	Pointer to ASCIIZ path specification
DSMODE	Access and file-sharing mode

6.2.16.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSHAND	Returned file handle

6.2.16.3 Comments
Functions 3DH and 5BH are only supported for MS-DOS version 3.00, and later. If these functions are attempted on an earlier version of MS-DOS, SVC-61 will signal an exception and DSRES will contain 1 (invalid function number).

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.17 Close file (function 3EH)
This function closes a file previously open with file handles.

6.2.17.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3EH
DSHAND	File handle (from previous open or create)

6.2.17.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.17.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.18 Read sequential (function 3FH)
This function reads data from the file or device specified by the file handle argument. This data is written to a designated memory location.

6.2.18.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3FH
DSHAND	File handle (from previous open or create)
DSNBYT	Number of bytes to transfer
DSBUFF	Pointer to buffer area

6.2.18.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSNBYT	Number of bytes read

6.2.18.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.19 Write sequential (function 40H)
This function writes data to the file or device specified by the file handle argument.

6.2.19.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	40H
DSHAND	File handle (from previous open or create)
DSNBYT	Number of bytes to transfer
DSBUFF	Pointer to buffer area

6.2.19.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSNBYT	Number of bytes written

6.2.19.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.20 Delete file (function 41H)
This function deletes the specified file from the MS-DOS system.

6.2.20.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	41H
DSNAME	Pointer to ASCIIZ path specification

6.2.20.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.20.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.21 Position file pointer (function 42H)
This function changes the current location in the file, the file pointer, to a position relative to the start of file, end of file, or current position.

6.2.21.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	42H
DSHAND	File handle (from previous open or create)
DSMODE	Method code (binary value): #00 Offset from beginning of file #01 Offset from current position #02 Offset from end of file
DSP32B	Offset address
DSPAR1	Most significant part of offset in DSP32B
DSPAR2	Least significant part of offset in DSP32B

6.2.21.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSPAR1	Most significant part of offset (updated)
DSPAR2	Least significant part of offset (updated)

6.2.21.3 Comments
For function 42H, the two function specific parameters, DSPAR1 and DSPAR2, can be treated as a single quantity in PIC 9(9) COMP format, DSP32B.

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.22 Change file attributes (function 43H)
This function gets or sets the attributes of a file.

6.2.22.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	43H
DSMODE	Function mode: #00 Get file attributes #01 Set file attributes
DSATTR	File attribute information (if DSMODE = 1)
DSNAME	Pointer to ASCIIZ file specification

6.2.22.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSATTR	File attribute information (if DSMODE = 0)

6.2.22.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

The format of the file attribute byte is:

Bit	Mask	Meaning
0	#01	Read-only
1	#02	Hidden file
2	#04	System file
3	#08	Volume label
4	#10	Directory
5	#20	Archive
6	#40	Reserved
7	#80	Reserved (shareable flag on Novell server)

6.2.23 Get default directory (function 47H)
This function returns an ASCIIZ string with the full path of the current directory, not including the drive and leading backslash character (\).

6.2.23.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	47H
DSMODE	Drive number (0 = current, 1 = A, 2 = B to 26 = Z)
DSBUFF	Pointer to a 65-byte scratch buffer

6.2.23.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.23.3 Comments

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.24 Get first matching directory entry (function 4EH)

This function locates the first occurrence of a matching file name, given an ASCII string, which can include wild-cards.

6.2.24.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	4EH
DSATTR	Attribute to use in search: #00 Normal #02 Normal and hidden #04 Normal and system #06 Normal, hidden and system #08 Volume labels #10 Directories
DSNAME	Pointer to ASCIIZ file specification
DSBUFF	Pointer to buffer for DTA

6.2.24.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSBUFF	DTA returned to the area addressed by this pointer

6.2.24.3 Comments

If function 00H (see section 6.2.1) returns an exception, the attribute information, DSATTR, for functions 3CH, 4EH and 5BH, must be supplied in "low-endian", Intel format rather than in "big-endian", Cobol format.

For functions 4EH and 4FH, DSBUFF must point to a 43 byte area which will be used as the "DTA". An MS-DOS Programmer's guide will describe the information returned in this area. **The area must not be modified between calls to these functions.**

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.25 Get next directory entry (function 4FH)

After a successful call to function 4EH (see section 6.2.24), this function continues to find files that match the specified criteria. The DTA must retain the information originally placed there by the call to function 4EH.

6.2.25.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	4FH

DSBUFF	Pointer to buffer for DTA

6.2.25.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSBUFF	DTA returned to the area addressed by this pointer

6.2.25.3 Comments
For functions 4EH and 4FH, DSBUFF must point to a 43 byte area which will be used as the "DTA". An MS-DOS Programmer's guide will describe the information returned in this area. **The area must not be modified between calls to these functions.**

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.26 Rename file (function 56H)
This function renames a file or moves it to another directory on the same disk.

6.2.26.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	56H
DSNAME	Pointer to ASCIIZ current file name
DSBUFF	Pointer to ASCIIZ new file name

6.2.26.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code

6.2.26.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.27 Get or set file date and time (function 57H)
This function gets or sets the file's last modified date and time in the directory entry.

6.2.27.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	57H
DSMODE	Function mode: #00 Get the date and time #01 Set the date and time

DSHAND	File handle (from previous open or create)
DSPAR1	Time (if DSMODE = 1)
DSPAR2	Date (if DSMODE = 1)

6.2.27.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSPAR1	Time (if DSMODE = 0)
DSPAR2	Date (if DSMODE = 0)

6.2.27.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.28 Create uniquely named file (function 5AH)
This function creates a file with a guaranteed unique name in the specified directory.

6.2.28.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC 5AH

DSATTR File attribute:

#00	Normal file
#02	Hidden file
#04	System file
#06	Hidden and system file

DSNAME Pointer to ASCIIZ path specification, ending in backslash (\)

6.2.28.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSHAND	Returned file handle
DSNAME	Pointer to ASCIIZ file specification with file name appended

6.2.28.3 Comments
Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.2.29 Create new file (function 5BH)
This function creates a new file in the specified directory.

6.2.29.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	5BH
DSATTR	File attribute:
	#00 Normal file #02 Hidden file #04 System file #06 Hidden and system file
DSNAME	Pointer to ASCIIZ file specification

6.2.29.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or MS-DOS error code
DSHAND	Returned file handle

6.2.29.3 Comments
If function 00H (see section 6.2.1) returns an exception, the attribute information, DSATTR, for functions 3CH, 4EH and 5BH, must be supplied in "low-endian", Intel format rather than in "big-endian", Cobol format.

Functions 3DH and 5BH are only supported for MS-DOS version 3.00, and later. If these functions are attempted on an earlier version of MS-DOS, SVC-61 will signal an exception and DSRES will contain 1 (invalid function number).

Refer to an MS-DOS Programmer's guide for further information regarding this MS-DOS function.

6.3 SVC-61 Function Numbers for Btrieve Functions
It is not possible to use SVC-61 to access Btrieve files directly. All access must be performed using Btrieve UCI functions (see section 6.4). All SVC-61 functions in the range 1900 - 2999 must be invoked using the DS block described in section 6.1.2.

6.3.1 Miscellaneous SVC-61 Status Functions
The following functions are included in SVC-61 to provide status information:

Function	Operation
1900	Get UCI type
1901	Get Btrieve Parameters
1902	Set Btrieve Parameters

6.4 Btrieve Universal Channel Interface (UCI) Functions
The following SVC-61 functions (listed in decimal notation) are processed by the Btrieve Universal Channel Interface (UCI). The UCI is fully described in sections 6.8 and 6.9.

Function Operation

2000	Initialise channel
2001	Open channel (using existing Btrieve database)
2002	Close channel
2003	Clear channel
2004	Delete current record
2005	Read record
2006	Update current record
2007	Write a new record
2008	Change the current key
2009	Open channel (creating Btrieve database)
2010	Read index or file information
2011	Read channel debug operation
2012	Write channel debug operation
2013	This function is not supported in the Btrieve UCI
2014	This function is not supported in the Btrieve UCI
2015	Delete record (via record number)
2016	Update record (via record number)
2017	Write or update by key
2018	Add index
2019	Delete index
2020	Position record pointer
2021	Close delete Btrieve database
2022	Unlock records

6.5 SVC-61 Programming Notes

The following points should be considered when using SVC-61.

6.5.1 SVC-61 Interface Conventions

All file and directory names passed to SVC-61 must be ASCII strings terminated by a byte containing binary-zero. For example, the file "C:\DATA\MYFILE" can be established using the following Global Cobol statements:

```
77   NAME      PIC X(?)
     VALUE     "C:\DATA\MYFILE"
     VALUE      #00
```

6.5.2 Error Handling and Exceptions

If DSFUNC is set to an unrecognised value or if any MS-DOS or Btrieve functions return an error, SVC-61 will generate an exception. For normal MS-DOS errors, SVC-61 will generate exception 1. For a critical error, signalled by MS-DOS using INT 24H, SVC-61 will generate exception 2. In both cases, the error returned in DSRES will be one of the MS-DOS "Extended Error Codes" obtained by using the "Get Extended Error Code" MS-DOS function call (function 59H). Refer to your MS-DOS Programmer's guide for further details.

If DSFUNC is set to an unrecognised value, an exception 1 will be returned and DSRES will contain 1.

No exceptions are returned from the status routines 1900, 1901 and 1902.

6.5.3 Status Function 1900

Function 1900 returns the following values:

DSRETN   Comment

0	No UCI available
130	Btrieve UCI available
132	Btrieve UCI available (sic)

6.5.4 Status Function 1901

Function 1901 can be used to obtain various parameters that affect the operation of Btrieve. The actual parameter returned is determined by the value of DSPARP(1):

DSPARP(1)	Meaning
0	Return maximum number of key segments
1	Return value of "fileflags" for file creation
2	Return the Btrieve page size for file creation
3	Return the Btrieve Allocation size for file creation

The value of the specified parameter is returned in DSRETN.

6.5.5 Status Function 1902

Function 1902 can be used to set the various parameters that affect the operation of Btrieve. The actual parameter to set is determined by the value of DSPARP(1):

DSPARP(1)	Meaning
0	Set the maximum number of key segments
1	Set the value of "fileflags" for file creation
2	Set the Btrieve page size for file creation
3	Set the Btrieve Allocation size for file creation

The value of the specified parameter must be established in DSPARP(2).

6.6 Interface Control Block Specification

This section describes the format of some secondary control blocks used by the SVC-61 interface. To provide compatibility between the Btrieve UCI and the C-ISAM UCI (see Chapter 7), the indices and key specifications are passed in C-ISAM, not Btrieve, format.

6.6.1 Btrieve Index Information

The Btrieve index information returned by SVC-61 to the Global Cobol program is converted to the following format:

```
01   di
  02 diNKEY PIC 9(4) COMP          * No. of defined indexes
  02 diRECS PIC 9(4) COMP          * Record size in bytes
  02 diIDXS PIC 9(4) COMP          * Index node size
  02 diNREC PIC 9(9) COMP          * Number of data records
```

This control block is available as copy-book IW in the S.IS copy-library.

6.6.2 Btrieve Key Descriptions

Btrieve (C-ISAM) key descriptions are passed to the Cobol program in the following format.

```
     01   kp
      02  kpFLAG PIC X(2)                * Btrieve key type
      02  kpPART PIC 9(4) COMP           * Number of parts in key
      02  FILLER OCCURS n                * Where n is kpPART
        03     kpSTRT PIC 9(4) COMP           * Offset of key part
        03     kpLENG PIC 9(4) COMP           * Length of key part
        03     kpTYPE PIC X(2)                * Type of key part
```

This control block is available as copy-book IY in the S.IS copy-library.

6.7 User Constants for SVC-61 Functions

This section lists some useful constants that may be required when using SVC-61.

6.7.1 Short List of MS-DOS Error Codes

The following MS-DOS Extended Error Codes may be returned by SVC-61:

1	Invalid function
2	File not found
3	Path not found
4	No handles available
5	Access denied
6	Invalid handle
7	Memory control blocks destroyed
8	Insufficient memory
9	Invalid memory block address
10	Invalid environment
11	Invalid format
12	Invalid access code
13	Invalid data
14	Reserved
15	Invalid drive
16	Attempt to remove current directory
17	Not the same device
18	No more files
19	Disk write-protected
20	Unknown unit
21	Drive not ready
22	Unknown command
23	CFC error
24	Bad request structure length
25	Seek error
26	Unknown media type
27	Sector not found
28	Out of paper
29	Write fault
30	Read error
31	General failure
32	Sharing violation
33	Lock violation
34	Invalid disk change
35	FCB unavailable
36	Sharing buffer overflow
37	Code page mismatch
38	Error handling EOF

39	Handle disk full

6.7.2 Short List of Unix Compatible Error Codes

To provide compatibility between the MS-DOS Btrieve UCI and the Unix C-ISAM UCI, the following Unix error codes are returned by the Btrieve UCI:

EPERM	1	
ENOENT	2	
EINTR		4
EIO	5	
ENXIO	6	
EBADF	9	
EAGAIN	11	
EACCES	13	
EFAULT	14	
EBUSY	16	
EEXIST		17
EXDEV	18	
ENOTDIR	20	
EINVAL	22	
EMFILE	24	
ETXTBSY	26	
EFBIG		27
ENOSPC	28	
ESPIPE		29
EROFS		30
EMLINK	31	
EPIPE		32
ERANGE	34	
ENOLOCK	46	

6.7.3 List of C-ISAM codes

To provide compatibility between the MS-DOS Btrieve UCI and the Unix C-ISAM UCI, the following C-ISAM error codes are returned by the Btrieve UCI:

EDUPL	100
ENOTOPEN	101
EBADARG	102
EBADKEY	103
ETOOMANY	104
EBADFILE	105
ENOTEXCL	106
ELOCKED	107
EKEXISTS	108
EPRIMKEY	109
EENDFILE	110
ENOREC	111
ENOCURR	112
EFLOCKED	113
EFNAME	114
ENOLOK	115
EBADMEM	116
EBADCOLL	117
ELOGREAD	118
EBADLOG	119
ELOGOPEN	120
ELOGWRIT	121

ENOTRANS	122
ENOSHMEM	123
ENOBEGIN	124
ENONFS	125
EBADROWID	126
ENOPRIM	127
ENOLOG	128
EUSER	129
ENODBS	130
ENOFREE	131
EROWSIZE	132
EAUDIT	133
ENOLOCKS	134

6.7.3.1 Internal UCI Errors Returned in DSERR
The following errors, generated internally by the UCI, are returned in the DSERR field:

96	Btrieve TSR not loaded
97	Invalid op-code
98	Invalid partial key
99	UCI not available

6.7.4 Btrieve specific errors returned
The Unix and C-ISAM compatible errors, returned in DSERR, DSSTA1 and DSSTA2 (and listed in sections 6.7.2 and 6.7.3) reflect the general nature of the error and are quite sufficient for normal operation. However, for detailed debugging, the specific Btrieve error code is available in DSBTRV. The Btrieve Programmer's Manual contains a detailed description of these error codes.

6.7.4.1 Btrieve Engine Error Codes
Error codes 0 - 199 are returned from the Btrieve Engine:

01	Invalid operation parameter
02	I/O error
03	File not open
04	Cannot find key value
05	Duplicate key value
06	Key number invalid
07	Key number changed
08	Current positioning invalid
09	End of file
10	Key field not modifiable
11	Invalid file name
12	Cannot find file
13	Extended file error
14	Cannot create pre-image file
15	I/O error on pre-image file
16	Expansion error
17	Close error
18	Disk full
19	Unrecoverable error
20	Btrieve TSR not loaded
21	Key buffer too short
22	Data buffer too short
23	Position block is not 128 bytes
24	Invalid page, or data buffer size
25	Cannot create specified file
26	Number of keys invalid

27	Invalid key position
28	Invalid record length
29	Invalid key length
30	Not a Btrieve file
31	File already extended
32	File cannot be extended
33	Btrieve cannot unload
34	Extension filename invalid
35	Directory error
36	Transaction error
37	Another transaction is active
38	Transaction control file I/O error
39	Unmatched End/Abort Transaction
40	Trying to access too many files
41	Disallowed operation
42	Accelerated mode file not closed
43	Invalid record address
44	Invalid key path
45	Invalid key flags
46	File access denied
47	No. of files open exceeds maximum
48	Invalid alternate collating sequence
49	Invalid extended key type
50	File owner already set
51	Invalid owner name
53	Invalid language interface version
54	Corrupt variable length record
55	Invalid attribute in auto-increment key
56	Incomplete index
58	Compression buffer too short
59	Specified file already exists
60	Reject count reached
61	Work space too small
62	Incorrect descriptor
63	Invalid extended data buffer parameter
64	Filter limit reached
65	Incorrect filed offset
66	Too many open databases
67	Cannot open SQL data dictionaries
68	Cannot perform Delete Cascade operation
69	Corrupt data in Delete Cascade operation
71	Violation of Referential Integrity definition
72	Cannot open Referential Integrity referenced file
73	Referential Integrity definition out of synch
74	Aborted transaction
76	Conflict on reference file
77	Wait error
78	Deadlock condition
79	Programming error/System corruption
80	Record level conflict
81	Lock error
82	Positioning lost
83	Record outside of transaction
84	Record locked
85	File locked
86	File table full
87	Handle table full
88	Incompatible mode error
91	Server error
92	Transaction table full

93	Incompatible record lock types
94	Permission error
95	Session no longer valid
96	Communications environment error
97	Communication buffer too small
98	Internal transaction error
99	Requester cannot access server
100	No cache buffers available
101	Insufficient operating system memory
102	Insufficient stack
103	Chunk offset too big
104	Cannot find local information
105	Cannot create file with VAT's
106	Cannot get next chunk
107	Cannot update/retrieve chunks

6.7.4.2 Client-Based Btrieve Error Codes
Error codes 1000 - 1999 are returned from Client-based Btrieve:

1001	Multiple locks out of range
1002	Cannot allocate required memory
1003	Memory size too small
1004	Page size out of range
1005	Invalid pre-image file drive option
1006	Pre-image buffer size option out of range
1007	Open files option out of range
1008	Invalid configuration options
1009	Invalid transaction filename
1011	Compression buffer size out of range
1013	Task table full
1014	Stop warning encountered
1015	Invalid pointer parameter
1016	Btrieve already initialised
1017	Requester cannot find WBTRVRES.DLL

6.7.4.3 Btrieve Requester Error Codes
Error codes 2000 - 2999 are returned from the Btrieve Requester:

2001	Insufficient memory allocated
2002	Option invalid or out of range
2003	No local access to specified file
2004	SPX not installed
2005	Incorrect version of SPX
2006	No available SPX connection
2007	Pointer parameter invalid

6.7.5 Constants Used by Btrieve
The constants passed to the Btrieve UCI for read operations are identical to those used by the C-ISAM UCI:

ISFIRST	0
ISLAST	1
ISNEXT	2
ISPREV	3
ISCURR	4
ISEQUAL	5
ISGREAT	6
ISGTEQ	7
ISLOCK	0x100

The key type parameters passed to Btrieve UCI are those specified in the Btrieve Extended Key Types and Codes (refer to the Btrieve Programmer's Manual for further details). Only the first six data types are supported, types 0 - 5. When these data types are passed to an add index operation, if the top bit of the word is set (i.e. making it negative), the index will be collated in descending order. The following key data types are supported:

String type	0
Integer type	1
Float type	2
Data type	3
Time type	4
Decimal type	5

Where these data types have indeterminate data lengths, the key size passed in the index will determine the length required. Note that there are some restrictions on the lengths. For example, Integer types MUST have an even number of bytes.

The file type is not used by the UCI create file operation, which takes it's value of "file flags" from an internal constant. This constant can be obtained using UCI operation code 1901 (see section 6.5.4), and set using UCI operation code 1902 (see section 6.5.5).

6.8 The Universal Channel Interface (UCI)

The SVC-61 "base functions" (i.e. those with a function code in the range 1 to 1799) merely invoke a specific MS-DOS function. The Universal Channel Interface (UCI) supplements the basic SVC-61 interface by providing a set of functions to access Btrieve databases.

The differences between the UCI functions and the standard Btrieve calls is that the translation between the Global format record structure, and the Btrieve format record structure, is automatically performed inline by the UCI using a record conversion table created using RCBUILD (see Chapter 5). Partial records can be read/modified/written without requiring information about the rest of the record or allocating space for the whole record. You are STRONGLY recommended to use UCI functions, rather than basic Btrieve functions, in order to access raw Btrieve databases from within Global System Manager. Note also that the Native Indexed Access Method (NIAM) is available to perform higher-level functions on a Btrieve database (see Chapter 10 of the Global Development File Management Manual).

6.8.1 UCI Programming Notes

Before a call is made to the UCI both DSLNID and DSUSER must be established:

```
     MOVE $$LNID TO DSLNID    * Global System Manager node id
     MOVE $$USER TO DSUSER    * Global System Manager user number
```

6.8.1.1 UCI Locking Considerations

The UCI does NOT perform any explicit Btrieve file level locking. The UCI does assert file and record locks, not only to maintain its internal positioning, but also when a Read with Lock operation is attempted. If a UCI function returns with a lock error then the function should be retried as the condition is probably temporary (assuming no non-UCI process has a lock outstanding on the file).

All application level locking must be performed through the standard Global locking mechanisms (e.g. using the Global Cobol LOCK verb). If both a Global application, using the UCI, and a non-Global, MS-DOS application require write access to a particular MS-DOS file then the functions available in SVC-61 could be used to implement an external locking mechanism (which the non-Global MS-DOS application must be aware of).

6.8.1.2 UCI Error Considerations

The UCI returns an exception when DSERR is not zero. In this case DSRETN is usually -1.

If an error occurs whilst converting a record to, or from, Global format then an exception (or STOP CODE) will be returned and DSERR will be zero. If the error occurs on a read then the current position in the file is given by DSRECN. If the error occurs on a write then the record was not written.

If a returned error is not described explicitly for a particular function call, the file position cannot be guaranteed. The error returned **may** indicate the current file position. When an error code is returned by the UCI, an additional Btrieve error may also be returned, and these error descriptions should be referred to.

6.8.1.3 UCI Conversion Mode Parameter

The conversion mode parameter is an optional fourth parameter and Is for use together with the RECORD AREA1 and RECORD AREA2 conditionals in the conversion table (see section 5.2.2). It allows different record areas from the same record to be converted using the same conversion table. If the conversion mode is not supplied then a mode of 0 is assumed.

The allowed conversion modes are:

Mode	Description
0	Return/rewrite all fields as specified in the conversion table.
1	Return only those fields that are wholly within the first 512 bytes of the Global format data record.
2	Return/rewrite only those fields within RECORD AREA2. No other fields are converted. All conditionals are evaluated.
3	Return/rewrite onlythose fields within RECORD AREA1. No other fields are converted. All conditionals are evaluated.
4	Return/rewrite no fields at all. All fields are converted as specified in the conversion table with any output thrown away.

6.9 UCI Functions

This section describes every UCI function (see section 6.4) in complete detail.

6.9.1 Initialise channel (function 2000)

This function allocates a channel for the Btrieve database and translation table. An initialised channel is required before any other UCI function can be performed (with the exception of clear channel).

The DMAM translation table is described in the Global Cobol Data Management Manual. Any file using DMAM translation must follow the key field rules as described in the Global Cobol Data Management Manual and Chapter 5 of this manual. The DMAM translation table should not generally be required.

6.9.1.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2000
DSPARP(1)	Pointer to the MS-DOS pathname for the Btrieve database. The MS-DOS pathname is an ASCII string which is terminated by a byte of binary zero.
DSPARP(2)	Pointer to the conversion table.
DSPARP(3)	Pointer to the DMAM translation table or, if the DSPARN(3) redefinition of this field contains -1, no translation table is required.
DSLNID	$$LNID
DSUSER	$$USER

6.9.1.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	Channel number, or -1 if an error occurred.

6.9.1.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	There are no more free channels available.
EBADARG	One of the pointers is invalid.

6.9.2 Open channel using existing Btrieve file (function 2001)

This function opens the Btrieve database for the specified channel. The Btrieve database file must already exist. A channel must be opened before any operations can be performed on the Btrieve database. In addition to opening the database, this operation finds the first index in the file and sets the current position to the first record in that index.

6.9.2.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2001

DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

6.9.2.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.2.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised, the channel is already open, the channel does not belong to this user or the file does not exist.
?	Other errors may be caused by the file access and positioning operations. Refer to the Btrieve Programmer's Manual for further details.

6.9.3 Close channel (function 2002)
This function closes the Btrieve database for the specified channel and marks the channel as closed.

6.9.3.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2002
DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

6.9.3.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.3.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT The channel was not initialised, open or does not belong to this user.

6.9.4 Clear Channel (function 2003)
This function clears the specified channel, or all the channels allocated by this user. If the channel was open it is closed first. The channel is now free to be reallocated.

6.9.4.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2003
DSPARN(1)	Channel number. If DSPARN(1) contains -1 then all channels for this user are affected.
DSLNID	$$LNID
DSUSER	$$USER

6.9.4.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.4.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel number is out of the range of valid channel numbers.
?	A system error may occur closing one or more of the Btrieve files. The file may not have been closed properly but the channel has still been cleared. Refer to the Btrieve Programmer's Manual for further details.

6.9.5 Delete current record (function 2004)

This function deletes the current record from the database. The current record is specified by the record number returned from the most recent operation to successfully set the current position.

6.9.5.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2004
DSLNID	$$LNID
DSUSER	$$USER

6.9.5.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.5.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOCURR	There is no current position or the current record has already been deleted.

?	A system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.6 Read record (function 2005)

This function reads the specified record from the Btrieve database using the currently active index and the specified read mode. The portion of the record returned depends upon the conversion mode. This record will become the current record.

6.9.6.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2005
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record buffer.
DSPARN(3)	Read mode (as for isread, see section 6.7.5).
DSPARN(6)	Conversion mode (see section 6.8.1.3).
DSRECN	Record number.
DSLNID	$$LNID
DSUSER	$$USER

6.9.6.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSPARP(2)	Points to the record buffer as before. The buffer contains the record just read, unless a read error occurred, in which case no record is returned and the contents of the buffer remain unchanged. If an error occurred whilst converting the record into Global format, the contents of the buffer could have been corrupted.
DSRECN	Contains the record number of the new record.

6.9.6.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ELOCKED	The record was temporarily locked. The UCI only uses transient locks in which case the operation could be retried. But it could have been locked by another user.
EFLOCKED	The file was temporarily locked. The UCI only uses transient locks and so the operation should be retried.

ENOREC	The record could not be found.
EENDFILE	The current position is at the beginning or end of the index.
ENOCURR	The current position is not known or the current record does not exist.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details. The current position has been lost.

6.9.7 Update current record (function 2006)

This function updates the current record with the record data specified. If the key part of the record for the current index is changed, and if the key is defined as "modifiable" then the record is repositioned in the index.

6.9.7.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2006
DSPARP(2)	Pointer to record.
DSPARN(6)	Conversion mode (see section 6.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

6.9.7.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Record number of updated record.

6.9.7.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOCURR	The current record does not exist or the current position is not known.
ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried. Note that the record could be locked by another user.
EFLOCKED	The file was temporarily locked. The UCI only uses transient locks and so the operation should be retried.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details. Assume that the current position is not known.

6.9.8 Write a new record (function 2007)

This function writes the specified record to the Btrieve database as a new record. This record then becomes the current record. If the write fails, the current position remains unchanged.

6.9.8.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2007
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record to write.
DSPARN(6)	Conversion mode (see section 6.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

6.9.8.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Record number of new record.

6.9.8.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EFLOCKED	The file was temporarily locked. The UCI only uses transient locks and so the operation should be retried.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.9 Change the current key (function 2008)

This function changes the currently active index to that specified by the supplied key description. The specified key must correspond to an existing index. If KPPART = 0 then indexing is via record number (natural ordering). The current record becomes logically prior to the first record in the file.

6.9.9.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2008
DSPARN(1)	Channel number.

DSPARP(2) Pointer to key description for the primary index (see section 6.6.2).

DSLNID	$$LNID
DSUSER	$$USER

6.9.9.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.9.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EBADKEY	Part of the new key is invalid. The current index should remain active but it is suggested that the key is changed back explicitly.
?	If an error occurs changing to the new index it should be assumed that the current index and position are now unknown and attempts to access the file should not be made until these are reset.

6.9.10 Open channel creating Btrieve database (function 2009)
This function creates and then opens the Btrieve database for this channel. The Btrieve database is created with the specified record length and with the specified key as the primary index. The Btrieve database must not previously exist.

A channel must be opened before any operations can be performed on the Btrieve database.

If an error is returned the file is not created and the channel is not open.

6.9.10.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2009
DSPARN(1)	Channel number.
DSPARN(2)	Record length.

DSPARP(3) Pointer to key description for the primary index (see section 6.6.2).

DSLNID	$$LNID
DSUSER	$$USER

6.9.10.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.10.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel is not initialised, does not belong to this user or is already open.
EBADKEY	Part of the key description is invalid. The file has not been created and the channel is not open.
?	A Btrieve system error has occurred during the operation. Refer to the Btrieve Programmers's Manual for further details.

6.9.11 Read index or file information (function 2010)
If the index number is 0, the database dictionary information is returned, and should be accessed as a DI block. A value greater than 0 will return the key information for that index, and should be accessed as a KP block. For example, a value of 1 will return the first index for the file, (i.e. index number 0 in the Btrieve numbering scheme); a value of 2 will return index number 1, etc.

6.9.11.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2010
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to buffer for returned data.
DSPARN(3)	Index number. A value of 0 indicates that the files dictionary information is requested. A value of 1 indicates the primary index.
DSLNID	$$LNID
DSUSER	$$USER

6.9.11.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSPARP(2)	Pointer to the dictionary information, or the key description for a particular index (see section 6.6.2).

6.9.11.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
?	If an error occurs while reading the requested information from the file the requested information is not returned.

6.9.12 Read channel debug information (function 2011)

This function is used for debugging the UCI and should never be used by external developers. It returns channel status information from channel control block. This function does not check the User Number or Node id, thus any channel can be accessed. The information returned by this function is implementation specific.

THIS OPERATION SHOULD NEVER BE USED AND IS DOCUMENTED FOR COMPLETENESS ONLY.

6.9.12.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

```
DSFUNC    2011

DSPARN(1) Channel number

DSPARP(2) Debug operation code:

          0    return channel flags
          1    return file name
          2    return conversion table
          3    return translation table
          4    return record buffer
          5    return key description table
          6    return internal buffer.

DSPARN(3) Pointer to buffer for returned debug information

DSPARN(4) Length of returned data
```

6.9.12.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

```
DSRETN    0, or -1 if an error occurred
```

6.9.12.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	Invalid channel number.

6.9.13 Write channel debug information (function 2012)

This function is used for debugging the UCI and should never be used by external developers. It writes channel status information into a channel control block. This function does not check the User Number or Node id, thus any channel can be accessed. The information returned by this function is implementation specific.

THIS OPERATION SHOULD NEVER BE USED AND IS DOCUMENTED FOR COMPLETENESS ONLY.

6.9.13.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2012
DSPARN(1)	Channel number.
DSPARP(2)	Debug operation code:

0	set channel flags.
1	set file name.
2	set conversion table.
3	set translation table.
4	set record buffer.
5	set key description table.
6	set internal buffer.

DSPARN(3)	Pointer to buffer for sent debug info.
DSPARN(4)	Length of sent data.

6.9.13.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.13.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	Invalid channel number.

6.9.14 UCI Function 2013
This function is not supported in the Btrieve UCI.

6.9.15 UCI Function 2014
This function is not supported in the Btrieve UCI.

6.9.16 Delete record via record number (function 2015)
This function deletes the record with the specified record number from the Btrieve database.

6.9.16.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2015
DSPARN(1)	Channel number.
DSRECN	Record number to delete.
DSLNID	$$LNID

DSUSER	$$USER

6.9.16.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.16.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
EFLOCKED	The file was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
?	A Btrieve system error has occurred deleting the file. Refer to the Btrieve Programmer's Manual for further details.

6.9.17 Update record via record number (function 2016)
This function updates the record specified by the record number. It is possible to change the key values for this record.

6.9.17.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2016
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record.
DSPARN(6)	Conversion mode (see section 6.8.1.3).
DSRECN	Record number.
DSLNID	$$LNID
DSUSER	$$USER

6.9.17.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.17.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.

ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
EFLOCKED	The file was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.18 Write or update by key (function 2017)
This function updates the record specified by its key contents. If the record does not exist a new record is written. This record becomes the current record. If an error occurs the previous position is maintained. The current index must not allow duplicates.

6.9.18.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2017
DSPARP(2)	Pointer to record.
DSPARN(6)	Conversion mode (see section 6.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

6.9.18.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Updated or written record.

6.9.18.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EINVAL	The current index specifies natural ordering or allows duplicates.
ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
EFLOCKED	The file was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.19 Add index (function 2018)
This function adds the index specified by the key description to the Btrieve database.

6.9.19.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2018
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description for the primary index (see section 6.6.2).
DSLNID	$$LNID
DSUSER	$$USER

6.9.19.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.19.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EFLOCKED	The database is locked and this operation cannot gain the exclusive access it needs to update the index list.
EDUPL	This index is not a duplicate index but records have been found which have duplicate key values for this index.
EBADKEY	The key description being supplied for the new index is invalid.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.20 Delete index (function 2019)
This function deletes the index specified by the key description from the Btrieve database.

6.9.20.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2019
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description for the primary index (see section 6.6.2).
DSLNID	$$LNID
DSUSER	$$USER

6.9.20.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.20.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EFLOCKED	The file is locked and this operation cannot gain the exclusive access it needs to update the index list.
EBADKEY	The key description being supplied for the new index is invalid.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.21 Position record pointer (function 2020)
This function sets the current record position according to the specified search criteria without the record being read. This function enables you to position on a partial key by specifying the last key segment and the partial length of this segment.

A value of -1 in DSPARN(4) signifies that the whole key is to be used.

A value of 0 in DSPARN(5) signifies that the whole segment is to be used.

6.9.21.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2020
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description for the primary index (see section 6.6.2).
DSPARN(3)	Position mode (modes as for isread - see section 6.7.5).
DSPARN(4)	Last key segment.
DSPARN(5)	Length of last key segment (0-complete).
DSPARN(6)	Conversion mode (see section 6.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

6.9.21.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSPARP(2)	Point to the record buffer as before. The buffer contains the key value for the current record. The contents of this buffer remain unchanged unless an error occurs whilst converting the record into Btrieve format when the contents of the buffer may be unpredictable.
DSRECN	Record number.

6.9.21.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOREC	There is no record matching the search criteria, on the database.
EENDFILE	A next/previous type operation has reached the end of it's possible records.
98	The partial index specification passed to the UCI is invalid. Either the partial key is greater than the number of segments in the key, or the partial segment offset is beyond the end of the segment, or the end of the partial segment occurs within a field which cannot be split (e.g. a floating point number).
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.22 Close delete Btrieve file channel (function 2021)

This function closes the file channel specified and deletes the Btrieve database associated with that channel.

6.9.22.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2021
DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

6.9.22.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.22.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

6.9.23 Unlock records in Btrieve database (function 2022)

This function unlocks any outstanding locks on the file channel specified.

6.9.23.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2022
DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

6.9.23.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

6.9.23.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOTOPEN	The channel is not open.
?	A Btrieve system error has occurred. Refer to the Btrieve Programmer's Manual for further details.

7. Interfacing to the Unix Operating System

This chapter describes SVC-61 and the Universal Channel Interface (UCI). These interfaces allow Unix files and Informix C-ISAM databases to be accessed directly from within Global System Manager and are only available on Global System Manager (Unix) configurations.

The basic SVC-61 interface is used by both the Unix SVC-61 File Converter (see Chapter 4) and the Unix File Access Method (see Chapter 11 of the Global Development File Management Manual).

The arguments required by the Unix functions (see section 7.2) are described in detail in the Unix "System Calls and Library Routines(S)" for the Unix system. **PLEASE CONSULT THIS MANUAL FOR FULL INFORMATION REGARDING THE UNIX SYSTEM CALLS PROVIDED BY SVC-61.**

In addition to the Unix functions described in section 7.2, SVC-61 also allows C-ISAM functions to be executed (see section 7.3). Although all Global System Manager (Unix) configurations support the SVC-61 Unix functions (function codes in the range 1 to 999), the SVC-61 C-ISAM functions (function codes 1000 to 1999) are not supported on all Global System Manager (Unix) configurations. Please consult your Global Configuration Notes for further details.

The arguments required by the C-ISAM functions are documented in the "C-ISAM Programmer's Manual" supplied by Informix. **PLEASE CONSULT THIS MANUAL FOR FULL INFORMATION REGARDING THE C-ISAM FUNCTIONS PROVIDED BY SVC-61.**

The Universal Channel Interface (UCI) can be considered an extension to SVC-61. Function codes 2000 to 2999 are used for UCI functions (see section 7.4). The UCI is not supported on all Global System Manager (Unix) configurations. Please consult your Global Configuration Notes for further details.

The arguments required by the UCI functions are documented in the "C-ISAM Programmer's Manual" supplied by Informix. **PLEASE CONSULT THIS MANUAL FOR FULL INFORMATION REGARDING THE C-ISAM FUNCTIONS PROVIDED BY THE UCI.**

Important note-1: Prior to V8.1, some Global System Manager (Unix) configurations included the D-ISAM functions (supplied by Byte Designs Inc.) as an alternative to the C-ISAM functions (supplied by Informix Software Inc.). The use of D-ISAM as a substitute for C-ISAM for some Global System Manager (Unix) configurations has been removed for Global System Manager V8.1. A special SVC-61 function call (function 1900) is available to allow you to determine if the C-ISAM functions are supported on a particular configuration.

Important note-2: The SVC-61 interface for Global System Manager (Unix) supports both 16-bit Cobol/Speedbase programs and 32-bit Speedbase programs. Developers of 32-bit applications should be aware of the differences in the SVC-61 interface.

7.1 Using SVC-61 to Invoke a Unix or C-ISAM Function

A Unix, C-ISAM or UCI function is called from Global System Manager using a Global Cobol statement of the form:

```
    SVC 61 USING ds
```

where ds is a request block.

7.1.1 SVC-61 DS Control Block

The format of the ds request block for the SVC-61 functions depends on the application environment.

7.1.1.1 SVC-61 DS Control Block for 16-bit applications

The ds request block a 16-bit application is defined below:

```
     01   DS
       02 DSFUNC    PIC 9(4) COMP       * Function code
       02 DSRECN    PIC S9(9) COMP      * C-ISAM isrecnum
       02 DSSTA1    PIC 9(2) COMP       * C-ISAM isstat1
       02 DSSTA2    PIC 9(2) COMP       * C-ISAM isstat2
       02 DSKPART   PIC 9(4) COMP       *  C-ISAM  max  number  of  key
parts
       02 DSLNID    PIC X                    * System ID
       02 DSUSER    PIC 9(2) COMP       * User number
       02 DSEXTR    PIC X(10)           * Specialised data
       02 DSERR     PIC S9(9) COMP      * Result code errno
       02 DSDATA    PIC PTR             * Pointer to return data
       02 DSSIZE    PIC 9(4) COMP       * Max size of return data
       02 DSRET          PIC X(4)            * Returned value
       02 DSPAR OCCURS 6 PIC X(4)       * Up to 6 parameters

     77   DSRETN REDEFINES DSRET PIC S9(9) COMP
     77   DSRETP REDEFINES DSRET PIC PTR

     01    FILLER REDEFINES DSPAR OCCURS 6
       02 DSPARP    PIC PTR
       02 FILLER         PIC X(2)

     01    FILLER REDEFINES DSPAR OCCURS 6
       02 DSPARN    PIC S9(9) COMP

     01    FILLER REDEFINES DSEXTR
       02 DSUCI     PIC 9(4) COMP       * UCI error code
       02 DSFD           PIC 9(4) COMP       *  UCI  specific  C-ISAM  FD
number
```

7.1.1.2 SVC-61 DS Control Block for 32-bit applications

The ds request block a 32-bit application is defined below:

```
     01   DS
       02 DSFUNC    PIC 9(4) COMP       * Function code
     *
     * As explained below 4000 must be added to the function code to
     * indicate a 32-bit SVC-61 operation.
     *
       02 DSRECN    PIC S9(9) COMP      * C-ISAM isrecnum
       02 DSSTA1    PIC 9(2) COMP       * C-ISAM isstat1
       02 DSSTA2    PIC 9(2) COMP       * C-ISAM isstat2
       02 DSKPART   PIC 9(4) COMP       *  C-ISAM  max  number  of  key
parts
       02 DSLNID    PIC X                    * System ID
       02 DSUSER    PIC 9(2) COMP       * User number
       02 DSEXTR    PIC X(10)           * Specialised data
       02 DSERR     PIC S9(9) COMP      * Result code errno
       02 FILLER         PIC SPT             * Unused (was DSDATA)
```

```
        02 DSSIZE    PIC 9(4) COMP        * Max size of return data
        02 DSRET          PIC X(4)             * Returned value
        02 DSPAR OCCURS 6 PIC X(4)        * Up to 6 parameters
        02 DS32DAT   PIC PTR              * Pointer to return data
        02 DS32ERR   PIC X                     * 32-bit addressing error
code

      77   DSRETN REDEFINES DSRET PIC S9(9) COMP
      77   DSRETP REDEFINES DSRET PIC PTR

      01   FILLER REDEFINES DSPAR OCCURS 6
        02 DSPARP    PIC PTR

      01   FILLER REDEFINES DSPAR OCCURS 6
        02 DSPARN    PIC S9(9) COMP

      01    FILLER REDEFINES DSEXTR
        02 DSUCI     PIC 9(4) COMP        * UCI error code
        02 DSFD           PIC 9(4) COMP        * UCI specific C-ISAM FD
number
```

All fields are in Global Cobol format.

Important note: The 16-bit DS control block (copy-book DY), and all the other Global Cobol control blocks described in this chapter, are defined as copy-books within the S.IS copy library. The S.IS copy-library is NOT distributed with the V8.1 Global File Converters product although it is available on request.

7.2 SVC-61 Function Numbers for Unix Functions

This section describes the Unix functions that are available using the SVC-61 interface. **THIS SECTION SHOULD BE READ IN CONJUNCTION WITH THE UNIX "SYSTEM CALLS AND LIBRARY ROUTINES(S)" MANUAL.**

Disclaimer: All the Unix functions described below are passed directly to the Unix operating system. In general, SVC-61 does not validate the DS request block before invoking the Unix function. Software developers using this interface should be aware that misuse can cause serious problems.

DSFUNC		Operation (parameters)	[DSRET returned]
10	4010	opendir (ptr to dir name)	[dir stream]
11	4011	readdir (dir stream)	[ptr to dir entry] (see section 7.6.1)
12	4012	telldir (dir stream)	[location]
13	4013	seekdir (dir stream, location)	[ ]
14	4014	rewinddir (dir stream)	[ ]
15	4015	closedir (dir stream)	[ ]
16	4016	chdir (ptr to path name)	[0 or -1]
17	4017	mkdir (ptr to path name, mode)	[0 or -1]
18	4018	rmdir (ptr to path name)	[0 or -1]
19[c]	4019	fopen (ptr to file name, type)	[file stream]
20	4020	freopen (ptr to file name, type, stream)	[new stream]
21	4021	fdopen (file descriptor, type)	[file stream]
22 [c]	4022	fclose (file stream)	[0 or EOF]
23	4023	fflush (file stream)	[0 or EOF]
24	4024	ferror (file stream)	[0 or not 0]

25	4025	feof (file stream)	[0 or not 0]
26	4026	clearerr (file stream)	[ ]
27	4027	fileno (file stream)	[file descriptor]
28[C]	4028	fread (ptr to buf, size, items, file stream)	[amount read]
29[C]	4029	fwrite (ptr to buf, size, items, file stream)	[amount written]
30[C]	4030	fseek (file stream, offset, from)	[0 or not 0]
31	4031	rewind (file stream)	[ ]
32	4032	ftell (file stream)	[offset of current position from BOF]
33	4033	getc (file stream)	[character or EOF]
34	4034	fgetc (file stream)	[character or EOF]
35	4035	getw (file stream)	[word or EOF]
36	4036	access (ptr to path name, amode)	[0 or -1]
37	4037	chmod (ptr to path name, mode)	[0 or -1]
38	4038	chown (ptr to path name, owner, group)	[0 or -1]
39	4039	umask (cmask)	[previous mask]
40[B]	4040	mknod (ptr to path name, mode, dev)	[0 or -1]
41	4041	mktemp (template)	[template]
42	4042	creat (ptr to path name, mode)	[file descriptor or -1]
43	4043	not used	
44	4044	lockf (file descriptor, function, size)	[0 or -1]
45	4045	open (ptr to path name, oflag, mode)	[file descriptor or -1]
46	4046	dup (file descriptor)	[new file descriptor or -1]
47	4047	read (file des, ptr to buf, no. of bytes)	[amount read or -1]
48	4048	write (file des, ptr to buf, no. of bytes)	[amount written or -1]
49	4049	lseek (file descriptor, offset, whence)	[location or -1]
50	4050	stat (ptr to path, buffer)	[0 or -1] (see section 7.6.2)
51	4051	fstat (file descriptor, buffer)	[0 or -1] (see section 7.6.2)
52	4052	fcntl (file descriptor, command, arg)	[varies with cmd] (see section 7.6.3)
53	4053	link (path1, path2)	[0 or -1]
54	4054	unlink (path)	[0 or -1]
55	4055	close (file)	[0 or -1]
56[A C]	4056	fopen (ptr to file name, pointer to type)	[file stream}
57[A]	4057	freopen (ptr to file, ptr to type, stream)	[new stream]
58[A]	4058	fdopen (file descriptor, ptr to type)	[file stream]
59	4059	getenv (ptr to environment variable,size)	[DSDATA points to the value of the variable]
60	4060	getcwd (ptr to buffer, size)	[path name of current dir]
61	4061	putc (character, file stream)	[0 or EOF]
62[A]	4062	putc (ptr to char, file stream)	[0 or EOF]

Note-A Functions 56, 57 and 58 provide the same functionality as functions 19, 20 and 21, respectively, except that for functions 56, 57 and 58, the second variable (type) is passed via a pointer. Function 62 provides the same functionality as function 61, except that for functions 62, the first variable (char) is passed via a pointer.

Note-B Misuse of the mknod function can cause serious problems with the Unix system. To prevent accidental or deliberate misuse of this function, SVC-61 will not allow mknod to create "Character Special" or "Block Special" Unix files.

Note-C These stream i/o operations are mapped to the corresponding raw i/o operations by SVC-61 with automatic conversion of the associated parameters, where necessary:

Stream i/o operation	Raw i/o operation
fopen	open
fclose	close
fread	read
fwrite	write
fseek	lseek

To by-pass the automatic operation conversion to call the desired Unix stream i/o function, add 200 to the basic opcode (e.g. opcode 219 will invoke the fopen function).

7.3 SVC-61 Function Numbers for C-ISAM Functions

This section describes the C-ISAM functions that are available using the SVC-61 interface. **THIS SECTION SHOULD BE READ IN CONJUNCTION WITH THE "C-ISAM PROGRAMMER'S MANUAL" SUPPLIED BY INFORMIX SOFTWARE LTD.**

Disclaimer: All the functions described below are passed directly to C-ISAM. SVC-61 does not validate the DS request block before invoking the C-ISAM function. Software developers using this interface should be aware that misuse can cause serious problems.

Function	Operation (parameters)
1000 5000	isaddindex (file descriptor, key description)
1001 5001	isaudit (file descriptor, ptr to file name, mode)
1003 5003	isbuild (ptr to file name, record length, ptr to key description, mode)
1004 5004	isclose (file descriptor)
1007 5007	isdelcurr (file descriptor)
1008 5008	isdelete (file descriptor, ptr to record)
1009 5009	isdelindex (file descriptor, ptr to key description)
1010 5010	isdelrec (file descriptor, record number)
1011 5011	iserase (pointer to file name)
1012 5012	isflush (file descriptor)
1013 5013	isindexinfo (file descriptor, ptr to buffer, number)
1014 5014	islock (file descriptor)
1017 5017	isopen (ptr to file name, mode)
1018 5018	isread (file descriptor, ptr to record, mode)
1020 5020	isrelease (file descriptor)
1021 5021	isrename (ptr to old name, ptr to new name)
1022 5022	isrewcurr (file descriptor, pointer to record)

1023 5023	isrewrec (file descriptor, record number, ptr to record)
1024 5024	isrewrite (file descriptor, ptr to record)
1026 5026	issetunique (file descriptor, uniqueid)
1027 5027	isstart (file descriptor, ptr to key description, length, ptr to record, mode)
1028 5028	isuniqueid (file descriptor, uniqueid)
1029 5029	isunlock (file descriptor)
1030 5030	iswrcurr (file descriptor, ptr to record)
1031 5031	iswrite (file descriptor, ptr to record)

All functions return a DSRET result of [0 or -1], except for functions 1003 and 1017 which both return a DSRET result of [file description or -1].

7.3.1 Extended C-ISAM Functions
The following C-ISAM functions are not available with D-ISAM:

Function	Operation (parameters)
1002 5002	isbegin ()
1005 5005	iscluster (file descriptor, ptr to new key description)
1006 5006	iscommit ()
1015 5015	islogclose ()
1016 5016	islogopen (ptr to logname)
1019 5019	isrecover ()
1025 5025	isrollback ()

7.3.2 Unsupported D-ISAM Functions
The following D-ISAM functions are NOT supported by the SVC-61 interface:

Function	Operation (parameters)
1100 5100	isrelcurr (file descriptor)
1101 5101	isrelrec (file descriptor, record)
1102 5102	isseecurr (file descriptor)
1103 5103	isseekey (file descriptor)

7.3.3 Miscellaneous SVC-61 Status Functions
The following functions are included in SVC-61 to provide status information.

Function	Operation (parameters)	[DSRET returned]
1900 5900	gsmisam ()	[0, 1, 2, 0x80, 0x81 or 0x82]
1901 5901	gsmvalue (mode)	[value]

7.4 C-ISAM Universal Channel Interface (UCI) Functions
The following SVC-61 functions are processed by the C-ISAM Universal Channel Interface (UCI). The UCI is fully described in sections 7.8 and 7.9.

Function	Operation
2000 6000	Initialise channel
2001 6001	Open channel (using existing C-ISAM database)

2002	6002	Close channel
2003	6003	Clear channel
2004	6004	Delete current record
2005	6005	Read record
2006	6006	Update current record
2007	6007	Write a new record
2008	6008	Change the current key
2009	6009	Open channel (creating C-ISAM file)
2010	6010	Read index or file information
2011	6011	Read channel debug operation
2012	6012	Write channel debug operation
2013	6013	Read records from RS file
2014	6014	Write records to RS file
2015	6015	Delete record (via record number)
2016	6016	Update record (via record number)
2017	6017	Write or update by key
2018	6018	Add index
2019	6019	Delete index
2020	6020	Position record pointer
2021	6021	Close delete C-ISAM file channel
2022	6022	Unlock records on C-ISAM file channel

7.5 SVC-61 Programming Notes
The following points should be considered when using SVC-61.

7.5.1 SVC-61 Interface Conventions
All file and directory names passed to SVC-61 must be ASCII strings terminated by a byte containing binary-zero. For example, the name "data/myfile" can be established using the following Global Cobol statements:

```
     77   NAME      PIC X(?)
          VALUE     "data/myfile"
          VALUE     #00
```

7.5.2 Error Handling and Exceptions
If DSFUNC is set to an unrecognised value, SVC-61 will generate an exception code 2 with a value of -1 in DSERR.

If a Unix function returns an error, SVC-61 will generate an exception code 1 with the value of the Unix variable errno in DSERR. Common values for errno are detailed in section 7.7.3 but please refer to the relevant Unix documentation for full information.

No exceptions are returned from the two status functions (function codes 1900 and 1901).

7.5.3 Opening and Closing Unix File Channels
It is extremely important to ensure that all Unix directories and files opened using SVC-61 functions are subsequently closed. An end of job routine MUST be provided to close any Unix files that have been opened using SVC-61 functions.

Similiarly, it is essential that no attempt is made to use SVC-61 to close Unix file channels which have NOT been opened using previous SVC-61 functions.

Failure to obey this advice may result in unpredictable data corruption.

7.5.4 File Descriptor Returned by Open Function

Function 45 (open) returns a Unix File Descriptor (FD). The value of the FD is normally passed as a parameter to SVC-61 for subsequent operations (e.g. read) on the open file.

7.5.5 C-ISAM and D-ISAM isstat1 and isstat2

The C-ISAM and D-ISAM functions return the value of isrecnum when they return to the calling program. However D-ISAM routines do not return isstat1 and isstat2. This dichotomy has been resolved within SVC-61 by returning the following:

For system errors isstat1 = 3;

For all C-ISAM errors (99 < iserrno < 135) isstat1 and isstat2 are set up as specified in the table in Appendix C of the Informix C-ISAM Programmers Manual.

7.5.6 Buffer for Directory Information

Function 11 (readdir) requires a pointer to an area where the directory information is returned to be established in DSDATA. The length of the area must be established in DSSIZE.

7.5.7 Buffer for Environment Variable Information

Functions 59 (getenv) and 60 (getcwd) require a pointer to an area where the variable information is returned to be established in DSDATA. The length of the area must be established in DSSIZE. The actual length of the variable is returned in DSSIZE. **Important note**: An extra 2 bytes at the end of the area must be allocated for Unix to insert zero terminator(s).

7.5.8 Buffer for Key Descriptor

Function 1103 (isseekey) requires a pointer to an area where the key descriptor is returned to be established in DSDATA. The length of the area must be established in DSSIZE.

7.5.9 Status Function 1900

Function 1900 returns the following values:

DSRETN	C-ISAM/D-ISAM	UCI?	Comment
0	No	No	Non C-ISAM configuration
1	C-ISAM	No	Non-standard BACNAT
2	D-ISAM	No	Non-standard BACNAT
128	No	Yes	Non-standard BACNAT
129	C-ISAM	Yes	C-ISAM configuration
130	D-ISAM	Yes	Non-standard BACNAT

Note that the byte-value returned in DSRETN is really a collection of bit flags:

Bit	Meaning
0	1 = C-ISAM supported 0 = C-ISAM not supported
1	1 = D-ISAM supported 0 = D-ISAM not supported

```
     2 - 6          Reserved for future use

     7         1 = UCI supported
               0 = UCI not supported
```

7.5.10 Status Function 1901

Function 1901 returns the value of various constants. Mode 1 returns the constant NPARTS and mode 2 returns the constant MAXKEYSIZE (C-ISAM only).

7.6 Interface Control Block Specifications

This section describes the format of some secondary control blocks used by the SVC-61 interface. Note that SVC-61 converts this information from a C structure (of unpredictable format) to a standard Global Cobol control block.

7.6.1 Directory Information

Directory information returned by SVC-61 to the Global Cobol program is converted to the following format (see the Unix file format DIRENT(F) for further details):

```
     01   de
       02 deINO          PIC 9(9) COMP  * Unique file number
       02 deRECL    PIC 9(4) COMP  * Length of name
       02 deNAME    PIC X(?)       * Name (NULL terminated)
```

This control block is available as copy-book DZ in the S.IS copy-library.

7.6.2 Status Information

Status information returned by SVC-61 to the Global Cobol program is converted to the following format (see the Unix function STAT(S) and file format STAT(F) for further details):

```
     01   st
       02 stMODE    PIC X(2)       * File mode
       02 stINO          PIC 9(4) COMP  * Inode number
       02 stDEV          PIC X(2)       * ID of device
       02 stRDEV    PIC X(2)       * Special ID device
       02 stLINK         PIC 9(4) COMP  * Number of links
       02 stUID          PIC 9(4) COMP  * User ID
       02 stGID          PIC 9(4) COMP  * Group ID
       02 stSIZE         PIC 9(9) COMP  * File size
       02 stATIM    PIC 9(9) COMP  * Time of last access
       02 stMTIM    PIC 9(9) COMP  * Last data modification
       02 stCTIM    PIC 9(9) COMP  * Since Jan 1 1970
```

This control block is available as copy-book IU in the S.IS copy-library.

7.6.3 File Segment Locking Information

File segment locking information returned by SVC-61 to the Global Cobol program is converted to the following format (see the Unix function FCNTL(S) and file format FCNTL(F) for further details):

```
     01   fl
       02 flTYPE PIC 9(4) COMP          * Locking type
       02 flWHEN PIC 9(4) COMP          * Starting offset
       02 flSTRT PIC 9(9) COMP          * Relative offset
```

```
        02 flLEN PIC 9(9) COMP      * Size
        02 flSYID PIC 9(4) COMP          * RFS system ID
        02 flPID PIC 9(9) COMP      * Process ID
```

This control block is available as copy-book IV in the S.IS copy-library.

7.6.4 C-ISAM Index Information

C-ISAM index information returned by SVC-61 to the Global Cobol program is converted to the following format (see the "Determining Index Structures" section in Chapter 2 of the Informix C-ISAM Programmer's Manual for further details):

```
     01   di
       02 diNKEY    PIC 9(4) COMP  * No. of defined indexes
       02 diRECS    PIC 9(4) COMP  * Record size in bytes
       02 diIDXS    PIC 9(4) COMP  * Index node size
       02 diNREC    PIC 9(9) COMP  * Number of data records
```

This control block is available as copy-book IW in the S.IS copy-library.

7.6.5 C-ISAM Key Descriptions

C-ISAM key descriptions returned by SVC-61 to the Global Cobol program are converted to the following format (see the "Key Structures" section in Chapter 2 of the Informix C-ISAM Programmer's Manual for further details):

```
     01   kp
       02 kpFLAG    PIC X(2)       * Describes compression
       02 kpPART    PIC 9(4) COMP  * Number of parts in key
       02 FILLER OCCURS 8              * Where n is kpPART
         03    kpSTRT    PIC 9(4) COMP  * Offset of key part
         03    kpLENG    PIC 9(4) COMP  * Length of key part
         03    kpTYPE
          04   kpCTYP    PIC 9(4) COMP  * Type of key part
```

This control block is available as copy-book IY in the S.IS copy-library.

7.7 User Constants for SVC-61 Functions

This section lists some useful constants that may be required when using SVC-61.

7.7.1 Constants for Specific Unix Functions

The following constants may be required when calling Unix functions via SVC-61:

```
     fseek               0 = from beginning
                    1 = from current position
                    2 = from end of file

     lseek               0 = from beginning
                    1 = from current position
                    2 = from end of file

     open           O_RDONLY       0000 (octal)
                    O_WRONLY       0001 (octal)
                    O_RDWR         0002 (octal)
```

	O_NDELAY	0004 (octal)
	O_APPEND	0010 (octal)
	O_SYNC	0020 (octal)
	O_CREAT	0400 (octal)
	O_TRUNC	1000 (octal)
	O_EXCL	2000 (octal)
fcntl	O_RDONLY	0000 (octal)
	O_WRONLY	0001 (octal)
	O_RDWR	0002 (octal)
	O_NDELAY	0004 (octal)
	O_APPEND	0010 (octal)
	O_SYNC	0020 (octal)
fcntl	F_DUPFD	0
	F_GETFD	1
	F_SETFD	2
	F_GETFL	3
	F_SETFL	4
	F_GETLK	5
	F_SETLK	6
	F_SETLKW	7
	F_CHKFL	8
fcntl locking types	F_RDLCK	1
	F_WRLCK	2
	F_UNLCK	3
lockf	F_ULOCK	0
	F_LOCK	1
	F_TLOCK	2
	F_TEST	3

7.7.2 Constants used by C-ISAM and D-ISAM

The following constants are required for some C-ISAM functions:

isaudit	AUDSETNAME	0
	AUDGETNAME	1
	AUDSTART	2
	AUDSTOP	3
	AUDINFO	4
isopen	ISINPUT	0
	ISOUTPUT	1
	ISINOUT	2
	ISTRANS	4
	ISNOLOG	8
	ISAUTOLOCK	0x200
	ISMANULOCK	0x400
	ISEXCLLOCK	0x800
isbuild	ISINPUT	0
	ISOUTPUT	1
	ISINOUT	2
	ISTRANS	4
	ISNOLOG	8
	ISAUTOLOCK	0x200
	ISMANULOCK	0x400
	ISEXCLLOCK	0x800

isread	ISFIRST	0
	ISLAST	1
	ISNEXT	2
	ISPREV	3
	ISCURR	4
	ISEQUAL	5
	ISGREAT	6
	ISGTEQ	7
	ISLOCK	0x100
	ISWAIT	0x400

Key data types	CHARTYPE	0
	INTTYPE	1
	LONGTYPE	2
	DOUBLETYPE	3
	FLOATTYPE	4
	CHARSIZE	1
	INTSIZE	2
	LONGSIZE	4
	FLOATSIZE	sizeof(float)
	DOUBLESIZE	sizeof(double)

7.7.3 Unix and C-ISAM System Errors

The following Unix and C-ISAM system errors are likely to be encountered when using SVC-61.

7.7.3.1 Short List of Unix System Errors

The following Unix system errors are likely to be encountered when using SVC-61. Please refer to the Unix function INTRO(S) for a complete list and further details).

EPERM	1
ENOENT	2
EINTR	4
EIO	5
ENXIO	6
EBADF	9
EAGAIN	11
EACCES	13
EFAULT	14
EBUSY	16
EEXIST	17
EXDEV	18
ENOTDIR	20
EINVAL	22
EMFILE	24
ETXTBSY	26
EFBIG	27
ENOSPC	28
ESPIPE	29
EROFS	30
EMLINK	31
EPIPE	32
ERANGE	34
NOLOCK	46

7.7.3.2 Short List of C-ISAM Errors

The following C-ISAM errors are returned by the UCI in the DSERR field.

EDUPL	100
ENOTOPEN	101
EBADARG	102
EBADKEY	103
ETOOMANY	104
EBADFILE	105
ENOTEXCL	106
ELOCKED	107
EKEXISTS	108
EPRIMKEY	109
EENDFILE	110
ENOREC	111
ENOCURR	112
EFLOCKED	113
EFNAME	114
ENOLOK	115
EBADMEM	116
EBADCOLL	117
ELOGREAD	118
EBADLOG	119
ELOGOPEN	120
ELOGWRIT	121
ENOTRANS	122
ENOSHMEM	123
ENOBEGIN	124
ENONFS	125
EBADROWID	126
ENOPRIM	127
ENOLOG	128
EUSER	129
ENODBS	130
ENOFREE	131
EROWSIZE	132
EAUDIT	133
ENOLOCKS	134

7.7.3.3 Internal UCI Errors Returned in DSERR

The following errors, generated internally by the UCI, are returned in the DSERR field:

97	Invalid UCI operation code
98	Invalid partial key
99	UCI not available

7.7.3.4 Internal UCI Errors Returned in DSUCI

The following errors, generated internally by the UCI, are returned in the DSUCI field. These errors are returned together with one of the Unix or C-ISAM errors (in DSERR - see section 7.7.3) and provide more information as to the cause of the error.

7.7.3.4.1 UCI Errors from Function 2000

The following internal UCI errors may be returned from function 2000 (see section 7.9.1):

Code	Meaning
1	No more free channels available.
2	Invalid key description structure.

3	Insufficient free memory for translation buffer.
4	Invalid file name or insufficient memory for file name.
5	Invalid conversion table structure.

7.7.3.4.2 UCI Errors from Function 2001
The following internal UCI errors may be returned from function 2001 (see section 7.9.2):

Code	Meaning
6	Channel specified is not open.
7	Cannot open C-ISAM file.
8	Cannot read dictionary information from C-ISAM file.
9	Cannot read key information from C-ISAM file.
10	Cannot obtain enough memory for a record buffer.
11	The index that the file is being opened on has duplicate entries, but there would be insufficient available channels left in the pool, for the UCI to function, if this one were removed.
12	Cannot open RS file.

7.7.3.4.3 UCI Errors from Function 2002
The following internal UCI errors may be returned from function 2002 (see section 7.9.3):

Code	Meaning
13	Invalid channel number.
14	Error from file close operation.

7.7.3.4.4 UCI Errors from Function 2003
The following internal UCI errors may be returned from function 2003 (see section 7.9.4):

Code	Meaning
15	Invalid channel number.

7.7.3.4.5 UCI Errors from Function 2004
The following internal UCI errors may be returned from function 2004 (see section 7.9.5):

Code	Meaning
16	No current record number in UCI.
17	Current record not in database.
18	Current record number beyond End-Of-File.

7.7.3.4.6 UCI Errors from Function 2005

The following internal UCI errors may be returned from function 2005 (see section 7.9.6):

Code	Meaning
19	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
20	A read next/previous/current operation has been attempted but no current position available.
21	A read next operation has been attempted but the UCI reports an End-Of-File condition.
22	The UCI has attempted to lock the file in order to reposition in an index, but the file is already locked.
23	The UCI has attempted to reposition in an index but an End-Of-File condition has been detected.
24	A read previous operation has been attempted but the UCI reports a Start-Of-File condition.
25	A read current operation has been attempted but the UCI reports an End-Of-File condition.
26	An error from the isread operation has been detected.
27	An error from the record conversion routine (SVC-69) has been reported.
28	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
29	A read next/previous/current operation has been attempted but no current position is available.
30	A read next operation has been attempted but the UCI reports an End-Of-File condition.
31	The UCI has attempted to lock the file in order to reposition in an index, but the file is already locked.
32	The UCI has attempted to reposition in an index but an End-Of-File condition has been reported.
33	A read previous operation has been attempted but the UCI reports a Start-Of-File condition.
34	A read current operation has been attempted but the UCI reports an End-Of-File condition.
35	An error from the record conversion routine (SVC-69) has been reported when attempting to convert the key.
36	An error from the isread operation has been detected.

37	An error from the record conversion routine (SVC-69) has been reported when attempting to convert the data.

7.7.3.4.7 UCI Errors from Function 2006
The following internal UCI errors may be returned from function 2006 (see section 7.9.7):

Code	Meaning
38	No current record in the UCI.
39	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
40	The current record is not in the database.
41	Error reading the current record.
42	The current record has been repositioned.
43	An error from the record conversion routine (SVC-69) has been reported when attempting to convert the data.
44	An error from the isrewcurr operation has been detected.

7.7.3.4.8 UCI Errors from Function 2007
The following internal UCI errors may be returned from function 2007 (see section 7.9.8):

Code	Meaning
45	An error from the record conversion routine (SVC-69) has been reported when attempting to convert the data.
46	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
47	Error from first D-ISAM iswrcurr (documented for completeness only).
48	Error from first D-ISAM isdelrec (documented for completeness only).
49	Error from second D-ISAM isdelrec (documented for completeness only).
50	Error from second D-ISAM iswrcurr (documented for completeness only).
51	Error from D-ISAM iswrcurr (documented for completeness only).
52	Error from C-ISAM iswrcurr operation.

7.7.3.4.9 UCI Errors from Function 2008
The following internal UCI errors may be returned from function 2008 (see section 7.9.9):

Code	Meaning
53	Incorrect key definition.
54	The file channel had been returned to the pool, and could not be reopened.
55	The new index has duplicate entries, and the old one did not. Therefore, the channel must be removed from the pool, but there would be insufficient available channels left in the pool, for the UCI to function, if this one were removed.
56	Error from C-ISAM isstart.

7.7.3.4.10 UCI Errors from Function 2009

The following internal UCI errors may be returned from function 2009 (see section 7.9.10):

Code	Meaning
57	Invalid channel number
58	Channel already in use.
59	Channel already open.
60	Channel not for this node or user.
61	Invalid primary key definition.
62	Error from isbuild, or no buffer space available for the C-ISAM file.
63	This index has duplicate entries. Therefore, the channel must be removed from the pool, but there would be insufficient available channels left in the pool, for the UCI to function, if this one were removed.
64	Error from open for RS file.

7.7.3.4.11 UCI Errors from Function 2010

The following internal UCI errors may be returned from function 2010 (see section 7.9.11):

Code	Meaning
66	Error from the C-ISAM isstart function.
67	Error from the C-ISAM isindexinfo function.

7.7.3.4.12 UCI Errors from Function 2011

The following internal UCI errors may be returned from function 2011 (see section 7.9.12):

Code	Meaning
68	Illegal channel number.

7.7.3.4.13 UCI Errors from Function 2012

The following internal UCI errors may be returned from function 2012 (see section 7.9.13):

Code	Meaning
69	Illegal channel number.
70	Cannot change file name buffer.
71	Cannot change conversion table.
72	Cannot change key description.

7.7.3.4.14 UCI Errors from Function 2013

The following internal UCI errors may be returned from function 2013 (see section 7.9.14):

Code	Meaning
73	Cannot reopen RS file.
74	Cannot find file position.
75	Cannot read RS file.
76	Cannot read required data length.

7.7.3.4.15 UCI Errors from Function 2014

The following internal UCI errors may be returned from function 2014 (see section 7.9.15):

Code	Meaning
77	Cannot reopen RS file.
78	Cannot find file position.
79	Cannot write to RS file.
80	Cannot write required data length.

7.7.3.4.16 UCI Errors from Function 2015

The following internal UCI errors may be returned from function 2015 (see section 7.9.16):

Code	Meaning
82	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
83	Error from the C-ISAM isdelrec function.

7.7.3.4.17 UCI Errors from Function 2016

The following internal UCI errors may be returned from function 2016 (see section 7.9.17):

Code Meaning

85 The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.

86 Error from the C-ISAM isread operation.

87 An error from the record conversion routine (SVC-69) has been reported.

88 Error from the C-ISAM isrewcurr operation.

7.7.3.4.18 UCI Errors from Function 2017

The following internal UCI errors may be returned from function 2017 (see section 7.9.18):

Code	Meaning
89	The file channel had been returned to the pool, and could not be reopened.
90	Error from the C-ISAM isstart/isread operation.
91	An error from the record conversion routine (SVC-69) has been reported.
92	Error from the C-ISAM isrewcurr operation.
93	Error from the C-ISAM isstart or isread operation.
94	An error from the record conversion routine (SVC-69) has been reported.
95	Error from the C-ISAM isrewcurr operation.
96	Invalid index.
97	Invalid index.
99	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
100	Unable to lock the file.
101	An error from the record conversion routine (SVC-69) has been reported when converting the key.
102	Error from the C-ISAM read record operation.
103	Error from the C-ISAM iswrcurr operation.

7.7.3.4.19 UCI Errors from Function 2018

The following internal UCI errors may be returned from function 2018 (see section 7.9.19):

Code	Meaning
104	Cannot exclusively open the file.
105	Error from the C-ISAM isaddindex operation.

7.7.3.4.20 UCI Errors from Function 2019

The following internal UCI errors may be returned from function 2019 (see section 7.9.20):

Code	Meaning
106	Cannot exclusively open the file.
107	Error from the C-ISAM isdelindex operation.

7.7.3.4.21 UCI Errors from Function 2020

The following internal UCI errors may be returned from function 2020 (see section 7.9.21):

Code	Meaning
109	The file channel had been returned to the pool, and could not be reopened, or repositioned at the same place as before.
110	An error from the record conversion routine (SVC-69) has been reported when converting the data.
111	The partial key definition has more parts than the actual key.
112	The partial key segment length is greater than the length of that key segment.
113	The UCI cannot obtain enough space to make partial key.
114	A read next/previous/current operation has been attempted but no current position is available.
115	A read next operation has been attempted but the UCI reports an End-Of-File condition.
116	Trying to lock the file so as to reposition in an index, but file is already locked.
117	A reposition in an index has been attempted but the UCI reports an End-Of-File condition.
118	A read previous operation has been attempted but the UCI reports a Start-Of-File condition.
119	A read current operation has been attempted but the UCI reports an End-Of-File condition.
120	Error from the C-ISAM isread operation.
121	Error from the C-ISAM isread/isstart operation.
122	The compare of the partial index failed.

7.7.3.4.22 UCI Errors from Function 2021

The following internal UCI errors may be returned from function 2021 (see section 7.9.22):

Code	Meaning
123	Invalid channel number.
124	Error from the C-ISAM close operation.
125	Error from the C-ISAM iserase operation.

7.7.3.4.23 UCI Errors from Function 2022

The following internal UCI errors may be returned from function 2022 (see section 7.9.23):

Code	Meaning
127	File with lock is not open.
128	Error from the C-ISAM isrelease operation.

7.8 The Universal Channel Interface (UCI)

The SVC-61 "base functions" (i.e. those with a function code in the range 1 to 1799) merely invoke a specific Unix or C-ISAM function. The Universal Channel Interface (UCI) supplements the basic SVC-61 interface by providing a set of functions to access both C-ISAM databases and "flat" Unix files.

The differences between the UCI functions and the standard C-ISAM calls are that the limit of (logically) open files is defined by the number of UCI channels assigned within the Global configuration file, not the limit imposed by C-ISAM. Furthermore, and more importantly, the translation between the Global format record structure, and the C-ISAM format record structure, is automatically performed inline by the UCI using a record conversion table created using RCBUILD (see Chapter 5). Partial records can be read/modified/written without requiring information about the rest of the record or allocating space for the whole record. You are STRONGLY recommended to use UCI functions, rather than basic C-ISAM functions, in order to access raw C-ISAM databases from within Global System Manager. Note also that the Native Indexed Access Method (NIAM) is available to perform higher-level functions on a C-ISAM database (see Chapter 10 of the Global Development File Management Manual).

7.8.1 UCI Programming Notes

Before a call is made to the UCI both DSLNID and DSUSER must be established:

```
MOVE $$LNID TO DSLNID    * Global System Manager System ID
MOVE $$USER TO DSUSER    * Global System Manager user number
```

7.8.1.1 UCI Locking Considerations

The UCI does NOT perform any explicit C-ISAM file level locking. The UCI does assert file and record locks, but only to maintain its internal positioning. If a UCI function returns with a lock error then the function should be retried as the condition is probably temporary (assuming no non-UCI process has a lock outstanding on the file).

Because of a feature of C-ISAM, a lock asserted from a UCI C-ISAM function from another partition, or another user, on the same process, cannot lock against other users on that process.

All application level locking must be performed through the standard Global locking mechanisms (e.g. using the Global Cobol LOCK verb). If both a Global application, using the UCI, and a non-Global, Unix process require write access to a particular Unix file then the functions available in SVC-61 could be used to implement an external locking mechanism (which the non-Global Unix process must be aware of).

7.8.1.2 UCI Error Considerations

The UCI returns an exception when DSERR is not zero. In this case DSRETN is usually -1.

If an error occurs whilst converting a record to, or from, Global format then an exception (or STOP CODE) will be returned and DSERR will be zero. If the error occurs on a read then the current position in the file is given by DSRECN. If the error occurs on a write then the record was not written.

If a returned error is not described explicitly for a particular function call, the file position cannot be guaranteed. The error returned **may** indicate the current file position. Error codes returned by the UCI may also be returned by the Unix system calls, and these error descriptions should be referred to.

7.8.1.3 UCI Conversion Mode Parameter

The conversion mode parameter is an optional fourth parameter and is for use together with the RECORD AREA1 and RECORD AREA2 conditionals in the conversion table (see section 5.2.2). It allows different record areas from the same record to be converted using the same conversion table. If the conversion mode is not supplied then a mode of 0 is assumed.

The allowed conversion modes are:

Mode	Description
0	Return/rewrite all fields as specified in the conversion table.
1	Return only those fields that are wholly within the first 512 bytes of the Global format data record.
2	Return/rewrite only those fields within RECORD AREA2. No other fields are converted. All conditionals are evaluated.
3	Return/rewrite only those fields within RECORD AREA1. No other fields are converted. All conditionals are evaluated.
4	Return/rewrite no fields at all. All fields are converted as specified in the conversion table with any output thrown away.

7.8.1.4 UCI Rounding Errors

The UCI may automatically perform rounding for some data conversions **WITHOUT** giving any indication of the loss of accuracy. For example, attempting to convert a Global PIC 9(4) COMP field containing the value 0x7E7E to a C-ISAM Decimal D(4,0) field will appear to work. However, when the value is converted back to a Global field, the result will be 0x7E90, instead of the expected 0x7E7E. Only Global computational fields containing values within the picture clause type of the field (i.e. not the binary capacity of the field) will be converted with guaranteed accuracy.

7.8.1.5 Special Data Conversions

Two values are treated specially by the UCI when converting date fields. A date value of 0 in a Global PIC 9(6) COMP field is converted to a (strictly invalid) C-ISAM date of 0, and vice versa. A date value of 8,000,000 in a Global PIC 9(6) COMP field is converted to a (strictly invalid) C-ISAM date of 0x2000000, and vice versa.

7.9 UCI Functions

This section describes every UCI function (see section 7.4) in complete detail.

7.9.1 Initialise channel (function 2000)

This function allocates a channel for the C-ISAM database and translation table. An initialised channel is required before any other UCI function can be performed (with the exception of clear channel).

If DSPARN(2) is -1 then the channel is allocated to a basic Unix file. The UCI will then treat this file as a relative sequential (RS) file.

The DMAM translation table is described in the Global Cobol Data Management Manual. Any file using DMAM translation must follow the key field rules as described in the Global Cobol Data Management Manual and Chapter 5 of this manual. The DMAM translation table should not generally be required.

7.9.1.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2000
DSPARP(1)	Pointer to the Unix pathname for the C-ISAM database. The Unix pathname is an ASCII string which is terminated by a byte of binary zero.
DSPARP(2)	Pointer to the conversion table or, if the DSPARN(2) redefinition of this field contains -1, the file is to be treated as an RS file (the full Unix file name must be given).
DSPARP(3)	Pointer to the DMAM translation table or, if the DSPARN(3) redefinition of this field contains -1, no translation table is required.
DSLNID	$$LNID
DSUSER	$$USER

7.9.1.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	Channel number, or -1 if an error occurred.

7.9.1.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	There are no more free channels available.
EBADARG	One of the pointers is invalid.

7.9.2 Open channel using existing C-ISAM file (function 2001)

This function opens the C-ISAM files for the specified channel. The C-ISAM database must already exist. A channel must be opened before any operations can be performed on the C-ISAM database.

If the file is an RS file, it is opened with the record length supplied. The open mode is taken from the Unix open call. The following modes are allowed:

O_RDONLY
O_WRONLY
O_RDWR
O_NDELAY
O_APPEND
O_SYNC (if supported)
O_TRUNC

7.9.2.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2001
DSPARN(1)	Channel number.
DSPARN(2)	Record length (for RS files only).
DSPARN(4)	Open mode (for RS files only).
DSLNID	$$LNID
DSUSER	$$USER

7.9.2.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Current record (for RS files only).

7.9.2.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised, the channel is already open, the channel does not belong to this user or the file does not exist.
EMFILE	If the primary index allows duplicates then this error usually indicates that there are no free pooled file

	descriptors left. Each file that is accessed via an index that allows duplicates, removes one FD from the FD pool. The minimum number of FD pool entries is 2.
?	An error has occurred opening the C-ISAM database. Refer to C-ISAM/Unix error codes.

7.9.3 Close channel (function 2002)

This function closes the C-ISAM database for the specified channel and marks the channel as closed.

7.9.3.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2002
DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

7.9.3.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.3.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT The channel was not initialised, open or does not belong to this user.

?	A system error has occurred closing the C-ISAM database. The file may not have been closed properly but the channel has been closed.

7.9.4 Clear channel (function 2003)

This function clears the specified channel, or all the channels allocated by this user. If the channel was open it is closed first. The channel is now free to be reallocated.

7.9.4.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2003
DSPARN(1)	Channel number. If DSPARN(1) contains -1 then all channels for this user are affected.
DSLNID	$$LNID
DSUSER	$$USER

7.9.4.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.4.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel number is out of the range of valid channel numbers.
?	A system error has occurred closing one or more of the C-ISAM databases. The file may not have been closed properly but the channel has still been cleared.

7.9.5 Delete current record (function 2004)
This function deletes the current record from the database. The current record is specified by the record number returned from the most recent operation to successfully set the current position.

7.9.5.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2004
DSLNID	$$LNID
DSUSER	$$USER

7.9.5.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.5.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOCURR	There is no current position or the current record has already been deleted.
?	A system error has occurred. See your C-ISAM or Unix reference manual.

7.9.6 Read record (function 2005)
This function reads the specified record from the C-ISAM database using the currently active index and the specified read mode. The portion of the record returned depends upon the conversion mode. This record will become the current record.

7.9.6.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2005
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record buffer.
DSPARN(3)	Read mode (as for isread).
DSPARN(6)	Conversion mode (see section 7.8.1.3).
DSRECN	Record number.
DSLNID	$$LNID
DSUSER	$$USER

7.9.6.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSPARP(2)	Points to the record buffer as before. The buffer contains the record just read, unless a read error occurred, in which case no record is returned and the contents of the buffer remain unchanged. If an error occurred whilst converting the record into Global format, the contents of the buffer could have been corrupted.
DSRECN	Contains the record number of the new record.

7.9.6.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ELOCKED	The record was temporarily locked. The UCI only uses transient locks in which case the operation can be retried. But record could be locked by another user.
EFLOCKED	The file was temporarily locked. The UCI only uses transient locks and so the operation should be retried.
ENOREC	The record could not be found.
EENDFILE	The current position is at the beginning or end of the index.
ENOCURR	The current position is not known or the current record does not exist.
?	A system error, or C-ISAM error has occurred, you should refer to the Unix or C-ISAM error codes. The current position has been lost.

7.9.7 Update current record (function 2006)

This function updates the current record with the record data specified. If the key part of the record for the current index is changed then the record is repositioned in the index.

7.9.7.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2006
DSPARP(2)	Pointer to record.
DSPARN(6)	Conversion mode (see section 7.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

7.9.7.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Record number of updated record.

7.9.7.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOCURR	The current record does not exist or the current position is not known.
ELOCKED	The record was temporarily locked. The UCI only uses transient locks and so the operation should be retried.
EFLOCKED	The file was temporarily locked. The UCI only uses transient locks and so the operation should be retried.
?	A system error has occurred. See your C-ISAM or Unix reference manual. Assume that the current position is not known.

7.9.8 Write a new record (function 2007)

This function writes the specified record to the C-ISAM database as a new record. This record then becomes the current record. If the write fails the current position remains unchanged.

7.9.8.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2007
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record to write.

DSPARN(6)	Conversion mode (see section 7.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

7.9.8.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Record number of updated record.

7.9.8.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EFLOCKED	The file was temporarily locked. The UCI only uses transient locks and so the operation should be retried.
?	If the error code is less than 100 a system error has occurred. Otherwise a C-ISAM error has occurred and you should refer to the C-ISAM error codes.

7.9.9 Change the current key (function 2008)

This function changes the currently active index to that specified by the supplied key description. The specified key must correspond to an existing index. If KPPART = 0 then indexing is via record number (natural ordering). The current record becomes logically prior to the first record in the file.

Important note: If "natural ordering" is selected KPFLAG will be set to ISNODUPS.

7.9.9.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2008
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description.
DSLNID	$$LNID
DSUSER	$$USER

7.9.9.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.9.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EBADKEY	Part of the new key is invalid. The current index should remain active but it is suggested that the key is changed back explicitly.
EMFILE	An attempt was made to change the current index from one with no duplicates to one with duplicates and there were insufficient FD's in the FD pool. The minimum number of FD's in the FD pool is 2.
?	An error occurred changing to the new index. It should be assumed that the current index and position are now unknown and attempts to access the file should not be made until these are reset.

7.9.10 Open channel creating ISAM file (function 2009)

This function creates and then opens the C-ISAM database for this channel. A channel must be opened before any operations can be performed on the C-ISAM database. The C-ISAM database is created with the specified record length and with the specified key as the primary index. The C-ISAM database must not previously exist.

If an RS file is required, then the file is created and opened with the record length supplied. The open mode is taken from the Unix open call. The following modes are allowed:

O_RDONLY
O_WRONLY
O_RDWR
O_NDELAY
O_APPEND
O_SYNC (if supported)
O_TRUNC

The file permissions are the same as those provided to the Unix open system call. Write and execute permissions for "other" will not be granted.

If an error is returned the file is not created and the channel is not open.

7.9.10.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2009
DSPARN(1)	Channel number.
DSPARN(2)	Record length.
DSPARP(3)	Pointer to key description for the primary index.
DSPARN(4)	Open mode (for RS file only).

DSPARN(5)	File permission (for RS files only)
DSLNID	$$LNID
DSUSER	$$USER

7.9.10.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.10.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT The channel is not initialised, does not belong to this user or is already open.

EBADKEY	Part of the key description is invalid. The file has not been created and the channel is not open.
?	An error occurred whilst creating the C-ISAM database. If the error number is less than 100 then a system error occurred, otherwise a C-ISAM error occurred and you should refer to the C-ISAM error codes.

7.9.11 Read index or file information (function 2010)
If the index number is 0, the database dictionary information is returned and should be accessed as a DI block. A value greater than 0 will return the key information for that index, and should be accessed as a KP block. A value of 1 refers to the primary index for the file.

7.9.11.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2010
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to buffer for returned data.
DSPARN(3)	Index number. A value of 0 indicates that the files dictionary information is requested. A value of 1 indicates the primary index.
DSLNID	$$LNID
DSUSER	$$USER

7.9.11.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSPARP(2)	Pointer to the dictionary information, or the key description for a particular index.

7.9.11.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
?	An error occurred while reading the requested information from the file. The requested information is not returned.

7.9.12 Read channel debug information (function 2011)
This function is used for debugging the UCI and should never be used by external developers. It returns channel status information from channel control block. This function does not check the User Number or System ID, thus any channel can be accessed. The information returned by this function is implementation specific.

THIS OPERATION SHOULD NEVER BE USED AND IS DOCUMENTED FOR COMPLETENESS ONLY.

7.9.12.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2011
DSPARN(1)	Channel number.
DSPARP(2)	Debug operation code:

0	return channel flags.
1	return file name.
2	return conversion table.
3	return translation table.
4	return record buffer.
5	return key description table.
6	return internal buffer.

DSPARN(3)	Pointer to buffer for returned debug information.
DSPARN(4)	Length of returned data.

7.9.12.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.12.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	Invalid channel number.

7.9.13 Write channel debug information (function 2012)
This function is used for debugging the UCI and should never be used by external developers. It writes channel status information into a channel control block. This function does not check the User Number or

System ID, thus any channel can be accessed. The information returned by this function is implementation specific.

THIS OPERATION SHOULD NEVER BE USED AND IS DOCUMENTED FOR COMPLETENESS ONLY.

7.9.13.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2012
DSPARN(1)	Channel number.
DSPARP(2)	Debug operation code: 0 set channel flags. 1 set file name. 2 set conversion table. 3 set translation table. 4 set record buffer. 5 set key description table. 6 set internal buffer.
DSPARN(3)	Pointer to buffer for sent debug info.
DSPARN(4)	Length of sent data.

7.9.13.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.13.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	Invalid channel number.

7.9.14 Read records from RS file (function 2013)

This function reads the number of records specified by DSPARN(3) from record position DSRECN in the file into the buffer pointed to by DSPARP(2).

7.9.14.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2013
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record buffer.
DSPARN(3)	Number of records to read.
DSRECN	Starting record number.
DSLNID	$$LNID

DSUSER	$$USER

7.9.14.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Current record position after read.

7.9.14.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened or does not belong to this user.
0	DSPARN(3) records were successfully read before the end of file was reached.
?	An error occurred during the read. No records were returned. Refer to your Unix error codes.

7.9.14.4 Operating Notes
Partial records may be read into DSPARP(2) but will not be reported, as the returned value for the number of records read is rounded down to a whole number. This will occur if the file is not an integral number of records long. The last few bytes in the file will be read but not reported. The end of file record position given by DSRECN will point to these last few bytes.

Important note: Named pipes cannot be accessed because the FD pool handling within the UCI will result in inconsistent results.

7.9.15 Write records to RS file (function 2014)
This function writes the number of records specified by DSPARN(3) to record position DSRECN in the file from the buffer pointed to by DSPARP(2).

7.9.15.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2014
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record buffer.
DSPARN(3)	Number of records to written.
DSRECN	Starting record number.
DSLNID	$$LNID
DSUSER	$$USER

7.9.15.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Current record position after the write.

7.9.15.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened or does not belong to this user.
0	DSPARN(3) records were successfully written.
?	An error occurred during the write. No records were written. Refer to your Unix error codes.

7.9.15.4 Operating Notes
Partial records may be written but will not be reported, as the returned value for the number of records written is rounded down to a whole number.

If O_APPEND is used to open the file then the records are always written to the physical end of file even if the file is not an integral number of records long.

Important note: Named pipes cannot be accessed because the FD pool handling within the UCI will result in inconsistent results.

7.9.16 Delete record via record number (function 2015)
This function deletes the record with the specified record number from the C-ISAM database.

7.9.16.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2015
DSPARN(1)	Channel number.
DSRECN	Record number to delete.
DSLNID	$$LNID
DSUSER	$$USER

7.9.16.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.16.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
EFLOCKED	The file was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
?	An error occurred deleting the record. If the error number is less than 100, a system error occurred. Otherwise a C-ISAM error occurred and you should refer to the C-ISAM error codes.

7.9.17 Update record via record number (function 2016)

This function updates the record specified by the record number. It is possible to change the key values for this record.

7.9.17.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2016
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to record.
DSPARN(6)	Conversion mode (see section 7.8.1.3).
DSRECN	Record number.
DSLNID	$$LNID
DSUSER	$$USER

7.9.17.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.17.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
EFLOCKED	The file was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
?	An error occurred updating the record. If the error number is less than 100, a system error occurred. Otherwise a C-ISAM error occurred and you should refer to the C-ISAM error codes.

Also, if the current index allows duplicates, the following error codes may be returned:

-2	The current position is now past the end of the file.
-3	The current position is unknown.
-4	Both the current position and index are unknown. There has been a serious error. The current index must be re-established before continuing. All other read/write operations are undefined.
-5	The current record has become the next record as the current record had been deleted.
DSRECN	Record number of the new current record.

7.9.18 Write or update by key (function 2017)

This function updates the record specified by its key contents. If the record does not exist a new record is written. This record becomes the current record. If an error occurs the previous position is maintained. The current index must not allow duplicates.

7.9.18.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2017
DSPARP(2)	Pointer to record.
DSPARN(6)	Conversion mode (see section 7.8.1.3).
DSLNID	$$LNID
DSUSER	$$USER

7.9.18.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSRECN	Updated or written record.

7.9.18.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EINVAL	The current index specifies natural ordering or allows duplicates.
ELOCKED	The record was temporarily locked. The UCI uses only transient locks and so the operation should be retried.
EFLOCKED	The database was temporarily locked. The UCI uses only transient locks and so the operation should be retried.

? A system error occurred. See your C-ISAM or Unix reference manual.

7.9.19 Add index (function 2018)

This function adds the index specified by the key description to the C-ISAM database.

7.9.19.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2018
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description.
DSLNID	$$LNID
DSUSER	$$USER

7.9.19.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.19.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EFLOCKED	The database is locked and this operation cannot gain the exclusive access it needs to update the index list.
EDUPL	This index is not a duplicate index but records have been found which have duplicate key values for this index.
EBADKEY	The key description being supplied for the new index is invalid.
EKEXISTS	The index specified already exists.

? A system error occurred. See your C-ISAM or Unix reference manual.

7.9.20 Delete index (function 2019)

This function deletes the index specified by the key description from the C-ISAM database.

7.9.20.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2019

DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description.
DSLNID	$$LNID
DSUSER	$$USER

7.9.20.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.20.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
EFLOCKED	The database is locked and this operation cannot gain the exclusive access it needs to update the index list.
EBADKEY	The key description being supplied for the new index is invalid.
?	A system error occurred. See your C-ISAM or Unix reference manual.

7.9.21 Position record pointer (function 2020)
This function sets the current record position according to the specified search criteria without the record being read. This function enables you to position on a partial key by specifying the last key segment and the partial length of this segment.

A value of -1 in DSPARN(3) signifies that the whole key is to be used.

7.9.21.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2020
DSPARN(1)	Channel number.
DSPARP(2)	Pointer to key description.
DSPARN(3)	Position mode (modes as for isread).
DSPARN(4)	Last key segment.
DSPARN(5)	Length of last key segment (0-complete).
DSPARN(6)	Conversion mode (see section 7.8.1.3).
DSLNID	$$LNID

DSUSER	$$USER

7.9.21.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.
DSPARP(2)	Point to the record buffer as before. The buffer contains the key value for the current record. The contents of this buffer remain unchanged unless an error occurs whilst converting the record into C-ISAM format when the contents of the buffer may be unpredictable.
DSRECN	Record number.

7.9.21.3 Error Codes
The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOCURR	There is no current position in the file.
ENOREC	Record not found.
98	The partial key specification passed to the UCI was invalid. Either the partial key segment was beyond the number of key segments, or the offset into the key segment was beyond the end of that segment, or the end of the partial key was part way through an indivisible key segment (e.g. floating point), or there was insufficient space to create the partial key.

? A system error occurred. See your C-ISAM or Unix reference manual.

7.9.22 Close delete C-ISAM file channel (function 2021)
This function closes the file channel specified and deletes the C-ISAM database associated with that channel.

7.9.22.1 Entry Parameters
Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2021
DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

7.9.22.2 Exit Parameters
On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.22.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
?	A system error occurred. See your C-ISAM or Unix reference manual.

7.9.23 Unlock records on C-ISAM channel (function 2022)

This function unlocks any outstanding locks on the file channel specified.

7.9.23.1 Entry Parameters

Before calling the UCI, the following parameters must be established in the DS control block:

DSFUNC	2022
DSPARN(1)	Channel number.
DSLNID	$$LNID
DSUSER	$$USER

7.9.23.2 Exit Parameters

On return from the UCI, the following results are returned in the DS control block:

DSRETN	0, or -1 if an error occurred.

7.9.23.3 Error Codes

The following error codes are returned, in DSERR, if DSRETN contains -1:

ENOENT	The channel has not been initialised and opened, or does not belong to this user.
ENOTOPE	The channel is not open.
?	A system error occurred. See your C-ISAM or Unix reference manual.

8. Interfacing to the Windows NT Operating System

This chapter describes SVC-61. This interface allow Windows NT files to be accessed directly from within Global System Manager and is only available on Global System Manager (Windows NT) configurations.

The basic SVC-61 interface is used by both the Windows NT SVC-61 File Converter (see Chapter 3) and the Windows NT File Access Method (see Chapter ?? of the Global Development File Management Manual).

Important note-1: The Global System Manager (Windows NT) SVC-61 interface currently only supports MS-DOS compatible functions and MS-DOS FAT filing systems. All the arguments required by these MS-DOS compatible functions are described in Chapter 6. The publication recommended in Chapter 6 is relevant for the MS-DOS compatible functions described in this chapter.

Important note-2: Not all the MS-DOS functions available in the Global System Manager (MS-DOS and Windows) SVC-61 interface are supported in the Global System Manager (Windows NT) SVC-61 interface. In particular, the Btrieve UCI functions, available with Global System Manager (MS-DOS and Windows) are NOT supported on Global System Manager (Windows NT).

Important note-3: The SVC-61 interface for Global System Manager (Windows NT) supports both 16-bit Cobol/Speedbase programs and 32-bit Speedbase programs. Developers of 32-bit applications should be aware of the differences in the SVC-61 interface.

8.1 Using SVC-61 to Invoke an MS-DOS Compatible Win NT Function

An MS-DOS compatible Windows NT function is called from Global System Manager using a Global Cobol statement of the form:

```
     SVC 61 USING ds
```

where ds is a request block.

8.1.1 SVC-61 DS Control Block

The format of the ds request block for the SVC-61 functions listed in section 8.2 depends on the application environment.

8.1.1.1 SVC-61 DS Control Block for 16-bit applications

The ds request block a 16-bit application is defined below:

```
     01   DS
       02 DSFUNC    PIC X                         * Function code
     *
     * As explained below the top-bit of the function code must be
     * clear to indicate a 16-bit SVC-61 operation.
     *
       02 DSMODE    PIC X                         *   Subfunction   or
mode
       02 DSRES          PIC 9(4) COMP            *  Windows  NT  return
code
       02 DSHAND    PIC 9(4) COMP            * File handle (not used)
       02 DSNAME    PIC PTR                  * Pointer to file name
```

```
          02 DSBUFF    PIC PTR                  * Pointer to buffer
          02 DSATTR    PIC X(2)                 * File attributes
          02 DSNBYT    PIC 9(4) COMP            * Number of bytes moved
          02 DSPAR1    PIC X(2)                 * Function specific
          02 DSPAR2    PIC X(2)                 * Function specific
          02 DSPAR3    PIC X(2)                 * Function specific
          02 DSPAR4    PIC X(2)                 * Function specific
          02 DSHA32    PIC 9(9) COMP            * Win-32 file handle
          02 DSHAFI    PIC 9(9) COMP            * Win-32 find handle
          02 DSRES32   PIC 9(9) COMP            * Win-32 error code

        01   FILLER REDEFINES DSPAR1
          02 DSP1H     PIC X                         * Function specific
          02 DSP1L          PIC X                         *      Function
specific

        01   FILLER REDEFINES DSPAR2
          02 DSP2H     PIC X                         * Function specific
          02 DSP2L          PIC X                         *      Function
specific

        77   DSP32B REDEFINES DSPAR1 PIC 9(9) COMP   * Expect WARNING
```

All fields are in Global Cobol format unless specified below.

The file handle, DSHA32, is returned by Windows NT when the file is opened and **MUST NOT BE CHANGED IN ANY WAY.** If more than one Windows NT file is to be opened at one time then a separate DS block should be allocated for each open Windows NT file. Note that the DOS-compatible DSHAND field is not used.

8.1.1.2 SVC-61 DS Control Block for 32-bit applications

The ds request block a 32-bit application is defined below:

```
        01   DS
          02 DSFUNC    PIC X                         * Function code
*
* As explained below the top-bit of the function code must be
* set to indicate a 32-bit SVC-61 operation.
*
          02 DSMODE    PIC X                         *   Subfunction   or
mode
          02 DSRES          PIC 9(4) COMP            *  Windows  NT  return
code
          02 DSHAND    PIC 9(4) COMP            * File handle (not used)
          02 DSNAME    PIC SPT                  * Reserved for future use
          02 DSBUFF    PIC SPT                  * Reserved for future use
          02 DSATTR    PIC X(2)                 * File attributes
          02 DSNBYT    PIC 9(4) COMP            * Number of bytes moved
          02 DSPAR1    PIC X(2)                 * Function specific
          02 DSPAR2    PIC X(2)                 * Function specific
          02 DSPAR3    PIC X(2)                 * Function specific
          02 DSPAR4    PIC X(2)                 * Function specific
          02 DSHA32    PIC 9(9) COMP            * Win-32 file handle
          02 DSHAFI    PIC 9(9) COMP            * Win-32 find handle
          02 DSRES32   PIC 9(9) COMP            * Win-32 error code
          02 DS32NAME  PIC PTR                  * 32-bit ptr to file name
          02 DS32BUFF  PIC PTR                  * 32-bit ptr to buffer
          02 FILLER         PIC X(3)                 *    Reserved    for
future use
```

```
       02 DS32ERR   PIC X                          * 32-bit error code

     01    FILLER REDEFINES DSPAR1
       02 DSP1H     PIC X                          * Function specific
       02 DSP1L          PIC X                          *      Function
specific

     01   FILLER REDEFINES DSPAR2
       02 DSP2H     PIC X                          * Function specific
       02 DSP2L          PIC X                          *      Function
specific

     77   DSP32B REDEFINES DSPAR1 PIC 9(9) COMP   * Expect WARNING
```

All fields are in Global Cobol format unless specified below.

The file handle, DSHA32, is returned by Windows NT when the file is opened and **MUST NOT BE CHANGED IN ANY WAY.** If more than one Windows NT file is to be opened at one time then a separate DS block should be allocated for each open Windows NT file. Note that the DOS-compatible DSHAND field is not used.

8.2 SVC-61 Function Numbers for Windows NT Functions

This section describes the Windows NT functions that are available using the SVC-61 interface.

THIS SECTION SHOULD BE READ IN CONJUNCTION WITH AN APPROPRIATE PROGRAMMER'S GUIDE.

Disclaimer: All the Windows NT functions described below are passed directly to the Windows NT operating system. In general, SVC-61 does not validate the DS request block before invoking the Windows NT function. Software developers using this interface should be aware that misuse can cause serious problems.

In this section, all numbers are in hexadecimal notation. The numbers in brackets are the 32-bit equivalents of the 16-bit function numbers.

DSFUNC	Description
00	Get version number of SVC-61
0B - 0F	Reserved for Speedbase Gateway operations (3000 - 3999)
0F - 13	Reserved for external RPC operations (4000 - 4999)
30 (B0)	Get Windows NT version number
36 (B6)	Get free disk space (FAT filing system only)
39 (B9)	Create directory
3A (BA)	Delete directory
3B (BB)	Set default directory
3C (BC)	Create or open file
3D (BD)	Open old file
3E (BE)	Close file
3F (BF)	Read sequential
40 (C0)	Write sequential
41 (C1)	Delete file
42 (C2)	Position file pointer
47 (C7)	Get default directory
4E (CE)	Find first file
4F (CF)	Find next file

50	(D0)	Find close
56	(D6)	Rename file
5B	(DB)	Create new file

8.2.1 Get version number of SVC-61 (function 00H)

This function simply returns the version of SVC-61. It does not invoke any Windows NT functions.

8.2.1.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	00H

8.2.1.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 always (i.e. this function is supported)
DSPAR1	Portion of version number before the decimal point.
DSPAR2	Portion of version number after the decimal point.

8.2.1.3 Comments

This function returns the version number of SVC-61 (i.e. m.n) as two separate character fields. The number before the decimal point is returned in DSPAR1 as a character field. The number after the decimal point is returned in DSPAR2 as a character field. For example, the values returned by the 4.4 version of SVC-61 will be:

```
DSPAR1 " 4"
DSPAR2 "4 "
```

Note that the SVC-61 version number is completely independent of the BACNAT variant.

8.2.1.4 32-bit Programming Notes

The same function code (i.e. 00H) is used for both the 16-bit and the 32-bit interface.

The first version of SVC-61 that supports the 32-bit function codes is V4.5. All 32-bit programs that use SVC-61 should obtain the SVC-61 version number and abort with an error if the version is less than "4.5".

8.2.2 Get Windows NT version number (function 30H or B0H)

This function returns the Windows NT version number.

8.2.2.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	30H for 16-bit applications B0H for 32-bit applications
DSMODE	0 = Get OEM number 1 = Get Windows NT version flag

8.2.2.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DSP1H	Major version number (3,4 etc.)
DSP1L	Minor version number (3.51 = 51)

8.2.2.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.2.4 32-bit Programming Notes
Either function 30H or B0H can be used by 32-bit applications. However, for consistency with other 32-bit functions, we recommend the use of function code B0H.

8.2.3 Get free disk space (function 36H or B6H)
This function returns the amount of space available on a designated drive along with other selected information about the drive.

8.2.3.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	36H for 16-bit applications B6H for 32-bit applications
DSMODE	Drive number (0 = current, 1 = A, 2 = B to 26 = Z)

8.2.3.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DSPAR1	Sectors per cluster
DSPAR2	Number of available sectors
DSPAR3	Bytes per cluster
DSPAR4	Clusters on the drive

8.2.3.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

Important note: This function is only supported on DOS compatible FAT filing systems. If an attempt is made to use this function on an NTFS filing system the results will be unpredictable.

8.2.3.4 32-bit Programming Notes
Either function 36H or B6H can be used by 32-bit applications. However, for consistency with other 32-bit functions, we recommend the use of function code B6H.

8.2.4 Create directory (function 39H or B9H)
This function creates a sub-directory at the specified drive and path location.

8.2.4.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	39H for 16-bit applications B9H for 32-bit applications
DSNAME	Pointer to ASCIIZ path specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ path specification (for 32-bit applications)

8.2.4.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)

8.2.4.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.4.4 32-bit Programming Notes
Function B9H MUST be used by 32-bit applications.

8.2.5 Delete directory (function 3AH or BAH)
This function removes a sub-directory if it is empty.

8.2.5.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3AH for 16-bit applications BAH for 32-bit applications
DSNAME	Pointer to ASCIIZ path specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ path specification (for 32-bit applications)

8.2.5.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)

8.2.5.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.5.4 32-bit Programming Notes
Function BAH MUST be used by 32-bit applications.

8.2.6 Set default directory (function 3BH or BBH)
This function sets the current or default directory to match the designated string.

8.2.6.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3BH for 16-bit applications BBH for 32-bit applications
DSNAME	Pointer to ASCIIZ path specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ path specification (for 32-bit applications)

8.2.6.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)

8.2.6.3 Comments
Function 3BH (BBH) modifies the current directory and will affect the operation of the DDF file and DOSPRINT printer controllers if the pathname in the Windows Registry is not an absolute pathname (i.e. if the pathname is a relative pathname). If the pathname in the Registry is a relative pathname, both the DDF and DOSPRINT controllers expect Windows NT to remain in the "Global directory". This potential problem is easily solved by specifying full pathnames, including the drive letter, for the relevant Registry entries for these controllers.

Note that function 3BH (BBH) will not affect the operation of the DOSPRINT controller if the Windows NT printer device defined in the Registry is a physical printer (e.g. LPT1:) rather than a filename or directory name. Note also that function 3BH (BBH) will not affect the operation of the WINPRINT controller. Please refer to the Global Operating Manual (Windows NT) for further details.

Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.6.4 32-bit Programming Notes

Function BBH MUST be used by 32-bit applications.

8.2.7 Create or open file (function 3CH or BCH)

This function creates the designated file if it does not exist, or truncates it to zero length if it does exist. If the open succeeds, this function returns a file handle (a 32-bit number) to reference the opened file.

8.2.7.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3CH for 16-bit applications BCH for 32-bit applications
DSNAME	Pointer to ASCIIZ path specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ path specification (for 32-bit applications)
DSATTR	File attribute: #00 Normal file #02 Hidden file #04 System file #06 Hidden and system file

8.2.7.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)
DSHA32	Returned Win-32 file handle

8.2.7.3 Comments

Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.7.4 32-bit Programming Notes

Function BCH MUST be used by 32-bit applications.

8.2.8 Open old file (function 3DH or BDH)

This function opens the designated file and returns a file handle (a 32-bit number) to reference the opened file.

8.2.8.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3DH for 16-bit applications BDH for 32-bit applications
DSNAME	Pointer to ASCIIZ path specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ path specification (for 32-bit applications)
DSMODE	Access and file-sharing mode

8.2.8.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)
DSHA32	Returned Win-32 file handle

8.2.8.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.8.4 32-bit Programming Notes
Function BDH MUST be used by 32-bit applications.

8.2.9 Close file (function 3EH or BEH)
This function closes a file previously open with file handles.

8.2.9.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3EH for 16-bit applications BEH for 32-bit applications
DSHA32	Win-32 file handle (from previous open or create)

8.2.9.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)

8.2.9.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.9.4 32-bit Programming Notes

Either function 3EH or BEH can be used by 32-bit applications. However, for consistency with other 32-bit functions, we recommend the use of function code BEH.

8.2.10 Read sequential (function 3FH or BFH)

This function reads data from the file or device specified by the file handle argument. This data is written to a designated memory location.

8.2.10.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	3FH for 16-bit applications BFH for 32-bit applications
DSHA32	Win-32 file handle (from previous open or create)
DSNBY	Number of bytes to transfer
DSBUFF	Pointer to buffer area (for 16-bit applications)
DS32BUFF	Pointer to buffer area (for 32-bit applications)

8.2.10.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)
DSNBYT	Number of bytes read

8.2.10.3 Comments

Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.10.4 32-bit Programming Notes

Function BFH MUST be used by 32-bit applications.

8.2.11 Write sequential (function 40H or C0H)

This function writes data to the file or device specified by the file handle argument.

8.2.11.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	40H for 16-bit applications C0H for 32-bit applications
DSHA32	Win-32 file handle (from previous open or create)
DSNBYT	Number of bytes to transfer
DSBUFF	Pointer to buffer area (for 16-bit applications)

DS32BUFF	Pointer to buffer area (for 32-bit applications)

8.2.11.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)
DSNBYT	Number of bytes written

8.2.11.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.11.4 32-bit Programming Notes
Function C0H MUST be used by 32-bit applications.

8.2.12 Delete file (function 41H or C1H)
This function deletes the specified file from the Windows NT system.

8.2.12.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	41H for 16-bit applications C1H for 32-bit applications
DSNAME	Pointer to ASCIIZ path specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ path specification (for 32-bit applications)

8.2.12.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)

8.2.12.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.12.4 32-bit Programming Notes
Function C1H MUST be used by 32-bit applications.

8.2.13 Position file pointer (function 42H or C2H)

This function changes the current location in the file, the file pointer, to a position relative to the start of file, end of file, or current position.

8.2.13.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	42H for 16-bit applications C2H for 32-bit applications
DSHA32	Win-32 file handle (from previous open or create)
DSMODE	Method code: #00 Offset from beginning of file #01 Offset from current position #02 Offset from end of file
DSP32B	Offset address
DSPAR1	Most significant part of offset in DSP32B
DSPAR2	Least significant part of offset in DSP32B

8.2.13.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DSPAR1	Most significant part of offset (updated)
DSPAR2	Least significant part of offset (updated)

8.2.13.3 Comments

For function 42H (C2H), the two function specific parameters, DSPAR1 and DSPAR2, can be treated as a single quantity in PIC 9(9) COMP format, DSP32B.

Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.13.4 32-bit Programming Notes

Either function 42H or C2H can be used by 32-bit applications. However, for consistency with other 32-bit functions, we recommend the use of function code C2H.

8.2.14 Get default directory (function 47H or C7H)

This function returns an ASCIIZ string with the full path of the current directory, not including the drive and leading backslash character (\).

8.2.14.1 Calling Parameters

Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	47H for 16-bit applications C7H for 32-bit applications
DSMODE	Drive number (0 = current, 1 = A, 2 = B to 26 = Z)
DSBUFF	Pointer to a 65-byte scratch buffer (for 16-bit applications)
DS32BUFF	Pointer to a 65-byte scratch buffer (for 32-bit applications)

8.2.14.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)

8.2.14.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.14.4 32-bit Programming Notes
Function C7H MUST be used by 32-bit applications.

8.2.15 Find first file (function 4EH or CEH)
This function locates the first occurrence of a matching file name, given an ASCII string, which can include wild-cards.

8.2.15.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	4EH for 16-bit applications CEH for 32-bit applications
DSATTR	Attribute to use in search: #00 Normal #02 Normal and hidden #04 Normal and system #06 Normal, hidden and system #10 Directories
DSNAME	Pointer to ASCIIZ file specification (for 16-bit applications)
DS32NAME	Pointer to ASCIIZ file specification (for 32-bit applications)
DSBUFF	Pointer to buffer for DTA (for 16-bit applications)
DS32BUFF	Pointer to buffer for DTA (for 32-bit applications)

8.2.15.2 Return Parameters

On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)
DSHAFI	Win-32 find handle
DSBUFF	The DOS-compatible DTA is returned to the area addressed by this pointer (for 16-bit applications)
DS32BUFF	The DOS-compatible DTA is returned to the area addressed by this pointer (for 32-bit applications)

8.2.15.3 Comments
For functions 4EH and 4FH, DSBUFF must point to a 43 byte area which will be used as the DOS-compatible "DTA".

Refer to the relevant Programmer's guide for further information regarding this Windows NT function and the structure of the DOS-compatible "DTA".

8.2.15.4 32-bit Programming Notes
Function CEH MUST be used by 32-bit applications.

For functions CEH and CFH, DS32BUFF must point to a 43 byte area which will be used as the DOS-compatible "DTA".

8.2.16 Find next file (function 4FH or CFH)
After a successful call to function 4EH (CEH), see section 8.2.15, this function continues to find files that match the specified criteria.

8.2.16.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	4FH for 16-bit applications CFH for 32-bit applications
DSATTR	Attribute to use in search: #00 Normal #02 Normal and hidden #04 Normal and system #06 Normal, hidden and system #10 Directories
DSHAFI	Win-32 find handle (from a previous successful Find First - see section 8.2.15)
DSBUFF	The DOS-compatible DTA is returned to the area addressed by this pointer (for 16-bit applications)
DS32BUFF	The DOS-compatible DTA is returned to the area addressed by this pointer (for 32-bit applications)

8.2.16.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES 0 if no error occurred, or DOS compatible error code

DSRES32 Win-32 error code (if DSRES = 99)

DS32ERR 32-bit interface error (if DSRES = 100)

DSBUFF DTA returned to the area addressed by this pointer (for 16-bit applications)

DS32BUFF DTA returned to the area addressed by this pointer (for 32-bit applications)

8.2.16.3 Comments
For functions 4EH and 4FH, DSBUFF must point to a 43 byte area which will be used as the DOS-compatible "DTA".

Refer to the relevant Programmer's guide for further information regarding this Windows NT function and the structure of the DOS-compatible "DTA".

8.2.16.4 32-bit Programming Notes
Function CFH MUST be used by 32-bit applications.

For functions CEH and CFH, DS32BUFF must point to a 43 byte area which will be used as the DOS-compatible "DTA".

8.2.17 Find close (function 50H or D0H)
After a successful call to function 4EH (CEH), see section 8.2.15, this function MUST be used to close the Find Handler opened by the Find First function (see section 8.2.15).

8.2.17.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC 50H for 16-bit applications
D0H for 32-bit applications

DSHAFI Win-32 find handle (from a previous successful Find First - see section 8.2.15)

8.2.17.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES 0 if no error occurred, or DOS compatible error code

DSRES32 Win-32 error code (if DSRES = 99)

8.2.17.3 Comments
This function MUST be called after a "Find First File" (see section 8.2.15), "Find Next File" (see section 8.2.16) sequence.

Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.17.4 32-bit Programming Notes
Either function 50H or D0H can be used by 32-bit applications. However, for consistency with other 32-bit functions, we recommend the use of function code D0H.

8.2.18 Rename file (function 56H or D6H)
This function renames a file or moves it to another directory on the same disk.

8.2.18.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	56H for 16-bit applications D6H for 32-bit applications
DSNAME	Pointer to ASCIIZ current file name (for 6-bit applications)
DS32NAME	Pointer to ASCIIZ current file name (for 32-bit applications)
DSBUFF	Pointer to ASCIIZ new file name (for 16-bit applications)
DS32BUFF	Pointer to ASCIIZ new file name (for 32-bit applications)

8.2.18.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES	0 if no error occurred, or DOS compatible error code
DSRES32	Win-32 error code (if DSRES = 99)
DS32ERR	32-bit interface error (if DSRES = 100)

8.2.18.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.18.4 32-bit Programming Notes
Function D6H MUST be used by 32-bit applications.

8.2.19 Create new file (function 5BH or DBH)
This function creates a new file in the specified directory.

8.2.19.1 Calling Parameters
Before calling SVC-61, the following parameters must be established in the DS control block:

DSFUNC	5BH for 16-bit applications DBH for 32-bit applications

DSATTR   File attribute:

#00	Normal file
#02	Hidden file
#04	System file
#06	Hidden and system file

DSNAME   Pointer to ASCIIZ file specification (for 16-bit applications)

DS32NAME Pointer to ASCIIZ file specification (for 32-bit applications)

8.2.19.2 Return Parameters
On entry from SVC-61, the following results are returned in the DS control block:

DSRES   0 if no error occurred, or DOS compatible error code

DSRES32   Win-32 error code (if DSRES = 99)

DS32ERR   32-bit interface error (if DSRES = 100)

DSHA32   Returned Win-32 file handle

8.2.19.3 Comments
Refer to the relevant Programmer's guide for further information regarding this Windows NT function.

8.2.19.4 32-bit Programming Notes
Function DBH MUST be used by 32-bit applications.

8.3 SVC-61 Programming Notes
The following points should be considered when using SVC-61.

8.3.1 SVC-61 Interface Conventions
All file and directory names passed to SVC-61 must be ASCII strings terminated by a byte containing binary-zero. For example, the file "C:\DATA\MYFILE" can be established using the following Global Cobol statements:

```
     77   NAME      PIC X(?)
          VALUE     "C:\DATA\MYFILE"
          VALUE      #00
```

8.3.2 Error Handling and Exceptions
If DSFUNC is set to an unrecognised value or if any Windows NT or Btrieve functions return an error, SVC-61 will generate an exception. SVC-61 will generate an exception 1 for all error conditions.

All error conditions are returned in the Win-32 result field, DSRES32. Win-32 error codes between 1 and 98, that have equivalent DOS error codes are returned in DSRES. Win-32 specific error codes, with values higher than 99, are treated specially. If the Win-32 error condition has a DOS equivalent, the equivalent DOS error code is returned in DSRES. If the Win-32 error condition does not have a DOS equivalent, the special value 99 is returned in DSRES.

An invalid 32-bit address passed in either DS32NAME or DS32BUFF can cause a 32-bit addressing error. These errors are indicated by a result of 100 in DSRES and an alphabetic error code in DS32ERR. To summarise:

DSRES	DSRES32	DS32ERR	Description
1 - 98	Win-32 error	n/a	DOS compatible Win-32 error code
99	Win-32 error	n/a	Win-32 error with no DOS equivalent
100	n/a	32-bit address error code	Invalid 32-bit pointer passed in either DS32NAME or DS32BUFF

A description of the 32-bit address error codes is beyond the scope of this manual.

If DSFUNC is set to an unrecognised value, an exception 1 will be returned and DSRES will contain 1.

8.4 User Constants for SVC-61 Functions

This section lists some useful constants that may be required when using SVC-61.

8.4.1 Short List of DOS compatible NT Error Codes

The following DOS compatible Error Codes may be returned by SVC-61 (in DSRES):

1	Invalid function
2	File not found
3	Path not found
4	No handles available
5	Access denied
6	Invalid handle
7	Memory control blocks destroyed
8	Insufficient memory
9	Invalid memory block address
10	Invalid environment
11	Invalid format
12	Invalid access code
13	Invalid data
14	Reserved
15	Invalid drive
16	Attempt to remove current directory
17	Not the same device
18	No more files
19	Disk write-protected
20	Unknown unit
21	Drive not ready
22	Unknown command
23	CFC error
24	Bad request structure length
25	Seek error
26	Unknown media type
27	Sector not found

```
28   Out of paper
29   Write fault
30   Read error
31   General failure
32   Sharing violation
33   Lock violation
34   Invalid disk change
35   (not used)
36   Sharing buffer overflow
37   (not used)
38   Error handling EOF
39   Handle disk full
40   Win-32 specific error code (in DSRES32)
80   File already exists
```

Appendix A - FCONV Error Messages

The Physical Sector File Converter, FCONV, can detect a number of general error conditions as listed below. These error messages all start with "$56". Any other error messages are specific to the particular converter and are documented separately in Chapter 2. In some cases following an error a partially complete Global file may have been created.

The messages are described below in alphabetical order:

$56 BLOCK SIZE (nnnnn BYTES) TOO LARGE

A copy operation cannot proceed because FCONV is unable to acquire enough main storage for the single block buffer required for the conversion process. This condition should not arise on practical configurations. If it does you must run the utility on a configuration with a larger user area.

$56 CONVERSION TYPE NOT SUPPORTED

The file converter implementation does not support the conversion type you have requested, even though the type is valid as far as FCONV itself is concerned. You should check the appropriate file converter description for the exact specification of the conversion types supported.

$56 EMPTY INPUT FILE

End of file has been signalled when the file converter attempted to read the very first record of the specified input file.

$56 EMPTY PROGRAM FILE

The program file you are attempting to transfer from the host operating system contains no memory image records.

$56 FILE NOT FOUND

The input file is not present on the specified volume.

$56 INSUFFICIENT SPACE

Either there is insufficient contiguous space on the output volume to hold the file to be transferred, or the output volume's file directory is full.

$56 INVALID FORMAT FILE

The type of the input file, or the information it contains, is inconsistent with the conversion type you have requested. The message will also be displayed if the file is missing essential information (e.g. a program file which does not contain an entry point). When the error occurs on a host operating system file, this message will normally be preceded by an additional self-explanatory error message produced by the file converter.

$56 INVALID NATIVE FILE NAME

The name you have specified has been rejected by the file converter because it does not conform to host operating system naming conventions.

$56 INVALID FORMAT VOLUME ON unit

You have mistakenly mounted the non-Global volume on the unit reserved for the Global volume, or the Global volume where the non-Global one is expected.

$56 PROGRAM FILE LOAD SEQUENCE ERROR

A program file being copied from the non-Global volume contains a memory image block which loads at a lower address than that of its very first memory image block. It is essential that the first block loads at the lowest address of all.

$56 RECORD LENGTH GREATER THAN BLOCK LENGTH
$56 RECORD LENGTH NOT EQUAL TO BLOCK LENGTH

These errors should only appear when you are testing a new Native Interface Program. They indicate that an invalid record length has been returned to FCONV by a file converter READ function. The first occurs when transferring a program or text file from the host operating system if the record length ever exceeds the block length established by the OPEN function. The second is similar: The block length used in transferring a host operating system data file to Global System Manager must always remain the same.

$56 TYPE MUST BE D OR T

You have specified a conversion type other than D or T when attempting to transfer a file from Global System Manager to the host operating system. Only **D**ata or **T**ext files can be transferred in this direction.

$56 TYPE MUST BE D, T, A OR P

You have specified a conversion type other than D,T,A or P when attempting to transfer a file from the host operating system to Global System Manager. Only **D**ata, **T**ext, **A**bsolute program files, or **P**osition-independent program files can be transferred.

Appendix B - RCBUILD Error and Warning Messages

This appendix describes the error and warning messages that can occur when using RCBUILD to produce a record conversion table for the conversion of Global format ISAM or DMAM files using the Universal Channel Interface.

***** ERROR 1 -FILE- EXPECTED**

The "FILE" statement is missing or wrongly positioned on the first line.

***** ERROR 2 GLOBAL INPUT filename REQUIRED**

The name of the Global input file is missing or wrongly positioned on the first line.

***** ERROR 3 -GLOBAL- EXPECTED**

The "GLOBAL" statement is missing on the first line following the Global file name.

***** ERROR 4 -TO- EXPECTED**

The "TO" key word is missing from the first line following the "GLOBAL " statement.

***** ERROR 5 -GLOBAL- OR -UNIX/BTRIEVE- EXPECTED**

The "GLOBAL" or "UNIX" statement is missing from the first line.

***** ERROR 6 -RECORD- OR -INPUT- EXPECTED**

The "RECORD" or "INPUT" statement is missing or not in the correct order on the second line following the "GLOBAL" statement.

***** ERROR 7 -RECORD- EXPECTED**

The "RECORD" statement is missing from either the second or third lines.

***** ERROR 8 -LENGTH- OR input record length EXPECTED**

The "LENGTH" statement or the actual Global record length is missing or not in the correct position on the second line.

***** ERROR 9 -KEY- EXPECTED**

The "KEY" statement is missing from the second or third line following the record length.

***** ERROR 10 -LENGTH- OR input key length EXPECTED**

The "LENGTH" statement or the actual Global key length is missing from the second line.

***** ERROR 11 input key length EXPECTED**

The actual Global key length is missing from the second line following the "LENGTH" statement.

***** ERROR 12 -OFFSET- EXPECTED**

The "OFFSET" key word is missing or in the wrong position in the second or third line.

***** ERROR 13 THE NUMBER 4 EXPECTED**

The key offset number of "4" is required for the Global record statement on the second line.

***** ERROR 14 -UNIX/BTRIEVE- EXPECTED**

The "UNIX" or "BTRIEVE" statement is required as the first key word on the third line.

***** ERROR 15 -LENGTH- OR record length EXPECTED**

The "LENGTH" statement or the actual Unix/Btrieve record length is required following the "RECORD" statement on the third line of the source file.

***** ERROR 16 record length EXPECTED**

The Unix/Btrieve record length is required on the third line following the "RECORD" statement.

***** ERROR 17 -LENGTH- OR key length EXPECTED**

The "LENGTH" statement or the actual key length value is expected after the "KEY" statement on the third line.

***** ERROR 18 Key length EXPECTED**

The value of the Unix/Btrieve key length is expected following the "KEY" or "KEY LENGTH" statements on the third line.

***** ERROR 19 offset EXPECTED**

The offset value of the key within the Unix/Btrieve record is required on the third line following the "OFFSET" statement.

***** ERROR 20 TYPE qual EXPECTED**

There must be a qualifier following the conversion type in a field conversion line.

***** ERROR 21 -=- EXPECTED**

The "=" key must be present in a field conversion line to separate the two sides of the conversion.

***** ERROR 22 VALUE TOO HIGH**

The value given as a qualifier to a field type exceeds the limit for that field type.

***** ERROR 23 TOTAL DIGITS PRECISION TOO HIGH**

The total precision given as a qualifier to an ISAM decimal field exceeds the maximum allowed for that type.

***** ERROR 24 -)- EXPECTED**

The ")" indicating the end of a qualifier is missing.

***** ERROR 25 VALUE GREATER THAN 32**

The precision or the decimal places part of the qualifier of an ISAM decimal field exceeds 32.

***** ERROR 26 HEX OR STRING VALUE EXPECTED**

The hex value or character string required as the value of the comparison for an "IF" statement is missing.

***** ERROR 27 -END CONVERSION- EXPECTED**

The end of the source file has been reached before an "END CONVERSION" statement has been encountered.

***** ERROR 28 NUMERIC VALUE EXPECTED**

A numeric value (usually as part of a type qualifier) is missing from a field conversion line.

***** ERROR 29 OFFSET VALUE NOT PRESENT**

The offset value of a field within the appropriate record is missing from a field conversion line.

***** ERROR 30 VALUE IN QUOTES MUST BE MAX OF 1 CHAR**

The value in quotes for a "FIXED" field qualifier must only be one character long.

***** ERROR 31 ILLEGAL CONVERSION TYPE**

The conversion type supplied for this conversion line is inappropriate.

***** ERROR 32 AREAn EXPECTED**

An AREA statement followed by the area number is expected and was not supplied correctly.

***** ERROR 33 TRANS or DESC NOT ALLOWED ON UNIX SIDE**

The TRANS or DESC key words are only appropriate on the Global side of a conversion line and must not appear on the Unix or Btrieve side of a conversion.

***** ERROR 34 TRANSLATION TYPE MUST BE CHARACTER**

The format of a DMAM translation field must always be of a character type.

***** ERROR 35 TRANS/DESC MUST BE FOLLOWED BY REAL**

A descending field for a Unix conversion must be followed by a second conversion line which converts the original value of the field. Alternatively, a translation field for a Unix or Btrieve conversion must be followed by a second conversion line which converts the original value of the field.

***** ERROR 36 'OCCURS' EXPECTED**

The OCCURS key word is expected.

***** ERROR 37 INVALID IN AN OCCURRING GROUP**

A conditional line has been included as part of an occurring group. This is not allowed.

***** ERROR 38 INVALID QUALIFIER**

The qualifier supplied for the conversion item contains an invalid qualifier.

***** ERROR 39 RECORD LENGTH EXCEEDED**

The field is partially or wholly outside the length defined for the record.

***** WARNING 101 4 EXPECTED AS OFFSET VALUE**

An key offset of 4 is expected for the Global record description on the second line.

***** WARNING 102 MORE THAN 179 CONVERSIONS IN SOURCE FILE**

There are more than 179 field conversion lines in the source file. The maximum of 179 field lines has been exceeded and only the first 179 will be processed.

***** WARNING 103 EXTRA CHARACTER(S) AT END OF LINE**

Spurious characters are present at the end of a line.

Appendix C - Example RCBUILD Conversion Table

This appendix contains an example RCBUILD source conversion table:

```
*
* Source file S.PRDT
*
* Copyright 1991 TIS Software Ltd.
*
* Global ISAM to Unix conversion table for product file
*
* The offsets for the Global record are given in hex as this
* can be achieved simply by compiling the record using the Global
* Cobol compiler.
*
FILE PRD GLOBAL TO UNIX
GLOBAL RECORD LENGTH 320 KEY LENGTH 15 OFFSET 4
UNIX RECORD LENGTH 461 KEY LENGTH 15 OFFSET 2
*
* The TYPE and Link Fields. The Link field has no meaning in an
* ISAM file on Unix
*
PRRTYP    PIC X(2) #0000 = PIC X(2)      0    * Record type VC
PRLINK    FIXED #FF 2 #0002 = NOTHING   2
*
* Key field
*
PRPRCD    PIC X(15) #0004 = PIC X(15)   2    * Product code
*
* Data Fields
*
PRWRCD    PIC 9(2) #0013 = FLOAT        17   * No. of warehouse
                                             * records
*
* Warehouse dependent information
*
PRWHLC    PIC X(6) #0014 = PIC X(6)      21   * Location within
                                             * warehouse
PRSBOK    PIC 9(8,3) #001A = DOUBLE     27   * Book stock
                                             * Held in small units
PRSALC    PIC 9(8,3) #001F = DOUBLE     35   * Allocated stock
                                             * Held in small units
PRSBCK    PIC 9(8,3) #0024 = DOUBLE     43   * Stock on back order
                                             * Held in small units
PRSFWD    PIC 9(8,3) #0029 = DOUBLE     51   * Stock on forward order
                                             * Held in small units
PRSORD    PIC 9(8,3) #002E = DOUBLE     59   * Stock on order from
                                             * supplier held in
                                             * stocking units
PRSDIF    PIC 9(8,3) #0033 = DOUBLE     67   * Stock taking diff.
                                             * Held in small units
PRSUPP    PIC X(8) #0038 = PIC X(8)      75   * Supplier reference
PRLEAD    PIC 9(2) #0040 = SHORT         83   * Lead time (weeks)
PRREOD    PIC 9(8,3) #0041 = DOUBLE     85   * Reorder level
                                             * Held in stocking units
PRLMIN    PIC 9(8,3) #0046 = DOUBLE     93   * Minimum stock level
                                             * Held in stocking units
PRLMAX    PIC 9(8,3) #004B = DOUBLE     101  * Maximum stock level
                                             * Held in stocking units
```

```
PRDLSL    PIC DATE #0050 = UNIXDATE     109  * Date of last sale
                                             * (invoice)
PRDLST    PIC DATE #0053 = UNIXDATE     113  * Date of last stock take
PRDLSO    PIC DATE #0056 = UNIXDATE     117  * Date of last
                                             * supplier order
PRREOQ    PIC 9(8,3) #0059 = DOUBLE     121  * Normal reorder quantity
                                             * Held in stocking units
PRLPUC    PIC 9(5,4) #005E = DOUBLE     129  * Last purchase unit
                                             * Held for stocking units
FILLER    PIC X(4) #0062 = PIC X(4)     137  * Unused field once
                                             * &&VALUE
PRSUNV    PIC 9(8,3) #0066 = DOUBLE     141  * Unvalued stock
                                             * Held in small units
FILLER    PIC X(2) #006B = PIC X(2)     149  * Reserved for expansion
PRTAIL    PIC X(3) #006D = PIC X(3)     151  * Reserved for tailoring
*
* Product determinator
*
PRTYPE    PIC 9(2) #0070 = SHORT        154  * Product type
                                             * -1 - Comment extra
                                             * 0 - Extra
                                             * 1 - Service
                                             * 2 - Unstocked
                                             * product
                                             * 3 - Stocked product
*
* General product details
*
PRGRP     PIC 9(2) #0071 = SHORT        156  * Product group 1-99
PRDESC    PIC X(30) #0072 = PIC X(30)   158  * Product description
PRMSFS    PIC 9 #0090 = SHORT           188  * Microsafe product
                                             * type
                                             * 0 Ordinary product
                                             * 1 Manufactured
                                             * product
PRSUPR    PIC 9 #0091 = SHORT           190  * Superseded product
PRPALT    PIC X(15) #0092 = PIC X(15)   192  * Alternate product
PRACT     PIC 9(2) #00A1 = SHORT        207  * Active indicator
                                             * 0 - Active
                                             * 1 - Inactive
PRPRWH    PIC 9(4,4) #00A2 = DOUBLE     209  * Product weight
                                             * (Stock unit)
PRQDIS    PIC 9 #00A6 = SHORT           217  * Quantity disc 0-9
PRTDIS    PIC 9 #00A7 = SHORT           219  * Trade discount 1-9
PRSDIS    PIC 9 #00A8 = SHORT           221  * Allow settle. disc
                                             * 0 Disallow
                                             * 1 Allow
PRVAT     PIC 9 #00A9 = SHORT           223  * VAT code 1-9
PRNOM     PIC 9(2) #00AA = SHORT        225  * Nominal code 1-99
PRABKO    PIC 9 #00AB = SHORT           227  * Accept back orders
                                             * 0 no , 1 yes
*
* Product unit definitions
*
PRUNTS    PIC 9(2) #00AC = SHORT        229  * 2 sales units?
                                             * 0 - no, 1 - yes
PRSELU    PIC 9(2) #00AD = SHORT        231  * Default selling unit
                                             * 0 - small, 1 - large
PRDELU    PIC 9(2) #00AE = SHORT        233  * Stock-cost-weight
```

```
                                                  * unit
                                                  * 0 - small, 1 - large
PRDLGE    PIC X(5) #00AF = PIC X(5)      235  * Description - large
                                                  * unit
PRDSML    PIC X(5) #00B4 = PIC X(5)      240  * Description - small
                                                  * unit
PRCONV    PIC 9(5,4) #00B9 = DOUBLE      245  * Conversion factor
*
* Pricing information
*
PRSELP1   PIC 9(5,4) #00BD = D(11,4)     253  * Selling prices
PRSELP2   PIC 9(5,4) #00C1 = D(11,4)     260
PRSELP3   PIC 9(5,4) #00C5 = D(11,4)     267
PRSELP4   PIC 9(5,4) #00C9 = D(11,4)     274
PRSELP5   PIC 9(5,4) #00CD = D(11,4)     281
PRSELP6   PIC 9(5,4) #00D1 = D(11,4)     288
PRSELP7   PIC 9(5,4) #00D5 = D(11,4)     295
PRSELP8   PIC 9(5,4) #00D9 = D(11,4)     302
*
* May only be negative for an extra.
* Must be -100.00 to +100.00 for a percentage extra
* May only be accurate to > 2 decimal places for a non-extra
*
PRPRAC1   PIC 9      #00DD = SHORT       309  * Pricing acc/%used
PRPRAC2   PIC 9      #00DE = SHORT       311
PRPRAC3   PIC 9      #00DF = SHORT       313
PRPRAC4   PIC 9      #00E0 = SHORT       315
PRPRAC5   PIC 9      #00E1 = SHORT       317
PRPRAC6   PIC 9      #00E2 = SHORT       319
PRPRAC7   PIC 9      #00E3 = SHORT       321
PRPRAC8   PIC 9      #00E4 = SHORT       323
*
* For a non-extra 4=PRSELP(N) Held to 1/100 of a penny
*     2=PRSELP(N) Held to 1 penny
*     -1=PRSELP(N) is space
* For an extra  1=PRSELP(N) is a percentage
*     0=PRSELP(N) is a fixed price extra
*     -1=PRSELP(N) is space
*
PRSCST    PIC 9(5,4) #00E5 = D(11,4)     325  * Standard unit cost
                                                  * Held for stocking units
PRPRIU    PIC 9 #00E9 = SHORT            332  * -1=4 small,4 large
                                                  *     prices
                                                  * 0 = 8 Small prices
                                                  * 1 = 8 Large prices
*
* Product chaining fields
*
PRFORD    PIC 9(9) #00EA = LONG          334  * First DL record for
                                                  * product
PRLORD    PIC 9(9) #00EE = LONG          338  * Last DL Record for
                                                  * product
*
* Statistics
*     - Part A this period
*     A1 Sales this period
*
PRPSTC    PIC 9(7,2) #00F2 = D(11,2)     342  * Total cost
PRPNET    PIC 9(7,2) #00F6 = D(11,2)     349  * Net sales value
PRPDIS    PIC 9(7,2) #00FA = D(11,2)     356  * Sales discount
```

```
PRPUNT    PIC 9(8,3) #00FE = DOUBLE     363  * Units sold
                                             * Held in default selling
                                             * unit
*
*     A2 Receipts this period
*
PRPACS    PIC 9(7,2) #0103 = D(11,2)    371  * Actual cost
PRPAUN    PIC 9(8,3) #0107 = DOUBLE     378  * Units acquired
                                             * Held in stocking unit
*
*     - Part B This year
*     B1 Sales this year
*
PRYSTC    PIC 9(8,2) #010C = D(12,2)    386  * Total cost
PRYNET    PIC 9(8,2) #0111 = D(12,2)    393  * Net sales value
PRYDIS    PIC 9(8,2) #0116 = D(12,2)    400  * Sales discount
PRYDIS    PIC 9(8,2) #0116 = D(12,2)    407  * Sales discount
PRYUNT    PIC 9(9,3) #011B = DOUBLE     414  * Units sold
                                             * Held in default
                                             * Selling unit
*
*     B2 Receipts this year
*
PRYACS    PIC 9(8,2) #0121 = D(12,2)    422  * Actual cost
PRYAUN    PIC 9(9,3) #0126 = DOUBLE     429  * Units acquired
                                             * Held in stocking unit
*
* Microsafes information
*
PRSAFE    PIC 9     #012C = SHORT       437  * 1=Microsafes needs to
                                             * know about this product
                                             * 0=Microsafes already
                                             * Knows everything about
                                             * this product or does
                                             * not wish know
PRAVCS    PIC 9(9,2) #012D = D(12,2)    439  * Available costing
                                             * Held in small unit
*
PRMANP    PIC 9(6) #0132 = LONG         446  * 0= No interest to PX
                                             * += Record no of
                                             * structure
                                             * -= To be defined
FILLER    PIC X #0135 = PIC X           450  * Reserved for expansion
PRTAI2    PIC X(10) #0136 = PIC X(10)   451  * Reserved for tailoring
END CONVERSION
```

Appendix D - Universal Channel Interface (UCI) Stop Codes

This appendix describes the STOP codes that may occur within the record conversion routine (SVC-69) within the Unix Universal Channel Interface (UCI).

All STOP codes are of the form:

```
STOP 69xy
```

and indicate that an invalid pairing of conversion types has occurred when performing the a record conversion on a C-ISAM file. The digit x is the input conversion type and the digit y is the output conversion type. The conversion types are as follows:

Type	Description
0	nothing or fixed fields
1	Global computational fields, Unix INT and LONG
3	Global or Unix character fields
4	Global or Unix date fields
6	floating point fields
7	decimal fields
8	translation fields
9	descending translation fields