MUSIC/SP Administrator's Reference

Part IV - Chapter 17 (AR_P4.PS)

Part IV. Utilities

Chapter 17. System Utility Programs

Overview of Utilities

This chapter provides information on the the MUSIC/SP system utilities. The descriptions are arranged in alphabetical order by the utility name. The "Summary by Function" topic below can be used to determine the name of the utility program required for a given task.

These utilities often require special privileges and therefore cannot be run by the general user. The special privilege names and meaning can be found in the description of the CODUPD utility. The required privileges are described with each utility. If the files of the programs are specified as PRIVATE (as they are on the distributed system), the user executing them must have sufficient privileges to access them (i.e. LSCAN).

All of the programs run under the control of MUSIC. Most of the MUSIC utilities can be run either from batch or a terminal. It is recommended that programs producing at lot of output be run from batch.

To execute a utility program from batch, the following commands are used:

```
/FILE                   <--/FILE statements (if any)
/INCLUDE name           <--where "name" is the name of the utility
....                    <--parameters (if any)
....                    <--data for the program (if any)
```

To execute a utility program from a terminal, the terminal must be signed on to the system with a sign-on code of sufficient privileges. Note: the VIP privilege is active only after the /VIP ON command is issued at *Go time by a sign-on code that has the VIP privilege.

Most utilities can be run by simply typing the name of the utility when the terminal is in *Go mode.

Namelist Input

Many utilities use the FORTRAN Namelist facility to enter the options. Namelist, as extended by MUSIC, offers a powerful and concise way of entering optional parameters. For example if a utility has three parameters called A, B and C, then they could be specified as follows:

```
a=10.5,c=2,b='musicx'
```

Notice that they do not have to be in any specific order. Omitted parameters will retain their value. This means that unspecified parameters will take on default values. Should the utility ask for the options again at a later time, then you need only specify the ones that you want changed.

Errors detected by the Namelist facility will be displayed and then you will have a chance to respecify them.

Namelist parameters can be extended over several lines. This is done by ending the previous line with a comma (,) as in the example:

```
a=10,
b=20
```

Be very careful to enter the comma. If you omit it then the utility would not know that it must read any additional lines and so will not work as intended.

Additional Namelist examples are shown below:

```
a=z130          <--hexadecimal number 130 specified
a='musicx'      <--characters string given
a=true          <--parameter given a "true" value
a=10,4,35       <--three numbers specified for an array
```

Summary by Function

The system utility names are listed below by function.

Accounting

ACCTDS.SCAN
: Display information from accounting data set.

ACTDMP
: Process the system accounting file to produce session accounting records.

DSACT1

DSACT2
: Produce UDS file accounting records.

MFACCT
: Produce file accounting records.

NOWDOL
: Scan the terminal accounting records and update the NOW$ field in the code table records for each user.

Backup

CODUMP
: Make a backup copy of the code table on tape and optionally print entire code table.

DSARCH
: Create backup copies of UDS or SDS files on tape.

MFARCH
: Used to create an incremental backup of the save library on tape.

MFARC2
: Used to backup selected save files to tape.

Change

CDUMP
: Display and change main storage.

CODUPD
: Add, delete, change, and display, code table records.

DSKDMP
: Display and change records on disk using absolute disk address. Display the VTOC, providing the names and extents of all UDS and SDS files, and free space on a disk. Display detailed VTOC entry for an individual UDS or SDS file. Reorganize the VTOC free space pointers (format-5 DSCBs).

EDTCAT
: Create or change the system catalogue. The catalogue contains entries to define SDS files, specify which modules go in the link pack area, and issue CP commands at IPL time.

FILECH
: Change the access controls of a save file.

FIXINDEX
: Change entries in the save library index to fix index/directory errors.

FIXINDEX.AUTO
: Change entries in the save library index to fix index/directory errors.

FORMAT
: Format entire disk packs, SDS files, or individual tracks on disk. Create or change disk volume labels. Write data to disk by absolute disk address.

LDLIBE
: Add, delete, or change members in the system or user load library.

NUCGEN
: Change system nucleus and/or I/O configuration.

SUBLIB.GEN	Used to make additions or changes to the subroutine library.
SYSDMP	Display or change records on disk based on relative record number within a specific SDS file or DEB.
SYSGEN1	Write system nucleus to disk.
SYSREP	Apply minor changes to load library members. Similar to OS superzap.
SYSUPDATE	Change Load Library Directory in main storage
UCR	Add, change, or delete save library user control records. These records contain the allocation limits for save files on any given user code.

Code

CODPRV	Lists all privileged userids (codes).
CODUMP	Make a backup copy of the code table on tape and optionally print entire code table.
CODUPD	Add, delete, change, and display, code table records.
GEN.CODES	Create input for CODUPD which adds a group of access codes.
NOWDOL	Scan the terminal accounting records and update the NOW$ field in the code table records for each user.
TRANS$	Allow supervisor code to transfer funds to codes under this supervision.
UCR	Add, change, or delete save library user control records. These records contain the allocation limits for save files on any given user code.
UCRFIX	Scan the code table and adjust or create a user control record for each code record.
WHOACT	Display a list of active terminal sessions indicating the TCB number, device address and user code.
WHOALL	Display a list of all terminal sessions indicating the TCB number, device address and user code if present.
WHOSON	Display TCB information about specific user codes that are currently signed on.

Disk

BUFLOG	List usage and error information for 3330 and 2305 disks.
CHKDISK	Verify disk formatting.
DSKDMP	Display and change records on disk using absolute disk address. Display the VTOC, providing the names and extents of all UDS and SDS files, and free space on a disk. Display detailed VTOC entry for an individual UDS or SDS file. Reorganize the VTOC free space pointers (format-5 DSCBs).
FMFREE	Create a job to format all the free space on a disk pack.
FORMAT	Format entire disk packs, SDS files, or individual tracks on disk. Create or change disk volume labels. Write data to disk by absolute disk address.
INITFBA	Create a VTOC on an FBA disk.
IOTIME	Display usage and timing information for SDS files, disk volumes, and channels.
SYSDMP	Display or change records on disk based on relative record number within a specific SDS file or DEB.
UCB	Display a list of disk and tape devices on the system.

Display

ATTRIB	Display save file attributes such as space allocation, record length, access control and date last used.
BPOOL	Display information about the terminal buffer pool.
BSTATUS	Display information about current batch job.
CDUMP	Display and change main storage.
CODUPD	Add, delete, change, and display, code table records.
CONLOG	Displays the current contents of the console log.

COUNTS	Display information accumulated by the system COUNTS macro.
DSKDMP	Display and change records on disk using absolute disk address. Display the VTOC, providing the names and extents of all UDS and SDS files, and free space on a disk. Display detailed VTOC entry for an individual UDS or SDS file. Reorganize the VTOC free space pointers (format-5 DSCBs).
ENQTAB	Display information about the system enqueue table.
IOTIME	Display usage and timing information for SDS files, disk volumes, and channels.
LDCNTS	Displays usage information for the system load library.
LDLIST	Display names, sizes, and attributes of the members of the system or user load library.
LIBSPACE	Display a list of available free extents in the save library.
LOOKUP	Display the table of privileged program names.
LPA	Display information on Link Pack Area (LPA).
MAPMEM	Display memory usage.
MFINDEX	Display information about the save library index.
RATE	Display system load.
ROUTETABLE	Display contents of the $ROUTING table.
SSTAT	Display the current system status.
SYSDMP	Display or change records on disk based on relative record number within a specific SDS file or DEB.
UCB	Display a list of disk and tape devices on the system.
WAITS	Display system wait time information.
WHOACT	Display a list of active terminal sessions indicating the TCB number, device address and user code.
WHOALL	Display a list of all terminal sessions indicating the TCB number, device address and user code if present.
WHOSON	Display TCB information about specific user codes that are currently signed on.
XTELL	Delivers message whether receiver has MESSAGES ON or OFF.

Information

BPOOL	Display information about the terminal buffer pool.
BSTATUS	Display information about current batch job.
BUFLOG	List usage and error information for 3330 and 2305 disks.
CONLOG	Displays the current contents of the console log.
COUNTS	Display information accumulated by the system COUNTS macro.
ENQTAB	Display information about the system enqueue table.
IOTIME	Display usage and timing information for SDS files, disk volumes, and channels.
LDCNTS	Displays usage information for the system load library.
LOOKUP	Display the table of privileged program names.
LPA	Display information on Link Pack Area (LPA).
MAPMEM	Display memory usage.
MFINDEX	Display information about the save library index.
SSTAT	Display the current system status.
SYSDATE	Displays nucleus level and date/time of IPL.
UCB	Display a list of disk and tape devices on the system.
WAITS	Display system wait time information.
WHOSON	Display TCB information about specific user codes that are currently signed on.

Load Library

LDCNTS	Displays usage information for the system load library.
LDLIBE	Add, delete, or change members in the system or user load library.
LDLIST	Display names, sizes, and attributes of the members of the system or user load library.

SYSREP	Apply minor changes to load library members. Similar to OS superzap.
SYSUPDATE	Change Load Library Directory in main storage

Restore

CMSTAPE	Restore files from tapes produced by the CMS TAPE DUMP command.
DSRST	Restore UDS and SDS files from tapes produced by DSARCH
LOADPDS	Restore files from a tape produced by the IEBCOPY utility.
MFREST	Restore save file from tapes produces by MFARCH and MFARC2.
MOVEPDS	Restore file from tapes produced by the IEHMOVE utility.
UDSRST	Restore UDS files from tapes produced by DSARCH.

Save Library Files

ATTRIB	Display save file attributes such as space allocation, record length, access control and date last used.
ELOG.CLEANUP	Free up save file space by deleting unused editor log files.
FILE.DELETE	Delete groups of files at a time.
FILECH	Change the access controls of a file.
FPRINT	Print the content of a group of files.
MFARC2	Used to backup selected files to tape.
MFHASH	Hash a save file name to give the index block number for that file.
MFREST	Restore save file from tapes produces by MFARCH and MFARC2.
UCR	Add, change, or delete save library user control records. These records contain the allocation limits for files on any given user code.
UCRFIX	Scan the code table and adjust or create a user control record for each code record.

Save Library

ADDPDS	Create a PDS from an IEBUPDTE tape file.
CHKFILES	Scan Save Library to find any files with errors 65 and 67.
FIXINDEX	Change entries in the save library index to fix index/directory errors.
FIXINDEX.AUTO	Change entries in the save library index to fix index/directory errors.
GENSAV	Create multiple files from a single file.
LIBINDEX.CLEAN1	Clean Save Library Index overflow area.
LIBINDEX.CLEAN2	Clean Save Library Index auxiliary area.
LIBINTEG	Compare the save library space allocation maps with the index and directory listing any anomalies.
LIBSPACE	Display a list of available free extents in the save library.
MFACCT	Produce file accounting records.
MFARCH	Used to create an incremental backup of the save library on tape.
MFARC2	Used to backup selected files to tape.
MFINDEX	Display information about the save library index.
MFMOVE	Reclaim fragmented save library space by moving the files to other library SDS files.
NEWINDEX	Create a new save library index data set of a different size.
SETFBN	Reset Save Library backup numbers.
UCR	Add, change, or delete save library user control records. These records contain the allocation limits for save files on any given user code.

SDS

DSARCH	Create backup copies of UDS or SDS files on tape.
DSCOPY	Copy UDS or SDS files.
DSKDMP	Display and change records on disk using absolute disk address. Display the VTOC, providing the names and extents of all UDS and SDS files, and free space on a disk. Display detailed VTOC entry for an individual UDS or SDS file. Reorganize the VTOC free space pointers (format-5 DSCBs).
DSREN	Rename UDS files (see MUSIC/SP User's Reference Guide).
DSRST	Restore UDS and SDS files from tapes produced by DSARCH
EDTCAT	Create or change the system catalogue. The catalogue contains entries to define SDS files, specify which modules go in the link pack area, and issue CP commands at IPL time.
FORMAT	Format entire disk packs, SDS files, or individual tracks on disk. Create or change disk volume labels. Write data to disk by absolute disk address.
IOTIME	Display usage and timing information for SDS files, disk volumes, and channels.
MFMOVE	Reclaim fragmented save library space by moving the files to other library SDS files.
SYSDMP	Display or change records on disk based on relative record number within a specific SDS file or DEB.

Storage

CDUMP	Display and change main storage.
PRDUMP	Run after a MUSIC storage dump is taken to print a formatted listing of system storage, registers, control blocks, and trace table.
SYSUPDATE	Change Load Library Directory in main storage

TCB

WHOACT	Display a list of active terminal sessions indicating the TCB number, device address and user code.
WHOALL	Display a list of all terminal sessions indicating the TCB number, device address and user code if present.
WHOSON	Display TCB information about specific user codes that are currently signed on.

UDS

DSACT1	
DSACT2	Produce UDS file accounting records.
DSARCH	Create backup copies of UDS or SDS files on tape.
DSCOPY	Copy UDS or SDS files.
DSKDMP	Display and change records on disk using absolute disk address. Display the VTOC, providing the names and extents of all UDS and SDS files, and free space on a disk. Display detailed VTOC entry for an individual UDS or SDS file. Reorganize the VTOC free space pointers (format-5 DSCBs).
DSREN	Rename UDS files. See *MUSIC/SP User's Reference Guide* for a description.
DSRST	Restore UDS and SDS files from tapes produced by DSARCH
TAPUTIL	Dump or summarize selected files from tape, and copy files from one tape to another (see See *MUSIC/SP User's Reference Guide* for a description.
UDSRST	Restore UDS files from tapes produced by DSARCH.
UTIL	List, punch, copy, and merge data on cards, card images, tape, or disk. See *MUSIC/SP User's Reference Guide* for a description.

VTOC

DSKDMP	Display and change records on disk using absolute disk address. Display the VTOC, providing the names and extents of all UDS and SDS files, and free space on a disk. Display detailed VTOC entry for an individual UDS or SDS file. Reorganize the VTOC free space pointers (format-5 DSCBs).
INITFBA	Create a VTOC on an FBA disk.

Utilities Listed Alphabetically

The descriptions that follow are arranged in alphabetical order by the names of the utilities.

ACCTDS.SCAN: Display Information from Accounting Data Set

This interactive utility displays information from the detailed records in the system accounting data set, SYS1.MUSIC.ACCT. You can specify the starting and ending block for the scan, and selection criteria such as a userid or userid prefix. To start the utility, enter the command ACCTDS.SCAN.

ACTDMP: Accounting File Dump

This program reads the MUSIC Accounting File normally called SYS1.MUSIC.ACCT. Three (3) types of records are produced by this program. One is a summary record of each terminal user's terminal session (or batch job), showing processing unit utilization, session time, charge for number of characters received or sent to the terminal. Another record is produced when UDS files are deleted by users. The third type provides information that can be used to maintain systems availability records.

Records produced by ACTDMP represent accounting information that has accumulated in the accounting data set (SYS1.MUSIC.ACCT) since the last time ACTDMP was run (or since the last time the =RESET option was used during MUSIC IPL).

ACTDMP accumulates the detailed information from the MUSIC accounting files for each session. If a session is still active when ACTDMP is run, the current totals accumulated for that session are written to the accumulation table file ($ACT:ACCTAB). When ACTDMP is next run the totals are re-loaded from the file and the accounting process proceeds from where it left off. The accounting record for the session is only generated when ACTDMP detects that the session is complete.

If the accumulation table file is invalid or out of date, ACTDMP will skip to the first IPL record that is found and begin the accounting process from that point. This is a normal situation after the system has been loaded with the =RESET option.

Usage:

The following JCL is used to run this ACTDMP program from batch. See *Chapter 5 - Routine Maintenance* for the recommended time to run this program.

```
/INCLUDE ACTDMP
```

Its output is directly written to the file $ACT:ACT.RCDS. If a file by that name already exists, it will rename the old one to "$ACT:ACT.RCDSnn" where *nn* is a number 01 to 09.

The installation can run its own billing program after ACTDMP has run. It can be setup to read the file $ACT:ACT.RCDS as input.

The file $ACT:ACTDMP contains a parameter statement that can be changed if the installation wants to use a different name for the output file or change the number of backup copies or alter the primary size of the output file. The following is the parameter statement format:

columns 1-4	primary space for output file in k bytes
columns 6-7	number of backup copies (must be 01 to 99)
columns 9-28	userid:filename for output

The format of the accounting records produced is as follows:

Session Accounting Record

Column	Contents
1-3	Identification 'RRR'.
4-16	Userid (first 13 characters). The remaining 3 characters are in columns 51-53.
17-24	Identification. terminal - third parameter from /ID command. batch - columns 5-12 of /ID statement (jobname).
25-29	Processing time (60/100 service units). A *service unit* (SU) will approximate 1 million machine instructions if the recommended value chosen for the CFACT parameter at the time the NUCGEN utility was run. For example, on a 3 MIP machine, 1 SU represents about 1/3 seconds of processing time.
30-34	Reserved.
35-39	Terminal I/O charge in cents. terminal: 5 cents per 1000 chars received and transmitted. batch: .065 cents per card read + 0.3 cents per card punched + 0.1 cents per line printed
40-44	Terminal time - elapsed time /ID to /OFF (1/100ths hours). For a batch job, it indicates the amount of time between reading the ID statement and the time the last output line was printed.
45-46	Internal terminal number (TCB number) - 2 hexadecimal characters, i.e. "1A". "00" if batch.
47-48	Reserved (terminal identification number). ("01" for normal signon, "00" for added session.)
49-50	Physical line address - hexadecimal. Notice that only the last two characters of the terminal address appear. Thus, terminals on address 120 would show as 20.
51-53	Last 3 characters of userid.
54	Accounting record format identifier (">").
55-60	processing unit charge in cents. A surcharge is added to this rate when the user uses a region size of greater than 108K. The surcharge is 1% per 4K requested above 108K.
61-65	Terminal: connect charge in cents Batch: 60 cent handling charge + 150 cents per tape mount + 300 cents per disk mount.
66-69	Time of day of /ID (hours and minutes). This field may show the user's sign-on time greater than 24 hours. For example, if the system were loaded yesterday and a user signs on at 2 AM, the accounting record will show 2600 in this field. The user will see the correct time at the terminal.

70-76	Date in the form of 20JUL74.
77-80	Card sequence number.

Notes:

1. Reserved fields are not necessarily blank.
2. Batch jobs can be identified by columns 45-46 equal to zero ("00").
3. Columns 40-44 and 47-50 should be disregarded for batch jobs.
4. Columns 55 through 65 contain a charge based on the following rate table. These rates can be changed by modifying the table in the module MACCNT which is part of the program 'ACTDMP'.

Time of day	processing unit charge per 60 service units	Connect charge per hour
8:15- 9:30	$10	$3
9:30-12:00	$12	$4
12:00-14:00	$10	$3
14:00-16:30	$12	$4
16:30-18:00	$10	$3
all other times and all day Sat and Sun	$8	$1

MD1, MD2 Records (produced by ACTDMP and DSACT2 utilities)

Column	Contents
1-3	Identification: 'MD1' (from DSACT2) or 'MD2' (from ACTDMP, for a deleted data set).
4-19	File ownership id (16 characters, padded with blanks).
20-29	Current data and time: yymmddhhmm
30-36	Number of minutes to be billed, since last accounting.
37-42	Number of tracks to be billed. For an FBA device, 1 track = 32 blocks.
43	Disk device type: 1 printable character. 7=3350, 8=FBA, 9=3375, A=3380, B=3390, C=9345.
44	'B' for data set with BACKUP attribute, 'N' otherwise.
45-50	Volume name.
51-80	First 30 characters of data set name.

MUSL Cards (MUSIC Availability Log Records)

Column	Contents
1-4	Identification 'MUSL'
5-6	processing unit identifier (the contents of the 1-byte area at location 3E5 in main storage, in hex)
12-15	Time in form HHMM
16-21	Date in form DDMMYY
22	Identifying letter M
23	System State Indicator: =1 MUSIC IPL done =2 MUSIC system shutdown =3 (Reserved) =4 (Reserved) =5 Unscheduled system down (This is indicated by no previous shutdown record. The time and date is that of the last recorded activity.)

Privileges Required: VIP and FILES.

ADDPDS: Create a PDS from an IEBUPDTE tape file.

This program creates a MUSIC PDS from a tape in IEBUPDTE format (./ add format).

Usage:

```
/FILE 1 TAPE VOL(xxxxxx) RECFM(U)
/INC ADDPDS
FILE=n,PREFIX='code:',SUFFIX='.s',SELECT=,REPL=,PUBLIC=,SL=
<-- member names to be selected -- one per line if required
FILE=n, etc.
FILE=n, etc.
```

Control statements must be in order of ascending file number. The list(s) of files to be selected can terminate with an end of file, or "-" in column 1 if subsequent control statements are present. Selected files pertain only to the preceding control statement. Values specified on previous control statements are carried over to subsequent ones unless specifically modified.

Parameters:

FILE=n	*n* is a file number on the input tape.
PREFIX=	prefix of the newly created file name. Default is PREFIX=' '.
SUFFIX=	suffix of the newly created file name. Default is SUFFIX=' '. The total length of the PREFIX and the SUFFIX should not exceed 9 characters (or 14 characters if 'code:' is specified in the PREFIX).

REPL= if REPL=.true. is specified, then an existing save file is replaced by the newly created save file of the same name. Default is REPL=.false.

PUBLIC= if PUBLIC=.true. is specified, the files are saved public.

SELECT= if SELECT=.true. is specified, then only those './ add' statements with names that are specified in a save file (unit 5) are processed (one name is specified on each line starting at column 1 in the save file). The default is SELECT=.false. which processes all './ add' statements in the tape.

SL= if true, skips the header and trailer files of a standard labelled tape.

Privileges Required: None.

ATTRIB: Display File's Attributes

This program is used to list information associated with a file. Information such as access control, record length, record format, space allocated, number of records used etc. will be displayed.

Usage:

The program can be invoked in 2 ways:

1. Enter "ATTRIB name" or "AT name", where *name* is the name of the file desired. The file name may or may not have the user code prefix (code:), and special codes *COM and *USR may be used if desired. This will list the information about the specified file.

2. Enter ATTRIB or AT. This will go into conversational mode and allow information on many files to be retrieved. Enter /CANCEL or a blank line to terminate the program.

The following is a typical listing of the information associated with a file:

```
NAME=$HLP:HELP                PUBL XO XO(OWN)
TAG=
LRECL =   80      RECFM = 0200 (FC)    ACCESS CONTROL = 80 20 60 60
SPACE (K):  PRIMARY =      2     SECONDARY =      0     MAXIMUM =   -1
LINES =       2    HIGH BLK =      1  (MAX =      3)   EOF DISPL = 46
BACKUP NUMBER = 121     USAGE COUNT =      0    EXTENTS =  1
CREATED 01APR89  LAST OPN FOR READ 17APR89  FOR WRITE 12APR89 10:05:09
CREATOR: ABCD000    LAST WRITER: ABCD000
```

Refer to *Chapter 20 - File System* for a description of the access control (1 byte for the general control flags and 3 bytes for the access control bits). HIGH BLK indicates the number of 512-byte blocks used, and MAX is the number of 512-byte blocks available. EOF DISP indicates the displacement, in bytes, of the last byte of the file from the beginning of the last 512-byte block used. USAGE COUNT is the number of times the file has been used, and is equals to zero if the count is not kept.

Privileges Required:

No privileges for user's files or common or public files. LSCAN for other private files.

AUTOSUB: Automatic Submitter

This program reads a list of file names containing data to be submitted at specified times during the day. The program goes to sleep (using CALL DELAY) until the next specified time, when it wakes up and does the required submit.

This program is normally set up to run automatically when MUSIC is initialized. Consult the documentation on BTRM in the *MUSIC/SP Administrator's Guide* under Installing MUSIC/SP for more information.

The executor file for this program, complete with data and any /FILE statements needed, should be set up as the AUTOPROG for userid $MONxxx, where the subcode xxx is the device address of the pseudo terminal, defined as BTRM in the MUSIC NUCGEN job, to be used. The code $MONxxx must be allocated with the privileges LSCAN, FILES, and SYSCOM, and with unlimited time: PRIME(NL), NONPRIME(NL), DEFTIME(NL).

Since the program must not try to read from the terminal or write to the terminal, Unit 6 output should be directed to a file (or be defined as /FILE 6 DUMMY). Each time this program begins (i.e., at each MUSIC IPL), any items not done yet today but whose time has passed, are done. Each time an item is done, a record is added to a control file. The record format is:

```
*ID=nnnn DONE DDMONYY HH:MM:SS
```

nnnn is the ID number associated with the item (specified in the ID= parameter).

Virtual punches on device addresses 011 and 012 are used for the submit. The program assumes that these devices are defined for the MUSIC virtual machine.

Parameters:

The following parameters are for unit 5.

SPECS='filename' The name of the file containing the items to be done (see below).

CTLFIL='filename' The name of the control file. The control file must be initially set up as an empty file, with LRECL 80.

The list of things to do is specified by namelists in the specifications file (SPECS=). The parameters are:

ID=nnnn A unique ID number for the item. (Required.)

TIME=hhmm Time of day (24-hour clock) at which the action is to be taken. (Required.) Must be from 0000 to 2400. Avoid specifying the value 2400 (exactly midnight), because of complications due to the date changing at midnight.

DAYS=n1,n2,... Days of the week (Sunday=1, Monday=2, etc.) on which the item is to be done. If the current day is not one of these, the item is ignored at namelist read. Default is DAYS=2,3,4,5,6 (i.e., weekdays, Mon-Fri).

FILE='filename' Name of the file containing the card images to be submitted. % in column 1 is changed to /. (Required.) Normally these records are submitted to VM. They are submitted to the internal reader instead, if MUSIC is not running with VM and the SPLID name (see below) is "*" or starts with the characters "MUS". a submit to the internal reader can be forced by specifying SPLID='*INTRDR'. Internal reader queue 1 is used. A file for holding the records for each job is created with file name uuuu:@AUTOJOBnnnn.m, where nnnn is the 4-digit id number (see ID parameter

above) and m is a 1- to 4-digit sequence number. uuuu is the userid running autosub. These files are used only when the internal reader is used. for batch passwords, see the GENBPW parameter below.

JCLCTL='filename' This optional parameter specifies a file which can be used to adjust text (such as volume and file names) in submitted records. Refer to the comments in the source file $PGM:AUTOSUB.S for details.

GENBPW=TRUE This causes batch password records to be generated in the submitted MUSIC jobs. A password record is generated after each /ID record. The CODES privilege is needed for this. The default is GENBPW=FALSE.

SPLID='name' 1 to 8 character name of the virtual machine to which the data is to be spooled. Default is SPLID='MUSIC'.

CLASS='x' VM Spool class. default is CLASS='J' (as needed for MUSIC batch jobs). A queue number for the internal reader can be specified as CLASS='n', where n is a digit 1 to 9 (anything else causes internal reader queue 1 to be used).

TAG='xxx' TAG information for RSCS, if the data is spooled to RSCS. Default is no tag. *xxx* is 1-48 characters of tag information to be used on the VM command "TAG DEV uuu xxx" for the submit. This would be used if SPLID='RSCS'. The tag command is issued only if a nonblank tag string is specified. The default is blanks.

CMT='xxxxx' is 1 to 80 characters of comment info (optional). The comment is ignored by the program but can indicate what the item is used for.

START=yyyymmdd optional starting date for the item. Before this date, the item is ignored. For example, START=19860525. The current year is assumed if a year is not specified (i.e. if the value is less than 10000).

EXPIRE=yyyymmdd optional expiry date for the item. After this date, the item is ignored. For example, EXPIRE=19860619. the current year is assumed if a year is not specified (i.e. if the value is less than 10000).

SKIP=yyyymmdd,yyyymmdd,...
optional dates on which this item should be skipped. Up to 100 dates may be specified. If a year is not specified in a date (i.e. the value is less than 10000), the previous year in the list is used. For example, SKIP=19860521,0523,0525 skips 3 dates in May 1986. If a year is not specified in the first date in the list, the current year is assumed.

Note: Dates for start, expire and skip parameters must use 4-digit year (optional), 2-digit month, and 2 digit day of month, in that order. The dates are not checked for validity.

Privileges Required:

The code $MONxxx must be allocated with the privileges LSCAN, FILES, and SYSCOM. Also, the CODES privilege is needed if the parameter GENBPW=TRUE is used.

BPOOL: Display Buffer Pool Buffer Usage

This program can be used to find out about the utilization of the terminal buffer pool. It is helpful in determining whether the correct number of buffers have been allocated at NUCGEN time.

Usage:

This program is invoked by entering BPOOL on the terminal.

Privileges Required: CREAD.

BSTATUS: Display Status of Current Batch Job

BSTATUS, when executed on a terminal, wakes up every 5 seconds (approximately) and checks the status of the job running on batch. If there is a change, a line of information is displayed. This information includes the batch job code and subcode, the number of service units used so far, and the time of last activity (TLAT). If the job has ended, this is indicated.

Since BSTATUS only checks batch every 5 seconds or so, it is possible for short batch jobs to execute without being displayed by BSTATUS.

If a number is specified on the command, for example "BSTATUS 30", it defines the approximate number of seconds between checks. The default is 5 seconds.

Privileges Required: LSCAN and CREAD.

BUFLOG: List 3330 & 2305 Usage/Error Statistics Log

This program empties and prints the current contents of the usage/error logs for all 3330 devices on the system. If any of these counters overflow, the device automatically sends the log information to the processing unit which will cause MUSIC to print the information on the console. (When run under VM/370, all the counts will normally be zero as VM/370 will dump these counters just prior to performing this operation for MUSIC.)

The log for each drive consists of 24 bytes of data. This gives such information as the number of arm motions performed, the number of seek errors, the number of bytes of data read, the number of data checks, etc. The exact format and meaning of this data can be found in the IBM publication *Reference Manual for IBM 3830 Storage Control Model 1* (GA26-1592) or equivalent.

This program also prints any 2305 buffered logs which may have been dumped since MUSIC was last loaded. Further information on the usage of BUFLOG with 2305s can be found in the file $PGM:BUFLOG.S. Under VM/370, these logs are meaningless.

BUFLOG does not support the following direct access devices: 3310, 3340, 3350, and 3370.

Usage:

This program may be run from a terminal by typing BUFLOG or from batch with the control statements:

```
/INCLUDE BUFLOG
```

Privileges Required: VIP.

CDUMP: Interactive Storage Dump Utility

The CDUMP program allows the programmer to inspect and alter areas of main storage. If MUSIC is being run under VM/370, this program also allows an authorized user to dump VM's real storage.

Usage:

It is executed at a terminal by means of the command:

```
CDUMP
```

Parameters are entered separated by commas, with no extra blanks. This corresponds to standard MUSIC FORTRAN NAMELIST input.

A sample run of CDUMP is shown in Figure 17.1. Note that for lines in which all bytes are the same, only one byte is printed.

This utility can also be executed from batch. The control statement format follows.

```
/INCLUDE CDUMP
 control lines
```

Parameters:

START=	Specifies the first main storage location to be dumped. The default is zero.
LEN=	Specifies the number of bytes to be dumped. The default is 16.
M='name'	Specifies a system module name or entry point. Refer to nucleus map produced at NUCGEN time for valid names.
DISPL=	Specifies a value to be added to the one given on the START parameter. Default is 0.
REP=	Specifies the first main storage location to to be altered. The system will then prompt for the hexadecimal data.
CP=	Specifying CP=T will dump VM/370 real storage. Default is CP=F.
RET=	Specifying RET=T will stop the program. The program may also be terminated by the /cancel command in the usual way. Default is RET=F.
HELP=	Specifying HELP=T will print the names and default values of the various parameters.
LINE=	Number of bytes per line of output. The default value is 16. The only allowed values are 16 and 32.

SKIP=x,y After the regular dump is completed, x will be added to the START value and the dump will be performed again. In all, y segments are dumped. This allows the programmer to inspect small sections of main storage at regular intervals.

Several abbreviations and alternate keywords are allowed:

Keyword	Alternatives
START	S, ADDR, AD, A, BEGIN, B, FIRST, F
LEN	LENGTH, L, COUNT, C
DISPL	D, DISP
LINE	None
SKIP	None

Privileges Required:

LSCAN is needed to access the CDUMP file. If memory is to be dumped above the user region, the user must have the INFO or CREAD privilege. The CREAD privilege is required if VM Storage is to be dumped. The user must have VIP code privilege active in order to alter system storage.

```
*Go
cdump
*In Progress
CDUMP. ENTER S=,L=,M= OR HELP=T
?
start=z238
000238  00000398 FF953844 80000000 0000F0F6  *...q.n.. .....06*
?
m='tcs',len=128
833908  00010010 050F0003 00000000 00000065  *................*
833918  00000000 0000001D 00000001 00000000  *................*
833928  050784C7 00004650 00000317 01000040  *..dG...&.......  *
833938  0F400000 00000000 00000000 00000000  *. ..............*
833948  00  *.*
833958  0083FFE8 00000200 00000000 00000000  *.c.Y............*
833968  00000000 C3C6D4D5 F0F1F0F0 F1000000  *....CFMN01001...*
833978  81000000 00000000 0083D328 00000000  *a........cL.....*
?
first=z2c8,l=16
0002C8  700352D8 0F0398D8 0401D790 0101DA40  *...Q..qQ..P.... *
?
f=z352d8,length=32,skip=160,5
0352D8  00010020 00041001 00000000 00000000  *................*
0352E8  2FBE31E4 2FBD0000 C9C8CA00 006D006D  *...U....IH..._._*

035378  00020021 0404940B 00010001 00010006  *......m.........*
035388  2F503176 2F4F0003 C9C8CA00 006D006D  *.&..._..IH..._._*

035418  00030022 01041103 00000000 00010012  *................*
035428  2EE23108 2EE10000 C9C8CA00 006D006D  *.S......IH..._._*

0354B8  00040023 0104140B 00140014 00010002  *................*
035558  2E742E74 30990B01 C9CAC800 006D006D  *.....r..I.H._._*

035558  00050024 0204150B 00070007 00000000  *................*
035568  2E06302C 2E050000 C9C8CA00 006D006D  *........IH..._._*
?
s=z1000
001000  47F0F15A 47F0F162 47F0F54C 47F0F588  *.01!.01..05<.05h*
?
rep=z1000
ENTER REP
?
00000000
ORIGINAL STORAGE:
001000  47F0F15A 47F0F162 47F0F54C 47F0F588  *.01!.01..05<.05h*
NEW STORAGE:
001000  00000000 47F0F162 47F0F54C 47F0F588  *.....01..05<.05h*
?
/cancel
*Terminated
*Go
```

Figure 17.1 - Sample run of CDUMP

CHKDISK: Check System Data Sets for Correct Format

This program is used to detect areas on disk which have not been formatted correctly. It is also a convenient way of reading a data set to test for read errors.

CHKDISK processes only count-key-data (CKD) disk devices. It does not process FBA devices such as 3310 or 3370. Only the first extent of each data set is checked. All tracks of a data set are assumed to have the same number of records. The program checks the count area of each record to verify that the key length and data length match the values from the format-1 DSCB in the VTOC.

Usage:

To run the program, type CHKDISK, then enter:

```
VOL='volume',DSN='data set name'
or
VOL='volume',DSN='ALL'
```

DSN='ALL' processes all data sets on the volume. Initial defaults are the IPL volume (UCB 0) and DSN='ALL'.

Notes:

1. This program does not process FBA volumes (3310, 3370, 933x).

2. Only the first extent of a data set is checked.

3. All tracks of the data set are assumed to have the same number of records. The program checks the count area of each record to see that the key length and data length of the record match the values from the format-1 DSCB (key length and blocksize - key length). The block size in the DSCB is assumed to be *key len + data len.*

Privileges Required: LSCAN, CREAD, DREAD.

CHKFILES: Check All Files

This utility provides a way of finding all files which have errors such as index/directory mismatch (error 67) or directory in error (error 65). The bad files could then be processed by a program such as FIXINDEX or FIXINDEX.AUTO.

CHKFILES does an EXTRACT request for each file in the index. Only private index entries are used; common entries are skipped. For each non-zero MFIO return code from EXTRACT, the program displays a message and writes a record to the file defined on unit 1 (logical record length must be at least 100). By default, unit 1 is defined as new file FILE.ERRORS. You can edit the file $PGM:CHKFILES and change this if desired.

Usage:

Enter CHKFILES to run the program.

Privileges Required: LSCAN.

CMSTAPE: Retrieving CMS Tape Files

This utility program retrieves CMS files which were dumped to a tape by using the CMS TAPE DUMP command. The retrieved CMS files will be saved under separate MUSIC files. It can also be used to scan through the CMS tape to give information of the CMS files contained in it. The information given for each CMS file includes file name, file type, file mode, record format, logical record length, and number of records. Both block sizes of 805 and 4101 bytes on the CMS tape are supported.

The name of the file created for a CMS file is in the format of *fn*.*ft*, where *fn* is the file name, and *ft* is the file type of the CMS file. Sometimes, a fixup character will also be used in the name to make it a valid MUSIC file name, or to make the name unique in the user's library. For more details, see the description of the FIXUP parameter below.

The record format of the file created is similar to that of the CMS file. If the CMS file has a record format of F (fixed), the record format FC (fixed compressed) will be used for the file created. Likewise, a V (variable) record format CMS file will result a VC (variable compressed) record format MUSIC file.

Usage:

The following control statements are used to execute the program. Notice that the input CMS tape is defined with a record format of U (undefined).

```
/FILE 1 TAPE VOL(...) RECFM(U) SHR     (input CMS tape)
/FILE m ...     (if needed by the NAMES=m parameter)
/INCLUDE CMSTAPE
parameters separated by commas (see below)
filename filetype     )
filename filetype     )   if SELECT=TRUE is specified
...                   )
```

Parameters:

INPUT=n	Input unit number of the CMS tape. Unit number 15 cannot be specified as it is used for a work file. Default is INPUT=1.
TAPFIL=n1,n2,...	Specifies the physical tape files in the CMS tape to be processed. A maximum of 99 tape files can be specified. The numbers specified must be in ascending order. If TAPFIL=0 is specified, the entire CMS tape will be processed (the tape will be read until two consecutive EOF marks are encountered). Default is TAPFIL=1, meaning only the first tape file is to be processed.
NAMES=m	If specified, the names of all the created files will be written to unit *m*, one file name per line, starting on column 1. As above, unit number 15 cannot be used. Default is NAMES=0, meaning no names are to be written.
LIST=FALSE	Specifies that the names and the corresponding information of the created files are not to be printed. Default is LIST=TRUE.

REPL=TRUE — Specifies that if a file under the name of *fn.ft*, where *fn* and *ft* are the file name and file type of the CMS file retrieved respectively, already exists, then it will be replaced by the new contents. Default is REPL=FALSE (see below for more details).

FIXUP='x' — Specifies a fixup character 'x' to be used for generating an alternate name for the file to be created if the file under the name of *fn.ft* (see above) already exists and is not to be replaced. The fixup character will be added to the end of *fn.ft* and the resulting name is tried again. At most three such retries are done per file. The fixup character will also be used in the case where *fn* starts with a digit, which, of course, does not conform to the MUSIC file naming convention. In this case, the fixup character will replace the digit. The valid fixup characters are '$', '#', '@', and the letters A to Z. If an invalid fixup character is specified, the default character, which is '$', will be used.

SCANTP=TRUE — Specifies that the input CMS tape will only be scanned for information of the CMS files contained in it. No files will be created. If TAPFIL=0 is specified, the entire tape will be scanned. Otherwise, only the tape files specified in the TAPFIL parameter will be scanned. Default is SCANTP=FALSE, meaning files will be created for the CMS files contained in the tape.

SELECT=TRUE — Specifies that only certain CMS files are to be retrieved from the CMS tape. The file name and the file type of the CMS files to be retrieved are specified after the parameter line. For each CMS file to be retrieved, the file name must be specified on column 1, and the file type on column 10 on a new line. A maximum of 99 CMS files could be specified. Notice the TAPFIL parameter also applies for the tape files to be searched for the desired CMS files. The default is SELECT=FALSE, meaning that all CMS files in the specified tape files are to be retrieved. This parameter will be ignored if the parameter SCANTP=TRUE is specified.

Note: There is a temporary work file defined on unit 15 in the file CMSTAPE. The primary space defined for the temporary work file is 200K bytes. If this work file does not have enough space to process a particular CMS file, just run the program again to retrieve the failing CMS file, by using the SELECT=TRUE parameter, with a bigger work file defined before the line /INCLUDE CMSTAPE in the job set up mentioned above. The work file must have a record format of V (variable).

Privileges Required: None.

CODUMP: User Code Table Dump/Restore

The MUSIC user code table dump/restore program has several functions. It can copy the code table to magnetic tape or a sequential file, as 512-byte records. It can restore tapes created by the dump function. It can produce printed lists of authorized users. It is recommended that the user code table be regularly backed up to tape with this program. This ensures that if any hardware or software problems cause the code table to become unusable, a recent good copy is available. It should be noted that users can frequently change the code table by means of the user profile program (PROFILE). If a backup copy of the code table were to be restored and it was not quite recent, many user's passwords, autoprogs, operating defaults, etc., would be incorrect.

Another use of this program is to condense the code table. As many additions and deletions are made, space in the code table index can become unavailable. This space can be reclaimed by dumping and immediately restoring the code table. This should be done whenever the number of index entries used starts approaching the maximum number of index records. Both these numbers are printed during every dump or restore of the

user code table.

There should be no other users on MUSIC when a code table restore is done. Otherwise user updates to code records (from sign-ons or the PROFILE program) may produce code table errors.

The program creates either one or two magnetic tapes. If two are created, they are identical. The number of tapes to be created is specified by listing the unit numbers onto which the dumps are to be done (UNITS= parameter). To obtain a printout only, simply specify DUMP=T and the appropriate print option. A record length (LRECL) of 512 must be specified on the /FILE statement.

If an error is found while the code table is being dumped, a message is issued, along with a dump of the appropriate code and index records. Serious errors stop the program with the message `PROGRAM TERMINATED ABNORMALLY.` For other errors, the dump continues, but one or more code records may be missing from the output. Restoring the dump should correct the errors, but the missing codes will be lost and should be re-added manually, using CODUPD. If codes are found not to be in increasing alphabetical order, the dump file should be sorted before it is used in a restore job.

Usage:

```
/FILE x TAPE VOL(volnam) BLK(bbbbb) LRECL(512)
/INCLUDE CODUMP
parameters separated by commas (see below)
```

Namelist Input:

DUMP=T or F	F means do not dump. Default is T.
UNITS=x,y	Gives output units for dump. Default is 0.
PRINT=n	Print format for code listing. Default is 1. 0 no listing 1 listing of codes and subcodes
RESTOR=z	Gives input unit for restore. 0 means do not perform restore. Default is 0. Note: a restore cannot be done in the same job as a dump, so DUMP=F must be specified for a restore job.
RSTUCR=T or F	RSTUCR=T restores UCR limits as codes are restored. Default is RSTUCR=T. This option applies only to a restore operation. The UCR limits are stored in the Save Library index and control the allocation of file space. When codes are dumped, the UCR limits are obtained from the SL index and dumped with the code records. During a restore with RSTUCR=T in effect, the UCR limits are set to their values at the time of the dump. A new UCR is created if one does not already exist.

Privileges Required: LSCAN and CODES or MAINT.

CODPRV: List Privileged Userids

CODPRV lists all privilged userids (codes).

Privileges Required: LSCAN and CODES.

CODUPD: Userid Authorization

The Code Table Update Program (CODUPD) enables a privileged user, normally the system administrator, to modify the table of authorized MUSIC users. Each user is identified by a 1-16 character userid (user code), and is represented in the Code Table by a record which contains the user's userid, password, time limits, privileges, and other attributes. The userid (or sign-on code) is also the key used for accessing records in the Code Table.

The CODUPD program processes keyword-type commands, which are entered conversationally from a terminal or read from a file. Only one user may run CODUPD at a time (but see the CODXXX program below). The ADD command adds a new userid record to the table (enabling the user to sign on MUSIC). The DELETE command removes a record from the table. The CHANGE command modifies items in a record. The GET command displays information from a record. Changes made to the code table record of an active user take effect the next time the user signs on; they do not affect the user's current session.

The Code Table is made up of two system data sets: the code table index and the code table itself. The maximum number of userid records in the Code Table depends on the sizes of these two data sets, but is normally a number from 20000 to 50000. Refer to the descriptions of data sets SYS1.MUSIC.CODINDX and SYS1.MUSIC.CODTABL in the section "Direct Access Storage Usage" in this manual.

There is another record associated with each user: the User Control Record (UCR), which resides in the Save Library index. The UCR controls allocation of file space in the Save Library. It contains the limit for the userid's total file space and the limit for the size of each file. There is one UCR per userid; different subcodes of a userid share the same UCR. When a userid is added to the Code Table, CODUPD automatically creates a new UCR for the userid if a UCR does not already exist. When the last subcode of a userid is deleted, CODUPD automatically deletes the UCR. The UCR and UCRFIX utilities can also be used to alter these control records.

The User Profile Program (PROFILE) is a restricted version of CODUPD. It allows the user to modify some of the items in the user's own Code Table record, such as passwords and default time limit. PROFILE is described in the *MUSIC/SP User's Reference Guide*.

Further information on the structure of the Code Table and the internal workings of the update program may be found in *Chapter 18 - System Internals*.

Usage:

To run CODUPD at a terminal and enter commands conversationally, type the command

```
CODUPD
```

When the "?" prompt appears, you can enter any of the CODUPD commands (ADD, GET, etc.) For help enter the command HELP. Terminate the program by entering END.

To pass a single command to the program and then terminate, enter the following:

```
CODUPD command
```

where *command* is the CODUPD command to be executed. For example, entering CODUPD GET ABCD in *Go mode displays the code ABCD.

There is a version of CODUPD called CODXXX, which allows userids to be displayed and changed, but does not allow adds or deletes. The advantage of CODXXX is that multiple users can run it at the same time, and while a normal CODUPD is running. See the topic below.

To have CODUPD read commands from a file (for example CMDFILE), run the following job:

```
/INCLUDE CODUPD5
/INCLUDE CMDFILE
```

CODUPD5 is a version of CODUPD set up to read commands from unit 5 instead of from unit 9. This job can be run at a terminal or on batch.

Privileges Required:

Usually the system administrator userid ($000) is used to run CODUPD. Only one user may run CODUPD at a time. The user must have at least the CODES and LSCAN privileges. Also, as a security precaution, the user is not allowed to add, change, delete or get any userid with a privilege that the user does not have. An attempt to do so will result in an ACCESS DENIED message. An example of this is trying to get a userid that has the VIP privilege when the user has not used the /VIP ON command. Also, a userid with VIP must use the command /VIP ON before using the PROFILE program.

VIP Privilege Notes

The VIP privilege gives the user only the **ability** to request the VIP privilege. That is, when the user signs on, the VIP privilege is not automatic, but has to be requested by means of the command:

/VIP ON

This command may be entered in command (*Go) mode. The actual privilege may be turned off by the command /VIP OFF. This implementation is to ensure that the VIP privilege is not used accidentally (and destructively).

Note that to run the user profile (PROFILE) program, a user whose userid has the VIP privilege must first set /VIP ON. Also, before running CODUPD to access or change userids having the VIP privilege, VIP must be set on.

If VIP userids are run from batch, no special commands are needed to enable the privilege. For this reason, all VIP userids that are to be allowed from batch should be given the RESTR option (restricted access). This ensures that the userid cannot be used except on explicit operator authorization, by means of the console command /CTL CD-ON. It is also recommended that the RESTR option be given to all batch userids having special privileges.

Use of Subcodes

Use of a 1-8 character "subcode" is optional. A subcode allows users to share the same Save Library but have different userid attributes. File ownership is based on the userid excluding any subcode. This part of the userid is referred to as the "ownership id". The total length of the userid cannot exceed 16 characters.

It must be remembered that different subcodes of a userid share the same files and the same UCR (as discussed above), since file ownership is based on the userid without the subcode.

Because of the naming convention used for the /INPUT, Editor log, and Unit 10 Holding files, it is strongly recommended that different people using the same userid be assigned different subcodes. Otherwise, for example, they would share the same /INPUT file (@INPUT.sss, where *sss* is the subcode) and one user entering /INPUT would erase the other user's file or give a FILE IN USE message.

Typical Sign-on Userid Options

The CODUPD command options are described in detail in the topics which follow, but this topic will give you an idea of what is involved in creating a userid.

The following is a typical userid allocation command for a student or project member.

```
ADD ABCD PW(SAMPLE) PRIME(60) NONPRIME(180) DEFTIME(30) -
  BATCH(60) MAX$(300) TOTLIM(200) FILLIM(500) -
  SNGL NOCOM NAME(JOHN SMITH)
```

This creates the userid ABCD with the password SAMPLE. The time limit per terminal job is 60 SU (service units) during prime time (normally defined as 9AM-5PM weekdays), 180 SU during nonprime time, and 30 SU by default. The time limit for batch jobs is 60 SU. The userid has a fund limit of $300 and can store up to 200K on the Save Library (not counting &&TEMP temporary files) and can allocate individual files (including temporary files) as large as 500K. Only one terminal session is allowed at a time (SNGL). NOCOM prevents the user from storing files in the common index. The NAME parameter is optional.

A userid allocation for a research or commercial user or for a project or course supervisor would typically have larger numbers in the time limit, MAX$, TOTLIM, and FILLIM fields.

An instructor who will be using the TRANS$ program would have a userid that includes the SUPV privilege.

Userids for the MUSIC Support Staff would typically include the LSCAN privilege, and NOCOM would be omitted. Other options and privileges would be assigned as needed.

Format of CODUPD Commands

Commands may use columns 1 through 80 of each input line. The general format is:

```
command   keyword(value) keyword(value) ...
```

where:

command — is the command name (for example, ADD, CHANGE, DELETE or GET) or its abbreviation. It must start in column 1 and must be followed by at least one blank.

keyword — is a keyword (or its abbreviation) that identifies a userid option or parameter. The first keyword after the command name is special; it is the userid that the command is referring to. The keywords after the userid can be in any order.

(value) — is the value being assigned to the userid option. It is enclosed in parentheses and must immediately follow its keyword, with no blanks between. Some options require a value and some don't. For example, the password option has a value that is a character string from 1 to 16 characters long, e.g. PASSWORD(ABCDE). The SNGL option (concurrent sessions not allowed) is an example of a keyword not followed by a value.

The value may be a character string, or a number, or in some cases a list of items separated by blanks or commas. For a character string, the string may contain special characters such as blanks, commas, and parentheses (but left and right parentheses must be matched); trailing blanks are ignored but leading blanks are not.

The parameters (i.e. the *keyword*(*value*) items) are separated from each other by 1 or more blanks and/or commas.

If the command is too long to fit in one line, it can be continued on another line by ending the line with a hyphen (-). The "-" character is normally preceded by a blank, to provide a blank between the parameters. A parameter can be split between lines if necessary. Several continuation lines can be used.

Here are some examples of continued commands:

```
ADD ABCD001 PASSWORD(PWWWWWWD) PRIVONLY MAX$(750) -
  NAME(JOE PROGRAMMER) ID(123-4567) -
  TOTLIM(300),FILLIM(400)

CHANGE ABCD001 AUTOPROG(PROG1.SETUP) TAG(THIS USERID IS USED -
  FOR DEMONSTRATION PURPOSES) SNGL
```

A command that starts with an asterisk (*) is treated as a comment (it is not processed). However, comment lines may not be used between lines of a continued command. The COMMENT parameter can be used to add comments to a command.

Commands can be typed in mixed upper and lower case; they are automatically converted to upper case by the system before they are passed to CODUPD, except that values for some keywords (such as NAME and TAG) are left as entered.

Summary of CODUPD Commands and Keywords

The following table lists the command names, their abbreviations, and the keywords which may be used with them. Some of the keywords have abbreviations or aliases. The commands and keywords are described in detail later.

Note that the first parameter of an ADD, CHANGE, GET or DELETE command is always a userid specification. The remaining keyword-type parameters may appear in any order.

Command	Abbreviations	Keywords
ADD	A	userid specification SUBCODE(xxxxxxxx) TYPE(n) PASSWORD(string) BATCHPW(string) PWEXP(n) NEWPW sign-on options: FIXPW, SNGL, etc. operating defaults: NOCOM, etc. privileges: LSCAN, FILES, etc. PRIME(n) NONPRIME(n) DEFTIME(n) BATCH(n) AUTOPROG(filename) ALWAYSPROG(filename) FIRST(n) LANGUAGE(langname) ROUTE(string) TERMINAL(string) ITABS(n1 n2 n3 ...) OTABS(n1 n2 n3 ...) TAB(x)

		BACKSPACE(x) IDLE(x) HEX(x y) MAX$(n) ADD$(n) NOW$(n) TRK/UDS(n) TOTLIM(n) FILLIM(n) XSESSIONS(n) START(yyyy/mm/dd) EXPIRY(yyyy/mm/dd) NAME(string) ID(string) TAG(string) LIKE(userid) COMMENT(string)
CHANGE	C, CH	Same keywords as for the ADD command, except that TOTLIM, FILLIM, and LIKE are not allowed.
GET	G	userid specification SUBCODE(xxxxxxxx) SHOW$ SHOWSPACE BRIEF DUMP COMMENT(string)
DELETE	DEL	userid specification DELUCR DELMAILBOX COMMENT(string)
END	(none)	
HELP	(none)	
.LIST	(none)	
.NOLIST	(none)	
.PRINT	(none)	
.NOPRINT	(none)	

General Information on CODUPD Commands

ADD Creates a new record in the Code Table, for the specified userid. The userid must not already exist in the table. If a subcode is added, the total length of the new userid cannot exceed 16 characters. Default values are used for any fields which are not specified on the ADD command (see below).

CHANGE Changes the contents of an existing record in the Code Table. The userid and subcode in the

record cannot be changed, since they make up the 16-character key by which the record is accessed, via the Code Table Index. Only the specified fields are changed.

GET — Displays information from a Code Table record. Options (such as SHOW$, SHOWSPACE) can be used to select specific information. If no options are used, the entire userid record is displayed in a readable format. The display includes some fields from the userid record that do not correspond to command parameters, such as the date of creation of the record, date and time of last sign-on and last batch job, date of last sign-on password change, and date of last batch password change. See the sample output below.

DELETE — Removes a record from the Code Table. The record is usually displayed as it is deleted.

END — Terminates the CODUPD program.

HELP — Displays information on how to use CODUPD. The text is taken from the file $COD:@HLP.CODUPD. (The corresponding file for the PROFILE program is $HLP:@GO.PROFILE.)

.LIST — Causes subsequent command input lines to be displayed on unit 6 as they are read. The default is .LIST for batch jobs and CODUPD5, and .NOLIST for CODUPD terminal jobs.

.NOLIST — Suppresses the display of command input lines.

.PRINT — Causes the userid record to be displayed after any successful ADD, CHANGE or DELETE commands that follow the .PRINT command. This display is always omitted when the command specifies a range of userids. Default is .PRINT for CODUPD and .NOPRINT for CODUPD5.

.NOPRINT — Suppresses the automatic userid record display.

Return Codes for CODUPD and PROFILE

0 — No errors or unusual conditions.
4 — One or more commands were invalid or unsuccessful.
8 — CODUPD is already in use; try again later.
16 > — A fatal error occurred. Note that some commands preceding the error may have been successful. Also, a large return code results if the program is ended by /CANCEL instead of by the END command.

Description of CODUPD Keywords

In the descriptions that follow, any abbreviations or aliases for the keyword are shown at the right. Usually the shortest abbreviation is shown; in most cases, intermediate forms are also allowed.

Many option-setting keywords have a negative form obtained by putting NO in front of the keyword. For example, SNGL, NOSNGL; FILES, NOFILES. One of them turns the option on and the other turns it off.

Some keywords define numeric limits, such as limits on job time or file space. For these, a value NOLIMIT can be specified instead of a number. Abbreviations NOLIM and NL can be used. For example, PRIME(NOLIMIT), MAX$(NL), TOTLIM(NOLIM). In the userid record, NOLIMIT is represented by the value -1.

Many options can be removed or reset to their normal values by specifying a null value: keyword(). For example, to remove a NAME field, specify NAME(), which sets the field to all blanks. OTABS() removes

all output tabs. EXPIRY() removes the expiry date. TAB() undefines the input tab character, etc.

userid specification

This must be the first keyword on an ADD, CHANGE, GET or DELETE command. It is a 1-16 character userid.

Wild characters * (matches any group of 0 or more characters) and ? (matches any single character) can be used in the userid, except on the ADD command, to refer to several userids.

A range of userids, e.g. XX01-15 or XXAA-BC, can also be specified. See the topic below. To avoid confusion with a userid range, do not define a subcode that starts with "-".

The userid must start with a letter (A to Z), or digit (0 to 9), or one of the four special characters, $, @, _, or #. Avoid assigning userids starting with the $ sign as they may conflict with the userids used to store system files. (The topic "Usage of $ Userids on MUSIC" in Chapter 22 - System Programming contains a list of these $ userids.)

SUBCODE(xxxxxxxx) SC SUBC

Specifies a 1 - 8 character subcode. Letters, digits, and special characters can be used, although special characters other than $ # @ / _ and . should be avoided. If no subcode is wanted, omit the SUBCODE parameter on the ADD command.

SUBCODE(*) can be used on a GET or CHANGE command. It means that the command will be applied to each subcode of the userid. In general, wild characters * and ? can be used in the subcode, except on the ADD command.

A range of numeric subcodes, e.g. SUBCODE(001-025), can also be specified. See the topic below.

TYPE(n)

A type number (0 to 255) can be assigned to each userid in the Code Table. It is copied to the XTCB at sign-on. The intention is to use the type number in various authorization files to specify e-mail access, e-mail retention period, access to ADMIN, etc. It should indicate the general category of user, but NOT detailed information such as course number or department number.

Each site can choose their own numbering scheme, but a suggested scheme is given below.

If no type is specified when the userid is created, and for userids that exist from before, the type number is 0 (general user).

CODUPD Command Examples:

```
ADD ABCDE TYPE(30)
CHANGE XYZ TYPE(40)
```

A TSUSER call is provided to get the type number for the current userid:

```
CALL TSUSER(15,NUM)
```

Field in Code Record:	$CDTYPE	(1 byte at record + x'18')
Field in XTCB:	XTBITYP	(1 byte at XTCB + x'3F')

(Note that the
above field lengths and locations may change in the future.)

Suggested Userid Types:

0- 9	Student; general user
10-19	Anonymous access userid (e.g. CWIS)
20-29	Demo or free userid; external or guest user
30-39	Course instructor; group supervisor; academic
40-49	Special user; university officer; company executive
50-59	Research user
60-69	Commercial user
70-79	University or company administrator; non-academic
80-89	Computing Center administrator/staff
90-99	Computing Center operations; system maintenance
200	MUSIC system administrator (userid $000 or equivalent)
210	MUSIC system userid ($xxx)

PASSWORD(string) PW PASSW

The terminal (sign-on) password, from 1 to 16 characters. Letters, digits and special characters can be used. It must not contain imbedded blanks.

PASSWORD() results in no password being required at sign-on time. This should be used with caution.

PASSWORD(*NOLOGON) defines a special password that prevents sign-on except as a BTRM.

It is strongly recommended that all users be assigned passwords, and that users be encouraged to change their own passwords frequently (using the PROFILE program). The date that each of the passwords was last changed is remembered in the userid record and is displayed by the GET command.

PWCASENS

The Profile option PWCASENS can be used to make the user's sign-on password case sensitive. When this option is set, the user's sign-on password is considered to be mixed case, and the exact upper or lower case letters must be entered at sign-on time. WARNING: When you set PWCASENS, you should also set a new password, so that you know the exact case of the letters in the password.

BATCHPW(string) BPW

The password for batch jobs, from 1 to 8 characters. Letters, digits, and special characters can be used. A batch password may or may not be required, depending on the installation. BATCHPW() results in no batch password. A batch password should be specified if the user will be submitting jobs to MUSIC batch. If no batch password keyword is specified on an ADD command, the batch password is set to the same value as the terminal password.

PWEXP(n)

A password lifetime can be specified for each userid, if desired. *n* is a number of days. After *n* days have passed, the user must choose a new password, which is valid for another *n* days. This feature improves security by requiring users to change their passwords regularly. Different userids can have different password lifetimes. Unlimited lifetime can be specified by PWEXP(0). The auto-program that enforces password changes is $COD:PWEXP.

NEWPW

This is used on an ADD or CHANGE command for setting a new terminal password. After the command is entered, you will be prompted to enter the new password in a non-display field.

Sign-on Options

Various keyword options can be set to control the sign-on procedure. They are:

SNGL — Only one terminal at a time can be signed on to this id. This prevents the userid from having multiple concurrent sessions on different terminals. If the user tries to sign on and the userid is already signed on, the user is given the option of cancelling the previous session.

FIXPW — The user is not allowed to change the terminal or batch password.

NONCAN — (or NOCAN or NOCANA) The user may not cancel the auto-program (AUTOPROG) that runs after sign-on. Also, the user may not change the auto-program name.

UNAME — A user name parameter must be entered at sign-on time. (This is not the CODUPD NAME parameter.)

DISABLE — Marks the userid record as not authorized, but does not delete it from the Code Table. The user will not be allowed to sign on or run batch jobs. The userid can later be re-authorized by the ENABLE option.

RESTR — The userid cannot be used from batch unless the operator specifically allows it by entering the console command "/CTL CD-ON". This can be used as a security precaution for privileged userids.

ANYSC — Allows use of any subcode. Applies to 4-character userids without a subcode.

The opposites of the above options are: NOSNGL, NOFIXPW, CAN (or CANA), NOUNAME, ENABLE, NORESTR.

Operating Defaults

These keyword options control various things during the user's session or job.

NOINPROG — This option suppresses the `*In progress` message that is displayed at the beginning of terminal jobs.

NOINVIS — Makes userids visible to programs like FINGER. Use the INVIS option if you do not wish other users to know that this userid is signed on.

SLASH — The slash (/) character is required at the beginning of each command in *G0 mode. E.g. /LIST must be typed instead of LIST.

EXEC — The implied /EXEC command is not allowed. I.e. /EXEC filename must be typed instead of *filename*.

PRIVONLY — (Private only.) The user is not allowed to create files which are accessible by other userids. The options PUBL and SHR cannot be used when saving files. Also, new UDS files must be private.

NOCOM — The user is not allowed to create files in the common index. This is less restrictive than PRIVONLY. SHR can be used when saving files, but not PUBL. If another user wants to read a file created as SHR, this user must specify the owner's userid on the front of the file name, as in /LIST ABCD:FILEXYZ.

FIXALPROG — The user is not allowed to change the always program name. Also, the command /CANCEL ALL cannot be used to cancel an always program. Abbreviation FIXALP may be used.

NOMULTI Access to multi-session is restricted to that initiated from programs. Multi-session commands and functions keys are disabled.

SIGNOFF User is automatically signed off when auto program finishes.

JA1 to JA8 These 8 options refer to the Submit Facility. If the model JCL for a submit processor has a nonzero access restriction number n (1 to 8), then option JAn must be set in the user's userid in order for the user to submit jobs to that processor. See *Chapter 8 - Job Submission and Retrieval Programs* in this manual for a discussion of this topic.

The opposites of the above options are: INPROG, INVIS, NOSLASH, NOEXEC, NOEDMSG, NOPRIVONLY, COM, NOFIXALPROG, MULTI, NOSIGNOFF, NOJA1, ..., NOJA8.

Privileges

These keywords grant special privileges to the user, allowing this person to do things that nonprivileged users are not allowed to do. Many of the system maintenance programs require one or more of these privileges. The privileges are listed in approximate order of increasing power.

SUPV Save Library supervisor privilege. The user can create, read, modify and delete any file or UDS file, provided the first 2 characters of the owner's userid match the first 2 characters of the user's userid. Also, the user can run the TRANS$ program to transfer funds and assign new passwords. For example, userid XY00 with the SUPV privilege can supervise any userid starting with XY. This is useful for courses and project groups.

LSCAN The user can inspect private files, and use the CODE parameter on the /LIBRARY command. Note that most system utility programs have private executor files, and therefore at least the LSCAN privilege is required in order to run them.

SYSCOM Allows the user to send data directly to a VM virtual punch or printer, which may be spooled to MUSIC, some other virtual system, or RSCS. Without this privilege, only the standard programs (SUBMIT, VMSUBM, etc.) can be used to spool data.

CREAD Lets the user inspect any location in main storage, even if the storage is fetch protected. VM's real main storage can also be read (via the CDUMP program) if the MUSIC virtual machine is authorized to do so.

INFO The user can run general system information programs.

FILES The user can create, read, modify and delete any file or UDS file.

SYSMNT The user can run some system programmer maintenance programs. For example, the Load Library and System Subroutine Library can be updated.

CODES Access to the Code Table is allowed. The user can run the CODUPD program.

DREAD The user can read from any location on disk.

MAINT The userid can run computer room maintenance programs, such as accounting and file archive utilities.

VIP The ability to modify any location on disk, to modify any part of MUSIC's main storage, and to run any utility program not provided for by the above privileges. For each terminal session, this privilege is active only after it has been specifically

requested by the user typing /VIP ON.

JOBLIM This option allows a program to increase its execution time limit.

PRIV12 to PRIV15
These keywords correspond to privilege bits which are not currently used. Only the userid $000 has these privileges. (There should be at least one userid in the system, namely $000, that has all possible privileges, otherwise additional future privileges could not be assigned.)

To remove a privilege, simply type NO before the corresponding keyword, as in NOSUPV, NOLSCAN, etc.

It is recommended that privileges be granted and used with great care. The security of the entire system disappears if privileges are granted without discretion. No privilege (except possibly SUPV), should ever be assigned to personnel not directly responsible for the maintenance of MUSIC.

The FILES, LSCAN and SUPV privileges allow a user to have access to files belonging to other users. Refer to the topic "Working with Files" in *Chapter 20 - File System* in this manual for techniques of how to access and work with files belonging to other userids.

PRIME(n) PRI
Maximum number of service units (SU) per terminal job during prime time. Prime time normally means 9:00 AM to 5:00 PM on weekdays (Monday through Friday). PRIME(0) means that the user cannot sign on during prime time. PRIME(NOLIMIT) can be specified. NOLIMIT can be abbreviated NOLIM or NL, as in PRI(NL). The number *n* in this and the other time limit parameters cannot exceed 32000.

The definition of prime time can be changed by changing the system module URMON, near labels TPRON and TPROF.

A service unit is a measure of work done by a job, and is defined elsewhere in this manual. On some systems 1 SU is equal to 1 second of processing unit processing.

NONPRIME(n) NONPR
Maximum number of service units (SU) per terminal job during nonprime time (see the description of PRIME above). NONPRIME(0) means no access, and NONPRIME(NOLIMIT) means no limit on job service units.

DEFTIME(n) DT DEF
Default time limit, in service units, for terminal jobs that do not specify a time limit via the "/SYS TIME=m" control statement. The number *n* must be from 1 to 32000, or NOLIMIT.

BATCH(n) BAT
Maximum number of service units (SU) that any one batch job is allowed to use. BATCH(0) means that the user may not run batch jobs. BATCH(NOLIMIT) may be specified.

AUTOPROG(filename) AUTO PROG
Specifies the name (1 to 22 characters) of a file that is to be executed automatically each time the user signs on. The auto program can be made non-cancellable by the NONCAN option (described above under "sign-on options"), if desired. To remove the auto program option, specify AUTOPROG().

A sample autoprogram is provided for doing extra user authentication at sign-on time. See $pgm:guard1.sample.s . It uses pre-established key values known only to the user's workstation and to the autoprogram. The autoprog sends a random, time-of-day dependent number (a "nonce") to the

workstation. The user runs a program on the workstation to combine the nonce with the fixed keys, giving a one-time password, which he or she enters and which the autoprog verifies. The only values transmitted over the communications channel are the nonce and the one-time password, both of which are different for each sign-on. This make connection over the Internet more secure.

ALWAYSPROG(filename) ALWAYS ALPROG
The name (1 to 22 characters) of a file that is to be executed automatically every time the terminal would otherwise be in *Go mode. This means that whenever a terminal job ends and no additional job is scheduled, the always program is run. For example, the always program could display a menu of available functions and let the user choose which one to do next. To remove this option, use ALWAYSPROG(). Note that the PROFILE program allows the user to change (or remove) the always program name, unless the FIXALPROG option is specified for the userid (see "operating defaults").

FIRST(n)
The number n is from 1 to 255 and identifies a one-time program that is to be run the first time the user signs on. For example, this could be a system program that prompts for information about the new user and stores it in a log file. The program id numbers are defined in the system SIGNON module. If an auto program also exists, the one-time program is run first. To remove this option, specify FIRST(0). See "Defining a First-Time Program" in *Chapter 22 - System Programming* for additional information.

LANGUAGE(langname)
This specifies the national language for messages, etc. Not all applications support all languages. If an application does not support the language you request, it uses English. National language names are: English, French, Kanji (Japanese), Portuguese, Spanish, German. Other language names MAY also be accepted. Most language names have a 2 or 3-character abbreviation, for example LANG(ENG). To remove the LANGUAGE option, specify LANGUAGE() or LANGUAGE(DEFAULT), both of which use the site-dependent default language.

ROUTE(name)
A default route name (1 to 8 characters) to be used by programs such as SUBMIT. To set no default route name in the userid record, use ROUTE(), which sets the name to all blanks.

TERMINAL(name) TERM
Defines a default terminal type, which is a name from 1 to 8 characters long. Terminal type names are defined in the system module TRMCTL, and identify various specific terminal models or classes of terminals. Examples are TERM(3270), TERM(TTY) and TERM(DECII).

Some userid options, such as BACKSPACE and OTABS, are defined for a specific terminal type. At sign-on time, MUSIC will check the type of terminal used against the one defined with this option. Only if they match will the options be set as defined in the userid record. This prevents, for example, the tab settings for a 2741 terminal from being used when the user signs on using an ASCII (TTY) type terminal.

The TERMINAL parameter can also used to define a terminal subclass. to enable the system to distinguish between a 72-character TTY terminal and a 132-character TTY terminal.

Refer to the *Chapter 7 - Terminal Configuration and Tailoring* and *MUSIC/SP User's Reference Guide* for more information.

The terminal type is removed from the userid record by TERMINAL().

ITABS(n1 n2 n3 ...)
Defines the default input tab positions. From 1 to 80 column numbers (separated by blanks or commas) can be specified. Each is a number from 1 to 80. Input tabs make it easier to enter data in

specific columns, by pressing a tab key or a logical tab character to space to the next tab position in the input line. The TAB parameter can be used to define a logical tab character. To remove all input tabs, use ITABS().

OTABS(n1 n2 n3 ...)
Defines the default output tab positions. From 1 to 11 column numbers (separated by blanks or commas) can be specified. Each is a number from 1 to 254. MUSIC uses these tab settings to speed up output to the terminal, provided the user's terminal matches the terminal type specified in the TERMINAL parameter. It is up to the user to set the physical tabs correctly. To remove all output tabs, use OTABS().

TAB(x)
Defines an input character that MUSIC is to interpret as a tab operation (i.e. space to the next tab position in the input line). *x* is a single character, or the 2-digit hexadecimal value of a character. Any character other than a letter, digit, slash, blank, equal sign, or comma can be used. For example: TAB(&), TAB(35), TAB(E0). To undefine the tab character, use TAB(), which is equivalent to TAB(00).

BACKSPACE(x) BS BACKSP
Defines an input character that MUSIC is to interpret as a backspace operation (i.e. delete the preceding character). *x* is specified as in the TAB parameter (see above).

IDLE(x)
Defines a character that MUSIC is to use as the idle character for output to the terminal. *x* is specified as in the TAB parameter (see above).

HEX(x y)
Defines an input replacement character. When MUSIC sees the character *x* in a terminal input line, it will replace it with *y*. *x* and *y* are specified as in the TAB parameter (see above) and are separated by a blank or commas. *y* may be any of the 256 possible characters. Example: HEX(&,02) means that "&" will be interpreted as 02 (hex). To remove this option, enter HEX().

MAX$(n)
This sets the limit for the number of dollars of computing funds that the userid is allowed to use. The number may be any integer 0 or more, or NOLIMIT.

When a new userid record is created, its funds usage number (the NOW$ field) is set to 0. Each time the NOWDOL accounting program is run, the userid's usage charges since the previous NOWDOL run are added to the NOW$ field. When the NOW$ field reaches or exceeds the MAX$ limit, the user is not allowed to sign on or run batch jobs, until a higher MAX$ limit is set or the ADD$ parameter is used to increase the limit. A supervisor userid can also run the TRANS$ program to change the MAX$ limit.

The MAX$ and NOW$ fields can be displayed by the SHOW$ option of the GET command. NOW$ is shown as the amount of funds *used* and MAX$ as the *limit*. Note that, internally, the MAX$ field of the userid record is in dollars, while the NOW$ field is in cents. The values are always displayed as $n.nn amounts by SHOW$.

ADD$(n)
Adds *n* dollars to the userid's MAX$ limit. The number *n* may not exceed 1 million. To set unlimited funds, specify MAX$(NOLIMIT).

NOW$(n)
(This parameter is normally not used, since only the NOWDOL and TRANS$ programs are supposed to update the NOW$ field.) This sets the userid's funds usage number (the accumulated usage charge) to a specific value. The value can be specified as a number of dollars, or as a

dollar.cents amount. For example: NOW$(55), NOW$(129.17).

TRK/UDS(n)

Gives the maximum size, in tracks, for new UDS files created by the user. This limit does not apply to temporary (&&TEMP) UDS files. The number may be from 0 to 32000, or NOLIMIT. A limit of zero prevents the user from creating any permanent UDS files.

TOTLIM(n)

Specifies the total space limit in the user's UCR (User Control Record). This is the limit, in units of 1K = 1024 bytes, on the total space of all the user's files, not counting temporary files. TOTLIM(NOLIMIT) gives unlimited space.

This parameter can be used only on an ADD command. If a UCR already exists for the userid, the UCR value is set to the larger of the existing value and *n*. To change the limit after the userid is created, use the UCR program.

FILLIM(n)

Specifies the individual file limit in the user's UCR (User Control Record). This is the maximum amount of space, in units of 1K = 1024 bytes, that can be used for any one file in the user's Save Library, including temporary files. FILLIM(NOLIMIT) allows unlimited size files, up to the system-wide limit per file (normally 4000K).

It is recommended that FILLIM never be set less than 300K, since some of the compilers require temporary files up to this size.

This parameter can be used only on an ADD command. If a UCR already exists for the userid, the UCR value is set to the larger of the existing value and *n*. To change the limit after the userid is created, use the UCR program.

XSESSIONS(n) XSES

Defines the maximum number of extra sessions per terminal. The value *n* must be 0 to 250, or NOLIM. The default on an ADD command is 3.

START(yyyy/mm/dd)

Gives the starting date (year/month/day) for the userid. The user will not be able to sign on or run batch jobs before this date, even though the userid record exists. *yyyy*, *mm*, and *dd* must be 4, 2, and 2 digits respectively. For example, START(1985/07/31). To remove a starting date, specify START().

EXPIRY(yyyy/mm/dd) EXP

Gives the expiry date (year/month/day) for the userid. The user will not be able to sign on or run batch jobs after this date, even though the userid record still exists. *yyyy*, *mm*, and *dd* must be 4, 2, and 2 digits respectively. For example, EXPIRY(1985/12/25). To remove an expiry date, specify EXPIRY().

NAME(string)

The user's name, up to 48 characters. The character string is stored as is, without any processing or checking, and is optional. It may contain blanks and special characters. To set the field to all blanks, use NAME().

ID(string)

A user identification or account number, up to 16 characters. It could be used to store the user's telephone number, for example. The character string is stored as is, without any processing or checking, and is optional. It may contain blanks and special characters. To set the field to all blanks, use ID().

TAG(string)

A descriptive field, up to 64 characters, that could be used indicate the purpose or origin of the userid. The character string is stored as is, without any processing or checking, and is optional. It may contain blanks and special characters. To set the field to all blanks, use TAG().

LIKE(userid)

This parameter is valid on an ADD command only. It specifies an existing userid whose Code Table record is to be used as a model for the userid being created. All fields (except for those listed below) are copied from the existing record, then any overriding values specified on the ADD command are filled in. UCR limits are also copied from the model userid's UCR.

The following userid record fields are not copied from the model record: userid, date of creation (set to the current date), NOW$ (set to zero), date and time of last sign-on and last batch job (set to zero), and dates of last password changes (set to zero).

Examples:

```
ADD ABCD LIKE(EFGH) PW(ANEWPW)
ADD XXAA002 LIKE(XXAA001) TAG(MY 2ND SUBCODE)
```

COMMENT(string) CMT

The character string is a comment field on the command, and is not stored in the userid record. The string may be of any length.

SHOW$ PRINT$ $

This option on the GET command displays the userid's dollar amounts (the MAX$ and NOW$ fields). NOW$ is shown as the amount of funds *used* and MAX$ as the *limit*.

SHOWSPACE SHOWSP PRINTSPACE PRINTSP

This option on the GET command displays the values from the userid's UCR: the current total Save Library space, the limit on the total space (TOTLIM), and the limit per file (FILLIM).

BRIEF BR

This option on the GET command displays the user's ID and NAME fields.

DUMP

This option on the GET command displays the userid record in the form of a hexadecimal dump. The block numbers of the code and index records are also displayed. These are the block numbers that could be used in the SYSDMP utility program to access these records in the code and index data sets.

DELUCR

This option on the DELETE command deletes the UCR (User Control Record), even if other subcodes still exist. This option should not be used in most cases.

DELMAILBOX

This option on the DELETE command also deletes the user's mailbox file.

Default Settings for New Userid Records

Figure 17.2 shows the default options for an ADD command (when the LIKE parameter is not used). Any options specified on the command override these.

```
PASSWORD(MUSIC) BATCHPW(same as terminal password)
sign-on options: NOSNGL NOFIXPW CANA NOUNAME
    ENABLE NORESTR
operating defaults: INPROG NOSLASH NOEXEC NOEDMSG
    NOPRIVONLY COM NOFIXALPROG NOJA1 NOJA2 ... NOJA8
no privileges
PRIME(180) NONPRIME(180) DEFTIME(60) BATCH(180)
AUTOPROG() ALWAYSPROG() FIRST(0) ROUTE()
TERMINAL() ITABS(7 73) OTABS() TAB() BACKSPACE()
IDLE() HEX() MAX$(NOLIMIT) NOW$(0) TRKS/UDS(NOLIMIT)
TOTLIM(10000) FILLIM(NOLIMIT) START() EXPIRY()
NAME() ID() TAG() XSESSIONS(5) TYPE(0) PWEXP(0)
```

Figure 17.2 - New Userid Record Default Settings

An installation may change the above defaults by modifying the subroutine DEFLT in the CODUPD program. For example, coding could be added to set different defaults for different classes of userids.

Code and Subcode Ranges

A group of several code records can be referred to in a single CODUPD command by specifying a range of codes and/or subcodes. The ADD, CHANGE, GET or DELETE operation is applied to each code record in the range.

A range of codes is specified as AABB-CC, where all the codes start with AA (1 or more characters), the first code is AABB, and the last code is AACC. The sequence of codes can be numeric (each code ending in a 2-digit or 3-digit number) or alphabetic (each code ending in a 2-digit base-36 *number* whose *digits* are chosen from A, B, ..., Z, 0, 1, ...,9).

An example of a numeric code range is XY05-20, which refers to codes XY05, XY06, ..., XY20. With this method, up to 100 codes can be referred to per command, or up to 1000 codes if 3 ending digits are used.

An example of an alphabetic code range is XYAA-CF, which specifies the 78 (=2*36+6) codes XYAA, XYAB, ..., XYAZ, XYA0, XYA1, ..., XYA9, XYBA, XYBB, ..., XYBZ, XYB0, ..., XYB9, XYCA, XYCB, ..., XYCF. The lowest starting suffix is AA and the highest ending suffix allowed is 99, which gives 1296 (=36*36) combinations.

A subcode range must be numeric, using subcodes 000 to 999, and is specified as SUBCODE(nnn-mmm) For example, SUBCODE(010-135) refers to the subcodes 010, 011, 012, ..., 135.

The automatic code record display for an ADD, CHANGE or DELETE (see the .PRINT command above) is always omitted when a range of codes or subcodes is specified.

Examples:

```
ADD TS00-25 PASSWORD(GREEN)

CHANGE RHMM SC(000-015) PRIME(600)

GET XXBA-D9 SHOW$

DELETE PQMA-M9 SUBCODE(MUS)
```

Limited Access to the Code Table by CODXXX

Only one user at a time is allowed to run CODUPD to update the Code Table. This can be a nuisance if the code administrator is in the middle of a long CODUPD run and some other system user wants to display a code record and, for example, wants to make a minor change to it.

For this reason, a special version of CODUPD is available, called CODXXX, which allows concurrent access to the code table but does not allow ADD or DELETE commands. GET and CHANGE commands are allowed in CODXXX, but records cannot be added to or removed from the Code Table. CODXXX reads commands conversationally from the terminal.

Sample CODUPD Session

The following sample session shows a code being created, changed, displayed, and deleted. The lines entered by the user are preceded by "?" (MUSIC's conversational read prompt).

```
codupd
*In progress

CODE TABLE UPDATE  --  TUE DEC 18, 1984  16:19
?
add abcd sc(001) pw(mypassw)
  USERID=ABCD001         FILE OWNERSHIP ID=ABCD        TYPE=0
  ID=                   NAME=
  PASSWORD=MYPASSW    BATCH PASSWORD=MYPASSW
  NO PRIVILEGES
  TIME LIMITS (IN SERVICE UNITS):
    PRIME=180  NONPRIME=180  BATCH=180  DEFAULT=60
  MAX NUMBER OF EXTRA SESSIONS PER TERMINAL:    5
  PASSWORD CAN BE CHANGED BY USER
  AUTOPROG:  (NONE)
  INPUT TABS:     7  73
  NO OUTPUT TABS
  FUNDS ($):        0.00 USED,  NO LIMIT
  SAVE LIBRARY:  TOTAL =     0K   LIMIT = 10000K   MAX/FILE =  4000K
  MAX TRACKS PER DATA SET (UDS) AT ALLOCATION:  NOLIMIT
  CREATED 1984/12/18  (YEAR/MONTH/DAY)
  LAST SIGN-ON: 0                  LAST BATCH JOB: 0
  LAST PASSWORD CHANGE:  TERMINAL PW 0            BATCH PW 0
  PASSWORD LIFETIME:  NO LIMIT
  USERID OF CREATOR:  $000000

?
.noprint
?
c abcd001 autoprog(check.test) itabs(10 20 30) -
ENTER CONTINUATION LINE
?
name(J. Smith) id(123-456-789)
?
g abcd001
  USERID=ABCD001         FILE OWNERSHIP ID=ABCD        TYPE(0)
  ID=123-456-789       NAME=J. Smith
  PASSWORD=MYPASSW    BATCH PASSWORD=MYPASSW
  NO PRIVILEGES
```

```
  TIME LIMITS (IN SERVICE UNITS):
    PRIME=180  NONPRIME=180  BATCH=180  DEFAULT=60
  MAX NUMBER OF EXTRA SESSIONS PER TERMINAL:    5
  PASSWORD CAN BE CHANGED BY USER
  AUTOPROG:  CHECK.TEST               (CANCELLABLE)
  INPUT TABS:    10  20  30
  NO OUTPUT TABS
  FUNDS ($):        0.00 USED,  NO LIMIT
  SAVE LIBRARY:  TOTAL =     0K   LIMIT = 10000K   MAX/FILE =  4000K
  MAX TRACKS PER DATA SET (UDS) AT ALLOCATION:  NOLIMIT
  CREATED 1984/12/18  (YEAR/MONTH/DAY)
  LAST SIGN-ON: 0                  LAST BATCH JOB: 0
  LAST PASSWORD CHANGE:  TERMINAL PW 0           BATCH PW 0
  PASSWORD LIFETIME:  NO LIMIT
  USERID OF CREATOR:  $000000

?
g abcd001 brief $ showsp
  ABCD001   123-456-789       J. SMITH
  FUNDS ($):        0.00 USED,  NO LIMIT
  SAVE LIBRARY:  TOTAL =     0K   LIMIT = 10000K   MAX/FILE =  4000K
?
del abcd001
?
get abcd001
USERID ABCD001 NOT FOUND
?
end
*End
*Go
```

CONLOG: Incore Console Log Utility

This utility displays the current contents of the console log that is retained in memory. If you ar on a 3270-compatible terminal, you are placed into BROWSE to view the CONLOG output. An optional parameter can be specified to limit the list to a fixed number of entries. Only the most recent entries are listed. The utility always lists console entries in chronological order.

Several lines of addresses are printed before the console log listing. These addresses occupy the 20 bytes prior to the start of the console log. On each log entry, a sequence number, the start address of the entry, its length, and the message itself are printed. (See also CONSOLE program in *Chapter 4 - The System Console*.)

Usage:

From command mode:

```
CONLOG  nn  NOFS
```

Where *nn* is the number of lines to display and *NOFS* forces non-full-screen mode of operation.

Privileges Required: LSCAN and CREAD.

COUNTS: List System Counters

The program COUNTS dumps the system statistics counters. These are counts maintained by various system routines. The meanings of the individual counters can be found by listing the file XCOUNTS.

Usage:

From a terminal this program is run by means of the command COUNTS.

Note: These counters are reset at MUSIC IPL time (and only then).

Privileges Required: CREAD.

DSACT1, DSACT2: User Data Set Accounting

This utility is used in conjunction with the ACTDMP utility to provide billing information for User Data Set (UDS) files. Typically this utility is run weekly to produce input to the installation's billing program.

The records produced by this utility give the size and length of time each data set has been allocated. If this data set was allocated before the utility was last run, then the time reflects the amount of elapsed time since the last run of this utility. The ACTDMP utility produces records in this same format when a UDS file is deleted. Together these utilities maintain records of the usage of UDS files. Consult the description of the ACTDMP Utility for the output record format.

(Temporary data sets allocated without a data set name are not charged. These temporary data sets come out of a pool of system space in the data set SYS1.MUSIC.SCRATCH. There is a limit to the amount of space any user can get from this pool. These temporary data sets are only used during the duration of a single job. See the *MUSIC/SP User's Reference Guide* for information regarding the use and size limitations of these data sets.)

The DSACT utility is run in two parts. The first part (DSACT1) scans the UDS volumes and writes the data set names to a file. The second part (DSACT2) uses the information produced by the first step to update the date last billed on the VTOCs of the UDS packs and also produces the accounting records. The format of these accounting records (MD1) is given in the description of the ACTDMP utility.

This utility is separated into two parts for the sole purpose of allowing either part to be rerun if necessary. The second part DSACT2 must not be run until the first part has completed successfully. Although it is unlikely that either step will fail, this procedure does handle the case.

When these utilities are run for the first time, the installation must allocate a file with a LRECL of 65, RECFM(F), and number of records (NREC) greater than the number of data sets on all the UDS volumes. Make this data set private to protect systems security. When DSACT1 is run for this first time, you must specify the NOCHEK=T option.

When DSACT2 is run for the first time, the message `INVALID NUMBER OF MINUTES` may be printed.

To run the DSACT1 section use the following control statements. The /FILE statement must point to the file you allocated to contain the data set names. The logical record length of this file is 65. The record format must be F. The volume names of the UDS packs are given on the option statement. (DSACT1 will check to ensure that at least 12 hours have elapsed since it was last successfully run. This checking can be suppressed

by using the option NOCHEK=T. Normally you will not use this NOCHEK option.)

```
/FILE 1 NAME(namesfile) OLD NORLSE
/INCLUDE DSACT1
VOL='udsnm1','udsnm2',....
```

To run DSACT2 use the following:

```
/FILE 1 NAME(namesfile) OLD NORLSE
/FILE 2 NAME(output) NEW(REPL) SP(100)
/INCLUDE DSACT2
OUT=2
```

The OUT=n parameter specifies the I/O unit number where the accounting records are to be placed. Do not use n=1. n=7 can be used to punch the records. An option of PRINT=F can be given to suppress the printing of the data set information.

VERTAP=T can be used to check that the correct output tape is mounted.

If DSACT2 must be rerun, then the option of RERUN=T may be required. Thus the option statement in this case will read OUT=2,RERUN=T.

Privileges Required: VIP.

DSARCH: Data Set Archive

This program will copy to tape (or to sequential disk files), a data set (or sets). It will produce two identical copies simultaneously if desired. The dump is in the format of 80-byte card images. The dump of each data set is preceded by two header statements (starting with '*DS1' and '*DS2') and is followed by a trailer statement ('*DSEND'). For the contents of the *DS1 and *DS2 records, see the comments in file $PGM:DSDMP.S or refer to the description of the UDSARC utility in the *MUSIC/SP User's Reference Guide*. Check sums are maintained to guard against undetected I/O errors.

Usage:

```
/FILE n TAPE VOL(xxxxxx) OLD LR(80) BLK(nnnnn) RECFM(U)
/INCLUDE DSARCH
UNITS=n,m
dump parameters
```

Either one or two /FILE statements should be supplied depending on whether two copies of the dump are required. The output files should have a record size of 80 bytes.

The UNITS= keyword is followed by either one or two unit numbers corresponding to the /FILE statements. If two are used, the numbers are separated by a comma.

Each dump command following the UNITS= will dump one or more data sets. Each must contain a VOL= parameter and one of either DSN= or CODES=

Parameters:

VOL='volume','volume',...or VOL='/ALL/'
where 'volume' is a UDS volume name and '/ALL/' means all UDS volumes. (Abbreviation

V=)

DSN='name' (abbreviation D=) where 'name' is one of the following types -

'dsname'	actual 1 to 44 character data set name eg. use 'sys1.music.acct' not '%acct'
'pref..'	refers to all data set names starting with the given 1 to 42 character prefix
'/ALL/'	all data sets on the volume(s)
'/BACKUP/'	all data sets for which BACKUP was requested on the /FILE statement at allocation
'/NOBACK/'	all data sets for which BACKUP was not requested on the /FILE statement at allocation

Note: Only one name may be specified per NAMELIST. The total number of data set names and codes specified must not exceed 200/n, where *n* is the number of volumes.

CODES='code','code'...
(abbreviation C=) Where 'code' is a 4 character user code. All data sets belonging to the code(s) will be dumped from the specified volume(s). Up to 20 codes may be specified. However, if several volume names are given, then *n***m* must not exceed 200, where *n* is the number of volumes and *m* is the number of codes.

VERTAP=T Verifies that first record of existing tape file begins *DSARCHIVE to ensure that a data set archive tape is mounted. The default is VERTAP=F.

Examples:

Dump nucleus data set on MUSICX:

```
VOL='MUSICX',DSN='SYS1.MUSIC.NUCLEUS'
```

Dump all UDS belonging to $L06 on all mounted volumes:

```
VOL='/ALL/',CODES='$L06'
```

Dump all data sets from MUSICZ if name starts with SYS:

```
V='MUSICZ',D='SYS..'
```

Privileges Required: LSCAN; DREAD or MAINT.

DSCHK: Dump Tape Verify

The DSCHK program scans a dump tape or tapes and verifies that the format is correct, check sums tally and if two tapes are input, that they are identical.

```
/FILE    one or two file statements, RECFM(u) for tape only
/INCLUDE DSCHK
UNITS=n,m,NAMES=TRUE,INFO=TRUE
                 FALSE     FALSE
```

In the parameter list, *n* and *m* are the unit numbers of the /FILE statements pointing to the dumps. If NAMES=TRUE is specified, the names and volumes of all data sets on the tape(s) are listed. If INFO=TRUE is used all data set header statements are printed.

Privileges Required: None.

DSCOPY: Data Set Copy Utility

The DSCOPY utility program described in the *MUSIC/SP User's Reference Guide* may also be used to copy system data sets. The following is a sample of how it can be used. First enter /VIP ON, then execute a file containing the following:

```
/FILE 1 UDS(%name1) VOL(volume1) SHR
/FILE 2 UDS(%name2) VOL(volume2) OLD
/INCLUDE DSCOPY
```

The above example will copy from SYS1.MUSIC.name1 to SYS1.MUSIC.name2. Both data sets must have the same blocksize and the receiving data set must have been allocated and formatted before the copy is done.

The DSCOPY program cannot be used for copying files. Use the /COPY command for this. Another technique for creating a copy of a file is to archive it to a file or UDS file (MFARC2 program), then restore to the target file (MFREST program). The editor may also be used for copying most files, but not for files with record format U or load modules with record format FC or VC.

Privileges Required:

The user must have the VIP privilege active at the time the job is run.

DSKDMP: General Disk Utility

This program enables the system programmer to inspect and change records on direct access devices by absolute disk locations. It includes functions for inspecting Volume Table of Contents (VTOC) entries of disk packs. (Refer to the SYSDMP utility if you want to inspect and alter contents of a SDS or UDS data set by relative block number.)

The program is normally run from a terminal, but can be used at batch. The command to execute it from a remote terminal is:

```
DSKDMP
```

The primary keywords accepted by DSKDMP are:

CYL=x,y	Specifies the range of cylinders to be dumped. If the second one is omitted, it is assumed to be equal to the first. The default values are both zero. This parameter is not valid on FBA volumes.
HEAD=x,y	Range of heads (tracks) to be dumped. If the second one is omitted, it is assumed to be equal to the first. The default head values are zero. This parameter is not valid on FBA volumes.
RECORD=x,y	Range of record numbers to be dumped. If the second is omitted, it is assumed to be equal to the first. The default is one. On FBA volumes, this parameter gives the absolute block numbers For type 3 and 5 requests, the

record number must be one less than the read record number.

LEN=	Specifies the number of bytes to be read or written. The default is 16. Note, if the length written is less than the physical record size on disk, then the remainder of the record will be filled with zeros. This zero fill is automatically done by the hardware.
UNIT=	Specifies the internal MUSIC unit control block or Data Extent Block on which the I/O is to be done.
ADDR='cuu'	Specifies the physical device address on which the I/O is to be performed.
VOL='xxxxxx'	Specifies the logical volume name onto which the I/O is to be done. The names which can be used are detailed below.
TYPE=	Specifies the type of operation to be performed. The values of TYPE are given below. On FBA volumes, only TYPE=1 is valid.
ECHO=TRUE or FALSE	Specifies whether the above parameters are to be printed prior to printing the dump information.
REP=	Gives the displacement into the record at which some data is to be changed. The changed bytes are entered next. The record must have been previously read by means of a CYL, HEAD, RECORD read request.
WRITE=TRUE or FALSE	Set to TRUE when it is desired to rewrite the current buffer. TYPE must be set to either 1 or 2. The buffer must have been filled by a previous read request. All other parameters are ignored if WRITE=TRUE is given. Refer to the NORD parameter for usage notes.
NORD=TRUE or FALSE	Set to TRUE if no actual disk reading is to take place. This option can be used to alter the disk location for a subsequent write operation.
VTOC=TRUE or FALSE	Record is assumed to be part of a disk pack VTOC (Volume Table Of Contents) will be displayed appropriately (and if it 'looks' like a VTOC record). This command forces LEN=140,TYPE=7. VTOC=TRUE stays in effect until it is set FALSE. This parameter is not valid on FBA volumes.
LISTV=TRUE or FALSE	This command causes a VTOC listing (sorted by track address) to be produced. It checks for overlapping data sets and gaps (lost space). A GAP message is normal for a volume that is used by DOS or DOS/VS. A UNIT, ADDR, or VOL specification should accompany this parameter.
LISTDS=TRUE or FALSE	Similar to LISTV but extents are sorted by data set name. Any gaps or overlaps are ignored.
FREE=TRUE or FALSE	Similar to LISTV but only free space is listed.
ZAPF5=TRUE or FALSE	All free space information records (ie. format 5 DSCBs) are cleared from the VTOC. The VTOC will indicate that no free space is left on the pack at the end of this operation. This command is usually used before a FORMF5 command. A UNIT, ADDR, or VOL specification should accompany this parameter. This parameter is not valid on FBA volumes.
FORMF5=TRUE or FALSE	New format 5 DSCBs (free space information) are created. The ZAPF5 operation must have been performed previously to clear the old format 5

	entries. A UNIT, ADDR, or VOL specifications should accompany this parameter. You should perform a LISTV function to verify that the number of data sets on the pack will not exceed the table size in this program before doing the FORMF5 operation. This parameter is not valid on FBA volumes.
DSN='...'	This command causes the VTOC entry for the specified data set name to be printed.
END	This command terminates DSKDMP.
HELP	This command displays information on how to use the program.

Several abbreviations and alternative keywords are allowed.

Keyword	Alternatives
CYL	CC, C
HEAD	HD, HH, H
RECORD	RCD, REC, RR, R
LEN	L, COUNT, CNT
UNIT	U, UCB
ADDR	UAD, ADR, A
VOL	VOLUME, V
TYPE	none
ECHO	none
REP	none
WRITE	WRT
VTOC	none
LISTV	none
LISTDS	none
FREE	none
ZAPF5	none
FORMF5	none
DSN	none
NORD	none
END	none
HELP	none

Device Specification

The disk drive to be used by this utility is specified by one of three parameters: UNIT, ADDR, VOLUME. If more than one of these is given, a check is made to ensure that the parameters all indicate the same device. An error message is issued if they conflict.

VOLUME Names

The names to be specified for MUSIC system volumes are as follows:

SYSRES	System residence device
zzzzzz	Disk volume zzzzzz.

TYPE Specification

Value	Meaning
1	Read/write data
2	Read/write key, data
3	Read count, key, data
4	Read home address
5	Read count
6	Read record R0
7	Read VTOC record (See VTOC parameter)

Notes:

1. Only TYPE 1 or 2 may be used if the record is to be rewritten. Counts, home addresses, and record R0 cannot be rewritten with this program.

2. If TYPE 3 or 5 is used, the record number requested must be one less than the real record number.

3. The default parameters at the start of program execution are:

```
CYL=0,HEAD=0,RECORD=0,LEN=0,VOL='SYSRES',TYPE=1,ECHO=FALSE
```

4. Parameters are entered in standard MUSIC FORTRAN NAMELIST form. No blanks should be used and parameters are separated by commas.

5. With the exception of WRITE and REP, all parameters retain their values unless explicitly changed. WRITE and REP must be explicitly stated each time they are needed.

6. If the UNIT= parameter is used to input a DEB number, or if VOL= LIBEnn is used, the cylinder, head and record specified will be translated to a real disk address according to the position of the data set on the real drive and the pseudo device format of the data set. If the DEB number specified has no pseudo device format, the request will be rejected.

7. To copy data from one record to another, first read the record with a VOL, CYL, HEAD, RCD, LEN (or equivalent) request. Then issue a similar request for the address to which the data is to be written, not specifying LEN, but specifying NORD=TRUE. Then use the WRT=TRUE command to write the buffer to the new location.

REP Data Format

Up to one card image of data can be specified for each REP command. More than one REP command may be given before the record is rewritten. The data must start in column one of the line. It is entered in hexadecimal. Commas may be inserted at will between bytes, but not between hexadecimal digits of a byte. A blank (or end of card) terminates the data.

Sample Execution

A sample run of this program illustrating most of its features is given in Figure 17.3. The first operation requested is to print the data portion of the record located at cylinder 99, head 0, record 1. Several bytes starting at byte 14 of the record are replaced. The record is then written back to the device.

If a line is to be printed in which all bytes are the same, only one copy of the byte is shown.

Note that changes to disk records should be made with extreme care.

The next request prints records 1, 2 and 3 of the track located at cylinder 0, head 0 of the system residence volume. These records are the IPL record, the IPL program, and the volume label. The program is requested to read and print 32 bytes of each record including the count and key fields.

Note that the actual length of the disk blocks can be determined by this technique. It is shown in the second word of each record. In the above example they were x'18', x'AA8' and x'50'.

Privileges Required:

The DREAD and CREAD privileges are needed to run this program. The VIP privilege must be active in order to use the WRT, ZAPF5, or FORMF5 commands.

```
*Go
dskdmp
*In Progress

DEFAULTS
VOLUME = SYSRES
TYPE = 1   READ DATA
DISK UCB = 0
CC =   0,   0     HH =  0,  0     R =  1,  1     LEN =   16
?
vol='disk01',cc=99,len=262
CCHHR= 99  0  1
0000  D3C2F0F1 F0F9F1F5 F6F7FFB5 BE191201  *LB01091567......*
0010  00  *.*
/skip 20
?
rep=14
ENTER REP
?
16,00,AB89
0000  D3C2F0F1 F0F9F1F5 F6F7FFB5 BE191600  *LB01091567......*
0010  AB890000 00000000 00000000 00000000  *................*
0020  00  *.*
0030  00  *.*
0040  00
/sk 20
?
write=true
RE-WRITTEN   99   0   1  DATA
?
type=3,vol='sysres',r=0,2,cyl=0,len=32,echo=true
VOLUME = SYSRES
TYPE = 3   READ C K D
DISK UCB =  0     HH =  0,  0     R =  0,  2     LEN =   32
CCHHR=  0  0  0
0000  00000000 01000018 00000000 00001000  *................*
0010  06001000 20000AA8 00000000 00000000  *.......y........*

CCHHR=  0  0  1
0000  00000000 02000AA8 05F0D207 0070F396  *.......y.0K...3o*
0010  8200F39E 48100002 4130F114 D204F140  *b.3.......1.K.1 *

CCHHR=  0  0  2
0000  00000000 03000050 E5D6D3F1 D4E4E2C9  *.......&VOL1MUSI*
0010  C3E7F000 00000101 40404040 40404040  *CX0.....        *

?
end
*Terminated
*Go
```

Figure 17.3 - Sample Run of DSKDMP

DSRST: Data Set Restore

This restore program will restore the contents of a data set from a dump tape created by the data set archive program.

Usage:

```
/FILE n ... (dump tape, lrecl 80, RECFM(u) for tape)
/INCLUDE DSRST
IN=n,TAPFIL=m,VOL='vol',DSN='dsn',
TOVOL='tovol',TODSN='todsn'
```

Where *n* is the unit number of the /FILE statement, *m* is the tape file number (default m=1), *vol* is the volume label where the data set originally resided and *dsn* is its original name. The data set on which the data is to be restored is pointed to by TOVOL and TODSN. The TODSN must already exist though no /FILE statement is required to point to it. All parameters must be supplied with the exception of the input unit number which defaults to 1.

More than one data set can be restored in the same job by supplying additional sets of NAMELIST parameters. If IN=5, only one data set can be restored. Only the DSN and TODSN parameters must be entered for each subsequent request. The VOL, DSN, TOVOL and TODSN may be abbreviated as V, D, TOV, TOD respectively.

Note: Archive and restore jobs are normally run from batch, although if a disk data set is used instead of tape, they may be run from a terminal.

Contents of Information Records

In a dump produced by UDSARC, each data set is preceded by a *DS1 record and a *DS2 record. The *DS1 record has the volume name, the data set name, and the date and time of the dump. The *DS2 record contains the following information:

Column 8	B for backup, N for no backup.
9-12	Device type.
13-15	Data set organization.
16-19	Record format.
20-25	Block size.
26-31	Logical record length.
32-37	No. of tracks (no. of physical blocks if FBA device).
38-40	No. of extents.
41-44	No. of blocks per track (32 if FBA device).
45-48	Key length.
49-53	FBA physical block length (0 if not FBA device).

Privileges Required: LSCAN, VIP.

EDTCAT: System Catalog Creation Utility

This utility program creates a system catalog on disk. It also produces a listing of the new catalog. If run from batch the job format is:

```
/INCLUDE *COM:EDTCAT
/INCLUDE xxxxxx
```

where *xxxxxx* is the name of a file containing the catalog statements. The file $GEN:SYSCAT contains the statements used to create the catalog on the production system created at the time MUSIC was installed. If run from a terminal, a /INPUT and /ENDRUN replace the /ID and /END respectively. When being run from a terminal, the program pauses after the displaying information concerning the catalog location, for permission to proceed. If it is desired **not** to continue, a /CANCEL command must be entered. Any other response will allow catalog to be re-written.

In all cases, the first two lines of the catalog input must be:

```
'CATALOG' 'volume'
 data set name
```

The word 'CATALOG' (in quotations) must be the first parameter of the first line. The next parameter is the volume label of the pack on which the catalog is to reside (also in quotations). Columns 1-44 of the next line contain name of the catalog. This is normally SYS1.MUSIC.CATALOG.

The data set must have previously been allocated and formatted.

System Catalog

The system catalog contains the following types of records:

- System Data Set statements
- Pseudo Device statements
- Operator Command statements
- Volume Mount Request statements
- Link Pack Routine Names

System Data Set Statements

Column	Description
1-8	Unique 1-8 byte identification used during catalog console editing
10-13	Data set identification U000 for the Save Library Index Unnn for Save Library spaces Bnnn for batch spooling Fnnn for others (here nnn will be the DEB number, for U and B SDS, the DEB number is assigned at initialization time).
15-16	Pseudo-device type (same as column 7-8 of pseudo cards). If set to 00, no pseudo-device requests may be made.
18	=R if data set must be found for proper system operation. =M if system is to attempt to run without this data set, but a warning message should be issued =I same as M but no message is issued.
19	=L when data set is located, issue an informative message =N no message

20	=S keep separate statistics for this SDS =N keep statistics under the physical device.
22-24	Number of records per track for this data set. If this is used, it overrides the normal value calculated from the device characteristics and the block size. Zero indicates to use the normal value.
26-31	Volume serial number on which the SDS is to be found. If blanks are specified, a search for the SDS will made on every ready device.
33-76	Data set name

Pseudo Device Statements

Column	Description
1-6	PSEUDO
7-8	Pseudo device type referenced by column 15-16 of data set statements.
10-12	Flag bytes - must be zero
14-18	Blocks per track on pseudo device
20-24	Tracks per cylinder on pseudo device.

Operator Command Statement

Column	Description
1-5	OPCMD
6-8	Unique three digit number to distinguish one OPCMD statement from another.
10-12	/CP
14-80	VM command to be issued. Sample command might be: OPCMD001 /CP SET RUN ON

Volume Identification Statements

Column	Description
1-6	VOLUME
7-8	Unique two digit number to distinguish one VOLUME statement from another.
10-15	Volume label of the disk pack. A message may be issued if the volume is not mounted at IPL time, depending on the contents of column 18. This statement can be used to ensure that the proper UDS packs are present. Note that it is **required** to identify the UDS packs in this manner in order that users may create new files on the pack.
17	The letter A in this column means that users can allocate new files on this volume. The

letter P means that non-privileged users may not refer to any UDS file on this volume. The letter N means that users can refer to UDS files on this volume, but they cannot allocate new files on it.

18 This column contains a letter code specifying whether a message is to be sent to the operator at IPL time if it is not mounted. The allowed values are R, M and I. See the description of column 18 for System Data Set statements for meaning of the letters.

19 The letter R in this column means that MUSIC will not attempt to write to this disk during normal processing of user jobs. Privileged users could deliberately write to this disk via utility programs such as DSKDMP.

Link Pack Area Names

Column	Description
1-5	RESPG
6-8	Unique 3-character id to distinguish one RESPG statement from another.
10-17	Eight character module name to place in the Link Pack Area. Refer to *Chapter 21 - Load Library and Link Pack Area* for details of usage of this area.
19-22	The letters FLPA or PLPA. FLPA means that main storage will be allocated to hold the specified module. PLPA means that the re-entrant module named will be written on the first page data set in page format. FLPA is assumed if this field is blank.
24-30	Virtual storage location (in hex) of where the PLPA member is to be loaded. This field is ignored for FLPA items. The storage location must be at least 300000. The location must be chosen so that the module ends before location 700000. Members that can be simultaneously used by the same program must not overlap their virtual addresses.

In the catalog statements you can use special characters in the volume name fields to allow for a more flexible catalog. The character ? in any position of the volume name will be taken to mean that it will match any character. The character $ in any position in the volume name will match the corresponding character in the volume name of the IPL volume.

Privileges Required:

The VIP privilege is needed to run EDTCAT. Refer to the topic "VIP Privilege Notes" in the CODUPD writeup for details of how to activate this privilege.

ELOG.CLEANUP: Delete Empty Editor Log Files

This utility deletes any files whose names are of the form @ELOG.xxx.n and which are empty (i.e. have a line count of 0). These are log files corresponding to edit sessions which terminated normally. It is run by entering ELOG.CLEANUP (on a terminal) or /INCLUDE ELOG.CLEANUP (on batch).

This program should be run regularly, at a time when there is little activity on the system. A recommended time is each Friday night.

Privileges Required: FILES and LSCAN.

ENQTAB: Display Enqueue Table

A privileged system programmer can use the ENQTAB utility to display the contents of the system's enqueue table. This program is run by entering ENQTAB at a terminal. Optionally, you can enter "ENQTAB string" where *string* is a character string to be searched for in enqueue names and file names. Only entries containing the *string* (as a substring) are displayed. If no string is specified, all entries are displayed.

The ENQTAB utility is useful for determining the number of entries in the enqueue table, and for finding which user (identified by TCB number) has control of a particular resource, such as a file or a UDS. Once the TCB number is known, the user code can be found by running the WHOALL utility. The size of the enqueue table may be increased if necessary by changing the system module ENQRTN. There should be approximately 800 entries for each 100 users signed on.

Privileges Required: LSCAN, CREAD, and DREAD.

FILE.DELETE: Delete a Group of Files

This program deletes (purges) all files whose code and name match specified prefixes.

Note: A similar function can be done by using a file name pattern on the /PURGE command.

Usage:

```
/INCLUDE FILE.DELETE
parameters separated by commas (see below)
```

Parameters:

CDPREF='xxx' Prefix for user code part of file name (0 to 4 characters). This parameter must be specified.

NMPREF='xxx' Prefix for name part of file name (0 to 17 characters). Either NMPREF or NAME (but not both) must be specified.

NAME='xxx' An actual file name (1 to 17 characters). All files with this name whose code matches the specified code prefix will be deleted. NMPREF and NAME must not both be specified.

DELETE=T This parameter must be specified to actually delete the files. If this parameter is omitted, the file names will be displayed but not deleted.

The entire Save Library index is scanned for file names which match both the specified code and name prefixes, and all such files are deleted.

If CDPREF is all blanks, it is considered to match all codes, and similarly for NMPREF. However, CDPREF and NMPREF must not both be blank.

Examples:

```
1.    CDPREF='AB',NMPREF='XY.Z',DELETE=T

2.    CDPREF=' ',NAME='MYPROG',DELETE=T
```

Example 1. deletes all files which have a code starting with AB and a name part starting with XY.Z. Example 2. deletes all files which have MYPROG as their name part.

Privileges Required: LSCAN, FILES.

FILECH: Alter File Names or Attributes

This program changes the name and/or access control of a file or a list of files.

Input control statements are read from unit 5. Each control statement has the following format:

1. Old name in columns 1 to 22 (with or without project code).

2. New name in columns 24 to 45 (or blank if no change in name).

3. 4-byte access control (in hex) in columns 47 to 54 (or blank if no change in access control). Refer to *Chapter 20 - File System* for a description of the access control bits. Access control may also be specified by keywords as follows:

   PRIV
   PUBL (public, but entry not in common index)
   COM (private, but entry in common index)
   SHR
   PRIV,XO
   PUBL,XO
   COM,XO
   SHR,XO

The context editor COPYCOL command can be used to help create the second column of file names. Simply edit a file that contains the list of files you want FILECH to work on. Issuing the editor "COPYCOL 1 22 24 999" command will then create an identical list starting in column 24. This two column list can then be further edited as required.

Privileges Required: LSCAN and FILES.

FIXINDEX: Remove Save Library Index Entry

This program can be used to remove an entry from the Save Library index if something is wrong with it.

This could be necessary if the file cannot be deleted by a /PURGE because the index entry does not point to a valid directory block (file system error 65 or 67).

Note: FIXINDEX should be used when the number of files is small, or when it is desirable to work conversationally. For a larger number of files, utility program FIXINDEX.AUTO should be used rather than FIXINDEX. Even for a small number of files, you will find that FIXINDEX.AUTO is easier and safer to use.

If the file to be removed has the PUBL or COM attribute, both the private and common index entries must be removed. To locate the entry in the common index, specify NAME='*COM:filename' to MFHASH. (FIXINDEX.AUTO automatically removes both.)

First, use MFHASH to get the DEB number for the Save Library index and the index block number that contains the incorrect entry. Next use SYSDMP to dump the index block to get the displacement of the entry and its length. Note the first 4 bytes of the entry. Finally, run the FIXINDEX program specifying the appropriate values.

The following is an example of the procedure described above.

```
*Go
mfhash

-- MFHASH --

DEB NUMBER OF SAVE LIBRARY INDEX =   9
NUMBER OF BLOCKS IN INDEX PRIMARY AREA =   6041
START BLOCK NUMBER =   9
NUMBER OF SEGMENTS IN INDEX =   7

ENTER: NAME='CODE:NAME' OR NAME='*COM:NAME' OR CODE='CODE'
?
name='ih39:.0000476'
IH39:.0000476           INDEX BLOCK NUMBER =   4778
/ca
*Terminated

*Go
sysdmp

SYSDMP -- ENTER PARAMETERS (DEB,DSN,VOL,BLK,...) OR HELP
?
deb=9,blk=4778,blksiz=512
DEB   9 FOUND, BLK SIZE =   512
BLK    4778 READ, LENGTH =     512
0000  13C9C8F3 F94BF0F0 F0F0F4F7 F6000000  *.IH39.0000476...*
0010  010CC300 00000000 00000000 00000000  *..C............*
/can
*Terminated
```

```
*Go
fixindex

-- SAVE LIBRARY INDEX ZAP PROGRAM --

ENTER DEBNUM=,BLKNUM=,DISPL=ZXX,ZAPLEN=ZXX,VER=ZXXXXXXXX
(TO REMOVE BYTES FROM AN INDEX BLOCK)
?
debnum=9,blknum=4778,displ=z00,zaplen=z13,ver=z13c9c8f3

DEBNUM =   9   BLKNUM =  4778
DISPL =   0 (Z0000)   ZAPLEN =  19   VER = Z13C9C8F3

0000  13C9C8F3 F94BF0F0 F0F0F4F7 F6000000  *.IH39.0000476...*
0010  010CC300 00000000 00000000 00000000  *..C............*
-------//--------
-------//--------
01F0  00  *.*

TYPE OK TO REREAD, CHANGE AND WRITE THE BLOCK
?
ok

CHANGED BLOCK:
0000  00  *.*
0010  00  *.*
-------//--------
-------//--------
01F0  00  *.*

ENTER DEBNUM=,BLKNUM=,DISPL=,ZAPLEN=,VER=ZXXXXXXXX
(TO REMOVE BYTES FROM AN INDEX BLOCK)
/can
*Terminated
*Go
```

Privileges Required: LSCAN, DREAD and either MAINT or VIP.

FIXINDEX.AUTO: Automatically Remove Save Library Index Entry

This program is an automated version of FIXINDEX. It removes entries from the Save Library index for files which have error 65 or 67 (directory in error or index/directory mismatch). It is recommended that this program be used rather than FIXINDEX, in most cases.

WARNING: There must be no other users or jobs on the MUSIC system when this job is run. Otherwise the Save Library index may be severely damaged.

Usage:

```
/INCLUDE FIXINDEX.AUTO
code:name1
code:name2
   ...
```

The full file names to be removed from the index are read from unit 5, one name per line, starting in column 1. These names could be found by running the CHKFILES program.

Privileges Required: LSCAN, FILES, DREAD, VIP.

FMFREE: Formatting Free Space

The FMFREE program reads the format-5 DSCB's of a specified disk volume and outputs to unit 10 (or to unit 7 if batch) a job to format all the free extents of the volume. A blocksize of 512 is usually used. The LISTV (list VTOC) function of the disk dump utility (DSKDMP) should be used before running FMFREE, to verify that the free extents do not overlap any allocated space on the volume.

Usage:

```
FMFREE
or
/INCLUDE FMFREE
VOL='volume',BLKSIZ=n,OUT=unit
```

The program may be run from batch or terminal. The default blocksize is 512. The default output unit number is 10 (terminals) or 7 (batch). FMFREE contains default /FILE statements, defining unit 10 as file @FMFREE.OUTPUT and unit 7 as file @FMFREE.OUTPUT7. After running FMFREE, execute the resulting format job in order to actually format the free space.

Note: There is no need to format free space on FBA devices.

Privileges Required:

DREAD and CREAD privileges are needed to run FMFREE. The format job requires the VIP privilege.

FORMAT: Disk Format Utility

The disk format utility is available to MUSIC maintenance personnel for use in formatting disk packs. Along with the program, there are sets of control statements saved in the Save Library. These files have the appropriate control statements for the disk format utility for formatting the standard MUSIC data sets. The format program is normally run from batch. The MUSIC control statements for running a standard system data set format job are:

```
/INCLUDE FORMAT
TITLE='page heading'
VOLUME='xxxxxx'
DEVICE=nnnn   or   DEVICE='FBA'
VOLID='xxxxxx'
DSN='data set name'
/INCLUDE FMTxxx (if needed - see below)
```

The statements used to format a complete User Data Set pack are:

```
/INCLUDE FORMAT
TITLE='page heading'
VOLUME='xxxxxx'
DEVICE=nnnn,CYL=0,cccc   or   DEVICE='FBA'
VOLID='xxxxxx'
/INCLUDE FMTUDS
```

If the FORMAT program is used from a terminal, the control statements may be entered as SYSIN lines (as in the above jobs) or conversationally. The program first reads any input on SYSIN, then switches to read conversationally from the terminal. The normal way of terminating FORMAT is by entering /CANCEL in response to a read prompt. The conversational reads may be suppressed by using the statement /FILE 9 DUMMY before /INCLUDE FORMAT.

The text within quotations of the TITLE statement will be printed at the top of the printed output to identify the job. The value of cuu on the UNIT statement specifies the unit address of the pack to be formatted. The value within quotations on the VOLUME and VOLID statements is the volume label with which the disk pack was initialized. The DEVICE= parameter specifies the type of device being formatted. The DEVICE specification may be one of the following: 2314, 3340.35, 3340.70, 3330, 3330.11, 2305.1, 2305.2, 3350, 3380, 3380.2 (double capacity models), or 'FBA'. For a VM/370 minidisk on a non-FBA device, the CYL parameter must be used to specify the number of cylinders in the minidisk. For further information on these control lines (or those within the files), see "Format Utility Commands".

The /INCLUDE statement points to one of the files listed below. It is required to format entire User Data Set packs and for System Data Sets explicitly mentioned in the list.

FMTUDS to format an entire disk pack as a User Data Set (UDS) volume. (No DSN parameter is needed.) /INCLUDE FMTUDS must not be used if only part of the volume is being formatted.

FMTCOD to format a User Code Table data set

FMTCDX to format a User Code Index data set.

While formatting the disk, any bad tracks found are noted on the output listing along with their alternate tracks. They should be retained for future reference. Note that 2314 and 3340 packs should *not* have any bad tracks, if they do, care should be taken to allocate dummy data sets over the bad tracks so that they will never be used. Alternate tracks on 3330's and 3350's are allowed but may cause system performance degradation due to the automatic alternate seek done by the hardware.

The following is a detailed description of each of the format utility commands. Most commands are made up of FORTRAN NAMELIST-like input. Care should be taken to make sure character parameters are enclosed in quotes. Parameters are separated by commas. Commands too long for one input line can span lines by ending each line at a comma.

Format Utility Commands

Page title generation:

```
TITLE='text'
```

Prints 'text' (maximum 72 columns) at top of next page of output.

Comment generation:

```
COM='text'
```

Prints 'text' (maximum 72 columns) on output.

Volume specification:

```
VOLUME='vvvvvv'
```

Specifies the disk to be formatted was online at IPL time and had the volume label *vvvvvv*.

Unit address specification:

```
UNIT='cuu'
```

Specifies pack to be formatted is mounted on the disk drive with physical unit address *cuu*. This parameter is particularly useful when it is required to point to a pack that was not mounted at IPL time.

Note that either the VOLUME or the UNIT parameter must be specified before any of the following parameters can be used.

Device specification:

```
DEVICE=xxxx,HEAD=a,b,CYL=c,d  or  DEVICE='FBA',BLOCK=nnn
```

Specifies device type. Valid types are given in the table below. Use 3340 for 3344 devices and use 3350 only when operating in native mode. The HEAD and CYL parameters are optional. If given, they override the default head and cylinder ranges for the specified device type. The HEAD and CYL parameters should be used with great care. Figure 17.4 gives the defaults for these values.

Device	Head Range	Cylinder Range
2314	0,19	0,199
3340.35	0,11	0,347
3340.70	0,11	0,695
3330	0,18	0,403
3330.11	0,18	0,807
2305.1	0,7	0,47
2305.2	0,7	0,95
3350	0,29	0,554
3375	0,11	0,958
3380	0,14	0,884
3380.2	0,14	0,1769
3380.3	0,14	0,2654
3390	0,14	0,1112
3390.2	0,14	0,2225
3390.3	0,14	0,3338
9345	0,14	0,1439
9345.2	0,14	0,2155

Figure 17.4 - Default Head and Cylinder Ranges for Formatting Disks

When formatting a 'minidisk' under VM/370, the CYL parameter must be used to define the number of

cylinders on the minidisk.

If 'FBA' is specified CYL and HEAD parameters are ignored. The BLOCK parameter is optional, it can be used to specify the number of blocks on the device and is useful only in the event the device was not attached at IPL time.

Volume label checking:

```
VOLID='xxxxxx',NEWID='yyyyyy'
```

Label of pack on specified unit is read. If pack label is not equal to *xxxxxx*, an error message is issued. If 'SCRATCH' is specified, the label is read and printed but no compare is done.

If the NEWID parameter is specified, the pack is relabeled with the name *yyyyyy*. This parameter is optional.

Data Set Format Request:

```
DSN='data set name',DATA=Zgg
```

The specified data set is to be formatted. The block size specified when it was allocated will be used. The DATA parameter is optional and is described under the next command.

Absolute Format request: (CKD)

```
START=a,b,END=c,d,KL=e,DL=f,DATA=Zgg,NREC=h,LABEL=xxx.
```

Parameters a,b specify the starting cylinder and head (relative to the start of the logical volume as modified by the HEAD and CYL commands) of the area to be formatted; c,d are the ending cylinder and head locations. (Specifying END=9999,9999 will be taken as the end of the pack as determined from the DEVICE specification.) These parameters must be specified but all the following ones are optional.

KL specifies the key length of the records.

DL specifies the length of the data portion of the record.

DATA can be used to specify the character used to fill the formatted records. The default is DATA=ZFF. Use DATA=Z00 to format the Save Library Index data set (SYS1.MUSIC.UIDX).

NREC specifies the number of records per track. If omitted, a value calculated from DL, KL, and the device type is used.

LABEL is used to specify whether the label record (cyl=0, head=0,record=3) is to be preserved. The value of *xxx* should be either TRUE or FALSE. This parameter may be TRUE only if cyl=0,head=0 is included in the START-END range.

The default values for the optional parameters are:

```
KL=0,DL=512,DATA=ZFF,LABEL=FALSE
```

Once KL or DATA parameters are specified, the new value remains in effect until explicitly changed.

Absolute Format request: (FBA)

```
START=a,END=b,DL=t,DATA=Zgg,NREC=h,LABEL=xxx
```

The parameters have the same functions as the CKD case with the following exceptions.

START - starting block number (1st block is 0)

END - Ending block number

KL - is NOT valid and should not be used

Write Data Set Record Request:

```
WRDATA=n
```

A data record is written to the nth block of the data set last formatted by a DSN statement. The first block is numbered zero. The actual data to be written is specified on statements following the WRDATA command. Its format is the same as the data for the RCD command documented below.

Data record request:

```
RCD=c,h,r,REPT=n,RPT=r      or      RCD=b,REPT=n
```

This command allows special data to be written to records already formatted on the disk pack. The parameters *c,h,r* specify the absolute cylinder, head, and record number of the record to be written. The data may be written more than once on successive records. REPT defines the number of times the record is to be written. If REPT is given, RPT should also be given to specify how many records can fit on a track. If omitted, it is assumed that all records requested can be written on the same track as the first one. The second form is used for FBA devices *b* specifies the starting block for the write. The data to be written is specified in the card(s) following the RCD command.

The cards specify either EBCDIC character data or hexadecimal data. As many of these cards as necessary may be used, intermixed as needed. A blank card (blank at least in column 1) indicates end of data specification.

The card format is:

1 C for EBCDIC character data
X for hexadecimal data

2-3 Number of bytes (right-adjusted) of data in the data area of the card. This can be omitted if all the data is punched (that is, no trailing blanks) and no excess data is present on card. For hexadecimal data this is the number of bytes of data (normally one-half of the number of characters punched on data area of card).

4-6 Repetition factor. Data will be repeated this many times. If omitted, one is assumed.

7-9 Must be blank

10-80 Data in appropriate format

Privileges Required:

A code with the VIP privilege must be used. The command /VIP ON must be in effect if the program is run from a terminal.

FPRINT: File Print Routine

This utility prints the contents of the specified list of files. The TAG, attributes and dates last used are printed for each. Each page contains a title giving the file name. Sample JCL to run FPRINT follows:

```
/INCLUDE FPRINT
UNITS=6,ALLUP=T,MAXLIN=NN  (Sample line..see below for details)
FILENAME1
FILENAME2
....etc
```

Parameters:

UNITS=6 Output to unit 6

ALLUP=T Will translate lower case to upper case characters

MAXLIN=nn Will stop listing each file after the specified number of lines have been printed on unit 6

SKIPRD=T Stops the reading of a file after MAXLIN lines have been printed. This saves time for large files, but the file total line count may be omitted. The default is SKIPRD=F.

Privileges Required: None.

GEN.CODES: Generate Groups of Sign-on Codes

This program generates groups of MUSIC user codes, with 4-character random passwords. Output is in the form of ADD commands acceptable to the CODUPD utility. The commands are written to unit 1.

Most installations will want to modify this program to suit their particular needs. The coding here should provide a good starting point to work from.

Usage:

```
Step 1.        /FILE 1 NAME(CMDFILE) NEW
               /INC GEN.CODES
               GROUP='xx',NUM=n,BASE=b,OPT='xxx',OPT2='yyy'
               GROUP='xx',NUM=n,BASE=b,OPT='xxx',OPT2='yyy'
               etc.

Step 2.        /INC CODUPD5
               /INC CMDFILE
```

Step 1 generates the ADD commands, written to the file CMDFILE in the above example. Step 2 actually adds the codes to the Code Table. CODUPD5 is identical to the CODUPD utility, except that it reads commands from Unit 5 instead of 9 and does not print as many information messages.

Parameters:

GROUP='xx' This specifies the first two characters for each user code in the group.

NUM=n Specifies the number of codes to be generated in the group. The maximum is 99 for BASE=10, or 1296 for BASE=36.

BASE=10 OR BASE=36
Defines how the 3rd and 4th characters of the codes are to be generated. BASE=10 uses a 2-digit decimal number, starting at 01. BASE=36 uses the characters A to Z as well as 0 to 9: AA, AB, AC, ..., AZ, A0, A1, ..., A9, BA, BB, etc. BASE=10 is assumed if no base is specified. With BASE=10, the maximum number of codes per group is 99. With BASE=36, the maximum is 1296. The sub code is 000 in all cases.

OPT='xxx' Specifies optional additional CODUPD parameters to be placed on the ADD commands. The maximum length of the character string *xxx* is 59. If you need more than this, specify some of the parameters in OPT2='yyy'. These will be placed on a continuation line in the command file.

OPT2='yyy' More optional CODUPD parameters if the desired parameters do not fit in OPT='xxx'. The maximum length for *yyy* is 80 characters.

Example:

```
/FILE 1 NAME(ADD.FILE) NEW
/INC GEN.CODES
GROUP='XY',NUM=25,BASE=10,OPT='MAX$(100) TOTLIM(200)'
GROUP='AB',NUM=150,BASE=36
GROUP='T2',NUM=300,BASE=36,OPT='PRIME(32) NONPRIME(60)',
OPT2='BATCH(60) DEFAULT(16) AUTOPROG(HELLO)'
```

Privileges Required: LSCAN

GENSAV: Generate Multiple Files from One File

This utility can be used to create individual files from one original file. The utility can create a separate file for each object deck in the original file. Alternately it can produce separate files based on "./ ADD" control statements in the original file.

The original file is usually a tape but it could be a file or a UDS file.

Sample control statements:

```
/FILE N TAPE   ...SOURCE TAPE
/INCLUDE GENSAV
parameters (or blank line)
/INCLUDE FILNAM (IF REQUIRED FOR SELECTION OF './ ADD' CARDS)
```

Parameters:

INPUT input unit of the source tape. Default is INPUT=1.

FILE vector of length 10 containing file #s to be searched on input tape. Default is FILE=1,0,0,0... ie. only file 1 is used

PREFIX prefix of the newly created file name. Default is PREFIX=' '.

SUFFIX suffix of the newly created file name. Default is SUFFIX=' '. The total length of the prefix and the suffix should not exceed 9 characters (or 14 characters if 'code:' is specified in the prefix).

REPL if REPL=.true. is specified, then an existing save file will be replaced by the newly created save file of the same name. Default is REPL=.false.

PUBL if PUBL=.true. is specified, then the files are saved public.

SELECT if SELECT=.true. is specified, then only those './ add' cards with names that are specified in a save file (unit 5) will be processed (one name is specified on each line starting at column 1 in the save file). The default is SELECT=.false. which will process all './ add' cards in the tape.

OBJ OBJ=f causes the source to be scanned for "./ add" cards and moved to save files accordingly. If OBJ=t the source is scanned for *esd* cards. Default=.false.

Privileges Required: None.

INITFBA: Initialize FBA Packs

This program creates a volume label and VTOC on an FBA volume. Also, the first block of the volume is zeroed.

Usage:

```
/INCLUDE INITFBA
parameters (separated by commas) (see below)
additional parameter statements for other volumes (if desired)
```

Parameters:

ADDR='cuu' This parameter is the device address of the volume. It must be specified. This address must have been defined as an FBA device in the NUCGEN device statements when MUSIC was generated, or in the =CONFIG specifications when MUSIC was IPLed. The device may or may not have been ready when MUSIC was IPLed.

VERIFY='xxxxxx'

xxxxxx is the old volume name. This must be specified if a volume label (VOL1) already exists on the volume. Verification may be bypassed by specifying VERIFY='SCRATCH'.

NEWVOL='xxxxxx'

xxxxxx is the new volume name to be used. Must be specified.

OWNER='xxxxxxxxxx'

This is the owner identification to be placed into the new volume label (maximum of 10 characters). The default is OWNER='MUSIC'.

VTOC=n The starting 512-byte block number for the new VTOC. The first block of the volume is number 0. It must be 2 or more. The default is 32.

BLOCKS=n This is the number of 512-byte blocks in the new VTOC. It must be 2 or more, and should

be even. The default is 640 = 20 * 32. Each block can hold 3.5 DSCBs. At least one DSCB is required for each SDS and UDS allocated on the pack.

Example:

```
/INCLUDE INITFBA
ADDR='1A0',VERIFY='MUSXXX',NEWVOL='MUSIC2'
ADDR='1A1',VERIFY='MUSYYY',NEWVOL='MUSIC3'
```

Privileges Required: VIP.

IOTIME: List Input/Output Usage Statistics

This program prints a variety of values associated with DASD and magnetic tape data sets and devices. Cumulative values are logged for each device and channel on the system. In addition, individual logs may be maintained for specific system data sets. The system catalog contains parameters specifying which (if any) data sets are to be logged separately.

For each data set/device/channel listed, the following items are printed (if applicable).

1. Unit Control Block (UCB) or Data Extent Block (DEB) number.
2. Identification
3. Volume Label
4. Usage Count: Total number of I/O requests
5. Busy: Total time (in minutes, seconds and milliseconds) spent executing all I/O requests. Time is measured between the SIO and the first (usually only) interrupt for the device.
6. Busy+Wait: Time spent executing I/O requests while the processing unit was in wait state awaiting completion of the I/O.
7. Queue Count: Number of requests which could not be processed upon receipt but had to be queued due to a device or channel busy condition.
8. Queued: Total amount of time spent in queues waiting for a resource. Note that this value is cumulative. For example, if 3 requests were concurrently waiting for a device for one second, a total of 3 seconds would be added to this counter during this period.
9. Queue + Wait: Time spent in queues waiting for a resource while the processing unit was in wait state.

IOTIME is normally run from batch due to the large amount of output produced.

Usage:

```
/INCLUDE IOTIME
```

It may be run from a terminal by entering IOTIME.

The values printed by this program are those accumulated since the last IPL. Counters printed as '******'

are those in which I/O is in progress at the time the program was run.

Privileges Required: INFO.

LDCNTS: Load Library Usage Count Display

The LDCNTS program prints out load count statistics. Counts are automatically maintained for members loaded from the system load library.

To assign a count to a member loaded from a user load library, run the LDLIBE utility specifying an option such as:

```
NAME='xxx',CHANGE=T,NWCID=n
```

Privileges Required: LSCAN and CREAD.

LDLIBE: Load Library Update

The LDLIBE Utility converts link edited modules into the format used on the system SYS1.MUSIC.LOADLIB data set. Specifically it can perform the following functions:

- Create a load library
- Add, delete, replace and rename members
- Modify attributes of members
- Dump/restore members

This program copies load modules created by the MUSIC Linkage Editor to load libraries. These load modules should be created under the following rules:

- If the module will normally be loaded at a specific address, then specify this address on a .ORG statement. This will save the relocation step that must otherwise be performed.
- Place any .ADD statements before the .ORG statements. These ADD statements are used to define values for external symbols.
- The load module's name must be the same as the member name on the load library. The name is either given on the linkage editor NAME statement or with the NAME= parameter on the /JOB statement.
- The NOSEGTAB option must be used on the /JOB statement.
- Use the MODE=OS option on the /JOB statement if the conflicts with the predefined names of IBCOM, FIOCS#, etc., would otherwise exist.

The following shows the required JCL:

```
/FILE . . .      0 to 4 file statements
/INCLUDE LDLIBE
description statement
command statement
 . . .
 . . .
```

The file statements are used as required to define data sets for dump/restore, load modules and, in the case of UDS type load libraries, the actual load library.

The description statement (1 per job) describes the load library. It has the following parameters which are specified in namelist format.

LIBE=n *n* is the DEB number or unit number of the load library. If LIBE='SYST' is specified, the system load library will be assumed. LIBE may be abbreviated to LIB.

DSN='datasetname',VOL='volume'
The name and volume of the load library data set can be specified instead of the LIBE=n parameter. This allows the LDLIBE program to work with a non-standard or alternate system load library. Use of DSN and VOL requires the VIP privileges, and may result in slower execution of the program.

IN=m *m* is the unit number of the input load module data set.

TEST=T Will suppress all writes to the load library. TEST=F is the default.

DISP='NEW' Specifies that a new load library is to be created.

DIRSIZ=k *k* is the number of directory entries to be allocated. DISP='NEW' must be specified. This is the maximum number of members the load library will be able to contain.

NOTE='text' text to append to $PGM:LDLIBE.LOG when the LDLIBE job is attempted. The text can be used to document the change or why it was done. The file $PGM:LDLIBE.LOG must exist before the LDLIBE job is run for the text to be appended to the file. Otherwise, the text is ignored.

Command statements, if specified, tell the LDLIBE program what to do. The following is a list of parameters that you could specify on a command statement.

NAME='member'
Specify member to be processed. When adding a member, this is the name of the member in the load module file (IN=).

RENAME='name'
name to be assigned to the added member in the load library (if different from NAME=).

DEL=T To delete member.

REPL=T To replace member. This option should not be used unless the member will not to be used until the system is re-IPLed.

RENT=T Member is to be flagged as reentrant.

CNTID=n Usage count ID. Default is 0. The maximum is 50. (Additional information can be found under the LDCNTS writeup.)

DUMP=n *n* is the unit number of data set for dump. A logical record length of 80 is used. Any block-size that is a multiple of this can therefore be used.

RESTOR=n *n* is the unit number of data set for restore.

CHANGE=T To modify attributes of an existing member in the load library.

Note: If CHANGE=T is specified, the attributes which can be changed (RENT, CNTID, and NAME) should be specified as NWRENT=, NWCID=, and NWNAME= respectively.

Examples:

1. Creating a load library on a user data set.

```
/FILE 1 UDS(CODEXXXX) VOL(MUSIC1) OLD
/INC LDLIBE
LIBE=1,DISP='NEW',DIRSIZ=20
```

2. Adding the member XX1 to the system load library.

```
/FILE 1 UDS(CODELMOD) VOL(MUSIC1) OLD
/INC LDLIBE
LIBE='SYST',IN=1
NAME='XX1'
```

3. Changing some attributes and the name of the member XX1 in a UDS load library, adding the member XX2 and XX3, and replacing XX4.

```
/FILE 1 UDS(CODEXXXX) VOL(MUSIC1) OLD
/FILE 2 UDS(CODELMOD) VOL(MUSIC1) OLD
/INC LDLIBE
LIBE=1,IN=2
NAME='XX1',CHANGE=T,NWRENT=T,NWCID=1,NWNAME='XX5'
NAME='XX2'
NAME='XX3'
NAME='XX4',REPL=T
```

4. Dumping members of the system load library to a card image file.

```
/FILE 1 NAME(DUMP.FILE) OLD
/INC LDLIBE
LIBE='SYST'
DUMP=1
name specification     (see below)
     . . .
     . . .
```

name specification may be any of the following:

name	single name in col 1-8
name1-name2	all members from name1 to name2
prefix..	all members beginning with "prefix"
*ALL	all members

The RESTOR operation follows the same conventions.

Privileges Required: LSCAN, DREAD, SYSMNT or VIP.

LDLIST: Load Library Directory List

The LDLIST program displays a listing of a load library directory. The listing shows the name, length, starting block number, origin, entry point address, and attributes of each member.

Usage:

To list the system load library, enter LDLIST. For other load libraries, use the following job:

```
/FILE . . .    (if UDS load library)
/INCLUDE LDLIST
    parameters separated by commas
```

Parameters:

LIBE=n *n* is the unit number or DEB number of load library. If LIBE='SYST' is specified, the system load library is listed.

Privileges Required: LSCAN and DREAD.

LIBINDEX.CLEAN1: Clean Save Library Index Overflow Area

There are two programs for cleaning up the Save Library Index overflow area: $PGM:LIBINDEX.CLEAN1 and $PGM:LIBINDEX.CLEAN2.

Parts 1 and 2 are independent, but normally part 1 would be run before part 2.

Important: This program can be run only when there is no other activity on the Save Library. No other users should be on the system when this program is run. otherwise the Save Library Index may be damaged.

Usage:

```
/INC LIBINDEX.CLEAN1
NOWRT=T OR F
```

To run the program with disk writes bypassed, set NOWRT=T. To actually update the index, set NOWRT=F.

LIBINDEX.CLEAN1 reads Save Library Index blocks, looking for pointers to auxiliary blocks. It moves as many index entries as possible from the auxiliary block to the primary block, and, if all is moved, it sets the pointer in the original block to zero.

This cleanup makes the /LIBR command more efficient, and avoids running out of auxiliary blocks.

Messages are issued for the following cases:

1. all entries (if any) moved from auxiliary to primary and pointer in primary has been set to zero.

2. No entries were moved to primary, because there was not enough space in primary ("no change").

3. Some, but not all, entries were moved to primary.

4. The auxiliary block points to another auxiliary block. In this case, no change is made. (This should be very rare.)

Privileges Required: LSCAN, DREAD, CREAD, and VIP.

LIBINDEX.CLEAN2: Clean Save Library Index Auxiliary Area

The $PGM:LIBINDEX.CLEAN2 program (part 2) works in conjunction with $PGM:LIBINDEX.CLEAN1 (part 1 above). Part 2 cleans up the Save Library Index auxiliary area.

Important: This program can be run only when there is no other activity on the Save Library. No other users should be on the system when this program is run, otherwise the Save Library Index may be damaged.

Usage:

```
/INC LIBINDEX.CLEAN2
NOWRT=T OR F
```

To run the program with disk writes bypassed, set NOWRT=T. To actually update the index, set NOWRT=F.

This program moves used auxiliary blocks to the beginning of the overflow area. This is necessary from time to time because MFIO never frees auxiliary blocks, and would eventually run out of blocks in the overflow area. This causes severe errors in the library, because MFIO does not handle the error condition correctly. Errors 67, etc. occur.

Privileges Required: LSCAN, DREAD, CREAD, and VIP.

LIBINTEG: Check Save Library Integrity

The LIBINTEG utility, normally run as a batch job, reads the entire Save Library index and all file directory blocks, in order to check the correctness of the library space maps.

Note: This utility should be run only when there is no allocation or deallocation activity on the Save Library. To ensure this, no users should be allowed to sign on the system while LIBINTEG is being run.

Usage:

```
/FILE ... (as needed)
/INCLUDE LIBINTEG
parameters separated by commas (see below)
```

Parameters:

SPCINF=n — Unit number of an optional sequential output file (normally a UDS), LRECL=150. This records file name, directory pointer, and extent info for each private entry in the index. It can be scanned later to find which file or files are allocated to a particular space unit in the library. The default is SPCINF=0, meaning that this output file is not used.

COMINF=n — Unit number of an optional sequential output file (normally a UDS), LRECL=69. This records file name, directory pointer, and '0' (private) or '1' (common) for each entry in the index. If COMCHK=T, this file is later sorted by file name and read back to detect inconsistencies in the index (e.g. private and common have different index pointers, common has no private entry, more than 1 private or common entry for a file). The WORKU parameter must be specified if this parameter is used. The default is COMINF=0, meaning that this output file is not used.

COMCHK=T/F — Default is F if COMINF=0, otherwise T. Specifying COMCHK=F suppresses sorting and checking of COMINF data. The idea is that the sorting and checking could be done later by a separate program. This is desirable for very large Save Libraries.

WORKU=n — Unit number of a sort work file (a UDS). This is required when the COMINF parameter is used. The default is WORKU=0, i.e. no work file.

WRTBAC=T — This option causes reconstructed space maps to be written to disk, if they are different from the existing maps. The VIP privilege is required if this parameter is used. The default is WRTBAC=F (no writes to disk).

Note: Use the WRTBAC=T parameter with caution. Under no circumstances should any users be signed on to MUSIC when this option is used.

Privileges Required:

LSCAN, DREAD, and CREAD. The VIP privilege is also required if the parameter WRTBAC=T is used.

LIBSPACE: Show Library Space Status

Type LIBSPACE to display the space status of each Save Library data set. The system data sets making up the Save Library are numbered consecutively starting at 1. The data set names are normally SYS1.MUSIC.ULnn. LIBSPACE shows the amount of free (unused) space on each data set and the total free space available on the library as a whole. For each data set, the amount of space in the largest 5 free extents is also shown. Space values are in units of 1K = 1024 bytes.

As in the MVS system, MUSIC never uses more than 5 extents to satisfy a single request for primary or secondary space allocation. For example, it may be that a Save Library data set has 200K of free space but the largest 5 extents are only 2K each. That would mean that the largest file the system could allocate on that data set is 10K. The free space is there, but it is fragmented and therefore not useful for allocating larger files.

The LIBSPACE utility calculates a *fragmentation index* for each SL data set. It is a number between 0 and 1 which is a rough measure of how fragmented the space is. It is defined as (t-s)/8096, where t is the total free space in the data set and s is the sum of the 5 largest free extents. The higher the index, the more fragmented the space is. SL data sets with the highest fragmentation indices are the best candidates for the free space reorganization procedure described in *Chapter 20 - File System*. The MFMOVE utility is used to do the reorganization.

Privileges Required: LSCAN, DREAD and CREAD.

LOADPDS: Restore Members from OS IEBCOPY Tape

This program can restore, to the MUSIC Save Library, members of OS partitioned data sets which have been dumped onto tape using the IBM IEBCOPY utility.

Usage:

```
/FILE 1 TAPE VOL(xxxxxx) RECFM(U)
/INC LOADPDS
FILE=n, PREFIX='code:', SUFFIX='.s', SELECT=,REPL=, PUBLIC=
member names to be selected - one per line if required
FILE=n, etc.
FILE=n, etc
```

Parameters:

FILE=n *n* is file number on input tape.

PREFIX='code:' prefix for generated file names.

SUFFIX='.s' suffix for generated file names.

SELECT= can be TRUE or FALSE. If TRUE, a list of member names must follow.

REPL= can be TRUE or FALSE. if TRUE, existing files are replaced.

PUBLIC= if TRUE, files are made PUBLIC.

Control statements must be in order of ascending file number. The list(s) of files to be selected can terminate with an END OF FILE or "-" in column 1, if subsequent control statements are present. Selected files pertain only to the preceding control statements. Values specified on previous control statements are carried over to subsequent ones unless specifically modified.

Note: Record format "U" should be specified for the tape regardless of its actual record format.

Care should be taken when specifying the file number for standard labeled tapes, since MUSIC counts both header and trailer labels as files.

Privileges Required: None.

LOOKUP: Display the Table of Privileged Programs

This program displays the table of privileged programs. The table consists of file names and associated privileges, as defined in the system module LOOKUP. Each of the file names in the table should have the exec-only (XO) attribute, in order for the privileges to apply.

Usage:

To run the program, simply type LOOKUP when in *Go mode.

Privileges Required: LSCAN and CREAD.

LPA: Display the Contents of the Link Pack Area

This program displays the names of all the members in the Fixed Link Pack Area (FLPA) and the Pageable Link Pack Area (PLPA). Other information such as starting address, length, and entry point address is also displayed for each member.

Usage:

To run the program, enter LPA when in *Go mode.

Privileges Required: LSCAN and CREAD.

MAPMEM: Memory Map

This utility displays memory usage.

```
 MAPMEM: MEMORY REPORT    THU MAY 12, 1994  13:58:17

TOTAL MEMORY:  8192K,   PAGE POOL:  4740K
TCBS:  105, RCBS:   44, MAXRRS:   592K, MAXMPL:  8

                          V-START V-END   R-START R-END    LEN
LOW CORE                  000000  000FFF  000000  000FFF     4K
PAGE POOL 1                                000000  030000   192K
NUCLEUS                   800000  867FB0  030000  097FB0   415K
TCBS,UCBS,TERM BUFS,ETC.  867FB0  8850A4  097FB0  0B50A4   116K
BPOOL                     8850A4  8A5034  0B50A4  0D5034   127K
TCPR/TCPW BUFS FOR TCP/IP 8A5034  8A5034  0D5034  0D5034     0K
SYS DS BLKS, DEBS, ETC.   8A5034  8A6AF0  0D5034  0D6AF0     6K
LIB BITMAP CACHE (ULMAPS) 8A6AF0  8B7370  0D6AF0  0E7370    66K
FIXED LPA                 8B7370  A7F2E0  0E7370  2AF2E0  1823K
RAM DISK                  A7F2E0  AFF2E0  2AF2E0  32F2E0   512K
TRACE TABLE               AFF2E0  B032E0  32F2E0  3332E0    16K
NUCLEUS MAP TABLE         B032E0  B04000  3332E0  334000     3K
RCBS                      B04000  B5C000  334000  38C000   352K
PAGE POOL 2                               38C000  7FFFFF  4559K
```

Figure 17.5 - Sample run of the Map Memory program

Usage:

This program is available from the ADMIN facility option 1 14. You can also run this program by entering MAPMEM when in *Go mode.

Privileges Required: CREAD

MFACCT: Save Library Accounting Dump

This program scans the entire MUSIC Save Library and produces printed and/or card image output giving information on library utilization. A record is produced for each user code, giving the total number of files in the user's library, the total space occupied by the files, and the corresponding cost. The format of these records is given at the end of this writeup.

MFACCT is normally run when no users are signed on the system. If this is not the case, then the UPDUCR parameter described below must left as FALSE, since apparent mismatches would be likely.

To run this program from batch, the following deck set up is used:

```
/INCLUDE MFACCT
    input parameters
```

The input parameters are:

PRINT=TRUE or PRINT=FALSE

OUT=n (unit number for output)

CHARGE=X (charge rate in cents/K-bytes)

UPDUCR=TRUE or UPDUCR=FALSE

The PRINT parameter specifies whether printed output should be produced.
The OUT parameter specifies the unit on which the card images are to be produced. If 1, 2, 3, or 4 is specified, a /FILE statement for a tape or disk data set must be present (placed before the /INCLUDE statement). If OUT=0 is specified, no card image output is produced. The CHARGE parameter specifies the number of cents to be charged for each 1024 bytes of library space. The number of cents may be given as a floating point value (with a decimal point). The UPDUCR (for *Update User Control Record*) parameter indicates whether the User Control Records should be updated to match the actual library space if a discrepancy is found. If more than one parameter is used, they should be separated by commas. The default parameters are:

```
PRINT=TRUE,OUT=1,CHARGE=3.0,UPDUCR=FALSE
```

If the default parameters are not to be changed, the entire statement can be omitted.

If the program is run from a terminal, the input parameters are entered conversationally.

MFACCT Statements

Column	Contents
1- 4	Identification 'ARXD'
5- 8	First 4 characters of file ownership id; ends in a + if id is greater than 4 characters.
9-11	User subcode '000'
12-19	Blank
20-25	Time of MFACCT run, HHMMSS
26-31	Date of MFACCT run, DDMMYY
32-39	Total cost in cents
40	System number or blank (n from call SYSENV(2,n))
41-47	Total space of user's files, in units of 1024 bytes
48-52	Total number of files
53-55	Charging rate, in 1/10 cent per 1024 bytes
56	Blank
57-72	File ownership id (16 characters, trailing blanks)

Privileges Required: DREAD or MAINT.

MFARCH: Save Library New Program Archive

This program is designed to provide backup copies of files. It copies to tape all files which were created or modified since the previous run of the program. In addition, over a period of a specified number of program runs, it attempts to dump all files at least once, even those files which are not modified. Each file's attributes, tag field, etc. are dumped along with the file's data.

The program should be run on a regular schedule, such as once a night. It can be run while terminal users are active, but you may find it more convenient to shut down MUSIC and reload with NOTERM before running the archive.

If you have not run this utility before there is a chance that a large number of files have been changed on your system. This will cause this utility to backup an unusual number of files the first time it is run. This would probably exceed the capacity of a single tape reel. This could also happen if you changed an unusual number of files during a system upgrade. To avoid this you should run the SETFBN utility which will solve this problem and start you off in a way that a reasonable portion of your library is backed up each night over your cycle.

A companion program, MFCHEK, reads the output from MFARCH, ensures that it is readable, cross-checks several statement and file counts within the output, and updates the master backup number. The master backup number cycles between 1 and 255. It is kept in a system file, and is increased by 1 for each successful MFARCH/MFCHEK run. It is important to run MFCHEK after running MFARCH; otherwise the master backup number will not be updated, and the same files will be dumped again the next time MFARCH is run.

If desired, two identical dump tapes can be produced. These two tapes normally would be interchanged on their drives before being read by the checkout program. MFCHEK compares the two tapes to make sure they are identical. This procedure minimizes undetected tape hardware problems and checks that the data is valid.

The MFREST program is used to retrieve files from a dump tape and restore them to the library.

To run the archive program, the following statements are used:

```
/FILE 1 TAPE BLK(4000) LRECL(80) VOL(TAPE1) OLD RECFM(U)
/FILE 2 TAPE BLK(4000) LRECL(80) VOL(TAPE2) OLD RECFM(U)
/INCLUDE MFARCH
UNITS=1,2,PERIOD=n,MAXOLD=m
```

If only one copy of the output tape is desired, omit the file statement for unit 2 and use UNITS=1 instead of UNITS=1,2. A different tape blocksize may be used if desired. To fit as much data on a tape as possible, use a large block size, such as BLK(16000), and use the highest density, i.e. DEN(6250) if available.

UNITS=n,m specifies the output unit number(s) for the dump. If two units are specified, two identical copies of the dump are produced. specifying unit 0 suppresses dump output. Default is units=1. Units 3 and 4 are reserved for work files and hence cannot be used.

PERIOD=n specifies the number of MFARCH/MFCHEK runs over which the program attempts to dump every file at least once, even those files which are not modified. The default is PERIOD=12. For example, with PERIOD=12, a set of 15 tapes could be used. Each MFARCH run would use the next tape in the cycle, so that the set of tapes should contain at least one copy of every file.

A file is a candidate for archiving if (1) the file was created or modified since the last MFARCH/MFCHEK run, or (2) x is greater than or equal to PERIOD, where x=k-f (if k-f>0) or x=k-f+255 (if k-f<=0). Here, *k* is the master backup number and f is the file's backup number. The file's backup number is set to *k* once it has been archived by MFARCH.

MAXOLD=m is the maximum number of non-new, non-modified files to be dumped (case (2) above). The default is MAXOLD=2500. MAXOLD*PERIOD should exceed the total number of files in the library, since approximately (total files)/PERIOD *old* file can be expected as candidates for dumping on each run.

MAXREC=n If a non-zero number is specified, it is the approximate maximum number of 80-byte logical records that should be written to each output tape. After that many records, the program will stop, with a message that there are more files to do and that the operator should run another MFARCH job to dump the remaining files to the next tape. Also, a job return code 2 is set (if there are no serious errors). This is to handle the case where there are too many files to fit on one tape. MAXREC should be calculated as about 90% of the number of records on a full tape, based on tape density, blocksize, tape gap length, etc. For example, for a 2400-ft tape at 6250 bpi, blksize 32000, gap 0.3 in., a full tape holds about 2,120,000 80-byte records, so use about MAXREC=1900000. The MAXREC=n parameter is ignored if dumping files from specified library data sets (ARCLIB=). Default is MAXREC=0, i.e. the program will assume an infinite size tape.

Note: an MFCHEK job should be run before the second MFARCH job, so that the master backup number will be updated. Otherwise the same files will be dumped again.

MBNFIL='filename'
specifies the file which contains the master backup number. The default is MBNFIL='$PGM:MASTER.BACKUP.NUM'. and this file is automatically set up during MUSIC installation. The same file name should be specified to the MFCHEK program, via the MBNFIL parameter.

ARCLIB=n1,n2,...
causes all files in library numbers *n1, n2,* ... to be archived to tape. Library number *nn* is the system data set SYS1.MUSIC.ULnn. From 1 to 100 numbers may be specified. The regular dumping of new or modified files is not done if the ARCLIB parameter is used.

NAMES=n requests that the names of all archived files are to be written to unit number *n*, one file name per record, starting in column 1. The unit number *n* must be from 7 to 15. The file must have record format V or VC, or have a record length of 64 or more.

LISTNG=n specifies an output unit number for the sorted listings of the names and attributes of the archived files. The unit number *n* should be from 6 to 15. This provides a way of suppressing the listing or directing it to a file. The file should have record format V or VC, or have a record length of at least 133. LISTNG=0 suppresses the listing output. The default is LISTNG=6 (output to the printer).

VERTAP=T causes the output tape or tapes to be read to check that an archive tape is mounted. The first record of the first tape file is read and checked for *MFARCHIVE in columns 1-10. The default is VERTAP=F (no checking). VERTAP must be omitted or specified as F if a tape is being written by MFARCH for the first time.

VERVOL=T causes the volume name in the first record of the output tape or tapes to be compared with the volume name on the /FILE statement. If the names are different, MFARCH assumes that the operator mounted the wrong tape and does not do the archive. This parameter is ignored unless VERTAP=T. The *MFARCHIVE header record should contain VOL=vvvvvv in columns 51-60, where *vvvvvv* is the volume name. Starting with MUSIC/SP Release 1.1, MFARCH puts the current volume name on the header record when it writes to the tape. The default is VERVOL=F (no volume name checking).

MFCHEK=T Causes the MFCHEK function to be done at normal end of MFARCH, as part of the same job. The advantage of this is that a second tape mount is avoided. MFCHEK reads and checks the output tape(s) and adds 1 to the master backup number. Refer to the description of the MFCHEK utility. When MFCHEK=T, MFARCH rewinds the output unit(s) and calls MFCHEK as a subroutine. All of the required parameters for MFCHEK must follow the MFARCH parameters in the input stream. The default is MFCHEK=F.

UPDATE=F (Normally not used) causes the program not to change the file's backup number as the file is dumped. The default is UPDATE=T.

In the following example, only one output tape is produced, volume name checking is done, file names and output listings are written to files instead of being printed, and the MFCHEK function is done in the same job. The final parameter line is for MFCHEK. The tape number 03 in the /FILE statements could be generated automatically by the AUTOSUB utility when this job is submitted to MUSIC.

```
/FILE 1 TAPE VOL(ARCH03) LR(80) BLK(16000) DEN(6250) OLD
/FILE 11 NAME(ARCH03.LIST) RECFM(VC) NEW(REPL) SPACE(100)
/FILE 12 NAME(ARCH03.NAMES) LRECL(22) NEW(REPL)
/INCLUDE MFARCH
UNITS=1,PERIOD=8,MAXOLD=2000,VERTAP=T,VERVOL=T,
LISTNG=11,NAMES=12,MFCHEK=T
UNITS=1,UPDMBN=T
```

Note on work files:

Work files on units 3 and 4 are defined in the MFARCH executor file. /FILE 3 should have a record length of 80 and at least as many records as the number of files to be archived. /FILE 4 is a sort work file, and must be twice that size. The standard MFARCH file allows for archiving up to 7500 files. If you need more, provide larger work files. /FILE 4 must be a UDS file.

Information on 3480 and 3490 Cartridge Tapes:

You don't have to change anything in the MFARC2 jobs to use 3480 cartridge tapes. You can specify

DEN(6250) or DEN(HIGH), they give the same result. The actual density is 38000bpi, but MUSIC does not know about that.

For the MFARC2 tape size parameters, use TAPDEN=38000,TAPSIZ=500, TAPGAP=0.3. A high block-size like 32000 is recommended. These numbers are equivalent to 2.1 million 80-byte records. (See the TAPELEN program.)

The IBM manuals say that nominal capacity of a cartridge recorded in 18-track mode is 200MB, i.e. 2.5 million 80-byte records. Using density 38000BPI, block size 32000, and gap 0.12 inches, this gives 501 feet. It's not clear what the actual gap size is.

Some 3490 models can record in 36-track mode, resulting in a capacity of 400MB. Also, the 3490E can use the "enhanced capacity" cartridge (physically different), which doubles the capacity to 400MB (18-track mode) or 800MB (36-track mode).

All of the above capacity numbers are for uncompacted data, i.e. IDRC (Improved Data Recording Capability) off. Capacity with IDRC can be much higher.

Privileges Required: LSCAN, FILES, MAINT, DREAD, and CREAD.

MFARC2: Selective File Archive

This program archives to tape (or disk) a specified group of files. The dump format is the same as that produced by the MFARCH utility. The MFCHEK program (with UPDMBN=FALSE) can be used to check the dump, and MFREST can be used to retrieve files from the dump.

Usage:

The following control statements are used to execute the program:

```
/FILE 1 TAPE BLK(4000) LRECL(80) VOL(TAPE1) OLD RECFM(U)
/FILE 2 TAPE BLK(4000) LRECL(80) VOL(TAPE2) OLD RECFM(U)
/INCLUDE MFARC2
parameters separated by commas (see below)
code:name       )
code:name       )   optional additional file names
...             )
```

One or two output tapes may be used. For one tape, omit the second /FILE statement and specify UNITS=1. For two tapes (two copies of the dump), specify UNITS=1,2. A different tape blocksize may be used if desired. Units 3 and 4 may not be used.

Parameters:

The following may be specified on the parameter statement. If more the one statement is needed, terminate with a comma and continue parameters on the next statement.

UNITS=n,m (or TAPE=, OUTPUT=, OUT=)
: The output unit number or numbers. A zero unit number suppresses output. The default is UNITS=1.

TAPFIL=n
: When archiving to a multi-file tape, this parameter specifies the sequence number (1,2,...) of the tape file to be written. The default is TAPFIL=1. Note that writing to file *n* (TAPFIL=n)

does not disturb existing files 1 through n-1, but destroys any following files on the tape.

CODES='xxxx','xxxx',...
: A list of one or more userids, or userid patterns, whose files are to be archived. A maximum of 3000 userids may be specified. For each userid all files belonging to that userid are archived. These are actually file ownership ids, rather that userids. A pattern contains wild characters * and/or ?. If the UDATE parameter is given, only files satisfying the date condition are archived. The special userids *COM and *USR may not be specified in the CODES parameter.

ALL=TRUE
: This causes the entire library (all codes) to be archived, subject to the UDATE parameter if UDATE is used.

UDATE='ddmonyy'
: Specifies a 7-character date, for example, 01JAN78. The 3-letter month abbreviation consists of the first 3 letters of the month's name. Only files whose last-read and last-written dates are both less than or equal to UDATE are archived. This provides a means of archiving inactive files.

NAMES=n
: This parameter, if used, requests that the names of all archived files are to be written to unit number *n*, one file name per record, starting in column 1. Do not use unit 3 or 4, since the program uses these as work files. For example, the resulting file of names could be used to purge the archived files: change the names to purge commands by using the editor command C//PURGE /*, then feed the purge commands to the Editor. The file must have record format V or VC, or have record length 64 or more.

LIST=FALSE
: This suppresses the printed list of archived file names. The default is LIST=TRUE.

JOBFIL='filename'
: This parameter is used if a single tape is not large enough to hold the output. It allows continuation jobs to be scheduled, in order to complete the archive on additional tapes. *filename* is the name of a file containing sample control statements to be used for the continuation job. It should specify the same parameters as in the initial job. Do not place /ID, /PASSWORD, /PAUSE, or /END statements in this file. The file is not modified by the program. A new file is created for the submit. Characters "01" in any tape volume names in /FILE statements are changed to the run number, and appropriate run=n and indstb=n parameters are added. Characters "01" at the end of a file name, defined by a /FILE statement with the NEW or NEW(REPLACE) option, are automatically replaced by the 2-digit run number (this is useful for the output file for the NAMES=n option). The new file is called @ARCJOB.CONT, and the continuation job is submitted to the internal reader, class 1, with a /PAUSE and using the same code/subcode as the current job. Example of file contents:

```
/FILE 1 TAPE VOL(ARC01A) LR(80) BLK(16000) DEN(6250)
/INC MFARC2
ALL=T,LIST=F,JOBFIL='THIS FILE',
TAPSIZ=2400,TAPDEN=6250,TAPGAP=0.3,TAPBLK=16000
```

: (the 2nd tape volume will be ARC02A, the 3rd ARC03A, etc.)

TAPSIZ=n,TAPDEN=n,TAPGAP=x,TAPBLK=n
: Values used in estimating the length of tape used and available. These parameters are used only if the JOBFIL parameter is specified. TAPGAP is REAL*4. The others are INTEGER*4. TAPSIZ is in feet, TAPDEN in BPI, and tapgap (interblock gap size) in inches. TAPBLK is the tape blocksize. The program will reduce TAPSIZ by 50 feet as a safety margin.

Defaults are: TAPSIZ=2400,TAPDEN=1600,TAPBLK=16000 the default interblock gap is TAPGAP=0.3 if TAPDEN=6250, or TAPGAP=0.6 if tapden is other than 6250.

Note: The nominal tape gap for 9-track 1600 BPI is 0.6 inches; for 6250 BPI it is 0.3 inches.

A list of additional specific file names to be archived may be specified after the parameter statement, one file name per statement, starting in column 1. Each file name must include the 4-character userid. The special userids *COM and *USR may be used here, but they will automatically be changed to the actual userid of the owner of the file. The UDATE parameter does not apply to these files.

Wild characters * and ? (file name patterns) may be used in the file names, in order to archive groups of files. Files to be excluded from the archive can be specified by placing - (dash) in column 1 and a name or pattern starting in column 2. Exclusions do not apply to specific non-pattern file names or to the CODE parameter.

The optional parameter DUMPNAME=filename may appear on the file name statement. It specifies a different file name to appear in the archive output. For example:

```
/FILE 1 ...
/INCLUDE MFARC2
UNITS=1
ABCD:FILEX
ABCD:FILEY      DUMPNAME=ABCD:FILEPQ
ABCD:SAMPLE     DUMPNAME=EFGH:TEST
```

The files ABCD:FILEY and ABCD:SAMPLE will appear in the output with names ABCD:FILEPQ and EFGH:TEST.

The following example shows how all files belonging to several codes (ABCD, DEFG, and all codes starting with JK) can be dumped in one run:

```
/FILE 1 ...
/INCLUDE MFARC2
CODES='ABCD','DEFG','JK*'
```

Privileges Required:

LSCAN, FILES, MAINT, SYSCOM and CODES. SYSCOM and CODES are needed only if JOBFIL parameter is used.

MFCHEK: Save Library Archive Checkout

The statements used to run the Save Library archive tape checkout program are as follows:

```
/FILE 1 TAPE VOL(xxxxxx) SHR LR(80) BLK(nnnnn) RECFM(U)
/FILE 2 TAPE VOL(xxxxxx) SHR LR(80) BLK(nnnnn) RECFM(U)
/INCLUDE MFCHEK
parameters separated by commas (see below)
```

As with the archive program, omit the second /FILE statement and use UNITS=1 if only one tape is involved.

The following may be specified on the parameter statement.

UNITS=n,m The input unit number or numbers. For two tapes, specify UNITS=1,2. The default is UNITS=1.

UPDMBN=TRUE This causes the program to add 1 to the master backup number if the checkout is successful. This should always be specified when MFCHEK is run in combination with MFARCH, except that the UPDMBN=TRUE parameter must not be used if the ARCLIB parameter of MFARCH was used. MFCHEK can also be used to check dump tapes produced by the MFARC2 utility described in this chapter. In that case, omit the parameter UPDMBN=TRUE. The default is UPDMBN=FALSE.

NAMES=TRUE This causes a listing of the names of all the files contained on the archive tape to be printed. The default is NAMES=FALSE.

INFO=TRUE This causes the information line for each file contained on the archive tape to be printed. The information line has the file name, logical record length, record format, access control options, total space allocated, the number of 512-byte blocks dumped, and the time and date the file was archived. The default is INFO=FALSE. If the file name is larger than 22 characters, the complete file name appears on the additional second INFO line.

TAPFIL=n This specifies the tape file sequence number (1,2,...) containing the archive output to be checked. The default is TAPFIL=1 which means the first tape file is to be checked. This parameter is used only when reading from a multi-file tape.

NFILES=n Specifies the number of tape files to process. Default is 1. For example, TAPFIL=3,NFILES=2 would process tape files 3 and 4.

CHKMBN=T Specifies that the master backup number on the tape header record (*MFARCHIVE record) is to be compared with the current master backup number. If they are unequal, the master backup number is not updated. This parameter is used only if UPDMBN=T. the default is T if UPDMBN=T, F otherwise.

MBNFIL='filename'
Specifies the full file name of the file containing the master backup number (max 22 chars). This is used only if UPDMBN=t. Default is MBNFIL='$PGM:MASTER.BACKUP.NUM'.

SCAN='xxx' Causes info line (see the info parameter) to be printed for each file whose full name begins with the specified characters *xxx*. *xxx* can be 1 to 64 characters long. File names include the userid (code:...). For example, SCAN='ABCD:TEX' prints info on all files belonging to userid ABCD and starting with 'TEX'. The name part of the file name must not start with a backslash. Use SCAN='ABCD:XYZ" but not SCAN='ABCD:\XYZ'

If the archive or checkout job fails, or checkout does not print the message `CHECK-OUT OK,` either or both jobs should be rerun, as appropriate, until a successful checkout is obtained. A rerun of the archive job dumps the same files as the original run.

Privileges Required: LSCAN and FILES.

MFHASH: Save Library Index Hasher

MFHASH is used to locate the Save Library index block number, given the file name or UCR code. The first block of the index is numbered 1.

To locate the index block number for a given private file name specify:

```
NAME='code:filename'
```

To locate the index block number for a given file name in the common index specify:

```
NAME='*COM:filename'
```

To locate the index block number for a UCR entry specify:

```
CODE='code'
```

Privileges Required: None.

MFINDEX: Summarize Save Library Index Usage

This program reads the Save Library index and displays various values and tables for the number of files, the index space used, file name lengths, etc.

Privileges Required: LSCAN.

MFMOVE: Reorganize Save Library Free Space

The MFMOVE utility can be used to move files from fragmented library data sets, eliminating this fragmented condition. (The LIBSPACE utility can be used to locate these fragmented data sets.)

It can be run on batch or on a terminal, but must be run on a code with the VIP privilege. The program can process a number of Save Library data sets in a given run, and functions by moving files from those specified libraries to free space elsewhere. Normally only a small number of libraries (say 4 or 5) are specified. The program suppresses allocation on the libraries it is processing for the duration of its execution. For this reason, there should be sufficient space in the remaining library data sets to contain the files to be moved. If for some reason the program is cancelled or runs out of time, the specified library data sets will be unavailable for the allocation of new files until the next IPL.

Example:

```
/SYS TIME=MAX
/FILE 1 NAME(ERRS) NEW(REPL)
/INCLUDE MFMOVE
LIBS=5,6,8,9
```

This will move files from Save Library data sets 5, 6, 8 and 9 to elsewhere in the Save Library. If for some reason a file cannot be moved, it is left as it is and its name and the error condition are recorded on unit 1.

Privileges Required: LSCAN, CREAD and VIP.

MFREST: Restoring of Archived Files

The MFREST utility reads an archive tape produced by the MFARCH or MFARC2 programs, searches for specified files, and copies the files to disk, using either the original names or new names.

The files to be restored are specified by one or more parameter statements. Each parameter statement gives the name of a file (or group of files) to be searched for on the tape, using the NAME parameter, and optionally, gives the name of the file (or group of files) to which the files are to be restored, using the TO parameter. The first parameter statement may specify other options (INPUT, TAPFIL, FIXUP, ALL, RLSE, REPL, SETUI, FIXUCR, SETBUP, PERIOD, and CURMBN) which apply to the entire job.

The control statement set up is as follows:

```
/FILE 1 TAPE VOL(vvvvvv) BLK(nnnnn) LRECL(80) SHR RECFM(U)
/INCLUDE MFREST
first parameter statement
second parameter statement
...
```

Parameters:

NAME='filename' or NAME='prefix*'
: The full file name (including ownership userid) to be searched for in the dump. If the last character is an asterisk (*) or a plus sign (+), all files whose names start with the specified prefix will be restored. NAME= may be abbreviated to N=. Up to 3000 NAME parameters can be specified in a single run of this utility. If the NAME parameter is used, the ALL=TRUE parameter should not be used.

TO='filename' or TO='prefix*'
: The full file name (including ownership userid) to which the file is to be restored. Usually, this parameter is omitted, in which case, the original name is used if possible. TO= may be abbreviated to T=. If the TO parameter is used, the ALL=TRUE parameter should not be used. An ending * or + indicates a prefix.

INPUT=n or IN=n
: The input unit number for reading the dump. The default is INPUT=1.

TAPFIL=n
: Specifies the tape file sequence number to restore from. The default is TAPFIL=1.

EOFTXT='xxxxx'
: Specifies a character string (up to 80 chars) which indicates end of data on the input unit. When this parameter is specified (and the string is nonblank), the program keeps reading after EOF (used for restoring from MUSIC/SP service tape).

FIXUP='x'
: Specifies a fixup character, *x*, to be used for generating an alternate name if the receiving file already exists (and should not be replaced). The fixup character is added to the end of the TO name, and the resulting name is tried. At most, 3 such retries are done per file. The default is FIXUP='$'. To prevent fixups, specify FIXUP=' '.

ALL=TRUE
: Causes all files to be restored. Default is ALL=FALSE. Do not use the NAME or TO parameters if ALL=TRUE is used.

RLSE=TRUE
: Releases unused space as files are restored. Default is RLSE=FALSE.

REPL=TRUE
: Causes the TO file to be replaced if it already exists on the library. Default is

REPL=FALSE.

LIST=F — Suppresses messages for successful restores. Message is still issued if name FIXUP was done or check sum was incorrect. Default is LIST=T.

SETUI=TRUE — Causes the program to restore certain file attributes. The attributes affected by this option are: usage count, use dates, code of creator, code of last writer. The default is SETUI=FALSE (i.e., do not restore these attributes).

FIXUCR=TRUE — With this option, MFREST temporarily creates a UCR (User Control Record) for the target code if one does not exist, or temporarily increases the UCR limits if necessary. This enables the file to be restored. After the file is restored, the temporary UCR add or change is undone. The default is FIXUCR=FALSE, in which case a file error 40 or 41 would occur if there is no UCR or the UCR limit is reached. Note that the code running MFREST must still have an adequate UCR, even if FIXUCR=TRUE is used, in order to allow creation of a temporary file for the restore. The UCR fixup is done when the temporary file is closed and renamed to the target file.

SETBUP=TRUE,PERIOD=n,CURMBN=m

The parameter SETBUP=TRUE causes each restored file to be marked as backed up, and the file's backup number to be set to a nonzero value within the range specified by PERIOD=n and CURMBN=m.

The default is SETBUP=FALSE, in which case PERIOD and CURMBN are ignored, and each restored file appears as a new or modified file and its backup number is set to zero. This means that the incremental archive utility (MFARCH) will dump the file to tape on its next run.

SETBUP=TRUE is useful when many files are being restored, to prevent MFARCH from archiving the files all at once. PERIOD=n specifies the MFARCH period and CURMBN=m specifies the current master backup number. The restored files are assigned backup numbers evenly distributed among the n numbers preceding m (0 is skipped and 255 is considered to precede 1.) For example, PERIOD=4, CURMBN=1 spreads the backup numbers among 255, 254, 253, and 252. PERIOD=5, CURMBN=3 spreads them over 2, 1, 255, 254, and 253. The defaults are PERIOD=12, CURMBN=1.

PERIOD=n — Used with SETBUP=T, it specifies the MFARCH period number.

CURMBN=m — Used with SETBUP=T, it specifies the current master backup number.

MBNFIL='filename'

Specifies the full file name of the file containing the master backup number (max 22 chars). This is used only if UPDMBN=T. Default is MBNFIL='$PGM:MASTER.BACKUP.NUM'.

Important note: The parameters INPUT, TAPFIL, FIXUP, ALL, RLSE, REPL, SETUI, FIXUCR, SETBUP, PERIOD, and CURMBN may only be specified on the *first* parameter statement, but they automatically apply to all the other parameter statements also.

Examples of parameter statements:

```
NAME='ABCD:FILE1',REPL=TRUE
NAME='XY*',TO='PQ*'
NAME='CCDE:PROGA',TO='CCRM:COPY.PROGA'
NAME='ABCD:PAY*',TO='ABCD:OLDPAY*'
```

This restores ABCD:FILE1 to ABCD:FILE1, all codes XY... to PQ..., CCDE:PROGA to CCRM:COPY.PROGA, and all files ABCD:PAY... to ABCD:OLDPAY... Any existing files will be replaced, since REPL=TRUE is specified on the first parameter statement.

The following example restores three specific files, with replacement of existing files:

```
/FILE 1 TAPE VOL(ARCVOL) BLK(4000) LRECL(80) SHR
/INCLUDE MFREST
REPL=TRUE,NAME='ABCD:PROG1.S'
NAME='ABCD:PROG1.LKED'
NAME='XY25:SAMPLE'
```

The following example restores all files for code DEFG, but adds XX at the beginning of each name:

```
/FILE 1 TAPE VOL(TAP123) BLK(3200) LRECL(80) SHR
/INCLUDE MFREST
NAME='DEFG:*',TO='DEFG:XX*'
```

Privileges Required: LSCAN, FILES, and MAINT.

MOVEPDS: Restore Files from OS IEHMOVE Tape

This program reads a sequential card-image file (normally on tape) which is a dump, produced by the MVS utility 'IEHMOVE', of a partitioned data set (PDS). It outputs the PDS members either as a single sequential file with each member preceded by a ./ ADD statement, or as individual save files. The record format of the original PDS must be fixed (i.e., F or FB or FBA, etc.). It must not be V or U. The original PDS must not contain MVS load modules.

Control Statements:

```
/FILE 1 ... (The input tape or file, LRECL=80)
/FILE 2 ... (The output file, needed only if DOTSL=T)
/INCLUDE MOVEPDS
parameters (separated by commas)
member names (needed only if SELECT=T)
```

Parameters:

FILE=n	The file sequence number (1,2,...) of the input, if reading from a tape. (Files are counted as on a non-labeled tape.) Default is FILE=1.
DOTSL=T or F	If DOTSL=T, output is to a single sequential file on unit 2, with each member preceded by "./ ADD NAME=xxxxxxxx". Output LRECL=80. "./ ALIAS NAME=yyyyyyyy" statements may also be put out. If DOTSL=F (the default), each member is put out as a separate save file (alias names are ignored.)
SELECT=T or F	If SELECT=T, the member names to be restored follow the parameters, one statement starting in column 1. They may be in any order. The maximum number of names is 500. These names must be actual member names, not alias names. If SELECT=F (the default), all members are restored.

The following options PREFIX, SUFFIX, REPL, PUBL, and LRECL only apply if DOTSL=F.

PREFIX='xxx'	Prefix characters to be placed before the member name when forming the save file name. Default is no prefix characters.
SUFFIX='xxx'	Suffix characters to be placed after the member name when forming the save file name. Default is no suffix characters.
	Note: The total number of prefix and suffix characters must not exceed 9 (or 14 if PREFIX starts with "CODE:").
REPL=T or F	Replace an existing save file. Default is F.
PUBL=T or F	Create public save files. Default is F.
LRECL=n	Logical record length for the save files. Truncation or blank padding will be done. The default is the logical record length of the original PDS.

Privileges Required: None.

NEWINDEX: Create a New Save Library Index

This program creates a new save library index data set of a different size.

Warning: There must be no other users or BTRMS on the system (and no batch jobs) when this program is run.

This program copies all save library index entries from the production index to a new index, rehashing the names according to the size of the new index. This is needed in order to change the number of segments in the index, or the number of blocks per segment.

The new index is defined on unit 1 as:

```
/FILE 1 UDS(%xxxxxxx) VOL(xxxxxx) OLD
```

Unit 2 is a work file, logical record length is 80, with room for at least as many records as there are entries in the index. Units 3 and 4 are sort work files, each at least as big as unit 2.

The privileges that are required are: VIP, DREAD, and LSCAN.

Return codes:

```
  0 success
  1 failure
 >1 system error
```

Notes:

1. The new index must be formatted (by the format utility) with 0's, i.e. using the option data=z00 in format, before this program is run.

2. When you initialize the new index by the ULINIT utility, use the "NSEG=n" parameter to specify the desired number of segments. The number of segments must be a prime number, e.g. 7, 11, 13, 17, 19, 23 or 29. ULINIT must be run after formatting and before this program. When increasing the index

data set size, you should also increase the number of segments (the NSEG=n parameter for ULINIT), so that the size of each segment stays about the same. Otherwise /LIB and /DIR commands will be slower. The number of blocks per segment is approximately m/(n+1), where m is the number of blocks in the entire data set and n is the number of segments.

3. The MFINDX subroutine used must be the version that puts the index entry into common block /MFINEN/. (MFINDX distributed with music release 5.2 does not do this, but the one in music/sp 1.0 and later releases does.)

4. If this program must be rerun, the new index must be reformatted to zeros, and ULINIT run, before the rerun.

5. This program may take a long time to run.

6. After the entries have been successfully moved, the system catalog must be changed to point to the new index and music re-IPL'd before any changes are made to save library files, otherwise the new index will not reflect the changes.

7. See also file $PGM:NEWINDEX.DOC.

Important: Verify that all programs worked before changing the catalog to use the new index.

NOWDOL: User Time Accounting Update

This program reads the accounting statements produced by the accounting dump program (ACTDMP) and adds processing unit charges plus connect charge for each user code to the NOW$ field of the user's Code Table record. If the installation wishes to maintain the NOW$ fields, the NOWDOL program should be run daily using the output of the program ACTDMP.

The deck set up for executing NOWDOL is as follows:

```
/INCLUDE NOWDOL
```

A user will be prevented from signing on MUSIC if the user's NOW$ value exceeds the amount allocated to the user with the MAX$ parameter of the CODUPD program. The CODUPD program can be used to increase the user's MAX$ limit.

A user may use the PRINT$ command of the User Profile Program (PROFIL) to inspect the MAX$ and NOW$ fields. The system administrator may also inspect these fields by using the '$' option on the GET command of CODUPD. The NOW$ field is initialized to zero when a code is authorized.

Privileges Required: MAINT and CODES.

NUCGEN: Nucleus Generation Utility

This utility is run whenever changes are required to the MUSIC nucleus. This nucleus contains the main controlling program for the entire MUSIC system. This controlling program is made up of more that 50 modules. It also contains the I/O devices and installation-dependent parameters. When MUSIC is IPLed, the nucleus is read into main storage where it resides when MUSIC is operational. Changes to the nucleus

made by this utility take effect the next time MUSIC is IPLed.

The NUCGEN utility produces a sequential file that contains a job stream that will create a new nucleus when run. This sequential file can be written on tape. This tape can then be IPLed which will run the job stream to put on the new nucleus. If this nucleus does not work correctly then you can go back to using the tape that contained the previous copy of the nucleus by IPLing from that one.

The NUCGEN utility can also be run so that the new nucleus is written to disk without using a tape. This involves running the SYSGEN1 utility using as input the sequential file produced by this NUCGEN utility. While this process is faster than using a tape, it does not allow you to go back to a copy of the previous nucleus should the new one cause MUSIC not to operate correctly.

The ADMIN facility described in the *MUSIC Administrator's Guide* uses the method that bypasses the use of tape. A tape backup copy of the existing nucleus can be made first by selecting one of the menu options.

When using the ADMIN facility you only need be familiar with the device specification statements and options rather how to run the NUCGEN utility itself. Before we describe those parameters, we will describe how to run the NUCGEN utility directly without using the ADMIN facility.

The following is the way to run NUCGEN to produce a tape that contains a job stream that will form a new MUSIC nucleus The first time you run this utility you specify the file $GEN:NUCLEUS as input. It creates an output tape called OURGEN that will be used in a later step. This OURGEN tape contains all the object decks which form the MUSIC nucleus. (Subsequent runs of the NUCGEN utility can use the output of the last run as input.) For more information about modifying MUSIC's nucleus, see *Chapter 18 - System Internals*.

Prepare and run the following job using the options and device specifications as detailed below. Consult the printed output to verify that this program has run successfully. Omit the /ID and /END commands if the job is submitted from the terminal. The file NUCGEN.SAMPLE contains sample control statements for this step.

```
/ID                 $000 000 060 999 999
/FILE 1 NAME($GEN:NUCLEUS) SHR
/FILE 2 TAPE BLK(80) LRECL(80) VOL(OURGEN)
/FILE 4 NAME($GEN:NUCMAP) NEW(REPL)
/INCLUDE *COM:NUCGEN
/INCLUDE *COM:MDLSYG
.... options ....        (see below)
.. device statements ..  (see below)
DEVEND
... replacement nucleus object modules (optional)
/END
```

Notes:

1. The /FILE 2 statement can be replaced with the following if SYSGEN1 will be used.

   ```
   /FILE 2 NAME($GEN:NUC) NEW(REPL) SP(500)
   ```

2. If a replacement object module is specified more than once, the first one is used.

Options

Separate all options from each other by commas. The following gives a sample set of options. Each option is explained below.

```
IN=1,OUT=2,XMAP=4,
SYSRES='160',CONSOL='01F',PRINTR='000',
CFACT=100,3,
REGION=3000,0,
BPOOL=300,
MAXTRC=15,MAXCOR=65536,MAXRRS=392,RAMDSK=512,ULMAPS=1,
LEVEL='MUSIC',SIGNON='MUSIC/SP,'
```

IN — Specifies the unit number of the input file. Use IN=1 in this case.

OUT — Specifies the unit number of the output file. Use OUT=2 unless the nucleus is to be punched on cards in which case use OUT=7. The output tape will always have one file.

TAPFIL — Specifies which file of the input tape is to be used. If input is from a file, then use TAPFIL=1. If not given, then TAPFIL=1 is assumed.

XMAP — XMAP=4 specifies that a copy of the storage map will be produced on unit 4. This storage map shows the location of the modules which form the MUSIC nucleus. A /FILE statement for unit 4 must then be provided. The default of XMAP=0 will not produce a copy of the map on a data set.

SYSRES — Specifies the disk address where the nucleus is to be written. (This is not the address of the starter system pack.) Normally this is the address of volume MUSICX. Make sure that you enclose the address in single quotation marks as in the example above.

CONSOL — Specifies the device address of the console to be used when the nucleus is written on disk.

PRINTR — Specifies the device address of the printer to be used when the nucleus is written to disk. If specified as PRINTR='000', then no storage map is printed when the nucleus is written to disk.

CFACT — Specifies two charging factor numbers. Both are used to calculate the number of service units a job uses. The first number gives processor time component. It is recommended that a number be chosen that is approximately 100 times the MIP rate of your processing unit. MIP means million instructions per second. It is not a precise number but is useful to determine a reasonable number for this option. If you upgrade to a faster processor you then need only adjust this number and users will see their jobs taking approximately the same number of service units. Some examples might be: 20 for 4331M1, 42 for 4331M2, 70 for 4361M3, 100 for 4361M4, 160 for 4341M12, 230 for 4381M1, 300 for 4381M2, 63 for 9370M20, M40, 92 for 9370M60, 230 for 9370M90, 200 for 4381-11, 300 for 4381-12, 400 for 4381-13.

The second number is used to factor in the I/O charge. The I/O charge is done by multiplying the number of SIO instructions by this factor and then dividing by 300. Paging SIOs are not counted. A recommended number is 3 meaning that 100 SIOs will result in a charge of one service unit.

REGION — This option takes two numbers. The first specifies the maximum size of the user region. Valid values range from 256 to 3000 meaning 256K to 3000K. The number must be a multiple of 4. The default is REGION=1024,0. Note that if the value for REGION is too small, a large program like MAIL will not be able to run.

The second number specifies the region size value to be used when the user specifies /SYS REG=MAX. The use of REG=MAX should be discouraged. This parameter is provided for compatibility with the IUP version of MUSIC. For example, these sites can install MUSIC/SP and increase the user region maximum from 256K to 2000K without causing

large charges for users who used REG=MAX when they really just wanted 256K not the new larger 2000K. If the number is set to 0, then users will not be allowed to use MAX specification. The default for the second number is 0.

BPOOL Specifies the number of buffers to be allocated in the buffer pool. This pool is used to buffer terminal I/O between the user program and the terminal itself. Each buffer is 516 bytes long. Allocate at least 4 buffers for each terminal control block (TCB). The number of TCBs is the number of terminals (including BTRMs) plus the number of extra TCBs allocated for multi-session (see the XSES parameter). The utility program BPOOL can be used during production to monitor the usage of the buffer pool. If the number used consistently approaches the number allocated this parameter should be increased.

MAXTRC Specifies the maximum amount of main storage MUSIC will use for its trace table. This number is expressed in K (K=1024) bytes. Recommended size is in the range 10 to 20.

MAXCOR Specifies the maximum amount of main storage that MUSIC will use. This number is expressed in K (K=1024) bytes and must be a multiple of 2. If the specified size exceeds that of main storage then this option will be ignored. It is recommended that the maximum value of 65536 be used so that MUSIC will use all the storage available to it as defined in its VM directory.

MAXRRS=n Maximum real region size in K. This must be 72 to 952. The default is 272. (*n*+8 should be a multiple of 40 otherwise it is rounded down.) You may need to increase the size of the swap data sets if a value greater than 272 is used, and a lot of jobs run with large regions. In the messages from the NUCGEN program, a value of -1 for MAXRRS indicates the default value.

LEVEL Specifies an 8-character identification to be printed when this nucleus is used. The current date will automatically be added to this identification. The MUSIC dump print program (PRDUMP) will also print the nucleus identification in MUSIC main storage dumps.

SIGNON Specifies a sign-on message (max 50 chars) to be displayed when a user connects to your production MUSIC system. You may use this field to show your company name at sign-on time. The system will add the words " SIGN ON." to the end of the message specified by this option, and add "*" at the front.

XSES=n Specifies the number of extra terminal control blocks (TCBs) to be allocated for multi-session support. If this parameter is omitted, the number of extra TCBs is 50 per cent of the number of terminals defined. If multi-session is heavily used, a larger number of extra TCBs may be needed, however the total number of TCBs, including terminals, BTRMs, and extras, will not exceed 999 regardless of the number specified. If you do not wish to use the multi-session feature you should specify XSES=0 and set the profile limit for extra sessions to zero for all users. In the messages from the NUCGEN program, a value of -1 for XSES indicates the default value. (The XSES parameter must be omitted in MUSIC/SP Release 1.0. It is valid in Releases 1.1 and higher.)

RAMDSK=n Specifies the amount of memory to be reserved for a RAM disk. It is expressed in kilobytes (1K = 1024 bytes). During system startup, files are loaded from disk into this area. The file $PGM:RAMDISK.LIST contains a list of the files that should be loaded. Typically they should be high usage, read only files such as command files, load modules, REXX procedures, macros, and panel definitions. Considerable saving in I/O overhead can be obtained by having files in RAM, since subsequent read access requires no I/O operations at all. If a file is changed it is automatically removed from RAM and the system reverts to using the DISK version of the file until the next time that RAM disk is loaded.

The amount of space you specify depends on what files you want to load into RAM. It must

be at least as large as the files plus a few K for an index. The RAMREP utility can be used to monitor RAM disk usage. This will help you adjust the file selection and space requirments to your own specific needs.

See also the RAMDLD utility.

ULMAPS=n This field specifies the maximum size (in number of K bytes) of the memory area used for caching the space bit maps of the library data sets (SYS1.MUSIC.ULm). Specify ULMAPS=1 to cache all bitmaps in memory. Specify 0 to use no bitmap caching. For best performance, 1 is recommended - unless your library is unusually large AND memory is scarce. ULMAPS=1 eliminates almost all bitmap reads from disk. For more information, run the MAPMEM and COUNTS utilities. If ULMAPS is not specified, ULMAPS=0 is assumed.

MPLLIM=n This parameter sets the limiting value for the maximum multi-programming level (the MAXMPL number reported by the SSTAT program). The standard and default value is MPLLIM=10. On some MUSIC/SP systems with large amounts of memory, specifying a value for MPLLIM greater than 10 may give higher performance. You may try a value between 10 and 30, for example. Note that a larger MAXMPL also results in a smaller time slice length. You can set the MPLLIM value via ADMIN 4 10 5.

Setting Up Device Statements

These statements are used to specify the addresses to be used by the production MUSIC nucleus. Each statement may contain 2 or 3 fields called *Devtype*, *Address*, *Options*. Devtype starts in column 1 and each field is separated from each other by blanks. Commas MUST be used to separate items in the options field from each other. An address range can be specified as shown in the following example:

```
TAPE 180-183 9TRK
```

A sample set of device statements is shown below. Lines starting with an asterisk (*) are comments and are ignored. The device statements can appear in any order.

```
* UNIT RECORD DEVICES:
2540 00C READER
2540 00D PUNCH
1403 00E PRINTER
1052 01F CONSOLE
* DISK DEVICES:
3350 148-14B
F512 260-261
3380 270-272
* TAPE DRIVES:
TAPE 180-183 9TRK
* PSEUDO TERMINALS THAT HAVE BACKGROUND JOBS STARTED AT IPL TIME
BTRM 005
BTRM 010-019
* TERMINAL DEVICES:
TTY 020-03F DIALUP,300
TTY 040-047 DIRECT,1200
3270 0C0-0CF DIRECT
* AUTOMATICALLY SIGNED ON TO BE USED FOR AUX PRINTERS
TTY 060 DIRECT,1200,SIGNON
3270 0F0 DIRECT,SIGNON
DEVEND
```

The following describes each device type in detail. Some device type names also apply to other devices of the same nature. For example, a 3211 printer is generated using the device type of 1403.

If you do not have a card punch device, omit the record "2540 xxx PUNCH", and similarly for the card reader. Disk and tape devices can have channel addresses from 1 to F. **All other devices must be on channel 0.** BTRM devices must also be on channel 0. The only exception is that 3270 terminals can be on channels 1 through 15 if no tapes or disks are also on that channel. (The channel address is the first digit of the 3-digit device address.) The total number of disk and tape devices must not exceed 64.

Specifying Unit Record Devices

2540 This device type is used to specify any supported card reader or punch. You must specify READER or PUNCH in the options field as appropriate. Only one reader and one punch may be defined. Note: the extra virtual punches, readers, and printers, used with the SUBMIT, AUTOPR, VMREAD, and VMPRINT, must not be defined here. Use an address of 000 if you do not have a reader or punch.

1403 This device type is used to specify any supported batch line printer. You must specify PRINTER in the options field. Only one printer can be defined.

1052 This device type is used to specify any supported console. Only one console may be defined. (This address can be automatically changed at IPL time. See *Chapter 3 - Loading the System* for details.)

Specifying Disk and Tape

2314 This device type applies to any device in the 2314 class such as 2314 or 2319.

3330 This device type applies to any device in the 3330 class such as 3330 or 3333. No distinction is required between Model 1, 2 or 11.

3340 This device type is used to specify a 3340 or 3344 type of disk. No distinction is required between Model 35, 70 and 70F. Define each logical 3344 volume separately as a 3340 device.

2305 This device type is used to specify a 2305 fixed-head disk device. The options field must specify MOD1 or MOD2 as appropriate. Since multiple requesting is not supported on MUSIC, the device address should end with a 0 or 8.

3350 This device type is used only when the 3350 DASD device is working in native mode. (Use the 3330 specification for this device if it is in 3330 compatibility mode.)

F512 FBA (fixed block architecture) DASD device, such as 3310, 3370, 9332, or 9335.

3375 3375 disk device.

3380 3380 disk device. Refer to the topic "3380 Model AA4 Addressing Notes" in Chapter 1 for cautions about 3380 device addressing.

3390 3390 disk device.

9345 9345 disk device.

TAPE This device type is used to specify any supported magnetic tape device. The options field must be 7TRK, 9TRK, or 8809 (the IBM 8809 tape drive), as appropriate. Not more than 4 tape

devices can be used in any one user job.

See the topic "Tape Drives on IBM 9371 Processor" in Chapter 1 for more information.

Special Specifications

TERM This specification is used to give the preparatory I/O command to the telecommunications control unit before a line is enabled. This specification is required if using an IBM 2702 communications control unit. It may be optionally used with the Integrated Communications Adapter (ICA) on the S/370 Model 125 processing unit. In all other cases, this specification is meaningless.

For 2702 units, specify the 2 character SAD command followed by 2 zeros in the address portion of this card. For example, use 1700 for a SAD command of 1. (The SAD identifications for you particular 2702 unit can be obtained from your local hardware support personnel.)

For the ICA on the Model 125, specify the 2 characters 2B followed by the 2 character *set line mode* options in the address field of this card. Refer to the *S/370 Model 125 Functional Characteristics* manual (GA33-1506) for the meanings of these options. Examples are 2BDC, 2B9C. The line mode may also be set at IMPL time. Refer to Chapter 1, topic "Transmission Control Units" at the beginning of this publication for details.

The options field of this card must be of the form CTLn, where n is a number from 1 to 9. For example, you can use CTL3. This CTLn option is used to relate to the terminal specification cards. When the TERM specification is required, then all affected terminal device cards must have the corresponding CTLn specified in their options field. An example is shown below:

```
TERM 2BDC CTL1
2741 030-3F CTL1
```

Up to 9 TERM cards may be specified defining the 9 possible CTLn options. The device re-configuration of MUSIC at IPL time allows the specification of one 4-character value that applies to all terminal types defined at that time. This 4-character value is the same as that which can be specified in the address portion of the TERM card just described.

Specifying Terminals

IBM This specification is used to denote dialup IBM terminals such as 2741, 3767, 1050 and CMCST devices. MUSIC will automatically handle any of these terminals on the same line. (If 3767 terminals are used at 300 baud, then those 300 baud lines cannot be used for the 2741, 1050 and CMCST devices.) Specify DIALUP in the options field. The line speed should be specified as 134. Lines defined as 300 baud for 3767 terminals should use the line speed option of 300.

3767
1050 These specifications must be used with directly connected IBM-type terminals. The options field must specify DIRECT. A numeric line speed option should also be given in the options field.

3270 This specifies any terminal in the 3270 product line or a TTY terminal connected through a protocol convertor that emulates a 3270 terminal. They must be locally attached to the processing unit or be specified as SPECIAL in the VM directory for MUSIC.

Caution: Make sure that the tens digit of the 3270 address is not used by any other type of

terminal. For example, if a 3270 device has address 0E2, make sure that no other non-3270 device has an address in the range 0E0 to 0EF.

TTY This option specifies any of the asynchronous ASCII (TTY) class of terminals. It is also used for the IBM 3101 terminal and personal computers connected to MUSIC via asynchronous lines which do not come through a protocol converter. (Ones that do come in through a protocol converter look like 3270 terminals to MUSIC and must be specified as such.) Auxiliary ASCII printers should also be designated as TTYs.

The option field may specify either DIRECT or DIALUP depending on the type of connection. The speed option should be used in order to ensure correct operation of these devices.

Normally the RETURN key on ASCII terminals is used as the end of line character. You can use the option RNA to specify that this RETURN key is not an active end of line character. When RNA is used, RETURN characters are ignored. The transmission control unit and MUSIC must both be set whether to accept the RETURN as end of line or not accept it. This specification can be done on a line by line basis.

BTRM No actual terminal will be attached to this address. This specification causes the system to create the appropriate control blocks, as if a terminal were there, and automatically start a background program when the system is IPLed. The system will attempt to automatically sign on the code $MONsss where the subcode, *sss*, is the device address of the BTRM. The background program started is the AUTOPROG specified in the profile for the code. This facility is used by programs such as AUTOSUB, AUTOPR, and VMREAD, which run as background tasks without a terminal. For further details see Chapter 22, topic "Defining BTRMs and Auto-Sign-on". BTRM device addresses must be on channel 0.

Note: The distributed system assumes there are BTRMs defined on addresses 005 and 010 through 019 for use by the SYSLOG, AUTOPR, VMREADX, RDMAILER and the ADMIN facility. If you do not have them defined, the programs will not function.

Terminal Options

The following options can be specified on the terminal device statements.

DIALUP

For a TTY device, DIALUP should be specified if the terminal is connected to the communications controller via a switched line. This usually means a modem connection that can be made and broken by the user. Virtual TTY ports defined in MUSIC's VM directory as SPECIAL should also have the DIALUP option specified. TTY devices used as auxiliary printers should always be defined as DIALUP regardless of the connection.

If the DIALUP option is specified for a 3270, MUSIC will issue a host reset command at sign off time, if the terminal is really a TTY device connected through a protocol convertor. (See 7171 option).

DIRECT

The DIRECT option should be specified for TTY devices that are connected directly to the communications controller, or connected in such a way that they appear directly connected to the controller.

DIRECT should be specified for real 3270 terminals and for emulated 3270 terminals when you do not wish a host reset to automatically be issued at sign off.

Speed Option

This speed option should be given on all terminal specifications except the 3270. It is used by the terminal handler to determine the number of idles to send to each terminal as well as other speed-dependent functions.

nnnn The allowable options correspond to the line speed in bits per sec (baud). They are: 110, 134, 300, 600, 1200, 1800, 2000, 2400.

AUTOSPEED The line speed is determined dynamically by the system at time of the initial line connection. This option requires that the transmission control unit have the speed detection feature. See the topic "Transmission Control Units" in Chapter 1 of this publication. The DIALUP option must also be specified for these lines.

SIGNON

The SIGNON option indicates that once the line has been successfully enabled, the terminal is to be automatically signed on and a program started. This option is usually specified for terminals which are to function as auxiliary printers using the AUTOPR program. The system will attempt to automatically sign on the code $MONsss where the subcode, *sss*, is the device address of the terminal. The background program started is the AUTOPROG specified in the profile for the code. For further details see Chapter 22, topic "Defining BTRMs and Auto-Sign-on".

7171

The 7171 option indicates that the terminal is actually an ASCII terminal connected through a 7171 protocol convertor or through something that is compatible with the 7171 such as the 9370 ASCII Subsystem.

When specified on a TTY device definition, it indicates that 7171 ASCII transparent mode should be used. In this mode, protocol conversion is bypassed and the terminal functions as a native ASCII device.

When specified on a 3270 device device definition, normal protocol conversion is done and the terminal functions as a 3270 device. However, the TOTTY and TO3270 can be used to switch between 3270 emulation mode and ASCII transparent mode. This allows PCWS file transfer and number of other MUSIC to PC connectivity features to work.

If MUSIC is running under VM this option need not be specified. When a terminal connects from VM, MUSIC determines if it is connected through a protocol convertor by asking VM. The VM module DMKRIO should specify "FEATURE=EMUL3270" on the RDEVICE statements for these terminals, so VM will provide the correct information.

Privileges Required: LSCAN and FILES.

PRDUMP: Storage Dump Print

This program is designed to print storage dumps taken by the MUSIC stand-alone storage dump program. The dump program resides on the MUSIC1 disk pack. It is run by performing an IPL from this pack. The dump program dumps all of main storage onto a preallocated data set on the MUSIC1 pack. When MUSIC is running again, the storage dump print routine is run from batch to print the dump. This program, in addition to printing the contents of main storage in standard dump format, formats many of the MUSIC internal

tables and control blocks to enable quick analysis of the dump.

This program can also be run from a terminal. In this case it allows for the interactive inspection of the dump.

The storage dump print program is run as follows:

```
/FILE 1 UDS($PGM$DMP) VOL(MUSIC1) SHR
/INCLUDE PRDUMP
.....parameters   (if batch)
```

The file statement for unit 1 should point to the installation dump data set.

The parameters that may be specified on batch are:

MUSIC=TRUE/FALSE	If TRUE, format MUSIC status information
TRACET=TRUE/FALSE	If TRUE, format trace table
SELCHQ=TRUE/FALSE	If TRUE, format selector channel queues
NT=nnn	Number of trace table entries to format. A value of zero indicates that all entries are to be formatted.
TCB=TRUE/FALSE	If TRUE, format TCBs

If more than one parameter is to be specified, they should be separated by commas (MUSIC FORTRAN NAMELIST format).

The default parameters are:

```
MUSIC=TRUE,TRACET=TRUE,SELCHQ=TRUE,NT=500,TCB=TRUE
```

The code under which this job is run must be able to access the dump data set.

Privileges Required: LSCAN and FILES.

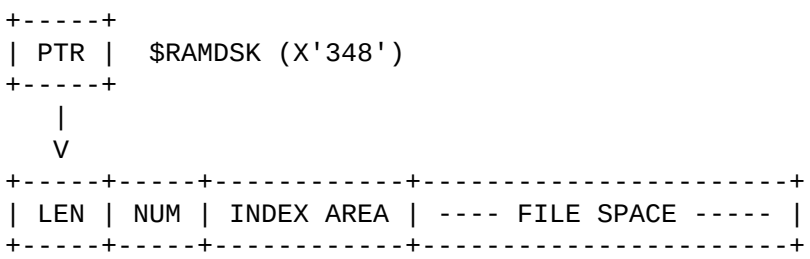
RAMDLD: Load MUSIC RAM Disk

To improve system performance it is possible to define an area of memory to be used as a RAM disk area. During system initialization files are brought from disk into this area. Subsequent access to these files requires no I/O operations. If the right files are chosen, a significant reduction can be made in the I/O overhead incurred in accessing high usage files. The RAM disk is read only. If a file in the RAM disk is changed the system automatically removes it from RAM and uses the modified version on disk until the next time RAM is loaded.

RAMDLD loads the RAM disk area from the Save Library. It is usually called by JOBONE to do this at system startup time, but it can also be run on its own to re-load the RAM disk while the system is running. VIP must be set ON to run this program. You can run the program by issuing the RAMDLD command. Note that reloading the RAM disk on a running system may cause a few users to experience temporary problems if they happen to be accessing an old RAM file at that instant.

The file $PGM:RAMDISK.LIST contains a list of files that are loaded by this program. If the program runs out of room in the RAM disk area it simply gives up. The size of this area is defined by the RAMDSK parameter in the NUCGEN. Note that once loaded this is a read-only RAM disk, so good candidates for this area are high access read only type files, like the execution files and load modules for common commands, programs, and utilities.

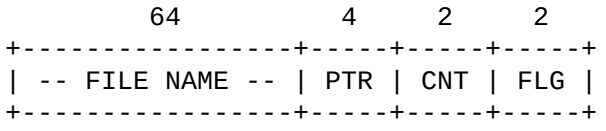
Location X'348' contains the pointer to the RAM disk area. If this is zero the program does nothing. The first word of the area contains the length of the entire area. This is used to calculate the number of index entries. The rest of the space is used for files. the format of the RAM disk area is as follows.



LEN Total length of area

NUM Number of index entries

Each index entry has the following format:



PTR address of first block of file

CNT usage count

FLG FLAGS X'80' - RAM copy is invalid (file changed)
X'40' - File is in common index

$PGM:RAMDISK.LIST

The format of the $PGM:RAMDISK.LIST file is simply a list of fully qualified file names (userid, subdirectory path, and name) that are to be loaded into the RAM DISK area in memory. There should be only one file name per line and it must start in column 1. Lines starting with a "*" are ignored and can be used as comments.

RAMREP: Report on RAM Disk Usage

This program displays a report on how RAM disk is being used. It is run by entering the RAMREP command. The name, location, flags, and usage count of each file in the RAM disk are displayed.

The usage count is reset to 0 when the RAM disk is loaded. This is usually done at IPL time. This is deffe-rent to usage count that can be maintained with the file and viewed by the ATTRIB command. The count seen by the ATTRIB command is updated by 1 when the file is loaded into the RAM disk and is not updated when it is used from the RAM disk area.

Example:

```
>ramrep
```

```
RAM Disk at     B359A0
Total RAM Space=   512K
Total Files    =   114


Location Flags   Count   Size     Name
-----------------------------------------------------
B391E8    4000    64     1.0 K    $REX:REXX
B395E8    4000     5     1.0 K    $REX:REX
B39DE8    4000     5     1.0 K    $PGM:ATTRIB
B3A1E8    4000     5    20.0 K    $PGM:ATTRIB.LMOD
B3F7E8    4000    11     1.0 K    $PGM:CD
B3FBE8    4000    11     2.5 K    $PGM:CD.LMOD
B469E8    4000    74     1.0 K    $PGM:EDITOR
B471E8    4000     8    12.5 K    $PGM:FT
B4C1E8    4000     5     1.0 K    $PGM:OUTPUT
B4C5E8    4000     5   155.0 K    $PGM:OUTPUT.LMOD
B731E8    4000     2     1.0 K    $PGM:PRINT
B735E8    4000     2     7.0 K    $PGM:PRINT.LMOD
B751E8    4000     1     1.0 K    $PGM:PRTSCR
B755E8    4000     5    11.0 K    $PGM:PURGE
B781E8    4000     0     1.0 K    $PGM:RD
B785E8    4000     0     4.5 K    $PGM:RD.LMOD
B797E8    4000     1     1.0 K    $PGM:SUBMIT
B79BE8    4000     2     1.0 K    $PGM:TELL
B79FE8    4000     2     4.0 K    $PGM:TELL.LMOD
B7AFE8    4000    81     1.0 K    $PGM:VIEW
B7B3E8    4000     0     1.0 K    $PGM:VM
B7B7E8    4000     0    31.0 K    $PGM:VM.LMOD
B8B9E8    4000     2     1.0 K    $EML:GETMAIL
B8C1E8    4000     8     1.0 K    $EML:MAIL
B9CDE8    4000    31     1.0 K    $EML:VIEW.MAIL.COMI
B9D5E8    4000    17     1.0 K    $EMD:MAIL.AUTHOR
BA37E8    4000    69     1.0 K    $EDT:EDITOR.SHOWTEXT
BADDE8    4000     1    14.0 K    $INT:FLIB
BB15E8    4000     3    22.0 K    $INT:FSI

Total Space=   330.0K ,Total Usage Count=     457   ( 10.90% of opens)
```

RATE: Check System Load

The RATE program samples I/O, SVC, Paging and Swapping rates every ten seconds and displays the results on the screen. These numbers can be used as a good indicator of the instantaneous system load.

The COUNT field is the count since the last time the system was loaded. The RATE/SEC field is the rate averaged over the past 10 seconds. The RESPONSE TIME is not an average. It is the time it took for the system to wakeup the RATE program itself. The counters maintained by RATE are as follows.

I/O	- I/O interrupts
SVC	- SVC interrupts
PAGE WR	- Pages written to disk by demand paging
PAGE RD	- Pages read by demand paging
PLPA RD	- Page read from the PLPA

SWAP RD - Swap read operations
SWAP WR - Swap write operations

```
>rate

      SYSTEM RATES

  MON JAN 22, 1990     12.24.01
  SYSTEM LOADED AT      7.59.38

               COUNT  RATE/SEC
I/O           465372     7.000
SVC          4404351    38.000
PAGE WR        15360     1.200
PAGE RD        13817     1.100
PLPA RD        31662     2.100
SWAP RD         1012     0.200
SWAP WR         1564     0.210

RESPONSE TIME:  0.13 SEC
```

ROUTETABLE: Display Routing Table

This program displays the contents of the routing table (member $ROUTING). See the topic "Setting up the Routing Table" in *Chapter 8 - Job Submission* for information about the $ROUTING load library module.

Return Codes:

0 Normal End
1 Routing table not accessible

Privileges Required: none.

RTM: Response Time Monitor

The response time monitor (RTM) program is designed to run continuously when MUSIC is operating. It periodically samples the response times for the various MUSIC queues, number of active users, and VM overhead numbers and writes its output to a Save Library file. One file is created for each day. If the system is reloaded on the same day, the information is appended to the file previously used with an information line noting that the system was reloaded. At midnight, a new file is created for the new day. The information in the output records is described below.

Automatic Program Initiation Facility (BTRM)

MUSIC contains a facility for automatically initiating programs when the system is loaded (IPLed). These programs run as user programs with the exception that no physical terminal is associated with them. This facility is used to run the RTM program.

The "phantom" terminal is defined in the MUSIC nucleus generation (NUCGEN) data as a BTRM. The

program to be run ($PGM:RTM) is defined as an AUTOPROG for a subcode of the userid $MON. The subcode is the 3-character hexadecimal device address of the BTRM terminal. $PGM:RTM is automatically executed, and remains running, whenever MUSIC is loaded (except when the NOTERM option is used at IPL).

To cancel the RTM program, enter the console command "/HALT n", where n is the TCB number of the BTRM. The TCB number can be found by entering "WHOSON $MON" or "ENQTAB RTM" at a terminal, or "/FIND $MON" as a MUSIC operator console command. To restart the program, enter the console command "/RESET n".

For a general discussion of BTRMs, refer to the section "Defining BTRMs and Auto-Sign-on" in Chapter 22 - System Programming.

The files for the RTM program are $PGM:RTM*.

You can adjust the recording period by changing the value of the PERIOD parameter in file $PGM:RTM. The distributed value is PERIOD=10, meaning that a line of data is written to the recording file every 10 minutes.

How to Install the Response Time Monitor

1. The default period for RTM is 10 minutes (PERIOD=10). This means that the program wakes up every 10 minutes and writes a statistics record to the output file. If you wish to use a different period, edit file $PGM:RTM and change the number of minutes in the last line.

   Other options in $PGM:RTM include NEWNAM=t and KEEP=n. NEWNAM=T uses file name $MON:RTM.yyyymmdd for the daily log file, instead of $MON:Ryyddd. The KEEP=n option automatically deletes old RTM log files which are more than n days old. The default is KEEP=0, which means that no log files are deleted. KEEP=n is effective only if NEWNAM=T is also used.

2. Choose a device address xxx for the BTRM's phantom terminal. Note that this address does **not** correspond to a real terminal or to a virtual device defined in the VM directory for MUSIC. xxx is a 3-digit hexadecimal value. The first digit should be 0. Choose a device address that is not already defined in the MUSIC NUCGEN as a terminal, unit record device, or another BTRM.

3. Create a user id (code) for $MONxxx by running the CODUPD utility. The subcode xxx is the BTRM device address chosen above. The special password *NOLOGON prevents sign-on except as a BTRM, for security reasons; you may assign a different password if you wish. Enter the following command to CODUPD to create the id:

   ```
   ADD $MON SC(xxx) AUTOPROG($PGM:RTM) PW(*NOLOGON) -
   FILES LSCAN CREAD BATCH(0) -
   PRIME(NOLIMIT) NONPRIME(NOLIMIT) DEFTIME(NOLIMIT)
   ```

4. Define the BTRM to MUSIC, by adding the record

   ```
   BTRM xxx
   ```

   to the device statements for the nucleus generation (NUCGEN) utility. Usually the control statements for NUCGEN are stored in the file $GEN:NUCGEN.JOB. Run the NUCGEN utility and IPL from the tape it creates, in order to apply the new nucleus. Refer to the description of NUCGEN earlier in this chapter. You can also apply the nucleus by ADMIN 4 10 5 ("NUCGEN: update I/O config. & nucleus"). The two methods are equivalent.

RTM should start running when you next IPL MUSIC, with data records accumulating in file

$MON:Ryyddd, where yy is the year and ddd is the day of the year. If the RTM option of NEWNAM=T is included in $PGM:RTM then $MON:RTM.yyyymmdd is used.

Response Time Monitor - Output Records

The Response Time Monitor program writes records to the file $MON:Ryyddd (where yy is the year and ddd is the day of the year or $MON:RTM.yyyymmdd if NEWNAM=T). Each record represents a sampling period, normally 10 minutes. The fields in the 80-byte record are as follows:

Cols. 1-8 Time of day (HH.MM.SS) at the end of the period sampled.

Cols. 9-12 Number of users signed on at the end of the period sampled.

Cols. 13-18 Number of requests of type 1 (see below) during the period.

Cols. 19-25 Average response time, in seconds, for type 1 requests during the period.

Cols. 26-31 Number of requests of type 2 during the period.

Cols. 32-38 Average response time, in seconds, for type 2 requests during the period.

Cols. 39-44 Number of requests of type 3 during the period.

Cols. 45-51 Average response time, in seconds, for type 3 requests during the period.

Cols. 52-57 Number of requests of type 4 during the period.

Cols. 58-64 Average response time, in seconds, for type 4 requests during the period.

Cols. 65-71 The virtual CPU time used by MUSIC during the period, expressed as a percent of the length of the period. For example, if the period is 5 minutes and the reported percent is 40.00, then the virtual CPU time used by MUSIC was 0.40 * 300 = 120 seconds. The percent is the portion of the entire real machine used by the MUSIC virtual machine (not counting VM overhead). The field is zero if MUSIC is not running under VM.

Cols. 72-78 The total CPU time (virtual plus VM overhead) used by MUSIC during the period, expressed as a percent of the length of the period. The percent is the portion of the entire real machine used by the MUSIC virtual machine, including CPU overhead for privileged operations, CCW translation, etc. done by VM for MUSIC. The field is zero if MUSIC is not running under VM.

The percent VM overhead for MUSIC can be found by subtracting cols.65-71 from cols.72-78.

Cols. 79-85 The amount of time that the MUSIC user region was busy during the period, expressed as a percent of the period. This value is obtained from the "idle time" reported by the WAITS utility.

The types of requests are as follows:

Type 1 Request for the first time slice of a job (/RUN, /EXEC).

Type 2 Request for /ID, /CANCEL, /DISPLAY, /SAVE, /PURGE, etc.

Type 3 Request for a time slice when the job's immediately preceding time slice ended because of a

conversational read (or a call to the subroutine DELAY). This includes FSIO reads, such as done by the Editor in full-screen mode. It does not include spooled conversational reads, such as done by the Editor in INPUT mode on an ASCII terminal.

Type 4 Request for a time slice when the job's immediately preceding time slice went to completion. This is the case for non-conversational jobs, CPU-bound jobs, and batch jobs.

SETBFN: Set File Backup Numbers

This program resets the master backup number to 1 and sets the backup numbers in all files to values in the range 255, 254, ..., 255-n+1 (where n is the period specified below). At the same time, all files are marked as being backed up.

This program would be run if the incremental archive utility (MFARCH) could not fit all the new or modified files onto a single tape, and so could not terminate normally. This would be the case that if you have run MUSIC for some time without running the incremental backup or when you have made major changes to your system that caused an unusually large number of files to appear changed.

To run the utility, use the following statements:

```
/INCLUDE SETFBN
PERIOD=n,MBNFIL='filename'
```

If no parameters are needed, a blank line must be used after the /include.

PERIOD=n specifies the archive period number that will be used with MFARCH. See the description of MFARCH. Default is 12. Users of the ADMIN automatic maintenance facility should specify n as 10 as that is the period used there.

MBNFIL='filename'
specifies the name of the file containing the master backup number. Default is MBNFIL='$PGM:MASTER.BACKUP.NUM'. It is most unlikely that you need to specify this option.

Privileges Required: LSCAN, FILES, MAINT

SETBUF: Load Printer Buffer

This program can be used to load the buffers on the (real) system printer, if this has not already been done by VM. Normally it is not necessary to run SETBUF, since VM will do this function. The UCS or the FCB buffer may be loaded. The program must be run from batch.

Only one buffer may be loaded with each running of this program. If both an FCB and a TRAIN name are specified an error message will be displayed for the user. In this case neither buffer will be affected.

The system will issue the M308 console message when the UCS and/or FCB are loaded. The operator must respond with a "/REPLY 0" before the output will continue.

Two FCB buffers are supplied with MUSIC:

```
FCB Name    Description

 MUS6       SIX LINE TO THE INCH
 MUS8       EIGHT LINES TO THE INCH
```

UCS buffers images available are:

```
3203/1403  PRINTERS

AN    NORMAL AN ARRANGEMENT
HN    NORMAL HN ARRANGEMENT
PCAN  PREFERRED SET, AN
PCHN  PREFERRED SET, HN
QN    PL/I - 60 GRAPHICS
QNC   PL/I - 60 GRAPHICS
RN    FORTRAN, COBOL COMMERCIAL
YN    HIGH SPEED ALPHAMERIC
TN    TEXT PRINTING - 120 GRAPHICS
PN    PL/I PRINTING 60 GRAPHICS
SN    TEXT PRINTING - 84 GRAPHICS

3211 PRINTERS

A11   STANDARD COMMERCIAL
H11   STANDARD SCIENTIFIC
G11   ASCII
P11   PL/I
T11   TEXT PRINTING

3289 PRINTERS

F48   48 CHARACTER GRAPHICS
F64   64 CHARACTER GRAPHICS
F96   96 CHARACTER GRAPHICS
F127  127 CHARACTER GRAPHICS

3262 PRINTER

P48   48 CHARACTER EBCDIC
P52   52 CHARACTER AUSTRIA/GERMANY
P64   64/72 CHARACTER EBCDIC
P63   PREFERRED 64 CHARACTER EBCDIC
P96   96 CHARACTER EBCDIC
P116  116 CHARACTER FRENCH CANADIAN
P128  128 CHARACTER KATAKANA
```

A parm statement is read from unit 5 specifying which UCS or FCB are to be loaded. Also the option of FOLD may be specified. When fold is specified, it applies to either the UCS or FCB load.

The format of the parm statement is:

```
TRAIN='XXXX',FOLD=Y    ...  or  ...
FCB='YYYY',FOLD=Y
```

Where XXXX is the name of the print train (UCS) to be loaded. YYYY is the name of the FCB image to be loaded.

FOLD is either Y or not specified, the other options have no defaults and must be specified.

Example:

```
/INC SETBUF
TRAIN='PN',FOLD=Y
```

Privileges Required: LSCAN and MAINT.

SSTAT: System Status

This utility displays a screen that shows the system status. The screen is updated approximately every 10 seconds, or when a function or action key is pressed. Press F3 to exit this utility. This utility can be helpful in determining if there is a performance problem with your system.

```
------------------------------ System Status --------------- 09.24.03

Storage Size: 8192K  TCBs:   80  RCBS:  82  MAXMPL:  8  MAXRRS:  592K

Users:  18 Sessions:  23  Response Time: 0.02 (sec) CPU Usage: 49.68%

Running:   17      In Core:   16   Swapped:    1

 Active:    2       Queued:    0      Idle:   15

Total Pages: 5248K   Free Pages: 1052K


Activity       Count  Rate/Sec
--------   --------- ---------
I/O           747360   22.64
SVC          6841682  178.00
PAGE WR          406    0.00
PAGE RD         2884    0.00
PLPA RD        24639    0.00
SWAP WR            2    0.00
SWAP RD            3    0.00
```

Figure 17.6 - Sample Screen of the System Status program

Usage:

This program is available from the ADMIN facility option 1 13. You can also run this program by entering SSTAT when in *Go mode.

Privileges Required: CREAD

SUBLIB.GEN: Subroutine Library Creation

The MUSIC/SP system subroutine library consists of subprograms that are automatically available to programs written in Fortran, Assembler, PL/I, and other languages. The subroutine library is used by the MUSIC/SP loaders and the Linkage Editor. It is not used by /LOAD EXEC or /LOAD XMON.

The system subroutine library is contained in Save Library files with names of the form $SUB:SUBLIB.xxx. These files are created by the SUBLIB.GEN utility, which reads object modules as input and builds members and an index in the $SUB:SUBLIB.xxx file. Refer to the description of SUBLIB.GEN in the utilities chapter of this publication.

The standard subroutine library files are:

1. $SUB:SUBLIB.FG1 contains the library routines for the Fortran G1 compiler. The subprograms in this file are not available to OS-mode programs.

2. $SUB:SUBLIB.MUS contains the MUSIC-supplied system subroutines, plus other miscellaneous routines. The subprograms in this file are available to both non-OS-mode and OS-mode programs.

3. $SUB:SUBLIB.OS contains the library routines for VS Assembler, VS Fortran, PL/I, Cobol, and other OS-mode languages.

4. $SUB:SUBLIB.EXT is an optional library that could be used to contain subroutine packages installed locally at your site. For example, the large IMSL package of mathematical subroutines could be stored in this library. (Local subroutines could be added to $SUB:SUBLIB.MUS instead.) In the standard MUSIC/SP system, this file does not exist.

Each of the files $SUB:CREATE.xxx contains the control statements for the SUBLIB.GEN utility to create the corresponding subroutine library file. SUBLIB.GEN copies object modules, defined by /INCLUDE statements in the input, into the subroutine library file and builds an index of subprogram and entry point names.

Output, including informative messages, a list of the input modules, and an index listing sorted by name, is normally written to the file $SUB:LISTING.xxx, which you can keep for future reference. To have SUBLIB.GEN write messages and listings to the terminal (or to the printer if run on batch), use /FILE 6 PRT.

To add subprograms to the library, add appropriate /INCLUDE statements for the object modules to either $SUB:CREATE.MUS or $SUB:CREATE.OS and execute the file. This completely recreates the corresponding subroutine library file ($SUB:SUBLIB.MUS or $SUB:SUBLIB.OS). The previous contents of the file are deleted. You should normally add /INCLUDE statements at the end of the file. Routines added to $SUB:CREATE.OS are available only to programs running in MUSIC's OS simulation mode.

In order to replace a subroutine library file ($SUB:SUBLIB.xxx), there must be no user jobs running that access it. This usually means that no user jobs or BTRM jobs (userid $MON) should be running. To cancel all $MON jobs, enter the command "/FIND $MON" on the operator console. This displays the TCB numbers. Then enter "/HALT n" for each TCB number.

The general control statements for SUBLIB.GEN are:

```
/FILE 1 N(sublibfile) NEW(REPL) SPACE(nnnn)
/INCLUDE *COM:SUBLIB.GEN
INDEX=mmm
/INCLUDE objectfile
/INCLUDE objectfile
etc.
```

INDEX=mmm defines the number of 512-byte blocks to be reserved for the index at the beginning of the subroutine library file. Each index block can hold up to 51 routine or entry point names.

The following optional control statements may appear in the input, after the INDEX=mmm statement:

Comments Records with * in column 1 are ignored.

./ ADD NAME=xxxxxxxx
: This indicates the start of a member which may consist of more than 1 object module. The end of the member is indicated by the next ./ ENDUP or ./ ADD statement. Only the specified member name *xxxxxxxx* and any names given on ./ ALIAS statements are added to the index. If a ./ ADD statement is not in effect for an input object module, the csect and entry point names in the object module are added to the index and the output member contains only that object module. Use of ./ ADD allows grouping of several object modules into one member.

./ ALIAS NAME=xxxxxxxx
: This specifies an additional entry point name (alias) to be added to the index for the current member. Any number of ./ ALIAS statements can be used. They can appear anywhere after the ./ ADD statement and before the next ./ ADD or ./ ENDUP statement.

./ ENDUP Indicates the end of a member begun by a ./ ADD statement. A ./ ENDUP is also implied by the next ./ ADD.

/NOEP This statement indicates that the entry point names in the immediately following object module only are not to be added to the index.

Subroutine Library Considerations

In each job that requires the subroutine library, the subroutine library files to be used are defined by a /FILE statement with ddname SUBLIB. The Linkage Editor with /JOB MODE=OS also requires a /FILE statement with ddname SUBLIBOS. Each *xxx in the PDS parameter of these /FILE statements refers to the subroutine library file $SUB:SUBLIB.xxx. The files are searched in the order given. If a file does not exist or is not accessible, it is skipped. The following /FILE statements are automatically supplied by the /LOAD statement processor (module CTL):

For /LOAD FORTG1, /LOAD LOADER, and /LOAD LKED:

```
/FILE SUBLIB PDS(*FG1,*MUS,*EXT) DEF
```

For /LOAD LKED for /JOB MODE=OS:

```
/FILE SUBLIBOS PDS(*MUS,*OS,*EXT) DEF
```

For /LOAD ASM, /LOAD ASMLG and other OS-mode processors that use the loader:

```
/FILE SUBLIB PDS(*MUS,*OS,*EXT) DEF
```

Since these are generated with the DEF parameter, a job can use an overriding /FILE statement that may specify different libraries or a different search order.

For special applications, it is also possible to create a private subroutine library, by means of the SUBLIB.GEN utility. Then use of

```
/FILE USERLIB NAME(filename)
```

would cause the private library to be searched ahead of the standard library, for the current job only. Note that for ddname USERLIB, the NAME parameter is used rather than PDS. Here, *filename* is the name of the private subroutine library file, created as /FILE 1 by the SUBLIB.GEN utility.

Privileges Required: LSCAN and FILES.

SYSDATE: Display Nucleus Level and IPL Date/Time

SYSDATE displays the current date and time, the date and time that MUSIC was loaded (IPL), and the nucleus level and date. The nucleus level is the value specified by the LEVEL parameter of the NUCGEN utility. The nucleus date is the date NUCGEN was run.

Privileges Required: None.

SYSDMP: Relative Block Disk Utility

The SYSDMP utility can be used to inspect or alter the contents of a SDS or UDS data set by relative block number. (Refer to the DSKDMP utility if you want to do similar operations by absolute disk location.)

The data set is pointed to by either (a) DEB number or (b) data set name and volume name.

The parameters are as follows. Separate them by commas.

DEB=n Specifies DEB number.

DSN='dsname' Specifies the data set name.

VOL='volume' Specifies the disk volume name.

BLK=n or BLK=n,m or BLK=n,m,i
n is the starting block number (the first block of a data set is numbered 1). Default is n=1. *m* is the ending block number to be dumped. Default is *m=n*. *i* is the block number increment. Default is i=1. For example, BLK=3,10,2 dumps blocks 3, 5, 7, and 9.

BLKSIZ=n Length of each block (used in read and write). Default is DEB or data set blksize.

ADDR=n or DISPL=n
Displacement for start of dump display. Default is 0.

LEN=n or DMPLEN=n
Number of bytes to be dumped. Default is 32.

REP=n	Displacement for start of REP (bytes to be modified).
WRITE	Causes block to be written back to disk.
COPYTO=n	*n* is the unit number of a sequential file (default 0, i.e. no output). Each requested block is written as a record to the sequential file. This option is turned off by specifying COPYTO=0. For example, BLK=1,999,COPYTO=1 can be used to copy the records of a MUSIC catalog data set to a file.
HELP	Displays information on how to use the program.
END	Terminates the program.

Notes:

1. All parameter values are remembered between parameter reads, except REP, WRITE, and HELP. Also, the following are reset when a new DEB or data set is started: BLK=1,ADDR=0,DMPLEN=32.

2. A block is read only when a new data set is started, or when BLK or BLKSIZ is specified, or if the previous request was for more than one block.

3. Each of the parameters REP, WRITE, HELP, END must be used alone. They may not be used in combination with other parameters.

4. Replacement text (in hex) is read immediately after the REP specification. Commas may be used to separate bytes or groups of bytes. The text ends at the first blank.

5. Abbreviations: VOL: V, ADDR: A, DISPL: D, LEN: L, DMPLEN: DL.

6. A system data set may be specified as DSN='%XXXXXXX'.

Example:

```
DEB=226,BLK=15,DISPL=Z2A0,DMPLEN=64

V='MUSICX',DSN='%SCRATCH',BLK=318,L=512
```

Privileges Required: LSCAN, CREAD, DREAD, VIP to change disk.

SYSGEN1: Write New System Nucleus to Disk

This utility program reads the nucleus data created by the NUCGEN utility and writes a new system nucleus to the current MUSIC/SP IPL volume. An IPL record is also written to the volume. The current nucleus on disk is replaced. The new nucleus will take effect at the next IPL of MUSIC/SP.

As a method of applying a new nucleus, SYSGEN1 is an alternative to IPLing from a tape or VM spool file created by the NUCGEN utility. SYSGEN1 is fast and easy, but has the disadvantage that no backup copy (on tape or as a VM spool file) is produced. The IPLable backup is useful for restoring a working nucleus if a future nucleus fails.

Usage:

Run the following job after typing the command "vip on":

```
/FILE 1 N(filename)
/INCLUDE *COM:SYSGEN1
```

where filename is the output file created by the NUCGEN utility, (normally defined as /FILE 2 in $GEN:NUCGEN.JOB).

Return codes:

0 System nucleus has been successfully written to disk.

1 An error occurred; the nucleus was not successfully written to disk.

Privileges Required: VIP, LSCAN, FILES, DREAD, CREAD

SYSREP: Load Library Patching

The SYSREP program applies *reps* (patches or fixes) to load library members. It replaces specified existing bytes by new bytes, at a specified displacement within a member. The VER (verify) statement gives the existing data and the REP (replace) statement gives the replacement data. Use the following control statements:

```
/FILE . . .    (if UDS load library)
/INCLUDE SYSREP
LIBE=n,INPUT=n,TEST=T/F
NAME membername
BASE baddr
VER addr hexdata
REP addr hexdata
 . . .
 . . .
```

Notes:

1. LIBE=n specifies the unit number or DEB number of load library. If LIBE='SYST' is specified, the system library will be assumed. LIBE can be abbreviated as LIB.

2. INPUT=n specifies the unit number for the input data. The default input unit number is 5.

3. TEST=T specifies that changes will not actually be done to the load library member. The default is TEST=F.

4. Reps may be applied to more than one member per job by using a NAME statement to specify the next member.

5. If a verify fails, no further changes will be made until the next NAME statement.

6. If the base address *baddr* is not specified, it defaults to zero. If specified, it causes the verify (VER) and/or replacement (REP) to apply at displacement *addr-baddr* from the start of the member. The base

address is reset to zero each time a NAME statement is processed.

7. *hexdata* may contain commas, but must not contain embedded blanks.

8. Statements with * in column 1 are considered to be comments and are ignored.

Privileges Required: LSCAN, DREAD, CREAD, VIP or SYSMAINT.

SYSUPDATE: Update Load Library Directory in Main Storage

This program performs one of two functions, which are referred to by the keywords LDDIR and NONRES.

The LDDIR function of SYSUPDATE compares the Load Library directory entries in main storage with those on disk and updates any entries in main storage which are different. Differences could be due to updates (by the LDLIBE utility) to the Load Library since MUSIC was last IPLed, which result in new starting block number, length, etc. for some members. SYSUPDATE causes the changes to take effect immediately, rather than after the next IPL as would otherwise be the case.

However, only existing entries in main storage are updated. No new entries are created. Also, the entry is not updated if the member is in the Link Pack Area (FLPA or PLPA).

The NONRES function of SYSUPDATE temporarily removes member names from the list of members in the Link Pack Area (FLPA and PLPA). This prevents the use of an LPA copy of a member. The change remains in effect only until the next IPL of MUSIC.

To run the program, type SYSUPDATE, then enter LDDIR or NONRES when prompted. The command /VIP ON must be entered before running SYSUPDATE.

Privileges Required: LSCAN, CREAD, DREAD, and VIP.

TRANS$: Transferring Funds Between User Codes on MUSIC

The TRANS$ program can be used by a course or project supervisor to transfer computing funds from one MUSIC user code to another. It can also be used to change their passwords. This allows authorized persons to distribute allocated funds among the users that they supervise. This authorization is given via the SUPV code privilege. The code that the funds are transferred from, and the code receiving the funds must both start with the same two characters as the code with the SUPV privilege. For example, the authorized code XY00 could use TRANS$ to move funds from XY15 to XY25.

A log file of all TRANS$ change requests can be kept. The installation can set up this log file by specifying its name in the file $PGM:TRANS$.

Each user code on the MUSIC system has two dollar values associated with it: a current amount and a limiting amount. The current amount starts at zero and is increased each night by the charges for that day by the NOWDOL system utility. When the current amount reaches the limiting amount, the code may no longer be used until the limit is increased by adding more funds. This can be done by the MUSIC System Administrator, or by using the TRANS$ program to transfer funds from another code.

A user can use the SHOW$ command of PROFILE to display the current and limiting amounts. (This program is described in the *MUSIC/SP User's Reference Guide*.) The money remaining is the limit minus

the current amount.

To run the TRANS$ program, type TRANS$ at the terminal. You will then be prompted to enter the request, which can be any of the following commands:

TRANSFER $nnn FROM xxxx TO yyyy

ALLOCATE $nnn FROM xxxx TO yyyy

GET xxxx

CHANGE cccc PW=pppppppp

HELP

END

On the TRANSFER command, *$nnn* is the number of dollars (without cents) to be transferred from code *xxxx* to code *yyyy*. The $ sign may be omitted. Each code may be specified as a 4-character code (e.g. xy15) or as a 7-character code and subcode (e.g. xy17001). The command name TRANSFER may be abbreviated TR, TRA, TRAN, etc. If "FROM xxxx" is omitted, the transfer is from the user code running the TRANS$ program.

The ALLOCATE command is handled like the TRANSFER command, except that only enough is transferred to make the unused amount in the receiving code the specified number of dollars (nnn). Normally a range or list of codes would be specified. This does a *top up* on each of the codes. Abbreviations AL, ALL, ALLOC may be used.

The GET command displays the dollar values for the specified code.

The CHANGE command is used to assign a new 1- to 8-character password *pppppppp* to the specified code *cccc*.

The HELP command displays information on how to use the TRANS$ program.

The END command stops the program.

A numeric or alphanumeric range of codes can be specified on the GET and for the *to* code on the TRANSFER and ALLOCATE commands. A list of codes, within parentheses, can also be specified. Type the HELP command for further information.

Examples of TRANS$ commands:

```
TRANSFER $100 FROM XY32 TO XY75

TR 150 FROM UV25001 TO UV17003

GET XY79

GET UV15999
```

Sample TRANS$ session (on user code XR00):

```
trans$
*In Progress

-- MUSIC FUNDS TRANSFER --

ENTER YOUR REQUEST, OR HELP
?
get xr51
  XR51      $LIMIT:   500     USED:   491     LEFT:     9

ENTER REQUEST, OR END
?
transfer $250 from xr73 to xr51
$250 TRANSFERRED FROM XR73 TO XR51

ENTER REQUEST, OR END
?
get xr51
  XR51      $LIMIT:   750     USED:   491     LEFT:   259

ENTER REQUEST, OR END
?
end
*End
*Go
```

Privileges Required: SUPV.

UCB: Display Unit Control Blocks

This program displays Unit Control Blocks (UCB). Each UCB represents a disk or tape device on the system. For each UCB the list includes: main storage address of the UCB, device address, type, volume name, and the number of errors that have occurred since the last IPL of MUSIC.

To run the program, type UCB in *Go mode.

Privileges Required: LSCAN and CREAD.

UCR: User Control Record Utility Program

In order to create a file, a user must have a User Control Record (UCR) associated with the user's sign-on code. This record contains five values:

1. 1. Limit of the total space for code (TOTLIM)

2. Maximum size of a file the code can allocate (FILLIM)

3. The current total space of the code

4. High water mark to date

5. Reserved

The above values are in units of K (1024) bytes. A value of -1 means *no limit*.

This program is used to maintain the UCR records. It can be invoked by entering UCR and will accept the following requests conversationally:

GET	Display the UCR values for a code
SET	Set values 1 and 2 for a code. A new UCR will be created if one does not exist. Optional parameters TOTLIM=n and FILLIM=n may be used to change the limits individually.
REP	Replace a UCR for a code
DEL	Delete a UCR for a code
HELP	Ask for information about the UCR program
END	Terminate the UCR program

UCR values are specified by the UCR parameter, as shown in the examples below. The CODE parameter specifies a single userid, or a numeric range of userids. Do not include the subcode (if any) when specifying a userid to UCR. For example, CODE='XX01-25' applies the same request to each of the userids XX01, XX02, ..., XX25.

The Code Update (CODUPD) utility automatically creates a UCR record, if necessary, when a userid is allocated, and deletes the UCR when a userid is deleted.

Examples:

```
GET,CODE='ABCD'
SET,CODE='XY11-20',TOTLIM=500
SET,CODE='X123',UCR=5000,300
REP,CODE='DC77',UCR=10000,-1,327,327,0
DEL,CODE='DC88'
```

Privileges Required: LSCAN, FILES, CODES, or MAINT.

UCRFIX: Fixup UCR Records Based on Code Table

This program scans the code table and for each user code encountered, it adjusts the UCR (User Control Record) limits for that code, or creates a UCR record if there is none.

This program would be run if the Save Library was destroyed and it was necessary to restore files from incremental MFARCH archive tapes. UCRFIX should be run before the restores, to ensure that restores will not fail because of missing UCRs. Specify FACTOR=0 to do this. If you also want to increase existing UCR limits, to avoid restores that fail because the UCR total limit is reached, specify a value such as FACTOR=1.5. An up-to-date code table dump should be restored, if available, before running UCRFIX and before restoring files.

There are two UCR limits: (a) the maximum size of each file, in units of 1K = 1024 bytes, and (b) the maximum total amount of space occupied by the code, also in units of 1K. Limit (a) is set to the value -1 (i.e. no limit) by the program when it creates a UCR. Limit (a) for a UCR that already exists is not changed. Limit (b) is set according to the program parameters, as described below.

Usage:

```
/INCLUDE UCRFIX
TOTAL=n,FACTOR=x,MARGIN=k
```

TOTAL=n specifies the value for limit (b) when a new UCR is created. Default is TOTAL=10000.

FACTOR=x controls how limit (b) is to be changed for existing UCRs. The number *x* can be a floating point value such as 1.5. If FACTOR=0, existing UCRs are not changed at all. Otherwise limit (b) is set to min(x*c,c+k), where *c* is the total space currently occupied by the code's files and *k* is the MARGIN parameter. However, limit (b) is never decreased. The defaults are FACTOR=0 and MARGIN=500.

Privileges Required: LSCAN, FILES, CODES or MAINT.

UDSRST: User Data Set Restore

An alternate version of the restore program exists which can restore only User Data Sets (not System Data Sets). It does not require any privileges to be run and may run under the user's code who owns the file restored.

```
/FILE n TAPE ...    dump tape
/FILE m UDS(receiving data set) VOL(volume) OLD
/INCLUDE UDSRST
IN=n,OUT=m,VOL='from vol',DSN='old dsn'
```

The file statements point to the dump tape and the receiving data set respectively. Normally *n* is set to 1 and *m* to 2.

The other input parameters give the data set name and its original volume at the time it was dumped. This is used to locate it on the tape.

Contents of Information Records

In a dump produced by UDSARC, each data set is preceded by a *DS1 record and a *DS2 record. The *DS1 record has the volume name, the data set name, and the date and time of the dump. The *DS2 record contains the following information:

Column 8	B for backup, N for no backup.
9-12	Device type.
13-15	Data set organization.
16-19	Record format.
20-25	Block size.
26-31	Logical record length.
32-37	No. of tracks (no. of physical blocks if FBA device).
38-40	No. of extents.
41-44	No. of blocks per track (32 if FBA device).
45-48	Key length.

49-53	FBA physical block length (0 if not FBA device).

Privileges Required: None.

ULINIT: Initialize Save Library Data Sets

This program initializes save library data sets. Either an index or data space can be initialized. See the topic "Adding Space to the Save Library" in *Chapter 6 - System Reconfiguration* for more information.

VMSUBM: Submission to Other Systems

VMSUBM allows a MUSIC system running under VM to transmit data (i.e. jobs) to any other system on that machine. This is accomplished by creating a spooled punch file and then spooling it, via the VM SPOOL command, to a reader of the appropriate class defined on the target system. Configuration parameters are read in at execution time and are in MUSIC NAMELIST format. For an example of how to specify these parameters see the file $PGM:VMSUBM.

Configuration Parameters

UNIT	Addresses of spooled punches defined for use by VMSUBM and SUBMIT program. A maximum of ten addresses may be specified.

The following parameters describe the target systems. Up to ten may be defined.

INSYS	A list of names (1-8 characters long) which the MUSIC user uses to identify the target systems.
OUTSYS	A corresponding list of names which VM uses to identify the target systems.
TAGS	A list of corresponding tags to enable files to be spooled via RSCS to other processing units. This information is only used if the corresponding OUTSYS is RSCS. Each tag consists of two 8 character fields, the node (processing unit) name and the virtual machine name.
SPLCLS	The reader spool class on which each system is expecting input from MUSIC.
NUM	The maximum number of records that MUSIC can send to a particular system.
MINUM	The corresponding minimum number of records.

Privileges Required: None.

WAITS: List System Wait Statistics

This program displays a list of all the system wait-time statistics. Every time MUSIC goes into a wait state, a record is made of why it is waiting and for how long it waits. This information can be useful to the system programmer and installation management personnel. The meaning of the wait times can be found in the file XWAITS.

The program can be run from a terminal by entering WAITS. The times are those accumulated since the last IPL.

Privileges Required: CREAD.

WHOACT, WHOALL, WHOSON: List Terminal Users

Three programs are available that display lists of terminal users. The program WHOALL (run by a entering WHOALL) displays a list of all terminals (ports) defined on the system. For each port, it gives the virtual and real physical line address, terminal control block (TCB) number, user code and subcode, and terminal identification. At the end of each line, an asterisk is shown if the port is currently in use. If the port is not in use, the code, subcode, and terminal identification fields are those of the last active user. Fields of vertical strokes indicate a port where the last activity was a new user coming on but no valid sign-on has yet been received.

The program WHOACT (run by entering WHOACT) is similar to WHOALL except that it lists only active ports. (See Figure 17.7 for a sample run of WHOACT.)

The program WHOSON runs in a conversational mode. Simply type in the userid or groups of userids that you want information on. This program also shows how long it has been since the user last asked for user region service (TLAT). It gives an 8-char program name, which is the member name (for /LOAD XMON jobs) or the /LOAD name (for other jobs). Also, just before the program name in the output, there is a "flag", giving the job state: OFF (TCB is not signed on), RUN (running a job), *GO, RD (waiting for user input), SPI (spooled input). In the case of OFF and *GO, the program name is that of the previous job on that TCB.

```
/exec whoact
*In Progress

     W H O   A C T    26APR83
     - - -   - - -     14.55

   SEQ  TCB  VIRT REAL  USERID


    33   33  043  03C  MD69000
    34   34  044  03D  ML55000
    35   35  045  03E  DPG0888
    37   37  047  040  AD91SCR
    50   50  054  04D  AD87000
    52   52  056  04F  ED34000
    54   54  058  089  CFGC000
    55   55  059  08A  MT29000
    56   56  05A  08B  CCSO000
    57   57  05B  08C  DPJM000
    58   58  05C  08D  CCTW000
    85   85  0A1  5D9  CCSO000
*End
*Go
```

Figure 17.7 - Sample Run of WHOACT

Privileges Required: INFO and LSCAN.

XTELL: Sends Messages

This program is similar to the TELL command except that it delivers the message regardless of whether the target user has set MESSAGES ON or OFF.

Privileges Required: SYSCOM.