MUSIC/SP Administrator's Reference

Part 2 - Chapters 6 to 9 (AR_P2.PS)

# Part III. Customizing MUSIC/SP

# Chapter 6. System Reconfiguration

## Overview of System Reconfiguration

The ADMIN facility provides a series of menus under the topic "System Support Functions" that allow you to perform most of the basic configuration functions. (The ADMIN Facility is described in the *MUSIC/SP Administrator's Guide*.) This chapter is intended to provide background information on what is going on behind the menus. The programs and commands used often require privileges and should be run from the $000 userid (or equivalent) with VIP set ON.

## Modifying the System Catalog

The system catalog defines the names and volumes of system data sets, the members to be made resident in the link pack area, and VM commands to be executed automatically at IPL time. The catalog has to be modified if you are adding or moving system datasets, if you are adding or deleting modules from the FLPA or PLPA, or if you want to add some VM commands to be executed during initialization. Modifying the catalog is a two step process. First use the editor to make the changes to the file $GEN:SYSCAT, then use the EDTCAT utility to copy the catalog records to the catalog system dataset (SYS1.MUSIC.CATALOG). The new catalog takes effect the next time the system is loaded. If the system fails to initialize due to errors in the catalog, you can use the =EDIT IPL option to fix the catalog and allow the system to get going. See the description of the EDTCAT utility for more details.

## Defining New Terminals

Terminals are defined by device cards in the NUCGEN job. The "System Configuration" function of ADMIN can be used to make routine additions and deletions to the I/O configuration, including the terminals you have defined. You can do the same thing without using ADMIN by editing the NUCGEN jobstream in the file $GEN:NUCGEN.JOB, running the jobstream and installing the generated nucleus. See the description of the NUCGEN utility for more details.

If you are adding a large number of new terminals you may also have to increase the size of some system data sets and allocate more resources to the system to maintain system performance. Consult *Chapter 19 - Direct Access Storage* to determine if the SWAP, PAGE, and SCRATCH datasets are large enough for the number you intend to define. Additional terminals usually means additional users. You may have to add Save Library data sets to provide file space for these users. Use the LIBSPACE utility to determine what free space is available in the existing Save Library.

Additional terminals increase the systems main storage requirements. Main storage is not only required for additional control blocks and buffers, but is also needed for extra page pool space to support the increased load. The actual amount of storage required is dependent on a large number of factors, but you should have at least 1 Meg for every 32 users. Extra storage will give better performance. You may also want to spread swapping and paging over a number of different channels. This is important in optimizing performance on systems with a large number of users.

## Increasing the Number of DASD Channels

DASD channels are of the selector type and can only handle one I/O request at a time. If more than one channel is available, I/O operations can be done simultaneously on each channel. MUSIC/SP's performance can be improved by making more than one disk channel available. In order to take full advantage of this performance gain you must configure the system to use the available channels for both swapping and paging. Up to three channels may be used simultaneously for swapping and paging.

Allocate the additional swap and page data sets. *Chapter 19 - Direct Access Storage* has details as to the space and blocksize requirements. For optimal performance try to locate these data sets near the middle of the disk packs. This will minimize arm movement.

Use the FORMAT utility to format and initialize the new data sets. Add entries for the new data sets in the system catalog. The system will automatically use the new data sets the next time it is loaded.

Performance gains can also be made by balancing the channel load and placing high usage data sets on low usage disks. The IOTIME utility gives a breakdown of disk I/O requests by channel, DASD volume, and system data set. It is extremely useful in monitoring disk usage. The basic procedure for moving or enlarging a system data set is as follows.

- Allocate the new data set using different name.
- Format the dataset.
- Copy any required data.
- Modify the catalog to point to the new data set.
- Re-IPL the system and test that the system works.
- Delete the old data set after few days.

Care should be taken during the step that copies the data to make sure that the old data is not changed either during or after the copy process. The simplest rule is to make sure that there are no users on the system and do not create or delete any files. For data sets like PAGE, SWAP and SCRATCH this step can be skipped.

## Increasing the Main Storage Size.

MUSIC/SP's performance can be improved by giving the system more main storage. This enables the system to keep more user tasks in storage and reduces both the paging and swapping overhead. For performance reasons it is very important that VM does not page MUSIC/SP, so all of MUSIC/SP's pages should be locked or it should be run in the V=R region. The simplest way to lock MUSIC/SP's pages is to include a VM LOCK command in the catalog so it will be issued automatically when the system is started. There are a number of NUCGEN parameters that tell the system how to use main storage.

The MAXCOR parameter defines the main storage available. Since the system will automatically figure out how much it has, most people find it convenient to set MAXCOR to a large value and have the system figure out the real value for itself.

The REGION parameter defines the maximum virtual storage area for the user region. The default of 1 Meg is suitable for most applications. Increasing this has no major effect on system performance but may require you to enlarge the PAGE data sets if a significant number of users take advantage of the larger region.

The MAXRRS parameter defines the maximum real storage that a user can get. It determines how large a program can get before the system will page it. It also determines the maximum size that is involved in a swap operation. Most programs run quite well using the default MAXRRS of 272K, but programs whose

virtual storage "working set" is significantly larger than MAXRRS may experience performance problems due to excessive paging. The setting of MAXRRS can have significant effects on system performance. Increasing MAXRRS will improve the performance of user programs with large storage requirements by reducing the paging overhead. This performance gain will be at the expense of the system in general, in terms of an increased swap load and reduction in the multi programming level (MAXMPL), unless sufficient real storage is added to the system to accommodate the increase in the MAXRRS. To increase MAXRRS by nnnK add at least nnnK*MAXMPL to the total main storage. If you do not have enough main storage satisfy the formula above then you can trade off a higher swap load for improved performance of large programs.

If you are running VM/ESA with the ESA feature you can define up to 64MB for MUSIC. The storage above 16MB is added to the storage available to user programs. This will reduce swapping and allow you to increase the MAXRRS parameter as described above. The storage above 16MB cannot be used for FLPA, RAM disk or trace tables. There is no extra overhead in running user programs above 16MB line.

## Adding Space to the Save Library

Use the LIBSPACE utility to find the current amount of free (unused) space in the library. You can add to the space available by following the procedure given below.

To add more space to the Save Library, you must create additional library data set(s). There are from 1 to 180 Save Library data sets, named SYS1.MUSIC.ULnn (where nn=01, 02, ..). Each can hold up to approximately 57000K of user files. The data sets must be added in order. For example, if MUSIC was installed with 4 data sets (UL01 to UL04), the next to be added would be SYS1.MUSIC.UL05.

Note: It is possible to have up to 180 library data sets. The data set names from 100 on would then contain 3 digits instead of 2, e.g. SYS1.MUSIC.UL100.

The smallest recommended size for each library data set would be 16200 records, which can hold up to about 8096K of user files. The maximum size is 114,504 records, which can contain 57,248K of user files.

Older versions of MUSIC supported a maximum size of 16200 records. Each library data set can be a different size. Installations which have a large number of Save Library data sets will want to choose larger sizes for individual data sets, so that they will not exceed the maximum number of 180 data sets. Larger size means that individual user files within these data sets can be larger. For example, for the 16200 record size the maximum user file size would be about 16200 x 512 or 8 million bytes. The largest data set can accommodate a maximum file of about 57 million bytes.

A library data set can be increased in size at a later date by following the procedure in the next topic.

The following is an example of the procedure for adding a Save Library data set of a size of 16200 records. It is also described in *Chapter 19 - Direct Access Storage*. The example below assumes that UL01 through UL04 already exist and that UL05 is to be added. The device type is assumed to be 3350; the procedure is similar for other device types.

1. Calculate the number of tracks needed for 16200 records of 512 bytes each.

   e.g., 3350: 16200/27 = 600 tracks (round up)

2. Allocate the data set (SYS1.MUSIC.ULnn) and use the FORMAT utility to format it. First enter /VIP ON, then run the following job:

```
/FILE 1 UDS(%UL05) VOL(MUSICX) NEW
/ETC NTRK(600) BLK(512) BUFNO(0)
/LOAD IEFBR
```

Start the FORMAT program at a terminal and enter the following commands when prompted.

```
DEVICE=3350
VOLUME='MUSICX'
DSN='SYS1.MUSIC.UL05'
```

When the dataset has been formatted terminate the FORMAT program by responding to the prompt with /CANCEL.

3. Initialize that data set by running the following job.

```
/FILE 1 UDS(%UL05) VOL(MUSICX) OLD
/INC ULINIT
```

When prompted for an option enter LIB.

4. Add the following record to the system catalog. The file $GEN:SYSCAT normally contains the catalog.

```
ULIB05   U005 00 MNS 000 MUSICX SYS1.MUSIC.UL05
```

After changing $GEN:SYSCAT run the EDTCAT utility.

5. The new space will be available after the next IPL of MUSIC.

More than one library data set may be added at the same time, if desired.

Run the LIBSPACE utility to find the current number of ULnn data sets (4 in the above example).

The standard number of tracks in a ULnn data set (16200 512-byte records) for the common device types are:

```
3340 . . . .  1350 tracks
3330 . . . . . 810
3350 . . . . . 600
3375 . . . . . 405
3380 . . . . . 353
3310,3370  . . 507
933x . . . . . 507
```

## Enlarging an Existing Save Library Data Set

The following procedure describes the steps required to enlarge an existing Save Library data set. You might want to do this if you have a large number of data sets and do not want to approach the maximum of 180 data sets.

In the example below we are enlarging data set SYS1.MUSIC.UL05 that exists on MUSICX. The larger one is on MUSICY.

1. Allocate the new larger library data set. Follow step 2 of the previous topic to allocate and format the new one.

2. Create the file CPYLIB that contains the following JCL:

```
/FILE 1 UDS(%UL05) VOL(MUSICX) SHR
/FILE 2 UDS(%UL05) VOL(MUSICY) OLD
/INC DSCOPY
```

3. Create the file DOCAT that contains the following:

```
/INC EDTCAT
/INC $GEN:SYSCAT
```

4. Create the file FIXMAP that contains the following JCL that points to the new larger data set:

```
/FILE 1 UDS(%UL05) VOL(MUSICY) OLD
/INC ULINIT
```

5. Edit the file $GEN:SYSCAT to point to the new volume the larger data set is on.

6. Make sure no one else is on the system and that you make no changes to the Save Library during the following steps. If you do have to make changes, simply restart the procedure from step 7.

7. Type "CPYLIB" to run the job in that file to copy the old data to the new data set.

8. Type "FIXMAP" to enlarge the bitmap on the new larger data set. Enter the option "XLIB" when prompted. (If you typed another option here you must return to step 7.)

9. Type "DOCAT" to put the new system catalog on line.

10. Re-IPL the system. After this point the new larger data set will be in use. If everything is right, then delete the old data set.

## Reorganizing Save Library Free Space

Over time, the free (unused) space in each Save Library space may become randomly distributed throughout it. This means that MUSIC may not be able to obtain sufficiently large amounts of space from it. The allocation algorithm will use up to 5 extents from a single library data set to satisfy an initial request for file space. (Subsequently the file can grow to 16 extents and may span library data sets). For example, it might be that there is 200K free space but the largest 5 extents are 2K each. In this case only a 10K file can be allocated. This *fragmentation* condition can be spotted by looking at the output of the LIBSPACE program. Run the MFMOVE utility to eliminate this fragmentation. If your free space is often fragmented (every month or so) and users complain about being unable to allocate files, the problem is probably a lack of file space and you should consider adding some new datasets to the Save Library.

## Enlarging the Code Table

The MUSIC code table is comprised of two data sets: the index (SYS1.MUSIC.CODINDX) and the code records (SYS1.MUSIC.CODTABL). The standard size table can hold about 20000 user code records. This

can be increased to about 50000 if desired. Increasing the code table size has no impact on system performance. The procedure is:

1. Use CODUMP utility to dump the old code table to tape.

2. Allocate and format new system data sets SYS1.MUSIC.CODINDX and SYS1.MUSIC.CODTABL, using files FMTCDX and FMTCOD with the FORMAT utility. After formatting, the only authorized code on the system is $000 (with password MUSIC). The index should contain at least 340 records (block size 4096). The code table data set should contain the desired number of records, up to a maximum of about 50000.

3. Use the CODUMP utility to restore the code table from the tape produced in step (2). Code $000 must be used for this job.

*Note:* Before doing step (2), it is a good idea to rename the old index and code data sets (using DSREN utility). Then if something goes wrong, you can IPL MUSIC and change the catalog to point to the old data sets if necessary. In this way you will always have a usable system. After step (3), the old data sets can be deleted.

## Enlarging Main Storage Dump Data Set

The system main storage dump data set, called $PGM$DMP on the MUSIC1 volume, is set up to handle a storage size of up to 16 megabytes. Some installations may want to change its size. Consult the job in the file $GEN:DMPGEN.SAMPLEJOB for the commands that were used to create the original one. Simply delete the old UDS file and rerun the job specifying a different number of records. For example, use 16385 records to handle an 8 megabyte storage dump. Note that the dump data set must be only one extent. You can check this by using the $INFO option of the UTIL program.

## Configuring the RAM-Disk

To improve system performance it is possible to define a 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.

The RAMDLD utility loads the RAM disk area from the Save Library. It is called automatically by JOBONE 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 to be loaded into RAM. If RAMDLD 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.

The format of the $PGM:RAMDISK.LIST file is simply a list of fully qualified file names (Code,

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. A sample $PGM:RAMDISK.LIST file is distributed with the system.

The RAMREP program produces a report on RAM disk utilization. It is useful in monitoring the performance of your RAM disk area. It is important to monitor RAM disk usage because if the files in RAM are not often used the memory assigned to them would be better used as part of the page pool or LPA.

The performance benefit will vary from system to system but, using a RAM disk area of 400-800K, you should be able to get between 15% to 25% of all open requests be for files in RAM.

To enable the RAM disk performance option take the following steps.

1. Determine how much memory to allocate for the RAM-Disk. An area of 512K is large enough to hold the files defined in the distributed RAMDISK.LIST file.

2. Define a RAM disk area in the NUCGEN (ADMIN 4 10 5).

3. Add any local high usage files to $PGM:RAMDISK.LIST.

4. When you next IPL the RAM-Disk will be activated automatically.

## Configuring MUSIC Programs

There are two important files that configure many programs on MUSIC. They are:

$EML:MAIL.CONFIG
and
$TCP:TCPIP.CONFIG

These files contain important information specific to your site, and when these programs are run they affect many MUSIC programs on your system.

MAIL.CONFIG contains information about MAIL programs and programs that pertain to MAIL such as POP servers and MCS (MUSIC/SP Client/Server). Refer to *Chapter 9 - Electronic Mail Facility*.

TCPIP.CONFIG contains information about many servers that use TCP/IP, such as: TELNET, FTP, Gopher, Web, etc. Refer to *Chapter 16 - Configuring MUSIC for TCP/IP*.

## Benchmark Programs

The benchmark programs (userid: $BMK) BM1 thru BM8 can be used to measure machine speed, mainly for the purpose of comparing the speed of the various machines that run MUSIC. They measure CPU processing speed (floating-point and integer instructions) and disk i/o speed. Since the programs should be run on a quiet MUSIC system, without any swapping or paging or competition from other jobs, they measure the speed of the hardware, not the efficiency of MUSIC.

The programs are:

BM1 Instruction speed for double-precision floating-point calculations, involving arrays. The program

should display the result X = 499461.038961036

BM2 Instruction speed for integer calculations, involving arrays. The program should display the result X = 871192

BM3 Disk speed for sequential read of 1000 512-byte blocks, 1 block at a time.

BM4 Disk speed for sequential read of 8000 512-byte blocks, 20 blocks at a time.

BM5 Disk speed for reading 1000 512-byte blocks, 1 block at a time, with a seek of a varying number of cylinders required for each read. The seek distance varies from 0 to 8000 blocks, with an average of about 4000 blocks.

BM6 Same as BM3, but writes instead of reads.

BM7 Same as BM4, but writes instead of reads.

BM8 Same as BM5, but writes instead of reads.

*Note:* Please refer to $BMK:BM.DOC for complete documentation on these programs.

# Chapter 7. Terminal Configuration and Tailoring

## Terminal Definition

Terminals and workstations are defined in the NUCGEN. The device address, the type of terminal and various connection options are specified. See the section on the NUCGEN program for details.

In addition to the information specified in the NUCGEN, the system module TRMCTL contains definitions of specific terminal types. Each terminal definition is assigned a unique name. To use a particular terminal definition, the user must specify this name during sign on or in the profile. These terminal definitions contain many parameters, such as line length, carriage return rate, tab and backspace characters, screen addressing, translate tables and other special characteristics.

All parameters for a terminal definition are specified as arguments of the TERMINAL macro instruction. New TERMINAL definitions are added to the TRMCTL module immediately following the existing ones.

The module TRMCTL contains sample TERMINAL specifications for several varieties of terminals. The conditional assembly statements will skip the generation of them. These extra specifications are used at McGill University.

Once changed, the object module for the new TRMCTL module is added to the NUCGEN generation module after the DEVEND statement. The nucleus generation step is then performed to produce a tape which is then IPLed to effect the change.

Some parameters described below are not valid for all the major classes of terminals supported by MUSIC. Each description specifies which of the generic terminal types honor each parameter. Descriptions with no terminal type explicitly mentioned are valid for all types. The major supported types are :- TTY, 2741, 1050, 3270.

When parameters call for the specification of terminal control characters, these are the codes as seen by the processing unit. In particular the codes seen from ASCII terminals differ from those described in the vendor's publications. Refer to topic "ASCII to S/370 Code" in this chapter for further information.

### Terminal Macro Parameters

APL — This parameter indicates which type of APL character set is available on 3270 type terminals.

| | |
|---|---|
| APL=DAF | Data analysis feature (3272) |
| APL=TEXT | APL/text feature (3274) |
| APL=NO | None (default) |

BLANK — This parameter must be the hexadecimal value of a blank in terminal code.

BS — This parameter must be the hexadecimal value of a physical backspace character in terminal code.

BUFSIZ — Size, in bytes, of terminals built in buffer. This is usually used when transmitting data to a terminal at a line speed higher than the terminal can print. Periodically MUSIC sends a sequence to the terminal saying its buffer is full and wait for a response before continuing output.

BUFSEQ Two byte sequence used to inform buffered terminals that its buffer is full (see BUFSIZ).

CLEAR This parameter contains a three-character (hex) clear screen sequence to be used by MUSIC to clear TTY type screens. If not specified, MUSIC will not attempt to clear the screen.

CRONLY This parameter controls the ending sequence sent to a TTY terminal at the end of an output line which is not to be followed by a line feed. Normally this parameter points to a sequence containing a carriage return. Allowed values are the same as for READNL.

ENDNL This parameter controls the ending sequence sent to a TTY terminal at the end of an output line. Allowed values are the same as for READNL.

FASTBS This parameter may have a value of YES or NO. It indicates whether the terminal has a fast physical backspace. That is, one that does not require idle characters following it. It is valid only for TTY type terminals.

FFDELAY This parameter gives the number of idles to be transmitted after a forms feed operation. A forms feed will only be attempted if FORMS=TRUE is specified.

*Note:* This delay is usually related to line speed and the length of the forms.

FOLDNL This parameter controls the ending sequence sent to a TTY terminal at the end of the first line of folded output (that is, a line too long for the physical carriage length). Allowed values are the same as for READNL.

FORMS This parameter may have a value of YES or NO. It indicates whether the terminal is equipped for vertical forms control.

HEX This parameter contains a pair of hexadecimal bytes which will be used to translate input text for this terminal. After translation to EBCDIC, any occurrence of the first parameter will be replaced by the second value. The two values must be contained in parentheses and separated by a comma. This parameter may be overridden by a user profile or by the terminal command /CTL.

IDLE This parameter consists of one or two hexadecimal characters which act as an idle on this terminal type. That is, these characters will have no effect on the terminal output. If two values are to be used, they should be in parentheses, separated by a comma. This idle character is used only by TAB routine to know which characters do not cause the type head to move. The system always uses the character X'DF' to transmit idles to the terminal as this character is recognized by the transmission control hardware.

IDLES This parameter is not processed by the TERMINAL macro. It simply is allows a convenient method of recording the number of idles to be transmitted (as calculated by RETRAT, RETBAS, MINSIZ).

INBS This parameter contains the EBCDIC value of the character to be treated as a backspace during input. This parameter may be overridden by a user profile or by the terminal command /CTL.

INTAB This parameter contains the EBCDIC value of the character to be treated as a horizontal tab during input. This parameter may be overridden by a user profile or by the terminal command /CTL.

LF This parameter must be the hexadecimal value (in terminal code) of a character which causes a line feed function on the terminal.

LFDELAY This parameter gives the number of idles to be transmitted after a line feed operation (for example, on a double spaced print line).

LINE This parameter gives the physical line length of the terminal. It is used by the system to split (or fold) lines on TTY type terminals which cannot print a full 133 bytes per line. This parameter should be specified for all terminal types as it can be used by application programs to format output.

LINUM This parameter contains the number (in decimals) of consecutive lines of output which can be sent to a TTY screen before it goes into *MORE mode*. If not specified, the terminal will scroll normally. (Value 1-238)

MINSIZ This parameter is only valid for TTY terminals. It is used to specify the minimum length of a line transmitted to the terminal. If the text of a line is not long enough, idles are added before the CR/LF sequence. No idles are added after the CR/LF. If MINSIZ is specified, RETRAT and RETBAS must not be used.

MORE This parameter contains a five-character (hex) screen addressing sequence which is used to issue the `MORE...` message to a TTY type terminal when it enters *MORE mode*. Use DF as a fill character if the required sequence is less than five characters. This can occur when the number of lines per screen is exceeded or a page skip carriage control is issued. If not specified, no message is issued.

NAME This parameter consists of 1-8 alphanumeric characters. It is used by MUSIC users on the /ID and /RESTART commands and with the PROFILE program to specify terminal control types. The NAME parameter need not be unique. See the SPEED parameter for further details. This parameter is required.

NUMBER This is the internal number by which the system identifies the terminal specifications. All numbers must be unique. User defined specs have numbers starting at 25. The maximum number is 100. There must be a specification supplied for each generic terminal type in the system (TTY, 2741, 1050, 3270). Its NUMBER must be equal to the internal code for the terminal type (these are supplied in the original TRMCTL module). The NUMBER is placed in a user code table record when the TERM command of the user PROFILE program is used. For this reason, once a specification is added, it should not be removed (or have its NUMBER changed), unless you are certain that no user profiles reference it. This parameter is required.

PREP This parameter indicates whether a *prepare* command should be used in buffered terminal protocol and *more mode*. PREP=YES is the default. PREP=NO should be specified for terminals attached via a packet-switched network, since the prepare command has no effect in that case.

PRSEQ This parameter contains the bytes (in terminal code, in hexadecimal), to be transmitted to a TTY type terminal before input is read. It may contain codes to turn on paper or magnetic tape readers. It may be from zero to 3 8-bit bytes long.

READNL This parameter is a code number indicating the type of CR/LF sequence to be transmitted to a terminal after an input line is received. The values may be:-

0 = No sequence (except possible idles)
1 = Line feed
2 = CR/LF
3 = X-OFF, CR/LF
4 = CR

If any other sequences are required, the module TERMIO must be changed accordingly.

RETRAT
RETBAS — These parameters are used for TTY, 2741, and 1050 terminals to control the amount of time allowed for carriage return - line feed operations. Following the CR/LF (or equivalent), the system transmits to the terminal the specified number of IDLE characters. These characters have no visual effect at the terminal, but since they take time to transmit, they give the terminal time to perform its new-line function. The number of idles to be transmitted is calculated as follows:-

$$IDLE\_COUNT = \frac{LINE\_LENGTH * 16 + RETBAS}{RETRAT}$$

where LINE-LENGTH is the number of characters just types on the terminal, and RETRAT and RETBAS are the values of the corresponding parameters. By setting the parameters appropriately, many different timing characteristics can be generated. For example, by setting RETBAS to zero, the number of idles would be proportional to LINE-LENGTH. By setting RETRAT to a value greater than 133*16, the idle count would be independent of LINE-LENGTH. The line speed must be taken into consideration when determining these values. The maximum value for each is 32767. RETRAT and MINSIZ cannot both be specified as 0.

RETURN — This parameter must be the hexadecimal value (in terminal code) of a character which causes a carriage return or new line function on the terminal.

RTDELAY — This parameter gives the number of idles to be transmitted after a typical short carriage return operation. This delay may be used during password blankouts, and before messages immediately following terminal input. It is valid for TTY and 2741 type terminals. The maximum value for RTDELAY is 36.

SHIFT — This parameter may be the hexadecimal value of the upper case shift indication in the terminal code. It should be specified only for terminals which transmit and receive shift characters. IBM 2741 and 1050 terminals are examples of these. MUSIC uses this bit mask to minimize the sending of shift characters in cases where they will not really be needed, such as for blanks, tabs, backspaces and idles. A value of 00 indicates no optimization is to be done.

SPEED — This parameter gives the line speed (in bits/second) for which this specification is to be used. The value may be one of 110, 134, 300, 600, 1200, 1800, 2000, 2400, LOCAL, or ANY. LOCAL is used for 3270s only. If ANY is used, this terminal specification may be used at any line speed. The combination of NAME and SPEED must be unique. This parameter is required.

TAB — This parameter must be the hexadecimal value of a physical tab character in terminal code.

TABRAT
TABBAS — These parameters are used to calculate the number of idles transmitted following an output tab on a TTY, 2741, or 1050 type terminal. The formula is the same as for RETRAT and RETBAS except that the character count used is the number of spaces to be tabbed over instead of the line length. The value of TABRAT must not be zero.

TRAN — This parameter is used to set the translate table number (TTAB) in the TCB. If not specified, the default for that terminal type is used. (See module TRANTB).

TYPE — This parameter identifies the generic type of terminal being specified. It must be one of

TTY, 2741, 1050 or 3270. It is required.

UCONLY This parameter translates output to upper case letters.

# 3270 Emulation

## Overview - 3270 Emulation

Most software running on IBM S/370 processors is designed to work on 3270 terminals. It is possible to connect ASCII devices such as PCs or terminals to IBM processors through protocol convertors that make the ASCII devices function like 3270 terminals. The following information is specific to the **7171 ASCII Device Attachment Control Unit** and the **9370 ASCII Subsystem.**

The 9370 ASCII subsystem is implemented as a line adapter card that plugs directly into the 9370 rack. The 7171 Control Unit attaches to a S/370 channel as would a local 3174 control unit. ASCII terminals, printers or PCs are attached using a standard RS-232 full duplex asynchronous interface. The terminals can be directly connected using null-modem cables or through switched lines and modems.

From the host point of view, the terminals are treated exactly like regular 3270 style terminals and they should be defined as such in the host I/O configuration tables. There are some features of these protocol convertors that the host software can take advantage of if it knows it is dealing with a protocol convertor. To do this special options have been added to the I/O configuration tables.

## Configuration

VM's I/O configuration is defined in the module DMKRIO. To let VM know that a terminal is connected through a 7171 or through the ASCII Subsystem, the FEATURE=EMUL3270 option should be specified on the RDEVICE statement for the terminal. Specifying this does two things. It causes VM to automatically drop the line when the user logs off and it allows VM to give MUSIC correct information about the device. If you do not want VM to do the automatic line drop you can specify the E3270HLD feature on the RDEVICE macro. If the EMUL3270 feature is not specified, MUSIC will not recognize that the terminals are connected through a protocol convertor.

On MUSIC these terminals are defined as 3270 devices in the NUCGEN. If MUSIC is running under VM, the terminal features are passed to MUSIC from VM when the terminal connects to MUSIC. So if FEATURE=EMUL3270 is specified in the VM configuration, MUSIC will recognize the terminal as being on a protocol convertor. If MUSIC is running without VM, the 7171 option should be specified on the device statement for the terminal in the NUCGEN. The 7171 option is ignored if MUSIC is running under VM.

In addition to supporting 3270 emulation, terminals and PCs connected through the protocol convertor can support applications that use ASCII transparent mode such as PCWS file transfer and KERMIT. If the DIALUP option is specified on the device statement in the NUCGEN, MUSIC will also automatically issue a "host disconnect" command when the user signs off. This causes any switched line connections to be broken and frees up the line for subsequent users to dial in. If DIRECT is specified on the NUCGEN device statement the "host disconnect" sequence is not issued. The DIRECT and DIALUP options only effect terminals that connect through a protocol convertor, they have no effect on real 3270 terminals.

## ASCII Transparent Mode Support

The 7171 and the 9370 ASCII Subsystem both support the "transparent" transmission of ASCII data. This feature is particularly useful when the capabilities of the ASCII device fall outside the realm of the 3270, such as in the areas of file transfer, vector graphics, text processing, and personal computing. In addition to supporting applications such as KERMIT that were designed to use transparent mode, MUSIC also supports applications that were written to run on native ASCII devices. In other words you can connect a terminal, PC or printer to MUSIC through the 7171 or ASCII subsystem and treat as if it were connected through regular ASCII asynchronous port.

MUSIC/SP's "ASCII Transparent Mode Support" intercepts the I/O requests at the channel program level and the commands and data transformed so that they perform the required functions. This is done at the lowest level of terminal support and is thus completely transparent to the application.

Terminals that connect to MUSIC in 3270 emulation mode can switch to ASCII transparent mode using the TOTTY command. Once this command is issued, protocol conversion is terminated and the terminal or PC functions as an asynchronous ASCII device. The user can switch back to emulation mode using the TO3270 command.

MUSIC ports can also be configured to start out in ASCII transparent mode when the terminal first connects. In this case the port appears as if it were a regular ASCII port. This is useful for special applications. It could be used, for example, to attach a dedicated ASCII printer or to define a group of ports for exclusive ASCII use. These must be defined as TTY devices in the MUSIC NUCGEN. The "7171" option must also be specified. For example,

```
TTY 0F0-0F3 DIALUP,1200,7171
```

The above example defines ports 0F0, 0F1, 0F2, and 0F3 as dialup 1200 baud ASCII lines. The baud rate, in this case 1200, is not used by MUSIC/SP to govern the transmission rate of the data, it is simply used to calculate the number of IDLES that should be inserted after certain control sequences, such as "LINE FEED", to give the terminal time to perform the control operation. Since most modern terminals do not require IDLES, the speed specified is generally of no consequence. The protocol convertor has automatic line speed detect and allows a variety of speeds on a particular port, so despite the fact that 1200 baud was specified in the NUCGEN, a terminal could successfully use the port at any of the allowed speeds.

*Note:* Although the protocol convertor can detect the speed of a terminal, it is not setup to pass this information back to MUSIC, so AUTOSPEED should never be specified in the NUCGEN.

## Connecting with PCWS

There are two ways to use connect to the system through a 7171 or ASCII subsystem using PCWS.

### PCWS using VT100 Emulation.

This is the simplest and in most cases the most efficient way of connecting your PC to MUSIC through ASCII lines. The protocol convertor ports should be defined as regular 3270 devices on both VM and MUSIC. Don't forget to specify FEATURE=EMUL3270 in the RDEVICE macro for VM. The user starts PCWS in VT100 emulation mode (ALT-V) and then connects to the protocol convertor. When prompted for a "TERMINAL TYPE", the user can use VT100 (standard monochrome) or VT100P (color). The PC can now be used as a regular 3270 terminal. (The VT100P terminal definition comes with MUSIC and must be installed in the protocol convertor). The following summarizes the procedure.

1. Start up PCWS in VT100 mode.
2. Connect to the protocol convertor and press Carriage Return.

3. Select a terminal type of "VT100" or "VT100P".
4. Wait for the VM logo to display and connect to MUSIC/SP as you would from a real 3270 terminal.

When the user starts a file transfer using XTMUS or XTPC the system automatically switches to ASCII transparent mode, does the file transfer and switches back. If you want to use native PCWS page mode instead of VT100 emulation, enter the command "TOTTY PCWS" to change to transparent mode and then press ALT V to switch out of VT100 emulation.

**Native Page Mode PCWS**

Originally PCWS was designed to connect PCs to MUSIC through native ASCII ports. It is possible to use native PCWS using the ASCII transparency feature of the 7171 or ASCII Subsystem, however since the overhead is quite high, VT100 emulation is better. However in circumstances where FULL duplex communication is not possible or very expensive (i.e. using packet switched networks), it is more efficient to use native PCWS on a line configured for ASCII transparency.

A group of lines are defined as TTYs with the 7171 option.

```
TTY 0F0-0FF DIALUP,1200,7171
```

The lines are dedicated to MUSIC in the VM directory. (For example, "DEDICATE 0F0 140" where 140 is the real address.)

The user starts PCWS in PAGE mode and then connects to one of the ports on the protocol convertor that is dedicated to MUSIC. When prompted for a "TERMINAL TYPE", the user can enter TYPETERM or PCWS. (The PCWS terminal definition comes with MUSIC and must be installed in the protocol convertor. It is equivalent to TYPETERM). After pressing the ENTER key the MUSIC sign-on message will appear. The following summarizes the procedure.

1. Start up PCWS in PAGE mode.
2. Connect to the protocol convertor and press Carriage Return.
3. Select a terminal type of "TERMTYPE" or "PCWS".
4. Press Carriage Return to get the MUSIC sign on message.

## ASCII Transparency: Usage Notes

When ASCII transparency mode is used certain functions do not work exactly the same as they would through a regular asynchronous communications controller. Specifically in the handling of the BREAK key and PREPARE command.

**BREAK**

The BREAK key is treated as an error and if the user presses it the terminal will be disabled till the "Error Reset" sequence, (Ctrl-R), is entered. Due to this the BREAK key cannot be used to interrupt program execution as it is on regular ASCII lines. When connected in transparent mode the break function can be accomplished by pressing the key defined as the 3270 RETURN key. It would probably be convenient for your users if you modify the TYPETERM key definitions so that 3270 RETURN key is different from the CARRIAGE RETURN key that is normally used to enter lines in ASCII mode. During output the system may continue outputting for a few lines before issuing a "break time" read.

If you are using PCWS to connect to MUSIC in transparent mode, there is an option in PCWS to send a Carriage Return when pressing the Break key instead of a Break signal. Enter SETUP mode in PCWS and change the "Page Mode Break" option on MENU 3 on the MISCELLANEOUS SETTINGS panel.

### PREPARE, Scrolling, and Pacing

MUSIC uses the PREPARE channel command in scrolling and pacing operations. The PREPARE command terminates on the receipt of ANY character from a terminal. Thus in scrolling, if the page is full MUSIC writes the `more...` message and issues a PREPARE command. The user can go to the next page by pressing any key. The protocol convertor does not transmit characters to the host as they are typed, but waits for a *line turnaround* character such as CARRIAGE RETURN before sending any data to the host. For most users this means that they should press "CARRIAGE RETURN" to go to the next page instead of CLEAR. MUSIC also uses this PREPARE command in handshaking with buffered terminals to prevent buffer overflow. In this case if the terminal cannot be configured to respond with a CARRIAGE RETURN when it has emptied it's buffer the alternative below should be investigated. *Turnaround* sequences other than CARRIAGE RETURN can be added to the terminal definition tables in the 7171 or ASCII subsystem.

### XON/XOFF Pacing

Since the 7171 and the ASCII Subsystem communicate with the terminals in full duplex mode they can perform XON/XOFF line pacing automatically without intervention from the operating system. MUSIC uses the buffered terminal protocol mentioned above to perform the same function. Since the MUSIC approach involves more overhead and relies on the PREPARE command, it is recommended that if pacing is required you use the XON/XOFF support provided by the protocol convertor. When using this type of pacing it is important that MUSIC does not disrupt the protocol by transmitting XONs and XOFFs of its own. The default ascii terminal type (See type ASCII in the module TRMCTL) will avoid sending XONs and XOFFs. The PCWS terminal type will also avoid this.

## Terminal Definition Tables for 7171 and ASCII Subsystem

MUSIC/SP provides three Terminal Definition Tables to enhance the ease of use of PCWS through the 7171 or ASCII Subsystem.

The first terminal type, "PCWS", is an alternative to the "TYPETERM" terminal type. It exists simply to remove any confusion for general users.

The second terminal type, "VT100P", should be used when connecting with PCWS in VT100 mode. The "P" is added to differentiate it from IBM's "VT100" terminal type. This TDT is added to take advantage of PCWS's extensions to the VT100 emulator, specifically colour and keystroke handling. It should NOT be used as a replacement for IBM's "VT100" terminal type. The terminal definition makes the keystrokes in VT100 mode quite similar to the keystrokes in 3270 mode of PCWS.

The third terminal type, "VT52P", should be used when connecting with PCWS in VT52 mode. Again, the "P" is added to specify that it is really designed for use with the PCWS VT52 emulator.

To install these terminal definition tables, download the files named $PCW:PCWS.TRM, $PCW:VT52P.TRM, and $PCW:$VT100P.TRM to a PC.

On the 7171 Add the terminal types "PCWS", "VT100P", and "VT52P" to the control file used for the 7171 terminal definition table generation. Re-link the image file for the 7171, and re-load the image file to your 7171. If you are not sure on how to add terminal definition tables, refer to the *7171 ASCII Device Attachment Control Unit Reference Manual and Programming Guide* (GA37-0021).

If you are using a 9370 ASCII Subsystem consult the *ASCII Subsystem Terminal Installation and Customization Guide* (SA33-1564) for information on how to IMPORT a 7171 .TRM file.

# Terminal Translate Tables

The module TRANTB contains the terminal translate tables. The beginning of the module contains a table of pointers to the actual translate tables. Each entry in this table contains a table name, some flag bytes, and the addresses of three translate tables (input, printed output, punched output). If the translate table address is zero, no translation will be done. MUSIC uses the $$TTAB field of the TCB to index this table. $$TTAB is set to one of the system defaults when the line is enabled. When a user signs on, this default may be overridden by the value specified in the TRAN parameter of the TERMINAL macro for that specific terminal type. (Module TRMCTL). An example of this is the 3270A terminal type. Using this mechanism an installation may add special function translate tables which apply only to specific terminals.

MUSIC also provides support such that a user can dynamically change to another set of translate tables in mid-session. Each of the entries in TRANTB has an eight byte name. The user need only specify that name on a /TEXT command to use those tables. For example:

| | |
|---|---|
| /TEXT ABC | Switch to the table set named ABC |
| /TEXT STANDARD | Go back to standard tables |

There is the restriction that the *new* table set must be of the same type as the old one. (i.e. byte 9 of the new entry must match byte 9 of the old one). This avoids using tables that were not designed for a particular terminal type.

The MUSIC system, as distributed, does not contain any extra translate tables. For this reason the form of the /TEXT command documented above, is NOT documented in the *MUSIC/SP User's Reference Guide*.

# Enhanced ASCII Support in MUSIC

The following describes the flexible ASCII terminal support provided in the MUSIC system. This support not only enhances the performance of terminals in the operational area of speed and accuracy but also in the user's eyes in terms of usability.

As previously mentioned the module TRMCTL contains a number of control blocks which have information pertinent to the operation of terminals. These control blocks contain information such as speed, idles required, line length, special control sequences, and tab chars. They are generated by the TERMINAL macro. Each one has an associated name, which the user may specify on the /ID command or in the user profile.

There are some recent additions to this TERMINAL macro which were implemented specifically to address the problems posed by ASCII screens and buffered terminals. These features have also turned out to be quite useful in setting up communication protocols with a wide variety of mini and micro computers.

## Controlled Scrolling

One major problem encountered when using an ASCII screen is trying to list a long file. Once the screen fills up old data just disappears (scrolls off the top) as new data is added. To avoid this three parameters were added to the TERMINAL macro. The number of lines on the screen, a clear screen sequence, and a screen address for a `MORE...` message must be specified. When the line count is exceeded or a "next page" carriage control is issued, MUSIC issues the `MORE...` message indicating that the current page is full. This is referred to as *MORE mode*. The terminal user, having read what is on the terminal's screen, tells the system that the next page is wanted by pressing the return key. (Other keys will work). MUSIC will then

resume outputting by clearing the screen and starting again from the top.

**LINUM=nn**

This is the number (in decimal) of consecutive lines which MUSIC will send to the terminal. When this number is exceeded MUSIC enters *MORE mode*. Normally LINUM should be one less than the actual number of lines on the screen since one line is required by the MORE message. Installations using IBM 3270s may wish to set LINUM to 22 on their ASCII screens for compatibility purposes, since MUSIC uses 22 lines for output on 3270s. During a terminal session the user may change the value of LINUM using the /CTL LINES=nn command.

**MORE=aabbccddee**

This five character sequence plus the text `MORE...` is sent to a terminal when LINUM is exceeded or when a *next page* carriage control is encountered. This sequence is sent to the terminal untranslated, so it should be specified in raw ASCII characters. (Refer to topic "ASCII to S/370 Code" in this chapter for details.) Usually this sequence is a screen addressing sequence used to position the MORE message at an appropriate location on the screen. The lower right hand corner is a good place. If a terminal has no screen addressing capabilities there are two alternatives. Specifying idles as the screen addressing sequence will cause the `MORE...` message to appear on the next line. If the sequence is specified as zeros, or is not specified, MUSIC will send a BELL character to the terminal to inform the user of the *MORE* condition.

**CLEAR=aabbcc**

This three character sequence (in raw ASCII) is used to clear the screen after *MORE mode* and in the event a clear screen carriage control (X'70') is sent.

These three parameters may be specified in various combinations. For example if only LINUM was specified then MUSIC would ring the terminal alarm and wait when the line count was exceeded. No `MORE...` message, or clear sequence would be sent in this case. If only MORE was specified the line counter would never overflow, thus *MORE mode* would only occur if a next page carriage control was encountered. If CLEAR is omitted MUSIC simply never attempts to clear the screen.

## Buffered Terminals

Some terminals have a built in buffer. Characters arrive from MUSIC at a given rate and are stored in this buffer while the terminal processes them in its own time. There are two advantages to this scheme:

1. Some functions at a terminal take a substantial amount of time. (Forms feeds, setting special features, plotting, etc..). Normally MUSIC would have to allow idle time for these events to take place. If the terminal is buffered MUSIC can send these time consuming sequences without idles, knowing that the terminal can save subsequent data in its buffer and catch up later on. Generally the terminal can perform much faster since it knows exactly when it can continue with the next operation. Take for example the case of a forms feed. MUSIC would have to allow enough idles for an entire page to be fed through since it doesn't know how many lines are left until the next page. A buffered terminal would continue the output immediately the forms feed was completed, regardless of how long it took.

2. Buffered terminals can be run at a higher speed than the same terminal without the buffer feature. Data can be sent to the buffer at say 1200 baud, while the terminal is printing at 300 baud. This requires some protocol to avoid overflowing the buffer. A printer rated 300 baud must be able to handle *worst case* data at that speed, meaning that it can usually exceed that speed for the average case.

Because of hardware limitations, MUSIC supports terminals in half duplex mode where communication can only be in one direction at a time. Therefore the only protocol is to have MUSIC stop periodically when it considers the terminal's buffer is almost full, and wait for the terminal to give the go-ahead to continue when it is empty. Two new parameters have been added to the TERMINAL macro to allow this.

**BUFSIZ=nnnn**

This decimal number indicates the size of the terminal's buffer. MUSIC counts the characters sent to the terminal. When this count exceeds BUFSIZ, a special sequence (BFSEQ) is sent to the terminal and MUSIC waits for some response before sending the next line.

**BFSEQ=aabb**

This is the two character sequence MUSIC sends to inform a terminal that its buffer is full, or near full. MUSIC will not send any more data until the terminal responds to this.

Note that the terminal must be capable of following this protocol. When it detects the special sequence in its buffer it should automatically send an acknowledging sequence back to MUSIC indicating that it is ready to accept more data. MUSIC will accept any character a valid response.

Some terminals use what is called the X-ON/X-OFF protocol. This only works on full duplex asynchronous lines which are normally not supported on most IBM telecommunication controllers. There is a technique to overcome this problem on MUSIC if the terminal has an answer back drum. The technique is to have MUSIC put in a sequence to trigger the drum as the last piece of information sent to buffer just before it fills up. First consult the terminal owner's manual to find out how to set the answer back drum to be a single DEL character. (Other non-printable characters might work as well.) Then generate a MUSIC terminal type using the options BFSEQ=DFAD and an appropriate BUFSIZ parameter.

Some examples of the use of the above parameter can be found by looking at the source of the module TRMCTL under the $SYS code.

## Applications to Mini/Micro Computers

Both the *MORE mode* and *buffered terminal* protocols have interesting applications in communication with mini/micro computers. Generally speaking the most common way to communicate between MUSIC and a mini is to program the mini to *look* like an ASCII terminal. Problems arise however when the mini requires *think time* to process the data it receives from MUSIC. This situation occurs most often in the area of file transfer, when the mini requires time to transfer data to cassette tape, floppy disk or some other medium. Both protocols described above can be used to stop outputting every once in a while and give the *terminal* time to respond. This can be done by having a terminal type defined with LINUM=1. MUSIC enters "MORE mode" after each line of output and sends a `MORE...` message to the terminal. The mini, on recognizing the *MORE* sequence sends a character when it is ready for the next line. A more sophisticated case might be a mini with an input buffer of a given size programmed to use buffered terminal protocol.

Regardless of whether any of the above protocols are used one of the key questions, in programming mini/micro computers to talk to MUSIC, is: "What control characters are involved?"

The first thing is that data is transferred in *reverse ASCII*. That is ASCII as documented on your 370 Reference card or the manual for your particular terminal, but all bits are reversed and a parity bit added in the low order end. (See topic "ASCII to S/370 Code" in this chapter for details.)

The next thing is the control characters at the end of each data line.

1. When MUSIC wants data (a read), it sends a *read prompt* to the terminal. This can be up to 3 bytes long and is defined by the PRSEQ parameter in the TERMINAL macro for that terminal type. By default this is a DC1 character (X-ON).

2. To send data to MUSIC simply send the data followed by a carriage return character (CR). The data may be up to 80 bytes long in general or up to 250 bytes long for conversational reads (read 9). One should keep in mind that each conversational read requires a full time slice, thus if large volumes of data are to be transferred it would be more efficient to use spooled input (SYSINR) which, though it accepts only 80 byte records, does not require user region service for every record.

3. When MUSIC sends data to the terminal it is usually terminated by a sequence of control characters which is dependent on the operation. These control sequences may be defined in the TERMINAL macro. Examples are: What to send:- after a line (ENDLN), after a folded line (FOLDLN), after a read is completed (READLN), etc.. These sequences are usually a combination of carriage return, line feed, and X-OFF (CR/LF/DC3). The most common are:

   | | |
   |---|---|
   | after a read | CR/LF |
   | after a complete line | DC3/CR/LF |
   | after a folded line | CR/LF |

   For example take a terminal in *Go mode when a user types /users.

   | | |
   |---|---|
   | a) terminal sends | /users CR |
   | b) MUSIC sends | CR/LF |
   | c) MUSIC sends | 001 USER SIGNED ON DC3/CR/LF |
   | d) MUSIC sends | DC1 (read prompt) |
   | e) the terminal may send the next line. | |

   MUSIC, by default, translates reverse ASCII to EBCDIC on input and vice versa on output. Sometimes it may be required to bypass this translation and send/receive raw characters directly from a program. On output the X'41' carriage control can be used to prevent translation. On input, calls to the subroutines NOTRIN and TRIN can be used to turn translation off and on.

## ASCII to S/370 Code

Many installations will want to tailor their MUSIC system to support some special ASCII terminal features. Often these features are triggered by sending some special sequence of characters. It is therefore important to understand how the characters are translated by MUSIC and the hardware so that the appropriate controls will be correctly sent.

The bit patterns received by the control unit are assembled into bytes before they are sent to the processing unit. ASCII codes are made up of 7 character bits plus one parity bit. These ASCII character bits are labeled from the left as 7, 6, 5, 4, 3, 2, 1. The control unit reverses the order of these bits and has the parity bit being the right most bit after inversion. For example the letter W is 1010111 in ASCII. The control unit reverses the bit order to 1110101. If the character was sent with even parity, then a right most bit of 1 would be added to form the string 11101011 which would be sent to the processing unit as the hex bytes EB. MUSIC's input translation table will then convert this to E6 which is the character W in EBCD code. The reverse process occurs on transmission.

MUSIC has a facility to bypass the MUSIC translate table on output. That is done by using the special hex carriage control of 41 as documented in the *MUSIC/SP User's Reference Guide*. The transmission control unit's translation cannot be bypassed. This is usually of little concern. Note, however, that the hex character of DF is recognized by the control unit as an idle character. It never transmits an idle character down the

line.

The following table shows each of the 128 ASCII codes and how they are converted by the transmission control unit. The three last columns show the byte stored in main storage depending on whether even, odd or mark parity is used. (Mark parity is the parity bit always on.)

| CHAR | ASCII | EVEN | ODD | MARK |
|---|---|---|---|---|
| NUL | 0000000 | 00 | 01 | 01 |
| SOH | 0000001 | 81 | 80 | 81 |
| STX | 0000010 | 41 | 40 | 41 |
| ETX | 0000011 | C0 | C1 | C1 |
| EOT | 0000100 | 21 | 20 | 21 |
| ENQ | 0000101 | A0 | A1 | A1 |
| ACK | 0000110 | 60 | 61 | 61 |
| BEL | 0000111 | E1 | E0 | E1 |
| BS | 0001000 | 11 | 10 | 11 |
| HT | 0001001 | 90 | 91 | 91 |
| LF | 0001010 | 50 | 51 | 51 |
| VT | 0001011 | D1 | D0 | D1 |
| FF | 0001100 | 30 | 31 | 31 |
| CR | 0001101 | B1 | B0 | B1 |
| SO | 0001110 | 71 | 70 | 71 |
| SI | 0001111 | F0 | F1 | F1 |
| DLE | 0010000 | 09 | 08 | 09 |
| DC1 | 0010001 | 88 | 89 | 89 |
| DC2 | 0010010 | 48 | 49 | 49 |
| DC3 | 0010011 | C9 | C8 | C9 |
| DC4 | 0010100 | 28 | 29 | 29 |
| NAK | 0010101 | A9 | A8 | A9 |
| SYN | 0010110 | 69 | 68 | 69 |
| ETB | 0010111 | E8 | E9 | E9 |
| CAN | 0011000 | 18 | 19 | 19 |
| EM | 0011001 | 99 | 98 | 99 |
| SUB | 0011010 | 59 | 58 | 59 |
| ESC | 0011011 | D8 | D9 | D9 |
| FS | 0011100 | 39 | 38 | 39 |
| GS | 0011101 | B8 | B9 | B9 |
| RS | 0011110 | 78 | 79 | 79 |
| US | 0011111 | F9 | F8 | F9 |
| SP | 0100000 | 05 | 04 | 05 |
| ! | 0100001 | 84 | 85 | 85 |
| " | 0100010 | 44 | 45 | 45 |
| # | 0100011 | C5 | C4 | C5 |
| $ | 0100100 | 24 | 25 | 25 |
| % | 0100101 | A5 | A4 | A5 |
| & | 0100110 | 65 | 64 | 65 |
| ' | 0100111 | E4 | E5 | E5 |

| CHAR | ASCII | EVEN | ODD | MARK |
|---|---|---|---|---|
| ( | 0101000 | 14 | 15 | 15 |
| ) | 0101001 | 95 | 94 | 95 |
| * | 0101010 | 55 | 54 | 55 |
| + | 0101011 | D4 | D5 | D5 |
| , | 0101100 | 35 | 34 | 35 |
| - | 0101101 | B4 | B5 | B5 |
| . | 0101110 | 74 | 75 | 75 |
| / | 0101111 | F5 | F4 | F5 |
| 0 | 0110000 | 0C | 0D | 0D |
| 1 | 0110001 | 8D | 8C | 8D |
| 2 | 0110010 | 4D | 4C | 4D |
| 3 | 0110011 | CC | CD | CD |
| 4 | 0110100 | 2D | 2C | 2D |
| 5 | 0110101 | AC | AD | AD |
| 6 | 0110110 | 6C | 6D | 6D |
| 7 | 0110111 | ED | EC | ED |
| 8 | 0111000 | 1D | 1C | 1D |
| 9 | 0111001 | 9C | 9D | 9D |
| : | 0111010 | 5C | 5D | 5D |
| ; | 0111011 | DD | DC | DD |
| < | 0111100 | 3C | 3D | 3D |
| = | 0111101 | BD | BC | BD |
| > | 0111110 | 7D | 7C | 7D |
| ? | 0111111 | FC | FD | FD |
| @ | 1000000 | 03 | 02 | 03 |
| A | 1000001 | 82 | 83 | 83 |
| B | 1000010 | 42 | 43 | 43 |
| C | 1000011 | C3 | C2 | C3 |
| D | 1000100 | 22 | 23 | 23 |
| E | 1000101 | A3 | A2 | A3 |
| F | 1000110 | 63 | 62 | 63 |
| G | 1000111 | E2 | E3 | E3 |
| H | 1001000 | 12 | 13 | 13 |
| I | 1001001 | 93 | 92 | 93 |
| J | 1001010 | 53 | 52 | 53 |
| K | 1001011 | D2 | D3 | D3 |
| L | 1001100 | 33 | 32 | 33 |
| M | 1001101 | B2 | B3 | B3 |
| N | 1001110 | 72 | 73 | 73 |
| O | 1001111 | F3 | F2 | F3 |

| CHAR | ASCII | EVEN | ODD | MARK |
|---|---|---|---|---|
| P | 1010000 | 0A | 0B | 0B |
| Q | 1010001 | 8B | 8A | 8B |
| R | 1010010 | 4B | 4A | 4B |
| S | 1010011 | CA | CB | CB |
| T | 1010100 | 2B | 2A | 2B |
| U | 1010101 | AA | AB | AB |
| V | 1010110 | 6A | 6B | 6B |
| W | 1010111 | EB | EA | EB |
| X | 1011000 | 1B | 1A | 1B |
| Y | 1011001 | 9A | 9B | 9B |
| Z | 1011010 | 5A | 5B | 5B |
| [ | 1011011 | DB | DA | DB |
| \ | 1011100 | 3A | 3B | 3B |
| ] | 1011101 | BB | BA | BB |
| ↑ | 1011110 | 7B | 7A | 7B |
| _ | 1011111 | FA | FB | FB |
| ` | 1100000 | 06 | 07 | 07 |
| a | 1100001 | 87 | 86 | 87 |
| b | 1100010 | 47 | 46 | 47 |
| c | 1100011 | C6 | C7 | C7 |
| d | 1100100 | 27 | 26 | 27 |
| e | 1100101 | A6 | A7 | A7 |
| f | 1100110 | 66 | 67 | 67 |
| g | 1100111 | E7 | E6 | E7 |
| h | 1101000 | 17 | 16 | 17 |
| i | 1101001 | 96 | 97 | 97 |
| j | 1101010 | 56 | 57 | 57 |
| k | 1101011 | D7 | D6 | D7 |
| l | 1101100 | 36 | 37 | 37 |
| m | 1101101 | B7 | B6 | B7 |
| n | 1101110 | 77 | 76 | 77 |
| o | 1101111 | F6 | F7 | F7 |
| p | 1110000 | 0F | 0E | 0F |
| q | 1110001 | 8E | 8F | 8F |
| r | 1110010 | 4E | 4F | 4F |
| s | 1110011 | CF | CE | CF |
| t | 1110100 | 2E | 2F | 2F |
| u | 1110101 | AF | AE | AF |
| v | 1110110 | 6F | 6E | 6F |
| w | 1110111 | EE | EF | EF |
| x | 1111000 | 1E | 1F | 1F |
| y | 1111001 | 9F | 9E | 9F |
| z | 1111010 | 5F | 5E | 5F |
| { | 1111011 | DE | DF | DF |
| split bar | 1111100 | 3F | 3E | 3F |
| } | 1111101 | BE | BF | BF |
| ~ | 1111110 | 7E | 7F | 7F |
| DEL | 1111111 | FF | FE | FF |

# Chapter 8. Job Submission and Retrieval Programs

## Overview of Job Submission

This chapter contains information regarding the internals and setup of the various programs involved in the submission of batch jobs and processing output from these jobs. The SUBMIT command is used to submit jobs to MUSIC batch or other virtual machines, job output can be retrieved and inspected using the OUTPUT command, and the PRINT command is available to schedule printing.

The following is a brief summary of the programs covered in this chapter:

- SUBMIT - Submit a job to batch

- OUTPUT |
- PRINT |
- AUTOPR |- Process printed output
- VMREADX |
- $ROUTING |

- VMSUBM - Spool a reader file to VM

- VMPRINT - Spool a print file to VM

# Customizing SUBMIT

The SUBMIT program can be used to submit jobs to MUSIC batch or to any other batch machines accessible through VM. The SUBMIT program can be invoked from either *Go mode or the Editor through the SUBMIT command. Details of the SUBMIT command, the /INFO statement and the parameters involved are documented in the *MUSIC/SP User's Reference Guide*.

## Setup Requirements

The SUBMIT program requires that spooled punches on addresses 011, 012, and 013 be defined in the VM directory. Information about which system a job is submitted to, job control statements, tag information, and the like are contained in special *model-JCL* files stored under the code $JCL. Files already exist on the distributed system to enable submission to MUSIC batch. If batch jobs are to be submitted to other batch machines, model-JCL files must be set up for that purpose.

## Setting Up Model-JCL Files

The model-JCL for the SUBMIT command is contained in Save Library files under the code $JCL. The file names are of the form "$JCL:JSUB.procname", where *procname* is the processor name that the user would specify on the /INFO statement or the *TO* parameter of the SUBMIT command. With the exception of the model-JCL to submit a job to MUSIC batch, these files must be set up by the installations system support personnel.

There are basically four parts to the model-JCL file. A system definition statement containing information

about the system to which the batch job is to be sent, a symbol table describing the parameters that are substituted into the JCL, and the template statements for header and trailer JCL. These are described in detail after the example below. Figure 8.1, which is from the file $JCL:JSUB.MUSIC and is the model-JCL used when submitting a job to MUSIC, is intended to serve as a reference.

```
----+----1----+----2----+----3----+----4----+----5  COLUMN NUMBER

*        JM0                                                    <--system
&CODE          01 C  01 16 05 =CODE                             <--
&SUB           01 C  00 08 04 =SUB                                |
&USERID        04 C  01 16 07 =USERID                             |
&TIME          01 N  03 03 00000010 00000001 00000999             |
&PAGES         02 N  03 03 00000500 00000001 00000999             |symbol
&CARDS         02 N  03 03 00000000 00000000 00000999             |table
&PW            02 CR 01 08 03 =PW                                 |
&CLASS         02 C  02 02 02 AA                                  |
&ROUTE         01 R  01 08 06 =ROUTE                              |
&FORMS         02 C  01 08 00                                     |
&COPIES        03 N  01 03 00000001 00000001 00000255             |
&MSG           01 C  01 50 00  Enter message for operator       <--
&/
/ID MUSJOB     &USERID &TI &PA &CA R=&ROUTE C=&COPIES F=&FORMS <--
/PAUSE &MSG                                                       |
/PASSWORD=&PW                                                     |JCL
&/                                                                |
/END                                                              |
&/                                                             <--
************
```

*Figure 8.1 - Sample Model-JCL File*

1. SYSTEM DEFINITION STATEMENT (1ST STATEMENT)

   This statement describes the system to which the job is to be submitted.

| | | |
|---|---|---|
| SPOOLID | COL 1-8 | VM logon ID of system to receive the job. |
| SPOOL CLASS | COL 10 | VM spool class to which the data is submitted. For submitting to a MUSIC system, this defines the VM class (normally J) corresponding to submit parameters CLASS(AA). The other possible values of the CLASS(xx) parameter are SA, TA, AO, SO, and TO, which automatically use VM classes S, T, E, F, and G respectively. |
| MUSIC SYSTEM | COL 11 | The character M should be coded if the target is a MUSIC system. |
| ACCESS | COL 12 | See note 1. |
| EXPAND | COL 13 | If the character X is coded, any /INCLUDE statements in the submitted files will be expanded. |
| TAGINFO | COL 16 | If non-blank, a VM tag command will be issued. This is used by RSCS to send jobs to various systems. |

2. SYMBOL TABLE (ONE LINE PER SYMBOL)

   SUBMIT allows symbolic parameter substitution in the model-JCL. These parameters can be

overridden by the user when the appropriate keyword parameter is specified on the /INFO statement or SUBMIT command. Symbolic parameters are described one per statement.

| | | |
|---|---|---|
| & | COL 1 | |
| SYMBOL | COL 2-13 | Name of the symbol. |
| MIN ABBR. | COL 15,16 | Number of characters for the minimum abbreviation. |
| TYPE | COL 18 | C-Character, N-Numeric, R-Route info. |
| | COL 19 | R-required, F-Fixed (see note 3). |
| MIN LENGTH | COL 21,22 | Minimum length of value. |
| MAX LENGTH | COL 24,25 | Maximum length of value. |
| TEXT LENGTH | COL 27,28 | Actual length of value (char only). |
| DEFAULT TEXT | COL 30-80 | See note 2. |
| DEFAULT VALUE | COL 27-34 | (Numeric only). |
| MIN VALUE | COL 36-43 | (Numeric only). |
| MAX VALUE | COL 45-52 | (Numeric only). |

3. HEADER JCL

   This is separated from the symbol table by a &/ statement. The symbolic variables which are to be replaced are coded as a "&" followed by the valid abbreviation of one of the symbols defined above. If the symbol is not followed by a comma, a bracket, or a blank, it must be terminated by a period.

4. TRAILER JCL

   This is separated from the header JCL by a &/ statement. It is output after all the user data has been sent.

**Notes**

1. Setting Access restrictions.

   The access restriction field can be set to a value 0 through 8. Zero indicates that anyone can use the procedure. If a value of 1 through 8 is specified, the SUBMIT checks that the appropriate JCL access option is set in the user profile. These options can be set in the user profile using the keywords JA1 through JA8. For example, in order to submit a job using model JCL with an access code of 4, the option JA4 must have been specified in the user's profile for the sign-on code.

2. Default text.

   The default text field can contain the actual default text or one of a number of special keywords described below. The text length field must specify the exact length to be used. A length of zero indicates that there is no default text regardless of anything in the text field.

| | |
|---|---|
| =USERID | the user's sign-on userid |
| =CODE | the user's file ownership id |
| =SUB | the user's subcode |
| =ROUTE | the user's default output destination (from ROUTE table) |
| =PW | the user's batch password (from code table) |

3. Symbol type.

   A symbol can be defined as *character* or *numeric* or *route information* by placing a C, N, or R in column 18. Character data is checked only for a valid length. Numeric data must fall within the range that has been specified. Also leading zeros will be omitted during substitution, within the limits defined

for the number's length. If a symbol is designated as R (routing information), SUBMIT will verify if the location specified is valid, if the ROUTING table is present in the link pack area.

A symbol can be designated as required by putting an R in column 19. If the user does not specify a value for this symbol on the SUBMIT command or /INFO statement, SUBMIT will prompt the user to enter a value. The text for the prompt is taken from the default text field. The text length field should be set to zero in this case to indicate that no default value is present and that the actual data appearing in the text field is to be used as a prompt. The first character of the prompt is used as a carriage control. If the carriage control is specified as "?", the response area will be blanked out in a manner similar to the password area at sign-on time. Only character variables can be designated in this way.

A symbol can be designated as fixed, (i.e. the user can't change it) by putting an F in column 19. This is usually used with the special default such as =CODE and =ROUTE so that SUBMIT will put in the appropriate value and at the same time disallowing the user from changing it.

# Processing Print Files and Batch Output

Output sent back to MUSIC from batch jobs, or created by the PRINT command, is managed through a single *output queue* by a group of utility programs. The internals of this *output queue* and the utility programs are detailed in this topic. Figure 8.2 illustrates the various components.

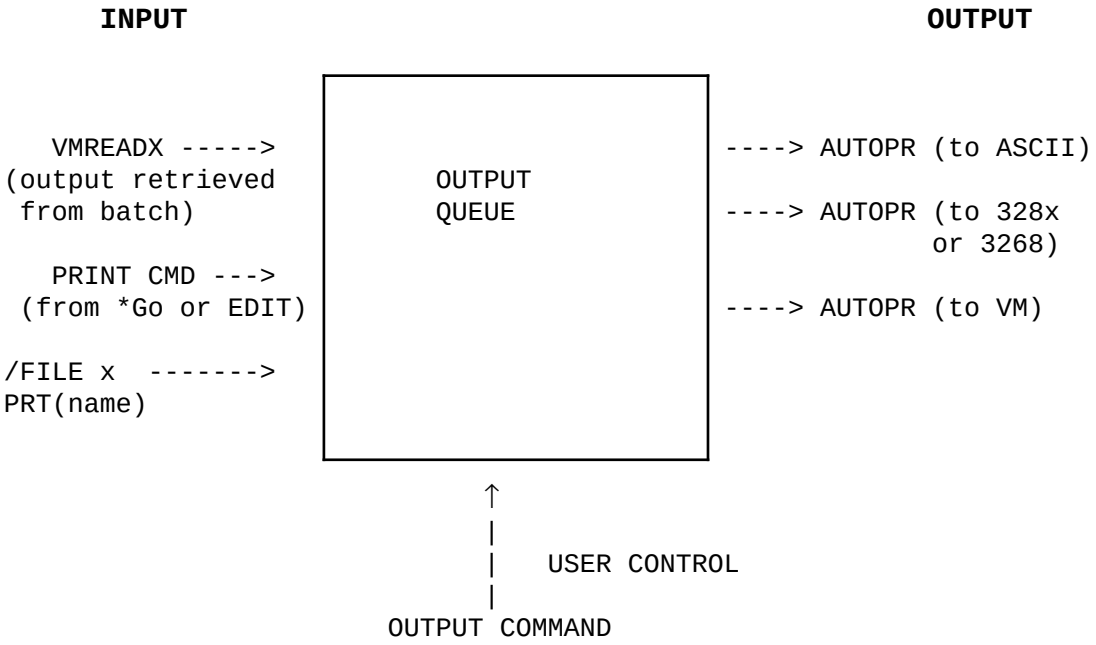


*Figure 8.2 - Output Queue Components*

The PRINT command can be used from *Go mode or the editor to schedule a file for printing at one of the printers defined to the system. This command creates a *print* file and places an entry into the output queue indicating who owns it and where it is to print.

The VMREADX program can also create print files. Operating systems such as CMS, VSE, VS1, MVS, and other MUSIC systems, can be set up to send job output back to a MUSIC system. Output to be processed by VMREADX should be spooled to MUSIC as class M files. VMREADX puts this output into *print* files and updates the output queue indicating that it is to be held for the user.

The OUTPUT command allows a user to inspect any files that are being held for the user or are waiting to print. The user can then purge these held files or schedule them to be printed at any of the printers defined to the system. Supported printers include local 3268 and 328x series printers, local or remote ASCII printers (with or without keyboards) and VM virtual spooled printers. The printers are driven by a service program AUTOPR which scans the output queue for files to print and sends the appropriate data stream to the printer in question.

Central to the entire subsystem is the OUTPUT QUEUE file which is effectively a list of all *print* files in the system. Each entry in this list contains the owner's sign-on code and the location code of the printer for which the file is destined. The actual output data is kept in separate Save Library files. These files are all stored under a special user-ID of $PRT and do not count against the owner's allocation.

## System Components

There are six major components in this system. Later on, each will be described in detail, indicating what must be done to get the various components working in the required environment.

| | |
|---|---|
| OUTPUT QUEUE | Contains a list of all *print* files in the system. (In file $PRT:PQ). |
| OUTPUT command | Allows user to inspect and print any output that is being held for the user. (Source in $PGM:OUTPUT.S). |
| PRINT command | Allows user to schedule the printing of a Save Library file at one of the printers on the system. (Source in $PGM:PRINT.S). |
| PQ command | Allows user to get a list of what files are scheduled to print on a given printer. (Source in $PGM:PQ.S). |
| AUTOPR program | Runs as a BTRM or AUTO-SIGNON program and drives the printers. A device dependant routine is called to do the actual I/O. (Source in $PGM:AUTOPR.S). |
| VMREADX program | Runs as a BTRM program and puts any print file spooled via VM by other operating systems into the output queue. (Source in $PGM:VMREADX.S). |

## Output Queue

The OUTPUT QUEUE is kept in a Save Library file ($PRT:PQ). Each Q entry corresponds to a logical record in the file. Since direct access is used, the file has record format F. Each entry contains information as to who owns the file and where it is to print. The actual data portion of the print file is kept in a separate Save Library file whose name is derived from the entry's position in the OUTPUT Q file. For example the data for 309th entry in the OUTPUT Q is kept in the file $PRT:PQ.00309. The first record in the file (entry 0) is maintained as a pointer to the next free entry. When this pointer reaches the last entry, it is reset to point to the beginning of the file. This has the effect of making the OUTPUT Q logically a circular file. The last entry in the file MUST contain only FF's (hex). Free entries contain 00's (hex). All access to the OUTPUT Q is done via the subroutine QSCANX and its entry points. (See $PGM:QSCANX.S for details.) The format of each used entry is described in file $MCM:PQENT.M.

### Enlarging the OUTPUT Q

The OUTPUT Q file distributed with the system has room for about 2000 entries. This should be sufficient for most installations. The file can easily be enlarged using the editor by inserting empty entries (i.e. entries containing X'00's) just BEFORE the LAST entry. The maximum number of entries that should be allocated

in the OUTPUT Q file is 99999. The following is an example to add 1000 empty entries:

```
*Go
edit $prt:pq;last;up;xin;insert 00;ain;dup 999;save
```

*Note:* This can ONLY be done when NO ONE else is accessing the Q. (i.e. VMREADX and AUTOPR programs must be cancelled prior updating the print queue file.)

## Output Management Facility

The OUTPUT Management Facility allows you to inspect the batch output queue from a terminal. To use this facility enter "output" in command mode. For a description of this facility press F1 (help) after the facility is invoked or refer to the *MUSIC/SP User's Reference Guide*.

## The PRINT Command

The PRINT command is used to send a file to a printer. The user's file is copied to a print file and an entry placed in the output queue. The print file will have a logical record length of 133 with carriage control added if required. The output will be printed whenever the service program (AUTOPR) handling the particular printer gets around to it. The format of the print command is as follows:

```
PRINT filename R(location) CC
```

*filename* is the name of the file to be printed. This is the only parameter that need be specified. Routing information can be given via the R parameter where *location* is the printer location name. CC should be specified if carriage control characters are provided by the user. If the installation has provided a ROUTING table in the Link Pack Area, the PRINT program will use it to set the default location (printer name), and also to verify that any user specified *location* is valid.

## Configuring the AUTOPR Program

This program runs the auxiliary printers. A copy of the program is set up to run on each of the printers defined on the system. It scans the MUSIC print queue until it finds some output routed to its printer, prints the output, and deletes it from the queue. The program must always be run under the code $MON unless special authorization for some other code is granted through the Routing Table. AUTOPR requires the FILES and SYSCOM privileges. If it is run from the file $PGM:AUTOPR, these are granted as program privileges, otherwise the code running the program must have these privileges.

The program can be used to drive dedicated printers on specific ports. These could be ASCII or 328x/3268 or 3262 (models 3 and 13) devices. These devices should be set up to be automatically signed on using the SIGNON parameter on the device specification for that address in the NUCGEN jobstream. The auto sign-on feature will attempt to sign on the code $MONsss, where the subcode *sss* corresponds to the device address of the terminal or printer in question. Before the printer will function, the above code and subcode must be allocated in the code table, and the appropriate AUTOPR program set as the auto-program for that code. The code must be assigned the privileges FILES, MAINT, LSCAN, and SYSCOM, and must have unlimited time: PRIME(NL), NONPRIME(NL), DEFTIME(NL), BATCH(0), PW(*NOLOGON). See the topic "Defining BTRMs and Auto Signon" in *Chapter 22 - System Programming* for a more detailed

discussion of the auto sign-on feature. 328x/3268 printers should be specified as 3270 devices in the NUCGEN. ASCII printers should be specified as TTYs. Examples are given below.

The program can also be used to send the print file from the queue to a user as mail. The recipient userid/ nickname is determined from the FORMS field for this output job, the ID=xxxx (where xxxx is an e-mail address or nickname or userid) in the TAG field for the route name MAIL in the system routing table, or is the owner of the output data. This can be used to send AUTOSUB output as mail to a user. No physical terminal is required, and the AUTOPR program is set up to run as a BTRM type terminal in the NUCGEN jobstream. On a BTRM, the system will attempt to sign on the code $MONsss, where the subcode sss corresponds to the device address specified for the BTRM in the NUCGEN. Before output data can be sent to users as mail, the above code and subcode must be allocated in the code table, and the appropriate AUTOPR program set up as the auto-program for that code. See the topic "Defining BTRMs and Auto Signon" in *Chapter 22 - System Programming* for a more detailed discussion on BTRMs.

The program can also be used to spool the print files from the queue to the VM system printer, RSCS, or other virtual machines. The *location* information is used to route the file to the appropriate VM printer. Each location name maps to a specific VM USERID. If the USERID is RSCS the location information is passed to RSCS via the print file's TAG. No physical terminal is required, and the program is set up to run as a BTRM type *terminal* in the NUCGEN jobstream. One such *terminal* can service ALL the different VM printers. On a BTRM, the system will attempt to sign on the code $MONsss, where the subcode *sss* corresponds to the device address specified for the BTRM in the NUCGEN. Before the printer will function, the above code and subcode must be allocated in the code table, and the appropriate AUTOPR program set up as the auto-program for that code. See the topic "Defining BTRMs and Auto Signon" in *Chapter 22 - System Programming* for a more detailed discussion on BTRMs. When you generate the MUSIC system a BTRM running AUTOPR is setup on address 010 to spool print files to VM using a virtual printer on address 017.

The program reads two parameter lines from unit 5. The first one is shown below. (The second one depends on the entry given on the TYPE parameter.)

| | |
|---|---|
| LOCS='name1','name2',... | The location name(s) defined for the particular printer. Printers can be assigned more than one name. Different printers can also be assigned the same name. For example if two printers were to service the same terminal room they should be given the same location name. |
| WAIT=nn | The number of seconds to wait if it finds that a print file is busy and there is nothing else to print. |
| TYPE='xxxx' | The type of printing device. This can be...<br>-328X (328x/3268 type printers)<br>-3286 (3286 model 1, no CR or FF support)<br>-TTY (ASCII async printer)<br>-VM (a VM spooled printer)<br>-MAIL (used for ROUTE MAIL support) |
| USERS='c1','c2'... | A list of up to 100 user codes that are to be authorized to send files to this printer. If a user is not authorized the print file is just deleted. If not specified all users can send files to this printer. |

When TYPE='VM', a second parameter statement must be included describing the virtual printer(s) to be used. Normally the VMID and CLASS parameters need not be specified, since this information is taken from the Routing Table entry for the named printer.

| | |
|---|---|
| UNITS=Zuad1,Zuad2,... | Unit address(es) of the virtual printer(s). Normally one printer is sufficient. These printers must also be defined in MUSIC's VM directory as virtual 1403 devices and must not have the same address as the batch printer which is defined in the NUCGEN. The "Z" indicates that the address is given in |

| | |
|---|---|
| | hexadecimal. The default is Z17 (hex 17) and is not the same as BATCH. |
| VMID='id1','id2',... | The VM USERIDs of the virtual machines to which the location names correspond. These must be entered in the same order as the printer location names specified in the LOCS parameter. Files routed *name1* will be spooled to *id1*, files for *name2* will be spooled to *id2* etc. By default *id1* is SYSTEM, the VM system printer, and the rest are set to RSCS. The values specified in VMID are used only if this information is not available from the Routing Table. The values specified in CLASS are used only if this information is not available from the Routing Table. |
| CLASS='c1','c2',... | VM print classes corresponding to the USERIDs specified in VMID. The default is class A. |

When TYPE='3286', the second parameter statement must be included describing the length and width of the forms.

| | |
|---|---|
| LINLEN=n | The number of characters in a line. The default is 132. |
| PAGELN=n | The number of lines on a page. The default is 66. |

## Examples

When creating the auto-program files for AUTOPR great care must be taken in entering the parameters, especially the quotes and the commas. It is recommended that you sign on to the code and verify that the auto-program starts up all right, before attempting to run the real printer. (To sign on to that userid, you must specify a password other than "*NOLOGON".)

1. In this example AUTOPR is set up to spool output to the VM system printer as class A print files.

   - Allocate the code $MON SC(010) specifying AUTO1 as the auto-program, and privileges FILES, MAINT, LSCAN, and SYSCOM, and unlimited time: PRIME(NL), NONPRIME(NL), DEFTIME(NL), BATCH(0), PW(*NOLOGON).

   - Set up the file $MON:AUTO1 to contain the following.

     ```
     /INCLUDE AUTOPR
     LOCS='SYSTEM',WAIT=20,TYPE='VM'
     VMID='SYSTEM',UNITS=Z17,CLASS='A'
     ```

   - Define a BTRM on address 010 by adding the following device card to the NUCGEN.

     ```
     BTRM 010
     ```

     Note: These are set up by default when you install MUSIC.

2. In this example AUTOPR is set up to spool output to RSCS printers REMOTE1 and REMOTE2 as class B print files. Output for CENTRAL is spooled to the VM system printer as class A files and output for CMS1 is sent to the VM user CMS1 as class X files. The files are spooled via a virtual printer on address 017.

   - Allocate the code $MON SC(010) specifying AUTO1 as the auto program, as in example 1.

- Set up the file $MON:AUTO1 to contain the following.

```
/INCLUDE AUTOPR
LOCS='REMOTE1','REMOTE2','CENTRAL','CMS1',WAIT=20,TYPE='VM'
VMID='RSCS','RSCS','SYSTEM','CMS1',
UNITS=Z17,CLASS='B','B','A','X'
```

- Define a BTRM on address 010 by adding the following device card to the NUCGEN.

```
BTRM 010
```

- Add the names REMOTE1, REMOTE2, CENTRAL, and CMS1 to the ROUTING table.

3. A 1200 baud ASCII printer on address 047 which has location name of ROOM112.

   - Allocate the code $MON SC(047) specifying AUTO47 as the auto program, as in example 1.

   - Set up the file $MON:AUTO47 to contain

```
/INCLUDE AUTOPR
LOCS='ROOM112',WAIT=20,TYPE='TTY'
```

   - Add the following device card to the NUCGEN jobstream and do a NUCGEN. ASCII printer should always be specified as DIALUP if they have no keyboard, since if DIRECT is specified MUSIC waits for an interrupt from the keyboard before starting output.

```
TTY 047 DIALUP,1200,SIGNON
```

   - Add the name ROOM112 to the ROUTING table.

4. A 3287 printer on address 0DC which is known as PRINTER6.

   - Allocate the code $MON SC(0DC) specifying AUTODC as the auto program, as in example 1.

   - Setup the file $MON:AUTODC to contain

```
/INCLUDE AUTOPR
LOCS='PRINTER6',WAIT=20,TYPE='328X'
```

   - Add the following device card to the NUCGEN.

```
3270 0DC DIRECT,SIGNON
```

   - Add the name PRINTER6 to the ROUTING table.

5. In this example AUTOPR is set up to send mail to user $000 if the route is MAILX and to the output data owner if the route is MAIL.

   - Allocate the code $MON SC(010) specifying AUTO1 as the auto-program, as in example 1.

   - Set up the file $MON:AUTO1 to contain the following.

```
/INCLUDE AUTOPR
LOCS='MAIL','MAILX',WAIT=20,TYPE='MAIL'
```

   - Define a BTRM on address 010 by adding the following device card to the NUCGEN.

```
BTRM 010
```

- Add the names MAIL and MAILX to the ROUTING table. These entries would be defines as follows:

```
ROUTE NAME=MAIL,TYPE=MUSIC,CLASS=A
ROUTE NAME=MAILX,TYPE=MUSIC,CLASS=A,TAG='ID=$000'
```

6. The AUTOPR utility, used for printing files and job output, has support for the HP LaserJet 4Si printer. It is considered a TYPE=VM printer, and is specified by the value 1 in the new XTYPE parameter. Here is an example of AUTOPR control statements for handling a system printer name and two HP 4Si printers called HP and HP2:

```
/INC AUTOPR
LOCS='SYSTEM','HP','HP2',TYPE='VM'
VMID='SYSTEM','RSCS','RSCS',
UNITS=Z17,CLASS='A','A','A',
XTYPE=0,1,1
```

**AUTOPR Notes**

AUTOPR is usually run under the code $MON. It can be run from another user code if the code is authorized by the system administrator in the ROUTING table. The program will only process files that have been sent to the user's printer and will not allow access to other print files in the system. This mechanism allows users to operate their own printers. However, since regular users do not have access to the auto sign-on feature, this technique is limited to ASCII printers with keyboards.

The following are the most common causes of problems with AUTOPR.

- The userid $MONsss is not correctly defined in the code table. This can be checked by signing on to the code from a terminal and verifying that the auto-program starts correctly and does not produce error messages. (To sign on to that userid, you must specify a password other than "*NOLOGON".) Make sure that sss is defined as the subcode: ADD $MON SC(sss) etc.

- The parameters specified for AUTOPR are incorrect. This can be checked by signing on to the code from a terminal and verifying that the auto-program starts correctly and does not produce error messages.

- The BTRM or auto-signon terminal is not correctly defined in the NUCGEN. Check that the device address specified in the NUCGEN matches the subcode assigned to $MON.

- The program appears to be running but no files are printing. If printing via VM, check that a virtual printer is defined, in addition to the batch printer, whose address matches the UNITS parameter. This printer should be defined as a virtual 1403. If printing to an ASCII or 328x printer, check that the device is correctly connected to the system.

# Configuring the VMREADX Program

VMREADX processes VM spool files that are send to the MUSIC/SP virtual machine. Print files are put into the output queue. Reader files are put into a reader queue where they are processed by MUSIC/SP's MAIL Facility.

VMREADX replaces the older VMREAD program. The major difference between the two programs is in their handling of electronic mail from outside the system. VMREAD delivers mail in the old MEMO format. VMREADX uses the MAIL Facility. In this respect the two programs are not compatible. If, during some conversion period, you wish to use the MEMO program to receive mail from outside the system, use VMREAD instead of VMREADX. Do not specify the RDRCOD parameter if VMREAD is used. Other parameters are the same.

The following information pertains to VMREADX's processing of print files. The *jobname* field in the Q entry is set from the FILENAME field of the spool file. This is important, as MUSIC bases the ownership of a print file on the first four characters of this *jobname*. VSE and VS1 can be set up to set this FILENAME field automatically from the assigned jobname. Under CMS, SPOOL and CLOSE commands can be used to set this name field. If a route destination of MUSIC is specified for a MUSIC batch job, MUSIC will automatically spool the output back to itself, setting the name field directly from the user code of the batch job. Other systems like MVS may have to be slightly modified to use the NAME parameter when closing VM print files. If the FILENAME field is blank, *NO-OWNER is used. The print file will by default be added to the print queue with a printer location name of HOLD, indicating that the file is being held for inspection by the user. If the spool file's FILETYPE field matches a MUSIC printer location name the file will be sent directly to the printer.

A copy of VMREADX must be run for each spool file class you wish to process. Normally two copies of VMREADX are required, on for class M and one for class A. Class M is used for print files and mail. Class A is used for mail.

No physical terminal is required, and the program is setup to run as a BTRM type *terminal* in the NUCGEN jobstream. On a BTRM, the system will attempt to sign on the userid $MONsss, where the subcode *sss* corresponds to the device address specified for the BTRM in the NUCGEN. The userid $MONsss should be allocated in the code table with the appropriate VMREADX program set up as the auto-program. VMREADX requires that the code have the MAINT, FILES, and SYSCOM privileges, and unlimited time: PRIME(NL), NONPRIME(NL), DEFTIME(NL).

The program reads the following parameters from the input stream.

| | |
|---|---|
| SYSCOD='$PRT' | Code under which the print files are to be stored. |
| RDRCOD='$VMR' | Code under which the reader files are to be stored. |
| DELAY=nnn | Number of seconds to wait between output scans. Since VMREADX wakes up automatically when a file arrives this can be set to a large number. |
| READER=Zaddr | Unit address of the virtual reader VMREADX will use ("Z" means hexadecimal). This reader should be defined in MUSIC's VM directory. |
| DAYS=nnn | Number of days that output is to be kept before it is automatically deleted. This applies only to print files. |
| MSGS=n | Output unit number for messages from this program or 0 meaning send messages to the operator console. Normally n would be 0 or 6. |
| TRACE=n | Used in a debugging situation to set the tracing level. If n is 0 no tracing is done. |

## Example

The following example is how VMREADX is set up by default after MUSIC has been installed. The BTRM on address 011 is used for VMREADX. By default VMREADX looks for a virtual reader on address 018. Note that the BTRM address is in no way related to the address of the virtual reader.

- The userid $MON SC(011) should be allocated with AUTO2 specified as the auto-program, and with privileges FILES, MAINT, LSCAN, and SYSCOM, and with unlimited time: PRIME(NL), NONPRIME(NL), DEFTIME(NL).

- The file $MON:AUTO2 should contain

```
/INCLUDE VMREADX
SYSCOD='$PRT',RDRCOD='$VMR',READER=Z18,DAYS=7,DELAY=600
```

- The NUCGEN job should have a BTRM defined on address 011.

- The VM directory entry for MUSIC/SP should have a class M reader defined on virtual address 018.

## Notes

Any print file spooled to MUSIC class M will be processed by the VMREADX program and placed in the MUSIC output queue. The DIST field specified on the CLOSE command for the print file will be used to indicate file ownership. The FILETYPE field can optionally be specified on the close command to send the file directly to one of MUSIC's printers. Systems wishing to send output to MUSIC need only fulfill these conditions.

CMS — Issue the appropriate SPOOL and CLOSE commands, for example:

```
SP PRT CONT
... Write output to printer ...
SP PRT MUSIC CL M NOCONT
CLOSE PRT DIST userid NAME jobname prtname
```

where *jobname* is to be the jobname associated with the MUSIC print file, *prtname* (optional) is the name of a MUSIC printer, and *userid* is the first 8 characters of the MUSIC file ownership id (the userid without the subcode, if any). If *prtname* is omitted, the file is held in MUSIC's output queue.

VS1 — If VS1 is generated with the VM HANDSHAKING option, it will automatically close any print file using the JOBNAME as the name field on the CLOSE command. Generate a virtual printer to handle the output to go to MUSIC. This printer can be setup to process only a special SYSOUT class, or associated with a specific location. The user would send output to MUSIC by specifying the special SYSOUT class or printer location in the JCL. When the VS1 machine logs on, the appropriate VM commands should be issued to spool this printer to MUSIC Class M. The VM DIST (distribution) field should be set to the first 8 characters of the MUSIC ownership id (the userid without the subcode, if any).

VSE — If VSE is generated with the VSE-VM/370 Linkage enhancements, it will automatically close any virtual print files. As with VS1, a specific printer should be chosen for MUSIC output and the appropriate commands issued when VSE is logged on to spool this printer to MUSIC Class M.

The following are the most common causes of problems with VMREADX.

- The code $MONsss is not correctly defined in the code table. This can be checked by signing on to the code from a terminal and verifying that the auto-program starts correctly and does not produce error messages.

- The parameters specified for VMREADX are incorrect. This can be checked by signing on to the

code from a terminal and verifying that the auto-program starts correctly and does not produce error messages.

- The BTRM is not correctly defined in the NUCGEN. Check that the device address specified in the NUCGEN matches the subcode assigned to $MON.

- VMREADX appears to be running but the spool files are not being processed. Check that the address specified in the READER parameter matches that of a virtual reader and that the reader is spooled to correct class (M or A).

- The reader files disappear but nothing shows up in the output queue. Check that MUSIC's batch reader, (the one defined in the NUCGEN), is spooled to class J. If it is spooled to class M, A or *, the files will be processed by batch rather than VMREADX and deleted since they are not valid batch jobs.

## Setting up the Routing Table

The load library module $ROUTING must be set up to provide a table of valid printer location names. When this module is included in the LPA, programs such as SUBMIT, PRINT and OUTPUT, use this table to set default locations and to verify that any user specified location is valid. If it is removed from the LPA, no checking can be done and users can specify any eight character sequence as a printer name.

The subroutine ROUTE (see $SUB:ROUTE.S for details of calling sequence) is provided to access this information. It has three functions: to return the default printer name for a specific terminal; to verify if a printer name is valid; and to determine if a user is authorized to run the AUTOPR program for a specific printer.

The $ROUTING module has provision for a system default location, usually the machine room printer. There is also a table of printer names versus REAL terminal address range. This can be used to set a default printer name for terminals in a particular terminal room or building, such that when a user submits a job, output is automatically returned to the location associated with the terminal from which the job was submitted. Printer location names which are not associated with particular terminals must also be included in this list with a dummy address range (X'FFFF'-X'FFFF') if they are to be recognized as valid by the ROUTE subroutine.

The REAL address of the terminal is used in the $ROUTING module. For remote 3270's, this is the line address of the control unit, thus all terminals on that control unit will have the same real address. If a terminal is connected via VM PASSTHRU, the *logical device* address assigned by PASSTHRU is used. This may be different each time the terminal connects via PASSTHRU.

This REAL address is contained in the RDEV field of the TCB and is listed by the WHOALL program. The real address was chosen over the virtual (MUSIC's) address due to the fact that the virtual address can be different, each time the terminal *dials* in from VM.

The default table defines SYSTEM and MUSIC as valid printer names. Output sent to SYSTEM will be sent to VM's system printer. Output sent to MUSIC will be placed on hold in the output queue for inspection by the user via the OUTPUT command. If other printer names are required they must be added to the routing table. The module containing the table is generated from assembler language source by the following procedure.

1. Edit the file $PGM:$ROUTING.S to fill in the appropriate address ranges and routing information. The comments in the source explain the table format. All locations to which printed output can be directed should be included in the module.

2. Assemble the module by executing the file. The object deck will automatically be saved.

3. Linkedit the module by executing the file $PGM:$ROUTING.LKED.

4. Place the new table in the load library by executing the file $PGM:$ROUTING.LD.

5. The new routing table will be loaded in to the LPA at the next IPL.

The system utility program ROUTETABLE can be used to display the contents of the routing table. See the program description in *Chapter 17 - System Utility Programs*.

# Chapter 9. Electronic Mail Facility

## Overview of the Mail Facility

The MAIL Facility allows you to send and receive electronic mail to and from other computer users. To use this facility enter "MAIL" in command mode. For a description of how to use this facility press F1 (help) once the facility is invoked or refer to the *MUSIC/SP Mail and Conferencing Guide*.

## Configuring MUSIC's MAIL System

This section is intended for system programmers and administrators who are responsible for configuring the MUSIC Mail program so that it can communicate with systems outside of MUSIC. If MAIL is to be used only internally, within a MUSIC system, the configuration process is still required to set up the RDMAILER BTRM to handle recurring, postdated and forwarded mail, but you can ignore the information about external mail, VM, RSCS, SMTP, and mailers.

Although communicating with the variety of outside mail networks is in principal quite simple, the large number of options available makes choosing a particular set of options a daunting task for those unfamiliar with current electronic mail technology. On MUSIC/SP, the MAIL.CONFIG program is used to set the options for the MAIL and RDMAILER programs. A MAILER.PROFILE file should be created and one or more VMREADX programs are setup to read incoming mail from VM. On VM, depending on the configuration, mailer tables may require adjusting, RSCS may require modification, or SMTP may require configuration changes. After describing the options available in configuring the Mail Facility, emphasis is placed on specific configurations that are most common. These configurations are:

1. MUSIC/SP and CMS on a single processor.
2. Existing network connection with modification to RSCS.
3. Existing network connection to the Internet with SMTP configuration changes.
4. Using a VM mailer.

## The Postmaster

Each site that runs electronic mail should designate an individual to act in the capacity of postmaster. In general, the postmaster should monitor mail usage and keep up to date on issues concerning electronic mail. This can include the development, dissemination, and enforcement of a mail code of conduct, as well as knowledge of the MUSIC Mail facility, mail networks, and list-servers.

The postmaster must, on a daily basis (or as required), monitor and when appropriate, respond to mail delivered to the postmaster code ($EMP). This is where notification of mail delivery problems and other general inquires about mail and user addresses, from both internal and external mail users, is deposited. Mail sent to "postmaster" will be placed in the $EMP mail box. A useful procedure is to make the individual's (postmaster) userid a surrogate for $EMP. This will allow viewing and sending of mail on behalf of the postmaster userid (code), without having to sign on to it.

The Postmaster Mail Filter program can be set up to automatically handle mail in the postmaster's mailbox. This program can unsubscribe users from mailing lists and perform a number of other tasks. See the topic Mail Filter Program later in this chapter.

The Postmaster is known by the MUSIC/SP Mail facility by a number of names or userids. All mail addressed to the following userids is delivered to the postmaster: postmaster, postmast, system, mailer, maint, support, root, and all userids starting with $EM.

The postmaster should carefully monitor the file usage by the mail system. This can be done with the MSTAT program (described later). It is recommended that the MAIL.CLEANUP program be run monthly. However, monitoring may suggest a more frequent schedule.

The postmaster should review the following topics and help determine the site's requirements.

- Review this chapter on the Mail facility so as to have a broad understanding of the components and flow of the mail facility and other related issues.
- Review RDMAILER and MAIL parameters with consideration to customizing it for your site.
- Review and become familiar with using the MAIL program, with the expectation that you will have to consult and educate your user community on mail usage.
- Review the items on the ADMIN 4 5 Configure/Administer The Mail Facility menu.

- The setting of prime time and the allowed size of mail text files to be send to remote sites during prime time.
- The system wide expiry date. This is a default amount of days used to set the mail's expiry date. It is very useful in keeping file usage down.
- User-specific limits for expiry date to override the system wide expiry date on a user or code table TYPE basis.
- User-specific mail item limits which limit a user to viewing a number of mail items and forcing the user to delete some of the viewed items before more items from the mailbox are presented.
- A site email policy and add it to the MAIL FAQ item on Mail's main menu (item A).
- Review the delivery of autoforwarded, postdated, and recurring mail since these items are given to RDMAILER for delivery (ie they reside in $EMD's mailbox until they are delivered, expire or are expired). You may need to deal with these types of mail.

# Mail System Components

There are a number of components that work together to provide access to electronic mail networks. Some of these are part of MUSIC/SP, others run under VM.

Figures 9.1 and 9.2 later show the relationships between the components.

## MAIL

The MAIL program is the user interface to the electronic mail system. Internal mail is managed through a system of mailboxes, text files, and distribution files. By default, these files are maintained on the userids $MBA to $MBG, and $MDAA to $MD99. The mailboxes for individuals have file names of the format $MB?:Buserid where ? can be A to G. The text and distribution files have rather cryptic names that have significance only to the mail system.

When mail is sent outside the system, MAIL transfers the file to a special outbound mailbox designated for outside mail. This mailbox has the file name $MBE:B$EMD or $MBD:B$EMD000 depending on whether the namelist parameter PCODE is set to $EMD or $EMD000. Note that outbound address checking is not

done by MAIL, so errors that occur in delivering the mail are not reflected to the user right away.

Similarly, mail automatically forwarded to another userid either local or remote, or mail that is postdated or recurring, is also transferred to the special outbound mailbox.

## BIGMAIL

BIGMAIL is a version of MAIL with a 3000K region size.

## RDMAILER

The RDMAILER program runs in the background as a BTRM and is responsible for sending mail outside the system and also for receiving mail from the outside. It also handles postdated mail, recurring mail and forwarded mail. RDMAILER controls the special outbound mailbox. When it finds an item in this box it verifies the destination address and spools the file to the appropriate service machine on VM for ultimate delivery. Usually SMTP, RSCS, or a mailer virtual machine provides this service. If RDMAILER determines that it can't deliver the mail it sends an error to the original sender, and depending on the type of error, will also send a note to the *postmaster*, which by default is the userid $EMP. In the case where a VM mailer or SMTP is being used to deliver the mail, RDMAILER will usually leave the verification of the destination address up to the VM mailer or SMTP respectively.

Mail coming from the outside eventually arrives on MUSIC/SP as a virtual reader file. The program VMREADX (see below) places these files into the reader queue used by RDMAILER. RDMAILER uses information from the queue entry, the file's tag, and the file's content, to determine where the file is from and to whom it should be delivered. If a file can be delivered to a local user, an entry is added to the appropriate mailbox. If the mail cannot be delivered, a message is sent to the sender if possible, and depending on the severity of the problem, the postmaster is informed.

The operation of RDMAILER is influenced by the parameters specified to the MAIL.CONFIG program and by the contents of the file $EML:MAILER.PROFILE. MAIL.CONFIG defines the local environment: the local time zone, the node name of the MUSIC/SP machine, the name(s) of the RSCS machine(s), and the local node name of the processor. $EML:MAILER.PROFILE defines the external electronic mail network. It is the same format as the mailer profile used by the Columbia VM Mailer on BITNET. This is described later.

From 1 to 10 auxiliary RDMAILER BTRMs can be configured to deliver mail received from the outside (ie incoming mail processing). See the sections Configuring Mail and Mail Administration for further information on this topic.

## VMREADX

VMREADX runs in the background as a BTRM. It reads in virtual reader and print files spooled to MUSIC/SP from other virtual machines running under VM. It puts incoming reader files into MUSIC/SP's reader queue under the userid $VMR. It is from this reader queue that RDMAILER gets the incoming mail. Incoming print files are put into the output queue under the userid $PRT. One consequence of the design is that VMREADX can only process one class of files at a time, so if you have mail arriving on both class A and M, (not an unusual situation), you will have to run two copies of VMREADX, one to process class A and one for class M.

*Note:* VMREADX is the enhanced version of the VMREAD program used with the original version of MAIL. When VMREADX is installed, the BTRM that runs the old VMREAD program should be disabled by deleting its $MONxxx userid or removing the BTRM specification for VMREAD from the NUCGEN. (xxx - BTRM device address.)

## RSCS

RSCS is the component of VM that transfers electronic mail between physical machines in a network of VM computer systems. One drawback is that the transfer mechanisms of RSCS assume a two level addressing scheme of USERID/NODE where the NODE represents a particular CPU, and USERID is the VM logon ID of the virtual machine. No allowances are made for the fact that the virtual machine, for example MUSIC/SP, can support hundreds of users. Two solutions are available.

1. Operating above the RSCS transport mechanism is a standard electronic mail format in which the sender and recipient are identified in the envelope and text of the mail itself. Thus, a remote site need only use RSCS to deliver mail to the MUSIC/SP virtual machine. RDMAILER figures out who the recipient is from the mail envelope. In this case there is no need for any modifications to RSCS. This method is recommended for new MUSIC/SP sites. Those that are already defined on BITNET may find it more convenient in the short term to keep using method 2 (below) rather that redefine their system to the network.

2. McGill offers some suggested modifications to RSCS to allow the MUSIC/SP virtual machine to appear as a distinct NODE in the RSCS network. Sites who have applied these modifications to allow their MUSIC/SP system to communicate with BITNET and other networks can leave them in place and replace MEMO with MAIL without changing their sites definition in the various tables that describe the network.

## VM Mailers

A number of programs are available to provide network electronic mail services to VM processors through an RSCS network. Most support an electronic mail format that consists of BSMTP commands and addresses comprising the mail envelope and RFC822 headers in the mail text itself. This standard allows a variety of networks to exchange electronic mail.

If your site runs a mailer program, the MUSIC/SP MAIL system can take advantage of the services it provides. RDMAILER will simply send any outbound mail to the VM Mailer. The mailer verifies the recipient address, provides the appropriate envelope and routing, and passes the file to RSCS. If a problem occurs, the mailer will send a message to the sender on MUSIC describing the problem. In order for MUSIC/SP to use the mailer's facilities, you must usually identify the MUSIC/SP virtual machine to the mailer in some way.

The Columbia VM Mailer is very popular on BITNET. It uses a "MAILER PROFILE" file to describe the network topology. The VM node and userid of the MUSIC machine must be entered in the "INCOMING:" section of this file, if the mailer is to accept mail from MUSIC/SP for delivery to the network. An entry must also be included in the "OUTGOING:" or "DOMAINS:" section of the file if the mailer is to deliver mail to the MUSIC/SP system. Since other network nodes must also have this information, you should contact the network coordinators (BITNIC) to have your MUSIC/SP node defined to the network. There are a number of advantages to using the VM mailer as the gateway to the outside world, the most important being that you only have to maintain one set of network topology tables.

If you do not run a mailer service machine on VM, RDMAILER can provide all the required services.

## SMTP

SMTP is the Simple Mail Transfer Protocol client and server virtual machine on VM that transfers electronic mail to/from other SMTP machines in the Internet. SMTP is a part of the VM TCP/IP program product (5735-FAL).

SMTP can be set up to communicate with MUSIC/SP's RDMAILER program. This allows MUSIC/SP users

using the MAIL program to exchange mail with others on the Internet.

There are a number of things to consider in this setup. See the Sample Configurations section later and the file $TCP:TCPIP.SMTP.DOC for further details.

This setup does not use the TCP-to-RSCS Gateway interface for SMTP.

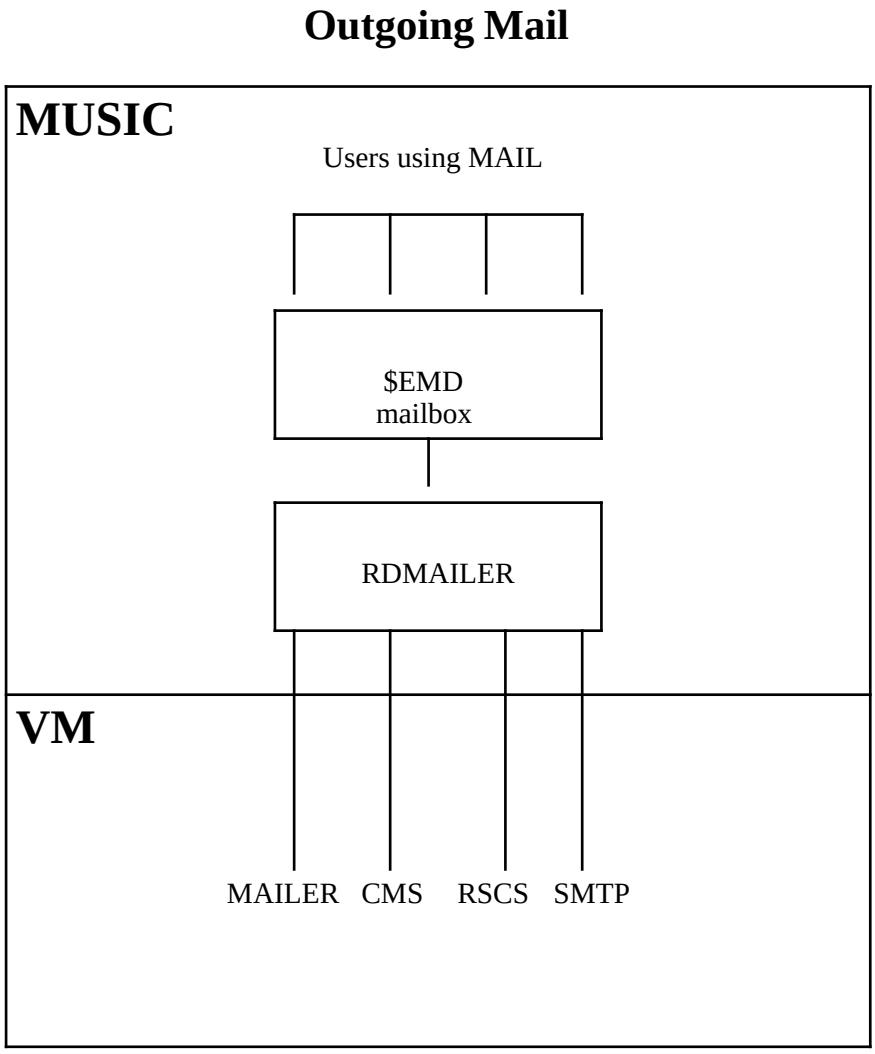


*Figure 9.1 - Information flow for outgoing mail.*

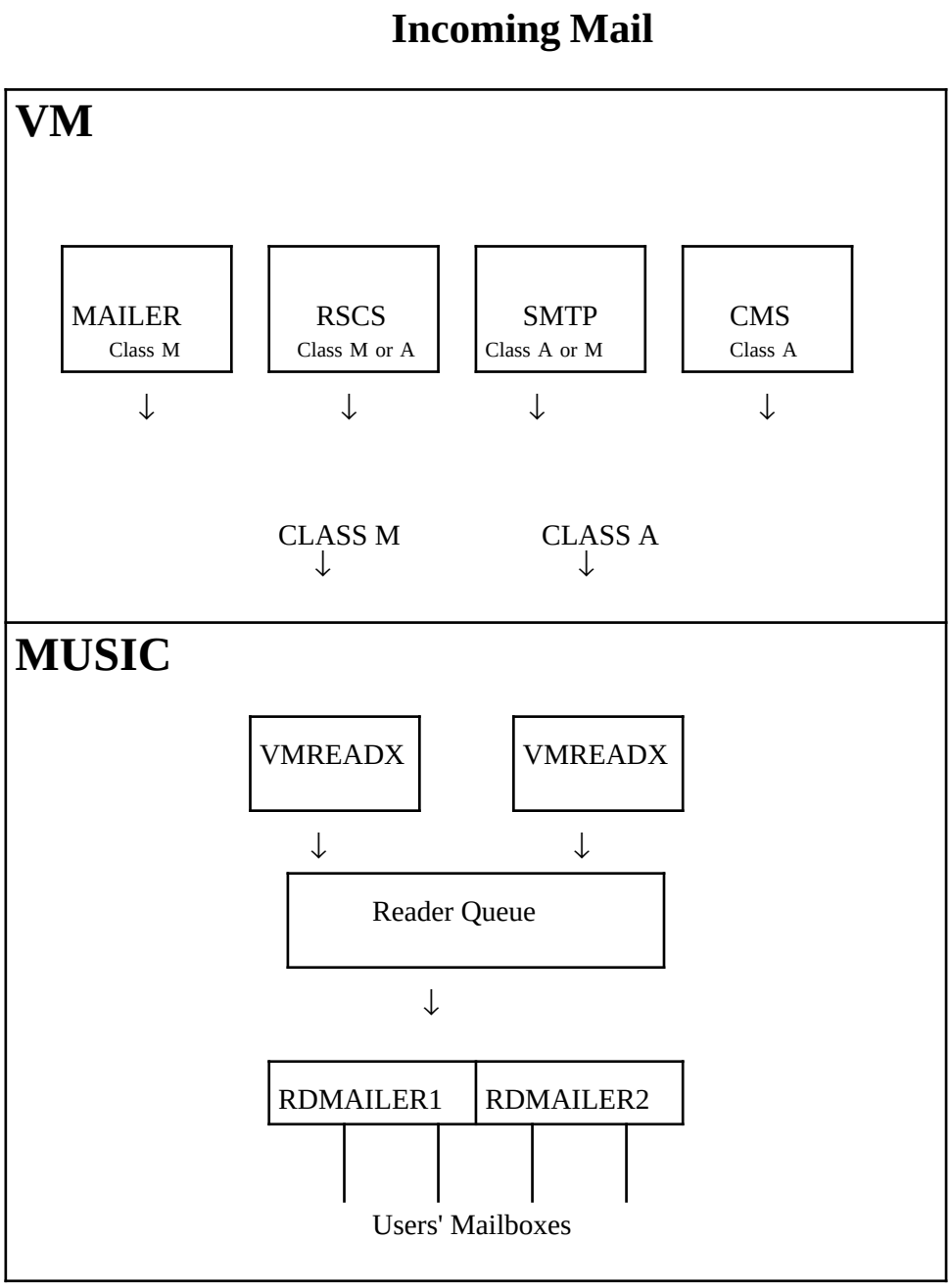


*Figure 9.2 - Information Flow for Incoming Mail*

## SENDFILE

SENDFILE is used to send a copy of a file to another MUSIC or CMS user on your computer, or on other computers that are connected to yours via RSCS. Sendfile does not currently use the nickname file. Therefore, you will have to specify the exact userid and VM node name. See the SENDFILE command in the *MUSIC/SP User's Reference Guide* for usage details.

Files sent to MUSIC via SENDFILE are received via MAIL. The mail item has a subject line specifying it is a sendfile.

## GETMINFO

GETMINFO invokes the GETMAIL utility program with the namelist parameter INFO=TRUE. This program gets a list of information for the user's incoming mail items and stores this list to a file.

GETMINFO has an entry in the system LOOKUP table with the FILES and CODES privileges to access the user's mailbox which is not stored on the user's code.

At program termination, the end of job return code is set via CALL EOJ(x).

## GETMAIL

GETMAIL invokes the GETMAIL utility program with the namelist parameter INFO=FALSE. Thus, this program gets your incoming mail and stores the text in a file.

This program has an entry in the system LOOKUP table with the FILES and CODES privileges to access the user's mailbox which is not stored on the user's code.

At program termination, the end of job return code is set via CALL EOJ(x).

## SENDMAIL

SENDMAIL invokes the SENDMAIL utility program which is the fast-track method to send a piece of mail. SENDMAIL requires that the text you wish to send already exist in a file.

This program has an entry in the system LOOKUP table with the privileges FILES, LSCAN, CODES, and SYSCOM to do the mail send.

At program termination, the end of job return code is set via CALL EOJ(x).

## $EML:PREMBK

PREMBK is a user friendly, front end to the Mailbook program that allows users to enter a mailbook file name. Then it calls the Mailbook program. PREMBK is invoked from the MAIL main menu (option 7) if a file name is not entered. If a file name is entered with the option, the Mailbook program is invoked directly.

## MAILBOOK

MAILBOOK invokes the MAILBOOK utility program which allows you to view your mail logs (also known as notebooks, mailbooks or folders) that you have created previously by either copying your outgoing mail to a file or "XLOGDELing" your incoming mail to a file. You can enter the file name of the mailbook when you invoke the program. If you do not, the PREMBK program will be invoked where you can enter a mailbook filename. The MAILBOOK program can also be invoked from the MAIL main menu (option 7). The MAILBOOK program user interface has the same look and feel as the MAIL program.

The MAILBOOK program is configured with MAIL.CONFIG. See the section "Configuring Mail" for further details. The MAILBOOK program can be used as a general viewer for files which have a separator line (73 equal signs) between items. The MAILBOOK program can also read mail list digests where the mail item is composed of a number of individual mail items. The namelist parameters COPY=t or f, DELETE=t or f, and POST=t or f can be used to control those functions within the program. COPY allows or disallows copying the item to a file. DELETE allows or disallows deleting the item from the mailbook file. POST allows or disallows all send functions of the MAILBOOK program. POST=t also sets

DELETE=f. Setting POST=t would make this a POST only mailbook.

## MAIL.CLEANUP

The MAIL CLEANUP program, $EML:MAIL.CLEANUP, is designed to delete mail text and distribution files that have expired. See the topic "Running the MAIL.CLEANUP Program" for more details.

## MAILBOX

This program displays the mailbox filename for a given userid. If the userid is not given, the mailbox filename for the signed-on user is displayed. ADMIN 4 5 7 can be used to run this program. The program is invoked manually as follows:

```
MAILBOX userid
```

## $EML:DIRECT

The DIRECT program, MAIL main menu option 5, allows you to create, change, and remove entries from your private directory file, @NAMES. This file is used by the MAIL facility to enable you to refer to private nicknames without having to remember userids.

## $EML:DIRECT.PRINT

The DIRECT.PRINT program is used by the DIRECT and DIRECT.PUBLIC programs to create a readable copy of the directory in the file @NAMES.LISTsss which the user can print on the printer of their choice. If the userid has a subcode "sss", then it is suffixed to the file name. Otherwise, the file name is @NAMES.LIST.

## $EML:DIRPUBL

DIRPUBL is a public program that allows users to browse the public directory. MAIL main menu option 6 invokes this program.

## $EML:DIRECT.PUBLIC

$EML:DIRECT.PUBLIC program allows you to create, change, and remove entries from the public directory file, $EML:@NAMES. This file is used by the MAIL facility to enable you to refer to public nicknames without having to remember userids.

It is important to understand the search order of the nickname files. When a user sends mail to someone, the MAIL facility attempts to resolve the name given for that person. The name is resolved by searching the user's nicknames file, then searching the public nicknames file, and then trying the name as a userid (ie the system code table). If the name is still not resolved to a userid(s), the user is told that the name is invalid.

$EML:DIRECT.PUBLIC works exactly like the Mail Directory program, DIRECT, also known as the private directory program. Thus, the description of that program in the Mail and Office Applications Guide can be used as a reference.

There is no default public directory. You must create one for your site using $EML:DIRECT.PUBLIC if you wish to create public nicknames.

$EML:DIRECT.PUBLIC requires the FILES privilege to run. ADMIN 4 5 5 can be used to run this program.

### $EML:URL

The URL program is invoked when a URL hotspot in MAIL or MAILBOOK is processed. The URL program parses the URLs and calls the appropriate client program to process the URL. The MAIL help topic URL gives further details of URL support and the Mail facility.

### $EML:FMAIL

FMAIL is a version of the MAIL program that exits the MAIL program when a user returns to the MAIL main menu from any other MAIL function. This program can be used to present a tailored view of the MAIL program. See the MAIL Administration section for the topic "How to create your own MAIL menu" for further information.

### $EML:LM

LM, also known as LIST MANAGER, is a facility that allows users to manage their subscriptions to BITNET and Internet mail lists. MAIL main menu option 8 invokes this facility.

### XTELL

The XTELL command is the same as the TELL command except that the message is displayed on the user's screen even if the user has suppressed messages by issuing the /MESSAGE OFF command.

---

## Configuring MAIL

### 1. Allocate BTRMs to run VMREADX and RDMAILER

Add the following BTRM specifications to the NUCGEN job and create a new nucleus.

```
BTRM 013-017
```

This adds five BTRM's, three for RDMAILER and two for VMREADX. The virtual addresses shown (013-017) need not be used. Just make sure that the ones you choose do not correspond to any real devices and that they match the addresses later specified in the MAIL.CONFIG program.

### 2. Allocate a Virtual Reader

Allocate a virtual reader for each of the VMREADX programs by adding the following entries to MUSIC's VM directory.

```
SPOOL 18 2540 READER M
SPOOL 19 2540 READER A
```

The above example uses 18 as a class M reader and 19 as a class A reader. Note that the addresses of the virtual readers must NOT be the same as the BTRMs.

## 3. Run MAIL.CONFIG

Before running this program cancel any running BTRM programs. If this is not done you may get "FILE IN USE" messages when the MAIL.CONFIG attempts to generate the MAIL program files. These files include:

- $EML:MAIL
- $EML:MPROF
- An auto program file for RDMAILER
  0 to 10 auto programs for auxiliary RDMAILERs
- Two auto program files for VMREADX
- $EML:RDMAILER
  0 to 10 $EML:RDMAILERnn for auxiliary RDMAILERs
- $EML:GETMAIL
- $EML:GETMINFO
- $EML:SENDMAIL
- $EML:MAILBOOK
- $EML:MAIL.CLEANUP
- $EML:PHONEX
- $EML:DMSDDL
- $PGM:TELL
- $PGM:XTELL
- $TDO:SCHED
- $TDO:MEET
- $TCP:POPD
- $TCP:POPPASSD
- $MCS:MCSH
- $EML:QPUTI
- $EML:NETCNV

Fill in the fields on the Configuration screen to indicate all the Mail parameters for your site. Each field (Mail parameters) is described below.

ACCESS — The programs that use this parameter are: MAIL, MPROF, RDMAILER, GETMAIL, GETMINFO, SENDMAIL, MAILBOOK, and PHONEX. This parameter defines who can use the MAIL program. Enter a name for the file containing the authorization table. The default file name is $EMD:MAIL.AUTHOR and it allows access for all. For information about the authorization table layout see the topic "Authorization Table".

AFWDON — The AFWDON=f parameter can be used to not allow users to set the "Forward mail to" field in their Mail Profile. The "Forward mail to" field is used in the MAIL facility to automatically transfer mail sent to this userid to the given addresses in this field. This parameter can be used when the site wants all verification of sent mail to be done on a system different from the local MUSIC system. AFWDON=t allows the "Forward mail to" field to be set. AFWDON=t is the default. Note that SNOOP/VIP users can override this feature.

ALIAS
MUSNOD — The programs that use this parameter are: MAIL, GETMAIL, GETMINFO, MAILBOOK, MPROF, RDMAILER, SENDMAIL, PHONEX, SCHED, and MEET. Enter alternate names (up to 4) for your MUSIC system domain name. The name variable can be up to 132 characters in length. There is no default. This allows your MUSIC system to receive mail addressed to it, under the name specified in MUSNOD or one of those specified in ALIAS1-4. These names would have to be domain names registered with the Internet, BITNET, or known to a mailer that has been designated to resolve domain names for your MUSIC system. For example, if the domain name was MUSICX.DOMAIN.NETWORK, then the alias could be given as MUSICX or MUSICX.DOMAIN.NETWORK.

ALTSYS — The ALTSYS=f parameter can be used to not allow users to set an alternate systemid in their Mail Profile by using the F2:Alt System or ALTSYS command. ALTSYS=t allows an alternate system to be set. ALTSYS=t is the default. Note that SNOOP/VIP users can override this feature.

BITNET — The RDMAILER program uses the BITNET parameter to send email to any remote sender using a local BITNET email address. This parameter can aid a site in its migration from BITNET to Internet email addressing by advertising the change to these senders. BITNET=T sends email. BITNET=F is the default and it does not send email.

When BITNET=T, RDMAILER executes the file $EML:BITNET to send mail when it receives mail destined to a local BITNET email address. The body of the mail is the contents of the file $EML:@NOTICE. Either of these files can be tailored to meet the site's needs.

CLASS — The CLASS parameter for RDMAILER specifies the spool class for the virtual punches used for spooling mail text to other systems, such as VM. The default is class M.

CODTBL — The CODTBL=t parameter can be used to force the MAIL program to use the CODE TABLE NAME field as a profile name if a profile name is not present in the mailbox. The default is CODTBL=t. This parameter is used by the MAIL, MPROF, MAILBOOK, SENDMAIL, and RDMAILER programs.

CONSOL — When true, messages about POPD, POPPASSD, and RDMAILER activity will be sent to the MUSIC console. The default is T.

COPY — COPY=t allows users to copy files both as a function of MAILBOOK and with the VIEW store command. If COPY=f, the COPY select code does not appear in the list, and both the copy function in MAILBOOK and the VIEW store command are not allowed. The default is COPY=t.

DBCS — The DBCS=t parameter is used for double byte character set support within the MAIL program. The default is DBCS=f.

DEADX — The programs that use this parameter are: MAIL, GETMAIL, GETMINFO, and SENDMAIL. This parameter removes dead records from mailboxes. Enter a percentage between 0 and 100. If the percentage of dead records in the mailbox is higher than your value, then the mailbox is compressed to remove the dead records. DEADX=20 is the default.

EPRIME — The RDMAILER program uses this parameter to define the end of prime time. The default is EPRIME='7 pm'. The time can be 1 to 12 characters long. RDMAILER will not release large mail files during prime time. The SPRIME parameter specifies the start of prime time and SIZE specifies the length for large files.

ERRFIL — The programs that use this parameter are: GETMAIL, GETMINFO, MAIL, MAILBOOK, and SENDMAIL. Enter a name of a file for storing the error messages for each program. The default file name is $EML:MAIL.MSGS.

EXPIRE — The RDMAILER program uses this parameter to set the expiry date for inbound mail. Enter the number of days to be added to the date of the mail received. The MAIL and SENDMAIL programs use this parameter to set the default expiry date for mail sent. Enter the number of days to be added to the current date. The MAIL, GETMAIL, and GETMINFO programs also use EXPIRE to expire mail in the mailbox older than the current date plus the expire value. Enter the age value for which you want mail kept in the mailbox. The default is 365 (days) or one year. It can take on values of 0 (no limit set) to 36500 days.

When EXPIRE is set to 0, a default expire date of one year after the mail release date is assumed, however the user can always change this value. A user or a group of users can be assigned a value to override this default. See the documentation on the mail authorization file for further information to set up overrides.

EXPIRE can also take on values between -1 and -1216 which represent the number of 30 day months. Thus if EXPIRE=-3, EXPIRE would take on the value of 90 days.

GATRSC The GATRSC parameter is used to define a domain name that can be added to any email address ending in .bitnet or with only a one part domain name that is 1 to 8 chars long. This can be used by a site to have mail sent to email addresses on BITNET automatically converted to an Internet email address. Then it is delivered to that Internet host which can deliver to the BITNET node. For example, assume the site had only Internet connectivity and its domain name was "somewhere.edu" and it wanted to allow its users to have easy access to BITNET email addresses. If the site defined GATRSC as "INTERBIT.CREN.NET", users could send mail to user@nodename. The email address would be transformed to user%nodename@INTERBIT.CREN.NET, and RDMAILER would deliver the mail to "INTERBIT.CREN.NET".

Some sites may not want to configure GATRSC as described above. With VM's SMTP, if a user sends mail to user@address, SMTP can send it to user@address.domain_name, where domain_name is defined within the SMTP profile on VM. If this method is used at the site, GATRSC should be left blank.

INSWCH The INSWCH parameter defines the number of incoming mail items that RDMAILER will process consecutively before giving outgoing mail processing a chance. The range is 5 to 1000. The default is INSWCH=25.

KILFIL The RDMAILER program uses the filename defined by this parameter as a kill file. The default filename is $EML:KILLFILE. The contents of the kill file are defined in the ADMIN facility, option 4 5 B "Update the RDMAILER kill file". The kill file contains email addresses of senders. When RDMAILER encounters an incoming mail item from a sender that matches something in the kill file, RDMAILER deletes the mail item without taking any other action. In other words, the mail item is killed and not delivered.

*Note:* Kill file processing can slow down incoming mail delivery by RDMAILER. It is best to keep the kill file to the minimum number of entries required.

KILSIZ This parameter defines the number of characters you want to set aside for kill file entries. The default is KILSIZ=16000 allowing for 200 entries of 80 characters each. Note that because KILSIZ is given as a character count, you must define KILSIZ for the maximum number of characters you expect in the kill file. For example, if you have 100 entries in the kill file, KILSIZ=8000 is required (i.e. 100*80=8000).

LOG The programs that use this parameter are: MAIL, POPD, POPPASSD, RDMAILER, and SENDMAIL. This parameter is used to keep a log of all mail transactions coming from and going to MUSIC. The information is written to the log file by the LOG server BTRM. Set to T if logging is to take place. The default is F. See the MAIL Logs section under "MAIL Administration" for further information.

MAILER This parameter is used with the RDMAILER program. It specifies a 1 to 8 character name for the VM mailer. VM's SMTP can be the VM mailer so use SMTP's virtual machine name here. There is no default value. Leave this field blank if a VM mailer or VM's SMTP is not being used. This field and the MNODE parameter must be specified if a VM mailer or VM's SMTP is being used as the mailer.

MAXRCD MAXRCD defines the number of Received: headers RDMAILER will allow. This protects against network and local mail loops, bouncing the rejected item to the local postmaster and the sender. If the number is negative, only the Received: headers above the body of the mail will be counted. If the number is positive, all Received: headers regardless of placement in the mail text (provided that it begins in column 1) will be counted. The allowed values are -5 to -99 and 5 to 99. 0 means do not check. The default is -30.

MAXRS MAXRS defines the number of re-sents RDMAILER will allow. This protects against network and local mail loops, bouncing the rejected item to the local postmaster and the sender. If the number is negative, only the resent statements above the body of the mail will be counted. If the number is positive, all resent statements, regardless of placement in the mail text (provided that it begins in column 1), will be counted. The allowed values are 5 to 99. The default is 15.

MBDEL MBDEL=t allows users to delete files as a function of MAILBOOK. If MBDEL=f, the DELETE select code does not appear in the list, and the DELETE function in MAILBOOK is not allowed. The default is MBDEL=t.

MNODE This parameter for RDMAILER specifies the RSCS node name of the processor running the VM mailer or VM's SMTP if it is being used as a VM mailer. This 1 to 8 character name may or may not be the same as VMNOD. Leave this field blank if a VM mailer or VM's SMTP is not being used. This field and the MAILER parameter must be specified if a VM mailer or VM's SMTP is being used as the mailer.

MYNODE This parameter is used by SENDFILE, TELL, XTELL, and NETCNV to set the dummy RSCS node name of your MUSIC system. The default is MUSIC.

NUMRDM NUMRDM is used to represent the number (0 to 10) of auxiliary RDMAILER BTRMs configured for processing incoming mail. RDMAILER must know this in order to determine which entries it should handle in the reader queue. The default is NUMRDM=0, which signifies one RDMAILER BTRM and it handles both incoming and outgoing mail. NUMRDM=1 signifies one RDMAILER for outgoing mail, and one RDMAILER for incoming mail. See the topic on multiple RDMAILER BTRMs in the MAIL Administration section for further information. The NUMRDM parameter is also used by the VMREADX btrms, POPD, MCSH, and QPUTI.

PERIOD The PERIOD parameter for MAIL.CLEANUP deletes mail text and distribution files. Enter the number of days (1 to 365) before the current date for which a mail file will be deleted if it was created on or before this new date. For example, if PERIOD=20 and today's date is 30NOV90, then all mail files created on or before 10NOV90 are deleted. The default is PERIOD=0 which means to delete the file only if the file's expiry date is before the current date.

POST POST=t is meant to be used as a "post only" mailbook (ie send operations are not allowed). Setting POST=t also forces MBDEL=f (i.e. mailbook delete). If POST=f, send operations are allowed on the MAILBOOK items. The default is POST=f.

PSTMST This parameter sets the postmaster userid for MAIL, MAILBOOK, RDMAILER, MPROF, and SENDMAIL. The postmaster userid is where notices of problems and bad or undeliverable mail is sent. The default is "$EMP". This userid must be different from PCODE.

Some sites may have PSTMST set to "$EMP000". This is okay and you do not need to change it to "$EMP". If you do change it from "$EMP000" to "$EMP", you must reconfigure the MAIL facility (ie run Mail Config option 3 to create the executor files) and you must rename the file "$MBF:B$EMP000" to "$MBC:B$EMP" before the MAIL facility is used by any of your users or the RDMAILER BTRM is restarted. This is the postmaster's

mailbox that is used by the MAIL facility.

QFILE This parameter for RDMAILER and QPUTI specifies the name of the reader queue for accessing incoming mail. The default is "$VMR:RQ".

RCLASS Sets the incoming mail class(es) that RDMAILER is to process. This corresponds to the VM spool class of the incoming mail. The default for RCLASS is 'M','A'.

RDREG This parameter defines the REGion size for the RDMAILER BTRMs. The default is RDREG=1024. This parameter allows for sites to define a larger region to accomodate more kill file entries and/or mail items in core.

RELAY The RDMAILER program uses the RELAY parameter to allow or disallow mail to be relayed through your site. RELAY=f disallows mail relaying. The default is RELAY=t. Mail relaying is a mailing method where someone outside your site uses your site to send mail to someone else not at your site. This method is used in mail spamming.

RMTALL The programs that use this parameter are: MAIL, MPROF, RDMAILER, and SENDMAIL. The RMTALL=t parameter can be used to force the sending of all outgoing mail via RDMAILER to another system or remote mailer that will do the send. This can be used when the site wants to verify all sender/recipient combinations on a system different from the local MUSIC system. The normal send method is for MAIL and RDMAILER to verify sender/recipient combinations. In the case where the remote system is not found in the MAILER PROFILE, RDMAILER sends the mail to MAILER at MNODE. See MAILER and MNODE namelist parameters for a description of these parameters. If RMTALL=t, all mail sent by the MAIL program will be put into the post office mailbox for delivery by the RDMAILER program. RDMAILER will send this mail to MAILER at MNODE. If RMTALL=f, all mail sent by the MAIL program will be sent the normal way. RMTALL in MPROF controls whether the user is allowed to use the ALTSYS support to change his/her default systemid. RMTALL=t disallows ALTSYS support, and RMTALL=f allows ALTSYS support. RMTALL=f is the default in all programs.

RMTERR This parameter is used by RDMAILER. When RMTERR=t, the default, bad mail is returned to the sender. When problems occur during the delivery of incoming remote mail, postdated mail, recurring mail, autoforwarding mail, or outgoing remote mail, the sender would be notified. These include bad address, invalid recipient, and other syntax errors.

RSCS This parameter for RDMAILER and SENDFILE specifies your local RSCS virtual machine. This node name of RSCS running on VM can be 1 to 8 characters. The default is "RSCS". If RSCS is not used, specify blanks.

*Note:* You should list here the names of all RSCS machines you have running on your VM system. This includes any dummy node names related to your implementation of the "RSCS mods" that McGill provides.

SCLASS This value is used internally for RDMAILER to indicate outgoing mail. It has nothing to do with the VM spool class used for outgoing mail (see CLASS). The default is "O". There is no reason to change this value.

SIZE This parameter for RDMAILER specifies the number of lines the mail text file can be in order to be sent during prime time. Prime time is defined by the SPRIME and EPRIME parameters. Mail text that exceeds this will be held until non-prime time is reached. The default is 13108 lines or 1 megabyte of data. SIZE of less than 100 lines will be rejected.

SLEEP Specify a positive integer for setting the sleep or delay value. RDMAILER will remain idle during the specified number of seconds before looking for new work. The default is 600

seconds (10 minutes). Since the programs that feed RDMAILER automatically wake it up, this value need not be set very low.

SMTP — Eight character node name of SMTP running on VM. If you do not have SMTP running under VM, leave this blank. This parameter is used by the RDMAILER program.

SNDTYP — This parameter for RDMAILER sends the MUSIC mail to the default mailer defined by the MAILER parameter. One of the following protocols can be entered:

BSMTP (default)
BITNET
DEFRT
LOCAL

SNOOP — Where 'privilege' is one of the following 'FILES', 'CODES', 'MAINT', 'LSCAN', 'DREAD', 'CREAD', 'VIP', 'INFO', 'SYSMNT', 'SUPV', or 'SYSCOM'. If a privilege is given, all users with this privilege can bypass the surrogate support test and look at anyone's mailbox. They just type the userid in the FOR() field. The default privilege to bypass the surrogate support is 'VIP'. If this value is changed, remember that the user must have at least FILES privilege since the GETMAIL program must write to the mailbox. The Mail Profile facility supports this feature with the FOR name command. This parameter is used by the MAIL, GETMAIL, GETMINFO, MPROF, and SENDMAIL programs.

SPRIME — The RDMAILER program uses this parameter to define the start of prime time. The default is SPRIME='7 am'. *time* can be 1 to 12 characters long. RDMAILER will not release large mail files during prime time. The EPRIME parameter specifies the end of prime time and SIZE specifies the length for large files.

UNITS — This parameter for RDMAILER specifies the device addresses of virtual punches used for spooling mail text to other systems. These devices are normally defined in the VM directory for MUSIC. Up to 10 virtual punch addresses (specified in HEX) can be given. Care should be taken in entering the "Z" to indicate a hexadecimal address. The default is 11 and 12.

VMNOD — This parameter of RDMAILER and SENDFILE specifies the node name of your processor in the VM RSCS network.

ZONE — The programs that use this parameter are: MAIL, MAILBOOK, DMSDDL, RDMAILER, and SENDMAIL. It defines the local time zone. *zone1* is the time zone to use from the start of the year until *date1* or from *date2* until the end of the year. *zone2* is the time zone to use from *date1* until *date2*. Valid values for *zone1* or *zone2* are EST, EDT, CST, CDT, MST, MDT, PST, PDT, UT, or -12 to +12 to give GMT +-12. The dates are in format DDMMM. The defaults are 'EST','EDT','01MAY','01OCT'

Typically *zone1* is the standard time zone, while *zone2* is the day light savings time zone. Some sites may choose to set *zone1* and *zone2* to the same value. *date1* is the day *zone2* is to come into effect. *date2* is the day *zone1* is to come into effect.

## 4. Special Namelist Parameters

Listed below are a number of namelist parameters that should no longer be used or that are available for the appropriate programs but are not available within the MAIL CONFIG facility.

FCODE — The FCODE parameter sets the file userids for the following programs: MAIL, MAIL.CLEANUP, RDMAILER, and SENDMAIL. The file userid(s) is the userid on which all mail distribution list and text files are stored. Only the first three characters of FCODE are used for the userid. The fourth and fifth characters used for the five character userid are

automatically generated. The default is "$md?", where files are generated on userids $mdAA to $md99, 1296 combinations. This allows mail files to utilize upto 1296 save library index segments or a lesser value, whichever is available. See the MAIL Administration section later for a discussion of this parameter and the Save Library index.

The FCODE parameter should not be changed from the default. The MUSIC system has been coded to skip UCR "accounting" on any userid starting with "$md", thus reducing the number of I/O operations on the system. Changing the FCODE parameter will increase the I/O load on system.

HOLDIN The HOLDIN parameter can be used to create an RDMAILER BTRM which will process only outgoing mail (ie HOLDIN=t). The default is HOLDIN=f.

HOLDOT The HOLDOT parameter can be used to create an RDMAILER BTRM which will process only incoming mail (ie HOLDOT=t). The default is HOLDOT=f.

MYNUM MYNUM is used to indicate which number within NUMRDM does this RDMAILER represent. For example, NUMRDM=5 and MYNUM=2 indicates that there are 5 RDMAILERs configured for processing incoming mail, and this one is number 2. MYNUM is used to determine which entries to handle in the reader queue. See the topic on multiple RDMAILER BTRMs in the MAIL Administration section for further information.

PCODE This parameter is used by most of the mail facility programs: GETMAIL, GETMINFO, MAIL, MAILBOOK, MPROF, RDMAILER, and SENDMAIL. It defines the outgoing mailbox for RDMAILER. The default is $EMD.

Some sites may have PCODE set to "$EMD000". This is okay and you do not need to change it to "$EMD". If you do change it from "$EMD000" to "$EMD", you must reconfigure the MAIL facility (ie run Mail Config option 3 to create the executor files) and you must rename the file "$MBD:B$EMD000" to "$MBE:B$EMD" before the MAIL facility is used by any of your users or the RDMAILER BTRM is restarted. This is the post office mailbox that is used by the MAIL facility.

## 5. Modify the $EML:MAILER.PROFILE file

You must modify the file $EML:MAILER.PROFILE to suit your particular needs. This file is used by RDMAILER to determine where to send outbound mail. If you intend to use a VM mailer or SMTP to handle all of the outbound mail, this file is NOT required. This file is required when you use RDMAILER as your MAILER. The format used in the Columbia VM Mailer's "MAILER PROFILE" file is used. RDMAILER uses the OUTGOING and DOMAIN sections. If for some reason you require the MUSIC mail system to have knowledge of the entire BITNET topology, the easiest way to create this file is to copy the mailer profile file from VM. Otherwise, you will have to get the files XMAILER NAMES and DOMAIN NAMES from LISTSERV@BITNIC and run $EML:MAILER.PROFILE.MK to make this file. See the description of the $EML:MAILER.PROFILE.MK program later in this section. Normally, however, the file only contains local additions and exceptions.

If you have Internet connectivity but not BITNET connectivity (ie you are running SMTP on VM and not a VM mailer), you do not need to modify the $EML:MAILER.PROFILE file. This way all outgoing mail will be sent to your SMTP virtual machine when it is defined in the MAIL.CONFIG MAILER parameter.

When a user sends mail externally the address is specified as:

```
userid@system
```

RDMAILER uses $EML:MAILER.PROFILE to verify that the "system" exists, to find out how it should get

it there, and to determine what format the mail should be in. If RDMAILER cannot find an entry for the "system" specified, it will pass the mail off to SMTP or a VM mailer if one has been defined. Otherwise the mail is sent back to the sender with an error indicated.

The program $EML:MAILER.PROFILE.MK can be used to make the file $EML:MAILER.PROFILE. See the description of this program under "Mail Utility Programs" for more details.

A sample $EML:MAILER.PROFILE is given below.

```
OUTGOING:
; Alias name     Node     Userid   Exit     Type Parm
; ----------     -------- -------- -------- ---- --------
MCGILL1          MCGILL1  ?        DEFRT    1
MCGILLM          MCGILL3  MUSIC    BSMTP    3
MCGILL2          MCGILL1  MAILER   BSMTP    3
END OUTGOING
; Table of mail domains that I can send to.

DOMAINS:
; Domain name    Node     Userid   Exit     Type Parm
; ----------     -------- -------- -------- ---- --------
MCGILL.CA        MCGILL1  MAILER   BSMTP    3
END DOMAINS
```

*Figure 9.3 - Sample Mailer Profile*

*Note:* ADMIN 4 5 4 can be used to edit the file $EML:MAILER.PROFILE. Make your changes using the MUSIC/SP Editor, save your changes, and restart the RDMAILER BTRMs to have the edited file put into use.

The sample shows an OUTGOING list and a DOMAINS list. These lists are headed by the "OUTGOING:" or "DOMAIN:" identifiers and terminated by an "END" statement. Lines that start with ";" are comments. The other lines describe individual nodes in the network. The first column is the node or domain name that is specified by the user sending the mail. The second column is the RSCS node name of the processor where the mail should be sent. This can be different from the first column. The third column is the VM USERID of the virtual machine that is to receive the mail. Normally this would be the VM mailer service machine. If a question mark is specified here, the USERID specified by the sender is substituted.

The fourth column defines the type of mail the recipient is capable of handling.

DEFRT — No mailer is being used. The mail is sent directly through RSCS to the user at the remote site.

BITNET — The receiving mailer can handle RFC822 format mail. The mail is sent to the mailer, who distributes it locally based on the RFC822 headers.

BSMTP — The receiving mailer can handle BSMTP format mail. The mail is sent to the mailer, who distributes it locally based on the BSMTP addresses. BITNET members are required to support BSMTP. This should be the default for non-local entries in this file.

Subsequent columns are not used by RDMAILER but are present in the mailer profile used by the Columbia VM Mailer.

Using the sample in Figure 9.3, if mail was sent to JOHN@MCGILL1, RDMAILER would send the file to userid JOHN on the processor identified to the RSCS network as MCGILL1.

Mail for JOHN@MCGILLM would be sent to the virtual machine MUSIC on the processor MCGILL3. Assuming this is a MUSIC machine, RDMAILER would distribute the mail to the user JOHN.

Anything sent to ...@MCGILL2 would be sent to the the userid MAILER at MCGILL1, regardless of the userid specified. The mailer would handle delivery to the specific user(s).

The domain table is interesting in that an exact match on system names is not required. For example, mail sent to JOHN@VM1.MCGILL.CA would be sent to MAILER at MCGILL1. It would be up to this mailer to figure out the VM1 part.

### 6. Create a Site Mail Profile

Modify the site Mail Profile if your site has a requirement to create a standard mail profile for all new mailboxes. If a site Mail Profile is not created, new users do not have any Mail Profile options preset and they must set the options themselves.

ADMIN 4 5 3 "Update site Mail Profile" can be used to create/modify a site Mail Profile that is copied to all new mailboxes at creation time. The Mail Profile facility is used to create the site Mail Profile. The site Mail Profile is stored in $EMM's mailbox. Only the Mail Profile from $EMM's mailbox is copied to new mailboxes at creation time. The Name and Email Id fields of the site Mail Profile are not copied to the new mailboxes.

This facility can be used to set any Mail Profile options that should be copied to new mailboxes. For example, the site wants mail sent to be copied to a file automatically. On the "Create and Send Mail Options" screen, simply set the "Copy the mail to be sent to a file" to Y, set a filename where to save the mail to be sent (e.g. MAIL.LOG), and set Append to this file to 'Y'. Filenames specified should probably not be userid prefixed. That way when the Mail Profile in the new mailbox is actually used, the filenames all default to the userid of the new mailbox.

## Sample Configurations

### Single processor network

In this case mail cannot be sent outside of the processor, but it is possible for mail to be exchanged by MUSIC systems and CMS users running on the processor. There are a number of considerations.

MAIL.CONFIG:

- Since there is no RSCS, SMTP, or VM mailers the fields RSCS, SMTP, MAILER, and MNODE should be left blank.

- Choose a node name for your MUSIC machine(s) and your VM processor by setting MUSNOD and VMNOD. These can be chosen somewhat arbitrarily but should have some meaning to your users. For example MUSIC and VM might be good choices.

$EML:MAILER.PROFILE:

- Define the VM node name in OUTGOING: section of the MAILER.PROFILE file as a DEFRT type node.

- Define any other MUSIC systems as BITNET nodes.

Example: This defines VM and a second MUSIC system called MUSICX.

```
OUTGOING:
; Alias name     Node     Userid   Exit     Type Parm
; ----------     -------- -------- -------- ---- --------
VM               VM       ?        DEFRT    1    TRUNCATE
MUSICX           VM       MUSICX   BSMTP    3
END OUTGOING
; Table of mail domains that I can send to.
DOMAINS:
; Domain name    Node     Userid   Exit     Type Parm
; ----------     -------- -------- -------- ---- --------
END DOMAINS
```

*Figure 9.4 - Defining VM and MUSICX*

On VM:

CMS users can send mail to MUSIC by spooling a punch file containing the text to MUSIC as a class A or M file. Suitable sender/receiver addresses should be included at the top of the file to identify the sender and the recipient of the mail.

```
To:   USR1@MUSIC
From: USR2@VM
```

The VM NOTE command is the normal method of sending electronic mail in CMS. If the MUSIC userid is used on the NOTE command, NOTE will attempt to send the mail to a CMS user with that userid. If the address of "userid AT MUSIC" is specified, NOTE will try to send the file to RSCS. Even if you are running RSCS, this will fail since RSCS can only send the files outside the system. One solution would be to modify NOTE to handle mail for MUSIC as special and spool it directly to MUSIC instead of RSCS. A simpler solution is to modify the VM file SYSTEM NETID so that the NODEID is VM and the NETID is MUSIC.

## Existing RSCS node

McGill provides a modification to RSCS to allow a MUSIC site to look like a node in an RSCS network. The application of this modification is described in the file $EML:RSCSV13.MODS for RSCS V1R3, $EML:RSCSV2.MODS for RSCS V2R2 or V2R3, and $EML:RSCSV3.MODS for RSCS V3R1. ($EML:RSCSV2.TELLMODS and $EML:RSCSV3.TELLMODS describe the required modifications for MAIL plus those required for intersystem TELL support.) The MAIL system can function with this mod in place, so you can switch from MEMO to MAIL without requiring everyone in the network to update their tables to reflect your change.

MAIL.CONFIG:

- Set the RSCS parameter(s) to be the name(s) of your RSCS machine(s).

- Set VMNOD to be the node of your processor in the RSCS network.

- Set MUSNOD to be the node assigned to your MUSIC machine. This is the same as the dummy node name specified in the modification to RSCS.

- If you want to use your VM mailer to handle outbound mail on MUSIC's behalf, set MAILER and MNODE to the appropriate values. If you have a VM mailer service machine it will be to your advantage to do this, since it eliminates the need to maintain the full MAILER.PROFILE tables on MUSIC.

See the following section if this is the case.

- If you want to use your SMTP virtual machine to handle outbound mail on MUSIC's behalf, set MAILER to SMTP and MNODE to the appropriate value.

$EML:MAILER.PROFILE:

- If you are not using a VM mailer or SMTP, the MAILER.PROFILE must contain a list of all the nodes and domains in the network that you wish to send to.

- An up-to-date copy of this information can be obtained from those administering the network.

  The program $EML:MAILER.PROFILE.MK can be used to make the file $EML:MAILER.PROFILE. See the description of this program under "Mail Utility Programs" for more details.

On VM:

External users can send mail to MUSIC using the same techniques that they use to send to any other node in the network. Local CMS users will have no trouble using the CMS MAIL program to send mail to MUSIC. Local CMS users can also send mail to MUSIC by spooling a punch file containing the text to MUSIC as a class A or M file. A suitable sender/recipient address should be included at the start of the file.

```
To:   USR1@MUSNOD
From: USR2@VMNOD
```

In the example above MUSNOD is the RSCS node reserved for your MUSIC system and has nothing to do with the logon ID of the MUSIC virtual machine.

The VM NOTE command has some problems when used by local CMS users. If the VM NOTE command is used and an address of "USER AT MUSNOD" is specified, NOTE will send the file to RSCS since it knows nothing about where MUSNOD is. RSCS will reject the file since MUSNOD is not on its list of external nodes. The NOTE command could be modified to handle the local MUSIC node as a special case and spool the file directly to MUSIC bypassing RSCS. An alternative is to create a special exec that: temporarily creates a SYSTEM NETID file on the user's minidisk pointing to MUSIC instead of RSCS; issues the NOTE command; then gets rid of the SYSTEM NETID file. In this way, NOTE will send the mail directly to the local MUSIC system instead of to RSCS.

## Using a VM Mailer

Since the software to maintain the network configuration resides on VM and since a variety of mail handling service machines run on VM, it makes sense that rather than duplicate these functions in the MUSIC system, the MUSIC mail system would take advantage of these services wherever possible.

MAIL.CONFIG:

- Set the RSCS parameter(s) to be the name(s) of your RSCS machine(s).

- Set VMNOD to be the network node name of your processor.

- Set MUSNOD to be the node assigned to your MUSIC machine.

- Set the MAILER and MNODE parameters to point to your local VM mailer. This need not be on the same physical processor as your MUSIC machine.

$EML:MAILER.PROFILE:

- This file should be empty except perhaps for some destinations that for some reason you do not want to go through the mailer.

On VM:

Your VM mailer should be able to handle BSMTP headers since MUSIC uses them to route the mail. Make the appropriate changes on VM to add the MUSIC node to the network topology tables. If you are using the Columbia VM Mailer this involves updating the MAILER PROFILE file. Add an entry for the MUSIC virtual machine to the "INCOMING:" list so the mailer will accept mail from MUSIC as valid. Also add an entry for your MUSIC node in the "OUTGOING:" or "DOMAINS:" list so the mailer will deliver mail to the MUSIC system. Other nodes in the network that want to communicate with your MUSIC system must make the same changes to their MAILER PROFILE files. Normally this distribution of MAILER PROFILE information is coordinated by a central site. (In the case of BITNET, it is node BITNIC.) See the description of the $EML:MAILER.PROFILE.MK program later for further details on obtaining this distribution information if you do not receive it automatically.

## Using SMTP

MUSIC MAIL takes advantage of the services SMTP offers and does not need to duplicate these services.

SMTP Configuration on VM:

- Set the MAILER configuration statement for the SMTP virtual machine for your MUSIC system to send all mail to MUSIC. The MUSIC/SP MAIL facility can handle NETDATA conversion. Use the following statement:

```
MAILER MUSIC NEW LOCAL RSCS NOUNKNOWN                    (TCP/IP 2.0)
MAILER MUSIC NETDATA SOURCEROUTES LOCAL RSCS NOUNKNOWN (TCP/IP 2.2)
```

MAIL.CONFIG:

- Set the RSCS parameter(s) to be the name(s) of your RSCS machine(s). No RSCS modifications are required if you are using only SMTP.

- Set VMNOD to be the network node name of your processor.

- Set MAILER to be either the VM mailer program if you are running one, or the SMTP virtual machine name if you are only running SMTP. This parameter is used by outgoing mail.

- Set MNODE to the respective node name of MAILER parameter as defined above. This need not be on the same physical processor as your MUSIC machine.

- Set the SMTP parameter to the SMTP virtual machine name that can deliver mail to MUSIC. This parameter is used for incoming mail.

- Set MUSNOD and ALIAS1 to 4 to be the fully qualified domain names, FQDN, representing your MUSIC system. You MUST also include in this list the node name of your VM system.

$EML:MAILER.PROFILE:

- If you are running SMTP and/or a VM mailer, this file should contain just comments and no entries. In this case, all mail is automatically sent to either the VM mailer or SMTP if there is no match with an entry in this file.

On VM:

External users can send mail to MUSIC using the same techniques that they use to send to any other Internet user.

Local CMS users can also send mail to MUSIC when they use the NOTE and SENDFILE execs distributed with VM TCP/IP program product as replacements for the CMS versions of these execs.

If you are using a VM mailer, it can be configured to communicate with the SMTP virtual machine also.

**Passing Commands to RDMAILER**

Commands can be passed to any RDMAILER BTRM via the ATTENTION program. While an RDMAILER BTRM is running, it can be told to perform a number of things as well as change the status of certain variables. If your site is running more than one RDMAILER BTRM, you must know the "ENQNAM" of the RDMAILER BTRM with which you want to communicate before you can send it a command. The "ENQNAM"s of RDMAILER BTRMs that handle incoming mail are RDMAILnn, where nn is the namelist parameter MYNUM. The "ENQNAM" of the RDMAILER BTRM that handles outgoing mail is RDMAILER. If you are running only one RDMAILER BTRM for incoming and outgoing mail, the "ENQNAM" is RDMAILER.

By typing "attention ENQNAM" from *Go, where ENQNAM has the value as described above, you can enter the following commands:

HOLD x — Hold the processing of the class specified. If *x* is not specified then all classes are held. *x* can be any of the values specified in RCLASS or SCLASS. HOLD IN/OUT is also supported to do all classes.

STOP — Terminates RDMAILER.

END — Terminates RDMAILER.

CANcel — Terminates RDMAILER.

Quit — Terminates RDMAILER.

RCn x — Sets the character specified by x to be RCLASS, the remotely originating mail class. n is a number from 1 to 5.

Go x — Restarts class x. If no class is specified then all RCLASSes and SCLASS are restarted. GO IN/OUT is also supported to do all classes. If the HOLDIN or HOLDOT namelist parameter is set to TRUE, the GO command is ignored.

SIze n — Sets the file size in lines allowed to be sent during prime time. n must be a positive integer greater than or equal to 100.

SLeep n — Set the integer n as the SLEEP value. n is a positive integer representing time in seconds.

Log ON/OFF — Set logging to on or off.

Next x — Process the next entry in class x. Processing of the other class will be carried on normally. If x is not specified then any RCLASS and SCLASS will be processed, whichever is encountered first in the queue.

*Note:* NEXT is only valid after a HOLD command is issued. STep x is the same as NEXT.

| | |
|---|---|
| SClass x | Set the class for sending locally originating mail to the character specified by x. |
| DOMains | Re-read the domains file and rebuild the list incore. |
| Show | Prints out the namelist parameters and RDMAILER options and the status of current settings and values. |
| KILFIL | Displays the kill file entries RDMAILER is currently using. This may not correspond to the entries in $EML:KILLFILE. |

## VMREADX Parameters

VMREADX handles the initial processing of inbound mail and replaces the VMREAD program. When converting from the old MEMO program to MAIL, VMREAD should be disabled by deleting the autoprogram that is set up to run VMREAD. By default this ran on BTRM address 011.

MAIL.CONFIG automatically allocates the codes and creates the files necessary to run VMREADX. In case there is reason to change the defaults, the program parameters are described below. They should be entered in namelist format as shown in the sample files that are given.

| | |
|---|---|
| READER | The device address of the virtual reader |
| DAYS | Number of days to keep old print files |
| DELAY | Number of seconds to delay before looking for work |
| TRACE | Print trace and debug messages. |
| MSGS | Unit number for messages (0 is console) |
| SYSCOD | Code for print queue ($PRT) |
| RDRCOD | Code for reader queue ($VMR) |
| MYNODE | Set to the dummy RSCS node name of your MUSIC system (MUSIC) |
| NUMRDM | Number (1 to 10) of RDMAILER BTRMS configured for processing incoming mail. (NUMRDM=1) |
| RWDLY1 | Number of seconds to wait between wakeups of RDMAILER after having finished processing all VM reader files and before going to sleep. (RWDLY1=30) |
| RWDLY2 | Number of seconds to wait between wakeups of RDMAILER after processing each punch file. (RWDLY2=60) |

The following autoprogram files show recommended setting for these parameters.

File: $MON:A014 (handle the class M reader)

```
/INC VMREADX
READER=Z18,DAYS=20,DELAY=60,TRACE=0,MSGS=0,
NUMRDM=2
```

File: $MON:A015 (handle the class A reader)

```
/INC VMREADX
READER=Z19,DAYS=20,DELAY=60,TRACE=0,MSGS=0,
NUMRDM=2
```

*Note:* VMREADX replaces all the functions of VMREAD so you should delete the VMREAD BTRM or at least disable it's autoprogram.

See the topic "Running Multiple RDMAILER BTRMs" in the Mail Administration section for further information on this topic.

## SMTP Notes

The MUSIC file $TCP:TCPIP.SMTP.DOC provides the necessary information to help you configure SMTP to work with the MUSIC MAIL facility.

If you are connected to both BITNET and the Internet and don't mind or prefer your mail arriving via the Internet link, you should send in an update to your node definition to BITNIC informing them of your domain name and that you no longer require nameserver support for your node. See the file DOMAIN GUIDE at LISTSERV@BITNIC for further details.

If this is your situation, and you do not have a VM MAILER, and you are using $EML:MAILER.PROFILE with the BITNET XMAILER NAMES and DOMAIN NAMES files that are sent every month, then you should change those nodes in the DOMAIN NAMES section of the mailer profile that have been defined to deliver their mail to SMTP at INTERBIT to your SMTP machine. Note that you should only change those nodes which have :interconnect.YES or :interconnect.MX. You can find the interconnect tag for the nodes in the raw DOMAIN NAMES file you are sent each month.

## Special File Names

The following special files are used by MAIL. The default PCODE value of $EMD is used where appropriate.

| | |
|---|---|
| $EML:DIRPUBL.MSGS | contains the DIRPUBL messages. The file has a VC record format and a record length of 1536. It is not a text-only file, and it should not be edited. This file is produced from $EML:DIRPUBL.MSGS.S by the $EML:MSG.BLDR program. |
| $EML:DIRPUBL.MSGS.S | contains the raw text for the DIRPUBL messages. This file has a VC record format and a record length of 512. It is a text-only file, and it can be edited. This file is used by the $EML:MSG.BLDR program to produce the $EML:DIRPUBL.MSGS files. This file is normally offline and may have to be restored from the source tapes if use is required. |
| $EML:MAIL.MSGS[@] | contains the MAIL messages. The file has a VC record format and a record length of 1536. It is not a text-only file, and it should not be edited. This file has National Language variants E (Spanish), F (French), P (Portugese), K (Kanji-Japanese) supported through the [@] placeholder in the file name. These files are produced from $EML:MAIL.MSGS[@].S by the $EML:MSG.BLDR program. |
| $EML:MAIL.MSGS[@].S | contains the raw text for the MAIL messages. This file has a VC record format and a record length of 512. It is a text-only file, and it can be edited. This file has National Language variants E (Spanish), F (French), P (Portugese), K (Kanji-Japanese) supported through the [@] placeholder in the file name. These files are used by the $EML:MSG.BLDR program to produce the $EML:MAIL.MSGS[@] files. These files are normally offline and may have to be restored from the source tapes if use is required. |
| $EML:PROFILE.MSGS | contains the MPROF messages. The file has a VC record format and a record length of 1536. It is not a text-only file, and it should not be edited. This file is produced from $EML:PROFILE.MSGS.S by the |

| | |
|---|---|
| | $EML:MSG.BLDR program. |
| $EML:PROFILE.MSGS.S | contains the raw text for the MPROF messages. This file has a VC record format and a record length of 512. It is a text-only file, and it can be edited. This file is used by the $EML:MSG.BLDR program to produce the $EML:PROFILE.MSGS file. This file is normally offline and may have to be restored from the source tapes if use is required. |
| $EML:VIEW.MAIL.COMI | contains a list of the commands available to VIEW for incoming mail. |
| $EML:VIEW.MAIL.COMO | contains a list of the commands available to VIEW for outgoing/acks/unreceived mail. |
| $EML:MAILER.PROFILE | contains the MPROF messages. The file has a VC record format and a record length of 1536. It is not a text-only file, and it should not be edited. This file is produced from $EML:PROFILE.MSGS.S by the $EML:MSG.BLDR program. |
| $EML:MAILER.TEMPLATE | serves as the template for $EML:MAILER.PROFILE. This is used by the program $EML:MAILER.PROFILE.MK. |
| $EMD:MAIL.AUTHOR | contains the authorization table for send operations and user/type specific expiry dates and mail item view limits. (The PCODE parameter can be used to specify an alternate code other than $EMD.) See the section "Authorization Table" below for information. |
| $MB?:Buserid | contains the user mailbox. $MBA to $MBG are used to store mailboxes. Use the MAILBOX program to determine the filename of a user's mailbox. See the description of the MAILBOX program under the section "Mail Utility Programs" in this chapter. |
| $MB?:Ouserid | contains the user overflow mailbox. $MBA to $MBG are used to store mailboxes. The overflow mailboxes contains the mail records that could not be added to the mailbox at delivery time. Invoking MAIL, GETMAIL, GETMINFO, copies the overflow mailbox to the real mailbox. |
| $MBG:B$EMM | contains the site Mail Profile. System Administrators can use ADMIN 4 5 3 "Update site Mail Profile" to update the Mail Profile for this special mailbox. The Mail Profile from this mailbox is copied to all new mailboxes when they are created. |
| $EMD:@MAILLOG.yyyymmdd | contains a log of all mail sent and received, and the RDMAILER progress and error messages if LOG=t. *yyyymmdd* is the date the log was started. This is the default mail log filename. This log file is not created if LOG=f. The log file is created by the MUSIC LOG server BTRM. See the MAIL Logs section under MAIL Administration for further information. |
| $MD??:cccc | are files that contain mail text and distribution files. *cccc??* is a 6 character string based on the date and time of day. See the MAIL Administration section later for further information on these filenames and the Save Library index. |
| @NAMES | is the name of the file of nickname information created and maintained by the DIRECT program. |

| | |
|---|---|
| $EML:@NAMES | is the name of the file of public nickname information created and maintained by the DIRECT.PUBLIC program. |
| $EML:MEDITOR | MAIL editor |
| $EML:MEDITOR.SHOWTEXT | MAIL editor showtext. Users can override the MAIL editor showtext by creating their own MEDITOR.SHOWTEXT file. |
| MEDITOR.USERCMDS | user MAIL editor commands file. These commands are the last editor commands issued. As an example this file can include commands to redefine F1/F13, F5/F17, or the show text. |
| $EML:MMSG.MAC | MAIL editor macro to put out a message when the text of the mail item cannot be merged into the text to be resent/forwarded. |
| $EML:MBKEDITOR | MAILBOOK editor |
| $EML:MBKMSG.MAC | MAILBOOK editor macro to put out a message when the text of the mailbook item cannot be merged into the text to be resent/forwarded. |
| $EML:MEDHELP.MAC | MAIL/MAILBOOK editor macro to invoke the MAIL/ MAILBOOK editor help |
| $EML:SAVE.MAC | MAIL/MAILBOOK editor SAVE macro |
| $EML:SEND.MAC | MAIL/MAILBOOK editor SEND macro |
| $EML:SIG.MAC | MAIL/MAILBOOK editor SIGnature macro |
| $EML:SUSPEND.MAC | MAIL editor SUSPEND macro |
| $EML:FMRGEDX | Editor for merging in the original mail headers in with the forwarded mail text (MAIL) |
| $EML:FMBKEDX | Editor for merging in the original mail headers in with the forwarded mail text (MAILBOOK) |
| $EML:FMEDX.MAC | Editor macro for putting the original mail headers onto the forward mail text (MAIL) |
| $EML:FMBKEX.MAC | Editor macro for putting the original mail headers onto the forward mail text (MAILBOOK) |
| $EML:MKNICK | REXX exec used by the MAIL MAKENICK function |
| $EML:PCEDIT | REXX exec to create a file using an PC editor and transferring the file to MUSIC |
| $EML:MAGFIL.SUPPORT | describes the "File to be executed when mail arrives" field on the MPROF General Options screen. |
| $EML:NAMDIR.MAC | Editor macro used by the DIRECT and DIRECT.PUBLIC programs. |
| $EML:NORECEIVE | contains a list of valid userids that are not allowed to receive mail. The mail |

| | |
|---|---|
| | is returned to the sender as if the userid did not exist. This file is used by RDMAILER. The userids are listed one per line. Wild card characters * and ? are allowed in the userids. RDMAILER must be restarted when this file is changed. This file can be modified by using ADMIN 4 5 6 "Update noreceive file". |
| $EML:MAIL.MAKE | Exec to remake the MAIL facility programs. This can be run after you have applied source changes and have recompiled/reassembled the source. This program can be run from ADMIN 4 5 8 "Remake Mail facility after local mods". |
| $EML:MAIL.CHGVNM | Program required to change all of the mailboxes to the new format required for MUSIC/SP 2.4 from the earlier version of the mailboxes. |
| $EML:MAIL.FILE.COUNT | output of the MSTAT program appended here |
| $EML:MBK.VIEW.CMDS | contains a list of the commands available to VIEW for MAILBOOK |
| $EML:MAIL.IND.DESC | describes the IND common block used in the MAIL facility |
| $EML:MAIL.LRMAIL.DESC | describes the LRMAIL common block used in the MAIL facility |
| $EML:MAIL.LRPROF.DESC | describes the LRPROF common block used in the MAIL facility |
| $EML:MAIL.LRSEND.DESC | describes the LRSEND common block used in the MAIL facility |
| $EML:MAIL.MAILNM.DESC | describes the MAILNM common block used in the MAIL facility |
| $EML:MAIL.MDFILE.DESC | describes the MAIL distribution file as used by the MAIL facility |
| $EML:MAIL.MLIST.DESC | describes the incore copy of the MAIL items as used by the MAIL facility |
| $EML:MAIL.NICKS.DESC | describes the nicknames file records as used by the private and public directory programs |
| $EML:MAIL.SUBS2.DESC | describes the low level mailbox routines (SUBS2) |
| $EML:MAILBOX.DESC | briefly describes the mailbox, and some of the underlying structure of the files used in the MAIL facility |
| $EML:MAIL.SEND.RC | describes return codes when sending mail |
| $EML:MAIL.SUBS2.RC | describes the return codes of the low level mailbox routines (SUBS2) |
| $EML:MPROF.MENU[@] | contains the MPROF menu. This file has National Language variants E (Spanish), F (French), P (Portugese), K (Kanji) supported through the [@] placeholder in the file name. |
| $EML:RDMAILER.SMTP.DOC | contains a description of how to get RDMAILER and SMTP to work together for a BITNET site. You should also refer to the file $TCP:TCPIP.SMTP.DOC. |
| $TCP:TCPIP.SMTP.DOC | describes how to configure the MUSIC MAIL facility with SMTP. |

| | |
|---|---|
| $EML:RSCSV13.MODS | contains suggested modifications for RSCS V1.3 for intersystem email communications |
| $EML:RSCSV2.MODS | contains suggested modifications for RSCS V2 for intersystem email communications |
| $EML:RSCSV2.TELLMODS | contains a full description of the suggested modifications for RSCS V2 for intersystem email and tell communications |
| $EML:RSCSV3.MODS | contains suggested modifications for RSCS V3 for intersystem email communications |
| $EML:RSCSV3.TELLMODS | contains a full description of the suggested modifications for RSCS V3 for intersystem email and tell communications |
| XMAILER_NAMES_filename | file obtained from BITNIC and used as input into the $EML:MAILER.PROFILE.MK program to make RDMAILER's $EML:MAILER.PROFILE. |
| DOMAIN_NAMES_filename | file obtained from BITNIC and used as input into the $EML:MAILER.PROFILE.MK program to make RDMAILER's $EML:MAILER.PROFILE. |
| $EML:INTEREST.GROUPS | contains a copy of the Internet list of lists used by the List Manager facility. |
| $EML:INTEREST.WIDX | contains the index required for the search engine in the List Manager facility. |
| $EML:KILLFILE | contains the kill file entries RDMAILER uses to match against senders of incoming mail items. |

**Other Temporary files used by MAIL**

| | |
|---|---|
| @MAIL.NEW.tcbn,<br>@MAIL.REPLY.tcbn,<br>@MAIL.FORWD.tcbn | are files used by the editor where the user types in the mail text. These should normally be deleted at send termination. It is purged before the editor is called if it exists. |
| &&TEMP | contains a log of results of the send mail operation. If any errors occur, this file is normally viewed to show the user the errors that occurred. This file is created as needed as a temporary file. |
| @MAIL.SUPSND.tcbn | used by the MAIL where the user has suspended the sending of a mail item. After the mail item has been stored in the user's mailbox, the file is deleted. |
| @DPUBL.KEYS | contains the user's function key definitions for the DIRPUBL program. |
| @MAIL.KEYS | contains the user's function key definitions for the MAIL program. |
| @NAMES.LISTsss | generated by the $EML:DIRECT.PRINT program to hold a readable copy of the directory. The user would print this file on the printer of their choice. If the userid has a subcode "sss", then it is suffixed to the file name. Otherwise, the file name is @NAMES.LIST. |

| | |
|---|---|
| @PROFILE.KEYS | contains the user's function key definitions for the MPROF program. |
| @PROFLOG.tcbn | contains the log file for the MPROF program. |
| @PHMSG.tcbn | contains the PHONEX message text created to send to the recipient. |
| @SMEXEC.tcbn | is the SENDMAIL exec created by PHONEX to send the message to the recipient. |
| @SENDFILE | contains a copy of the file sent using the SENDFILE program. It is purged at the end of execution. |
| @Gtcbn | is used by GETMAIL and GETMINFO when information has to be down-loaded to a PC file. |
| @MBQ.tcbn | contains a skeleton of the MAILBOOK item that is created to pass to the user's defined MAIL editor. |
| @@@MBK.MAC | is a file created by MAILBOOK to do the edit required for the automatic addition of the item's text to the new text for resend and forward functions. |
| @MBK.tcbn | contains a copy of the PC file that is used as the mailbook file for the MAILBOOK program . |
| @PCX.tcbn | file created by MAILBOOK when you send a PC file as the mail text. |
| @MAIL.PCX.tcbn | file created by MAIL or SENDMAIL when you send a PC file as the mail text. |
| @CURMBKtcbn | contains a copy of the current MAILBOOK item when you are performing an operation on an item. |
| @CURMAILsss | contains a copy of the current MAIL item when you are answering, replying to, forwarding, transferring, or resuming an item, or when the digest feature is invoked. If the userid has a subcode "sss", then it is suffixed to the file name. Otherwise, the file name is @CURMAIL. |
| @MCtcbn | file created by MAIL, SENDMAIL, or MAILBOOK when copying only the item's text to a PC file or when copying to a mailbook which is a PC file. |
| @MNtcbn | file created by MAIL, SENDMAIL, or MAILBOOK when the recipients are listed in a PC file. |
| @MDtcbn | is used by MAIL to build a display of the mail item's distribution list. |
| @MSUBLsss | contains the users subscription information from the LIST MANAGER facility, MAIL main menu option 8. If the userid has a subcode "sss", then it is suffixed to the file name. Otherwise, the file name is @MSUBL. |

## Authorization Table

Normally, the MAIL Authorization table is updated using ADMIN 4 5 2. See the *MUSIC/SP Administrator's Guide* for further details regarding this update procedure. The data records in the file

$EMD:MAIL.AUTHOR define two functions. The first type of data records define who can send mail and to whom they can send. The second type of data records defines an expiry date and a limit to the number of mail items presented for viewing for specific users, groups of users, or type of users.

**Type 1 Data Records (Send Controls)**

The format is free, but each line must contain three strings:

```
sender id     1-16 characters
receiver id   1-132 characters
system id     1-132 characters
domain        1-132 characters (optional)
```

An entry that starts with "*" in column 1 is a comment and is not read into the table.

The receiver id is the receiving user's userid, either given as a userid or as a full email address (user@system). By giving the receiver id as a full email address (user@system), the special names for system id as given below (*ANY and *NONE) can be used to allow or limit full email address access. Wild characters ? and * are supported within the full email address specification.

For the system id, there are several special names that can be used.

*NEVER The specified sender is not allowed to access the MAIL facility at all.

*NONE The specified sender-receiver combination is not allowed at all.

*LOCAL The specified sender-receiver combination is allowed only on this copy of MUSIC.

*OTHER The specified sender-receiver combination is allowed to send mail to systems other than MUSIC systems. They are not allowed to send to any MUSIC system.

*ANY The specified sender-receiver combination is allowed to send mail to any system, whether MUSIC or not. This option means no checking is done on the system id.

The domain name, if given, must be one of the domain names specified in MUSNOD or ALIAS1-4. This domain name will be used to generate the userid/system address for the sender ID. In this way, a user or class of users can appear to be at a different domain. A "*" here means that the domain name to be used is MUSNOD. Not specifying one is treated as having entered a "*". If the domain name is not one of the above, it is defaulted to MUSNOD.

The character "?" can be used as a "wild" character in both the sender and receiver ids. A "?" in the sender or receiver id in the table is considered to match any character in the corresponding position of an actual id.

The use of an "=" means that the relative position in both the sender - receiver id must match.

The '*' can be used to indicate any characters. For example cc* would match any code starting with cc. Analagous to this is cc?????, where the last 5 chars of the code can be anything. Special care should be used in mixing '*' and '=', since they can undo each other and expanding '*' would mess up the absolute position of the '='. So that a pattern of "==*" is acceptable, the pattern "*==" is not.

Wild chars '*' and '?' are allowed on the system name as well.

**Examples:**

```
*---SEND ID--RECEIVE----------SYSTEMID---------DOMAIN NAME
```

```
ABCD*          *           *NEVER
ABCD           =           *NONE
A123           *           *NONE
ABCD*          LISTERV*    *NONE
 *              *           *ANY
```

The first line is a comment line just as appears in the file $EMD:MAIL.AUTHOR.

The second line does not allow any userid starting with ABCD to access the MAIL facility.

The third line allows the userid ABCD to access the MAIL facility, but ABCD cannot send mail to him/herself.

The fourth line allows the userid A123 to use the MAIL facility, but A123 cannot send mail to anyone.

The fifth line allows any userid starting with ABCD to use the MAIL facility, but they cannot send mail to any email address starting with LISTSERV.

The sixth line allows anyone access to the MAIL facility and they can send mail to any email address.

**Type 2 Data Records (User Specific Controls)**

The format is free, but each line must start with a closing parenthesis, ")", and can contain 2 or 3 other parameters:

```
userid     1-16 chars
  or
T=type     integer value

days       integer value
items      integer value (optional)
POP Freq   integer value (optional)
```

An entry that starts with "*" in column 1 is a comment and is not read into the table.

The first parameter after the ")" can either be a userid or a T=type (i.e. userid type). The userid supports the wildcard characters "?" and "*". A "?" is considered to match any character in the corresponding position of the userid. A "*" is considered to match all or no characters in the corresponding position of the userid. For the userid type support, if the type given here matches the type assigned to the userid in the code record, a match is made. The CODUPD program can be used to assign a type to a userid or group of userids. The userid type supports values from 0 to 255. Zero is the default type value in the code record if a type has not been assigned previously. See the MUSIC Save Library file $COD:ID.TYPES for suggested settings of the type field in the code record.

The days parameter represents an expiry date for the specification. This sets a local expiry date for the specified user(s) or userid types, overriding the system expiry date (i.e. the MAIL.CONFIG EXPIRE parameter). This allows you to set an expiry date for a group of users to be a shorter or longer time than the system default. Valid values are 0 to 36500. Zero is the default value. Zero allows the user to set any expiry date upto the maximum allowable by the Mail facility of 36500 days. This parameter has the same properties as the MAIL.CONFIG EXPIRE parameter. This rule is enforced for surrogate support when you are sending mail for someone else on their behalf.

The days parameter is also used to expire mail items in a user's mailbox. If a piece of mail is older than is allowed by the expiry date setting, it is deleted from the user's mailbox. The mail facility filters expired mail items from the mailbox and does not present these items for selection when a user wants to view mail.

Instead, these items are placed in a list and they are deleted from the mailbox when the user exits viewing mail, refreshes the mailbox, or exits the program. 500 expired mail items can be deleted from the mailbox in any one instance in this way. If more expired items exist in the mailbox, they are not presented for viewing, and they will be deleted the next time mail is viewed. This process is repeated until there are no further expired mail items in the mailbox.

*Note:* When you send mail to a number of local recipients, the expiry date set for the item is the maximum of the expiry dates of all the recipients and the sender, and not the expiry date of the sender alone.

The items parameter is an optional parameter that represents the number of mail items which will be presented to a user for viewing even though there are more items in the mailbox. This forces the user to manage his/her mailbox since they have to delete mail items presented to view the other items in the mailbox. Valid values are 0, 12 to 32767. Zero is the default value. Zero allows the user to view all items in the mailbox without imposing a limit on the number of mail items presented.

The POP Freq parameter is an optional parameter that defines the frequency with which a user can gain access to his/her mailbox via a POP mail client. Some POP mail clients allow a user to check for mail as frequently as desired. Sometimes, even a one minute interval is allowed. Usually, a ten minute interval is sufficient. Everyone is sharing the network resources. Frequent mail checking can increase the network load and thus reduce the network throughput for everyone.

Setting this value for the user in this table, allows access to the mailbox via a POP mail client only so frequently. Eventually, the user will get the message and set the frequency to check for mail in their POP mail client to what is allowed by the system.

The value given here for a specification sets a local POP frequency for the specification, overriding the system POP frequency (i.e. the MAIL.CONFIG POPD POPFRQ parameter). This allows you to set a POP frequency for a group of users to be a shorter or longer time than the system default. Valid values are 0 to 1440. A value greater than zero represents minutes. Any value given must represent one day or less of time. Zero allows the user unlimited access to her/his mailbox via POP. This value has the same properties as the MAIL.CONFIG POPD POPFRQ parameter.

A message is returned to the POP mail client indicating the time to wait before access to the mailbox is granted. The Eudora POP mail client shows the to wait. Other POP mail clients may not show this information.

Examples:

```
) ABCD* 30 100
) T=0 30
) T=1-255 61 0
```

The first line allows any userid starting with ABCD to set an expiry date of no more than 30 days in advance of the send date, and limits mail item presentation to the first 100 items in the mailbox.

The second line allows any userid with a userid type 0 to set an expiry date of no more than 30 days in advance of the send date. Since the items parameter is not given, these users are not imposed with a limit on the number of mail items presented to them when they view their mail and presents all items.

The third line allows any userid with a userid type between 1 and 255 to set an expiry date of no more than 61 days in advance of the send date. It also does not impose a limit on the number of mail items presented to them when they view their mail and presents all items.

# Running the MAIL.CLEANUP Program

The MAIL CLEANUP program, $EML:MAIL.CLEANUP, is designed to delete mail text and distribution files that have expired. By using the expiry date encoded in the tag of these files, the program deletes the file if it's expiry date is in the past.

MAIL CLEANUP can be run when people are using the MAIL system. Also it can be cancelled at any time, since it does not touch the mailboxes and cause contention with mailbox access. File errors encountered while this program is running are reported in the program output, and the program proceeds. The program output gives a status line every one thousand files examined, and a summary is printed at the end of the report. Sample output is shown later in this section.

It is recommended that the MAIL CLEANUP program be run on a monthly basis (or more often if a site chooses).

This program requires the LSCAN and FILES privileges to run.

The program $EML:CHKMBOXES described in the MAIL Utility Programs section can be run to obtain statistics on the mailboxes.

### How the EXPIRY DATE and TIME are set on Mail files

Mail text and distribution files have an associated expiry date and time. When a user sends mail either via the SENDMAIL or MAIL programs, an expiry date is set for the sent mail. When either program is invoked, a default expiry date is established for the user. The user can override his/her default expiry date at send time and create a shorter expiry date for the sent mail. The user's default expiry date is assigned from an entry for the user in the mail authorization file, $EMD:MAIL.AUTHOR. If that does not exist, the MAIL CONFIG program EXPIRE parameter is used.

When RDMAILER receives mail to deliver to a local user or users, it uses the maximum expiry date of all recipients for the mail's expiry date. This expiry date is taken from the mail authorization file for each recipient, or from the MAIL CONFIG program EXPIRE parameter if an entry doesn't exist for the recipient just as described above. Postdated and Recurring mail are handled differently. The user is allowed to set any delivery date for either of these mail types. The expiry date set for postdated mail is the delivery date plus the expiry date allowed. The expiry date set for recurring mail is the day after the last delivery of the recurring item. Each mail item sent from the recurring mail item establishes its an expiry date allowed as each item is treated as an "original" send. As stated above, the expiry date allowed is taken from the mail authorization file for each recipient, or from the MAIL CONFIG program EXPIRE parameter if an entry doesn't exist for the recipient.

*Note:* When you send mail to a number of local recipients, the expiry date set for the item is the maximum of the expiry dates of all the recipients and the sender, and not the expiry date of the sender alone.

### Guidelines for the PERIOD parameter

The MAIL CONFIG program PERIOD parameter is used by the MAIL CLEANUP program. The PERIOD parameter represents the number of days before the current date for which a mail file will be deleted if it was created on or before this new date. For example, if PERIOD=20 and today's date is 30NOV90, then all mail files created on or before 10NOV90 are deleted. The default setting for PERIOD is PERIOD=0 which deletes the file only if the file's expiry date as encoded on the file's tag is before the current date. This is the preferred method for running MAIL.CLEANUP. This allows the Mail facility programs to determine the expiry date to set for the mail to be sent, and it allows MAIL.CLEANUP to honour the set expiry date.

Since mail items in the mailbox are expired and removed from the mailbox by the Mail facility programs independent of the MAIL.CLEANUP program, it is okay to have the value used to set the expiry date for mail sent to be a value less than the PERIOD value for the MAIL.CLEANUP program. The items are removed from the user's mailbox by the Mail facility programs prior to the mail text file being deleted off of the system by MAIL.CLEANUP. The user never sees an item for which the mail text file does not exist. If the reverse is done and the PERIOD value for the MAIL.CLEANUP program is less than the value used to set the expiry date, the user may see an item for which the mail text file does not exist. Then the item can only be deleted by the user.

If your site's mail policy changes and you must change the value used to set the expiry date for the user or instead decide to change the PERIOD value, it may turn out that the adjustment may allow the user to view an item for which the mail text file does not exist. This item can only be deleted by the user. This situation may arise since mail already received by the user has an associated expiry date that was assigned by the previous scheme. Depending on how the change is made, the Mail facility programs may not be able to expire the mail items before they are presented to the user.

## MAIL CLEANUP Program Listing

The following is a listing of the MAIL CLEANUP program, $EML:MAIL.CLEANUP.

```
/SYS NOPRINT,REG=512,TIME=MAX
/FILE 1 N($EML:MAIL.FCLEAN.OUT) NEW(REPL) LR(133) SP(500) DEF
/LOAD XMON
CLEANUP N($EML:MAIL.FCLEAN.LMOD)
PERIOD=0
```

The following is a listing of the file created by a sample run of MAIL CLEANUP, $EML:MAIL.FCLEAN.OUT.

```
Mail Cleanup MON APR 18, 1994 05:13:58
using parameters FCODE=$MD? PERIOD=  0

The date used for comparisons in this program is WED OCT 20, 1993
 (ie current date - period)
05:13 so far files,totspace,dels,delspace=        0        0        0
05:14 so far files,totspace,dels,delspace=     1000     3672       14
05:15 so far files,totspace,dels,delspace=     2000     7554       34
05:16 so far files,totspace,dels,delspace=     3000    11046       49
05:16 so far files,totspace,dels,delspace=     4000    14482       71
05:17 so far files,totspace,dels,delspace=     5000    18318       86
05:18 so far files,totspace,dels,delspace=     6000    21910      101
05:18 so far files,totspace,dels,delspace=     7000    25678      123
05:19 so far files,totspace,dels,delspace=     8000    29318      143
05:20 so far files,totspace,dels,delspace=     9000    32776      158
05:20 so far files,totspace,dels,delspace=    10000    36368      171
05:21 so far files,totspace,dels,delspace=    11000    40172      184
05:22 so far files,totspace,dels,delspace=    12000    44126      195
05:22 so far files,totspace,dels,delspace=    13000    47764      207
05:23 so far files,totspace,dels,delspace=    14000    51696      218
05:24 so far files,totspace,dels,delspace=    15000    55128      232
05:24 so far files,totspace,dels,delspace=    16000    59272      252
05:25 so far files,totspace,dels,delspace=    17000    63044      275
05:26 so far files,totspace,dels,delspace=    18000    67058      291
05:27 so far files,totspace,dels,delspace=    19000    70492      305
05:27 so far files,totspace,dels,delspace=    20000    74024      314
```

```
Total number of mail data files =     20250
Total space of mail data files =     74918 K
Number of files deleted=       316
Space of files deleted=       978 K
JOB TIME  325.31 SERVICE UNITS
```

# MAIL Utility Programs

## MAILBOX

This program displays the mailbox filename for a given userid. If the userid is not given, the mailbox filename for the signed-on user is displayed. This program can be run from ADMIN 4 5 7. The program is invoked as follows:

```
MAILBOX userid
```

## $EML:CHKMBOXES

This program looks at all of the mailboxes on the system and generates a statistics report for these mailboxes. Mailboxes are flagged in the report if they are larger than a certain size, have not been used for a number of days, have no corresponding userid, have a disabled userid, have a userid that has not been used for a number of days, have an overflow mailbox, or have encountered an error in the code table lookup.

The namelist parameter BIGBOX flags mailboxes in the report that are larger than BIGBOX K. The default value for BIGBOX is 300 (ie 300K). The namelist parameter NDAYS flags mailboxes in the report when either a mailbox has not been accessed in NDAYS days or a userid has not been accessed in NDAYS days. The default is 60 days.

The job output is written to units 6 and 1. Unit 1 is defined as file @CHKMBOXES.OUT. The program requires the privileges FILES, LSCAN, and CODES to run.

The following is sample job output from this program.

```
 -- CHKMBOXES --    MON APR 18, 1994   09:05
 Options:  BIGBOX=   300K   NDAYS=    60
$MBB:BGUEST                2K         BOXOLD               IDOLD
$MBB:B$000000           1584K  BIG                                OFLOW
$MBG:BJUNK                10K                       IDOFF  IDOLD
<stuff deleted here for brevity >

 Total number of mailbox files:       144
 Total space of mailbox files:       7790 K
 Number of extract errors:              0
 BIG     Number of large mailbox files:                   6
 BOXOLD  Number of old mailbox files:                    47
 NOID    Number of mailboxes with no userid:              0
 IDOFF   Number of mailboxes with userid disabled:        2
 IDOLD   Number of mailboxes with old userid:            52
 OFLOW   Number of mailboxes with overflow:               1
 IDERR   Number of userid lookup errors:                  0
```

## MAILBOX.FIX

From time to time, a user's mailbox will get damaged. This damage usually is an inconsistency in the mailbox control record and the true number and types of mail records in the mailbox. This is usually caused by editing the mailbox and manipulating the records and filing the corrupted version.

MAILBOX.FIX is used to re-align the control information in the mailbox with the actual contents. The program is invoked as follows:

```
MAILBOX.FIX userid
```

where *userid* is the userid of the corrupted mailbox.

## MSTAT

MSTAT is used to produce a snapshot of the current utilization of mail files. These are the files that are used by the mail facility to store mail text and distribution files. This program can aid in deciding when to schedule MAIL.CLEANUP.

The program is invoked as follows:

```
MSTAT
```

This program also generates a histogram in the file $EML:MAIL.FILE.COUNT.

## UCRADD

This program is used to either display or generate UCR records for a code series. This program is automatically invoked by the configurator program to generate the ucr records for the FCODE series, and by MSTAT to display the UCR's.

To display the UCR's for FCODE of $MD?, FMIN of 1 and FMAX of 7 use:

```
UCRADD $MD 1 7 GET
```

To add the UCR's:

```
UCRADD $MD 1 7 SET
```

To delete the UCR's:

```
UCRADD $MD 1 7 DEL
```

The MUSIC system no longer updates the UCR for files described by the FCODE, FMIN, and FMAX namelist parameters as of MUSIC/SP 3.1 when FCODE=$MD?. Thus the get feature of this program can not return accurate information.

## $EML:MAIL.MAKE

This program is a REXX exec that runs the various steps required to make the various Mail facility programs. It checks to make sure that the required object decks are online before running the link-edit for any of the programs. It can run the various steps required to make the load modules for MAIL and put them into the LPA. A system reIPL is required to put the new version of the MAIL program into production.

$EML:MAIL.MAKE can also relink the following programs: SENDMAIL, RDMAILER, GETMAIL, PROFILE, and MAILBOOK. All of these programs except for RDMAILER are put into production immediately after the link-edit is run as the link-edit recreates the load module. The RDMAILER BTRM must be restarted for it to use the new load module. This program can be run from ADMIN 4 5 8 "Remake Mail facility after local mods".

## $EML:MAILER.PROFILE.MK

This program makes the mailer profile, $EML:MAILER.PROFILE, for RDMAILER. This program should only be used by sites which rely exclusively on RDMAILER to deliver the mail. In this case RDMAILER does not send all of its mail to a MAILER for delivery (ie the RDMAILER options MAILER and MNODE are not used). This program can be called with or without parameters. You are prompted for the parameters if you do not specify the parameters when the program is invoked.

The program can be invoked as follows:

MAILER.PROFILE.MK XMAILER_NAMES_filename DOMAIN_NAMES_filename

- XMAILER_NAMES_filename is the MUSIC filename of the XMAILER NAMES as distributed by BITNIC. This file must be in its plain text format, ie not CMSDUMP or NETDATA format.

- DOMAIN_NAMES_filename is the MUSIC filename of the DOMAIN NAMES as distributed by BITNIC. This file must be in its plain text format, ie not CMSDUMP or NETDATA format.

$EML:MAILER.TEMPLATE serves as the template for $EML:MAILER.PROFILE. If you have any local mods, eg a MUSIC test system, you can define it in the template and it will be copied into the mailer profile.

You should request from BITNIC that the XMAILER NAMES and DOMAIN NAMES files be sent to you automatically when they are updated. If necessary, you can "GET XMAILER NAMES" and "GET DOMAIN NAMES" from LISTSERV@BITNIC.

This program stores the new mailer profile as MAILER.PROFILE on the user's account. The user must copy the file to $EML:MAILER.PROFILE to replace the previous version of this file. Also the RDMAILER BTRM must be told to use the new file. This can be done by issuing the MUSIC console command "/reply tcbn DOM" where tcbn is the RDMAILER BTRM tcb number.

The file $EML:MAILER.PROFILE can be edited manually using ADMIN 4 5 4 "Update Mailer Profile".

## $EML:INTEREST.BUILD

This program makes the index for the search engine required in the List Manager facility. It runs an ITSBLD job with input $EML:INTEREST.GROUPS and produces the index file $EML:INTEREST.WIDX.

See the description of the List Manager facility in the Mail Administration section below.

## $EML:MSG.BLDR

This message builder program takes the raw text message files and creates a message file usable by the Mail facility. The produced file has a couple of important characteristics: it contains record displacement information as hexadecimal values in the file, and it is created with the public attribute. The files created by MSG.BLDR and used by the MAIL facility should all be put in the RAM disk. The program is invoked as follows:

```
$EML:MSG.BLDR input_msg_file output_msg_file
```

So, for example, if you had changes you wanted to make to the Kanji version of the MAIL messages file, make the changes to the raw text messages file $EML:MAIL.MSGSK.S using the Editor with language set to Kanji, and then run the following:

```
$EML:MSG.BLDR $EML:MAIL.MSGSK.S $EML:MAIL.MSGSK
```

After the build is done, you should then reload the RAM disk if $EML:MAIL.MSGSK is in the RAM disk.

The raw text message files are normally stored offline and may have to be restored from the source tapes if use is required.

## $EML:MSG.CHECKER

This program can be used to verify the output of the $EML:MSG.BLDR program is correct. MSG.CHECKER takes the output from MSG.BLDR as input and produces a file identical to MSG.BLDR input. This validates the build operation. The program is invoked as follows:

```
$EML:MSG.CHECKER input_msg_file output_msg_file
```

Don't specify the same filename for the output_msg_file as the input_msg_file for the MSG.BLDR program. You will destroy the real message file for the MAIL facility.

So, for example, if you had made changes to the Kanji version of the MAIL messages file and have already run MSG.BLDR and you want to check the output messages file, you would run the following:

```
$EML:MSG.CHECKER $EML:MAIL.MSGSK MAIL.MSGSK.S
```

After the program is done, you should then do a compare on the output file MAIL.MSGSK.S and $EML:MAIL.MSGSK.S to guarantee they are equivalent. The compare is done as follows:

```
COMPARE $EML:MAIL.MSGSK.S MAIL.MSGSK.S
```

The raw text message files are normally stored offline and may have to be restored from the source tapes if use is required.

## $EML:UNRES.CK

This program is a REXX exec that checks the unresolved external references generated by the link-edits of the MAIL facility (i.e. running $EML:UNRES.CK). It runs a compare job of the expected unresolved external references which are stored by the exec in the file @UNRES1, against the unresolved external references the link-edit generated which are stored by the exec in the file @UNRES2. If the files do not compare equal, a new unresolved external reference has been introduced by the link-edit and this may indicate an error in the link-edit.

This exec is called by $EML:MAIL.MAKE for each link-edit step performed to allow the system administrator to verify that the link-edit step worked.

## $EML:NETCNV

The $EML:NETCNV program converts a file in NETDATA format created by the SENDFILE program into a readable file. The input filename is accepted as a program parameter, and it is automatically replaced with

the converted file.

The program is invoked as follows:

```
$EML:NETCNV input_filename
```

### $EML:QPUTI

The $EML:QPUTI program adds a RFC821/822 compliant mail file to the MUSIC reader queue so that it can be delivered as mail by RDMAILER to the recipients. The program takes an input filename as a parameter. The file is deleted after it is added to the reader queue. The program is invoked as follows:

```
$EML:QPUTI input_filename
```

### $EML:MAIL.BROADCAST

This facility can be used by system support personnel to broadcast a mail message to a large number of users. This broadcast facility delivers a mail message to each recipient. See the topic "Mail Broadcast Facility" later in this chapter.

The program is invoked as follows:

```
$EML:MAIL.BROADCAST
```

## MAIL Administration

### End of Year/Semester Cleanup

Your site may delete userids from the code table once in awhile, for example when a student has graduated from your institution. Since the mailboxes are not stored on the user's userid, but on system userids, you may also want to delete the mailboxes associated with these deleted userids. The way to handle this situation is to:

1. delete the mail text and distribution files
2. delete the userid from the code table
3. delete the mailbox

The mailbox does not contain the mail text per se, but it contains filenames where the mail text is stored. So the mail text and distribution files, which are also stored on system userids, must be deleted as done in step 1 above or by the MAIL CLEANUP program.

*Notes:*

1. The MAIL CLEANUP program only looks at the mail files on the system and does not examine mailboxes. So the MAIL CLEANUP program can be run to delete the mail files, but it does not have the power to remove the files unconditionally from the system. It only deletes mail files either when the expiry date of the file is in the past when the PERIOD parameter is set to zero, or when the creation date of the file is before the current date minus PERIOD days if the PERIOD parameter is not set to zero. So, the mail files could remain on the system until their expiry date has past. One approach to remove the user's mail files immediately is to follow the instructions given in the subject below entitled "How to clean up a mailbox quickly" before you remove the userid from the system.

Less work will be required at the end of the semester when MAIL CLEANUP is run regularly. MAIL CLEANUP purges expired mail text and distribution files from the system, returning the freed space to the system. A year's worth of accumulation can translate to a lot of purging to do at the end of the year.

2. Step 2 and 3 above can be done in one step by using the DELMAILBOX parameter of the CODUPD program's DELETE command. See the CODUPD program description for further details.

3. If you choose not to do steps 1, 2 and 3 together, you can do these three steps individually. First run a GETMINFO job to get all of the items from the user's mailbox and delete them. Then delete the userid from the code table. Then run MAILBOX userid (where userid is the userid whose mailbox you want to delete) to display the filename for the mailbox. Since the mailboxes are stored on system userids, using this program is the method required to determine the filename for a mailbox. The last step is to purge the file representing the mailbox.

   The following is a sample GETMINFO job which deletes all of a user's mail. Turn VIP ON before running the following GETMINFO job. The VIP privilege gives you access to the user's mailbox without signing on to the user's account.

   ```
   GETMINFO JUNK DISCARD ALL DELETE FOR(userid)
   ```

4. See the special subject below entitled "Mailing List Mail, Subscriptions, Unsubscriptions, and Digests" for information regarding this topic and end of semester issues.

## Mailing List Mail, Subscriptions, Unsubscriptions, and Digests

If your system is connected to BITNET or the Internet, your users may be allowed to subscribe to mailing lists and receive list mail. Typically, a user receives more mail than they send and mailing list subscriptions can generate a tremendous increase in incoming mail traffic on the MUSIC system. This increase in demand can impact your MUSIC system, so it must be monitored.

### Subscriptions

Users can subscribe to a mailing list directly after they find out how to subscribe to it, or they can use MAIL's main menu item 8, the List Manager facility, to subscribe to a list. The List Manager facility uses a list of lists from the Internet to allow users to subscribe to the lists therein. These subscriptions to lists done via the List Manager facility are stored in the file @MSUBL. If the userid has a subcode "sss", then it is suffixed to the filename (ie @MSUBLsss).

Your local mail policy may allow restrictions in the handling of list mail. An appropriate entry in $EMD:MAIL.AUTHOR could disallow a user or users from receiving external list mail. Further, you could add an appropriate entry in $EML:NORECEIVE to not allow a user or users to receive any external mail.

### Unsubscriptions

After you remove a userid from the system, there is a chance that mail from a mailing list will continue to be sent to the removed user. In this case, RDMAILER will automatically return a delivery error message to the sender in a special format recognized by the BITNET LISTSERV and LMAIL programs. When either of these programs receives this error message, the userid is automatically removed from the list. Other mailing list manager programs may not support this feature and you may have to ask the list manager to remove the user or users from the list. If a user has used MUSIC's List Manager to subscribe to various lists, the file @MSUBL or @MSUBLsss as described above exists on the user's account. This file contains subscription information that could be used to unsubscribe a user from these lists.

The postmaster's mailbox will receive a copy of the delivery error message returned to the sender when a mail recipient is unknown locally as would be the case when a userid is removed from the system.

There is a requirement of the LIST serviced by the BITNET LISTSERV program to unsubscribe a user automatically when it receives this special delivery error message sent to its list by RDMAILER. The LIST must be set up with the "Auto-delete =Yes,Semi-Auto" feature. This feature is not the default when the list has the "Validate= All" feature set. Suggesting this to the list owner would be beneficial to him/her as they would not have to look at all the "No such user" messages they receive. They would be dealt with automatically. Also, MUSIC users refusing mail from a list or lists cause a special delivery error message to be sent to the list. These users are dealt with in a fashion similar to the case just described where a recipient does not exist (ie the userid has been removed from MUSIC) and that intended recipient would be removed from the list if the LIST has the described feature enabled.

The postmaster can also manually unsubscribe a user from a mail list. This is convenient if the automated process described above has not removed the user from the mail list. When an invalid or disabled local user is not removed from a mail list, the mail sent by the list is returned to the sender and a copy is given to the postmaster. When the postmaster is viewing this item, the UNSUB command can be used to manually unsubscribe the user from the mail list. See the topic "Unsubscribing a User from a Mail List" in the Mail Filter Program section below for further information.

When the postmaster's incoming mail is being viewed, the UNSUB command can be used to unsubscribe a user from a the oria mail list

**Digests**

Mailing List Digests are a convenient way to follow a list of interest and get numerous posting as one mail item. The digest can be viewed with the MAIL DIGEST feature within VIEW.

Digest subscriptions are network friendly. You receive one mail item and not a multitude of individual items that make up the digest. Thus digest delivery demands less system resources.

## MAIL and the MUSIC Save Library

With heavy use of the MAIL system, you may find that your Save Library index is heavily populated. Sometimes, the index's auxiliary blocks may be used more frequently. This may slow down access to the Save Library in general. The MFINDEX program can be used to display information about the Save Library index.

If you find that access to the Save Library is slow (this could be manifested by the use of a lot of auxiliary blocks) you may want to increase the number of segments in your Save Library index. See the NEWINDEX program for further details on enlarging the index.

The MAIL facility is coded to allow 1296 userids for mail text and distribution files. The generated userids for use with these files are $MDAA to $MD99, which is 36*36 or 1296 combinations. As such, the MAIL facility can take advantage of upto 1296 segments in the Save Library index even though the number of segments must be a prime number. So the Mail facility functions without change as the number of segments in your Save Library index changes.

The FCODE namelist parameter does not appear within the MAIL CONFIG facility, although it gives the starting part of the userids ($MD) used for the mail text and distribution files. It should not be changed as system changes have been made to accomodate this fact to give a performance improvement. System performance may degrade if it is changed to something other than $MD.

## Adding MAIL.CLEANUP to your AUTOSUB job

The file $ADM:AUTO.EMAILCLN defines a set of jobs to run MAIL.CLEANUP. This set of jobs shuts down RDMAILER, runs MAIL.CLEANUP, then restarts RDMAILER. You could add this job to the BTRM that runs AUTOSUB by using ADMIN 4 9. One idea would be to add an MSTAT job before and after the MAIL.CLEANUP job to product a snapshot of the current utilization of mail files.

## MAIL Logs

The default mail log file is $emd:@maillogyyyymmdd, where yyyymmdd is the date the log was started. The log contains records for all mail sent and received, and it also contains progress and error message records generated by RDMAILER. The entries in the mail log from RDMAILER are prefixed with an asterisk. The log file is not created if the MAIL CONFIG parameter LOG is set to F.

The mail log records are generated by the MAIL, RDMAILER, and SENDMAIL programs. These log records are passed to the MUSIC LOG server BTRM with the application name MAIL. The MUSIC LOG server matches the received record's application name, MAIL, versus the application name as defined in the LOG server's definition file $PGM:SYSLG.DEFINE. It then writes the log records to the log file associated with application name. The default mail log file as defined in the LOG server's definition file is $EMD:@MAILLOG. With the default DAILY specification given, the mail log file used by the LOG server is $EMD:@MAILLOGyyyymmdd. You can change the filename or make the logs MONTHLY if so desired.

These logs can take up a lot of space on your system. You may want to delete the older files periodically. Also, it may be advisable to give the DAILY specification for the mail log if your site has considerable mail traffic.

For further information on the MUSIC LOG server BTRM, see its description in this manual.

## How to clean up a mailbox quickly

You may find a need to delete mail items from a mailbox quickly. As an example, a user has been away for awhile and forgot to sign off of a automatic distribution list. Now his/her mailbox is too large and MAIL complains that it needs "x more bytes of storage to run". A simple way to delete mail enmasse from a mailbox is to use the DISCARD option of the GETMINFO or GETMAIL program. You can use the GETMINFO or GETMAIL program's FOR parameter to discard mail from someone else's mailbox provided you have the privilege associated with the program's SNOOP option. See the description of MAIL.CONFIG for details on the SNOOP option.

**Examples:**

1. This example deletes all mail from the user's mailbox that was received from the NOVELL list.

```
GETMINFO JUNK DISCARD ALL DELETE SELECT FROM NOVELL
```

2. This example deletes all mail from the user's mailbox that was rece ived before 01APR93. It also shrinks the user's mailbox to the absolute minimum file size required.

```
GETMINFO JUNK DISCARD ALL DELETE SHRINK SELECT BEFORE 01APR93
```

## Space used by the Mail Facility

A user may find that he or she has received too much mail and is getting a message from MAIL that it "requires at least x more bytes of storage to run". This message results because MAIL is using the allotted region size to keep an incore copy of the mail items. The region size in $EML:MAIL dictates this value. Each mail item requires 136 bytes. So increasing the region size 100K would provide the user with the ability to see about 750 more mail items.

$EML:BIGMAIL is a version of $EML:MAIL with a region size of 3000K.

## How to load a Conference automatically with mail received

You can load a Conference directly with mail you receive. First you must copy the mail to a mailbook. This can be done by using the MAIL XL command or COPY command with the MAILBOOK option set to Y (yes), or by using the GETMAIL program. After the mailbook is created, you append the mailbook to the conference. This can be done manually by going into the conference and appending the mailbook to the conference. You can do this automatically by using the following command:

```
CONF conference_name APP topic mailbook_filename NOH
```

where *conference_name* is the name of the conference and *topic* is the topic within the conference which you want to add to mailbook_filename is the filename of the mailbook to add to the conference.

This automatic function is allowed for conference owners or any userid with the FILES privilege. All other non-privileged users must use the manual method to append mail to the conference topic.

In the following example, all of the mail that has been received from the NOVELL list is put into a mailbook and then that mailbook is appended to the NOVELL topic within the LANS conference.

```
GETMAIL NOVELL.LOG ALL DELETE SELECT FROM NOVELL
CONF LANS APP NOVELL NOVELL.LOG NOH
```

See the topic "Merging Information into Topics" in the help for the Conferencing facility for further details.

## How to have mail received handled automatically

There is a special feature within the Mail facility called MAGFIL support that allows you to have your incoming mail handled automatically. This feature is only available for users with the VIP privilege. If you turn on the VIP privilege and then invoke the Mail Profile facility, you will see another field on the General Mail Options screen. The field is called "File to be executed when mail arrives for you". Enter a filename in this field. This file could be a job that runs GETMINFO or GETMAIL.

There are many uses for this support. You could have mail from a particular person logged and deleted automatically. You could have mail from a automatic mailer or listserv logged and deleted or added to a conference automatically.

The mail text and distribution filenames are passed to the program designated in the "File to be executed when mail arrives for you" field when it is executed.

See the file $EML:MAGFIL.SUPPORT for further details.

## Setting up a MAIL route for AUTOPR

The MUSIC system can be configured with a ROUTE destination of MAIL. This means that output from a job or a print file can be routed to you as mail. One use for this is to have output from the AUTOSUB job routed to you as mail. This way you can check your mail for the job output instead of looking for printed output.

This function helps a site move to a paperless office.

See the topic "Configuring the AUTOPR Program" for further details.

## List Manager Facility Functions

The List Manager facility allows users to manage their subscriptions to BITNET and Internet mail lists. MAIL main menu option 8 invokes this facility.

The List Manager uses a file found on the Internet for the browse function from which users can view the available lists for subscription. This file is known as the Internet "interest groups" or "list of lists". It does not contain all of the mail lists on the Internet and BITNET. It is up to the individual mail list owner(s) to send a request into the "list of lists" coordinator for inclusion into this master list. This list is available via anonymous ftp on sri.com in the netinfo directory as file interest-groups. It may also be obtained through email by sending a message to mail-server@sri.com with "send netinfo/interest-groups" in the body of the message.

This file should be saved as the public MUSIC file $EML:INTEREST.GROUPS. The MAKPUBL program can be used to make the file public.

After this file has been saved public, you must run $EML:INTEREST.BUILD to make the ITS index required for the search engine within the List Manager facility. See the previous section "Mail Utility Programs" for a description of this program.

If you do not want to make this facility available to your users, rename the file $EML:LM.NOTAVAIL to $EML:LM. This will replace the List Manager with a pop up window telling users this facility is not available. The file attributes on $EML:LM should be public (PUBL) and execute only (XO). The MAKPUBXO command can be used to make a file public and execute only.

Each user has a file @MSUBLsss where their subscriptions to mail lists made via this facility are kept. If the userid has a subcode "sss", then it is suffixed to the file name. Otherwise, the file name is @MSUBL.

## Changing the Expiry Date of Incoming Mail

A privileged user can reset the expiry date on incoming mail for any user. This privileged user must possess the privilege given in the Mail facility SNOOP parameter or the user must have the VIP privilege. When you are looking at the user's incoming mail, enter an "E" (expire) select code beside the mail item and a menu is presented to change the expiry date and time associated with the mail item.

This function can be applied to any item that is displayed that has not had its mail files deleted by the MAIL CLEANUP program or has not been deleted by the user to preserve that item.

When a mail item has expired, it is not displayed to the user and it is deleted from the user's mailbox upon exit or a refresh.

## How to Retract a Delivered Message

There are two ways to retract a message delivered locally that should not have been delivered or was delivered by accident. If the item was sent with acknowledgements on, and the copy of the mail item on the outgoing mail list has not been deleted, you could expire the item to some date in the past. This updates the other local boxes. The other method requires the VIP privilege. So turn on VIP and then enter MAIL. You would then delete the item from the recipient's mailbox by typing their userid in the Mail For field, locating the item in their incoming mail list, and deleting it. You must look at the mail for all recipients and perform this operation.

## How mail items and files get deleted

Typically, a user deletes a mail item received and it is removed from the mailbox and the associated mail files are deleted if this user is the only recipient of the mail or is the last recipient to delete the mail. This deletion or tag reduction is done by the MUSIC LOG server BTRM for the MAIL program. GETMAIL and GETMINFO do their own deletions.

However, what happens when a user does not manage their mail? This is where MAIL CLEANUP becomes important. It can delete mail files off of the system based on expiry date or creation date depending on how it is run, but it does not modify the mailboxes. This task is done by the MAIL facility programs themselves, removing expired mail items from a user's mailbox automatically when the program is invoked. For instance, when a user invokes the MAIL program, the mail is examined before presentation to determine if any mail has expired. If there is mail to expire, it is not presented to the user in the incoming/outgoing mail list. It is removed 500 items per invocation from the mailbox when the user exists the MAIL program. Thus, in a sense, the mailbox management is done by the users themselves.

When expired mail is removed from the mailbox, it is the actual mail item that is removed. The mail text and distribution files may have already been purged from the system by a previous run of the MAIL CLEANUP program or they will be deleted on the next run.

## How does the expiry date get set on a mail item

Each user has a default expiry date, defined as the current date plus some value, usually a number of days. This value can be the system default expiry date which is defined by the EXPIRE namelist parameter. If an entry in the mail authorization file defines a different value (ie retention period) for the user, then this value is used in preference to the system default expiry date.

When mail is sent, the expiry date set is the maximum expiry date of all of the recipients' expiry dates. If the sender has acknowledgements turned on, the sender is also considered a recipient and its expiry date must be considered for the maximum expiry date. Acknowledgements are turned off by the system if the mail is sent offsite or is autoforwarded.

When mail is sent, an expiry date can be set on the Sending Mail Options screen. The displayed expiry date is the default for the sender. The expiry date displayed on this screen is a user override. If it is changed, it will be used instead of the maximum expiry date for all of the recipients. If the sender is a VIP user or the userid possesses the privilege defined by the SNOOP namelist parameter, the sender can set any expiry date in this override area.

The Mail facility limit to any expiry date is 100 years forwards from the current date.

## Adding your site's mail policy to Mail main menu item A, MAIL FAQ

If your site has a mail policy, you may want to add it to MAIL's FAQ (Frequently Asked Questions) item on Mail's main menu. The MAIL FAQ text can be found in the file $HLP:@EM.FAQSCR. You could either edit the file and make your changes or you could use the IDP program to update the file as this file is a help file. If you were to use IDP, you would run "IDP E $HLP:EMHELP", then select the topic FAQSCR by positioning the cursor on the topic and pressing F5 to edit the topic. Save your changes and your done. Don't forget to add a title line to the menu for the mail policy.

## How to Create Your Own MAIL Menu

You can create your own MAIL menu and include or exclude items from the distributed MAIL main menu. The program $EML:FMAIL uses the MAIL namelist parameter EXMAIL for this purpose. EXMAIL allows any MAIL subfunction to be invoked and exit MAIL totally when the subfunction is done. For example, if you run FMAIL 1, Incoming Mail would be presented. When you exit from this menu, you return to wherever you were when you invoked FMAIL 1. The following is a sample BBS MAIN.MENU file that could be a MAIL main menu.

```
)title Mail Facility
 Place the cursor on an item and press ENTER or RETURN.

   +?MAILSEND-? Create and Send Mail
   +?INMAIL  -? Read Incoming Mail
   +?OUTMAIL -? Outgoing Mail
```

The following is the sample BBS TOPICSX file associated with the above MAIN.MENU file.

```
05  TOPICS      TOPICSX
09  MAIN.MENU
08  MAILSEND    |FMAIL 2
06  INMAIL      |FMAIL 1
07  OUTMAIL     |FMAIL 3
```

## Restricting MAIL Access

There are a number of things you can do as the system administrator to curtail abuse or restrict access to mail by a user.

1. Delete the userid's files, the code itself and the mailbox with CODUPD. This is the most severe action that can be taken.

2. Disable the userid. All mail sent to this user is returned to the sender with the message "Unknown userid". Even though the userid is disabled, a user with the VIP privilege can inspect the mailbox and delete items if that is so desired.

3. Do not allow the userid to access the mail program, send mail offsite or to a particular person or place. This is accomplished by either adding an entry in $EMD:MAIL.AUTHOR barring the user from mail access, or sending mail to this person or place. You can also add an entry in $EML:NORECEIVE to disallow a person from receiving mail from the outside.

4. Reduce the number of mail items that a user can view as an aid in helping the user manage their own mailbox. Add an entry in $EMD:MAIL.AUTHOR for the user with this limit set to some number.

5. Reduce the retention period of mail for this user. This aids the user to manage their own mailbox. Add

an entry in $EMD:MAIL.AUTHOR for the user with this limit set to some number.

## Running Multiple RDMAILER BTRMs

RDMAILER can be configured to run as more than one BTRM. Up to 10 auxiliary RDMAILER BTRMs can be configured to handle incoming mail. One RDMAILER BTRM is available for outgoing mail. The MAIL.CONFIG program can be used for this configuration. Additional BTRMs are used to handle increases in incoming mail, like mailing list mail.

Before you reconfigure the number of RDMAILERs you run, make sure you have generated a MUSIC nucleus with enough BTRM addresses before you add the new BTRMs. The BTRM statement in $GEN:NUCGEN.JOB is used for this purpose.

Also, after running the MAIL.CONFIG program to reconfigure your setup, you must shutdown and restart the VMREADX BTRMs and all of the RDMAILER BTRMs. Only then will the BTRMs use the new configuration. A MUSIC shutdown will have the same affect.

RDMAILER can be configured in a number of ways. Here are two sample configurations.

- It can be configured as only one BTRM, handling both incoming and outgoing mail. This configuration is suggested if your site uses mail infrequently.

- It can be configured as two incoming BTRMs and one outgoing BTRM. The two incoming RDMAILERs both share the work of emptying the reader queue, and delivering the mail. This configuration is suggested for moderate mail usage.

There are some important features that RDMAILER uses when the MAIL system is configured to run more than one RDMAILER. First, the outgoing RDMAILER never handles incoming mail. Its duty is only outgoing mail, along with postdated, recurring, and autoforwarded mail.

The incoming RDMAILERs all handle only incoming mail and never outgoing mail, except it handles the autoforwarding for mail which it is trying to deliver. This reduces the autoforwarding demands on the outgoing BTRM. Thus each incoming RDMAILER has a different PCODE definition for the post office mailbox. The different PCODEs are generated automatically by the MAIL.CONFIG program.

Incoming RDMAILER's duty is to handle entries in the reader queue. Each incoming RDMAILER is assigned an id number n such that each RDMAILER only looks at those reader queue entries in the queue for it. So if there are two incoming RDMAILERs, then RDMAILER 1 processes entries 1,3,5,7,..., and RDMAILER 2 processes entries 2,4,6,8,... Thus, RDMAILER must know two factors, how many incoming RDMAILERs are running, and which one is it. Then it can process the right entries from the reader queue. VMREADX must also know how many incoming RDMAILERs are running so it can wakeup the correct RDMAILER when there is work to be done. The wakeup uses the "ENQNAM" namelist parameter for RDMAILER "$SRDMAILnn" where nn is 1 to 10. It is automatically generated by the MAIL.CONFIG program. You should not change this value, otherwise VMREADX will never wakeup RDMAILER. RDMAILER does have a builtin wakeup mechanism in case it never gets woken up.

You may find that the incoming RDMAILERs cannot keep up with the influx of MUSIC reader files or reader queue files. This can be determined by a buildup of MUSIC reader files, or a message on the MUSIC console indicating that VMREADX is delaying for "n" minutes as the reader queue is full. Before adding another incoming BTRM, make sure that all the VMREADX and RDMAILER BTRMs are functioning correctly. Also, the buildup could be temporary, and it may not warrant an additional BTRM. If the buildup persists for intermittent periods throughout the day, addition of one or more incoming BTRMs will alleviate the problem.

## When a RDMAILER BTRM stops

If a RDMAILER BTRM stops for any reason, a message is written to the MUSIC console and a tell message is posted to the userid indicated by the NOTIFY parameter in the MAIL CONFIG facility. This is normal if you have shut down RDMAILER by the standard method (i.e. the console command /REPLY tcbn STOP, where tcbn is the tcb number of the BTRM running RDMAILER). When RDMAILER stops any other reason, this action notifies you that something has happened.

To verify that a RDMAILER BTRM has stopped, issue the MUSIC console command /REPLY tcbn HELLO, when tcbn is the tcb number of the BTRM running RDMAILER you suspect has stopped.

### Incoming mail processing stopped

When a RDMAILER BTRM processing incoming mail stops, it is usually caused by invalid mail headers in the mail item it was trying to deliver. The first step in restarting the RDMAILER BTRM is to determine which reader file RDMAILER was processing when it stopped. The previous topic in this section entitled "Running Multiple RDMAILER BTRMs" gives basic information on how to determine which reader queue entry RDMAILER was processing when it stopped. Each reader queue entry has an associated file which is the actual spool file. Entry number one in the reader queue is used as a pointer to the next available entry in the reader queue. Thus entry two in the reader queue represents the first entry handled by a RDMAILER BTRM. The file $VMR:RQ.00001 is associated with entry two, as is $VMR:RQ.00002 with entry three and so on. The first entry in the reader queue that the stopped RDMAILER BTRM is to handle is typically the offending entry. Now that the entry has been identified, you can delete the offending reader file associated the entry, rename the reader file to another file name on your userid, and restart RDMAILER, or you can edit the file, fix the addressing, file the changes, and restart the RDMAILER BTRM.

Another situation can arise if the RDMAILER BTRM is not restarted. The VMREADX BTRMs read spool files and add entries into the reader queue that represent the files it has read. An incoming RDMAILER BTRM processes these entries in the reader queue, delivers the mail and frees up the entries so they can be reused by the VMREADX BTRMs. If RDMAILER is not running, it is not freeing up entries in the reader queue for reuse and there are less entries available for the VMREADX BTRMs to use. If all of the entries are used, the VMREADX BTRMs cannot do their job and they must shutdown too. When there are no entries available for the VMREADX BTRMs to use, they attempts to process spool files 10 times, delaying a number of minutes each time. If after 10 tries the reader queue is still full, they shut down and the spool files are not processed. Each time a VMREADX BTRM shuts down because the reader queue is full, it copies the current spool file it is processing to $VMR:VMFILE.nnnnn, where nnnnn is a number. These files can be placed back into the reader queue once an item becomes available for use by the $EML:QPUTI program. Note that when a VMREADX BTRM delays or stops because the reader queue is full, messages are written to the MUSIC console indicating what has happened.

### Outgoing mail processing stopped

When a RDMAILER BTRM processing outgoing mail stops, it is usually caused by bad addressing in the mail item it was trying to deliver. The first step to restarting the RDMAILER BTRM is to identify which outgoing mail item the RDMAILER BTRM was processing when it stopped. You must look at RDMAILER's mailbox to find the mail item. This mailbox is the mailbox associated with the userid defined in the PCODE namelist parameter for that RDMAILER BTRM. If your userid possesses the VIP privilege and you have turned it on, or your userid has the privilege given in the MAIL CONFIG facility SNOOP parameter, you can look at this mailbox by typing the PCODE userid in the MAIL For field of the MAIL program. Look at the incoming mail items in PCODE's mailbox for the offending mail item. (The items are incoming mail because they are sent by users to the RDMAILER BTRM for delivery.) Note that postdated and recurring mail are delivered by the RDMAILER BTRM, so the first items in the mailbox may not be the item sought. Now that the item has been found, you can delete the offending mail item and restart RDMAILER, transfer the item to another mailbox where it can be looked at, and restart RDMAILER, or you can edit the

file, fix the addressing, file the changes, and restart the RDMAILER BTRM. While viewing the mail item, the ECHO NAME command is helpful in determining the name of the file.

## Delivering SENDFILE mail items

The SENDFILE program can be used by either a VM, MUSIC, or BITNET user to send a file to a MUSIC user. SENDFILE uses the NETDATA format to encode a file. Although rare, a SENDFILE mail item can appear in the postmaster's mailbox that has not been decoded to a readable file. When this happens, the mail item has two identifiable characteristics. The first line of the mail body starts with the character string *\INMR01, and has similar strings in the first few lines. These strings are the NETDATA control record markers. The second characteristic is that the mail item subject is of the format "Sendfile: x y a".

The steps described below can be used to convert a mail item from its NETDATA format into a readable format, and then deliver the mail item to the intended recipients.

Steps:

1) Use the GET command when viewing the mail item to store it in the file name given by the mail subject.
2) Run "$EML:NETCNV filename" to convert the file to a readable format. The $EML:NETCNV program is described under "Mail Utility Programs".
3) Run "$EML:QPUTI filename" to have RDMAILER deliver the mail. The $EML:QPUTI program is described under "Mail Utility Programs".

## Mail and Public Directory Editor work file

The personal and public mail directories used by the Mail facility are updated using the MUSIC editor and the REXX macro facility. A directory can reach a size where a message appears telling the user that it cannot insert a new record into the file or that F3 or the file command cannot be used. The message appears because the Editor work file is full and it must be increased in size. Three programs, $EML:DIRECT, $EML:DIRECT.PUBLIC, and $EML:MKNICK use the common REXX macro $EML:NAMDIR.MAC to update the appropriate directory. All three programs have an Editor work file defined as /FILE 1 within the program. Increase the number of records (NREC) value on the /FILE 1 statement to increase the Editor work file.

## Creating a site Mail Profile

You can modify the site Mail Profile if your site has a requirement to create a standard mail profile for all new mailboxes. If a site Mail Profile is not created, new users do not have any Mail Profile options preset and they must set the options themselves.

ADMIN 4 5 3 "Update site Mail Profile" can be used to create/modify a site Mail Profile that is copied to all new mailboxes at creation time. The Mail Profile facility is used to create the site Mail Profile. The site Mail Profile is stored in $EMM's mailbox. Only the Mail Profile from $EMM's mailbox is copied to new mailboxes at creation time. The Name and Email Id fields of the site Mail Profile are not copied to the new mailboxes.

This facility can be used to set any Mail Profile options that should be copied to new mailboxes. For example, the site wants mail sent to be copied to a file automatically. On the "Create and Send Mail Options" screen, simply set the "Copy the mail to be sent to a file" to Y, set a filename where to save the mail to be sent (e.g. MAIL.LOG), and set Append to this file to 'Y'. Filenames specified should probably not be userid prefixed. That way when the Mail Profile in the new mailbox is actually used, the filenames all default to the userid of the new mailbox.

## Overflow Mailboxes

On occasion, mail cannot be added to the mailbox. The MAIL facility makes an attempt to deliver the mail instead of returning it to the sender. In this case, the mail is appended to a overflow mailbox for the interim. Invoking any Mail facility program that reads incoming mail copies the overflow mailbox to the real mailbox and updates the mailbox counters. The GETMAIL, GETMINFO, SENDMAIL, and POPD programs all do this function, as does reading incoming mail in the MAIL program or refreshing the incoming mail list.

The overflow mailbox has the same filename as the real mailbox, except the first character in the filename is O and not B. For example, if the real mailbox was $MBC:BABCD, the overflow mailbox filename would be $MBC:OABCD.

## RDMAILER Kill File

RDMAILER has the facility to delete any incoming mail item from a defined mail sender without delivering the mail item. This feature is commonly called a kill file and it is used to control mail spamming and mail bombing.

ADMIN 4 5 B "Update RDMAILER kill file" can be used to create/modify the kill file. The file $EML:KILLFILE is used to contain the kill file entries which are a list of email addresses. When RDMAILER finds incoming mail from any email address given in the kill file, RDMAILER deletes the incoming mail without delivering it and reports what it has done. The email addresses are listed one per line. Wild card characters * and ? are allowed in the addresses. RDMAILER can automatically detect when the kill file has been updated and it reloads it automatically.

The RDMAILER namelist parameter KILFIL defines the file $EML:KILLFILE for kill file processing.

The RDMAILER namelist parameter KILSIZ can be used to define the the number of characters you want to set aside in core for kill file entries. The default is KILSIZ=16000 which allows for 200 entries of 80 characters each. Since KILSIZ is given as a character count, you must define KILSIZ for the maximum number of characters expected in the kill file. For example, if there are 100 entries in the kill file, KILSIZ=8000 is required (i.e. 100*80=8000). If KILSIZ is set to a value greater than 16000, the MAIL.CONFIG parameter RDREG must be adjusted to its current value plus the difference in K between the new and old KILSIZ value. RDREG is the region size to use for RDMAILER. If a larger requirement for the kill file is set, then the region size for RDMAILER should be set higher by the same amount.

*Note:* Kill file processing can slow down incoming mail delivery by RDMAILER. It is best to keep the kill file to the minumum number of entries required.

## Changing the Release Date of Mail

Any user can send postdated mail. Postdated mail can be sent by setting the release date and/or time on the Sending Mail Options screen of the Create and Send Mail function to something in the future, and then sending the mail. When the mail is sent, it is given to the outgoing RDMAILER BTRM for delivery. RDMAILER will deliver the mail when the release date and time have been reached.

The user can change the release date and time on the mail after the user has sent it if acknowledgements were turned on when the mail was sent. In this case, a copy of the mail appears in the user's outgoing mail list. The select code R-Release allows the user to change the release date and time. When the release date and time are changed, they are updated in the copy in RDMAILER's mailbox. If the updated date and time represents the past, RDMAILER delivers the mail. Note that if RDMAILER has already released the mail and delivered it, the release date and time cannot be changed.

Prior to RDMAILER releasing the mail and delivering it, if acknowledgements were not selected when the

mail was sent, the only copy of the mail resides in RDMAILER's mailbox waiting to be delivered. A privileged user can change the release date and time on the incoming mail item in RDMAILER's mailbox. This privileged user must possess the privilege given in the Mail facility SNOOP parameter or the user must have the VIP privilege. When you are looking at $EMD's incoming mail, enter an "S" (releaSe) select code beside the mail item and a menu is presented to change the release date and time associated with the mail item. As above, if RDMAILER has already released the mail, it cannot be found in RDMAILER's mailbox as it has already been delivered.

This function can be useful to correct entry errors in the release date and time of a mail item.

## Mail Broadcast Facility

This facility can be used by system support personnel to broadcast a mail message to a large number of users. This broadcast facility delivers a mail message to each recipient. There may be other methods more suitable to announce something to your user community. Here is a brief list of some other methods of announcement to consider:

- MUSIC console command /MESSAGE ALL message_text
- MUSIC console command /DAILY message_text
- place text in file $PGM:ALERT.FILE and it is displayed at signon
- place text in file $PGM:NEWS.DATA, a one line entry in $PGM:ALERT.NEWS, and it is displayed when user enters MUSIC command NEWS
- create a popup window in FSI main menu so users see message text when the autoprog FSI is invoked after the user signs on

This facility uses the Mail Utility Program $EML:QPUTI to add one or more RFC821/822 compliant mail files to the MUSIC reader queue. The mail files are then delivered as mail by RDMAILER to the recipients. The FILES and SYSCOM system privileges are required to use the Mail Broadcast facility.

The Mail Broadcast facility has both a full screen interface for running the facility interactively and a batch mode which can be used by the MUSIC/SP automatic job submission facility.

In interactive mode, the Mail Broadcast facility can create an edit session for the user to create the recipient list and text to send if no filename is given for either field.

Any email addressing errors that occur while trying to deliver the mail broadcast are send back to the user as a mail message. In this case, the user is the email address designated as being the mail broadcast sender.

#### Mail Broadcast Facility versus the MAIL program

There are some differences between sending a mail broadcast using this facility and using the MUSIC/SP MAIL program or SENDMAIL program.

- the Mail Broadcast facility is faster
- the Mail Broadcast facility does not resolve nicknames so nicknames cannot be used with it
- both SENDMAIL and the Mail Broadcast facility can be used for the automatic submission of jobs (These jobs are run in BATCH mode.)

Help is provided when the program is run.

# MAIL Filter Program

The postmaster's mailbox can be a busy place and the burden on a human to deal with the mail can be very time consuming. The postmaster mailbox filter program can be used to deal with the mail automatically, eliminating or reducing the burden with the postmaster mail.

The postmaster filter program's main purpose is to automatically unsubscribe disabled or deleted local user-ids from the mailing lists for which they still receive list mail. Users refusing mail from mailing lists are treated the same way as a disabled or deleted userid by the filter program. They are unsubscribed from those mailing lists for which they are refusing mail.

The filter program includes the following features:

- ability to track unsubscribes for the previous month
- ability to remember unsubscriptions done for the current week and not reissue the unsubscribe for that week
- sourceroute handling
- mass mailer deletion
- mail gateway handling

## Where does Postmaster Mail come from?

The postmaster's mailbox can receive mail from two possible sources. It can be sent mail directly or by the MUSIC Mail facility's message transfer agent RDMAILER, which sends the postmaster a copy of the returned mail when a delivery failure occurs.

When the postmaster is sent mail directly, it is typically a request for help. However, there is a special case when the postmaster is sent mail by an automated process that is reporting an error. The postmaster's email address is used for the Errors To: field of the MUSIC Mail facility's autoforward feature. Thus any autoforwarded mail delivery errors are sent to the postmaster. This setting permits the postmaster to correct the problem and allow the mail to be delivered.

The delivery errors RDMAILER sends to the postmaster can be either mail sent to an invalid user or any other type of problem.

When mail is sent to an invalid user, the mail is returned to the sender and a copy is sent to the local postmaster. So when the MUSIC system administrator disables or removes a userid, the user may still receive mail from mailing lists. Since the userid is disabled or removed, it is now invalid. So the postmaster mailbox gets a copy of the returned mail.

## Running the Postmaster Mailbox Filter Program

Consult the section below entitled "First-Time Setup" before running the filter program for the first time.

The program can be run either automatically or manually. To run automatically, the MUSIC system administrator must set up an AUTOSUB job for the postmaster filter program. The file $ADM:AUTO.PSTMST defines a job to run the postmaster filter program. You could add this job to the BTRM that runs AUTOSUB by using ADMIN 4 9. It can be set up to run automatically every day of the week at a given time by

MUSIC's automatic submission facility. It can also be set up to run more than once a day.

When the program is run, the program statistics are appended to the file $EMP:PSTMSTSTATyyyymm, where yyyymm represents the year and month. The statistics are also displayed with the program output as is the actual actions taken by the program. When the job is set up to run automatically, it can be configured to have the program output sent to you via email by using R=MAIL on the ID statement in the job.

The program looks at all items in the postmaster's mailbox and takes action on what it is programmed to handle. It can deal with "mail gateway temporary send errors" and "mass mail senders". Also, it can unsubscribe users from identifiable mail lists. Items the program has taken action on are deleted after whatever processing that is required for the item is done.

For those items the program cannot take action on, the program does not mark these items in the mailbox as read (ie old). So if you read the postmaster's mail, all items are marked new, and you cannot tell which items were left by the filter program and which items arrived after the program was run. It may be convenient to run the filter program manually just before reading the postmaster's mail. Then, only the current mail will be visible.

The program can be run manually on any userid that meets the following requirements:

1) the userid must have the SYSCOM, FILES, and LSCAN privileges

2) the userid must be allowed as a surrogate for the postmaster's mailbox or the userid must have the privilege defined by the SNOOP namelist parameter of the MUSIC Mail facility. Your MUSIC system administrator can provide this information.

The program is run manually by typing at *Go:

```
$EML:PSTMST.UNSUB
```

## What does the Filter Program do?

The filter program looks at all items in the postmaster's mailbox. There are a number of cases that the program can handle: mail sent to the postmaster, mail sent by mass mailers, mail sent by identifiable mailing lists, mail sent by identifiable automated mail list managers, chain mail, and friendly mail destined for an invalid user.

1) Mail sent to the postmaster

   Mail sent by an individual requesting help is not acted on by the filter program.

   Mail gateways can report a temporary condition where the mail could not be sent for "x" time but will be retried for "y" days. This mail does not signify a failure but simply represents a delivery status notification (DSN). This type of mail is deleted from the postmaster's mailbox by the filter program. The file $EML:UNSUB.GATEWAYS defines the mail gateways recognized by the filter program and what criterion is used to indicate the DSN which can be ignored.

2) Mail sent by a mass mailer

   A mass mailer or spammer is someone who has sent mail to a large number of addresses. The postmaster mailbox will receive the mail for any invalid userids who were a part of the mailing. The filter program can deal with mass mailers statically and dynamically.

   In the static method, mass mailers can be identified when you are reading the postmaster's mail. The email addresses associated with these mass mailers can be added a static list. This list is used by the

filter program in future runs to delete mail sent by any email address within this static list. The static list is the file $EML:UNSUB.SPAMMERS.

In the dynamic method, the postmaster filter program identifies mass mailers that have sent mail to a number of invalid users. It deletes the mail from these senders once an initial number of mail items has been surpassed.

See the topic "Mass Mailers" below for further details.

3) Mail sent by identifiable mailing lists

When a userid is disabled or removed by the MUSIC system administrator, the user may still receive mail from mailing lists. Since the userid is disabled or removed, it is now invalid. When mail is sent to an invalid user, the mail is returned to the sender and a copy is sent to the local postmaster. When the filter program encounters an item from an identifiable mailing list, it sends a unsubscribe request to the mailing list manager on behalf of the user, and deletes the item from the postmaster's mailbox.

The filter program can identify a mail list if the mail is sent by listproc, listmanager, majordomo, list-serv, mailserv, maiser, mxserver, owner-, -owner, -request, -error, -errors, -admin, hpcnews@newsmaster.tgc.com, automail@power.globalnews.com, membership@match.com, info.apple.com, mailbase@mailbase.ac.uk, or wsmith@wordsmith.org.

The filter program can deal with unsubscribe requests that require a confirmation and subscription renewals. Some mailing list owners may have made these features available with the mailing list they administer.

4) Mail sent by identifiable automated mail list managers

Mail sent by any of the senders listed below is examined by the filter program to determine if the mail was sent in response to the unsubscribe request and it is deleted if it matches that criteria. These senders are:

```
*REPLY*, *POSTMAST*, *DAEMON*, *MAJORDOMO*, *SUPPORT*, *AGENT*,
*MAILBASE-ADMIN*, *LISTPROC*, *LISTSERV*, *MAISER*
```

5) Chain mail

When you are reading the postmaster's mail, you may notice that someone has sent chain mail to one or more local users. Some institutions have a Code of Conduct that forbids chain mail. If a local user is sending chain mail to invalid local user email addresses, you can have the filter program send this user the appropriate section of your Code of Conduct as a warning. The file $EML:UNSUB.CHAIN contains definitions for the variables required to support chain mail identification in the original subject of the mail item. If the originator address matches chain mail address that you trap, the warning is sent and the item is deleted. If the addresses don't match, the item is just deleted.

6) Friendly mail destined for an invalid user

Mail is often sent to invalid local users and winds up in the postmaster's mailbox because it could not be delivered. If the original mail subject contains any of a list of friendly words or phrases, e.g. hello, then the mail can be deleted. The list of friendly words/phrases, which when matched by the filter program allow it to delete the mail, is defined in the file $EML:UNSUB.FRIENDLY.

Otherwise the mail is left in the postmaster's mailbox.

## Filter Program Statistics Data

The filter program statistics data is appended to the file $EMP:PSTMSTSTATyyyymm when the program is finished. It is also displayed in the program output.

The following defines each of the displayed fields:

| | |
|---|---|
| Items in Postmaster mailbox | number of items in the mailbox when the program is started |
| Items read today | number of items actually examined today |
| Items deleted | total number of items deleted from mailbox |
| Errors | total number of errors encountered during processing, including mail fetch errors and unsubscribe request send errors |
| Total potential unsubscribes | number of total possible unsubscribes done, including unsubscribes sent and matches with previous unsubscribes |
| Unsubscribes sent | number of unsubscribes sent this run |
| Matches with previous info | number of unsubscribes already done within the last week |
| Entries reread to match | number of mail items that had to be reread to reach the bounced mail delimiter line |
| Entries sent to postmaster | number of items sent directly to the postmaster that were not acted on |
| Autoforward resolve errors | number of items in mailbox that were autoforward resolve errors |
| Subject not signifying a rejection | number of items that had the letters UNDELIVER in the subject |
| Mail gateway messages deleted | number of items sent to the postmaster by mail gateways that were temporary send errors |
| Friendly messages deleted | number of items which were deleted because the mail subject matched friendly phrases |
| Chain mail messages encountered | number of items identified as chain mail and deleted |
| Renewals deleted | number of items that represented subscription renewals that were deleted |
| Confirm unsubscription sent | number of items which required a confirmation of an unsubscribe request and the confirmation request was sent |
| Sendfile mail items delivered | number of sendfile items that were decoded and delivered |
| BCCs deleted (potential spammers) | number of items that had a BCC: RFC822 header and were deleted. Spammers use this method to suppress the mail list. |
| Total spams detected and deleted | |

number of items sent by mass mailers that were deleted

Total unique dynamic spams detected
number of unique addresses detected where at least "newspam" mail items from this email address were found in the postmaster mailbox. "newspam" is a variable defined in the filter program. The default is newspam=5.

Spam count for spamidn — number of items sent by mass mailer n as defined in $EML:UNSUB.SPAMMERS that were deleted

Dynamic spam detected n times for address:
number of times where "n" mail items from this email address were found in the postmaster mailbox. "n" is at least "newspam" in size. "newspam" is a variable defined in the filter program. The default is newspam=5.

Since the filter program can track more than one mass mailer, if any mass mailers are detected during the run, the final number of items detected for each mass mailer is displayed with the statistics.

## Filter Program Actual Actions Output

The filter program displays actions it has taken when it is run. These actions include:

- reporting the number of items found in the mailbox
- adding an entry to the WEEKDAY file and incore cache to record the unsubscribe request (see further information below)
- reporting an item has been previously unsubscribed
- reporting an item that has been sent by a mass mailer
- reporting any error that occurred during the run

## Other Features of the Filter Program

The current week's unsubscribe requests are stored in the files $EMP:WEEKDAY.n, where n is 1 to 7 representing Sunday to Saturday. When the program is run on Wednesday, the unsubscribe requests done for that day are stored in $EMP:WEEKDAY.4. An unsubscribe request is issued once for the week. Then, for the week following the request, any mail from the mailing list sent to the same user is simply deleted from the postmaster's mailbox without resubmitting the unsubscribe request.

When the entries in the WEEKDAY files are older than one week, the filter program copies them to the file $EMP:MONTH. This file contains the previous 4 weeks of unsubscribe requests. Also, the filter program keeps only the previous 4 weeks of requests in the file $EMP:MONTH. All entries in that file older than 4 weeks are deleted.

The file $EMP:MONTH is not used by the filter program. However, sometimes the user will continue to receive mail from the mailing list despite your efforts to unsubscribe them. This file can be used as a reference tool to find out when the unsubscribe request was sent. The previous request information stored in this file identifies that it was sent. However, since the user is still receiving mail from the mailing list, it appears that the unsubscribe did not take place. The unsubscribe request needs to be resent either automatically or manually, and possibly sent to another address. The unsubscribe request is resent automatically the week after the first request was sent by the filter program if the user is still receiving mail from the list. The topic

below "Unsubscribing a user from a mail list when viewing the postmaster's mail" describes how to resend the unsubscribe request manually.

## First-Time Setup

The following things must be done before the program is run for the first time:

1) the userid where the program is run must have the SYSCOM, FILES, and LSCAN privileges

2) the userid must be allowed as a surrogate for the postmaster's mailbox or the userid must have the privilege defined by the SNOOP namelist parameter of the MUSIC Mail facility. Your MUSIC system administrator can provide this information.

3) edit the file $EML:UNSUB.DEFINES and set the variables "mysite" and "sourcerouteid" to agree with your system settings.

4) edit the file $EML:UNSUB.GATEWAYS and define your local mail gateways or set gateways to zero if you have no local gateways. Also set the gateway mail criteria allowing deletion of the mail.

5) edit the file $EML:UNSUB.CHAIN and set the variables required for chain mail trapping.

6) run ADMIN 4 5 1 3 to reconfigure the Mail facility. The program $EML:QPUTI has been added to the configuration process and it must be created.

## Components

| | |
|---|---|
| $EML:UNSUB.DEFINES | local definitions required by the filter program |
| $EML:UNSUB.GATEWAYS | defines gateways and the criteria used for the filter program gateway matching |
| $EML:UNSUB.SPAMMERS | defines the known mass mailers |
| $EML:UNSUB.CHAIN | defines the variables used to trap chain mail |
| $EML:UNSUB.FRIENDLY | defines the friendly phrases used to match against the subject of friendly mail that can be deleted |
| $EML:UNSUBCL | program that manages the unsubscribe requests and moves requests from the $EMP:WEEKDAY.n files to the $EMP:MONTH file, and deletes requests from the $EMP:MONTH file |
| $EML:GETMINFO | program used to get a list of the items in the postmaster's mailbox |
| $EML:GETMAIL | program used to delete mail items from the postmaster's mailbox |
| $EML:QPUTI | program used to put the unsubscribe request in the MUSIC reader queue so that RDMAILER can deliver it to the intended recipient |

## Mass Mailers

As described above in the topic "What does the Filter Program do?", under item "2) Mail sent by a mass mailer", mass mailers can be dealt with in a static and/or a dynamic way.

1) Static method
When you are reading the postmaster's mail, you may notice that someone has sent unsolicited mail to a number of invalid local userids. Instead of deleting all of the items manually, the filter program can be set up to delete these items for you. If the email address of the mass mailer is appended to the file $EML:UNSUB.SPAMMERS, future filter program runs will delete all mail from any mass mailer listed in the file $EML:UNSUB.SPAMMERS.

Email addresses can be appended to the file $EML:UNSUB.SPAMMERS manually or automatically. The instructions at the top of the file $EML:UNSUB.SPAMMERS document the steps required to manually append an email address to the file. The changes must be saved before the new mass mailer addresses are used by future runs of the filter program.

The MUSIC Mail facility's SPAMMER command can be used to automatically append the pointed to email address to the file $EML:UNSUB.SPAMMERS. The email address is not appended to the file if it is already in the file. The SPAMMER command can be entered only while viewing the postmaster's mailbox or for a user possessing the CODE privilege defined by the MAIL program SNOOP namelist parameter or the VIP CODE privilege.

The file $EML:UNSUB.SPAMMERS used to track the mass mailers can either be used in a static fashion, never changing, or in a dynamic fashion (i.e. being updated whenever a new mass mailer is encountered). With the growing escalation of mass mailers on the Internet, the dynamic use of this file is of great advantage to the postmaster.

It is up to the person running the filter program to update this file, either by adding an entry for a mass mailer when one is identified by reading the postmaster's mail, or by deleting a mass mailer when one has instituted an aging policy for entries in this file.

2) Dynamic method
The postmaster filter program can be set up to identify mass mailers dynamically as it runs. The filter program "newspam" variable defines the number of times a sender email address can occur before it triggers the dynamic spam detection function. The default is newspam=5 (i.e. identify 5 mail items from the same sender before deleting subsequent mail items from this sender). Setting newspam=0 turns off dynamic mass mailer detection. When more than "newspam" mail items have been identified from the same sender, further items from this sender are deleted.

When the filter program runs, if a mail item is not acted on by any other action, it is examined as a candidate for mass mailing. A list of fifty dynamic mass mailers is maintained while the program runs. An age and least recently used algorithm is used to maintain this list. An entry with at least "newspam" mail items encountered is always kept.

Note that "newspam" mail items are left in the mailbox and are not deleted by the filter program. These items can be deleted manually. However, it may be convenient to leave these items in the mailbox to catch subsequent mass mailings by this sender in the coming days. You may also find it convenient to add these sender email addresses to the file $EML:UNSUB.SPAMMERS so that future filter program runs will automatically delete items from these senders.

A list of the dynamic mass mailers is printed at the end of the program report.

## Unsubscribing a User from a Mail List

The MUSIC Mail facility's UNSUB command, which can be entered only while viewing a mail item in the postmaster's mailbox, can help the postmaster with the manual task of unsubscribing a user from a mailing list. This manual unsubscribe process requires the tailoring of the unsubscribe request by the viewer. When an item has been identified by the person reading the postmaster mail as list mail which should be acted upon, and the original local recipient was either invalid or refusing mail from the list, then the unsubscribe

request can be done.

After the UNSUB command is entered, an edit session is presented with a prototype unsubscribe request displayed. Care must be taken to preserve the format given, otherwise the request sent when the changes are filed may cause other mail system problems. The SITE and LISTX editor macros can be used to set the variables "site" and "list" to their respective correct value as determined by examining the file being edited.

The prototype unsubscribe request presented is an RFC821/RFC822 compliant email message. It is very important that the structure remain as presented and any edification must pay attention to dangling commas for the RFC822 To: field.

Some mass mailers suggest a method different from that described above to have an email address removed from their mass mail list. The method is usually described within the received mail and it requires sending a reply to the original mail item with the word "remove" in the subject. Be forewarned that by sending the remove reply, you are in fact registering for mass mail. Mass mailers are using this method of false advertising to get a list of valid email addresses which they can use in the future.

## MAIL Exits

A number of exit routines are available in the Mail facility for use. These routines, EXROUT, EXDPRT, EXRADR, and EXDCDR allow the site to tailor various aspects of MAIL.

The EXROUT exit routine is a replacement for calls to the MUSIC routetable to determine routes available for the user.

The EXDPRT exit routine can be used as a substitute for MAIL's printer display feature that allows a user to select the printer for printing his/her mail.

The EXRADR exit routine allows the site to have a selection menu for recipients as input when the user enters "?" in a recipient field in MAIL.

The EXDCDR exit routine allows the site to decode encoded mail when the item has been selected but before the user's selected operation is performed on the mail item.

The following lists the Mail programs which are exit aware, and lists which exits are available in which programs.

```
$EML:MAIL     - EXROUT,EXDPRT,EXRADR,EXDCDR
$EML:MAILBOOK - EXROUT,EXDPRT,EXDCDR
$EML:MPROF    - EXROUT
```

In order to install the exit procedures, you must do the following:

1. Replace the file $EML:MAIL.exit_name.S with your file.

2. Compile/assemble your exit routine into the file $EML:MAIL.exit_name.OBJ.

3. The program $EML:MAIL.MAKE can be used to make sure that any routines (.OBJ) exist on disk before the appropriate Mail program(s) are link-edited. (Note that .OBJ files are commonly not resident - you might have to use ADMIN 3 5 to restore these files.) Once satisfied, re-link the appropriate link-edit link-edit by running $eml:MAIL.MAKE and selectively running the link-edit for MAIL, MAILBOOK, and/or PROFILE with this program.

Note that the exit object decks are listed in the file $EML:MAIL.EXITS, $EML:MBK.EXITS, and $EML:PROFILE.EXITS. These files are included in the link-edit for their respective program.

The program $EML:MAIL.MAKE reviews the linkage editor output after the link edit is done. If there are no errors, then your new version may be in production. Note the the MAIL program requires a system reIPL or a SYSUPDATE to put the new LPA module for mail into production.

## Description of Mail Exits

### The EXROUT Exit

This exit routine is offered as an alternative to "CALL ROUTE(2,ROUTENAME,IRC)".

The EXROUT exit is invoked in MAIL, MAILBOOK and MPROF as

```
CALL EXROUT(IRC,ROUTE)
```

where

```
IRC:    integer return code
        -1=this exit routine is not active
           MAIL,MAILBOOK,MPROF will call route
         0=route okay as checked by exrout
       <>0=something wrong with the route
           MAIL,MAILBOOK,MPROF will post an invalid route message.
ROUTE:  8 character routename
```

### The EXDPRT Exit

This exit routine is offered as an alternative to the MAIL/MAILBOOK PRTGET routine invoked by "CALL PRTGET(ROUTENAME)".

The EXDPRT exit is invoked in MAIL and MAILBOOK as

```
CALL EXDPRT(IRC,ROUTE)
```

where

```
IRC:    integer return code
        -1=this exit routine is not active
           MAIL and MAILBOOK will call PRTGET.
        0=exit routine did processing, use route on return.
ROUTE:  8 character routename returned back to MAIL/MAILBOOK
        padded with blanks.
        If non-blank, MAIL/MAILBOOK will place the route name
        into the printer name field on the print screen.
```

### The EXRADR Exit

This exit routine is called when a "?" is placed in any of the address fields to send mail (ie TO or CC fields). The substitution address is returned as the same field passed.

The EXRADR exit is invoked in MAIL as

```
CALL EXRADR(IRC,RADR)
```

where

```
IRC:    integer return code
        -1=this exit routine is not active
           MAIL is continue processing "?" as an email address.
        -2=no substitution address was given
           This means that from your selection panel no address was
           selected.  In this case the "?" gives invalid userid.
        0=resolve radr
        <>0=something wrong in processing
            MAIL will post the error message IRC found in the mail
            mail messages file $EML:MAIL.MSGS)
RADR:   132 character email address returned to caller to resolve
            (must be padded with blanks)
```

**The EXDCDR Exit**

This exit routine provides a means for mail to be decoded after a mail item has been selected but before the user's selected operation is performed on the mail item.

The EXDCDR exit is invoked in MAIL as

```
CALL EXDCDR(IRC,MFILE)
```

where

```
IRC:    integer return code
        -1=this exit routine does no processing.
         0=exit routine did processing.
MFILE:  64 character mail text file passed from MAIL.
```