308-012
Issue 2

AT&T SNA/RJE Emulator +

User's and System Administrator's Guides

NOTICE

The information in this document is subject to change without notice. AT&T assumes no responsibility for any errors that may appear in this document.

Table of Contents

User's Guide

Basic Concepts 1
Using the AT&T SNA/RJE Emulator+ 4
Sending Job Files 8
Receiving Job Files 18
Cancelling Jobs 23
Displaying the Local Job Queue 28
Displaying the Log File 29
The Console Operator Command 30

System Administrator's Guide

System Set-up 1
Running the AT&T SNA/RJE Emulator+ 17

Appendices

Appendix A: Environment Variables A-1
Appendix B: Sample Configuration Source Files B-1
Appendix C: Sample Host Configurations C-1
Appendix D: NCP Gens D-1
Appendix E: Error Messages E-1
Appendix F: Program Check Error Codes F-1
Appendix G: Communication Check Error Codes G-1
Appendix H: DC3770 and RJE Messages H-1
Appendix I: **iscconfig** Command Options I-1

Index

X-1

Table of Contents

Basic Concepts 1
RJE Jobs 1
- The Job Queue 1
- Job Types 1
- LU Sessions 2
- Job Output from the Host 2

Command Options 3
Configuring Your Environment 3

Using the AT&T SNA/RJE Emulator+ 4
The **rje** Command 4
- The **-h** *rjepipe* Option 5
- The **-M** *msg_file* Option 6

The Application Program Interface 7

Sending Job Files 8
Send Command Options 8
Invoking the Send Command from the Shell 11
Invoking the Send Command from a C Program 12
- **srje( )** Return Values 14

Responses to the Send Command 15
- Local Messages 15
- Host Messages 17

Receiving Job Files 18
Automatic Routing of Job Output 18

User Notification of Job Complete 21
Tag Files 22

Cancelling Jobs 23
Invoking the Cancel Command from the Shell 23
Invoking the Cancel Command from a C Program 23
- **crje( )** Return Values 26

Responses to the Cancel Command 26
- Local Messages 26

Displaying the Local Job Queue 28

Displaying the Log File 29

The Console Operator Command 30

List of Figures

Figure 1: Basic **rje** Command Options 5

Figure 2: **rje** Commands and Functions for the User 7

Basic Concepts

RJE Jobs

The most basic RJE concept is the job. Any command that sends something to the host creates a job. Before a job is sent, the SNA/RJE Emulator assigns it a local job number. A message containing this job number is sent to the user's terminal. This number remains associated with the job until it has been completely transmitted to the host. To cancel the job before it is sent, or to determine the job's position on the local queue of jobs waiting to be sent, you must know this local job number. (Note that when the host receives the job, it assigns its own job number to it, and this new number serves to identify the job as long as it is on the host system. The SNA/RJE Emulator does not use the host job number.)

The Job Queue

Jobs that are waiting to be sent to the host are kept on a local job queue. At most 60 jobs can be on the queue at any one time. The job queue consists of files named J*nnnn*, where *nnnn* is the local job number. The job queue is really several queues split according to session number (LU) and job type.

Job Types

There are three different types of job that the workstation may send to the host; type J (for job), type A (for APPL REQ), and type L (for log on).

- Type J jobs are jobs that the host will process and run. They are assigned a job number by the host, placed on the host's job queue, and then processed. The data in a type J job includes JCL (Job Control Language) that indicates how the job is to be processed by the host. Generally a user submits only type J jobs.
- Type A jobs send inquiry messages, such as JES commands, to the host. These jobs appear to the host to have come from the 3770 console keyboard. They do not create host jobs; they simply cause messages to be sent back to the 3770 console. 3770 console messages sent from the host are displayed at the local terminal that has been designated as the 3770 system console. If a log file is designated at system start-up time, these messages are also written to that file.

- Type L jobs are used to log on to the host at the start of a session, and to log off from the host at the end of a session. Session log on and log off commands are privileged commands requiring root privileges.

LU Sessions

The SNA/RJE Emulator process, **dc3770**, supports up to six logical units (LUs). Each LU can be used to establish a separate session with a host. Generally, all six are connected to the same host RJE subsystem (such as JES). As such, they act as parallel channels for one 3770 workstation, allowing several jobs to be sent or received at the same time. Normally, the user does not need to specify the LU to be used.

Job Output from the Host

Job output received from the host is initially placed in the **output** subdirectory of the SNA/RJE Emulator's **runtime** directory (normally, **/usr/snaadm/runtime/output**). When the file has been completely received, it is submitted to the **disptch** process for automatic routing. Automatic routing allows each user to specify a directory or file in which the output from a job is to be placed.

There are situations in which the user may need to know how the System Administrator (using the **rje −a −b** command) has assigned the output devices. For example, if a received file is to be left in EBCDIC (normally code set translation to ASCII is performed), the user may have to route the output, via JCL in the jobfile, to a specific destination assigned for EBCDIC files. Consult your System Administrator for specific information on how the output assignments have been made on your system. (See "Assigning Output Destinations" in the *AT&T SNA/RJE Emulator+ System Administrator's Guide.*)

The host can send output to any of 288 logical destinations. Each possible combination of LU (1 to 6), medium (C, E, or P), and subaddress (0 to F) can be selected by the host. Usually the host makes no distinction between one LU and another, so 48 destinations remain (3 media, 16 subaddresses). Output from a job can be directed to a specific destination by inserting a JES card such as:

```
/*ROUTE   PRINT   RMT195.PR15
```

into the JCL of the jobfile. This example directs the printed output from the job to Remote 195, printer 15 (medium P, subaddress 15).

Command Options

The AT&T SNA/RJE Emulator+ uses both environment variables and default values to obtain the values of any missing options on a command line. Thus, the user can usually enter a simple command without having to specify any options.

The following describes, step by step, how the options for a command are determined:

Step 1. The option may be specified by the user on the command line.

Step 2. If it is not, and there is an environment variable associated with the option, the value of the environment variable will be used.

Step 3. If there is not an associated environment variable; or if the environment variable has not been set, a default value will be used. (Hard-coded default values are provided for most options.)

If a value for an option cannot be obtained, the command may abort, or result in an error.

Configuring Your Environment

Before using the SNA/RJE Emulator, you should configure your environment as follows:

Step 1. Set the **BR** environment variable equal to the *rjepipe* your System Administrator used to start the SNA/RJE Emulator, and export it. (Check with your System Administrator for the full pathname of the *rjepipe* you should use.) For example:

```
BR=/usr/snaadm/runtime/hostx
export BR
```

Step 2. Set your **PATH** variable to include the **runtime** directory for the SNA/RJE Emulator. For example:

```
PATH=/usr/snaadm/runtime:$PATH
```

Note that these changes to your environment can be added to your **.profile** file, so that they will be executed automatically each time you log in.

Using the AT&T SNA/RJE Emulator+

The rje Command

After your System Administrator has started the SNA/RJE Emulator, and your environment is configured, you can use your terminal to submit files to and receive files from a remote host. You can also cancel jobs queued locally, display local session status and job information, and page through the system log file.

You can perform all these functions by executing the **rje** command as a UNIX system shell command either interactively from the command line or from a script.

All **rje** commands have the following format:

rje *−func* [**−h** *rjepipe*] [**−M** *msg_file*] *cmdopts...*

Figure 1 lists the basic options, describes them, and gives the environment variables and default values used when they are not specified.

Option	Description	Default Environ. Variable	Default Value
–func	*func* designates the function of the command. It must be the first option after **rje**. Valid user options are: **–s** send **–c** cancel **–d** display queue **–f** view log file **–o** console operator	—	—
–h *rjepipe*	*rjepipe* specifies the pipe to the SNA/RJE Emulator process, **dc3770**. Standard option for all **rje** commands.	**BR**	HOSTA
–M *msg_file*	*msg_file* specifies the message file to be used. Standard option for all **rje** commands.	**GEMFL**	**rje.msg**
cmdopts	*cmdopts* are the options specific to each command. For details see the command descriptions.	—	—

Figure 1: Basic **rje** Command Options

The –h *rjepipe* Option

The user should take particular note of this option and of the default environment variable **BR**. The **rje** process is invoked whenever you issue an **rje** command. The **rje** process communicates your command to the SNA/RJE Emulator process, **dc3770**, using a named pipe specified by *rjepipe*. (The *rjepipe* name is designated by the System Administrator when the

dc3770 process is started.) When you issue an **rje** command, the **rje** process must be provided with the name of the *rjepipe* it should use. **rje** obtains the name of the pipe in one of the following ways, in order:

1. The user may specify the pipe name in the **rje** command by using the **−h** *rjepipe* option. To do this the user must know the exact name of the pipe that the System Administrator used to start the **dc3770** process. For example the user might enter:

   rje −s −h /usr/snaadm/runtime/HOSTA file1

   This command would result in **file1** being processed and sent by the **dc3770** process named by the pipe **/usr/snaadm/runtime/HOSTA**.

2. If the user fails to specify the pipe name, the user's environment list is searched for the variable named **BR**. If **BR** is set, its value is used as the name of the *rjepipe*. (This is the easiest and the recommended way of providing a value for the **−h** *rjepipe* option. See "Configuring Your Environment.")

3. If the **BR** variable is not set, the **rje** process will look for a pipe named HOSTA in the following directories, in order:

   - the user's current working directory
   - **/usr/snaadm/runtime**
   - **/tmp**

4. If these steps do not produce a value for the **−h** *rjepipe* option, the **rje** command will abort and the message `HOST NOT FOUND` will be displayed.

The −M *msg_file* Option

The **rje** process refers to a central message file. The default message filename is **rje.msg**. The system looks for this filename in the current directory, in **/usr/lib**, or in **/usr/snaadm/runtime**. The **−M** *msg_file* command line option or the **GEMFL** environment variable can be used to override the default, much as **−h** *rjepipe* and **BR** were used above. Generally, the user only needs to specify this option if a message file other than the default file **rje.msg** is to be used.

The Application Program Interface

The send and cancel commands can also be invoked from a user-written C program using the Application Program Interface (API) functions. The C program calls the API external function, which then executes the command. The list of options or arguments available to the API functions is the same as it is for **rje** commands invoked from the shell.

The API functions are found in the library **/usr/lib/librje.a**. Therefore, this library must be linked to the user's object module(s). One way to do this is to enter:

cc usrprog1.c usrprog2.c −lrje

where **usrprog1.c** is a source file containing a C language program. (See the **cc**(1) and **ld**(1) manual pages in the UNIX System V *Programmer's Reference Manual* for further information about compiling and loading programs.)

Figure 2 lists the **rje** commands and API functions available to the general user. (Additional commands requiring root privileges are described in the *AT&T SNA/RJE Emulator+ System Administrator's Guide.*)

Function	rje Command	API Function
Send Job Files	**rje −s**	**srje( )**
Cancel Jobs in Queue	**rje −c**	**crje( )**
Display Job Queue	**rje −d −j**	—
View the Logfile	**rje −f**	—
Console Command	**rje −o**	—

Figure 2: **rje** Commands and Functions for the User

Sending Job Files

You can send a file (or files) to a remote host by executing the send command, **rje** **–s**, from the UNIX system shell, or by calling the **srje( )** function within a C program.

In the first case, the send command is invoked just like any other regular task from the command line.

In the second case, the send command is executed via an Application Program Interface (API) to the C program. In this case the C program calls the API external function **srje( )**, which executes the send command.

In both cases, the list of options supplied to the send command is the same.

To permit efficient job processing, it is recommended that you define multiple card readers, printers, and punches in the definition of the remote Host you are using.

Send Command Options

This section lists all the valid options to the send command and explains their use.

–t *session* *session* specifies the LU session the command affects. Its value must be in the range 0-6.

Value	Meaning
0	Command applies to any LU session that is not busy. **dc3770** decides which session will be used.
1-6	Specific session to which the command applies.

This option defaults to **–t** 0.

−s *subaddr* *subaddr* specifies the destination subaddress (as known to the host), where subaddress is a value in the range 0-F (hexadecimal). For typical batch subsystems, subaddress 0 corresponds to RDR0, subaddress 1 corresponds to RDR1, etc; subaddress F frequently means any reader.

This option defaults to **−s** F

−m *medium* *medium* specifies the medium of the virtual input device (as viewed by the host). A different medium can be specified for each file sent.

Value	Meaning
C	Card (Default)
E	Exchange Diskette

This option defaults to **−m** C

The medium specification determines how the data are to be formatted by the SNA/RJE Emulator and unformatted by the host. In practice, there is no need ever to use this option; it exists to provide full 3770 functional equivalence.

−e specifies that the file to be sent is in EBCDIC; code set translation will not take place. The **dc3770** process will use IRS, new-line, or linefeed characters as the record delimiter for the file. This option applies to any files that follow until either a **−a** or **−x** option is specified on the command line.

−a specifies that the file to be sent is in ASCII. Code set translation to EBCDIC will take place. (Unless **−e** or **−x** is specified, files are translated into EBCDIC before transmission.) The **dc3770** process uses the new-line character as the record delimiter for ASCII files.

Since files are assumed to be in ASCII, **−a** is the command line default. Therefore, this option applies to

all files until a **−e** or **−x** option is specified on the command line.

−x [*nnn*] specifies that the file is to be sent transparently. Code set translation to EBCDIC will not take place. (Unless **−e** or **−x** is specified, ASCII files are translated into EBCDIC before transmission). *nnn* is a number in the range 1-512 that specifies the record length to be used. For card or console files (media C and K) the default record length is 80; for diskette files (medium E) it is 128. This option applies to any files that follow until either a **−e** or **−a** option is specified on the command line.

−k *'text'* specifies that a type A job is to be sent, where *text* is a console command for the host system. (This option precludes the use of the **−a**, **−e**, and [**−p**] *path* options.) *text* should be enclosed in single quotes to avoid any expansions and/or substitutions of the characters by the UNIX shell, and cannot be longer than 120 characters. This option must be the last one specified on the command line. (Note that **−w** is set automatically by this option.)

[**−p**] *path* *path* specifies the pathname of the file to be sent. The **−p** preceding *path* may be omitted unless the pathname starts with a dash (−). Pathnames relative to the current directory are expanded to their full pathname (relative to the **root** directory). Up to six files may be sent. (The total combined length of all pathnames cannot exceed 100 characters.) When multiple files are indicated, they are concatenated and sent as one file to the host. Note that the medium of origin (specified in the **−m** *medium* option) may change, and that some files may be in EBCDIC (see the **−e** option) and others in ASCII (see the **−a** option).

If *path* is a dash (−), the data to be sent is taken from the standard input. This is useful if the user is so deep in directories that the pathnames are too long.

Files can then be sent with a command like:

cat file | rje −s −

The dash can also be useful if more than six files must be sent as a single job.

−w specifies that the command is to wait until the file has been completely transmitted to the host before returning. (If there is an error, control is returned immediately.) This option is set automatically by the **−k** option.

The following two options to the send command are privileged options; you must log in as root to use them. They are described in detail in the *AT&T SNA/RJE Emulator+ System Administrator's Guide.*

−son [−q] Privileged System Administrator option. Specifies that a type L job is to be sent. It is used to log LU sessions on and off the host. **−q** specified with the **−son** option will log off the session(s) with the host. It is the equivalent of specifying **−k 'LOGOFF'**. If a pathname or **−k** *'text'* is not specified, **−q** is taken as the default.

−sof [−q] Privileged System Administrator option. Identical in effect to **−son.**

Invoking the Send Command from the Shell

The format to execute the send command from the shell is:

rje −s [−h *rjepipe*] **[−M** *msg_file*] [*option...*] *file...*

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings. (See "The **rje** Command" above.) Up to six filenames may be specified on one command line. (The command does not wrap around to the next line.) Multiple files are concatenated and sent as one job stream to the host.

Simple filenames are expanded during processing to their full pathnames. For example, **rje −s myjob** would be expanded to **rje −s /usr/mylogin/myjob**. The total combined length of all the expanded pathnames for one command cannot exceed 100 characters.

Examples

1. **rje −s jclfile1 sysin1 jclfile2 −e sysin2**

   Sends a job on any available session (by default) as card data (by default) to subaddress F (by default). The job consists of four files, of which the fourth is in EBCDIC, and will not be translated.

2. **rje −s −e file1 file2 −a file3 file4**

   file1 and **file2** are already in EBCDIC; **file3** and **file4** are in ASCII and will be translated to EBCDIC before transmission.

3. **rje −s jclhdr −x bindata −a jcltail**

   jclhdr and **jcltail** will be translated to EBCDIC; **bindata** will not be translated.

4. **rje −s −t 5 −w −k '$DU'**

   Sends a status inquiry (type A job) to the host on session 5. The command waits until the file is sent before returning to the user.

Invoking the Send Command from a C Program

The send command can be started from a C program by using the API function **srje( )** found in the library **/usr/lib/librje.a**. (This library must be linked to the user's object modules. See "The Application Program Interface" above.)

The synopsis of **srje( )** is:

```
int srje (tok_no,ibuf,rtmsgs)
int tok_no;
char *ibuf;
int rtmsgs;
extern int srje_err;
```

where:

tok_no	must be 0.
ibuf	is a string containing all the options and filenames to be processed. In other words, it is the send command line without **rje** −s. The string must be null-terminated.
rtmsgs	is an integer (1 or 2) that is the equivalent of the −**w** option. 1 is the same as not specifying the −**w** option; 2 is the same as specifying it.
srje_err	is an integer error code defined in the API library.

Example

Compile using the command line

cc *example1*.**c** **-lrje**

where *example1* is the name of your **.c** file. When executing, be sure that the **BR** variable is properly set.

```
#include <stdio.h>

main()
{
    extern int srje_err;
    int rc;
    char ibuf[80];

    rc=srje(0,"-h /usr/snaadm/runtime/HOSTA file.a",1); /* LINE EX1 */
    if (rc==-1)
        printf("Send file.a failed, error code %d\n", srje_err);
    else
        printf("Send file.a queued, job number %d\n", rc);

    strcpy(ibuf,"-e file.b -a file.c");
    rc=srje(0,ibuf,2); /* LINE EX2 */
    if (rc==-1)
        printf("Send file.b file.c failed, error code %d\n", srje_err);
    else
        printf("Send file.b file.c succeeded, job number %d\n", rc);
}
```

In the preceding C program, **srje**( ) is invoked twice, once in LINE EX1 and again in LINE EX2. In LINE EX1, **file.a** will be sent to the **dc3770** process using the *rjepipe* named **/usr/snaadm/runtime/HOSTA**, and control will return to the program as soon as the job is queued. In LINE EX2, two files will be sent: **file.b** is already an EBCDIC file, and **file.c** is an ASCII file that will be translated to EBCDIC before being sent. The **−h** *rjepipe* option is not specified because it has been set in the environment variable **BR** and has been exported. Control will return to the program after the files have been sent to the host, or an error has occurred.

srje( ) Return Values

When **srje**( ) is invoked from a C program, the response is indicated by the return value of the **srje**( ) function call and the **srje_err** external variable as follows:

- If the call is successful
  - With **rtmsgs = 1**, the **srje( )** function returns the local job number assigned to the request, and **srje_err** is undefined.
- If the call is unsuccessful
  - **srje( )** returns a value of −1, and **srje_err** contains the error code indicating the cause of the failure.
  - The reason for a routine failure can be determined by declaring the **srje_err** external variable in your program, and then processing the contents of **srje_err** if the **srje( )** function returns a −1.
  - See the table in "Responses to the Send Command" for a list of the possible values for **srje_err**.

Responses to the Send Command

Local Messages

When the send command is invoked from the shell, responses from the SNA/RJE Emulator are displayed on your screen as English language messages. If the send command is invoked from a C program, the **srje( )** return value and the external variable **srje_err** are set to indicate the response. Descriptions of the possible SNA/RJE Emulator messages and the corresponding **srje( )** and **srje_err** values are given below:

1. `RJE: SRJE COMMAND QUEUED as Rnnnn`

   This indicates that the file has been queued internally for transmission, and its local job number is R*nnnn* where *nnnn* are decimal digits. (Note that *nnnn* is also the job number to use if the job must be cancelled.) This message is displayed only if the **−w** option is not specified.

   Correspondingly, with **rtmsg = 1**, the **srje( )** function returns the job number (*nnnn*) and **srje_err** is 0.

2. `RJE: FILE-OBJECT SUCCESSFULLY SENT`

   This indicates that the file has been transmitted to the remote host. This message is displayed only if the **−w** option is specified. (The **−w** option specifies that the command is to wait until the job has been sent before returning.)

Correspondingly, with **rtmsg = 2**, the **srje( )** function returns 0 and **srje_err** is set to the job number, *nnnn*.

3. `RJE: error message`

   This indicates that the file could not be transmitted for the reason given in *error message*. If this happens, the user should expect no further responses.

   Correspondingly, the **srje( )** function returns −1 and **srje_err** is set to indicate the error.

   The following table lists the possible *error messages* and the corresponding **srje_err** error codes for the send command.

Error Code	Error Message
155	INVOCATION ERROR
156	OPTIONS NOT SPECIFIED
157	HOST NOT FOUND
158	CONTROLLER NOT ACTIVE
159	HOST DOES NOT REPLY
160	AN INVALID OPTION IS PRESENT
161	OPTION REPEATED
162	INVALID PARAMETER FOR OPTION
163	REQUIRED OPTION IS MISSING
164	OPTION IS INVALID FOR CURRENT MODE
165	OPTION IS INVALID FOR COMMAND
166	FILE NAMES EXCEED MAXIMUM CHARACTER LENGTH
167	RDR FAILED TO SEND - CANNOT OPEN FILE
168	RDR FAILED TO SEND - CANNOT READ FILE
170	SRJE COMMAND REJECTED - QUEUE OVERFLOW
174	COMMAND FAILED - CHECK LOGFILE/CONSOLE FOR REASON

If you get the message HOST NOT FOUND, error code 157, it could be because an invalid pathname has been specified for the **−h** *rjepipe* command option. (Check the setting of your **BR** environment variable.)

Appendix H in the *AT&T SNA/RJE Emulator+ System Administrator's Guide* contains a complete list of error messages.

Host Messages

When the remote host receives a job, it normally responds with a message conveying the job number that has been assigned to the user's job. This message is displayed at the 3770 console (see the "The Console Operator Command") and written to the log file specified by the System Administrator at system start-up time.

Receiving Job Files

Job output and other files coming in from the remote host are initially written to the default output directory for the **dc3770** process, normally **/usr/snaadm/runtime/output**.

The system uses the following naming convention to generate a unique default name for every file received into the **output** directory:

- 1 character for the medium on which the file was received
  - C - card punch
  - E - diskette
  - P - printer
- 1 digit, 1-6, for the LU number on which the file was received
- a zero
- 1 hexadecimal digit, 0 to F, for the subaddress on which the file was received
- 2 digits, 01 to 12, for the month the file was received
- 2 digits, 01 to 31, for the day of the month the file was received
- a period
- a 4-digit sequence number, 0000 to 9999, which is incremented for each file received. A separate sequence number is kept for each medium

Automatic Routing of Job Output

If the dispatch option is specified by the System Administrator when the emulator is started every file placed in the output directory is examined for a PATH comment card. When a PATH comment card is found the output file is automatically routed to the specified directory or file. The format of the PATH comment card is:

//#(PATH=pathname)*

The output file is moved to the directory or file specified by *pathname*. If *pathname* specifies a directory, the output will be placed in that directory with the system-generated filename. If a filename is specified, the output

will be moved and renamed accordingly. The moved file will have the same owner as the directory it is placed in and will not be writable by 'other.' The following restrictions apply to the specification of *pathname*:

- It must be a full pathname.
- The length cannot exceed 31 characters (including slashes).
- If *pathname* specifies a directory, that directory must be writable by 'other.'
- If the specified file already exists and is writable by 'other' that file will be overwritten.

It is recommended that all output files be routed this way to make user identification of output files easier.

Every file received from the host by the emulator is placed in the output directory. If the dispatcher is active it will check the first 15,000 characters of every file placed in the output directory for a PATH statement. The PATH statement can be placed in any file. If it is placed in the JCL after the job card, all standard printer output associated with that job will be routed to the specified destination. If it is placed in a data file directed to standard punch, that punch file will be moved to the desired directory or file.

The following examples use the standard IBM Utility IEBGENER. (See the IBM Utilities manual for additional details on the use of IEBGENER.) In these examples SYSOUT=A specifies that the output will be sent to a printer and SYSOUT=B specifies that it will be sent to a card punch. Your host may use other classes; check with your System Administrator if you are not sure what the SYSOUT classes are for the host you are using.

In the following example the data in FILE1 is being sent to the printer (SYSOUT=A). Since the JCL and job-related information are also being sent to the printer, all the output from this job will be routed to one virtual printer file. After the file is placed in the output directory the dispatcher will move it to **/usr/*xxx*/jclout**, the file specified in the PATH statement.

```
//.. JOB ...
//*#(PATH=/usr/xxx/jclout)
//STEP    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSUT2   DD SYSOUT=A
//SYSIN    DD DUMMY
//SYSUT1   DD DSN=FILE1,DISP=SHR
//
```

In the following example the data in FILE1 is sent to a card punch (SYSOUT=B) and the JCL and job-related information is sent to the printer. Therefore the output from this job will be sent to two virtual files. The virtual printer file is moved to **/usr/*xxx*/jclout** based on the first PATH statement that is part of the JCL file. The virtual punch file is moved to **/usr/*xxx*/punout** based on the second PATH statement that is part of the punch file received from the host. (See an IBM JCL manual for a description of concatenated DD statements.)

```
//.. JOB ...
//*#(PATH=/usr/xxx/jclout)
//STEP    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSUT2   DD SYSOUT=B
//SYSIN    DD DUMMY
//SYSUT1   DD DATA
//*#(PATH=/usr/xxx/punout)
/*
//         DD DSN=FILE1,DISP=SHR
/
```

Note that the PATH statement will remain in the file **/usr/*xxx*/punout**. The user could delete it with the following statement:

fgrep –v '//*#(PATH=' /usr/*xxx*/punout > /usr/*xxx*/punout2

When routing output files, note that:

- When the dispatcher is running it maintains a log file (**DISPTCH.LOG**) in the RJE home directory. This file contains a record of all automatic routing requests and their status.

- If the PATH statement is used in the JCL, the MSGLEVEL on the JCL JOB statement must be non-zero. When the MSGLEVEL is set to zero the JCL is not printed and the PATH statement would not be part of the file placed in the output directory. The message in the **DISPTCH.LOG** would indicate that automatic routing had not been requested.
- When the PATH statement is part of the JCL some host systems will translate the *pathname* on the PATH statement to upper case. If this occurs the user must either create an upper case directory structure or ask the System Administrator to set the JES2 initialization parameter &PRTRANS to "no" so translation to upper case does not take place.
- Routing of job output is possible only if your System Administrator has started the emulator process with the **–dy** option, which invokes the dispatcher process. See the *System Administrator's Guide* for details.

User Notification of Job Complete

If the dispatch option is specified by the System Administrator when the emulator is started, users can receive a notification, via UNIX mail, of when a job is completed by having a USER field on a comment card in the JCL of the jobfile. For example,

```
//*#(USER=xxx)
```

where xxx is the UNIX logon id of the user who is to receive the notification. If both the PATH and USER options are to be used, they must both be present on the same comment card. For example,

```
//*#(PATH=/usr/sample/output,USER=logon)
```

When routing output files, note that:

- If the USER statement is used in JCL, the MSGLEVEL on the JCL JOB statement must be non-zero. When the MSGLEVEL is set to zero, the JCL is not printed and the USER statement would not be part of the file placed in the output directory.
- When the USER statement is part of the JCL, some host systems will translate the logon name on the USER statement to upper case. In this situation the user must either have an upper case logon name or have the System Administrator arrange to have the JES2 initialization

parameter PRTRANS set to "no" so translation to upper case does not take place.

- User notification of job completion is possible only if your System Administrator has started the emulator process with the **–dy** option, which invokes the dispatcher process. (See the *System Administrator's Guide* for details.)

Tag Files

When your System Administrator starts the SNA/RJE Emulator with the *-tg* option, a tag file (identified by the .TAG suffix) will be created for every file received. The tag file contains the following information about its associated file:

```
TIME RECEIVED:      MM DD YY
RECEIVED ON:        PRT, PUN or EXCH
TYPE:               TRANSPARENT or NON-TRANSPARENT
```

The tag file is moved into the appropriate destination directory if automatic routing is in effect, and an automatic routing jobcard was specified in the original jobfile sent to the host.

Cancelling Jobs

You can cancel a job queued locally for transmission by executing the cancel command, **rje −c**, from the shell, or by calling the **crje( )** function in a C program. Unless you have logged in as root, you may only cancel jobs you have sent.

A job can be cancelled only if it is waiting on the queue. If it is currently being transmitted, or if it has already been sent, the cancel command will fail and an appropriate message will be returned.

Invoking the Cancel Command from the Shell

The format to execute the cancel command from the shell is:

rje −c [**−h** *rjepipe*] [**−M** *msg_file*] *jobno*

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings. (See "The **rje** Command" above.) *jobno* specifies the name of the internally queued file. This name is returned to the user by the send command (see "Responses to the Send Command"). The *jobno* consists of four digits, and it may be preceded by R or J.

Examples

1. **rje −c R1234**

   This cancels job R1234.

2. **rje −c 1234**

   This also cancels job 1234. The leading R in the job name is not required.

Invoking the Cancel Command from a C Program

The cancel command can be executed from a C program by using the API function **crje( )** found in the library **/usr/lib/librje.a**. (This library must be linked to the user's object modules. See "The Application Program Interface" above.)

The synopsis of **crje( )** is:

```
int crje (host, jobno)
char *host;
int jobno;
extern int srje_err;
```

where:

host	is a character pointer to a buffer containing the HOST name.
jobno	is an integer that was returned from the **srje( )** function.
srje_err	is an integer error code defined in the API library.

Example

Compile using the command line

cc *example2*.**c -lrje**

where *example2* is the name of your **.c** file. When executing, be sure that the **BR** variable is properly set.

```
#include <stdio.h>

main()
{
    extern int srje_err;
    int jobno, rc;
    char *hostname="/usr/snaadm/runtime/HOSTA";

    jobno=srje(0,"file.a",1);
    if (jobno==-1)
        printf("Send file.a failed, error code %d\n", srje_err);
    else
    {
        printf("Send file.a succeeded, job number %d\n", jobno);
        rc=crje(hostname, jobno); /* LINE EX1 */
        if (rc==-1)
            printf("Cancel file.a failed, error code %d\n", srje_err);
        else
            printf("Cancel file.a succeeded.\n");
    }

    jobno=srje(0,"file.b",2); /* LINE EX2 */
    if (jobno==-1)
        printf("Send file.b failed, error code %d\n", srje_err);
    else
    {
        rc=crje(hostname, jobno); /* LINE EX3 */
        if (rc==0)
            printf("Crje should have failed\n");
        else
            printf("Cancel file.b failed as expected\n");
    }
}
```

This example demonstrates the connection between **srje( )** and **crje( )**. On successful completion, **srje( )** returns the job number, which can then be used as an argument to **crje( )**, provided the file is not sent immediately. This use of the job number is illustrated in LINE EX1. The invocation of **srje( )** in LINE EX2 specifies that it should wait for two return messages. This ensures that **srje( )** will return after the file has been sent. Therefore, the

invocation of **crje( )** in `LINE EX3` will fail, because the job will not be on the queue when **crje( )** is invoked.

crje( ) Return Values

When **crje( )** is invoked from a C program, the response is indicated by the return value of the **crje( )** function call, and the external variable **srje_err** as follows:

- When the call is successful
  - **crje( )** returns 0, and **srje_err** is undefined.
- When the call is unsuccessful
  - **crje( )** returns −1 or −2, and **srje_err** contains the error code indicating the cause of the failure.
  - To determine the reason for **crje( )** failing, declare the **srje_err** external variable in your program, and, if **crje( )** returns −1, process the contents of **srje_err**.
  - See the table in "Responses to the Cancel Command," below, for a list of error messages and corresponding error codes.

Responses to the Cancel Command

Local Messages

When the cancel command is invoked from the shell, responses from the SNA/RJE Emulator are displayed on the user's terminal. If it is invoked from a C program, the response is indicated by the return value of the **crje( )** function and the **srje_err** external variable. (Note that **srje_err** is used for both the **srje( )** and **crje( )** API functions.)

Descriptions of the possible SNA/RJE Emulator messages and the corresponding **srje_err** error codes are given below:

1. `RJE: SUCCESSFULLY CANCELLED Rnnnn`

   This indicates that the job has been successfully cancelled from the queue, and will not be sent to the host. Correspondingly, the **crje( )** function returns 0.

2. `RJE: error message`

   This indicates that the cancel command could not be completed for the reason given in *error message*. Correspondingly, the **crje( )** function returns −1, and **srje_err** is set to indicate the error. The following is a list of possible *error messages* and the corresponding **srje_err** error codes:

Error Code	Error Message
157	HOST NOT FOUND
169	CRJE COMMAND REJECTED - FILE NOT QUEUED

If you get the message HOST NOT FOUND, error code 157, it could be because an invalid pathname has been specified for the **−h** *rjepipe* command option. (Check the setting of your **BR** environment variable.)

NOTE

Appendix H in the *System Administrator's Guide* contains a complete list of error messages.

Displaying the Local Job Queue

To see if a job is on the local job queue waiting to be sent to the host, enter:

rje **−d** [**−h** *rjepipe*] [**−M** *msg_file*] **−j**

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings. (See "The **rje** Command" above.)

This command will display information about all the jobs on the queue. See Appendix H, messages 67 through 75, for information on the messages displayed by **rje −d**. For information about cancelling jobs on the queue, see "Cancelling Jobs" above.

Displaying the Log File

The find command, **rje -f**, is used to view the SNA/RJE system log file. (The log file must be specified by the System Administrator at system start-up time.) When the find command is executed, the user is prompted to enter a date and time. The log file is then searched for the first line whose time stamp equals or is less than the date and time entered by the user. If the time specified by the user is greater than any time found within the log file, the last line of the log file will be selected. Once the line is found, the log file is displayed a page at a time (using the UNIX system **pg** command), starting at the selected line.

The format of the find command is:

rje **−f** [**−h** *rjepipe*] [**−M** *msg_file*]

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings. (See "The **rje** Command" above.)

The system will then prompt you to enter a date and time as follows:

```
enter date/time (mm/dd/yy hh:mm[:ss])
```

`mm` `dd`, and `hh` do not require leading zeros; `mm` and `ss` do. The only optional field is `ss`. Self-explanatory error messages will be displayed if the date or time is in an incorrect format, or if the log file is not found.

The Console Operator Command

The console operator command, **rje −o**, is used to attach or detach the terminal that the command is issued from as the 3770 system console. (There can be only one 3770 system console.)

In addition to local SNA/RJE Emulator messages, all console operator messages sent from the remote host will be displayed at the terminal designated as the console. Console operator messages are also written to the system log file specified by the System Administrator at start-up time.

The format of the console operator command is:

rje −o [**−h** *rjepipe*] [**−M** *msg_file*] [**−ty**] [**−st**]

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings. (See "The **rje** Command" above.) The other options are:

−ty specifies that the console should be attached to the terminal from which the command is issued. This is the default.

−st specifies that the console should be detached from the terminal from which the command is issued.

Examples

1. To designate your terminal as the 3770 console, enter

   rje −o or **rje −o −ty**

2. To detach the terminal as the system console, enter

   rje −o −st

Table of Contents

System Set-up 1
Overview 1
Configuring the System Environment for SNA 2
- Invoking **snaenvset** 2
- Setting the User's PATH Variable 3
- Changing Environment Variable Defaults 3

Creating an SNA Controller Configuration File 4
- SNA Configuration Options 5
- Support for an 8100 Controller 6
- Creating a Configuration Source File 12
- Creating a Configuration Object File 14

Running the AT&T SNA/RJE Emulator+ 17
Starting the SNA Controller 17
- Error Messages 18

Starting the SNA/RJE Emulator 19
Running Multiple SNA/RJE Emulators 22
Assigning Output Destinations 24
Logging Sessions On and Off the Host 27
Local Status Command 29
Local 3770 Console Command 30
System Shutdown 32
- Stopping the **dc3770** Process 33
- **dc3770** Directory Cleanup 33
- Stopping the SNA Controller 34

Appendix A: Environment Variables A-1

Appendix B: Sample Configuration Source Files B-1

Appendix C: Sample Host Configurations C-1
Configuration – SNAJ1 C-1
Configuration – SNAJ2 C-3

Appendix D: NCP Gens D-1
MVS/JES NCP Gen D-1
DOS/VS POWER NCP Gen D-4

Appendix E: Error Messages E-1

Appendix F: Program Check Error Codes F-1

Appendix G: Communication Check Error Codes G-1

Appendix H: DC3770 and RJE Messages H-1

Appendix I: iscconfig Command Options I-1

List of Figures

Figure 1: SNA Configuration Source File Options 8

Figure 2: Arguments to **snopts** 14

Figure 3: Sample **snopts** Configuration Screen 16

Figure 4: SNA Process Initiation Error Messages 19

System Set-up

Overview

Before running the AT&T SNA/RJE Emulator+ software, the System Administrator must configure the system environment and create a controller configuration file that conforms to local and host requirements.

- Configuring the System Environment

  The AT&T SNA/RJE Emulator+ uses environment variables as pre-defined defaults for command options. This means that when an option to an Emulator+ command is omitted, the list of environment variables is searched for a specific variable associated with that option. If that variable is set, its value is used as the option to the command. Therefore, if the appropriate environment variables are set, the user need not specify most options when issuing a command, unless he chooses to override the defaults.

  Configuring the environment involves setting the environment variables used by the Emulator+. For convenience, two shell scripts, **snaenvset** and **snaenvcust**, are provided to set these variables to default values.

- Creating a Controller Configuration File

  The controller configuration file contains the parameters the SNA Controller processes need to communicate properly with a particular host. When the controller is started, the configuration file is used as input to customize the SNA controller processes with the options specified in the file.

  Alternate configuration files can be created for different remote host and local configurations.

Configuring the System Environment for SNA

The AT&T SNA/RJE Emulator+ processes refer to various environment variables during execution. These variables are used to provide information to the processes, such as default values for command options, and specific system start-up information.

The SNA/RJE system environment must be configured:

- Before you begin to use the SNA/RJE software for the first time.
- Before starting an SNA Controller.

Two shell scripts, **snaenvset** and **snaenvcust**, both in the **/usr/snaadm/runtime** directory, are used in configuring the system environment for SNA operation. Appendix A lists and describes the environment variables set by these scripts. It also provides the default values assigned to the variables by the scripts.

Invoking snaenvset

Before configuring the environment, the System Administrator should check that the default values set by the scripts are appropriate. If any of these values must be changed, refer to "Changing Environment Variable Defaults," below.

If the default values set by the scripts are appropriate, the environment can be configured for SNA operation by executing **snaenvset** as follows:

Step 1. Log in as root or super-user.

Step 2. Change directory to **/usr/snaadm/runtime**.

Step 3. Enter

```
. ./snaenvset
```

Step 4. The UNIX system commands **env**(1) and **set** (see **sh**(1)) can be used to check your environment. See the *AT&T UNIX System V User's Reference Manual* for information about these commands.

The **snaenvset** script needs to be executed only once per login. To have **snaenvset** execute automatically when you log in, add the following entry to your **.profile** file:

. **/usr/snaadm/runtime/snaenvset**

Setting the User's PATH Variable

The path name of the **dc3770** process' runtime directory must be added to the user's **PATH** variable before the **/usr/bin** directory to ensure that the **rje** command associated with the SNA/RJE Emulator+ is picked up before any other **rje** commands. For example:

PATH=/usr/snaadm/runtime:/usr/bin

Changing Environment Variable Defaults

The System Administrator may need to change the values of certain environment variables because of local configuration considerations. Before doing so, he should be aware of the following:

PCD specifies the communications port number that the SNA Controller is to use. You would want to change this variable if you were going to use a port number different from that provided by the **snaenvset** script.

SNAHOST is used to change the **CONFIG** variable, which specifies the full pathname of the controller configuration object file to be used when the SNA Controller is started. Configuration object files are named with the prefix **CSNA.**, for example, **CSNA.hostx**. **SNAHOST** is used to specify which of the **CSNA.** files is to be used. For example, if **CSNA.hostx** (located in **/usr/snaadm/runtime**) is to be used, **SNAHOST** should be set equal to **hostx**. When the **snaenvcust** script is executed, **CONFIG** will then be set appropriately to **/usr/snaadm/runtime/CSNA.hostx**.

To change the value of an environment variable temporarily:

Step 1. Execute **snaenvset**, as described above in "Invoking **snaenvset**".

Step 2. Reset the environment variable(s) that need to be changed. For example, to specify that the controller should use the configuration object file named **CSNA.hostb**, you would enter:

SNAHOST=hostb

Step 3. You must then execute the **snaenvcust** script by entering:

../snaenvcust

The **snaenvcust** shell script will re-customize the system environment using the new values.

To change the value of an environment variable permanently:

Step 1. Use a text editor such as **vi** to edit the **snaenvset** shell script, changing the setting of the environment variables(s) as required.

Step 2. Execute **snaenvset** as described above in "Invoking **snaenvset**".

Creating an SNA Controller Configuration File

A separate configuration file must be created for each of the different host and/or local configurations an SNA Controller may use.

Three steps are involved in creating an SNA Controller configuration file. The System Administrator must:

Step 1. Determine the configuration options needed to customize the SNA Controller for your particular site and host.

Step 2. In the **/usr/snaadm/cust/snaoptions** directory, create an SNA configuration source file in which each option is defined, as required.

Step 3. Create an SNA configuration object file. This involves processing the configuration source file through the **snopts** utility. When the **snopts** utility is executed from the **/usr/snaadm/runtime** directory, it creates an object file from your source file and displays the completed configuration.

SNA Configuration Options

To determine the options required for the SNA Controller configuration file, consult the Network Administrator at your host site for precise information regarding your configuration. The options you define in the configuration source file should coincide with your host configuration.

Configuration options for the SNA Controller are categorized as follows:

Line Options:

These options are used to ensure compatibility between the physical communication media on both sides of the link. Communication line characteristics such as full-duplex or half-duplex operation and NRZ or NRZI operation are defined by these options.

Controller Options:

These options are used to define the controller (PU) type and its associated devices (LU's). There must be agreement between the host and the controller as to the identity and nature of all the devices (terminals and printers) associated with the controller.

Dual-identity SDLC:

The AT&T SNA/RJE Emulator+ software is designed to run concurrently with the AT&T 3270 Emulator+ (Release 2.0.0 and later releases). Assuming the AT&T 3270 Emulator+ has been installed on your 3B Computer, the dual–identity SDLC feature of the SNA Controller (which is a component of both software packages) allows a 3770 SNA/RJE workstation and a 3270 workstation to operate simultaneously using one SDLC link to the host.

To implement dual–identity SDLC, the SNA Controller is configured with one 3770 type PU, and one 3270 type PU. Two separate sets of controller options are defined in the configuration file, one set for each PU (controller) type. However, only one set of line options is defined – both PU's use the same SDLC link. In addition, the line must be system-defined at the host as a multi-drop line with a station address for each PU.

For information on 3270 operation, and on configuring the controller options for a 3270 controller (PU) type, refer to the *AT&T 3270 Emulator+ System Administrator's Guide.*

NOTE

The AT&T SNA/3270 Emulator+ and the AT&T SNA/RJE software can also be run independently, using separate SDLC links.)

Support for an 8100 Controller

To emulate an 8100 controller, the dual-identity feature of the SDLC must be used along with the following configuration information:

- Line Options

  fdx
  pt-to-pt
  nrz

- Controller Options

  controller 8100
  xid 020000600000
  seg 265
  address c3

  lu2 3
  lu3 3
  lu4 3
  lu5 3
  lu6 3
  lu7 1
  lu8 1
  lu9 2
  lu10 2
  lu11 2
  lu12 2
  lu13 2
  lu14 2
  lu15 2
  lu16 2

Figure 1 lists the valid options and corresponding defaults that can be defined in an SNA configuration source file.

Option	Function	Default
Line Options:		
fdx/hdx	Sets full-duplex or half-duplex mode.	**hdx**
multipoint/pt-to-pt	Sets multi-point or point-to-point operation.	**pt-to-pt**
nrzi/nrz	Sets NRZI mode of transmission.	**nrz**
Controller Options:		
controller *aaaa*	Sets controller (PU) type to *aaaa*.	
lu*n t*	Assigns the LU numbered *n* where *n* is a number from 1 to 6 and *t* specifies the device type, which must be: 3 – 3770 RJE Device	LU is not defined.
xid *xxxxx*	Sets last 5 digits of the XID to *xxxxx* (hex).	**xid** 00000
address *xx*	Sets SDLC station address to *xx* (hex).	**address** 01
seg *n*	Sets SNA segment size to *n* (decimal).	**seg** 265
broadcast	Designates this station as the one to respond to broadcast messages from the host.	Not a broadcast station.

Figure 1: SNA Configuration Source File Options

The following is a functional description of each of the options that may be defined in the configuration source file.

Line Options

fdx/hdx Full-duplex (four-wire) operation allows two stations to transmit and receive at the same time. In half-duplex (two-wire) operation, the stations take turns transmitting, using pin-level (RTS, CTS) controls. (Dial-up lines are half-duplex.) SDLC is strictly a half-duplex protocol, so full-duplex traffic cannot occur even with four wires. However, when full-duplex is specified in a point-to-point configuration, even though true four-wire traffic is not possible, modem line turnaround delays are eliminated because RTS is always kept high. The default is **hdx** (half-duplex).

multipoint/pt-to-pt A multipoint configuration consists of two or more secondary stations communicating with a single primary station (the host). The SDLC station address is checked to determine if the message is for an enabled station. A point-to-point configuration consists of one secondary station communicating with one primary station. This line option must be defined as **multipoint** if the dual-identity SDLC feature is to be implemented. (Dual-identity SDLC allows two controller types to operate concurrently using one SDLC link. See the **controller** option below.) The default is **pt-to-pt** (point-to-point operation).

nrzi/nrz Non-Return To Zero Inverted (NRZI) transmission is used to reduce the probability of losing bit synchronization. Because bit synchronization is maintained by the pace of the bit stream itself, periodic signal polarity transitions are necessary. In NRZI transmission a polarity change occurs whenever a binary zero appears in the message. Since "bit stuffing" ensures that there will never be more than six contiguous bits with values of one, there must be a zero, and hence a polarity change, at least every six bits. Because it may be prohibited

by specific hardware limitations on some data communications equipment, NRZI transmission is optional. The default is **nrz** (NRZ transmission).

Only one set of line options should be defined in each source file. With dual–identity SDLC, the SNA Controller is configured with two controller types, but only one set of line options is used. (Both controller types will use the same SDLC link.)

Controller Options

controller *aaaa*

The controller type (specified by *aaaa*) denotes the physical unit (PU) type of the SNA product being emulated. Certain aspects of the controller's operation, and the types of devices that can be attached to the controller, are determined by the controller type. Up to two controller types can be defined in the source file.

For SNA/RJE emulation, controller type 3770 must be specified.

If a controller type is not specified, **snopts** will check the LU device type definitions. (See the **lu** option, below.) If an LU device type 3 is defined, 3770 is assumed as the controller type. An error message is given if there are any errors in the LU definitions, such as an illegal device type.

Dual-identity SDLC. When implementing dual-identity SDLC, each defined controller type requires its own **lu, xid, address,** and **seg** definitions. Two 3770 controller types may be defined, or one 3770 and one 3270 controller type may be defined. When a 3770 is defined for dual-identity, the 3770 controller definitions must occur before the 3270 controller definitions. Refer to the *AT&T 3270 Emulator+ System Administrator's Guide* for procedures for defining 3270 controller options. Appendix B contains sample dual-identity SDLC configuration source files.

lu*n t*

A maximum of six LU's may be defined for a 3770 controller type. Accordingly, valid values for *n* are 1 - 6.

A device type *t* must be assigned to each logical unit that is defined in the source file. Only device type 3 is valid for controller type 3770.

NOTE

Device types 1 and 2 are used only to define LU's for a 3270 controller type. Refer to the *AT&T 3270 Emulator+ System Administrator's Guide* for information on defining a 3270 controller.

xid *xxxxx*

The SDLC XID (eXchange station IDentification) is used mainly in dial-up configurations to identify the calling station to the host. When the controller receives the XID command, it must identify itself to the primary SDLC station using a 6-byte (12 hex digits) sequence number. The first 7 digits are coded automatically, and are dependent on the controller type specified. For controller type 3770, the first 7 digits will automatically default to 0200013.

The last 5 digits must be specified using the **xid** *xxxxx* option. This 5-digit number is an identification number that must be obtained from the Network Administrator at the host site.

The default value is 00000.

address *xx*

This option specifies the SDLC station address. Each SDLC secondary station is distinguished by a 1-byte (2 hex digits) station address. This is the first byte after the flag in an SDLC frame.

The default value is 01.

seg *n*

Segment size refers to the maximum size of the Path Information Unit (TH + RH + RU) sent from or to the controller.

The default value is 265.

broadcast — In the SDLC protocol, all stations will accept a frame with the broadcast address. In a multi-station configuration, or with dual-identity SDLC, the user can designate which station will reply to the broadcast message by specifying the broadcast option. If the option is not specified, the station will not be a broadcast station.

Creating a Configuration Source File

The configuration source file is a standard text file that can be created in the **/usr/snaadm/cust/snaoptions** directory using any text editor such as **vi**. Note the following characteristics of the source file:

- Configuration options may be specified in any order, one per line.
- The option must be the first item on the line.
- If the first character on a line is the pound sign (#), the line is treated as a comment and ignored.
- Extra text following each option specification is ignored.
- When an option is not specified in the source file, its default value will be used.
- Invalid options are ignored. (Error messages are not always given.) For example, if an option is misspelled or is not the first item on the line, it is ignored.

The following is an example of an SNA configuration source file, illustrating the characteristics noted above.

```
#This is a sample configuration source file defining one 3770
#physical unit.  Note the free format of this file.

controller 3770              Specifies the controller type.

        xid    abcd          Sets XID to abcd.
seg 265  Maximum segment size for this type of
#  SNA Controller.
broadcast                    Will respond to broadcast messages.
#The following LU's are configured:
lu1  3                       For controller type 3770, SNA/RJE operation,
                             LU device type must be set to 3 (RJE device).
lu03  3
lu4  3
lu5  3
        fdx                  This sets SDLC to full-duplex mode.

#The next three options are invalid, and therefore will be ignored
#by snopts.  Note that if a default is available for the option,
#it will be taken.
address ty                   ty is not a valid hex number.
multipnt                     It must be spelled "multipoint" to be
#                            valid, therefore this option will default
#                            to point-to-point operation.
-nrzi                        The option must be the first item on the
                             line.
```

A sample configuration source file is provided in the **/usr/snaadm/cust/snaoptions** directory. Appendix B contains additional examples of configuration source files.

Creating a Configuration Object File

To create an SNA configuration object file, in the **/usr/snaadm/runtime** directory invoke **snopts** using the command format:

snopts **−w** [*object_file*] [< *source_file*]

Figure 2 lists the arguments to **snopts.**

Option	Description	Default Environment Variable	Default Value
object_file	Names the output object file. Must be prefixed by CSNA.	**CONFIG**	**config**
source_file	Names the full pathname of the configuration source file.	—	interactive mode

Figure 2: Arguments to **snopts**

If the *object_file* is specified on the command line, the file name must contain the prefix **CSNA.**, for example, **CSNA.hostx.**

If *object_file* is not specified, the environment list will be searched for the variable **CONFIG.** If **CONFIG** has been set, its value will be used as the name of the output object file. If **CONFIG** has not been set, then **snopts** uses the default name **config** for the object file.

For *source_file,* the full pathname of the configuration source file must be specified, for example, **/usr/snaadm/cust/snaoptions/cSNA.hostb.**

If < *source_file* is not specified on the command line, **snopts** executes interactively. (See "Using **snopts** Interactively," below.)

The **snopts** utility creates the configuration object file from the configuration source file and displays a screen showing the configuration as **snopts** processed it.

NOTE

You should verify that all the options specified in the source file match the displayed configuration. If an option is not displayed or is incorrectly set, check your configuration source file for invalid options.

Example

snopts −w < /usr/snaadm/cust/snaoptions/config1

The configuration source file **config1** will be used by **snopts** to create a configuration object file. The name of the object file will default to the value of the **CONFIG** environment variable, if it has been set. If **CONFIG** has not been set, the object file will be named **config** and placed in the **runtime** directory.

Using snopts Interactively

The **snopts** utility can also be used interactively to create a configuration object file. When **snopts** is used in this way, a configuration source file is not specified; instead, **snopts** accepts input from your terminal as the source to create the object file. To use **snopts** interactively enter:

snopts −w [*object_file*]

The **snopts** utility will accept whatever you enter from your terminal as the configuration source options to be processed. You simply enter the configuration options, as required. When all the options have been entered, type the **ctrl** and **d** keys simultaneously to end the session. **snopts** creates the SNA configuration object file and displays the completed configuration. Once again, you should check this display to make sure all the required options have been properly defined.

Displaying an Existing Object File

To display an existing configuration object file, execute **snopts** without the **−w** flag.

snopts [*object_file*]

As in the previous example, if *object_file* is not specified on the command line, **snopts** will use the object file named by the **CONFIG** variable if that variable is set. If **CONFIG** is not set, **snopts** will create a file in the **runtime** directory with the default name **config**.

Figure 3 shows a sample **snopts** configuration screen.

```
SNA/SDLC CONFIGURATION

half duplex   nrz   point-to-point

Controller 3770

lu1  3
lu3  3
lu4  3
lu5  3

station address=03
xid=0020013abcd
segment size = 265
responds to broadcast message
```

Figure 3: Sample **snopts** Configuration Screen

Running the AT&T SNA/RJE Emulator+

After the system environment has been configured, and a Controller configuration file created for your particular site and host, the AT&T SNA/RJE Emulator+ system can be started.

The System Administrator may designate one terminal as the master terminal for the AT&T SNA/RJE Emulator+ system. The shell that runs on that terminal must have root privileges to start the SNA Controller and SNA/RJE Emulator processes. While these processes are being started, all status and error messages will be displayed on this terminal.

To run the AT&T SNA/RJE Emulator+, the System Administrator must:

Step 1. Start the SNA Controller process, **SNA**, and the **sdlc** program by executing the **startsna** shell script.

Step 2. Start the SNA RJE Emulator process, **dc3770**, by executing the **startdc** command. The **startdc** command will automatically issue an **rje −a −b** command to designate batch routing.

Step 3. Sign on to the host using the **rje −s −son** command.

Starting the SNA Controller

The SNA Controller is started by the shell script **startsna**, which is in the **/usr/snaadm/runtime** directory. The **startsna** script checks to see that no other controller processes are running on the communications port to be used by this SNA Controller. (The communications port number is obtained from the **PCD** environment variable.) **startsna** then executes the commands to start the **SNA** process and the **sdlc** program.

At the start of execution, the **SNA** process reads the object file containing the configuration options for the controller. (The name of the object file is obtained from the **CONFIG** environment variable.) Configuration information is passed to the **sdlc** program, which is downloaded and started on the communications board. Note that because **sdlc** is running on a separate processor, if you execute the UNIX system **ps** command, you will not see **sdlc** running in the UNIX operating system. Next, the **SNA** process enables and establishes communication paths to and from **sdlc** and the devices. As soon as **sdlc** and **SNA** are running, the SNA Controller is ready to communicate with the host. It is not necessary to have the SNA/RJE process, **dc3770**, active for there to be communication with the host.

To start the SNA Controller process **SNA** and the **sdlc** program, the System Administrator must:

1. Log in as root.
2. Change directory to **/usr/snaadm/runtime**.
3. Set the environment. (See "Configuring the System Environment for SNA" above.)
4. Execute the shell script **startsna**.
5. Wait for the copyright/release number message and controller ready message that are displayed when the SNA Controller has been started successfully.
6. Start the SNA/RJE Emulator.

NOTE

If connection to the host is via a dial-up line, you should dial in at this point.

Error Messages

The table in Figure 4 contains a list of error messages displayed if the **SNA** process fails during initialization. Failures may result from incorrect or incomplete set-up procedures. When these error messages are displayed, the **SNA** process is aborted; it must be restarted after you correct the problem. The error messages are listed in the order that the **SNA** process generates them. Some error messages are followed by a number. This number is the **errno** value returned by the failed system call, and may provide more specific information about the cause of the error. For information about **errno**, consult **intro**(2) in the *AT&T UNIX System V Programmer's Reference Manual.*

Error Message	Explanation/Possible Cause
sdlc open error	SDLC has not been started or device driver pathname for SDLC is incorrect.
name too long	Pipe prefix specification is too long (must be less than 100 characters).
path from devices open error	Inbound SNA-to-devices pipe does not exist.
configuration file not found	Named configuration object file does not exist.
configuration file open error	Permission bits incorrect for configuration file (should be read/write/not executable).
configuration file read error	Permission bits incorrect for configuration file.
error in configuration file	Configuration file is not the result of a successful execution of **snopts**.
too many lu's configured	More than 6 LU's are defined.

Figure 4: SNA Process Initiation Error Messages

Starting the SNA/RJE Emulator

Once the SNA Controller is running, the System Administrator can start the SNA/RJE Emulator process, **dc3770**, by executing the **startdc** command as follows:

Step 1. Log in as root

Step 2. Change directory to **/usr/snaadm/runtime**

Step 3. Check that the system environment has been properly set. (See "Configuring the System Environment for SNA," above.)

Step 4. Execute **startdc** using the following command format:

startdc [**−d** *directory*] [**−p** *snapipe*] [**−h** *rjepipe*] [**−M** *msg_file*] [**−P** *pu_num*] [**−D**] [**−X** *code_file*] [**−u**] [**−U** *unch*] [**−l** *logfile*] [**−dy**] [**−tg**]

The following list describes the options in detail:

−d *directory* commands **dc3770** to change its current working directory to the directory specified by *directory* before doing anything else. (If necessary, **dc3770** will create the directory.) This option is provided so that more than one **dc3770** process can run on the same system. See "Running Multiple SNA/RJE Emulators", below. The default is **/usr/snaadm/runtime**.

−p *snapipe* *snapipe* specifies the pipe to be used between the **dc3770** process and the appropriate SNA Controller. (This must be the *snapipe* name designated when the SNA Controller was started.) The default environment variable for this option is **P3274**, and the default value is **/tmp/P3274.$PCD**.

−h *rjepipe* *rjepipe* specifies the pipe to be used between the user interface process, **rje**, and the **dc3770** process. The default environment variable for this option is **BR**, and the default value is HOSTA.

−M *msgfile* *msgfile* specifies the message file to be used by the **dc3770** process. The default value is **dc3770.msg**.

−P *pu_num* *pu_num* specifies which 3770 controller (physical unit) the **dc3770** process is to use when the SNA Controller has been configured as two 3770 workstations. A value of 1 specifies that this **dc3770** process use the first PU defined in the controller configuration; a value of 2 specifies that it use the second PU. The default is 1, the first PU. Note that this option is used only when dual-identity SDLC is being used.

−D specifies that extra trace output be placed in the **dc3770** log file.

−X *code_file* is the name of the code set translation file used to override the default EBCDIC:ASCII mapping. It contains 256 bytes used for EBCDIC to ASCII translation, followed by 128 bytes used for ASCII to EBCDIC translation.

−u modifies the behavior of **dc3770** with respect to output of unprintable characters. If **−u** is specified, all characters will be output as indicated in the translation table. If it is not specified (the default), then for output files with transparency level 0 all unprintable characters will be replaced by the "unprintable character" character. (See the **−U** *unch* option below.)

−U *unch* specifies that the character *unch* be used for all "unprintable characters". If not specified, the default character ? is used.

−l *logfile* specifies the pathname to receive all system console output. All console messages received during the emulation session will be written to this read-only disk file. This file actually records most of the activity of the session. If not specified, there will be no log file.

−dy invokes the dispatcher process, **disptch**, which implements the automatic routing feature of the emulator.

−tg will cause .TAG files to be created for every file sent by the host. Associated with every file is a TAG file (identified by the .TAG suffix) that describes the "reception" of that particular file. The TAG file contains the following information:

```
TIME RECEIVED:    MM/DD/YY
RECEIVED ON:      PRT/PUN/EXCH
TYPE:             TRANSPARENT/NON-TRANSPARENT
```

TAG files are created only when the **−tg** option is specified.

NOTE

Standard UNIX system file names are 14 characters. Therefore if the output filename is more than 10 characters, the .TAG suffix will be truncated accordingly.

Examples

1. **startdc**

   This command starts a **dc3770** process with all options defaulted. **dc3770** is started using the values from the environment variables or hard-coded values for all options. The **disptch** process will not be started; there will not be a system log file; and the .TAG files will not be created.

2. **startdc –l logfile –dy –tg**

   This command starts a **dc3770** process, names the system log file **logfile**, invokes the **disptch** process so that automatic routing of output will occur, and causes .TAG files to be created.

When the **dc3770** process is started, the following subdirectories and files are created:

output	the directory for output files received from the host
jobs	the local job queue directory
tmp	the directory used by the **rje** process for temporary files
logfile	the RJE system message log file (includes 3770 console messages)
DISPTCH.LOG	the log file for the dispatch process
assignments	the binary file that retains the LU output assignments

The System Administrator should inform each user of the *rjepipe* name for the **dc3770** process he will be using. Each user should set the **BR** environment variable to the correct *rjepipe* name. See "Configuring Your Environment" in the *AT&T SNA/RJE Emulator+ User's Guide.*

Running Multiple SNA/RJE Emulators

There is a one-to-one relationship between a physical unit (controller type) configured in the SNA Controller and an SNA/RJE Emulator **dc3770** process. Therefore, a separate **dc3770** process can be started for each 3770 controller type that is configured in an SNA Controller. For example, if an SNA Controller has been configured with two 3770 controller types

(implementing dual-identity SDLC), then two **dc3770** processes can be started. Or, if two SNA Controllers are running (using separate SDLC links) and each has been configured as a 3770 controller type, then two **dc3770** processes can be started, one using each SNA Controller.

Each **dc3770** process must execute in its own separate directory structure. If the first **dc3770** process is running in **/usr/snaadm/runtime**, a directory other than this must be specified using the **−d** *directory* option when starting the second **dc3770** process. However, the **dc3770** command should always be executed from the **/usr/snaadm/runtime** directory. Each **dc3770** process must also have a unique *rjepipe* name which is specified using the **−h** *rjepipe* option.

To start a second **dc3770** process:

Step 1. Log in as root

Step 2. Change directory to **/usr/snaadm/runtime.**

Step 3. Check that the system environment is properly set. (See "Configuring the System Environment for SNA".)

Step 4. Enter the **dc3770** command including:

- the **−d** *directory* option to specify the new **runtime** directory for the **dc3770** process to execute in
- the **−h** *rjepipe* option to specify the *rjepipe* to be used for this **dc3770** process
- the **−P** *pu_num* option, if the SNA Controller that the **dc3770** process is going to use is configured as two 3770 workstations (dual-identity SDLC)
- the **−p** *snapipe* option, if a different SNA Controller is going to be used.

Example

dc3770 −d /usr/snaadm/run2 −h HOSTB −P 2

This starts a **dc3770** process in the **/usr/snaadm/run2** directory, using the second PU defined in the controller configuration. The full pathname of the *rjepipe* for this **dc3770** process is **/usr/snaadm/run2/HOSTB.**

NOTE

The System Administrator should inform each user of the *rjepipe* name for the **dc3770** process he will be using. Each user should set his **BR** environment variable to the correct *rjepipe* name. See "Configuring Your Environment" in the *AT&T SNA/RJE Emulator+ User's Guide.*

Assigning Output Destinations

By default, **dc3770** starts up with all possible logical destinations assigned to batch mode operation. The **rje −a** command assigns the output devices (virtual or real) to receive data from the host and permits you to change the default. This is a privileged command; you must have superuser privileges to use it.

The command format is as follows:

rje −a −b [**−h** *rjepipe*] [**−M** *msg_file*] [*option...*]

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings.

The following list describes the options for the **rje −a** command:

−t *session* *session* specifies the session the command affects. Its value must be in the range 0-6.

Value	Meaning
0	Command applies to all sessions that are not busy.
1-6	Specific session to which the command applies.

This option defaults to **−t** 0.

−m *medium* *medium* specifies the medium of the virtual destination device (as viewed by the host).

Value	Meaning
C	Card Image

E Exchange Diskette
P Printer

This option defaults to all.

−s *subaddr* *subaddr* specifies the destination subaddress (as known to the host), where *subaddr* is a value in the range 0-F (hexadecimal).

This option defaults to all.

−e The received output file will be left in EBCDIC. Code set translation to ASCII will not take place. The IRS, new-line, or linefeed character is used as the delimiter.

−b specifies batch routing.

−x *level* specifies the transparency level, where *level* can be

0 − same as **−a** option; normal processing (default)
1 − same as **−e** option; no code translation
2 − SCS data stream not expanded and removed; no code translation
3 − the SCS data stream codes are not expanded; no code translation
4 − the data written to the file will be formatted as follows: a two-byte length of the RU, followed by the exact PIU received (which is the TH, RH, and the full RU)

The transparency level options will rarely be used in an RJE environment. They are provided for flexibility in the use of **dc3770** as a general communications tool.

−c *ccpath* *ccpath* is the carriage control file pathname.

The carriage control definition generated is used only until the host sends a definition. Most host applications send their own carriage control definitions. It should, therefore, not be necessary to use this capability with most host applications.

ccpath is the name of the file containing the definition. The file contains a text description of the parameters supplied by an SVF or SHF command (that being the command that the

host sends to the printer for this purpose). Each parameter is on a separate line and consists of a keyword and a numeric parameter. Not all keywords need be supplied. The omission of a keyword, or a numeric parameter of zero, implies that the default value should be used.

The following is a list of the keywords and their permissible values:

MPL – maximum print line; this value specifies the last usable line of a page (form). Valid values are from 1 to 127. The default value is 1. None of the other carriage control parameters (**TM**, **BM**, **T***nn*) may be greater than the value of the **MPL** parameter.

TM – top margin; valid values range from 1 to the value of **MPL**. The default value is 1. The **TM** value is also used to set channel 1 of the virtual carriage-control tape.

BM – bottom margin; this specifies the line value following which an automatic skip to the top margin of the next page takes place. Valid values range from the value of **TM** to the value of **MPL**. The default value is **MPL**.

T*nn* – tab stop parameters, where *nn* is a number from 1 to 11. The vertical tab stop parameters set the line number values for use with the vertical tab (VT) or the select (SEL) functions (these functions are placed by the host within the data stream destined for the printer). Valid tab stop values are between the top margin and the bottom margin. 0 is also valid and means "no tab stop for this channel." T1 specifies the vertical tab stop setting for channel 2, T2 for channel 3, and so forth.

MPP – maximum presentation position. Defines the length of a line. Valid values are from 1 to 134; default is 132.

LM – left margin. Valid values are from 1 to **MPP**. The default is 1.

HT – horizontal tabstop. Sets a tab stop at the column position specified by the value. Valid values are from 1 to 134. Values may be specified in any order.

Examples

1. **rje −a −b −c ccpath −m P**

   assigns the default carriage control specified by **ccpath** to all printers.

2. **rje −a −b**

   is the command most likely to apply to your system. It is automatically issued upon start-up and designates batch routing for all output.

3. **rje −a −b −t 1 −x 1**

   assigns the 48 output destinations for LU session 1. Code set translation will not be performed for any outputs received on this LU.

Logging Sessions On and Off the Host

These commands log sessions on or off. They are privileged commands. The format of the log on command is:

rje −s −son [**−h** *rjepipe*] [**−M** *msg_file*]

The format of the log off command is:

rje −s −sof [**−h** *rjepipe*] [**−M** *msg_file*]

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings.

The following options may appear with either command:

−t *session* *session* specifies the session the command affects. Its value must be in the range 0-6.

Value	Meaning
0	Command applies to all sessions that are not busy.
1-6	Specific session to which the command applies.

This option defaults to **−t** 0.

−w wait; specifies that the command wait until the job has been sent to the host before returning control to the user. (If there is an error, control is returned immediately.)

−k *'text'* this option has one parameter, the string *text*. *text* should be enclosed in single quotes to avoid any expansions and/or substitutions of the characters by the UNIX system shell, and cannot be longer than 120 characters. **−k** *'text'* should be the last thing on the command line.

−p [*path*] This option has a single parameter *path*, which is a pathname to a file, the content of which is executed as a command.

−q quit; same as option **−k 'LOGOFF'**. If a pathname or **−k** is not specified, **−q** is the default.

Examples

1. **rje −s −son −t 3 signon.card**

   This logs on session 3.

2. **rje −s −son signon.card**

   signon.card is a file containing the sign-on information for the host. The following is an example of a sign-on card:

   ```
   /*SIGNON      RMT3
                (col 16)
   ```

3. **rje −s −sof −t 3**

   This logs off session 3.

4. **rje −s −sof**

   This logs off all sessions from the host. **−q** is assumed as the default, since neither a pathname nor **−k** are specified.

Local Status Command

This command gives the status of the sessions in progress, and the local job queue status. It has the following format:

rje −d [**−h** *rjepipe*] [**−M** *msg_file*] *option...*

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings.

The following are the available options:

−t *session* *session* specifies the session the command affects. Its value must be in the range 0-6.

Value	Meaning
0	Command applies to all LU sessions that are not busy.
1-6	Specific session to which the command applies.

This option defaults to **−t** 0.

−m *medium* specifies that the output assignments be displayed according to *medium*. This option can only be used with the **−a** option.

Value	Meaning
C	Card Image
E	Exchange Diskette
P	Printer

If *medium* is not specified, the default is all.

−s *subaddr* specifies that the output assignments be displayed according to *subaddr*, where *subaddr* is a value in the range 0-F (hexadecimal). For typical batch subsystems, subaddress 0 corresponds to RDR0, subaddress 1 corresponds to RDR1, etc; subaddress F frequently means any reader. This option can only be used with the **−a** option.

If *subaddr* is not specified, the default is all.

−z	indicates that the session status is requested by this command
−a	displays the output assignments
−d	displays decompaction table (1-6 or all)
−j	displays job queues
−l	displays the status of the multi-signal and exchange record length (ERCL) modes (see **rje −k −l m**). The ERCL options are as follows: medium = E (exchange medium) or C (card image) If medium = C, a hexadecimal value indicates maximum card length. The value, X'00', indicates an 80-column length and ERCL mode is not in effect. Any value other than X'00' indicates that ERCL mode is on.

Examples

1. **rje −d −a −t 1**

   This obtains all the output assignments for session 1.

2. **rje −d −z −j −d −l**

   This obtains all possible information about all the sessions.

Local 3770 Console Command

This command is used to control certain functions of the **dc3770** process. It is a privileged command. It has the following format:

rje −k [**−h** *rjepipe*] [**−M** *msg_file*] *option...*

where **−h** *rjepipe* and **−M** *msg_file* have their standard meanings.

The following list describes the other options of the **rje −k** command:

−q session log off requested; logs off the session(s) specified in the **−t** option. This causes **dc3770** to send the RSHUTD RU to the host on the specified sessions. The host then sends a SHUTD to **dc3770**. Neither the host nor **dc3770** starts new transmissions at this point. Finally, when everything is quiet, **dc3770** sends SHUTC to the host, and the host sends UNBIND to **dc3770**.

−t *session* *session* specifies the session the command affects. Its value must be in the range 0-6. This option only has meaning when used with the **−q** option.

Value	Meaning
0	Command applies to all sessions that are not busy.
1-6	Specific session to which the command applies.

This option defaults to **−t** 0.

−l *a* toggle an aspect of line setup status. The parameter *a* must be **m** or **r** to indicate which property to toggle. (Generally, there is no need to use this option.)

m − specifies that the multiple signal mode is to be toggled. When **dc3770** has something to send, and all available sessions are tied up, there is a discipline whereby **dc3770** may ask the host to interrupt the transmission of an outbound file temporarily, so that **dc3770** may send data on that session. This is done by **dc3770** sending an SNA signal RU to the host. When it receives this signal, the host application sends an RU with the end of chain and change direction bits set in its request header as soon as it can. This gives **dc3770** permission to send. Certain host systems will not hear **dc3770** unless it sends two signals. If multi-signal mode is set, **dc3770** sends two signals; if clear, it sends one.

r − specifies that the extended record length definition mode

is to be toggled. This applies only to the card image medium.

Normally, for the card image medium the record length definition is that all records whose length is equal to or less than 80 bytes are considered to be of the same (default) length; 80 bytes long. This is designated by a record length of 0 in the Function Management Header type 1 (FMH-1), which is sent at the beginning of a file.

When the extended record length definition mode is toggled on, every time the record length changes, a new FMH-1 is sent specifying the exact length of the new record. Extended record length definition mode should not be used unless it is important to inform the host of the exact length of every record. In the worst case, it could at least quadruple the transmission time required to send a file.

Examples

1. **rje −k −t 5 −q**

   This requests that session 5 be shut down.

2. **rje −k −l m −l r**

   This command line toggles both the multi-signal mode and the ERCL mode.

The **rje −d −l** command can be used to check the status of the multi−signal and ERCL modes.

System Shutdown

To shut down the system, it is necessary to:

Step 1. Stop the **dc3770** process, and

Step 2. Stop the SNA Controller.

Note that the SNA Controller processes should be stopped only after the **dc3770** process has been stopped.

Stopping the dc3770 Process

The following steps will stop the **dc3770** process.

Step 1. Log in as root.

Step 2. Change directory to the **runtime** directory for the **dc3770** process (normally **/usr/snaadm/runtime**).

Step 3. Check to see that the system environment has been properly set. (See "Configuring the System Environment for SNA".)

Step 4. Optionally, you may log off the LU sessions before stopping the **dc3770** process. However, this is not necessary, because **rje −q** shuts down all sessions to the host before stopping.

Step 5. Enter the command

rje −q [**−h** *rjepipe*] [**−M** *msg_file*]

This command will stop the **dc3770** process. There are no options to the command.

dc3770 Directory Cleanup

When the **dc3770** process is shut down, the **DISPTCH.LOG** file, the system log file specified at start-up time, and the subdirectories **output**, **jobs**, and **tmp** are not automatically cleaned up or removed. This allows the system to be restarted without losing output files or queued jobs.

The following files and subdirectories may need to be cleaned up by the user at system shutdown or before system start-up.

- The **DISPTCH.LOG** file contains information about how output was or was not routed by the **disptch** process.
- The log file contains 3770 console log file information. If the same log file name is used when restarting the SNA/RJE Emulator, the log file will be overwritten. Therefore, you may wish to save the log file by renaming it.

- The **output** directory contains all received files not routed by the **disptch** process. This directory may have to be cleaned out manually; that is, the System Administrator may have to send the files left in this directory to their owners. Depending on local usage patterns, it may be possible to automate this procedure partially.
- The **jobs** subdirectory contains the jobs still on queue to be sent to the host. Therefore, care should be taken before removing any files left in this subdirectory.
- The **tmp** subdirectory is a directory for temporary files used by the **rje** process. This directory should simply be cleaned or removed.

The following is a sample cleanup script:

```
mv logfile logfile.old
rm DISPTCH.LOG
rm jobs/*
rm tmp/*
```

Stopping the SNA Controller

The following steps will stop the SNA Controller:

Step 1. Log in as root.

Step 2. Change directory to **/usr/snaadm/runtime.**

Step 3. Check that the system environment is properly set. The **PCD** environment variable must be set to the communications port the SNA Controller is using, and the **P3274** variable should be set to the named pipe for this controller. This is very important if more than one SNA Controller is running.

Step 4. Enter the command

stopsna

The SNA Controller that is running on the port specified by the **PCD** environment variable will be stopped.

Appendices

Appendix A: Environment Variables A-1

Appendix B: Sample Configuration Source Files B-1

Appendix C: Sample Host Configurations C-1
Configuration – SNAJ1 C-1
Configuration – SNAJ2 C-3

Appendix D: NCP Gens D-1
MVS/JES NCP Gen D-1
DOS/VS POWER NCP Gen D-4

Appendix E: Error Messages E-1

Appendix F: Program Check Error Codes F-1

Appendix G: Communication Check Error Codes G-1

Appendix H: DC3770 and RJE Messages H-1

Appendix I: iscconfig Command Options I-1

Environment Variables

Variable	Description
BR	The value of this variable may be used to name the rje pipe when a **dc3770** process is started. The user interface, **rje**, may also use the value of **BR** to identify the rje pipe to the **dc3770** process. This variable corresponds to the **−h** option in the **dc3770** and **rje** commands. Note that **BR** is not set by **snaenvset**. Example: **BR=/usr/snaadm/runtime/HOSTA**
CONFIG	This is the full pathname of the configuration object file that will be used to start the SNA Controller. It is a composite of the value of the **rt** environment variable, **CSNA**, and the value of the **SNAHOST** environment variable. Default setting: **CONFIG=/usr/snaadm/runtime/CSNA.teke**
cust	This is the base path to the customization directories. It is a composite of the **sna** environment variable and **cust**. Default setting: **cust=/usr/snaadm/cust**
GEMFL	The value of this variable may be used to name the message file (the **−M** option in the **rje** commands). Default value: **GEMFL=rje.msg**
PCD	**PCD** specifies the port number for the SNA Controller to use. It is referenced by the **sdlc** process when the SNA Controller is started. On a 3B2 Computer, **PCD** indicates the ISC slot number (for example, 6). On a 3B5/15 Computer, **PCD** indicates the IOA device to be used (for example, 201). On a

Variable	Description
	3B4000 Computer, **PCD** is a combination of the pe number and the ISC slot number (for example, 120.6 for pe120, slot 6). Default setting: **PCD=1**
PATH	This is the standard UNIX environment variable modified to include the value of the **rt** environment variable as one of the entries. Example: **PATH=/usr/snaadm/runtime:/etc:/bin:/usr/bin**
P3274	This names the *snapipe* used for communications between the **sna** and the **dc3770** processes. (When the **dc3770** process is started, the *snapipe* for the SNA Controller it will use must be provided.) Each SNA Controller has a unique *snapipe*. Example: **P3274=/tmp/P3274.2**
rt	This is the base path to the **runtime** directory. It is a composite of **sna** and **rt**. It is used in the **startsna** script. Default setting: **rt=/usr/snaadm/runtime**
sna	This is the base path for all the Emulator+ directories. It is obtained from the **snaadm** login. Default setting: **sna=/usr/snaadm**
SNAHOST	This variable should be set to the suffix of the "CSNA." configuration object file that will be used to start the SNA Controller. Default setting: **SNAHOST=teke**

Sample Configuration Source Files

This appendix contains examples of SNA configuration source files.

Example 1:

This is a dual-identity SDLC type configuration. The first controller type (PU) is a 3770 SNA/RJE workstation with 6 LUs. The second controller type is a 3270 with 8 LUs configured. Note that each controller type has its own xid and address. (The line control options, point-to-point/multi-point, nrz/nrzi, and fdx/hdx, are independent of the controller types (physical units) specified.)

```
multipoint
nrzi

controller 3770

lu1     3
lu2     3
lu3     3
lu4     3
lu5     3
lu6     3

xid 12345
address 02
seg 265

controller 3270
lu0     2
lu1     2
lu2     2
lu3     2 termself
lu7     1
lu8     2
lu9     2
lu15    1

xid 12340
address  01
seg     265
broadcast
```

Example 2:

This example illustrates a dual-identity SDLC type configuration. Both controller types (PUs) are defined as 3770 workstations.

```
multipoint
fdx

controller 3770

lu1     3
lu2     3
lu3     3
lu4     3
lu5     3
lu6     3

xid 12345
address 01

controller 3770

lu1     3
lu2     3
lu3     3
lu4     3
lu5     3
lu6     3

xid 12341
address 03
```

Sample Host Configurations

Configuration - SNAJ1

This example represents a host definition for a remote SNA/RJE station.

Summary: This example defines to JES2 no compression and no compaction. This workstation will have a console, and the printer and punch will be operating in automatic forms mode.

System defaults will be taken on all printer, punch, and reader definitions unless specifically stated in the detailed outline.

Detail:

RMT3	Defines remote 3
LUTYPE1	Terminal Type
BUFSIZE=256	Defines the largest buffer size to be sent or received
NOCOMP	Specifies that this device does not have compression facility
NOCMPCT	Specifies that this device does not have compaction facility
CONSOLE	Specifies that this device has an operator console
DISCINTV=0000	Specifies (in seconds) terminal disconnect time if no successful text transmission has occurred
NUMPR=4	Specifies four (4) remote printers
NUMPU=3	Specifies three (3) remote punch devices
NUMRD=7	Specifies seven (7) remote readers
SETUPINF	Specifies that the printer and punch operate in automatic forms mode
WAITIME=0001	Specifies (in seconds) the time to wait after the processing of an inbound data stream has been completed

PRINTER DEFINITIONS:

PRWIDTH=132 -> PR1 - PR4
CKPTLNS=32767 -> PR1 - PR4
CKPTPGS=32767 -> PR1 - PR4

PUNCH DEFINITIONS:

CKPTPGS=32767 -> PU1 - PU3
CKPTLNS=32767 -> PU1 - PU3

NOSEP	Specifies that separator pages should not be generated for punch data (i.e. no header or trailer pages)

Configuration - SNAJ2

This example represents a host definition for a remote SNA/RJE station.

Summary: This example defines to JES2 that there is compression and compaction. This workstation will have a console, and the printer and punch will be operating in automatic forms mode.

The compaction table contains all alphanumeric characters including upper and lower case letters and most special characters.

System defaults will be taken on all printer, punch, and reader definitions unless specifically stated in the detailed outline.

Detail:

RMT4	Defines remote 4
LUTYPE1	Terminal Type
BUFSIZE=256	Defines the largest buffer size to be sent or received
COMP	Specifies that this device has compression facility
CMPCT	Specifies that this device has compaction facility
CONSOLE	Specifies that this device has an operator console
DISCINTV=0000	Specifies (in seconds) terminal disconnect time if no successful text transmission occurred
NUMPR=4	Specifies four (4) remote printers
NUMPU=3	Specifies three (3) remote punch devices
NUMRD=7	Specifies seven (7) remote readers
SETUPINF	Specifies that the printer and punch operate in automatic forms mode
WAITIME=0001	Specifies (in seconds) the time to wait after the processing of an inbound data stream has been completed

PRINTER DEFINITIONS:

PRWIDTH=132 -> PR1 - PR4
CKPTLNS=32767 -> PR1 - PR4
CKPTPGS=32767 -> PR1 - PR4

PUNCH DEFINITIONS:

CKPTLNS=32767 -> PU1 - PU3
CKPTPGS=32767 -> PU1 - PU3

NOSEP	Specifies that separator pages should not be generated for punch data (i.e. no header or trailer pages)

COMPACTION TABLE

COMPACT NAME=ATTIS, NUMBER=01
CHARS=(14,C1,C2,C3,C4,C5,C6,C7,C8,C9,D1,D2,D3,
D4,D5,D6,D7,D8,D9,E2,E3,E4,E5,E6,E7,E8,E9,81,
82,83,84,85,86,87,88,89,91,92,93,94,95,96,97, 98,99,
A2,A3,A4,A5,A6,A7,A8,A9,4C,4A,4E,4D,50, 5A,5E,5C)

NCP Gens

This is an example of an MVS/JES NCP gen. The line is configured as a multidrop line. The first PU, address C1, is configured as a 3274. The second PU, address C2, is configured as a 3770. Note that in the 3770 PU, DLOGMOD must be set to equal BATCH. This gen allows both the AT&T 3270 Emulator+ and the AT&T SNA/RJE Emulator+ to run concurrently on your 3B Computer using the same SDLC line to the host.

```
I1P4800S    LINE      ADDRESS=(014,HALF),
                      CLOCKING=EXT, EXTERNAL CLOCKING
                      DUPLEX=HALF, HALF-DUPLEX MODE
                      NEWSYNC=NO, DON'T SUPPLY NEW-SYNC SIGNAL TO MODEL
                      NRZI=NO, NON-RETURN-ZERO
                      PAUSE=0.2, AVERAGE DURATION OF POLLING CYCLE
                      RETRIES=5, FIVE RETRIES IN SEQUENCE
                      SERVLIM=5, MAX REGULAR SCANS OF SERVICE ORDER TBL
                      SPEED=4800, LINE SPEED
                      TRANSFR=9, 9 BUFFERS FOR MAX DATA FROM LINE AT ONCE
I1SRVC      SERVICE   ORDER=(I1327401,I1377702)
I1327401    PU        ADDR=C1, ADDRESS OF PU
                      DISCNT=NO, DON'T DISCONNECT WHEN NO-SESSIONS
                      IRETRY=YES, RETRY POLLING AFTER IDLE TIMEOUT
                      MAXDATA=265, MAX DATA BYTES IN PIU/PIU SEGMENT
                      MAXOUT=7, MAX PIU'S/PIU SEGMENTS BEFORE RESPONSE
                      PACING=1
                      PASSLIM=33, MAX CONSECUTIVE PIU'S/PIU SEGMENTS
                      PUTYPE=2, TYPE OF PU
                      RETRIES=(,1,4), M=64 IN SEQ, T=1 SECOND, N=4 SEQUENCES
                      VPACING=1 VTAM TO NCP
I111        LU        LOCADDR=2
I112        LU        LOCADDR=3
I113        LU        LOCADDR=4
I114        LU        LOCADDR=5
I115        LU        LOCADDR=6
I116        LU        LOCADDR=7
I117        LU        LOCADDR=8
I118        LU        LOCADDR=9
I1377702    PU        ADDR=C2, ADDRESS OF PU
                      BATCH=YES
                      DISCNT=NO, DON'T DISCONNECT WHEN NO-SESSIONS
                      DLOGMOD=BATCH,
                      IRETRY=YES, RETRY POLLING AFTER IDLE TIMEOUT
                      MAXDATA=521, MAX DATA BYTES IN PIU/PIU SEGMENT
                      MAXOUT=7, MAX PIU'S/PIU SEGMENTS BEFORE RESPONSE
                      PASSLIM=7, MAX CONSECUTIVE PIU'S/PIU SEGMENTS
                      PUDR=YES, DYNAM
                      PUTYPE=2, TYPE OF PU
                      RETRIES=(,1,5),
```

```
                        PACING=(5,1),
                        VPACING=(7,1) VTAM TO NCP
I121        LU          LOCADDR=1
I122        LU          LOCADDR=2
I123        LU          LOCADDR=3
I124        LU          LOCADDR=4
```

This is an example of DOS/VS POWER NCP gen. The line is configured as a multidrop line. The first PU, address C1, is configured as a 3770. Note that BATCH and DLOGMOD are set to equal BATCH. The second PU, address C2, is configured as a 3270. Note the BATCH is equal to NO and DLOGMOD is equal to S3270. This gen allows both the AT&T 3270 Emulator+ and the AT&T SNA/RJE Emulator+ to run concurrently on your 3B computer using the same SDLC line to the host.

```
IJS4800S    LINE    ADDRESS=(024),
                    ANSTONE=NO, NO ANSWER-TONE AFTER CALL-IN
                    ANSWER=ON, SET ANSWER MODE ON
                    CLOCKING=EXT, EXTERNAL CLOCKING
                    DUPLEX=HALF, HALF-DUPLEX MODE
                    INTPRI=2,
                    ISTATUS=ACTIVE, BRING UP WITH NETWORK
                    NEWSYNC=NO, DON'T SUPPLY NEW-SYNC SIGNAL TO MODEL
                    NRZI=NO, NON-RETURN-TO-ZERO
                    POLLED=YES,
                    PAUSE=0.6, AVERAGE DURATION OF POLLING CYCLE
                    RETRIES=(7,1,5)
                    SPEED=4800, LINE SPEED
                    TRANSFR=32, 32 BUFFERS FOR MAX DATA FROM LINE AT ONCE
IJ327401    PU      MAXLU=20
            CATALS  B.SWNETIJ
            BKEND
* 851218EVJJ        ADD ATT/SUMMIT
SWNETIJ     VBUILD  TYPE=SWNET
IJPU01      PU      ADDR=C1, ADDRESS OF PU
                    BATCH=YES
                    DISCNT=NO, DON'T DISCONNECT WHEN NO-SESSIONS
                    DLOGMOD=BATCH, DEFAULT LOGMODE ENTRY
                    IDBLK=013, 017 FOR A 3274
                    IDNUM=00021, XID NUMBER IS X'00023'
                    IRETRY=YES, RETRY POLLING AFTER IDLE TIMEOUT
                    ISTATUS=ACTIVE, INITIAL STATUS IS ACTIVE
                    MAXDATA=256, MAX DATA BYTES IN PIU/PIU SEGMENT
                    MAXOUT=7, MAX PIU'S/PIU SEGMENTS BEFORE RESPONSE
                    MODETAB=ISTINCLM, DEFAULT MODE TABLE ISTINCLM
                    PACING=1,
                    PASSLIM=7, MAX CONSECUTIVE PIU'S/PIU SEGMENTS
                    PUTYPE=2, TYPE OF PU
                    SSCPFM=USSSCS, CHARACTER CODED RUS ARE SUPPORTED
                    VPACING=1, VTAM TO NCP
I121        LU      LOCADDR=1
                    ISTATUS=ACTIVE
I122        LU      LOCADDR=2
                    ISTATUS=ACTIVE
I123        LU      LOCADDR=3
```

```
                        ISTATUS=ACTIVE
I124        LU          LOCADDR=4
                        ISTATUS=ACTIVE
IJPU02      PU          ADDR=C2, ADDRESS OF PU
                        BATCH=NO
                        DISCNT=NO, DON'T DISCONNECT WHEN NO-SESSIONS
                        DLOGMOD=S3270, DEFAULT LOGMODE ENTRY
                        IDBLK=017, 017 FOR A 3274
                        IDNUM=00IB2, XID NUMBER IS X'00IB2'
                        IRETRY=YES, RETRY POLLING AFTER IDLE TIMEOUT
                        ISTATUS=ACTIVE, INITIAL STATUS IS ACTIVE
                        MAXDATA=256, MAX DATA BYTES IN PIU/PIU SEGMENT
                        MAXOUT=7, MAX PIU'S/PIU SEGMENTS BEFORE RESPONSE
                        MODETAB=ISTINCLM, DEFAULT MODE TABLE ISTINCLM
                        PACING=1,
                        PASSLIM=9, MAX CONSECUTIVE PIU'S/PIU SEGMENTS
                        PUTYPE=2, TYPE OF PU
                        SSCPFM=USSSCS, CHARACTER CODED RUS ARE SUPPORTED
                        VPACING=1, VTAM TO NCP
IJ11        LU          LOCADDR=2
                        ISTATUS=ACTIVE
IJ12        LU          LOCADDR=3
                        ISTATUS=ACTIVE
IJ13        LU          LOCADDR=4
                        ISTATUS=ACTIVE
IJ14        LU          LOCADDR=5
                        DR=6
                        ISTATUS=ACTIVE
                        ISTATUS=ACTIVE
IJ15        LU          LOCAD
IJ16        LU          LOCADDR=7
                        ISTATUS=ACTIVE
IJ17        LU          LOCADDR=8
                        ISTATUS=ACTIVE
IJ18        LU          LOCADDR=9
                        ISTATUS=ACTIVE
```

Error Messages

Error Message	Explanation/Possible Cause
gem file *file* not found	The file containing error messages does not exist.
system error *errno* during opening gem file	System error during opening error message input file.
system error *errno* during opening gem output file	System error during opening the output file for error messages.
message file too small	File containing error messages is not of correct length.
CONFIGURATION FILE NOT FOUND	Configuration file does not exist
CONFIGURATION FILE OPEN ERROR	System error during opening configuration file.
CONFIGURATION FILE READING ERROR	System error during reading configuration file.
INVALID CONFIGURATION FILE VERSION *vers_number*	Incorrect configuration file.
LENGTH MISMATCH READING CONFIGURATION FILE	The length of configuration file is not consistent with the expected length.
TOO MANY CONTROLLERS CONFIGURED	The number of controllers in the configuration file is greater than the maximum limit.
CONFIGURATION ERROR: NO LU'S DEFINED	No LU's defined in the configuration file.
CONFIGURATION ERROR: TOO MANY LU'S DEFINED	More than maximum allowed number of LU's have been configured.
INADMISSIBLE SDLC	The station address is greater than

Error Message	Explanation/Possible Cause
STATION ADDRESS *address* CONFIGURED	255 or less than 1.
INADMISSIBLE DEVICE CLASS *class* CONFIGURED	Configured device class does not exist or is not consistent with controller type.
INADMISSIBLE LINK OPTIONS *mask* CONFIGURED	Link options mask contains non-applicable bits.
INVALID CONTROLLER TYPE *type*	Controller type, specified in configuration file, is not any of SNA3770, SNA3270, SNA8100

Program Check Error Codes

Value	Explanation
401	unknown data stream command
402	invalid buffer address in data stream
403	data follows 1-byte commands in a data stream
404	data stream ends in order-pending state
405	invalid source device on copy command, or source device buffer locked on copy command, or source and destination device incompatibility on copy command
406	ESC character missing in second position of command sequence
411	Request/Response Unit (RU) too long (LU.T1)
413	function not supported
420	exception response request received when definite response only specified by BIND
421	definite-response request received when exception response only specified by BIND
422	NO response not allowed
423	format indicator not allowed
430	sequence number error
431	chaining error
432	bracket error
433	data traffic inactive
434	direction error
443	read command must have Change Direction but not End Bracket
445	Activate Logical Unit (ACTLU) request is for neither cold activation nor Error Recovery Procedure (ERP)
450	BIND profile error
451	BIND primary protocol error
452	BIND secondary protocol error
453	BIND common protocol error
454	BIND screen size error
455	BIND LU profile error
456	BIND LU1 error
457	BIND cryptography specified

Value	Explanation
462	data stream error detected by LU.T1
470	unknown data byte X'00' - X'3F' or X'FF'
480	user request lost due to host SELECT (BS ___ arrival of message from controller)
490	buffer not available for write command
498	negative response received
499	exception request received

Communication Check Error Codes

Value	Explanation
501	Data Set Ready (DSR) lost
502	Clear To Send (CTS) lost
504	Normal Disconnect Mode (NDM)
505	NDM
510	Physical Unit (PU) is not active (this is SNA condition)
518	segmentation error; internal Deactivate Physical Unit (DACTPU) (this is an SNA condition)
519	received frame too long
520	timeout (no frames)
521	timeout (no flags)
525	20 Exchange Identification (XID) commands received in a row
528	Frame Reject Response (FRMR) sent; Frame Reject Mode (FRM) entered (internal DACTPU)
529	modem acting up (internal DACTPU)
530	clocking or CTS lost (internal DACTPU)

DC3770 and RJE Messages

MESSAGE LIST FORMAT:

Actual messages in the following are given as

00:"message text %[*parameter*]X"

where 00: is the message number, the characters in quotes are the fixed text of the message, and %[*parameter*]X is a parameter whose value will be plugged into the message.

Parameter format is %[*parameter*]X, where %[] serve to set off the parameter, *parameter* is the symbolic parameter name, and X is the substitution type.

Possible substitution types are:

- d decimal value; for example, %[*m4msgno*]4d. In this example, the optional length specifier (4d instead of just d) says that the value will always be expanded to 4 characters, with leading zeroes prepended.
- x hexadecimal; a length specifier is possible.
- s string value.
- c character value.
- E EBCDIC hexdump. Takes two parameters, address and length.
- A ASCII hexdump (not currently used).

STANDARD PARAMETER USAGE:

Parameters in the output message are given the symbolic names that follow; the normal meaning of each parameter is discussed. There are other possible parameters; the ones listed here are only those with "standard" meanings that apply to their usage in many messages.

m4msgno is the message number.
m4jobno is normally a job number, from 1 to 30000.
m4jtype is a job type, a, l, or j.
m4lu_no is the LU number to which the message applies.
m4errno is usually the errno value returned by a UNIX system call
non–standard values are:
70, 71, 73: see message 97.
83 record too long.
84 Wrong uid.

85 wrong slot or file index not found.
86 hostin file not readable.
87 hostin file empty.

m4uccmd is the command type.
m4media is a medium.
m4subba is a subaddress.
m4pathn is a pathname.
m4pname is a pathname.

ERROR MESSAGES

0:"unknown message %[*m4msgno*]d"

1:"RJE: stdin file operation error"
Applicability: **rje −s**
Category: error, from RJE only.
Parameter usage: standard
Explanation:
While trying to get input from **stdin** for the pathname "−", an error condition occurred.

2:"RJE: file-object %[*m4pname*]s not exist"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory

3:"RJE: error in sending msg through %[*m4pname*]s"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation:
An attempt to send your command to the controller failed.

4:"RJE: error in receiving messages"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: Error receiving responses to command.

5:"RJE: fail to connect SNA controller through %[*m4pname*]s"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory.

6:"RJE: fail to connect BSC controller through %[*m4pname*]s"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory.

7:"RJE: error in formatting output msg"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory.
Probable cause is a corrupt message file.

8:"RJE: error,code %[*m4errno*]d"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory.

10:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d QUEUED FOR LU %[*m4lu_no*]d"
Applicability: **rje −s**
Category: informational, to RJE only.
Parameter usage: standard
Explanation:
This message tells the user that the job has been accepted by the console task and has been queued for execution. If the job has been queued for LU 0, this signifies that it will be submitted for execution to the first LU available for jobs of this type.

11:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c INITIATED BY LU %[*m4lu_no*]d"
Applicability: **rje −s**, type J or type A
Category: informational, log file only.
Parameter usage: standard
Explanation:
This message tells the user that the execution of the job has begun. When message 10 for this job specified LU 0, this message tells the user which LU was finally chosen for execution of the job.

12:"CO%[*m4msgno*]4d: JOB DISPLAY REQUEST: NO JOBs QUEUED."
Applicability: **rje −d −j**

Category: informational, to RJE only.
Parameter usage: standard
Explanation:
When a request to display the job queues is made and there are no jobs queued, this message is the result.

13:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c REJECTED BY LU %[*m4lu_no*]d ILLEGAL MEDIUM"
Applicability: **rje –s**, type J jobs only.
Category: error, to RJE only.
Parameter usage: standard
Explanation:
When the LU is bound to the host application, byte 25 of the BIND RU contains bits that specify what media may or may not be sent to the host. If the specified LU (or unspecified LU if the LU parameter is 0) cannot send the media requested for the job, this message will be seen.
See also messages 14 and 98.

14:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c REJECTED BY LU %[*m4lu_no*]d ILLEGAL SESSION STATUS"
Applicability: **rje –s**
Category: error, to RJE only.
Parameter usage: standard
Explanation:
In order for the LU to send type J or type A jobs, it must be in session (a BIND was received from the host and accepted). In order to send type L jobs, it must be active (an ACTLU was received from the host and accepted).
When the user did not specify an LU, and when all but one of the possible LUs reject the job for whatever reason, the job is internally re-queued for a specific LU; if rejection occurs later, the LU parameter of the message will specify that LU rather than 0.
When the user did not specify an LU, and when some LUs reject the job for illegal medium and others reject it for illegal session status, either message 13 or message 14 will be seen, depending on which form of rejection happened last. This situation will occur only rarely.
See also message 98.

15:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c FOR LU %[*m4lu_no*]d CANCELLED"

Applicability: **rje −s**
Category: error, to RJE only.
Parameter usage: standard
Explanation:

This message is sent to the RJE program that requested cancellation of the job, as well as to any RJE program that may be awaiting completion of the job; in the first case, it indicates success of the cancellation, in the second it indicates failure of the job.

16:"CO%[*m4msgno*]4d: PERMISSION DENIED: COMMAND TYPE %[*m4uccmd*]c IS PRIVILEGED."
Applicability: **rje −k**
Category: error
Parameter usage: standard
Explanation:

Certain command types are "privileged". To use them, your effective userid must be root (0), or identical to the effective userid of **dc3770**.

17:"CO%[*m4msgno*]4d: CANCEL REJECTED, JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c LU %[*m4lu_no*]d: TOO LATE"
Applicability: **rje −c**
Category: error
Parameter usage: standard
Explanation:

The job to be cancelled is already in the process of being sent, and has in fact already been effectively completed: it is too late to cancel it.

18:"CO%[*m4msgno*]4d: CANCEL REJECTED, JOB J%[*m4jobno*]5d: JOB NOT FOUND"
Applicability: **rje −c**
Category: error
Parameter usage: standard
Explanation:

The job to be cancelled was not found on the internal job queues. Perhaps it has already been completed (whether successfully or not), perhaps it never existed (you simply typed the wrong job number).

19:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: JOB %[*m4jobno*]5d WAS PURGED."
Applicability: **rje −s**

Category: error, log file and RJE.
Parameter usage: standard
Explanation:
When the LU-LU session is lost (the host sends UNBIND), all type A or type J jobs queued for an LU are purged.
When even the SSCP-LU session is lost (DACTLU, DACTPU), all jobs of all types queued for the LU are purged.
When the state of the LU-LU session changes to "between brackets", any currently active type A or type J jobs are purged.
Jobs that are purged are considered to have failed; they are removed from the internal job queue.
See also messages 26 and 27.

20:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c
LU %[*m4lu_no*]d: JOB FAILED: ERROR CODE %[*m4whyxx*]d"
Applicability: **rje −s**
Category: error, to RJE.
Parameter usage: *m4whyxx* is an internal error code.
Others are standard.
Explanation:
The inbound job failed for one of the following reasons:
201: a record is larger than the largest RU that can be sent, but the BIND for the LU does not permit a record to be spanned from one RU to another.
others: a file management operation occurred. See the log file for details. The error number given is usually the *errno* returned by a UNIX system call.

21:"CO%[*m4msgno*]4d: OUTBOUND JOB FAILED: MED=%[*m4media*]c
SUB=%[*m4subba*]2x TYPE=%[*m4jtype*]c LU%[*m4lu_no*]d
ERROR=%[*m4whyxx*]d"
Applicability: no command
Category: error, to log file.
Parameter usage: *m4whyxx* is an internal error code.
Others are standard.
Explanation: See also message 102.
The outbound job failed for one of the following reasons:
201: an unsupported SCS function was encountered.
202: SCS parameter error.
101: for a "non-repeat" SCB, the count field is too large to fit within the current RU.

102: for a "repeated non-blank" SCB, the count field is too large to fit within the current RU.
103: for a "compacted data" SCB, the count field is too large to fit within the current RU.
104: an SCB count field of 0 was encountered.
For the above errors, see also message 102.
151: an FMH error was detected.
others: a file management operation occurred. See the log file for details. The error number given is usually the *errno* returned by a UNIX system call.

22:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c
LU %[*m4lu_no*]d: JOB FAILED: NEGATIVE RESPONSE %[*m4whyxx*]4x"
Applicability: **rje –s**
Category: error, to RJE
Parameter usage: *m4whyxx* is the first 2 bytes of the SNA sense code. Others are standard.
Explanation:
The inbound job failed because a negative response was received from the host. An explanation of the sense code data can be found in any of several IBM publications.

23:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d RESPONDS %[*m4lresp*]4x TO BIND: %[*m4bindl,m4binda*]E (END OF MESSAGE CO%[*m4msgno*]4d)"
Applicability: no command
Category: informational, log file.
Parameter usage:
m4lresp is the first 2 bytes of the SNA sense code.
m4bindl is the length of the BIND RU.
m4binda is the address of the BIND RU.
(in the output message there appears a hex dump of the BIND RU; see example, below.)
Others are standard.
Explanation:
A BIND has been received from the host. A response of 0000 means that it has been accepted. Other responses mean that it has been rejected: this is a matter to take up with your host system's system programmer.
The sense data and format of the BIND RU may be found in Part 1, Chapter 2 of IBM publication number GC20-1868, "Sessions Between Logical Units". The BIND RU itself starts at address 0009 in the hex dump (i.e., byte 0 of the BIND, which has a

hexadecimal value of 0x31, is on the line starting with "00000000", under the column labeled "8---"; its printable value, under the dash in "8-A", to the right of the line, is shown as '.' because 0x31 is not a printable character in EBCDIC.

Example:

```
CO0023: LU 1 RESPONDS 0000 TO BIND:
EBCDIC   0--- 2--- 4--- 6--- 8--- A--- C--- E---     0-2-4-6-8-A-C-E-
00000000 2d00 0101 16f9 6b80 0031 0103 03b3 a370     .....9,.....t.
00000010 8000 0185 8502 0001 1000 0091 00c0 0000     ...ee.....j.{..
00000020 0100 4000 08e5 d1c5 e2f0 f0f0 f100          .. ..VJES0001...
   (END OF MESSAGE CO0023)
```

24:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c
LU %[*m4lu_no*]d: JOB COMPLETED"

Applicability: **rje –s**
Category: informational
Parameter usage: standard
Explanation: The inbound job specified has been successfully complete

25:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d UNBOUND"

Applicability: no commands
Category: informational, log file.
Parameter usage: standard
Explanation: The LU-LU session has been lost.

26:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: %[*m4njobs*]d WERE PURGED: "

Applicability: **rje –s**
Category: error, log file
Parameter usage: standard
Explanation:

One or more jobs have been purged.
See also message 19, which is sent to an **rje** program waiting for job completion, and message 27, of which one or more occurrences will follow.

27:" J%[*m4jobno*]5d (PURGED)"

Applicability: **rje –s**
Category: error, log file
Parameter usage: standard
Explanation:

The job specified was purged.
See also message 19, which is sent to an **rje** program waiting

for job completion, and message 26, of which one occurrence will precede this.

28:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: SSCP DATA RECEIVED: %[*m4sdata*]s"
Applicability: no commands.
Category: informational, log file
Parameter usage: standard
Explanation:
The data given in the message was received on the SSCP-LU session.

29:"CO%[*m4msgno*]4d: OUTBOUND (MED=%[*m4media*]c, SUB=%[*m4subba*]2x) LU %[*m4lu_no*]d: JOB COMPLETED"
Applicability: no commands.
Category: informational, log file
Parameter usage: standard
Explanation:
An outbound transmission was successfully completed.

30:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: XRJE COMMAND COMPLETED"
Applicability: **rje −k**
Category: informational
Parameter usage: standard
Explanation:
The command was completed with no errors or warnings.
See also messages 31 to 33.

31:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: XRJE COMMAND: WARNING: RSHUTD REQUESTED IN IDLE STATE"
Applicability: **rje −k**
Category: informational
Parameter usage: standard
Explanation:
The command was completed with no errors.
The desired state was already in effect.
See also messages 30 to 33.

32:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: XRJE COMMAND: WARNING: EXTENDED RECORD-LENGTH MODE CHANGED WHILE TRANSMITTING"
Applicability: **rje −k**
Category: informational
Parameter usage: standard

Explanation:
The command was completed with no errors.
There is a very remote possibility that the currently active inbound job could fail.
See also messages 30 to 33.

33:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d: XRJE COMMAND:
WARNINGS: RSHUTD REQUESTED IN IDLE STATE EXTENDED
EXTENDED RECORD-LENGTH MODE CHANGED WHILE TRANSMITTING"
Applicability: **rje −k**
Category: informational
Parameter usage: standard
Explanation:
The command was completed with no errors.
The warnings from messages 31 and 32 are both given.
See also messages 30 to 33.

34:"CO%[*m4msgno*]4d: JOB QUEUE FULL"
Applicability: **rje −s**
Category: error, to RJE.
Parameter usage: standard
Explanation:
The job could not be accepted because too many requests are already outstanding.

35:"CO%[*m4msgno*]4d: CONTROLLER IS BUSY"
Applicability: all commands
Category: error, to RJE.
Parameter usage: standard
Explanation:
The command could not be accepted because too many other commands are already outstanding.

36:"CO%[*m4msgno*]4d: JOB J%[*m4jobno*]5d TYPE %[*m4jtype*]c
PERMISSION DENIED"
Applicability: none
Category: none
Parameter usage: none
Explanation: This message number is unused.

37:"CO%[*m4msgno*]4d: TYPE %[*m4uccmd*]c COMMAND BUSY"
Applicability: **rje −k**
Category: error, to RJE
Parameter usage: standard

Explanation:
Only one command of this command type may be executing at any given time. Since these are privileged commands, it should be simple enough to determine with whom you were in conflict and whether it is necessary to re-issue the command.

38:"CO%[*m4msgno*]4d: COMMAND REJECTED: TOO MANY
Applicability: **rje −s**
Category: error, to RJE
Parameter usage: standard
Explanation:
The number of **rje −s** commands which wait for the job to be completed is limited to fewer than the total number of simultaneous commands permitted.

39: #follow the next with a message from 40: to 60: selected by m4upyyy.
39:"CO%[*m4msgno*]4d: COMMAND SYNTAX ERROR: parameter number %[*m4parno*]d, reason was: "
Applicability: all commands
Category: error, to RJE
Parameter usage: standard
Explanation:
The parameter number indicated provoked an error; the reason for the error is given by the next message (41 through 60). See also messages 41 through 60.

40:"normal return, correct option."
Explanation: This message number is unused.

41:"invalid option."
Explanation:
A totally unrecognized option letter, or one inappropriate to the current command, was encountered; or
premature end of command line; or
invalid LU number; or
invalid transparency level.
Example:
rje −s −z

42:"invalid file mode (e | a) specified."
Explanation:
For **rje −a**, file modes **−a** and **−e** were both specified.
For **rje −s**, both **−x** and either **−a** or **−e** were specified.

43:"pathname expected."
Explanation:
A pathname was expected when the end of the command line was reached.

44:"repeated parameter."
Explanation: unused by **dc3770**; may be output by RJE.

45:"console string expected"
Explanation: The command line ended immediately after "**–k**".

46:"'m' or 'r' expected"
Explanation: The command line ended after "**rje –k –l**".

47:"invalid medium option."
Explanation:
For **rje –s**, medium was not C nor E nor K;
for **rje –a**, or **rje –d –a**, medium was not C nor E nor P;
for either case, command line ended after **–m**.

48:"more than 6 pathnames."
Explanation:
For **rje –s**, type J jobs, from 1 to 6 pathnames may be specified.

49:"only one pathname permitted for this command."
Explanation: self-explanatory.

50:"job number option is not numeric."
Explanation: unused by **dc3770**; may be output by RJE.

51:"invalid subaddress."
Explanation:
subaddress parameter was not a single hex digit (0-9, a-f, or A-F); or, command line ended prematurely.

52:"total length of saved parameters is too great (many long pathnames)."
Explanation:
self-explanatory; the limit is around 600 characters in total, 100 characters per pathname.

53:"LU number expected."
Explanation:
Command line ended after **–t**; or, no **–t** option was given with a command that requires it (**rje –a –1**).

54:"TRN option expected."

Explanation: Command line ended prematurely.

55:"invalid command"
Explanation: unused by **dc3770**; may be output by RJE.

56:"missing option letter after '—'."
Explanation: command line ended prematurely.

57:"option inconsistency."
Explanation:
This message is applied in cases where either one of several parameters must be chosen, or one choice precludes another, or both.

58:"no job number specified."
Explanation: unused by **dc3770**; may be output by RJE.

59:"no parameters at all."
Explanation: The command requires at least some parameters.

60:"numeric conversion result out of range"
Explanation:
A numeric value had either a non-digit or an unreasonable value.

61:"CO%[*m4msgno*]4d: FILE MANAGEMENT OPERATION FAILED: see log file for reason."
Applicability: various commands
Category: error, log file and to RJE.
Parameter usage: standard
Explanation: See the messages beginning with message 80.

62:"CO%[*m4msgno*]4d: COMMAND ERROR:
Applicability: **rje —s**
Category: error, to RJE
Parameter usage: standard
Explanation:
The console text sent by a type A or type L job may be no more than 120 characters in length.

63:"CO%[*m4msgno*]4d: POWER OFF IN PROGRESS"
Applicability: **rje -q**
Category: informational
Parameter usage: standard
Explanation: The command has succeeded.

65:"CO%[*m4msgno*]4d: RJE -A COMMAND SUCCEEDED FOR LU %[*m4lu_no*]d"
Applicability: **rje –a**
Category: informational
Parameter usage: standard
Explanation:
Self explanatory; a message for LU 0 comes last, and means that all assignments have succeeded.

66:"CO%[*m4msgno*]4d: RJE -A COMMAND FAILED PARTIALLY FOR LU %[*m4lu_no*]d"
Applicability: **rje –a**
Category: error, to RJE
Parameter usage: standard
Explanation:
The assignment failed because an outbound job was in progress for the selected combination of LU, medium, and subaddress.

67:"CO%[*m4msgno*]4d: JOBS ON QUEUE '%[*m4jtype*]c' FOR LU %[*m4lu_no*]d:"
Applicability: **rje –d –j**
Category: informational
Parameter usage: standard
Explanation:
This begins a job queue display.
The queue types are:
'a', 'l', 'j', are job types A, L, J, respectively; these are internally queued by the console task. The queue for LU 0 consists of jobs for unspecified LU.
'x' is "executing", already out of the purview of the console task, and owned by an individual LU. The queue for LU 0 consists of jobs for unspecified LU; jobs appear briefly on queue 'x' for LU 0 when they are offered to the various LUs to see which will accept.
See also message 68.

68:" J%[*m4jobno*]5d"
Applicability: **rje –d –j**
Category: informational
Parameter usage: standard

Explanation: see message 67.

69:"EXTENDED RECORD LENGTH MODE IS %[*m4exrcl*]2xMULTISIGNAL MODE IS %[*m4mults*]2x"
Applicability: **rje −d**
Category: informational
Parameter usage: standard
Explanation: result of the −l option.

70:"ASSIGNMENT FOR LU %[*m4lu_no*]d, MEDIUM %[*m4media*]c, SUBADDRESS %[*m4subba*]2x and PATHNAME: %[*m4pathn*]s"
Applicability: **rje −d −a**
Category: informational
Parameter usage: standard
Explanation:
Self-explanatory except when pathname is given as "BATCH OPERATION" (**rje −d −a −b**).

71:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d STATUS REPLY: SESSION STATE IS %[*m4sessf*]d, FULL STATE %[*m4xlumx*]d"
Applicability: **rje −d −z**
Category: informational
Parameter usage: standard
Explanation:
session state 0, full state 1 is inactive state; no ACTLU has arrived yet.
session state 16, full state 2 or 3 is "SSCP" state; SSCP-LU data may be exchanged, LU-LU data may not.
An ACTLU has been received; in full state 3, a BIND has also been received, but not an SDT yet.
session state 17 is "between brackets", no activity yet.
session state 18 is any send state, inbound data is flowing.
session state 19 is any receive state, outbound data flows.
full state 4 is between brackets.
full state 5 is send, 6 is send-in-chain, 7 is send-pending (a response from the host is expected in state 7).
full state 8 is receive, 9 is receive-purge (an error occurred).
full state 10 is an internal transition state, and is unlikely ever to be seen in this message.

72:"ACTIVE MEDIUM=%[*m4media*]c, SUBADDRESS=%[*m4subba*]2x"
Applicability: **rje −d −z**

Category: informational
Parameter usage: standard
Explanation:
If session state is 18 or 19, this is the medium/subaddress pair on which current activity is happening.

73:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d BIND DATA=%[*m4bindl,m4binda*]E"
Applicability: **rje −d −z**
Category: informational
Parameter usage: standard
Explanation: The BIND RU is repeated here.

74:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d HAS NO DECOMPACTION TABLE."
Applicability: **rje −d −d**
Category: informational
Parameter usage: standard
Explanation: self-evident; see also message 75.

75:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d DECOMPACTION TABLE:%[*m4funcx,m4decom*]E"
Applicability: **rje −d −d**
Category: informational
Parameter usage: standard
Explanation: self-evident; see also message 74.

76:"command is finished"
Applicability: all commands
Category: informational, to RJE
Parameter usage: standard
Explanation:
RJE does not print this message. It is used internally to indicate that processing is finished.

77:"THE FOLLOWING MESSAGE COULD NOT BE SENT TO PROCESS %[*m4ucpid*]d:"
Applicability: all commands
Category: informational
Parameter usage: standard, log file
Explanation:
If an RJE command is killed, its output will be diverted to the log file. See also messages 78, 197, and 198.

78:"THE PRECEDING MESSAGE COULD NOT BE SENT TO PROCESS %[*m4ucpid*]d."

Applicability: all commands
Category: informational
Parameter usage: standard, log file
Explanation:
If an RJE command is killed, its output will be diverted to the log file. See also messages 77, 197, and 198.

79:"ZZ%[*m4msgno*]d: error %[*m4errno*]d occurred while trying to read from user pipe."
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
The pipe from user commands was unreadable for some reason. The process will terminate.

80:"ZZ%[*m4msgno*]d: ERROR CREATING %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.
Files created by **dc3770** include job-queue files, the outbound assignment file, and outbound data files.

81:"ZZ%[*m4msgno*]d: ERROR OPENING %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.

82:"ZZ%[*m4msgno*]d: ERROR WRITING %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.

This message is self-explanatory.

83:"ZZ%[*m4msgno*]d: ERROR CLOSING %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.

84:"ZZ%[*m4msgno*]d: ERROR IN FCNTL FOR %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.
The operation in question is done for named pipes only, **-s** and **-p** options on the **dc3770** command line.

85:"hostout assignments affected are:"
Applicability: none
Category: none
Parameter usage: none
Explanation: This message number is unused.

86:"ZZ%[*m4msgno*]d: ERROR IN LSEEK FOR %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.
An lseek operation may be attempted on almost any file known to **dc3770**; because the operating system limits the number of files open at any one time, it is possible that **dc3770** will close its least-recently-used file and later reopen it when it is needed again. An **lseek()** is done before closing to remember the current position in the file, and another is done after

reopening to restore the position.

87:"ZZ%[*m4msgno*]d: ERROR IN READ FROM %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.

88:"ZZ%[*m4msgno*]d: ERROR LINKING %[*m4pname*]s TO %[*m4pathn*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.
Usually, this message occurs at the start of a session, when the **jobs** directory was not cleaned out before running **dc3770**.

89:"ZZ%[*m4msgno*]d: ERROR UNLINKING %[*m4pname*]s: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.

90:"ZZ%[*m4msgno*]d: SYSTEM ERROR IN %[*m4pname*]s OPERATION: errno was %[*m4errno*]d"
Applicability: not command-related
Category: error, log file
Parameter usage: standard
Explanation:
Messages 80 to 90 are referred to by message 61.
This message is self-explanatory.
The operation referred to is typically either **ftok**, **msgget**, or **msgrcv**.
Usually occurs because SNA was killed.

The process will terminate.

91:" AT&T-IS AT&T-IS
AT&T-IS 3770 SNA EMULATOR+ AT&T-IS
AT&T-IS**dc3770** version 3.0COPYRIGHT (c) 1985,1986
by SYSTEMS STRATEGIES INC., All Rights Reserved."
Applicability: not command-related
Category: informational, log file
Parameter usage: standard
Explanation: The log file will start with this message.

92:"**dc3770** terminated."
Applicability: not command-related.
Category: informational, log file
Parameter usage: standard
Explanation: On normal termination, the log file ends with this messa

93:"ZZ%[*m4msgno*]d: PATHNAME %[*m4pname*]s TOO LONG"
Applicability: various commands
Category: error, log file
Parameter usage: standard
Explanation:
Referred to by message 61.
Pathnames are generally limited to 100 characters.

94: # temp job filename format
94:"./jobs/TF%[*m4jobno*]5d"
Applicability: none
Category: none
Parameter usage: none
Explanation: This message number is unused.

95: # user pid suffix for pipe name
95:".%[*m4jobno*]d"
Applicability: none
Category: none
Parameter usage: none
Explanation: This message number is unused.

96:"ZZ%[*m4msgno*]d: HOSTOUT file error %[*m4errno*]d,
OUTPUT %[*m4pname*]s exists."
Applicability: not command related.
Category: error, log file
Parameter usage: standard

Explanation: Referred to by message 61.

97:"ZZ%[*m4msgno*]d: CARRIAGE CONTROL FILE %[*m4pname*]s
ERROR:code is %[*m4errno*]d"
Applicability: **rje −a**
Category: error, log file
Parameter usage: standard
Explanation: Referred to by message 61.
The *errno* parameter may have the following non-standard values:
70 number out of range.
T*nn* must be from 2 to 11,
parameter values may be from 0 to 127.
71 read error occurred during syntax analysis.
73 none of the other parameters may have values greater than **MPL**.

98:"CO%[*m4msgno*]4d: NO LUs AVAILABLE FOR TYPE %[*m4jtype*]c JOBS"
Applicability: **rje −s**
Category: error, to RJE
Parameter usage: standard
Explanation:
The job was rejected by the console task even before submission to any LU. See also messages 13 and 14.

99:"CO%[*m4msgno*]4d FILE ERROR : CANNOT STAT %[*m4pname*]s,
ERRNO %[*m4errno*]d."
Applicability: **rje −a**
Category: error, to RJE
Parameter usage: standard
Explanation:
While attempting to validate a user-supplied pathname parameter, a call to **stat**(2) failed. *Errno* 2 (ENOENT), "no such file", is the most likely cause.

100:"CO%[*m4msgno*]4d FILE ERROR : %[*m4pname*]s IS EMPTY."
Applicability: **rje −s**
Category: error, to RJE.
Parameter usage: standard
Explanation: Attempting to send an empty file is considered an error.

101:"CO%[*m4msgno*]4d FILE ERROR : %[*m4pname*]s
READ PERMISSION DENIED."
Applicability: **rje −s**

Category: error, to RJE
Parameter usage: standard
Explanation:
In order to send a file to the host, the user must have read permission on the file.
(If the old **rje** program from cBSCRJE was used, the file must have read permission for "others".)

102:"DC%[*m4msgno*]d: OUTBOUND DATA FORMAT ERROR:
error %[*m4errno*]d at position 0x%[*m4jobno*]x
in the following data:%[*m4njobs,m4jblst*]E"
Applicability: not command-related.
Category: error, log file
Parameter usage: standard
Explanation: see also message 21.
For reasons 101 to 104 and 201 to 202, the LU sends both message 21 to the console task and message 102 to the log file. Message 102 contains fuller information to assist in problem determination.

103:"CO%[*m4msgno*]4d: CANCEL J%[*m4jobno*]4d PERMISSION DENIED"
Applicability: **rje −c**
Category: error, to RJE
Parameter usage: standard
Explanation:
To cancel a job sent by another user, you must have root privileges.

104:"RJE: CAN'T MAKE RETURN MSG PIPE"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory.

105:"RJE: RETURN RESPONSE ERROR"
Applicability: **rje**
Category: error, from RJE only.
Parameter usage: standard
Explanation: self-explanatory.

106:"CO%[*m4msgno*]4d: OUTBOUND JOB STARTED LU %[*m4lu_no*]d
MEDIUM %[*m4media*]c SUBBADDRESS %[*m4subba*]2x
PATHNAME %[*m4pname*]s"
Applicability: not command-related.

Category: informational, to log file.
Parameter usage: standard
Explanation: Self-explanatory.

107:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d POWERED OFF, CAN ACCEPT NO REQUESTS"
Applicability: all commands.
Category: error, to RJE.
Parameter usage: standard
Explanation: A request has specified an LU that is unavailable.

110:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d POWERED OFF"
Applicability: **rje** -q
Category: informational
Parameter usage: standard
Explanation:
This message does not currently appear. It may be added in a later release.

111:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d CONNECTION FAILED: LU BUSY"
Applicability: process initialization.
Category: error, to log file.
Parameter usage: standard
Explanation:
dc3770 is already running on this SNA.
See also message 111.
If all 6 connections fail, the **dc3770** process will terminate normally.

112:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d CONNECTION FAILED: NO APPLICABLE LUS"
Applicability: process initialization.
Category: informational
Parameter usage: standard
Explanation:
Check the SNA configuration (**snopts**):
if the PU in the SNA process was configured for fewer than 6 LUs, then one or more instances of this message will appear. In such a case, the message is informational and represents no error.
If the PU was configured for 3270 usage, or some other error occurred, then all 6 LUs will send this message and the **dc3770** process will terminate normally.

115:"CO%[*m4msgno*]4d: LU %[*m4lu_no*]d REPORTS RESET OR COMMUNICATION CHECK %[*m4errno*]d:"

Applicability: not command-related.
Category: informational
Parameter usage: standard, except that *m4errno* is the code from the SNA engine

Explanation:

Followed by a message from 116 to 125, this message reports on exception conditions. Message 121 at the start of a session usually indicates that the switched connection to the host system has not yet been established.

116:" DSR SIGNAL DROPPED."

See message 115; *m4errno* is 1; communications check.

117:" CTS SIGNAL DROPPED."

See message 115; *m4errno* is 2; communications check.

118:" DISC RECEIVED."

See message 115; *m4errno* is 3; SDLC reset.

119:" SNRM RECEIVED."

See message 115; *m4errno* is 4; SDLC reset.

120:" RECEIVE BUFFER OVERFLOW."

See message 115; *m4errno* is 5; communications check.

121:" NON-PRODUCTIVE TIMEOUT."

See message 115; *m4errno* is 6; communications check.

122:" CONNECTION PROBLEM."

See message 115; *m4errno* is 7; communications check.

123:" FRMR SENT."

See message 115; *m4errno* is 8; communications check.

124:" MODEM ERROR."

See message 115; *m4errno* is 9; communications check.

125:" other reasons."

See message 115; *m4errno* has a value not on this list.

126:"ACTPU received"

See message 115; *m4errno* is 20; SDLC reset.

127:"DACTPU received"

See message 115; *m4errno* is 21; SDLC reset.

128:"Segmenting error"
See message 115; *m4errno* is 21; communications check.

155:"RJE: INVOCATION ERROR"

156:"RJE: OPTIONS NOT SPECIFIED"

157:"RJE: HOST NOT FOUND"

158:"RJE: CONTROLLER NOT ACTIVE"

159:"RJE: HOST DOES NOT REPLY"

160:"RJE: AN INVALID OPTION IS PRESENT"

161:"RJE: OPTION REPEATED"

162:"RJE: INVALID PARAMETER FOR OPTION"

163:"RJE: REQUIRED OPTION IS MISSING"

164:"RJE: OPTION IS INVALID FOR CURRENT MODE"

165:"RJE: OPTION IS INVALID FOR COMMAND"

166:"RJE: FILE NAMES EXCEED MAXIMUM CHARACTER LENGTH"

167:"RJE: RDR FAILED TO SEND - CANNOT OPEN FILE"

168:"RJE: RDR FAILED TO SEND - CANNOT READ FILE"

169:"RJE: CRJE COMMAND REJECTED - FILE NOT QUEUED"

170:"RJE: SRJE COMMAND REJECTED - QUEUE OVERFLOW"

171:"RJE: SRJE COMMAND QUEUED AS %[*m4pname*]s"

172:"RJE: FILE-OBJECT SUCCESSFULLY SENT "

173:"RJE: JOB CANCELLED"

174:"RJE: COMMAND FAILED - CHECK LOGFILE/CONSOLE FOR REASON"

175:"RJE: COMMAND ACCEPTED - CHECK LOGFILE/CONSOLE FOR SPECIFICS"

Applicability:	**rje**
Category:	error, from RJE only.
Parameter usage:	standard
Explanation:	self-explanatory.

These are the cBSC/RJE error messages.

197:"(debug) message read from sna (length %[*m4njobs*]d):%[m4njobs,*m4jblst*]E"
Applicability: no commands
Category: informational
Parameter usage: standard
Explanation: See message 198.

198:"(debug) message written to sna (length %[*m4njobs*]d):
%[*m4njobs,m4jblst*]E"
Applicability: no commands
Category: informational
Parameter usage: standard
Explanation: See also message 197.
This message records in the log file the entire and exact message sent to the SNA engine (message 197 records the message received). It will appear in the log file if **dc3770** is started with the option **−D**. A log file thus produced can be useful in reporting system problems.
SIGUSR2 toggles this behavior. Even if the process was not started with the **−D** option, this debugging trace can be turned on dynamically; receipt of the signal changes the state of this output (messages 197, 198, 77, and 78) from on to off or vice-versa.
When messages 197 and 198 are being produced, all output to user commands (except **rje −d −a**) is also reproduced in the log file, surrounded by messages 77 and 78.
Naturally, production of this output slows down the process; additionally, there is the danger that the log file thus produced will grow to an unreasonably large size. Therefore it is only in exceptional cases that this facility should be used.

199:
There may be no message numbers greater than 199.
This is an arbitrary limit imposed at compilation time.

iscconfig Command Options

The format of the **iscconfig** command is as follows:

iscconfig [**−P** *pe number*] *driver* [**−a**] [**−r**] [**−d**]

iscconfig manages the ISC card configuration on a 3B2 Computer or on an ACP attached to a 3B4000 Computer. Using this command, you can add new ISC cards to the configuration, remove cards from the configuration, or change the driver assignments on existing cards.

iscconfig also manages the device special files for the ISC cards in the configuration. It creates device entries when new cards are added to the configuration and removes them when cards are deleted from the configuration. The following is a description of the available options:

−P *pe number* is used only when configuring an ACP on a 3B4000 Computer. The *pe number* is the processor ID of the ACP to be configured.

driver is the name of the software driver to be configured.

The following options are mutually exclusive:

−a allows the user to add new cards or modify the driver assignment on existing cards.

−r allows the user to remove cards from the existing configuration.

−d displays the current configuration.

iscconfig also manages the creation of device entries in **/dev** and in **/adj/pe??/dev** for ACP's.

On a 3B2 Computer, device file names are of the form

type slot minor

where *type* is the lower case representation of the *driver* specified on the command line, *slot* is the slot number the ISC card is installed in, and *minor* is either 0 or 1 (for port 0 or port 1 on the ISC card). For example, a card in slot 4 assigned to the SNABSC driver will have device file names **/dev/snabsc40** and **/dev/snabsc41**. The major node for both devices is 4, and the minor numbers are 0 and 1 respectively.

On an ACP, device file names also incorporate the PE number of the ACP. They are of the form

type pe number slot minor

For example, a card in slot 4 of PE 120 assigned to the SNABSC driver will have device file names **/adj/pe120/dev/snabsc120.40** and **/adj/pe120/dev/snabsc120.41**. Redirect driver devices will also be created in **/dev** with the same names.

Index

Items in the index may appear in more than one document. Where this occurs, page numbers following an item are grouped by document. One of the following keys precedes each group of page numbers and tells you which guide to look in.

AG *Administrator's Guide*
UG *User's Guide*

A

API (Application Program Interface) functions **UG:** 7-8, 12-13, 23-24, 26
automatic routing **AG:** 21-22; **UG:** 2, 18, 20-22

B

BR environment variable **AG:** 20, 22, 24; **UG:** 3, 5, 6, 13-14, 16, 24, 27

C

comment card **UG:** 18, 21
console command **UG:** 7, 10, 17, 30
console log file **AG:** 21-22, 33; **UG:** 30
console terminal **UG:** 1, 17, 30
crje( ) **UG:** 7, 23-27

E

EBCDIC **AG:** 21, 25; **UG:** 2, 9-10, 12, 14

I

IRS character **AG:** 25; **UG**

J

job number **UG:** 1, 15,17, 25
job output **UG:** 21-22
job queue **AG:** 22, 29-30; **UG:** 1, 23, 28

L

librje.a **UG:** 7, 12, 23
log file **AG:** 20-22, 33; **UG:** 1, 4-5, 17, 20, 29-30

P

PATH environment variable **AG:** 3; **UG:** 3
PCD environment variable **AG:** 3, 17, 34
primary workstation **AG:** 6

R

rje −c UG: 23, 26-7
rje −s UG: 8, 11-12, 15-17
rtmsgs UG: 12, 13, 15

S

secondary workstation AG: 9
sign-on card AG: 28
srje( ) UG: 7-8, 12, 14-16, 24-6
srje_err UG: 12-13, 14-16, 24, 26-7

T

.TAG file AG: 21-22
tag file UG: 22
transparent mode UG: 10