HP 264X (ROVER) KERMIT USER GUIDE

John F. Chandler

Harvard/Smithsonian Center for Astrophysics

August, 1987

1. HP264x KERMIT (Rover-Kermit)

Program:	John F. Chandler (Harvard/Smithsonian Center for Astrophysics)
Language:	8080 Assembler
Version:	1.2
Date:	August 1987
Documentation:	John F. Chandler (CfA)

Rover-Kermit Capabilities At A Glance:

Local operation:	Yes
Remote operation:	No
Transfers text files:	Yes
Transfers binary files:	Yes
Wildcard send:	N/A
^X/^Z interruption:	Yes
Filename collision avoidance:	N/A
Can time out:	Yes
8th-bit prefixing:	Yes
Repeat count prefixing:	Yes
Alternate block checks:	Yes
Terminal emulation:	Host
Communication settings:	Host
Transmit BREAK:	Host
IBM communication:	Yes
Transaction logging:	No
Session logging (raw download):	Host
Raw upload:	Host
Act as server:	No
Talk to server:	Yes; SEND, GET, FIN, LOGOUT
Advanced commands for servers:	Yes; GENERIC, HOST, KERMIT
Local file management:	N/A
Handle file attributes:	No
Command/init files:	Host
Printer control:	Host

1.1. Summary of HP264x-Series Terminals

The 2640-series work stations combine terminal functionality with a variety of programmability and graphics features. The members of the series (all driven by 8080's) share a common set of ROM system entries for display, telecommunications, and device I/O, as well as a built-in loader driven by escape sequences with addresses and data expressed as octal character strings. The loader is not, however, a linking loader -- it accepts only absolute addresses and data.

Various members of the series offer storage devices in the form of cartridge tapes and diskettes. However, the diskettes are not used as random access media, rather as fast "tapes". This chapter will speak of all such devices as tapes.

The combination of absolute loader and primitive file system tends to push software design toward monolithic programs fitting into preassigned memory regions with immovable, globally known entry vectors. Insofar as the operating system entry vectors are the same among the different members of the series, these restrictions are an advantage in terms of software portability. Another result of these limits is that many customary Kermit functions, such as interactions with the local file system, are infeasible. At the same time, I/O is generally a simple matter for an application program, such as Kermit, since "one call does it all."

Rover-Kermit was developed on a 2647, which contains 64K of RAM in addition to 64K of bank-switched ROM.

Others in the series generally have no more than 64K total RAM+ROM, and certain system entries will vary accordingly along with the method of calling them. On the 2647, Kermit operates as a memory-resident system extension which can supersede the normal terminal function when needed. The 2647, being highly programmable, can be loaded with alternate state tables to emulate practically any other kind of terminal regardless of whether Kermit is present. The programmability of the others in the series varies -- the 2648, in particular, can not be set up to emulate other terminal types.

1.2. Rover-Kermit Description

Since Rover-Kermit runs on a standalone terminal, it is always in control of the screen -- it is always *local*. Thus, it always keeps the screen updated with the file name and the packet and retry counts, whether sending or receiving. See below for details of the screen layout. Because it is local, Rover-Kermit also monitors the keyboard and interrupts a file transfer in either direction upon recognizing a CTRL-X or CTRL-Z. As always in such cases, Rover-Kermit can stop unconditionally when sending but must rely on cooperation from the remote Kermit when receiving.

Rover-Kermit can call a timeout exit from a telecommunications input request and thereby break deadlocks automatically. The timeouts done by Rover-Kermit are "fuzzy" because they depend on the speed of the processor, the verbosity of the operating system code, and on the amount of keyboard activity. For the usual 2-MHz clock rate the timeout should closely approximate that requested by the remote Kermit anywhere up to 64 seconds. However, a request of zero is treated as 64, and requests over 64 are taken modulo 64. The timeout can also be disabled by command.

If despite the timeout capability, the transmission appears to be stuck, then manual intervention is probably needed above and beyond the usual response to a timeout (namely to retransmit the last packet or to NAK the expected packet to cause the remote host to send it again). In that case, the user may depress the CNTL and both shift keys simultaneously to gain keyboard access to telecommunications. At that point any string of characters up to, and including, a carriage return may be typed at the remote host to prompt further activity. The string will also be echoed locally regardless of the duplex setting. Since Kermit displays all characters transmitted by the remote host (except those within packets), the string will be displayed twice if the remote host also echoes. In any case, the record of the preceding non-packet traffic should indicate what is needed to restart the transmission. If all else fails, the transmission can be halted by pressing the "RESET TERMINAL" key. Rover-Kermit will then restart and wait for commands (such as QUIT). One note of warning: halting a transfer to or from tape will result in the abandonment of an entry in the Active File Table. Eventually, the entire table can be filled with abandoned entries, and it will be necessary to *Hard Reset* the terminal to recover the "lost" entries.

Screen Layout

While Rover-Kermit is running, the upper right corner of the screen always shows a list of Kermit commands. The remainder of the top eight lines are used for displaying Kermit's status. Figure 1-1 shows a snapshot of the display after a successful GET. The top line always shows Kermit's version number, and the second line provides a mnemonic label for the third, which displays the data Rover-Kermit will use for Send-Init negotiations. The third line also shows an asterisk under the word Core or Tape to indicate which form of storage is selected, plus the current definitions of the input and output "tape" devices. The fourth line (the Program State line) shows the most recent command (or error message from the remote Kermit). The fifth, sixth, and seventh lines show the most recent file transmitted and the packet and retry counts. The eighth line (the Message line) shows the most recent user prompt or error message, or else the Value-specified parameters after a PARM command. The remainder of the screen constitutes a window into the scrolled display workspace of the terminal. Text replies from a remote Server and any non-packet traffic during a transfer are added to the bottom of the workspace. While Rover-Kermit is running, the display can be scrolled by means of the ROLL UP, ROLL DOWN, HOME, and HOME DOWN keys (as during normal operation).

```
Rover Kermit 1.2                        Send, Receive, ...
            Btpp."8BR                   Core, Tape, Kermit, ...
     Parms: ~# @-#Y1~  Src: L  Dst: R   *
Get
      File: GOOD.DATA
    Record: 126
   Retries: 0
Transfer done
```

Figure 1-1: Screen snapshot after GET command

Rover-Kermit Commands

Rover-Kermit uses a minimal set of commands, each abbreviated to a single letter. Options, if any, will be requested by a subsequent prompt. While you are responding to a prompt, the characters you type will be upcased and displayed on the Message line, and both DEL and BACKSPACE will back up the cursor and remove the last character from the response string (though not from the display). Other control and function keys will be ignored (except RETURN, which terminates your response). During operation, Rover-Kermit maintains a display of all possible commands (and subcommands where appropriate), but offers no other HELP facility.

SEND
: Send the current file to the remote Kermit. Kermit will prompt you for the filename, but you need not enter one; if you just type a carriage return, Kermit will use a default name. In the case of a RAM file, the default is the name received with the file. However, for a tape file there is no name available, and the default is "A.B".

RECEIVE
: Receive file(s) from the remote Kermit. Store them on successive output tape files. Note: if you are receiving into the RAM file, each file received will overwrite any previous one.

GET
: When Rover-Kermit is talking to a Kermit Server on the remote host, you should use the GET command to request the server to send files to you. You will be prompted for the file name.

LOGOUT
: When talking to a remote Kermit Server, this command shuts down the server and logs it out, and also exits from Rover-Kermit to normal terminal operation. To protect you against inadvertantly striking the "L" key, Kermit prompts you for confirmation of the LOGOUT command.

FINISH
: Like LOGOUT, but shuts down the remote server without logging it out.

QUIT
: Exit to normal terminal operation.

KERMIT
: Send a command to a remote Kermit Server. You will be prompted for the command type (C for a remote host command, K for a Kermit command, or G for a generic Kermit command) and the text. The command character will not be displayed, but the command will appear on the Message line as you type it. Any reply will be displayed on the screen.

CORE
: Specifies input/output to the RAM file, up to 32K characters in length.

TAPE
: Specifies input/output to the cartridge tape units or other devices. The default device names under this option are L(eft tape) and R(ight tape) for input and output, respectively, but they may be reversed or set to any of the other possibilities, namely, S(ource), D(estination), G(raphics), N(ull), or U(serio). These represent the SOURCE or DESTINATION assignments in the terminal, the display (graphics for input, alpha for output), a null file, or an optional I/O device. See the terminal documentation for more details. The devices are selected via the PARM command.

PARM
: Sets or changes a parameter for Kermit operation. This command is divided into four sub-functions, each in turn abbreviated to a single keystroke (oN, oFf, Char, or Value), and each sub-function can operate on any one of a list of parameters. See below for a detailed description of the parameters and keystroke sequences.

Setting/Changing Parameters

Parameters for Rover-Kermit operation may be set by sequences of keystrokes typed while Kermit is running. Kermit provides a menu of possible options on the Program State line to guide in the entry of such sequences. All alphabetic keys are taken as upper case. Sequences consist of at least a "P", a sub-function, and a parameter name. The Value sub-function takes, in addition, a decimal number entered before the parameter name. The number must be entered with no imbedded blanks or commas. The values of all such parameters are displayed (in decimal) on the Message line after a PARM command completes. The Character sub-function, on the other hand, takes a character argument entered after the parameter name. Character parameters are permanently displayed on the third line, but the ON/OFF parameters (other than the eighth-bit flag, which doubles as a character parameter) are not displayed at all.

BLOCK PCB<n> sets the block check type to <n>. Available options are 1 and 2. The default is 1.

BUFSZ PV<n>B sets Rover-Kermit's buffer length to <n>. The valid range is 20-94, and 94 is the initial value.

DEST PCD<c> sets the output device code to <c>. The default is R, and other possible values are L, S, D, G, N, and U. See also the TAPE command.

HNDSHK
PV<n>H sets the IBM turnaround character to <n>. The default is 17 (XON), but any value 1-31 is valid.

IBM P(N|F)I sets the "IBM" flag. "ON" (the default) means that Rover-Kermit waits for the turnaround character before sending any packets (see HNDSHK above).

MARK PV<n>M sets the packet start character to <n>. The default is 1 (SOH), but any value 1-31 is valid.

QUOTE PCQ<c> sets the control quote character to <c>. The default is #. Note: letters may not be used for this character because the command interpreter upcases all keystrokes.

RETRY PV<n>R sets the packet retry limit to <n>. The default is 10, but any value 1-199 is valid.

RPTQ PCR<c> sets the repeat prefix character to <c>. The default is ~.

SOURCE PCS<c> sets the input device code to <c>. The default is L. Other possible values are R, S, D, G, N, and U. See also the TAPE command.

TIME PV<n>T sets the timeout limit to <n>. The default is 3, but any value 1-94 is valid.

TIMER P(N|F)T sets the timer on or off. The default is ON.

8-BIT P(N|F)8 sets the 8th bit prefix option on or off. "ON" (the default) permits 8th-bit quoting if the remote Kermit requests it.

8-BIT PC8<c> sets the 8th bit prefix character to <c>. Note: letters may not be used for this character because the command interpreter upcases all keystrokes.

Since Rover-Kermit is permanently resident, its configuration can also be set or examined from without by typing loader sequences or display commands at the operating system. Table 1-0 gives a list of memory locations that may be altered to change the Kermit defaults. Such settings may be combined as desired to obtain any particular configuration, and, since loader commands may come from any source, such commands may be called out from a tape file or sent by the remote host for convenience. Aside from this packaging property and the capacity for setting the I/O to HP-IB devices, these loader commands are obviously less convenient than the usual command sequences. Among other things, they can be executed only when Rover-Kermit is not running. Note: an I/O device code is actually a string ending with a carriage return, stored in a nine-byte array. The carriage return must always follow immediately after the device code.

For example, to assign the output to a printer on HP-IB#7, enter

```
<ESC>&c 107552a 110d 43d 67d 15D
```

To set the output back to the right tape, enter

```
<ESC>&c 107552a 122d 15D
```

To set the 8th-bit prefix to lower-case "z", enter

Name	Address	Values
BKTP	106773	Block check type. Default is 1 (61).
BUFSZ	106732	Buffer length for INIT. Default is 94. (176).
DEST	107552	Output device code. Default is R (122).
DPTQ	106725	Repeat prefix character. Default is ~ (176).
HNDSHK	104144	IBM turnaround character. Default is XON (21).
IBM	104155	315 if on, 1 if off. Default is ON.
MARK	105463	Packet start character. Default is SOH (1).
QUOTE	106720	Quote character. Default is # (43).
RETRY	105655	Retry limit. Default is 10. (12).
SOURCE	107605	Input device code. Default is L (114).
TIME	106741	Timeout limit for INIT. Default is 3. (43).
TIMER	105116	312 if on, 332 if off. Default is ON.
8-BIT	106772	8th bit quote option. Default is Y (131).

Table 1-1: Settable locations (addresses and values in octal)

```
<ESC>&c 106772a 172D
```

Error Messages

Error messages from Rover-Kermit are largely self-explanatory but brief. For conditions that terminate a file transfer, the same message is both displayed on the screen (on the Message line) and sent to the remote Kermit as an Error packet. In fact, whenever a transfer completes, some message will appear on the Message line, usually accompanied by an audible tone. The beep is suppressed only when a very short file transfer completes normally.

Remote host aborted	specifies that the remote Kermit issued an error packet.
Bad INIT data	specifies disagreement over the control quote character.
I/O error	specifies a tape I/O error or overflow of the RAM file.
Split prefix	specifies that a byte specification is apparently split between packets.
Bad repeat count	specifies that Rover-Kermit has attempted to decode a non-positive repeat count.
Bad packet type	specifies that Rover-Kermit has received an invalid type of packet.
Retry limit - <c>	specifies that Rover-Kermit has found an error repeatedly up to the retry limit. The last character indicates what kind of error: K => packet out of sequence, N => NAK or ill-formed packet, or T => timeout.
No local storage	specifies that no tape is inserted.

1.3. Rover-Kermit Source language

The Rover-Kermit has been assembled not with a standard 8080 assembler, but with a slightly eccentric one developed on an HP2647. The differences in notation are quite simple, but pervasive.

- A "*" in column 1 denotes a comment.
- Statement labels are never suffixed with colons.
- Octal constants are denoted by the suffix "Q".
- The current location counter is represented by the symbol "*".

- The operator ":" is used to represent concatenation of two quantities into a two-byte value, low-order byte first.

- There is a special pseudo-operation "ASCC" for assembling character strings. A string within quotes is assembled into standard ASCII codes. A quotation mark (apostrophe) may be inserted as a pair of quotes in the usual fashion. Control characters may be inserted by including triplets of decimal digits for the desired characters within quotation marks. Unless the string is followed by ",-", the assembler appends a null to the output. For example,

  ```
  ASCC  'It''s true!'013''
  ```

  yields: It's true!<CR><NUL>

- There is a special pseudo-operation "EJECT" to skip to the next page on the output listing.

- Quoted character strings are not allowed anywhere except in "ASCC" instructions. There is a function operation "CHAR", which evaluates to the ASCII character value of the next character after the delimiting blank. For example, "CHAR -" evaluates to 55 (octal) and may be used in expressions.

- There is a function operation "INSTR" which evaluates to the opcode of the symbol after the delimiting blank. For example, "INSTR MOV" evaluates to 100 (octal).

1.4. What's New

The following are the most important of the changes and improvements in release 1.2 of Rover-Kermit.

- Two-byte checksums.
- Mnemonic commands for setting parameters.
- More elaborate display of current settings.
- ^X/^Z interruption.
- Retain filespec on RAM file.
- Display any characters received while waiting for handshake.
- Fixed bug in creating repeat strings from runs longer than 94.
- Flush data communication buffer before starting any transaction (except the first in a given session).
- The ROLL and HOME keys now work for the conversation workspace.

1.5. Future Work

Work that could be done in future releases includes:

- Add multi-file transmissions from cartridge tape.
- Release Active File Blocks in case of error.
- Add three-byte block checks.
- Encode/decode error packets.
- Implement "A" packets.

Index

Table of Contents

List of Figures

List of Tables