Kermit for HP48/49/50 Calculators

Hoppi

Hoppi is an application for Apple macOS to allow communication with HP48/49/50-series calculators using the Kermit file transfer protocol. It supports both USB and RS232 calculator ports.

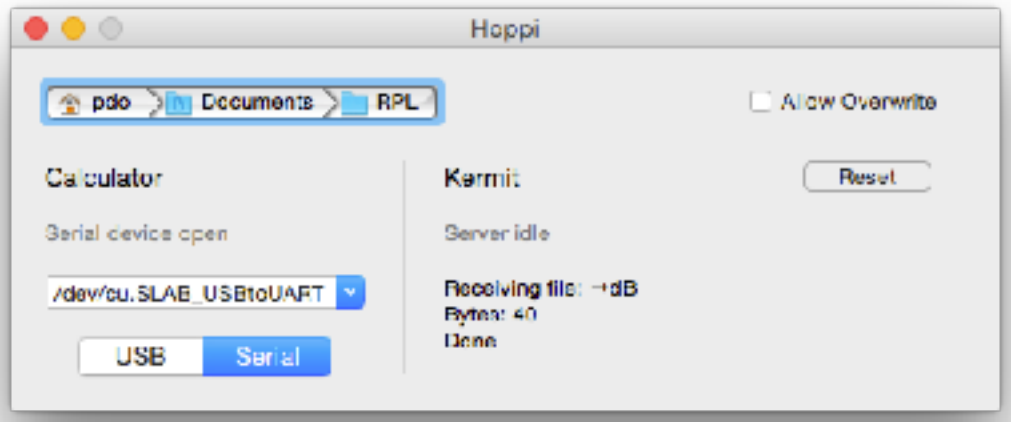
The application implements a Kermit server and presents a simple interface to the user allowing the state of the connection to be controlled and monitored.

In addition there is an RPL library, installable on the calculator, that allows easy access to Hoppi's advanced features, such as transferring arbitrary computer text files to and from the calculator as RPL string objects.

Hoppi Mac Application

Introduction

The Hoppi application main window is shown in the following figure, displaying its state after transferring a file called "→dB" from the calculator to the computer.



The main window is composed of three main areas:-

1. across the top of the window is the current workspace folder path,
2. on the left is the calculator status panel, and
3. on the right is the Kermit protocol status panel.

The current workspace folder is the folder on the computer from/to which all file transfers will be made. In this case it shows the “Documents/RPL/“ folder of my home directory. Click on any path component to open a dialog allowing you to choose a new folder. To allow files in this folder to be overwritten by transfers from the calculator tick the “Allow Overwrite” checkbox.

The calculator status panel, on the left-hand side of the window, shows the current state of the connection to the calculator. There are two modes of operation: USB or Serial. If USB is selected then Hoppi will attempt to communicate with the calculator via the calculator's USB port, and will show “calculator detected” when the calculator is attached and switched on. In this mode it will ignore the serial device choose-box. Alternatively, if Serial is selected, then Hoppi will try to communicate with the calculator's serial port, using the serial device selected in the drop-down choose-box.

Note: serial adaptor cables often connect to the computer via a USB port. However, the combination of cable and driver will appear as a serial device to any applications running on macOS, so to use one of these cables the Serial option should be selected. This applies, in particular, to those special all-in-one cables that connect the computer's USB port to an HP48 serial port. In short, select the option that corresponds to the *calculator* port that is being used.

The Kermit protocol status panel, on the right-hand side of the window, shows what is happening in the current transfer. In case of problems you can reset the state of the Kermit server by pressing the Reset button.

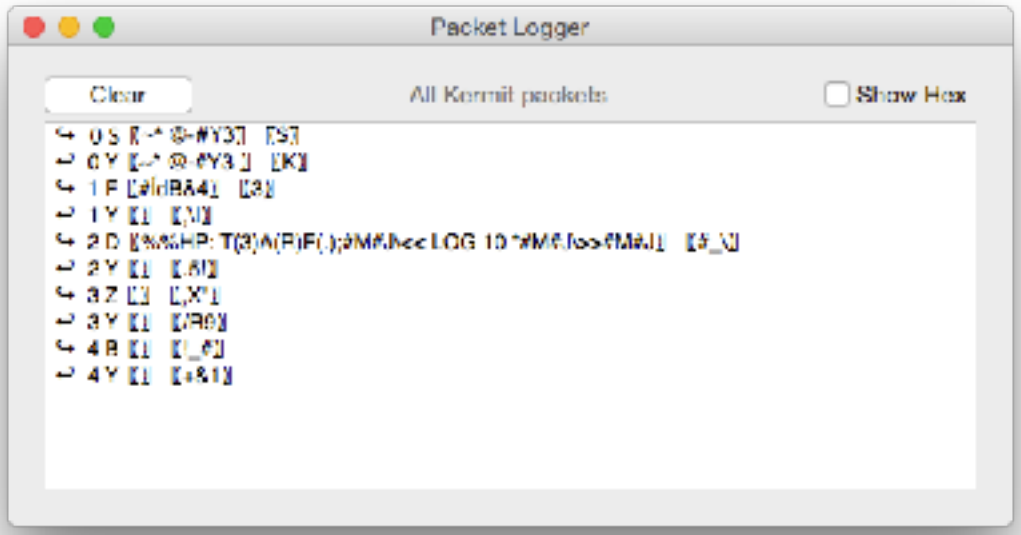
The Packet Logger Window

The packet logger window, accessible from the View menu, displays the contents of packets as they are sent and received.

The contents of each packet are displayed on a single line in five fields:-

1. a right-pointing arrow for a packet received from the calculator, a left-pointing arrow for a packet sent to the calculator,

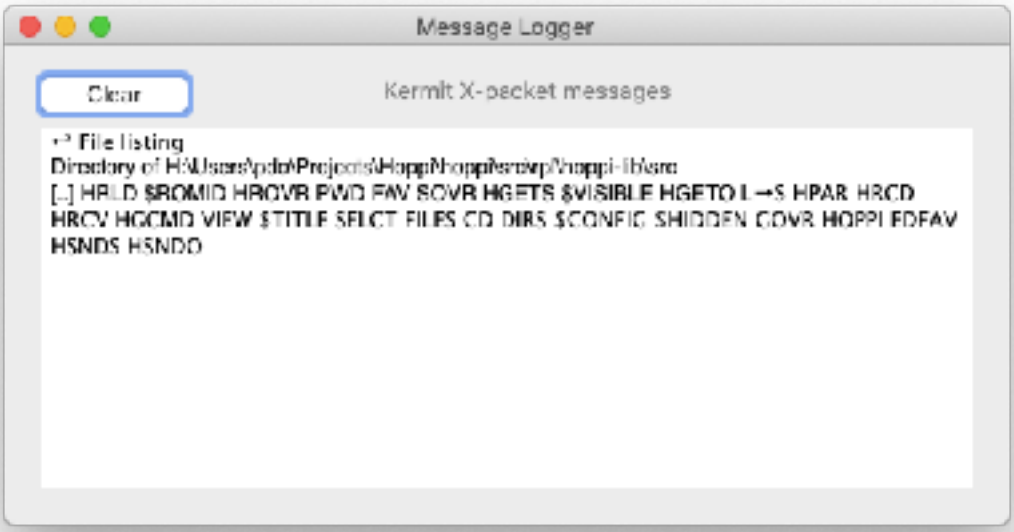
2. the sequence number of the packet,
3. the Kermit packet type (denoted by a single letter),
4. the data contents of the packet (enclosed in hollow brackets), and
5. the packet checksum (enclosed in hollow brackets).



In addition, ticking the “Show Hex” checkbox shows the data and checksum contents of each packet as sequences of hexadecimal byte values on a second line.

The Message Logger Window

As well as transferring files, the Kermit protocol supports the sending and receiving of arbitrary text messages. These are initiated by the sending of



special “X” packets, as opposed to file transfers which are initiated by “F” packets. The message logger window can be used to examine all such Kermit messages, both inbound and outbound.

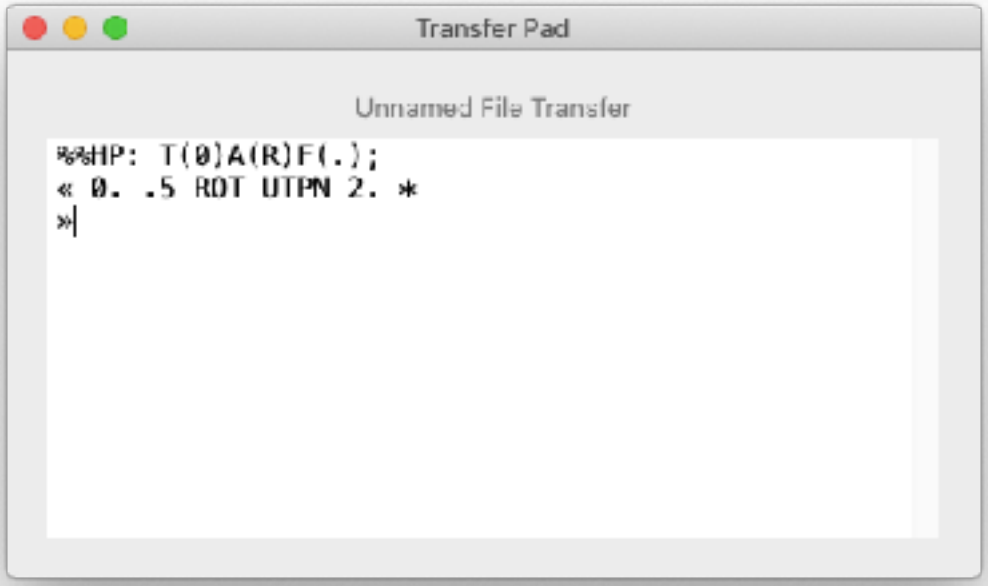
The Transfer Pad Window

The transfer pad window allows RPL objects to be transferred to and from the calculator interactively, without requiring them to be stored as files on the computer.

For example, a program found on a web page or in an online discussion forum can be copied and pasted into the transfer pad window, and from there transferred to the calculator stack by issuing the command

```
"" HGETO
```

on the calculator. This command performs an “unnamed” file transfer from the computer to the calculator and is from the Hoppi RPL library (described elsewhere in this document). Such unnamed transfers are interpreted by Hoppi as targeting the transfer pad window instead of a file.



Similarly, an unnamed transfer can be performed from the calculator to the computer, where it will be shown immediately in the transfer pad window. The command:-

```
"" HSNDO
```

from the Hoppi RPL Library achieves this, sending the object on the stack to the computer, as shown in the screenshot above.

The GROB Viewer Window

The GROB viewer window, accessible from the View menu, allows GROB objects to be displayed.

To send a GROB object from the calculator to the computer, it should be recalled to the stack then sent to the computer as an unnamed file transfer (see the Hoppi RPL Library):-

```
"" HSNDO
```

this sends the GROB to the transfer pad window, where it will be recognised by Hoppi and displayed in the viewer window too. See the previous section for details of unnamed transfers in Hoppi.

The image seen in the viewer window can be pasted into other macOS applications by selecting Copy from the Edit menu and then pasting into the other application as normal. In particular, the Preview application features a “New from Clipboard” feature in the File menu which can be used to capture the copied image. Then Preview’s export capabilities allow the image to be saved to the file format of your choice.

Transferring Files

The calculator's standard SEND, RECV and KGET commands can be used to transfer calculator objects to and from the computer, as they can with any Kermit server.

For example, to send a calculator object stored in global variable PGM1 to the computer, issue the command:-

```
'PGM1' SEND
```

on the calculator. And to transfer it back again, storing it in the current calculator directory, do:-

```
'PGM1' KGET
```

or possibly instead

```
RECV
```

followed by invoking “Send File…" on the computer (from the File menu) and choosing the “PGM1” file in the current workspace folder.

Alternatively, these calculator commands can be invoked in the FILER on an HP49G+/HP50G, or in the “Transfer” program on an HP48 calculator (accessible via right-shift I/O on the number 1 key). The latter can also be used to manage the computer's current workspace folder completely from the calculator. Note however that its equivalent on the HP49G+/HP50G has problems with these extended features, and instead the Hoppi RPL Library can be used.

In addition to transferring calculator objects Hoppi can be used to transfer the contents of arbitrary text files to and from the calculator, where they will appear as string objects. To send the contents of a text file to the calculator select the "Send File as String..." function from the File menu, choose the appropriate file, then issue the RECV command on the calculator. This will create a variable in the current calculator directory (named according to the file name) holding a string object corresponding to the contents of the text file.

To send the contents of a string object to the computer as a text file, select the "Receive File as String" function from the File menu then issue the SEND command on the calculator. This will write the contents of the string to a file on the computer, the file being named according to the name of the sent variable on the calculator.

Note that only the very next transfer will be treated as a "text file transfer” — subsequent transfers after this will revert to standard calculator object

transfers. This can be seen by noting the "Receive File as String" menu item will become ticked after first being selected, then become unticked when the transfer has taken place. To cancel a pending text file transfer simply select the "Receive File as String" menu item again and it will become unticked.

This text file transfer can also be achieved, under control of the calculator, using the HGETS command from the Hoppi RPL Library. The reverse process, sending the contents of a string object to a text file on the computer, can be accomplished using the complementary HSNDS command from the library.

Character Coding

The HP48/49/50 calculators use a character encoding that is different to that used by the computer. Although they share the majority of character codes, some of the more unusual symbols on the calculator must be mapped to and from appropriate Unicode equivalents when transfers are taking place. The following table shows the mapping used by Hoppi.

Calculator Code	Trigram	Symbol	Unicode Code	Unicode Name
31		…	0x2026	HORIZONTAL ELLIPSIS
127		▒	0x2592	MEDIUM SHADE
128	\<)	∠	0x2220	ANGLE
129	\x-	ā	0x0101	LATIN SMALL LETTER A WITH MACRON
130	\.V	∇	0x2207	NABLA
131	\v/	√	0x221A	SQUARE ROOT
132	\.S	∫	0x222B	INTEGRAL
133	\GS	Σ	0x03A3	GREEK CAPITAL LETTER SIGMA
134	\\|>	▶	0x25B6	BLACK RIGHT-POINTING TRIANGLE
135	\pi	π	0x03C0	GREEK SMALL LETTER PI
136	\.d	∂	0x2202	PARTIAL DIFFERENTIAL
137	\<=	≤	0x2264	LESS-THAN OR EQUAL TO
138	\>=	≥	0x2265	GREATER-THAN OR EQUAL TO
139	\=/	≠	0x2260	NOT EQUAL TO
140	\Ga	α	0x03B1	GREEK SMALL LETTER ALPHA
141	\->	→	0x2192	RIGHTWARDS ARROW

Calculator Code	Trigram	Symbol	Unicode Code	Unicode Name
142	\<-	←	0x2190	LEFTWARDS ARROW
143	|v	↓	0x2193	DOWNWARDS ARROW
144	|^	↑	0x2191	UPWARDS ARROW
145	\Gg	γ	0x03B3	GREEK SMALL LETTER GAMMA
146	\Gd	δ	0x03B4	GREEK SMALL LETTER DELTA
147	\Ge	ε	0x03B5	GREEK SMALL LETTER EPSILON
148	\Gn	η	0x03B7	GREEK SMALL LETTER ETA
149	\Gh	θ	0x03B8	GREEK SMALL LETTER THETA
150	\Gl	λ	0x03BB	GREEK SMALL LETTER LAMBDA
151	\Gr	ρ	0x03C1	GREEK SMALL LETTER RHO
152	\Gs	σ	0x03C3	GREEK SMALL LETTER SIGMA
153	\Gt	τ	0x03C4	GREEK SMALL LETTER TAU
154	\Gw	ω	0x03C9	GREEK SMALL LETTER OMEGA
155	\GD	Δ	0x0394	GREEK CAPITAL LETTER DELTA
156	\PI	Π	0x03A0	GREEK CAPITAL LETTER PI
157	\GW	Ω	0x03A9	GREEK CAPTIAL LETTER OMEGA
158	\[]	▪	0x25A0	BLACK SQUARE
159	\oo	∞	0x221E	INFINITY
160	\160		0x00A0	NON-BREAKING SPACE
171	\<<	«	0x00AB	LEFT-POINTING DOUBLE ANGLE QUOTATION MARK
176	\^o	°	0x00B0	DEGREE SIGN
181	\Gm	µ	0x00B5	MICRO SIGN
187	\>>	»	0x00BB	RIGHT-POINTING DOUBLE ANGLE QUOTATION MARK
215	\.x	×	0x00D7	MULTIPLICATION SIGN
216	\O/	Ø	0x00D8	LATIN CAPITAL LETTER O WITH STROKE
223	\Gb	ß	0x00DF	LATIN SMALL LETTER SHARP S

Calculator Code	Trigram	Symbol	Unicode Code	Unicode Name
247	\:-	÷	0x00F7	DIVISION SIGN

The above table is compatible with the mapping described by Chris Dreher at www.hpmuseum.org/cgi-sys/cgiwrap/hpmuseum/articles.cgi?read=1218.

Obtaining Hoppi

Hoppi is an open-source project, its source code and a collection of installable binaries are freely available from http://cloudycat.com/hoppi, just follow the links.

To install Hoppi simply copy it do your Applications folder. Note that Hoppi is an unsigned app, so to run it for the first time you must control-click on its icon in the Finder and press the “Open” button.

Version History

Hoppi 2.1

- Added the Transfer Pad window.
- Fixed a bug in the Unicode translation that could cause the application to crash.
- Runs on macOS 10.15 (Catalina) onwards.

Hoppi 2.0

- A complete rewrite of Hoppi 1.4 to the Swift language and using Apple’s Xcode development environment. This was done in order to ensure the continued survival of Hoppi going into the future.
- Functionally equivalent to Hoppi 1.4 (hopefully).
- Runs on macOS 10.15 (Catalina) onwards.

Hoppi 1.4

- Now builds on CCL 1.6 again, so support for PowerPC Macs is back.
- Hoppi-side transfer timeout is extended to 20 seconds to ensure very large objects can be transferred to the calculator (the kermit implementation on HP48 calculators slows down as the transfer progresses).
- Updating in the GUI of the number of bytes sent is now performed on a per-packet basis, instead of in 1024 byte bursts.

- Several different binary releases are available:-
    - Hoppi-1.4-Catalina.dmg — for use on macOS 10.15 (Catalina)
    - Hoppi-1.4-Mojave.dmg — for use on macOS 10.14 (Mojave)
    - Hoppi-1.4-Intel.dmg — for macOS 10.13 and earlier Intel Macs.
    - Hoppi-1.4-PPC.dmg — for use on OSX 10.5 PowerPC Macs.

Hoppi 1.3.4

- Now builds with Clozure CL 1.11 and so works on macOS Sierra (10.12). Will not build with versions of CCL below 1.11, so will no longer build on PowerPC Macs.

Hoppi 1.3.3

- Fixed handling of Kermit "F" generic command (G packet, F command).

Hoppi 1.3.2

- Fixed handling of file and folder names containing a semicolon.

Hoppi 1.3.1

- Corrected location of libhoppi-io.dylib in the application bundle. Now works on OSX 10.10.3.

Hoppi 1.3

- Added the message logger window.
- Added the GROB viewer window.

Hoppi 1.2

- Added the ability to send and receive text files as calculator strings.
- Included the Hoppi RPL Library version 0.4.
- Improved documentation.

Hoppi 1.1

- Now automatically re-opens last successfully opened serial device.
- Added support for Kermit's generic C and D commands.
- Added HRCD and HRLD UserRPL utility commands.
- Corrected handling of single-character directory names.

Hoppi 1.0

- Initial stable version.
- Created with Clozure Common Lisp (www.clozure.com/ccl).

Hoppi RPL Library

Introduction

The Hoppi RPL library is written primarily for HP49G+/HP50G calculators to facilitate reliable communication with the Hoppi Mac application. The motivation for writing this library originally stemmed from the unreliability of my HP49G+'s built-in “Transfer…” program (accessible via APPS → I/O functions → Transfer) when choosing computer folders and listing files — it would often generate nonsensical Kermit packets. Hence this library was born, with features to control and examine the Hoppi Mac application program state.

The library is written in UserRPL and its contents can also be installed on HP48G calculators, but here there is less need for the library since the built-in “Transfer…" program works reliably with Hoppi.

Library commands are as follows (one row per menu page):-

```
HOPPI HPAR  HGETO HGETS HSNDO HSNDS

HRCV  HRCD  HRLD  HROVR HGCMD
```

with the following one-line descriptions:-

Menu Page 1

- HOPPI — a program for easy interactive use of the library commands
- HPAR — change some calculator communication settings
- HGETO — get an object from a computer file
- HGETS — get a string representing the contents of a computer text file
- HSNDO — send an object to a computer file
- HSNDS — send the contents of a string to a computer text file

Menu Page 2

- HRCV — receive object from computer

- HRCD — change computer's current workspace folder
- HRLD — list files in computer's current workspace folder
- HROVR — get/set computer “Overwrite Enable” switch
- HGCMD — invoke Kermit generic command

These commands are described in more detail in the following sections.

Building and Installing

The library is written in UserRPL, and its source code is distributed along with the Hoppi Mac application. It is created by transferring all the files to a calculator directory then running the CRLIB command from the HP50G's development library (library 256). Alternatively, it is also available as binary file “hoppi.50g”.

Install this library into any port then do an ON+C restart to make its commands available.

Library Commands

The following subsections document each of the commands available in the Hoppi RPL Library.

HOPPI — an application for interactive file I/O

A simple (some might say crude) means of using the library commands interactively. Executing this command sets up a 3 page soft menu:-

```
FAV   PAR   CWD   OVR  DIRS  SELCT

GETV  GETO  GETS  RCV  FILES SELCT

SNDV  SNDO  SNDS       VARS  SELCT
```

where it is intended that these softmenu keys are to be used with the normal stack-based operation of the calculator. A one-line description of each softkey is:-

Menu Page 1

- FAV — recall a frequently used object to the stack from a custom list (or with left-shift edit the custom list, see below)
- PAR — change some calculator communication settings
- CWD — retrieve the current working directory (or with left-shift set it)

- OVR — retrieve the current "Allow Overwrite" setting (or with left-shift set it)
- DIRS — retrieve a list of remote subdirectories
- SELCT — select an item from a list (or with right-shift show a text-mode view of a long string or list of strings)

Menu Page 2

- GETV — get a variable from a remote file
- GETO — get an object (onto the stack) from a remote file
- GETS — get the contents of a remote file as a string on the stack
- RCV — receive an object (onto the stack) from a remote file
- FILES — retrieve a list of remote files
- SELCT — as menu page 1

Menu Page 3

- SNDV — send a variable to a remote file
- SNDO — send an object (from the stack) to a remote file
- SNDS — send a string on the stack to become the contents of a remote file
- VARS — get the list of variables of the current calculator directory
- SELCT — as menu page 1

where it should be noted that the PAR, GETO, GETS, RCV, SNDO & SNDS softkeys simply invoke the HPAR, HGETO, HGETS, HRCV, HSNDO & HSNDS library commands directly (the ``H'' prefix being removed to aid readability in the cramped soft menu). Also, the GETV and SNDV menu keys are aliases for the KGET and SEND commands, respectively.

In lieu of some proper documentation of these softmenu commands, a typical session using the Hoppi-specific commands may go as follows:-

1. Press CWD, resulting in a string representing the current remote directory appearing on the stack. It's too long to see properly, so press RS-SELCT to view it in text SCROLL mode. Press DIRS, getting a list of the remote subdirectories available. They're difficult to read so Press RS-SELCT to see them printed one-entry-per-line.

2. Press SELCT to choose one of these subdirectories. Press LS-CWD to change to it.

3. Press FILES to get a list of the remote files available. Press SELCT to choose one of them. Press GETO to transfer the contents of the file to the calculator stack, where it is automatically converted to a calculator object. Or maybe press GETS to transfer the contents of the file so that it is received as a calculator string object.

4. Recall a calculator object to the stack. Enter a name to use as a remote filename (either a string or a symbol). Press SNDO to send it as a calculator object.

5. Recall a calculator string to the stack. Enter a name for the remote filename and press SNDS to transfer the contents of the string to the remote file.

6. Change some calculator communication parameter settings (binary or ASCII mode, ASCII translation level, local overwrite enable) by pressing PAR.

7. Recall a frequently used calculator object to the stack by pressing FAV. This runs a CHOOSE command on the {HOME Hoppi Fav} calculator object which should be a list of two-element lists, each containing a display name string and the useful object itself. For example, this list can contain frequently used remote directories, so that by pressing FAV, UP/DOWN... ENTER then LS-CWD you can be taken directly to your remote directory of choice. Or you might store programs here (e.g. for selective backups), to be recalled then executed by pressing EVAL.

As you may have noticed, the Hoppi commands are all stack-centric, that is, they get and return the transferred objects from and to the stack. This is in contrast to the built-in RPL commands which tend to work with global variables. Keep this in mind when reading the command documentation in the following paragraphs, where a $ and {} notation has been used to suggest string and list arguments, respectively.

HPAR — Hoppi Parameters (—>)

Opens a form to set the calculator communication parameters relevant to Hoppi-specific commands.

HGETO — Hoppi Get Object ($name —> obj)

Retrieve the calculator object previously stored in the remote file named $name and return it to the stack.

HGETS — Hoppi Get String ($name —> $contents)

Read the contents of the remote file named $name and return as a string to the stack.

HSNDO — Hoppi Send Object (obj $name —>)

Send the object to the computer to be saved as file named $name.

HSNDS — Hoppi Send String ($contents $name —>)

Send the string $contents to the computer so that its contents become the contents of the file named $name.

HRCV — Hoppi Receive (—> obj)

Accept a file sent by the Hoppi program on the computer and return the received object to the stack.

HRCD — Hoppi Remote Change Directory ($dir —> $currdir)

where $dir should be a string representing the desired new directory.

The response will be the absolute path of Hoppi's current workspace directory.

Examples:-

```
1.   "/Users/pdo/RPL/" HRCD
```

will cause Hoppi to change to the given absolute path, returning "/Users/pdo/RPL/" as a string.

```
2.   ".." HRCD
```

will then cause Hoppi to change directory and return a "/Users/pdo/" string.

```
3.   "RPL" HRCD
```

will change directory and return "/Users/pdo/RPL/".

```
4.   "." HRCD
```

will not change directory, but will return "/Users/pdo/RPL/". So this variant can always be used to return the path of the current workspace directory without causing any changes.

```
5.   "" HRCD
```

will change to the user's home directory, "/Users/pdo/" in this case.

HRLD — Hoppi Remote List Directory

($dir —> $dir {$subdirs} {$files})

where $dir should be a string representing the desired directory to list. It can be relative or absolute, where relative means relative to the current workspace directory.

The command returns three values: the absolute path of the listed directory, as a string, a list of subdirectory name strings and a list of file name strings.

Examples:-

```
1.   "/Users/pdo/" HRLD
```

lists the given directory.

```
2.   "RPL" HRLD
```

lists the RPL subdirectory of the current workspace directory.

```
3.   "" HRLD
```

lists the current workspace directory.

HROVR — Hoppi Remote Overwrite (x —> x')

where x is a real number. If x > 0 then turn on "Allow Overwrite" or if x = 0 then turn it off. If x < 0 then don't change it. In any case return the new setting (0 for off, 1 for on).

HGCMD — Hoppi Generic Command

($cmd $param —> $response)

($cmd {$params} —> $response)

where $cmd should be a string containing a single character, $param a string containing a command parameter, and {$params} a list of strings containing command parameters.

Invoke a Kermit generic command and return its response. The response from Hoppi is returned in the $response string.

Version History

Hoppi RPL Library 0.4

- Basic functionality working.

Coda

Finally, why is it called Hoppi? Take the capital letters from “HP Packet I/O" and rearrange them. Plus it sounds frog-like (for Kermit). Well, I didn't say it was profound...

paul.onions@acm.org