

Contact

Reference document

V1.0.0

**©1988-1990 Todd A. Hitt
1447A Cliff Court, Columbus, Ohio 43204**

1. Introduction	1
2. Contact demonstration stack.....	2
2.1. Terminal emulator.....	4
2.2. Setup cards.....	8
2.3. GEnie™ bulletin board read.....	12
2.4. Help.....	13
3. UNIX™ Browser stack.....	14
3.1. Configuration card.....	15
3.2. Login card.....	16
3.3. Working directory card.....	17
3.4. Copy card.....	19
3.5. Move card.....	20
3.6. Details card.....	21
3.7. Find card.....	22
3.8. Terminal emulator card.....	23
3.9. Compatibility.....	24
4. Contact programming examples.....	25
4.1. Starting & Stopping Contact.....	26
4.2. A terminal emulator card.....	27
4.3. Sending a UNIX™ command.....	29
5. Commands Reference.....	31
5.1. Capture.....	32
5.2. Keyboard.....	34
5.3. LookFor.....	36
5.4. Receive.....	37
5.5. Screen.....	39
5.6. Serial.....	40
5.7. Serve.....	46
5.8. Terminal.....	47
5.9. Transmit.....	53
5.10. Type.....	55
6. Unimplemented terminal emulation features.....	57
7. History.....	58
8. Registration.....	59
9. Support and upgrades.....	60
10. Trademarks and Warranty.....	61

1. Introduction

Contact is a set of XCMDs for Hypercard and a driver which allow Macintosh users to create front ends for remote computer systems. Hypertalk developers can create Hypercard stacks that interface directly to bulletin board systems (BBSs), commercial information services such as GENie™ and Compuserve, and other remote computer systems. These "Contact application stacks" can act as the user's interface to the remote system, providing intelligent guidance to the user without requiring modifications to the remote system.

Contact is not in the public-domain. You may evaluate Contact free for **thirty days** after acquiring it. If after that time, you decide to keep Contact, you must register your copy. **The registration fee for single users is US\$10.** If you do not wish to keep Contact, you must delete all copies. **See the Registration section at the end of this document .**

Contact requires Hypercard version 1.2 and a suitably equipped Macintosh. Two Megabytes of memory is highly recommended although Contact will function in one Megabyte. Contact has been tested with System 6.0 on a Mac Plus, Mac SE, and Mac II. It may not work with prior machines or prior versions of the operating system, as it uses code segments larger than 32k.

The Contact distribution kit comes with the following items:

- the Contact demonstration stack,
- the UNIX™ browser stack,
- this Contact document,
- the Contact • Readme file, which contains release notes,
- Contact resources, which describes the resources provided in the distribution kit, and
- CopyContact.r, an MPW Rez file to copy Contact's resources.

The Contact demonstration stack is a simple stack which contains a terminal emulator card, a GENie™ bulletin board read card, and various setup cards. It is intended to be duplicated and modified, providing the foundation for your Contact application stack.

The UNIX™ browser stack provides an example of a relatively complete Contact application stack. It provides a simplified front-end to UNIX™ computer systems, giving persons the ability to manipulate files on the UNIX™ computer system without having to know UNIX™.

Neither one of these stacks is copy protected. They are intended for you to copy and modify for your own purposes.

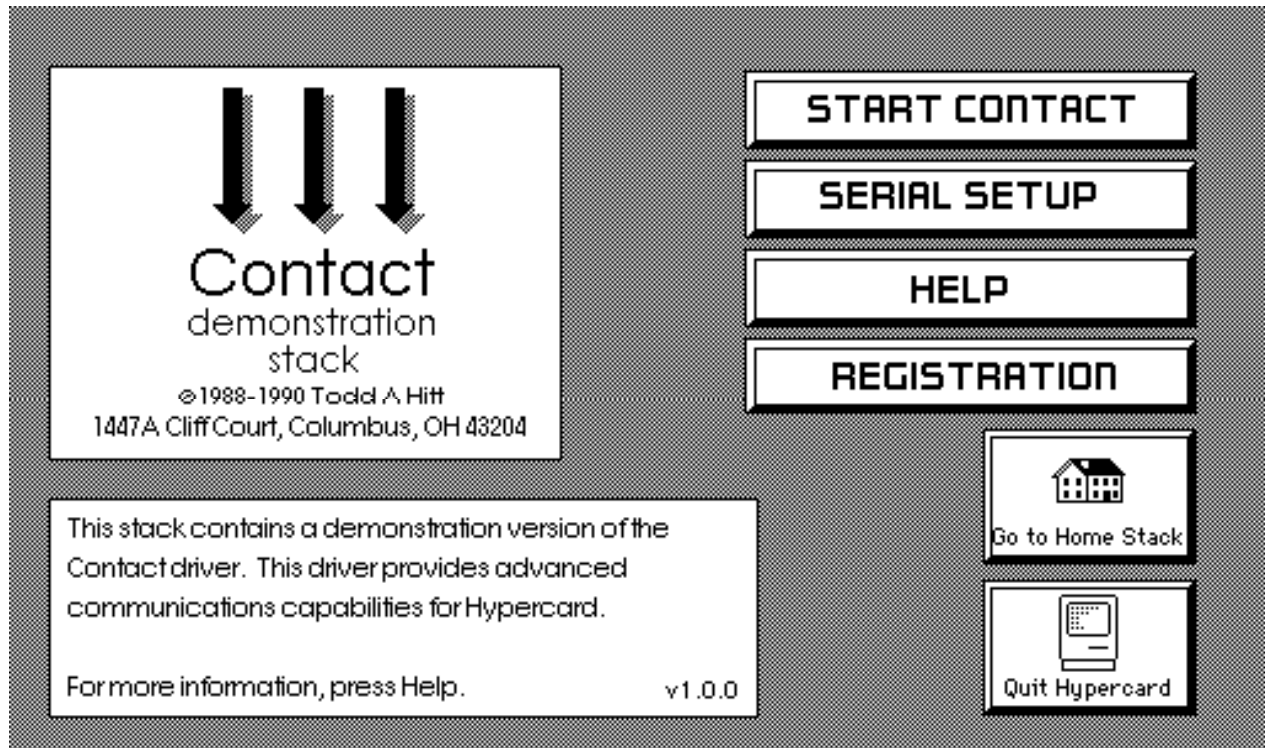
The stacks included in the Contact distribution kit were written with Contact version 1.0. When you register your copy, please make suggestions for new features. Some obvious ones are 132 column support and Xmodem file transfer. If I get a large enough response, these enhancements, or others, may become a reality.

This rest of this document discusses the provided stacks and how to use the Contact XCMDs.

2. Contact demonstration stack

The Contact demonstration cards contains a terminal emulator card, a GEnie™ bulletin board read card, and several setup cards. It demonstrates some of Contact's capabilities.

The figure shows the first card in the demonstration stack.



Press "Start Contact" to go on-line. Press "Help" for more help.

To go on-line with the remote system, use the mouse to press "Start Contact". This button starts the Contact driver, opens the serial driver, and goes to the Terminal emulator card.

You may want to change some serial parameters, such as baud rate, before you start. Use the mouse button to press "Serial Setup". This button will take you to the Serial Setup card where you can modify the serial parameters. When you are satisfied with the serial parameters, press the big return button. You will be returned to the first card where you can then press "Start Contact".



All of the cards in the Contact distribution stack, except the Help cards, contain a context-sensitive help line, near the bottom of the window. To find out what a button or text-field does, simply move the mouse over the button. The help line will change to a description of the item, if a description exists.

The bottom of all the cards in the Contact distribution stack contains a set of button that you can use to navigate around the stack. These buttons are:



Press this button to go to a help card describing this card.



Press this button to go to the terminal emulator card.



Press this button to go to a set of cards allowing you to change the serial and terminal emulator's parameters.



Press this button to go to the GEnie™ bulletin board read card.



Press this button to go to the previous card.



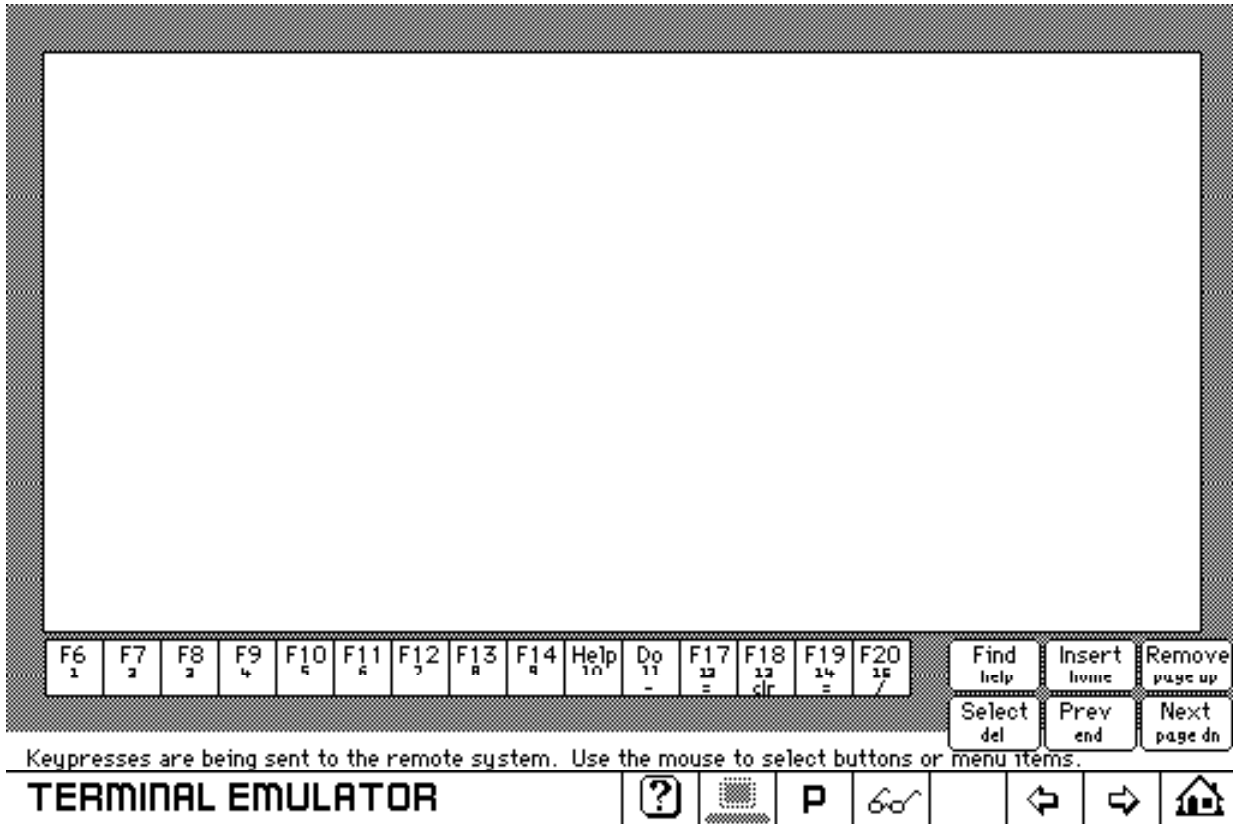
Press this button to go to the next card.



Press this button to go to your Home card.


2.1. Terminal emulator






One of the cards which comes in the Contact demonstration stack is a terminal emulator. The Contact driver emulates a VT220 terminal. This section describes how to use the terminal emulator card.




2.1.1. Keyboard

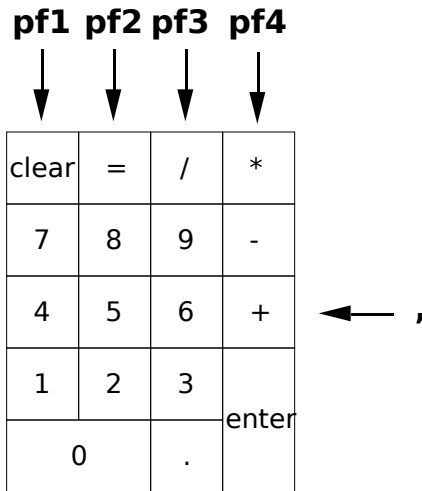
When this card is shown, and Hypercard's main window is the active window, keys that you press will be sent to the remote system. For Hypercard's main window to be active, you must hide any tool palette windows, the message box, and if you are not running Multifinder, you must close any desk accessories.

Sending control codes. If you have an ADB keyboard with a "control" key, hold the control key down while pressing a letter to send control codes. For example, to send an ASCII 3, hold down the control key and press the letter "c". Since you have control key, the  key can be used for menu short-cuts.

If you don't have an ADB keyboard with a control key, hold down the  key and press a letter. For example, to send ASCII 4, hold down the  key and press the letter "d". Any time this document refers to the control key, you must use your  key. Since you are using your  key for a control key, you won't be able to use your  key for menu short-cuts.

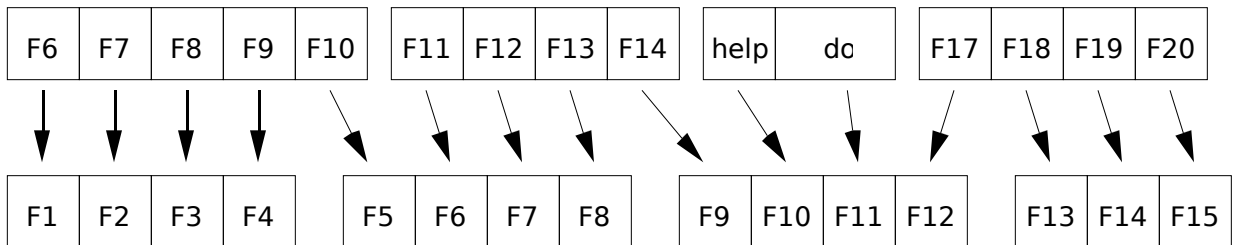
To send a NULL, ASCII 0 (sometimes call control-@), use control-space (remember some of you will have to use -space).

Using the keypad. The Macintosh keypad maps to the VT keypad positionally, not by name. Thus:



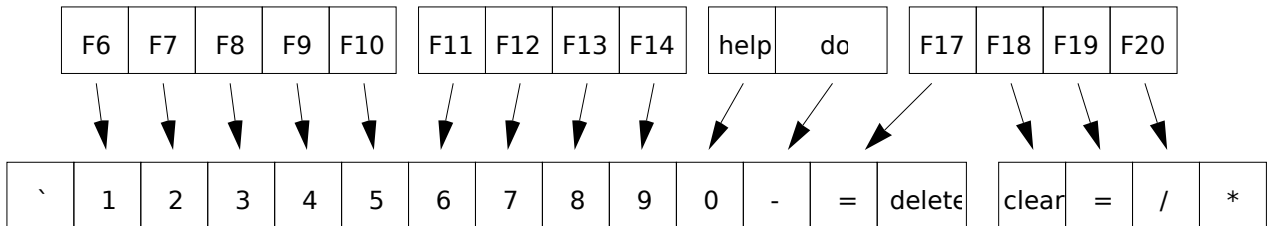
All of the other keys on the Macintosh keypad match the names on the VT keypad.

Sending function keys. Like the keypad, the function keys map positionally. If you have the extended ADB keyboard, you are in luck:



The top row shows the layout of the VT200 keyboard. The bottom row shows the layout of the Macintosh extended keyboard. Note that the spacing between keys is not the same as the VT220.

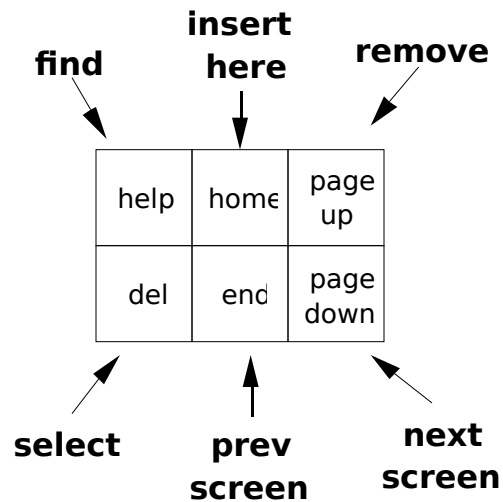
If you don't have an extended ADB keyboard, you can still send the function keys. Hold down the control key and press:



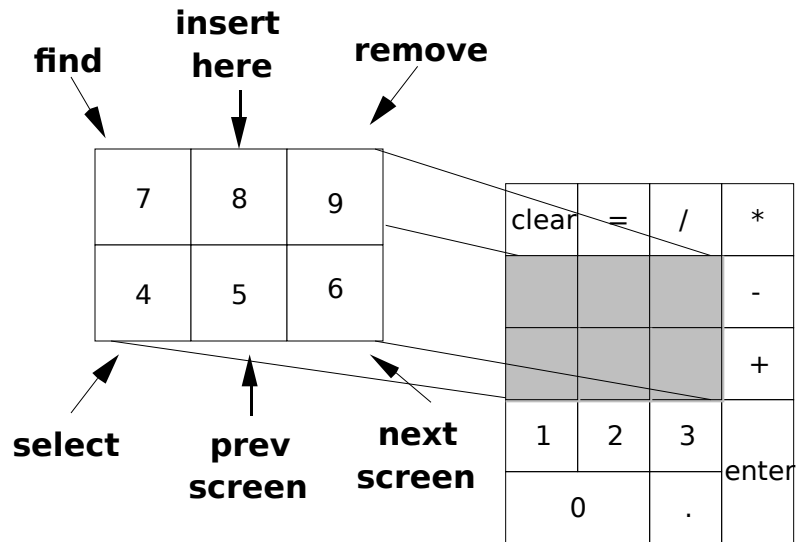
The top row shows the layout of the VT200 keyboard. The bottom row shows the top row of keys on the Macintosh keyboard. Note that the delete key is skipped.

Sending editing keypad keys.

If you have an extended ADB keyboard:



If you don't have an extended ADB keyboard, you can hold down your control key and press keys on the keypad:



2.1.2. Screen buttons

The terminal emulator card contains buttons on the screen that you can click on with the mouse to send function key sequences. One of these buttons look like:



This is the “Do” button. The top line tells you the name of the function key on the VT220 keyboard. The second line shows the name of the corresponding function key on the extended ADB keyboard, in this case F11. For function keys F6 through HELP, this also corresponds to the key you press with the control key on a different keyboard, such as the Macintosh Plus keyboard. For function keys DO through F20, the third line shows the corresponding control key, in this case control-minus.

2.1.3. Menu

The terminal emulator card adds a menu to the menubar. This menu contains six menuitems.


Refresh This menuitem causes the terminal emulator window to be redrawn. It does not cause Hypercard to update its portion of the screen.

Break This menuitem causes the terminal emulator to send the break signal to the remote system.

Disconnect This menuitem sets Data Terminal Ready (DTR) low for a short time and then back high. This is useful for modems which may use this as an “attention” signal.

Capture This menuitem causes Contact to capture text received from the remote system. The menuitem is checked when Contact is capturing. Select Capture to start the Capture. Select Capture again to halt the capture and the captured text will be placed on the Capture results card (which immediately follows the terminal emulator card).

Receive This menuitem causes Contact to store text received from the remote system into a text file. The menuitem is checked while Contact is storing text. Select Receive to start storing text. You will be asked for a file name to store the text in. Select Receive again to stop receiving text.

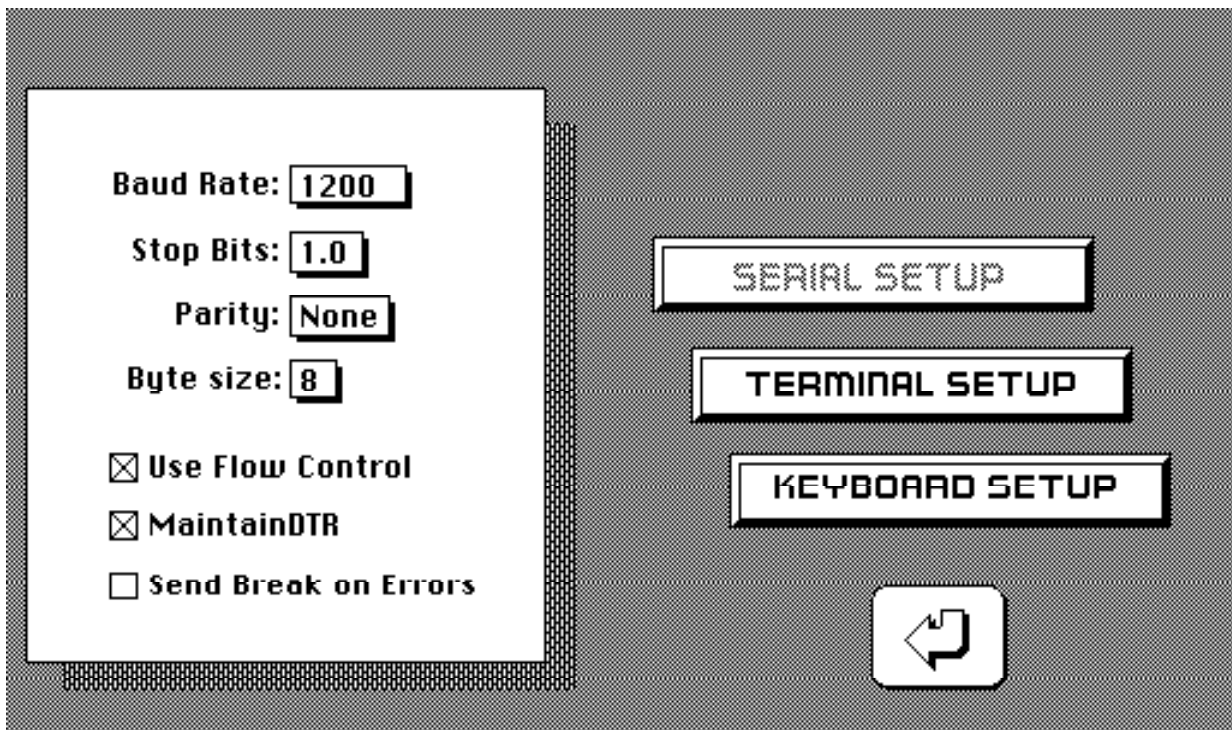
Transmit This menuitem causes Contact to send a text file to the remote system. You will be asked for the name of the text file to send. The contents of the file will be sent to the remote system. Press -period to stop the send.

2.2. Setup cards

The Contact demonstration stack contains three cards that allow users to setup different parameters in the stack and the terminal emulator.

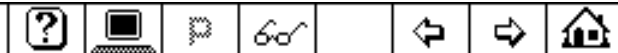
2.2.1. Serial Setup

When you press “Serial Setup” on the initial card or press the parameters button, you will be shown this setup card:



Press this button to go to the next card.

SERIAL SETUP



The three buttons labeled “Serial Setup”, “Terminal Setup”, and “Keyboard Setup” will move you between the three setup cards. The big return button return you to work.

This card allows you to change the serial port's parameters. You may change these parameters either before you start, or during operation.

Baud rate. This popup menu allows you to select the baud rate.

Stop bits. This popup menu allows you to select the number of stop bits.

Parity. This popup menu allows you to select the byte parity.

Byte size. This popup menu allows you to select the byte size.

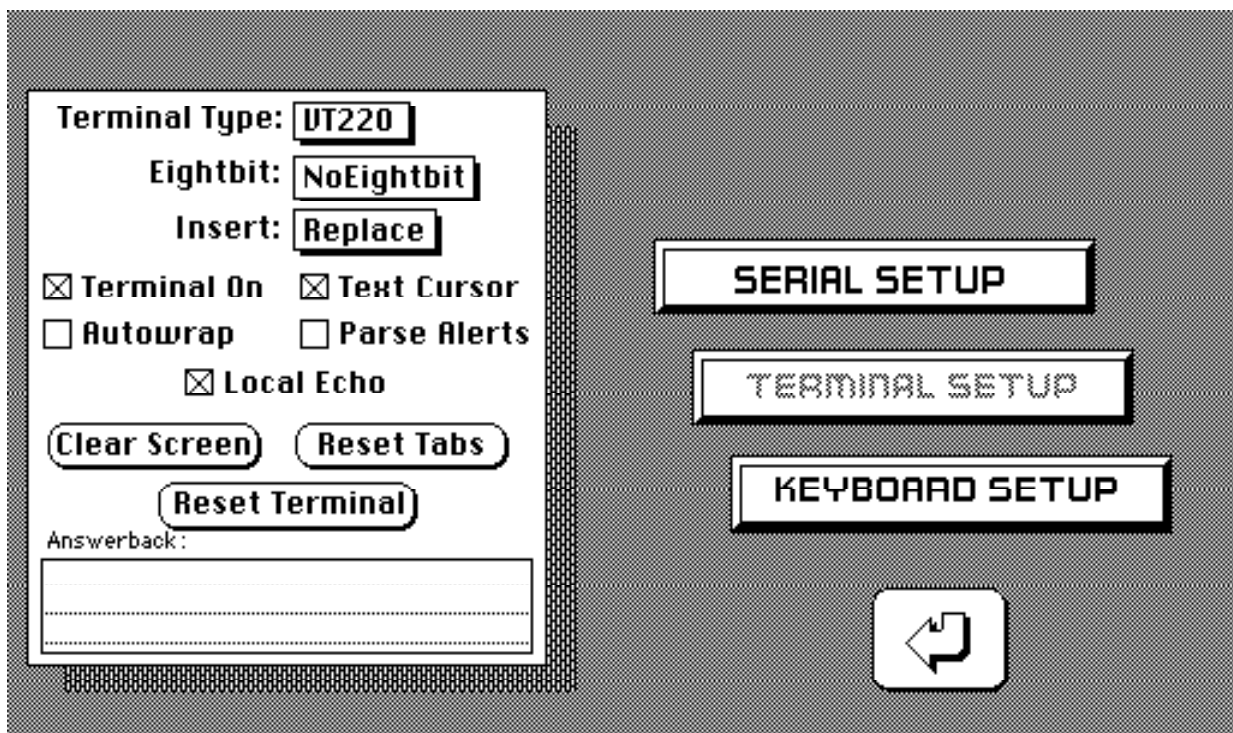
Use Flow Control. This checkbox controls whether Contact responds to Xons and Xoffs or not.

Maintain DTR. This checkbox causes Contact to maintain DTR in it's current state (asserted) when Contact is closed. This should have the effect of keeping the current connection established, allowing you to exit and reenter Contact without needing to reestablish the connection.

Break on Errors. If this checkbox item is selected, Contact sends a break when the Macintosh serial driver detects a framing error or an overrun error.

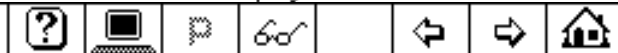
2.2.2. Terminal setup card

One of the setup cards is for terminal parameters:



This card allows you to change various parameters related to the terminal display.

TERMINAL SETUP



Terminal Type. This popup menu effects the ID string that is sent to the remote computer in response to Device Attributes request. It does not effect how Contact responds to any other escape sequences nor does it prevent you from using VT220 function keys in VT102 mode.

Eightbit. When this item is set to "Eightbit", Contact will send eight bit control codes for function keys and reports if, and only if, the terminal is in VT220 mode. You should also set the serial parameters to send eight bit bytes. Contact recognizes eight bit control codes it receives regardless of the setting of this item.

Insert/Replace. This item controls whether Contact inserts characters that are received or overstrikes existing characters.

Terminal On. This item controls whether the Contact terminal emulator is active. If it is not checked, then incoming serial characters will not be processed by the terminal emulator.

Text Cursor. This item controls whether the cursor is visible or not.

Autowrap. This item controls whether the cursor location wraps around to the first column when it goes past the last column.

Parse Alerts. When this item is checked, Contact will tell you when it receives an ANSI escape sequence it doesn't understand.

Local Echo. When this item is checked, Contact echoes characters sent to the remote system to the screen.

Answerback. Enter the text you want Contact to send to the response system in response to an Enquire (ASCII 05).

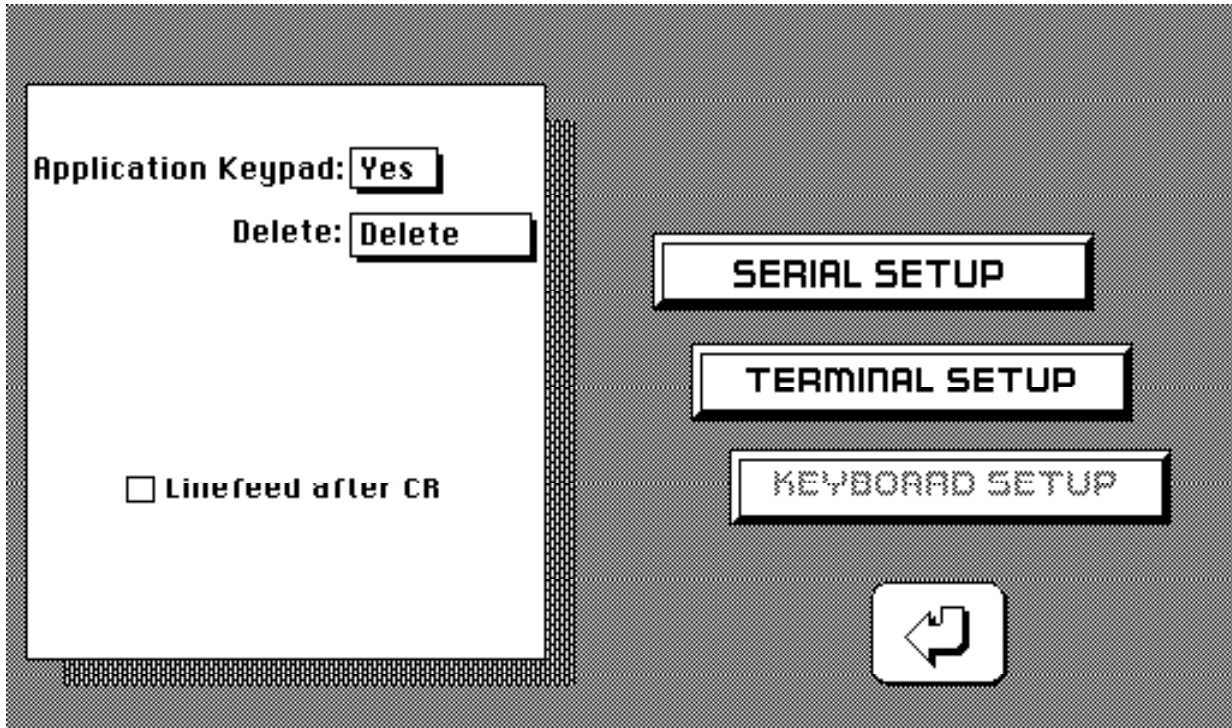
Clear screen. Clears the terminal's screen and resets the cursor position.

Reset tabs. Resets the tab stops to every eight characters.

Soft reset. Performs a soft terminal reset. This function resets the scroll region, turns the cursor on, selects overstrike mode, selects absolute origin mode, turns autowrap off, selects a numeric keypad, changes the cursor key mode to normal, resets the cursor save state, and resets the current video attributes.

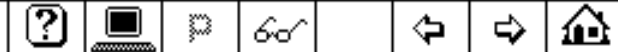
2.2.3. Keyboard setup card

Another one of the setup cards is to configure the keyboard.



This card allows you to change the keyboard parameters.

KEYBOARD SETUP



Application keypad. Controls whether the numeric keypad sends numbers or application escape sequences.

Delete. Controls whether the Macintosh's delete key sends a delete character (ASCII 127) or a backspace character (ASCII 8).

Linefeed after CR. When checked, Contact will send a linefeed after carriage returns.

2.3. GENie™ bulletin board read

The GENie™ bulletin board read card allows you to format and send a GENie™ bulletin board READ command. To use this card, you must be logged into a GENie™ roundtable displaying the bulletin board prompt. When full prompts are being displayed, this prompt will look like:

```
1 CATegories          10 INDEx of topics
2 NEW messages        11 SEARCh topics
3 SET category        12 DELEte messages
4 DEScribe category   13 IGNore category
5 TOPic list          14 PROMpt setting
6 BROWse new msgs     15 SCROll setting
7 REAd a message      16 NAME used in BB
8 REPlY to a topic    17 EXIt
9 STArt a topic       18 HELp
Enter #, <COMmand> or <HEL>p
1 ?
```

When you see this prompt on the terminal emulator, you may press the glasses button which will bring you to this card.

This card demonstrates how you might use Contact to put a Macintosh-like front end on a remote application.

This card formats a READ command for a GENie roundtable bulletin board. It then sends the command and goes back to the terminal card.

To use this, you must be in a GENie bulletin board.

Do It!

Cancel

Topics to read from- Read from all topics, a single topic #, or a range of topic #'s.

All or to

Messages to read:

All List all messages in topic(s)

New List only new messages in topic(s)

Last List only last message in topic(s)

Author List only messages entered by authors with name that starts with...

<

Date = List only messages with date that is...

> Format = YYMMDD

List options:

List author only Don't list the text of the message.

Enter a search specification for the messages you want to read from a GENie bulletin board.

GENIE ROUNDTABLE READ



P

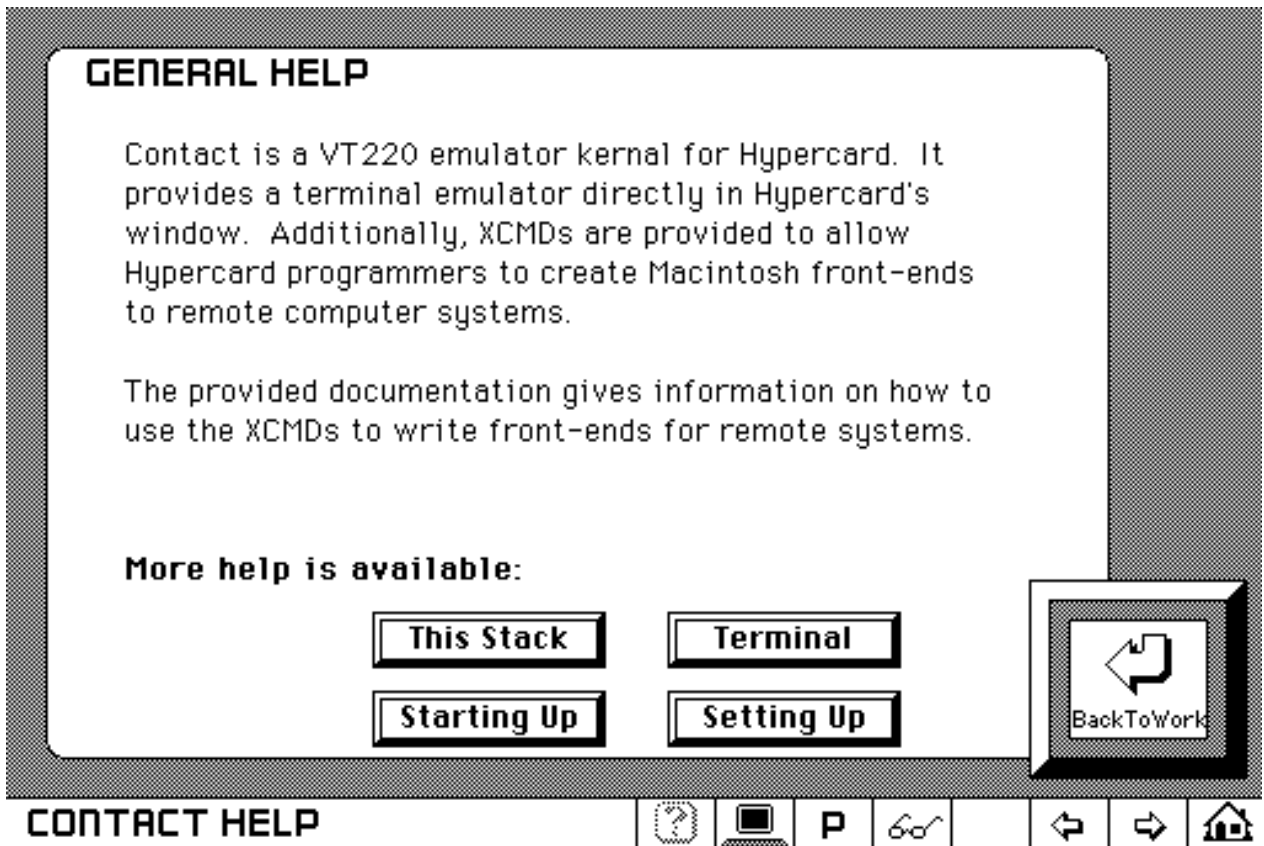


When you have entered the information you desire, press the “Do It!” button. The stack will send the READ command and then switch back to the terminal emulator card, where you can see the messages.

This card does not do any field validation. So make sure you enter the date and topic numbers in the correct format.

2.4. Help

Pressing the help button will bring you to one of the help displays. This is what the general help display looks like:

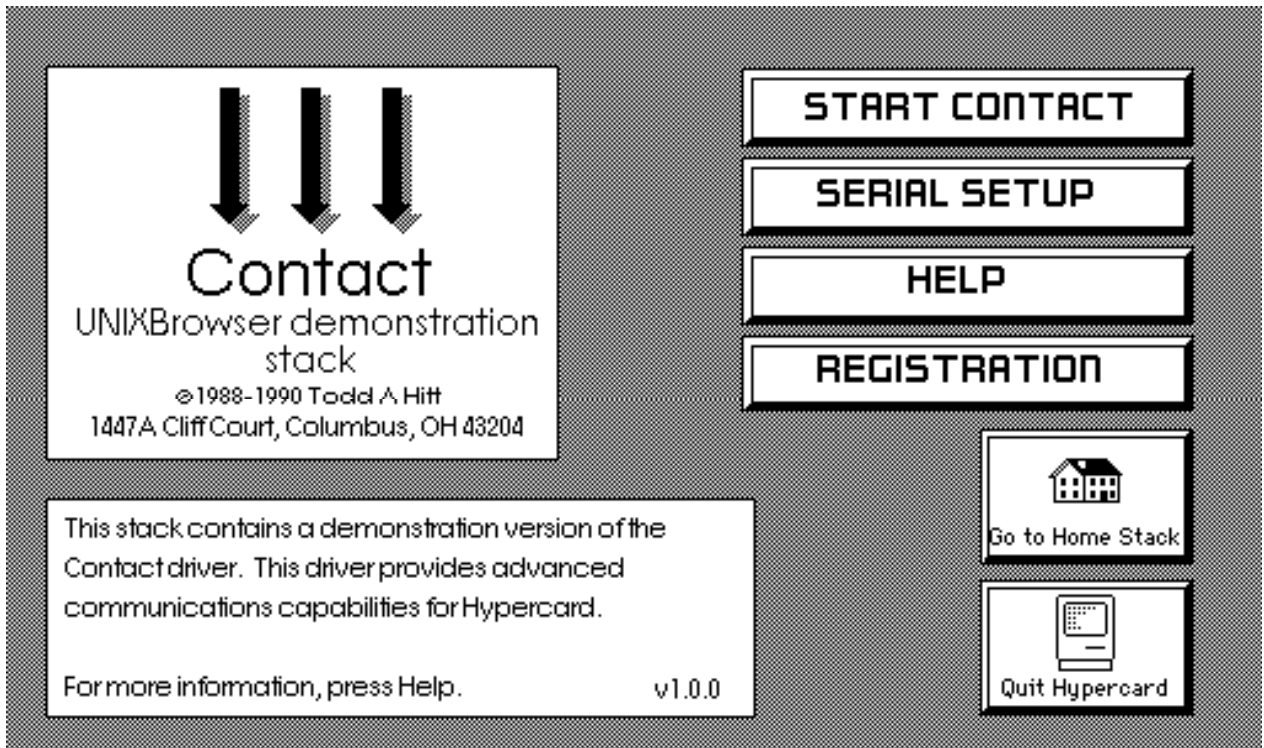


You can press any of the buttons at the bottom to display more help information. Pressing the “back to work” button returns you to the card you left.

3. UNIX™ Browser stack

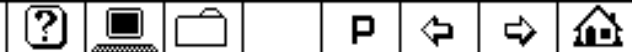
The UNIX™ browser has been tested with Sun™-OS.

The figure shows the first card in the stack. This card functions similarly to the first card in Contact demonstration stack.



Press "Start Contact" to go on-line. Press "Help" for more help.

WELCOME TO CONTACT



To go on-line with the remote system, use the mouse to press "Start Contact". This button starts the Contact driver, opens the serial driver, and goes to the login card.

3.1. Configuration card

The configuration card allows you to setup several parameters to control your UNIX™ session.

This card allows you to setup how the UNIX browser works with your host system.

Prompt:

Home dir:

Phone #:

Hayes Modem
 Direct Connect
 Autobaud Host

Prompt must be the first part of your system's prompt.

Home dir should be your home directory.

Phone # is the number to dial (if connected through a Hayes compatible modem).

Autobaud host tells the stack to wait for the host computer to select the right baud rate.

TEST CONTROL



You must put the first part of your system's prompt in the prompt field. This portion of the prompt must not change. For example, UNIX™ prompts often show the command number. As more commands are entered, this number increases. The first part of the prompt that is entered above must not have this number in it.

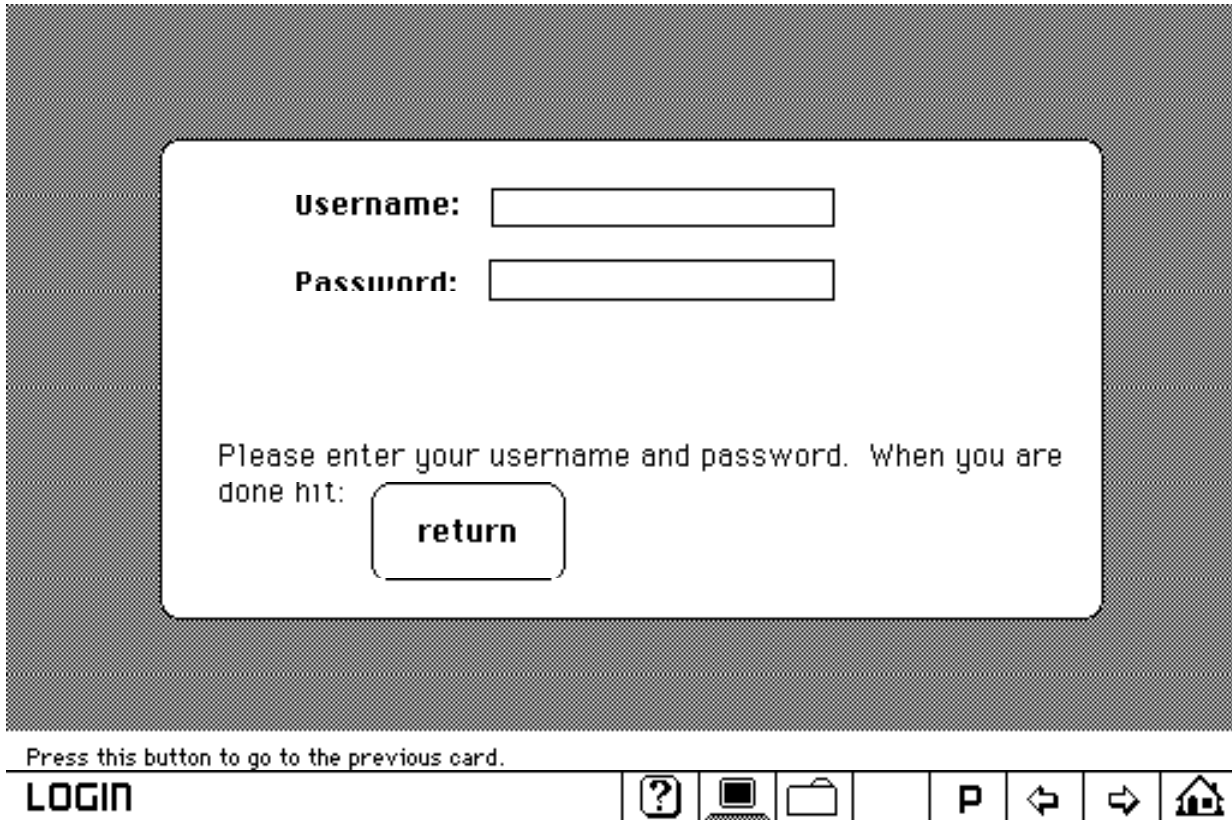
The home directory field contains the name of your home directory. You may use tilde (~).

If your Macintosh is connected to a Hayes™ compatible modem, you may put your UNIX™ system's telephone number in the phone # field and select the Hayes™ modem radio button. When you log in from the login card, the phone # will first be dialed.

If autobaud host is checked, then the UNIXBrowser stack will repeatedly send characters to the remote system until it echoes correctly. Once the correct baud rate has been determined, then the login will proceed.

3.2. Login card

The login card looks like this:








Username:

Password:

Please enter your username and password. When you are done hit:

return

Press this button to go to the previous card.

LOGIN | ? |  |  | P |  |  | 

Enter your username and password in the fields and then press return.

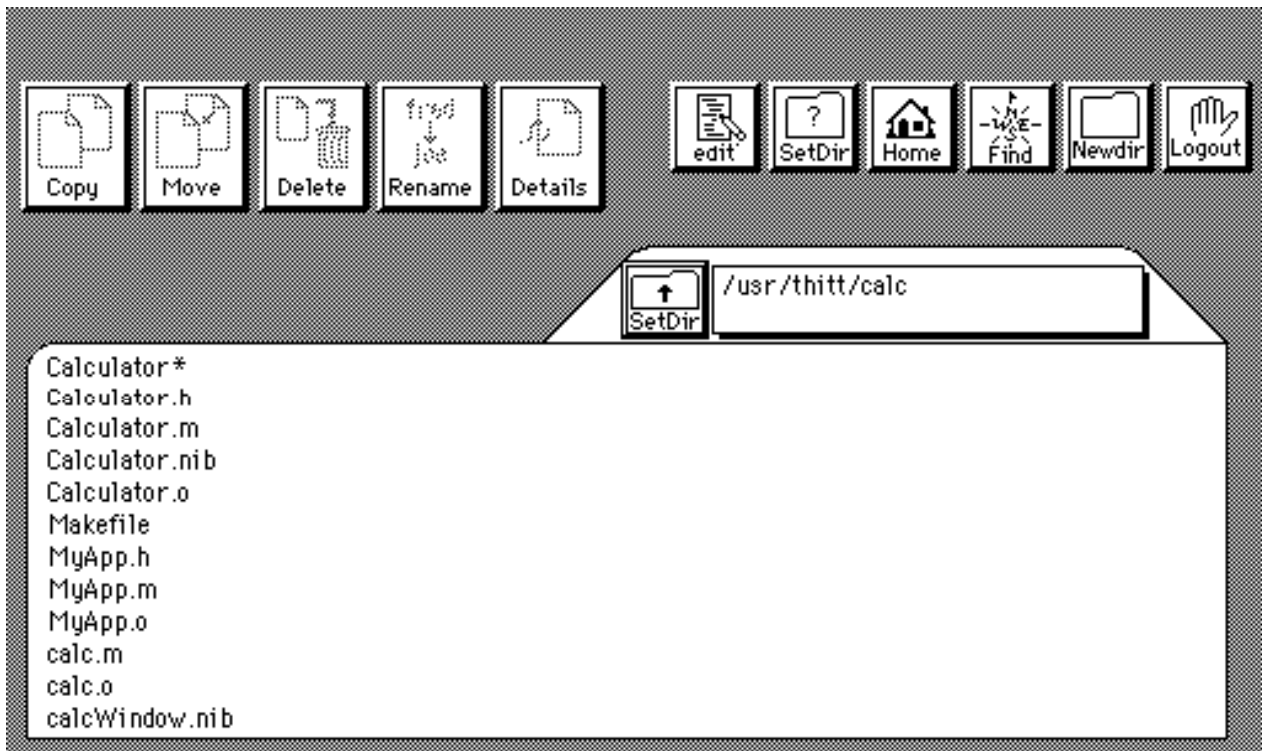
If you have specified Hayes™ modem on the setup card, Contact will dial the remote system. After that, if you have specified autobauding on the setup card, the Macintosh will perform autobauding.

The Macintosh will then attempt to log you in to the remote system.

After login, if your UNIX™ system requires any further input, such as setting the terminal type, you will need to perform that manually from the terminal emulator card.

3.3. Working directory card

The working directory card is your main work place. This card serves the same purpose as the Macintosh Finder.



Click on a file to select it. Click twice to open. Shift-click to select multiple files. Click & hold down for menu.

The center part of the display shows the files in your current display. Applications are displayed with an asterisk after them. Directories are displayed with a slash after them.

Selecting files Point and click on a file to select it. Hold down the shift key while clicking to select more than one file. Once you have selected a file, the buttons on the upper left of the card will be enabled, allowing you to copy files, etc.

Opening a file You can double-click on a file (don't hold down the shift key) to open the file. Alternately, you can hold down the mouse button on a file (again no shift key) to pop up a menu, giving you options for opening the file. Opening a text file will run emacs allowing you to edit the file. Opening an application runs that application. Opening a directory file changes the working directory to that directory.

Changing the working directory There are several ways to change your current working directory.

You can open a directory file to move down the hierarchy into that directory.

You can move up the directory hierarchy by clicking on the set directory up button. If you simply click, you will be moved up one level in the hierarchy. If you hold down the mouse button, a menu will pop up allowing you to move up more than one level at a time.



You can enter your working directory specification directly by clicking on the set directory button with the question mark. You will be presented with a dialog box in which you can enter your new directory specification.



You can also go to your home directory by pressing the home button.

Copying files Select one or more files and press the copy button. If you selected one file, you will be shown a dialog box where you can enter the name of the new file. A copy of the file will be made in your current directory. If you selected multiple files, you will be shown the copy card where you must indicate the destination directory. All of the files will be copied to the directory you indicate. See the section entitled **Copy card**.

Moving files Select one or more files and press the move button. You will be shown the move card where you must indicate the destination directory. All of the files you selected will be moved to the directory you indicate. See the section entitled **Move card**.

Deleting files Select one or more files and press the delete button. You can only delete a directory if it is empty.

Renaming a file Select a single file and press the rename button. You will be shown a dialog box where you can enter the file's new name.

File details Select a single file and press the details button. You will be shown the **Details card** where you can see information on the file and change its protection if you want.

Editing/Creating a file To edit an existing text file select the file and then press the edit button. To create a new text file, press the edit button without selecting any files.

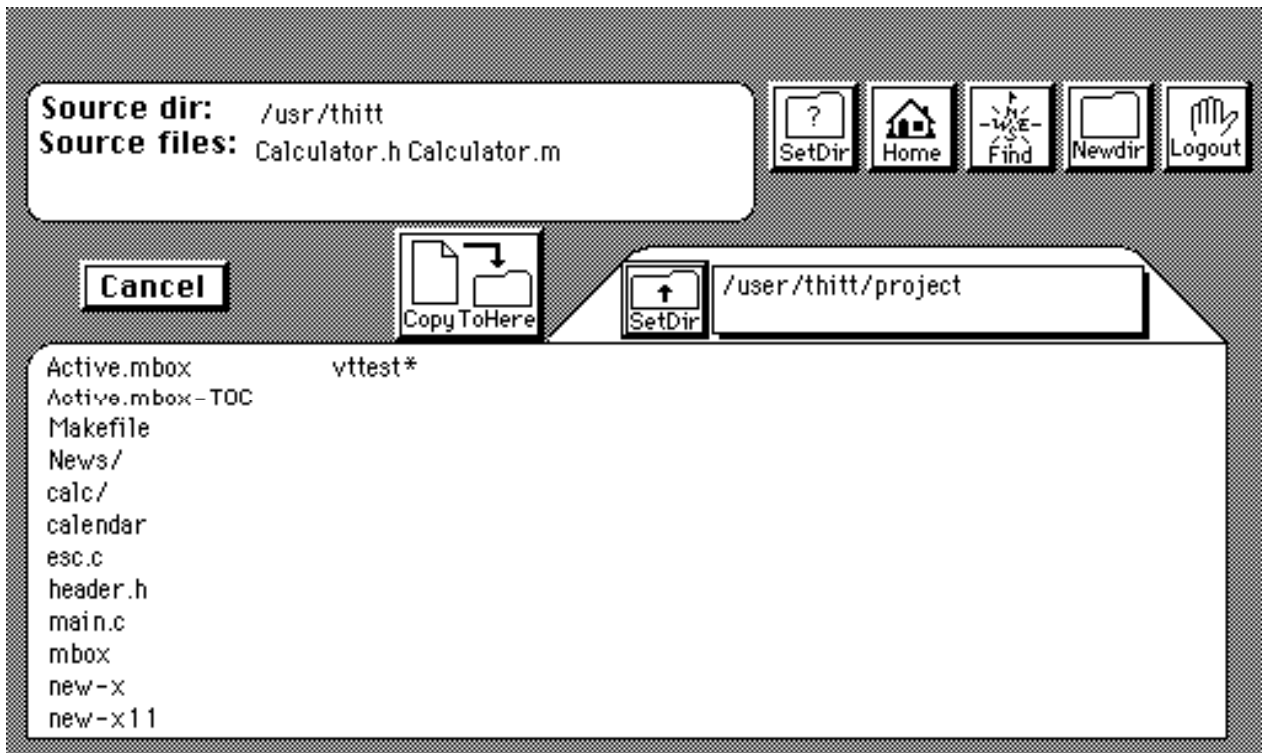
Finding a file Press the find button. You will be shown the find card where you can search for files.

Creating new directories Press the create directory button. You will be asked for the name for the new directory. The new directory will be placed in the current directory.

Logout Press the logout button. You will be logged out and then shown the login card, where you can log back in.

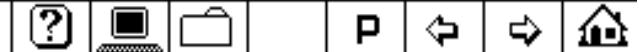
3.4. Copy card

When you select more than one file and press copy on the working directory card, you will be shown the copy card.



When you have the directory displayed that you want to copy the files into - press "CopyToHere".

COPY FILES

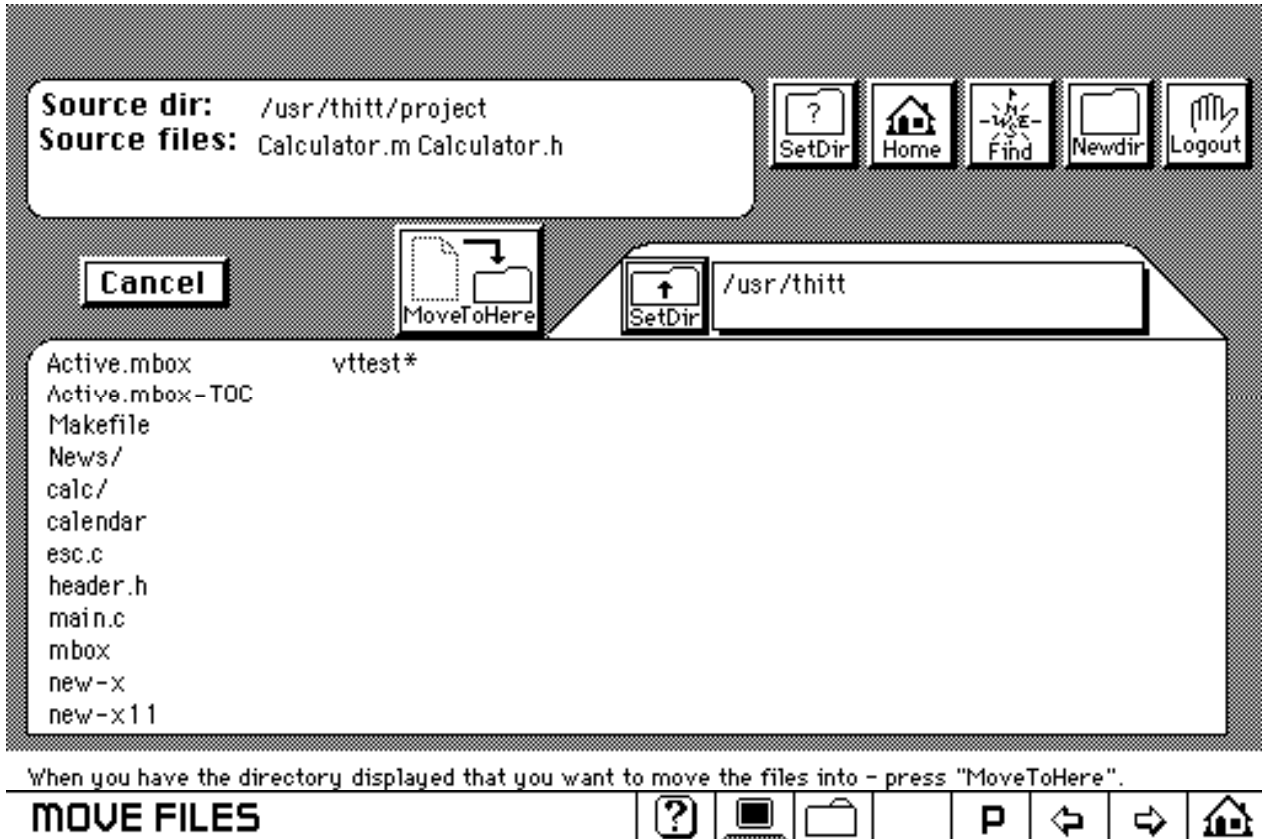


You must specify the destination directory for the duplicate files. Display the directory you want to copy the files into and then press Copy To Here. The files will be copied into the directory you are looking at. You can also select a directory file and then press Copy To Here. The files will be placed in the selected directory.

The buttons on the upper right of the card function the same as they do on the working directory display. This will allow you to move around in the directory hierarchy.

3.5. Move card

When you select files on the working directory display and press move, you will be shown this card:

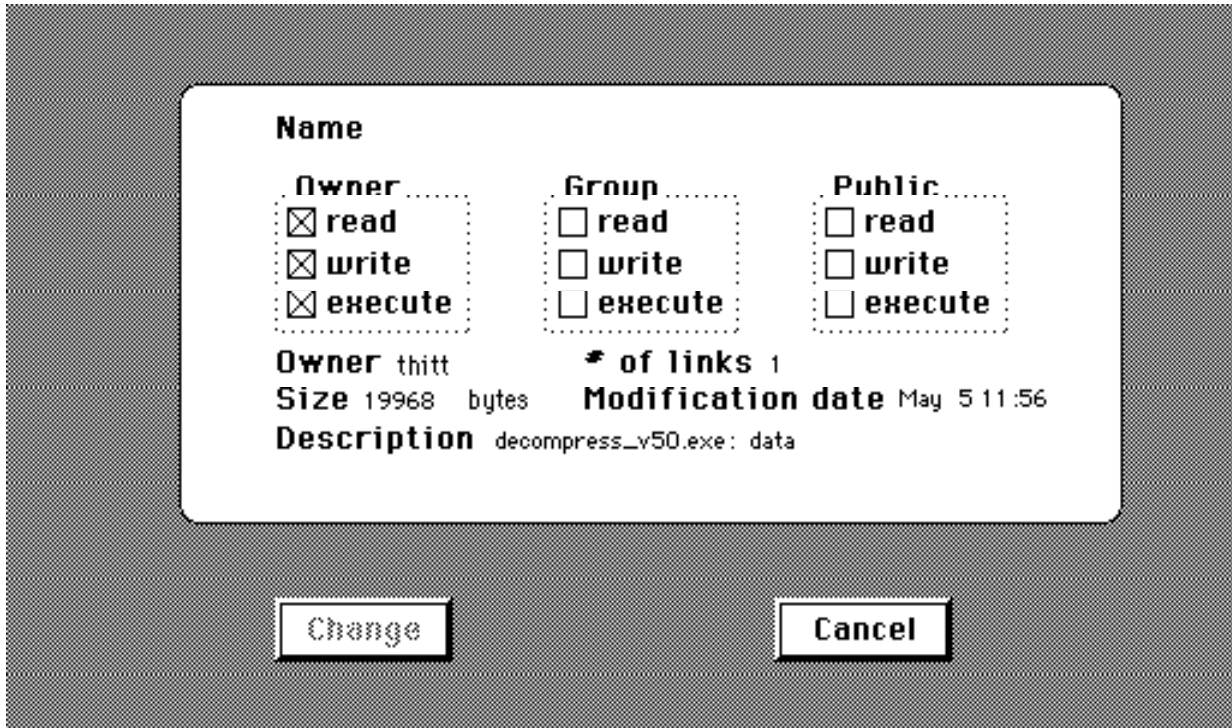


You must specify the destination directory for the files. Display the directory and then press Move To Here. The files will be moved into the directory you are looking at. You can also select a directory file and then press Move To Here. The files will be placed in the selected directory.

The buttons on the upper right of the card function the same as they do on the working directory display. This will allow you to move around in the directory hierarchy.

3.6. Details card

When you select a file on the working directory display and press Details, you will be shown this card.



You may change the file's protection by clicking on the checkboxes and then pressing change.

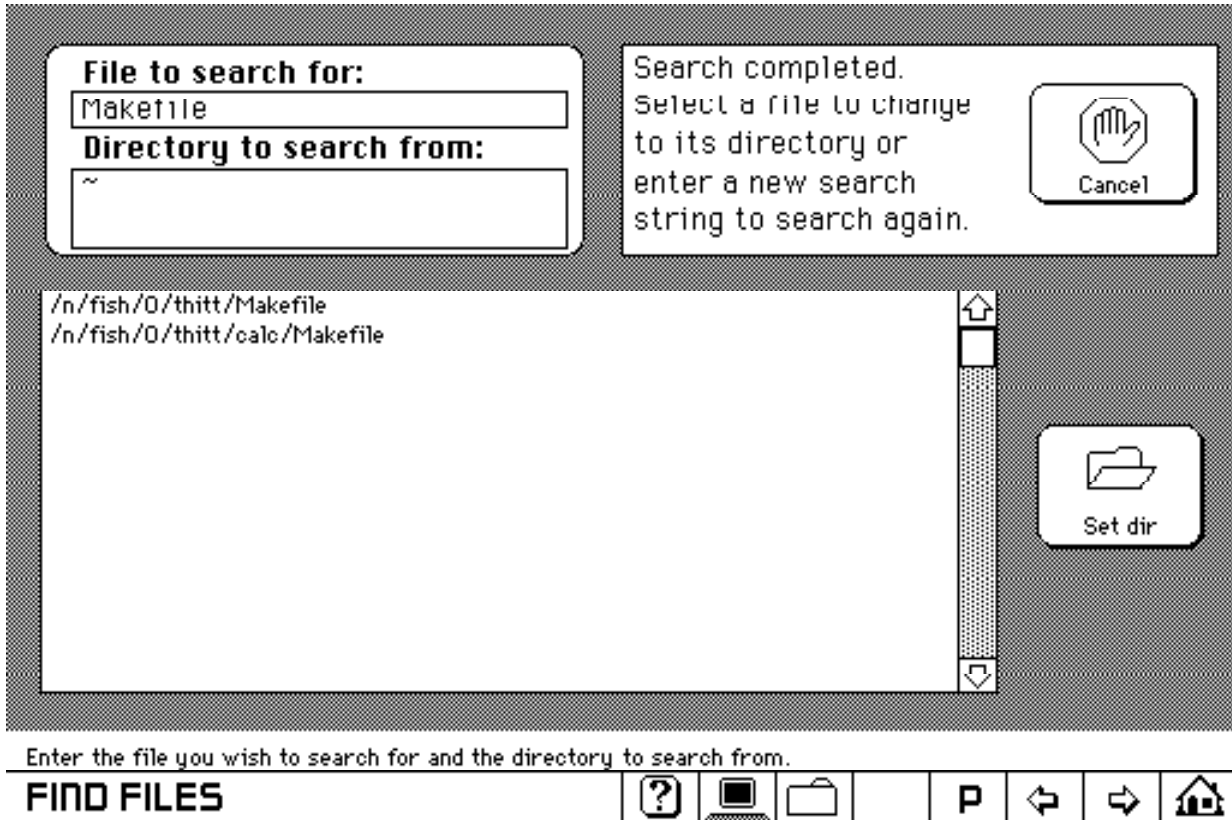
FILE DETAILS



This card shows you information on the file, such as its owner, size and last modification date. It also shows you the file's protection. If you own the file, you may change its protection by clicking on the checkboxes and then pressing Change.

3.7. Find card

When you press Find on the working directory display, you will be shown this card.



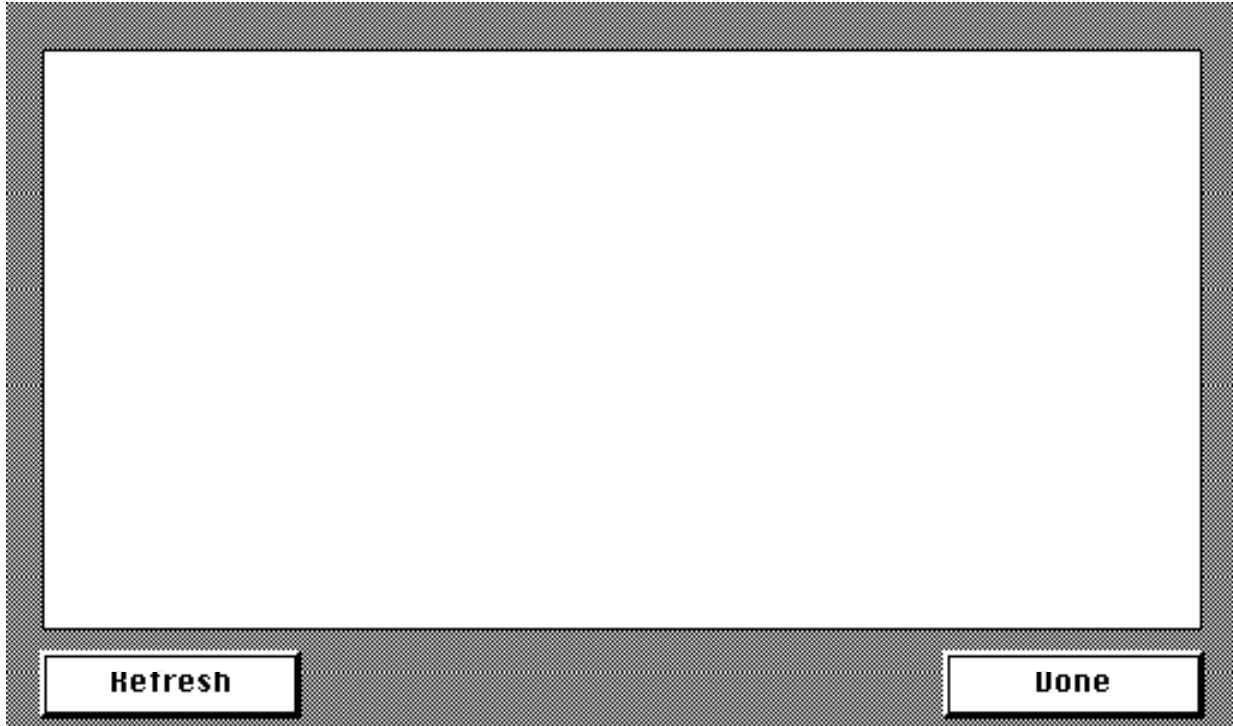
Enter the file you wish to look for. File names are case sensitive and wildcards are not allowed. You must also enter a directory specification to serve as the root of the search. The search will be restricted to files found in that directory or in directories below the starting directory. To search the entire file system, enter a slash (/). To search only files in your directory area, enter tilde (~).

Once you have entered the file and starting directory, press Start. To interrupt the search before it is complete, press and hold down the mouse button until the search aborts.

Once a search is complete, you can move to the directory where files were found by clicking on the file of interest and pressing Set dir.

3.8. Terminal emulator card

The terminal emulator card that looks like this:



This is the terminal emulator. When you are done with the program, and back at the prompt, press 'Done'.

TERMINAL EMULATOR



When you are done running an application program, such as emacs, and have returned to the shell prompt, press the Done button. You will be transferred to the working directory display.

3.9. Compatibility

This section discusses compatibility issues.

- Long file names - Long file names cause problems with the working directory display. In particular, the working display autowraps long file names.
- Invisible files - The UNIX™ browser does not display invisible files. This could cause difficulties in deleting directories (because the directory isn't really empty).

4. Contact programming examples

This section of the document gives you some examples of how to program using Contact's XCMDs and Hypertalk. It assumes you are reasonably familiar with Hypertalk and serial communications.

The first example, "Starting & Stopping Contact" tells you how to start and stop the Contact driver. You must do this before you can use any of the other XCMDs. Additionally, you **must** close the Contact driver when the Contact application stack is closed.

The second example shows how to create a terminal emulator card like the one in the Contact demonstration stack.

The last example shows how to issue a command to a remote computer system and read its responses, similar to the Find command in the UNIX™ browser stack.

4.1. Starting & Stopping Contact

Before you can use any of Contact's XCMDs or XFCNs, you must start the Contact driver. To do this, issue the Hypertalk command:

```
Serial "start"
```

This will start the driver and open the serial port. You may also wish to add parameters to set serial line parameters. For more details, see the section on the Serial "start" XCMD.

Once the Contact driver is running, you may use the other Contact XCMDs and XFCNs to program a dialog with the other computer.

Before closing the Contact application stack (including when quitting Hypercard), you must stop the serial driver. To do this, issue the command:

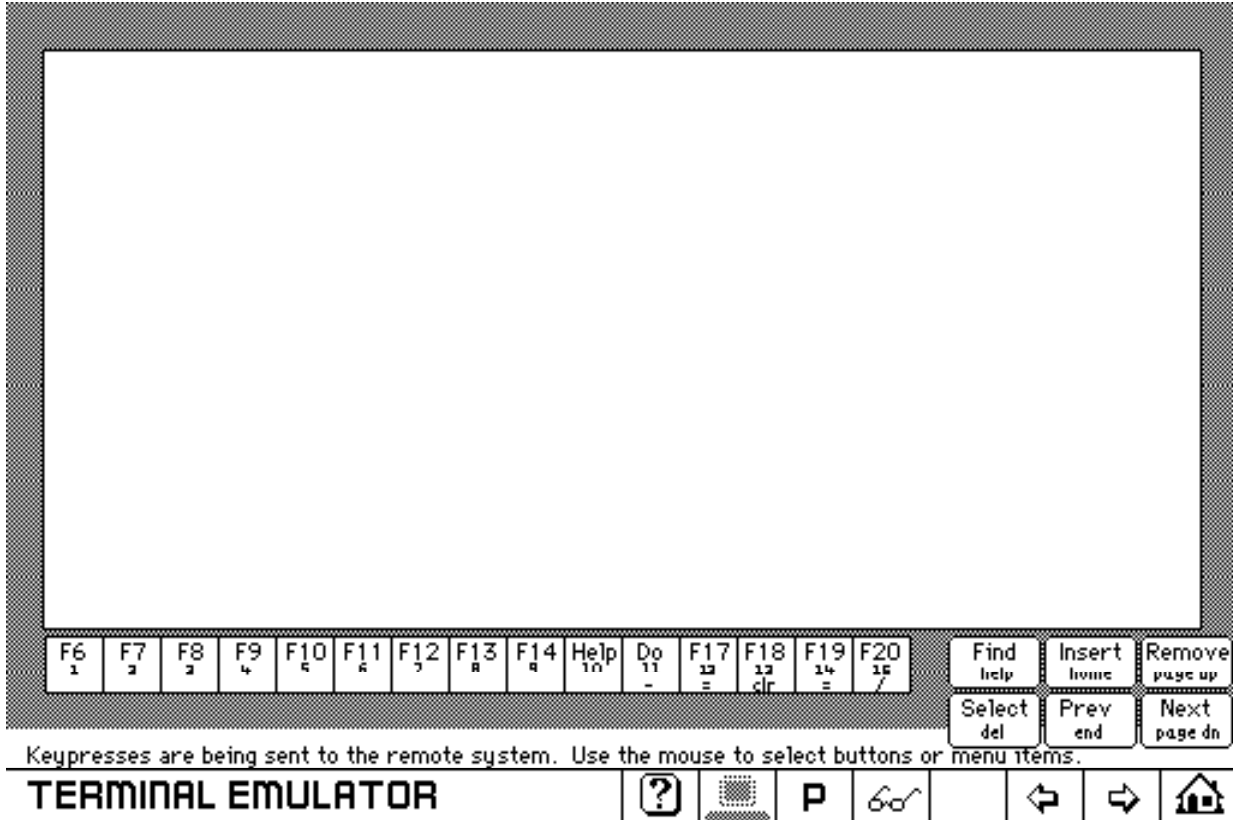
```
Serial "stop"
```

You must stop the Contact driver when closing the Hypercard stack because the Contact driver's resources are stored in the Hypercard stack. When the stack is closed, the driver's resources are no longer accessible.

You may want to perform the Serial "start" command in your stack's **openStack** handler. It is a very good idea to perform the Serial "stop" command in your stack's **closeStack** handler.

4.2. A terminal emulator card

The Contact demonstration stack contains a terminal emulation card. The card looks like this:



The **openCard** handler for this card contains these two statements:

```
Keyboard "on"  
Screen "on", "16, 20"
```

The Keyboard "on" statement causes keyboard input to be sent to the remote system, as long as Hypercard's main window is frontmost. The Screen "on" statement causes the terminal emulator display to be drawn in Hypercard's window. The second parameter "16, 20" causes the upper left corner of the screen to be located 16 pixels from the left of the window edge and 20 pixels from the top.

In the **closeCard** handler, you need to turn off the screen and the keyboard:

```
Keyboard "off"  
Screen "off"
```

By just turning the screen and the keyboard off, the terminal emulator will still be active, just not displayed. This means that while the user is looking at a different card, all characters received by the Macintosh will be acted upon by the terminal emulator, but the characters will not be displayed on the screen. When the screen is subsequently shown (via a Screen "on" command), the characters will then be shown.

It is important to turn both the screen and the keyboard off when Hypercard is not displaying the terminal emulator card. If the screen is left on, the terminal emulator display will overwrite Hypercard's window,

obscuring information. If the keyboard is left on, any keypresses will be sent to the remote system, even if the user has selected a text field.

The function key buttons were created using the Type command:

```
Type numToChar(27) & "[17~" --sends an F6
```

numToChar is a built-in Hypercard function that converts a number to a character. ASCII 27 is the escape character. The ampersand concatenates escape with "[17~", giving the escape sequence for F6. The Type XCMD sends this string to the remote system.

4.3. Sending a UNIX™ command

You can also program Hypercard to conduct complex dialogs with the remote computer system. This example sends a UNIX™ find command. It then examines the command's response to display the names of the files that were found.

```
Terminal "Hibernate"  
Type "find /* -name foo.c -print" & return  
get LookFor (return)  
  
put Capture (prompt & "|" & return) into x  
  
repeat while x is not empty  
  put x & return after cd fld "Found"  
  put Capture (prompt & "|" & return) into x  
end repeat
```

Terminal "Wakeup"

Terminal "Hibernate" tells the driver to stop accepting input. Normally, if serial input arrives to the Macintosh, the Contact driver automatically handles it as input for the VT220 terminal emulator. In this case, you don't want that to happen. You want this script to get every response that comes from the UNIX™ computer.

Type "find / -name foo.c -print" & return* sends the find command to the UNIX™ computer. This command searches the entire file system for foo.c. The Type XCMD does not automatically send a carriage return after it sends the text. Thus, you must explicitly tell the Type XCMD to send a carriage return by concatenating it onto the end of the string.

get LookFor (return) accepts the command's local echo. After sending the command to the UNIX™ computer, the first thing the UNIX™ computer does is send this string back to us, so that it appears on our screen. *get LookFor(return)* causes everything to be ignored up to the first return, which immediately follows the command echo. Most computer systems perform this local-echoing.

put Capture (prompt & "|" & return) into x gets a single line from the UNIX™ computer, or nothing if the UNIX™ computer sends the prompt string back. The prompt variable is a global variable which has been set to the UNIX™ system's prompt. The vertical bar tells the Capture XFCN to stop capturing input when it receives a carriage return or the prompt string.

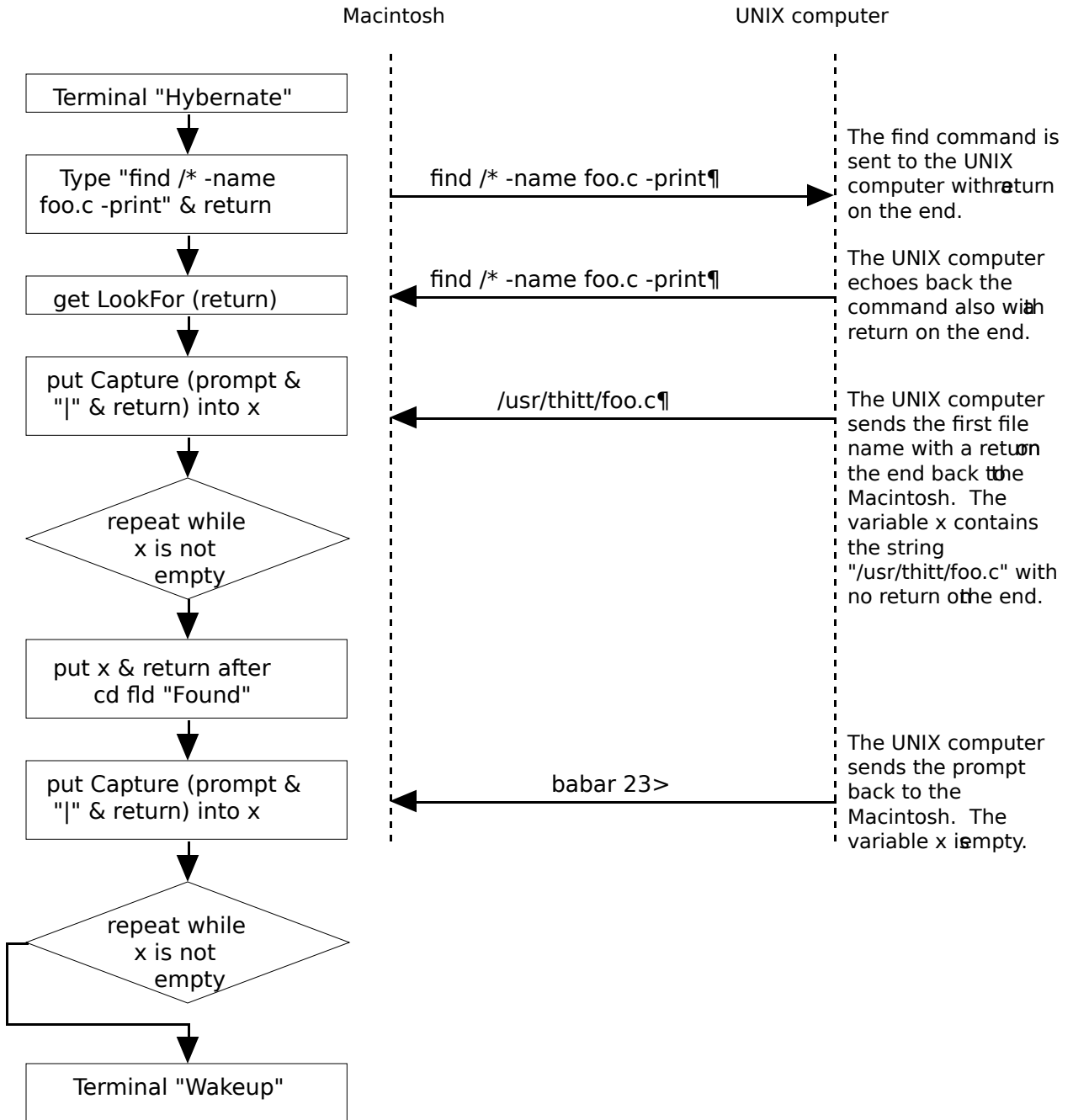
If the UNIX™ system finds a file, it will send back the file's name, followed by a carriage return. In this case, the Capture XFCN will return the file's name and Hypercard puts it in x. The carriage return, which is the end string, is not put in x.

When the find command has completed, UNIX™ computer sends the command prompt at the beginning of the line. In this example, that would mean that the Capture XFCN would return empty. Remember that Capture does not include the string which ended the input in the return value. Additionally, by default the Capture XFCN does not return any control characters, such as linefeeds. Thus the Capture XFCN returns empty. The loop keeps getting the output of the find command until x is empty.

put x & return after cd fld "Found" places the output of the UNIX™ find command after the current contents of the card field named "Found".

Terminal "Wakeup" allows the VT220 emulator to again accept input.

The following flowchart shows how this all operates when a single file is found. Carriage returns are shown with the paragraph symbol (§). The UNIX™ computer's prompt is "babar" followed by a space, a number, and then the greater than sign (>). The global variable contains only "babar".



5. Commands Reference

This section describes the existing XCMDs and XFCNs for Contact.

In general, all commands and their parameters are not case-sensitive. However, all strings sent to the remote computer or used to match input from the remote computer **are** case-sensitive.

5.1. Capture

This XFCN captures input from the remote system and returns it. To cancel capturing text, hit ⌘ - period. If the terminal is on, then the characters are displayed (and control sequences are interpreted as they come in).

Format:

```
get Capture ([end] [,mode] [,verbatim])
```

Parameters:

end

This is the optional string which is used to end the capture. The matched string is not included in the returned value. However it is consumed by the Capture XFCN. If this parameter is the empty string, then the capture is asynchronous and must be ended with a Capture call with mode equal to "Cancel".

The end string is case sensitive and has a simple OR capability. Values which are separated by a vertical bar ("|") are treated as separate possibilities which can end the input. The first one received will end input.

mode

```
"Async" | "Sync" | "Cancel"
```

Controls whether the Capture is asynchronous or synchronous. If no end string is specified (i.e. empty), then mode must be "Async", which is the default when no end string is specified. The default when an end string is specified is synchronous.

Asynchronous capture means that control returns immediately to the script while capturing occurs. The user may use the interactive terminal emulator, etcetera. The capture ends when a call to Capture with mode equal to "Cancel" or the end string (if not empty) is received. The received text is returned as the result of the Capture(" ", "Cancel") call.

Synchronous capture is only possible when an end string has been specified. Control does not return to the script until the specified end string is received or the user presses ⌘ -period.

verbatim

```
"True"|"False"|"Yes"|"No"
```

Defaults to "False".

When false, control characters, such as linefeeds are stripped from the returned value. Otherwise, the returned value is returned exactly as it was received.

Examples:

```
put Capture ("Fred|Barney") into txt
```

This example reads input until either the string "Fred" or "Barney" arrives. If the incoming characters are:

```
ABCDFREDHMB Barney
```

Then the returned value is "ABCDEFREDHM". Note that "FRED" did not end the capture as its case did not match "Fred". Also note that "Barney" is not included in the returned string.

Caveats:

If the end string contains control-characters and verbatim is False, then the return string may be truncated.

A maximum of 32k characters may be captured.

5.2. Keyboard

This XCMD allows you to control the keyboard configuration. It has five main variations.

- on - Enable keyboard input
- off - Disable keyboard input
- (no)Application - turn on and off the application keypad mode
- (no)Linefeed - send a a linefeed after carriage returns
- backspace|delete - delete key sends backspace or delete

5.2.1. Keyboard “on”

Allows Contact to read input from the keyboard. After issuing this command, all typing will be intercepted by Contact and interpreted as VT220 keypresses if Hypercard's main window is the frontmost window on the Macintosh screen.

Format:

Keyboard “on”

Examples:

Keyboard “on”

Keypresses on the Macintosh keyboard will be sent to the remote system - if Hypercard's main window is frontmost.

5.2.2. Keyboard “off”

Stops Contact from reading input from the keyboard.

Format:

Keyboard “off”

Examples:

Keyboard “off”

Keypresses on the Macintosh keyboard will be interpreted by Hypercard.

5.2.3. Keyboard “(no)Application”

Sets or unsets the VT220 Keyboard Application Keypad Mode.

Format:

Keyboard "(no)Application"

Examples:

Keyboard "noApplication"

Keypad keypresses will be sent as is. For instance, pressing the keypad zero key will send "0".

5.2.4. Keyboard "(no)Linefeed"

Controls whether a linefeed is sent after a carriage return or not.

Format:

Keyboard "(no)Linefeed"

Examples:

Keyboard "Linefeed"

Pressing the return key will send a carriage return followed by a linefeed.

5.2.5. Keyboard "backspace|delete"

Controls whether the Macintosh delete key sends a backspace or a delete. Contact defaults to sending a delete character.

Format:

Keyboard "backspace|delete"

Examples:

Keyboard "backspace"

The key marked delete on the Macintosh keyboard will send a backspace character.

5.3. LookFor

LookFor is similar to capture. However, it returns only the matched string instead of the preceding text and operates only in synchronous mode.

Format:

```
get LookFor (end [,timeout])
```

Parameters:

end

A string which is used to end the receive. The matched string is returned as a return value and consumed by the LookFor XFCN.

The end string is case sensitive and has a simple OR capability. Values which are separated by a vertical bar (“|”) are treated as separate possibilities which can end the input. The first one received will end input.

timeout

The number of seconds to wait until an end string has been received. If the timeout value is reached before an end string has been reached or the user presses ⌘-period, then empty is returned. If the timeout value is not specified, or is zero, then this XFCN never times out.

Examples:

```
txt = LookFor (“Fred|Barney”)
```

This example reads input until either the string “Fred” or “Barney” arrives. If the incoming characters are:

```
ABCDFREDHMBarney
```

Then the returned value is “Barney”.

5.4. Receive

Format:

```
Receive filename [,end] [,mode] [,verbatim]
```

Parameters:

filename

The name of the file to place the received text in. A full or partial path name may be specified. This parameter is required.

end

This is the optional string which is used to end the receive. The matched string is consumed. If this parameter is the empty string, then the receive is asynchronous and must be ended with a Receive call with mode equal to "Cancel".

The end string is case sensitive and has a simple OR capability. Values which are separated by a vertical bar ("|") are treated as separate possibilities which can end the input. The first one received will end input.

mode

```
"Async"|"Sync"|"Cancel"
```

Controls whether the Receive is asynchronous or synchronous. If no end string is specified (i.e. empty), then mode must be "Async", which is the default when no end string is specified. The default when an end string is specified is synchronous.

Asynchronous reception means that control returns immediately to the script while capturing occurs. The user may use the interactive terminal emulator, etcetera. The receive ends when a call to Receive with mode equal to "Cancel" or the end string (if not empty) is received.

Synchronous reception is only possible when an end string has been specified. Control does not return to the script until the specified end string is received or the user presses ⌘-period.

verbatim

```
"True"|"False"|"Yes"|"No"
```

Defaults to "False".

When False, control characters, such as linefeeds are not placed in the file. Otherwise, the file contains exactly what was received by the Macintosh.

Examples:

```
Receive "Capture.txt"
```

This example begins storing text in the file Capture.txt until cancelled:

```
Receive "", "Cancel"
```

Caveats:

If the end string contains control-characters and verbatim is False, then the contents of the file may be truncated.

5.5. Screen

The screen XCMD turns on and off the VT220 emulator's display. When the screen is on, incoming characters handled by the terminal emulator will be displayed on the screen. If the screen is off, the incoming characters will be handled and stored internally, but will not be displayed immediately.

Turning the screen on and off allows you to have a card set up as a terminal emulator. When you open the card, turn the screen on. When you close the card, turn the screen off.

5.5.1. Screen "on"

Screen "on" causes the the terminal emulator's display to be drawn inside Hypercard's main window.

Format:

Screen "on" [,rectangle]

Parameters:

rectangle

The rectangle in which the screen will appear. This is in local coordinates. At this time, only the top left point is used.

Defaults to (0, 0).

Examples:

Screen "on", "16, 20"

Caveats:

You can use Screen "on" to refresh the screen. Be sure to specify the rectangle.

5.5.2. Screen "off"

Screen "off" stops Hypercard from drawing the terminal emulator's display inside Hypercard's main window.

Format:

Screen "off"

Examples:

Screen "off"

5.6. Serial

This XCMD allows you to start or stop the Contact driver, set baudrates, send breaks, set parity, etc. There are 11 main forms the serial XCMD can take:

- start - start the Contact driver
- stop - stop the Contact driver
- disconnect - send a disconnect
- break - send a break
- baudrate - change the baudrate
- stopbits - set the stop bits
- parity - set the parity
- bytesize - set the byte size
- flow control - use or don't use flow control
- maintainDTR - maintain DTR when serial driver is closed
- breakOnError - automatically send a break on framing errors

The rest of this section describes each of these formats.

5.6.1. Serial “start”

Start the Contact driver and open the serial port. *Before issuing this command, make certain that Hypercard's window is the frontmost window on the screen. Close the message box, and any tool palette windows you have shown.*

Format:

```
Serial "start" [,baudrate] [,stopbits] [,parity] [,bytesize] [,useFlowControl]  
                [,maintainDTR] [,port]
```

Parameters:

Baudrate

```
"300"|"1200"|"1800"|"2400"|"3600"|"4800"|"9600"|"19200"|"57600"
```

The baud rate to communicate at.

Defaults to 1200.

Stopbits

```
"1.0"|"1.5"|"2.0"
```

The number of stop bits to send after each character.

Defaults to 1 stop bit.

Parity

```
"Even"|"Odd"|"None"
```

Defaults to no parity.

Bytesize

"5"|"6"|"7"|"8"

Defaults to 8.

useFlowControl

"Yes"|"No"|"True"|"False"

Controls whether the Macintosh responds to and sends XON and XOFF flow control characters.

Defaults to True.

maintainDTR

"Yes"|"No"|"True"|"False"

A True value causes the Macintosh to keep the Data Terminal Ready line (DTR) in its asserted state when Contact is stopped via a Serial "stop".

Defaults to False.

port

"Phone"|"Printer"

Specifies which serial port to open, either the telephone port or the printer port.

Defaults to Phone.

Examples:

```
Serial "start", "9600"
```

Starts the Contact driver and opens the serial port at 9600 baud. The rest of the serial parameters are the defaults.

5.6.2. Serial "stop"

Stop Contact and close the serial port. *Make sure to call this before closing the stack. This includes when quitting Hypercard.*

Format:

```
Serial "stop"
```

Examples:

```
Serial "stop"
```

Stops the Contact driver.

5.6.3. Serial “disconnect”

Sets Data Terminal Ready (DTR) low and then back high. Some remote devices use this signal as a hangup or attention signal. See the documentation on the remote system.

Format:

```
Serial "disconnect"
```

Examples:

```
Serial "disconnect"
```

5.6.4. Serial “break”

Send a break signal. Some remote devices use this signal as a hangup or attention signal. See the documentation on the remote system.

Format:

```
Serial "break"
```

Examples:

```
Serial "break"
```

5.6.5. Serial baudrate

Changes the baudrate.

Format:

```
Serial <baudrate>
```

Parameters:

Baudrate

```
"300"|"1200"|"1800"|"2400"|"3600"|"4800"|"9600"|"19200"|"57600"
```

The baud rate to communicate at.

Examples:

```
Serial "19200"
```

Changes the baud rate to 19200 baud.

5.6.6. Serial stopbits

Changes the number of stop bits.

Format:

```
Serial <stopbits>
```

Parameters:

Stopbits

```
"1.0"|"1.5"|"2.0"
```

The number of stop bits to send after each character.

Examples:

```
Serial "2.0"
```

Changes the number of stop bits to two.

5.6.7. Serial parity

Changes the parity.

Format:

```
Serial <parity>
```

Parameters:

Parity

```
"Even"|"Odd"|"None"
```

Examples:

```
Serial "Even"
```

Changes the parity to even.

5.6.8. Serial bytesize

Changes the number of bits per byte.

Format:

Serial <bytesize>

Parameters:Bytesize

"5"|"6"|"7"|"8"

Examples:

Serial "7"

Changes the number of bits per byte to seven.

5.6.9. Serial "(no)UseFlowControl"

Controls whether the Macintosh responds to Xons and Xoffs.

Format:

Serial "(no)UseFlowControl"

Examples:

Serial "noUseFlowControl"

The Macintosh will not send nor respond to Xons and Xoffs.

5.6.10. Serial "(no)maintainDTR"

When maintainDTR is set, the Data Terminal Ready line (DTR) will be maintained high when the Contact driver is closed. This should keep the connection with the remote system active.

Serial "(no)maintainDTR"

Examples:

Serial "maintainDTR"

DTR will be kept high when the Contact driver is closed.

5.6.11. Serial "(no)BreakOnError"

If set, sends a break signal on overrun or framing errors. This can be useful for getting UNIX™ systems to select the correct baudrate.

Format:

Serial "(no)BreakOnError"

Examples:

Serial "noBreakOnError"

The Macintosh will not send a break when it receives overrun or framing errors.

5.7. Serve

Allows the Contact driver to run interrupted for a number of seconds. Normally this is placed in an idle handler. This is only necessary to improve performance of the interactive terminal emulator.

Typically seconds should be zero. However, if you have a script which uses Hypercard's wait command, you may consider using Serve instead.

Format:

```
Serve <seconds>
```

Parameters:

seconds

The number of seconds to run for. This defaults to zero.

Examples:

```
Serve 1
```

5.8. Terminal

The Terminal XCMD has 15 main variations:

- Terminal "on" - turns the VT220 terminal emulator on
- Terminal "off" - turns the VT220 terminal emulator off
- Terminal "(no)LocalEcho" - Enables or disables local echo of characters
- Terminal "VT102|VT220" - The terminal type to send to the remote system
- Terminal "(no)eightbit" - Send eight bit control codes or seven bit control codes
- Terminal "(no)insert" - Overstrike or insert mode
- Terminal "(no)parsealerts" - Display alerts on receipt of unknown escape sequence
- Terminal "Answerback" - Sets the answerback message
- Terminal "reset" - Performs a soft reset
- Terminal "clear" - Clears the terminal screen
- Terminal "(no)TextCursor" - Display the cursor or not
- Terminal "resetTabs" - Reset tabs to every eight positions
- Terminal "Hibernate" - Causes the terminal emulator to pause
- Terminal "Wakeup" - Wakes the terminal emulator up

5.8.1. Terminal "on"

Turns the terminal emulator "on". All incoming serial information will be written to the emulator. If the screen is also "on" then the serial information will be displayed. If the screen is not displayed, the screen layout will be stored.

Format:

Terminal "on"

Examples:

Terminal "on"

5.8.2. Terminal "off"

Turns the terminal "off". Incoming serial information will be disregarded.

Format:

Terminal "off"

Examples:

Terminal "off"

5.8.3. Terminal “(no)localecho”

Controls whether the terminal emulator locally echoes outgoing characters or not. Contact defaults to nolocalecho.

Format:

```
Terminal “(no)localecho”
```

Examples:

```
Terminal “nolocalecho”
```

When a key is pressed on the Macintosh keyboard, the key will not be echoed to the screen locally. It is remote computer system’s responsibility to echo the characters.

5.8.4. Terminal “VT102|VT220”

Controls whether the terminal emulator emulator is a VT102 or VT220. In VT102 mode, all transmitted codes are seven bits and the device response is VT102. This does not disable function keys nor the recognition of VT220 sequences.

Format:

```
Terminal “VT102”|“VT220”
```

Examples:

```
Terminal “VT102”
```

The terminal emulator will send a VT102 identification code to the remote system.

5.8.5. Terminal “(no)eightbit”

Controls whether the terminal emulator sends eight bit control codes. In VT102 mode, the terminal emulator sends seven bit codes irregardless of this switch.

Format:

```
Terminal “(no)eightbit”
```

Examples:

```
Terminal “noeightbit”
```

The terminal does not send eight bit control codes. Instead it sends seven bit control codes.

5.8.6. Terminal “(no)insert”

Controls whether the terminal emulator is in insert mode or replace mode.

Format:

Terminal “(no)insert”

Examples:

Terminal “insert”

When the terminal receives a character, all characters to the right of the cursor position will be moved one column to the right to make room for the new character.

5.8.7. Terminal “(no)parsealerts”

Controls whether the terminal emulator displays an alert box if it receives an escape sequence it does not recognize.

Format:

Terminal “(no)parsealerts”

Examples:

Terminal “parsealerts”

Contact will display an alert dialog box when it receives an escape sequence it does not recognize.

5.8.8. Terminal “(no)autowrap”

Controls whether the terminal is in autowrap mode or not.

Format:

Terminal “(no)autowrap”

Examples:

Terminal “noautowrap”

The cursor will not wrap around to column one when it moves past the right side of the screen. Instead, the cursor will stay in the last column of the screen.

5.8.9. Terminal “answerback”

This call sets the answer back message. The string can be omitted so nothing will be transmitted when an ENQ is returned. Contact does not automatically append on a carriage return, so if you want one in the answerback message, include it. The maximum size of the string is 255 bytes.

Format:

```
Terminal "answerback" [,string]
```

Parameters:

String

Up to 255 characters to be transmitted. If this is empty, then nothing will be transmitted.

Examples:

```
Terminal "Answerback", "run foobar" & return
```

When Contact receives an ENQ, it will send back the string “run foobar” followed by a carriage return.

5.8.10. Terminal “reset”

Performs a soft terminal reset.

Format:

```
Terminal "reset"
```

Examples:

```
Terminal "reset"
```

5.8.11. Terminal “clear”

Clears the terminal's screen.

Format:

```
Terminal "clear"
```

Examples:

```
Terminal "clear"
```

5.8.12. Terminal “(no)textcursor”

Controls whether the cursor is visible or not.

Format:

Terminal “(no)textcursor”

Examples:

Terminal “textcursor”

The cursor will be displayed on the terminal emulator.

5.8.13. Terminal “resettabs”

Resets the terminal’s tabs to every eight characters.

Format:

Terminal “resettabs”

Examples:

Terminal “resettabs”

Tabs will be set to every eighth column.

5.8.14. Terminal “Hibernate”

Causes the terminal emulator not to accept input from the serial port. You typically put a Terminal “Hibernate” at the beginning of a Hypercard script which converses with the remote computer.

Terminal “Hibernate” is slightly different than Terminal “Off”. Terminal “Off” stops all input from going to the terminal emulator, even while a Contact XCMD or XFCN is operating. After a Terminal “Hibernate” call, the terminal emulator still interprets incoming characters, but only during a Contact XCMD or XFCN. The terminal emulator does not read characters from the serial port on its own volition. This keeps the terminal emulator from “stealing” incoming characters while other Hypertalk commands are executing.

Format:

Terminal “Hibernate”

Examples:

Terminal “Hibernate”

The terminal emulator will not receive input. Instead it will be buffered and read by the Hypercard script.

5.8.15. Terminal “Wakeup”

Causes the terminal emulator to again accept input from the serial port. You should put this at the end of a script that issued a Terminal “Hibernate”.

Format:

Terminal “Wakeup”

Examples:

Terminal “Wakeup”

The terminal emulator will receive and process input.

5.9. Transmit

Sends the contents of a text file out the serial port. To cancel sending text, hit ⌘ - period.

Format:

```
Transmit filename [,processchars] [,charWait] [,lineWait] [,turnAround]
                [,charEcho]
```

Parameters:

filename

The name of the file to send. A full or partial path name may be specified. This parameter is required.

processchars

"Yes"|"No"|"True"|"False"

Controls whether Contact processes received characters or not.

If this parameter is True or Yes, then Contact will read characters the serial port while this command is executing. If the terminal emulator is on, then they will be processed and displayed. This is usually useful for simply sending text to the remote system.

If this parameter is No or False, then Contact will not read any characters from the serial port. This is usually useful in scripts where a dialog is being carried on with the remote system.

Defaults to True.

charWait

Normally, the number of ticks to wait after sending each character that is not a carriage return. A tick is 1/60 of a second.

However, if charEcho is Yes or True, this is a timeout period in ticks. The Transmit XCMD will resume sending text after the timeout period has passed, even if it has not received the character's echo.

Defaults to zero.

lineWait

Normally, the number of ticks to wait after sending a carriage return. A tick is 1/60 of a second.

However, if a turnAround string is specified, this is a timeout period in ticks. The Transmit XCMD will resume sending text after the timeout period has passed, even if it has not received the turnAround string.

Defaults to zero.

turnAround

A character string to await after each carriage return is sent. This implies setting processChars to True and the lineWait parameter is interpreted as a timeout value. If no string is specified or the empty string is specified, the Transmit XCMD does not wait for any character string after sending each carriage return.

Defaults to empty.

charEcho

"Yes"|"No"|"True"|"False"

If True or Yes, then the Transmit XCMD looks for the remote system to echo each character sent. The charWait parameter is interpreted as a timeout value.

If False or No, then the Transmit XCMD does not look for the remote system to echo each character.

Defaults to False.

Examples:

```
Transmit foo.c, "Yes", 0, 120, ">"
```

This example sends the file foo.c to the remote system. Characters that the remote system sends back (such as the character echo) are processed and displayed by the terminal emulator. The Transmit XCMD does not wait for any period of time after sending each character. It does wait for the greater-than sign (>) after each carriage return is sent. However, if the greater-than sign is not received in two seconds (120 ticks = 2 seconds), it continues with the next line anyway.

5.10. Type

Sends the contents of a Hypercard container out the serial port. To cancel sending text, hit ⌘ -period.

Format:

```
Type container [,processchars] [,charWait] [,lineWait] [,turnAround]
           [,charEcho]
```

Parameters:

container

The Hypercard container or character string to send to the remote system. This parameter is required.

processchars

"Yes"|"No"|"True"|"False"

Controls whether Contact processes received characters or not.

If this parameter is True or Yes, then Contact will read characters from the serial port while this command is executing. If the terminal emulator is on, then they will be processed and displayed. This is usually useful for simply sending text to the remote system.

If this parameter is No or False, then Contact will not read any characters from the serial port. This is usually useful in scripts where a dialog is being carried on with the remote system.

Defaults to false.

charWait

Normally, the number of ticks to wait after sending each character that is not a carriage return. A tick is 1/60 of a second.

However, if charEcho is Yes or True, this is a timeout period in ticks. The Type XCMD will resume sending text after the timeout period has passed, even if it has not received the character's echo.

Defaults to zero.

lineWait

Normally, the number of ticks to wait after sending a carriage return. A tick is 1/60 of a second.

However, if a turn around string is specified, this is a timeout period in ticks. The Type XCMD will resume sending text after the timeout period has passed, even if it has not received the turnAround string.

Defaults to zero.

turnAround

A character string to await after each carriage return is sent. This implies setting processChars to True and the lineWait parameter is interpreted as a timeout value. If no string is specified or the empty string is specified, the Type XCMD does not wait for any character string after sending each carriage return.

Defaults to empty.

charEcho

"Yes"|"No"|"True"|"False"

If True or Yes, then the Type XCMD looks for the remote system to echo each character sent. The charWait parameter is interpreted as a timeout value.

If False or No, then the Type XCMD does not look for the remote system to echo each character.

Defaults to False.

Examples:

Type "35" & return

The string "35" followed by a carriage return is sent to the remote system.

6. Unimplemented terminal emulation features

- Blinking video renditions — Displayed as underlined
- Downloadable character sets
- Double height
- Double width
- KAM (Keyboard Action Mode)
- Smooth scrolling
- 132 columns
- Set auto repeat — Contact uses the setting from the control panel.
- National replacement char sets
- LEDs
- Downloadable function keys
- Compose sequences
- Reverse screen
- Printing escape sequences
- Test escape sequences
- Display controls
- VT52 mode
- hard reset —
 - Doesn't perform a disconnect.
 - Does reset tabs to every 8 columns.

7. History

1.0.0 Initial release.

8. Registration

Contact is not in the public-domain. Registration has the following benefits: one free upgrade to the next version (when it becomes available), the right to distribute stacks you create with Contact, input on features to be included in future releases, and technical support. Here is the policy:

1. **You may evaluate Contact for 30 days.** If after that time, you decide it is not suitable for your purposes, you must erase it from your disk(s).
2. **If you find Contact useful,** you must register your copy by sending US\$10 to Todd Hitt, 1447A Cliff Court, Columbus, OH 43204 within 30 days of receiving Contact.
3. **Each license is for a single user only.** Each registered copy may be used only on one computer at a time.

Site licenses are also available. Please contact me.

Distribution of stacks you create

You can distribute stacks created with Contact under the following conditions:

1. **You may not modify any of the Contact resources.** These resources are listed as "Required for Contact" in the document entitled **Contact resources**.
2. **You must have registered your copy of Contact.**

Note that with this policy, all users of your stack are responsible for registering their copies of Contact. If you desire to distribute stacks you create, and not require users to register their copies, you must contact me to receive a distribution license.

Redistribution of the Contact distribution kit

You may distribute the Contact distribution kit to anyone you like with the following conditions:

1. **You must include all of the contents of the Contact distribution kit.** You must distribute each of these exactly as you received them - make sure to give away or upload original versions.
2. **You may not charge anything for Contact.** Nor may you use Contact to induce anyone to buy a product. There are two exceptions to this policy: BBS's and commercial information systems (like GENIE™) that charge a fee for membership and connect time, but not for individual downloads. Additionally, non-profit user's groups may charge a copying fee for distribution. Note that companies which sell disks for profit may not distribute Contact without written permission.

9. Support and upgrades

If you find a bug or have a request, please let me know. I will endeavor to reply as soon as possible.

You can reach me via U.S. mail:

Todd Hitt
1447A Cliff Court
Columbus, OH 43204

Or via GEnie™ mail: THITT

Upgrades

Your registration fee entitles you to receive future updates. The first update will be sent free of charge to you in the mail when it becomes available. You will be responsible for ordering and paying for subsequent updates. You will be notified when the first such subsequent update becomes available.

10. Trademarks and Warranty

Trademarks

VT is a trademark of Digital Equipment Corporation.

Macintosh is a trademark licensed to Apple Computer, Inc.

UNIX is a trademark of A T & T.

GENie is a trademark of the General Electric Company.

Hayes is a trademark of Hayes Microcomputer Products, Inc.

Sun is a trademark of Sun Microsystems, Inc.

Warranty

Contact is provided with absolutely no warranty, to the extent permitted by applicable state law. Todd Hitt, and/or other parties provide Contact "AS IS" without warranty of any kind, either express or implied, including, but not limited to, the quality, performance, or fitness for a particular purpose. The entire risk of using this program is with you, the user. Should Contact prove defective, you assume the cost of all necessary servicing, repair, or correction.

In no event will the author(s) be liable for direct, indirect, special, incidental, or consequential damages, resulting from use, inability to use, or defect in this software (including but not limited to loss of data or data being rendered inaccurate) or its documentation, even if advised of the possibility of such damages.

The absence of warranty set forth above is exclusive and in lieu of all others, oral or written, express or implied.