HEWLETT
PACKARD

HP 41CV Emulator Application Card for the HP 48SX HP 82210A

HP82210A
Owner's Manual

Comments on the HP 41CV Emulator Application Card Owner's Manual

We welcome your evaluation of this manual. Your comments and suggestions help us improve our publications.

HP 41CV Emulator Application Card Owner's Manual

Please circle a response for each of the statements below. You can use the **Comments** space to provide additional opinions.

1 = Strongly Agree	4 = Disagree
2 = Agree	5 = Strongly Disagree
3 = Neutral	

▪ The manual is well organized.	1 2 3 4 5
▪ I can find the information I want.	1 2 3 4 5
▪ The information in the manual is accurate.	1 2 3 4 5
▪ I can easily understand the instructions.	1 2 3 4 5
▪ The manual contains enough examples.	1 2 3 4 5
▪ The layout and format are attractive and useful.	1 2 3 4 5
▪ The illustrations are clear and helpful.	1 2 3 4 5

- The manual length is: too long appropriate too short
- The parts I refer to most frequently are:

Chapter 1 Chapter 2 Chapter 3 Chapter 4
Chapter 5 Appendixes Index

Comments:____
____
____
____
____
____

Name:____
Address:____
City/State/Zip:____
Occupation:____
Phone:(___)____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 38 CORVALLIS, OR

POSTAGE WILL BE PAID BY ADDRESSEE

HEWLETT-PACKARD COMPANY
DOCUMENTATION DEPARTMENT
CORVALLIS DIVISION
1000 NE CIRCLE BLVD
CORVALLIS OR 97330-9973

HP 41CV Emulator Application Card for the HP 48SX HP 82210A

Owner's Manual

Edition 1 October 1990
Reorder Number 82210-90001

Notice

Corvallis Division
1000 N.E. Circle Blvd.
Corvallis, OR 97330, U.S.A.

Printing History

Edition 1 October 1990 Mfg. No. 82210-90001

Contents

Foreword

Welcome to the HP 82210A — the HP 41CV Emulator Application Card for the HP 48. This card provides an upgrade path from the popular HP 41 series to the HP 48.

About the HP 41CV Emulator Application Card

If you are familiar with any model of HP 41, this card will help you to quickly begin using the HP 48.

Plug the card into an HP 48 and you can:

- Copy programs or data from an HP 41 to the HP 48 using the HP82242A Infrared printer module plugged into the HP 41. You can also type HP 41 programs and data into the HP 48.
- Run HP 41 programs on the HP 48 without modification, provided only standard functions (or the extended functions and printer functions described in this manual) are used.
- Run HP 41 programs using HP 41 data on the HP 48.
- Carry out keyboard calculations, as if you were using an HP 41. A keyboard overlay is provided for this purpose.
- Modify your HP 41 programs once they are in the HP 48.
- Build a library of HP 41 programs in your HP 48.
- Exchange information between the HP 48 and the special environment in which the HP 41 programs and data exist.
- Invoke HP 41 programs from within HP 48 programs, or invoke HP 48 programs from within HP 41 programs.

- Extend the HP 41 system provided by this card. One way to do this is to write HP 48 programs that carry out functions not already on this card. For example, you can write programs to carry out HP 41CX Extended Functions or Time Functions.

The Card does not provide Extended Memory, but the emulator can use more data registers than are available in an HP 41.

The Card does not support HP-IL functions. Some alternatives are suggested in Appendix D.

About This Manual

This manual is organized as follows:

- Chapter 1 begins with some simple examples to get you started, describes some of the terminology used in the manual, and describes the general principals of the HP 41 emulator.
- Chapter 2 describes the emulator menus, the HP 41 emulator keyboard and how it is used for keyboard calculations, and the variables that the emulator uses.
- Chapter 3 describes how to send programs and data from an HP 41 (any model) to the HP 48.
- Chapter 4 explains how programs are compiled, how they are run, and how they can be modified.
- Chapter 5 provides examples of ways that HP 48 programs can be added to the HP 41 environment to carry out HP 41 functions not built into the emulator.
- The Appendixes provide additional pertinent information and references to related documents.
- The last page of this manual is a foldout of the HP 41 Emulator keyboard overlay.

This manual is **not** a substitute for your HP 41 and HP 48 manuals. If you need more information about either calculator, or about the HP 41 Infrared Printer module (or other HP 41 modules), you should read the appropriate manuals. If you do not have the manuals, or if you want more information, Appendix B includes a list of helpful books.

For over 10 years, HP 41 hand-held computers and accessories have served HP customers well. Hewlett-Packard and Zengrange provide this card so you can use the full power of the HP 48 calculator, without the inconvenience of having to rewrite your favorite HP 41 programs.

The Conventions Used in This Manual

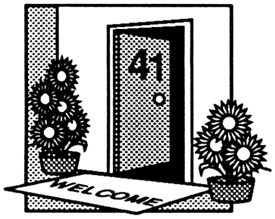
The following conventions are used throughout this manual:

- Keys on the keyboard of the HP 41 and the HP 48 are shown in boxes. For example, the ENTER key is shown as $\boxed{\text{ENTER}}$.
- Menu labels that appear at the bottom of the display on the HP 48 and correspond to the white, top row of the keyboard are shown with a gray background. For example, the EXIT menu key is shown as EXIT .
- Text that represents something from the display is shown in a special typeface. For example, the message "Yes" is shown as:

```
Yes
```

1

Getting Started



This chapter describes the general principles of the HP 41 emulator, including the following:

- Capturing HP 41 programs and data.
- Using the HP 48 editor to change an HP 41 program.
- Running HP 41 programs.

Installing and Removing the Card

To use the HP 82210A HP 41CV Emulator Application card, plug it into one of the two ports in the back of the HP 48. Installing and removing cards is described in your HP 48 Owner's Manual.

The HP 41CV Emulator card is a ROM (Read Only Memory) card, which means it does not use batteries to preserve its memory, and the programs in it cannot be changed.

When installing or removing cards, remember to follow these rules:

- Turn off the HP 48 before you insert or remove the card.
- If you remove a RAM card to make room for this card, make sure the card memory is free, not merged. See the section on RAM cards in your HP 48 Owner's Manual for details.
- If you accidentally remove a card with merged memory and see the message `Replace RAM, Press ON`, you can minimize memory loss by leaving the calculator on, reinserting the RAM card in the same port, and pressing [ON].
- When you insert the HP 41 Emulator card and turn the HP 48 on, the HP 48 will be reconfigured, and its stack and last arguments will be lost. User variables are not lost.

Card Care

When the card is not installed in the HP 48, its contacts are protected by a cover. Do not open this cover or touch the contacts. When you are not using the card, store it in a safe place.

For further information read Appendix A of this manual.

Using the Emulator Card

After installing the HP 41CV Emulator card, work through the example in this section to learn how it is used. This example may be all you need to know to use the HP 41CV Emulator card.

Attaching the HP 41 Environment Library to an HP 48 Directory

The HP 41CV Emulator card provides HP 48 commands contained in a set of menus and submenus. These menus are not built into the HP 48, nor are they user menus, instead they are built into the HP 41CV Emulator card. A set of menus and commands provided by a plug-in card is called a "library." Libraries are explained in the HP 48 Owner's Manual.

Before you can use the menus and commands in the HP 41CV Emulator card's library, the library must be "attached" to a directory in user memory. When you install the HP 41CV Emulator card, the library is automatically attached to the HP 48 Home directory. The HP 48 reconfigures itself to accept the card, and the HP 48 stack and last arguments are erased. If you want to attach the library to some other directory, see your HP 48 Owner's Manual.

Capturing and Running a Simple HP 41 Program

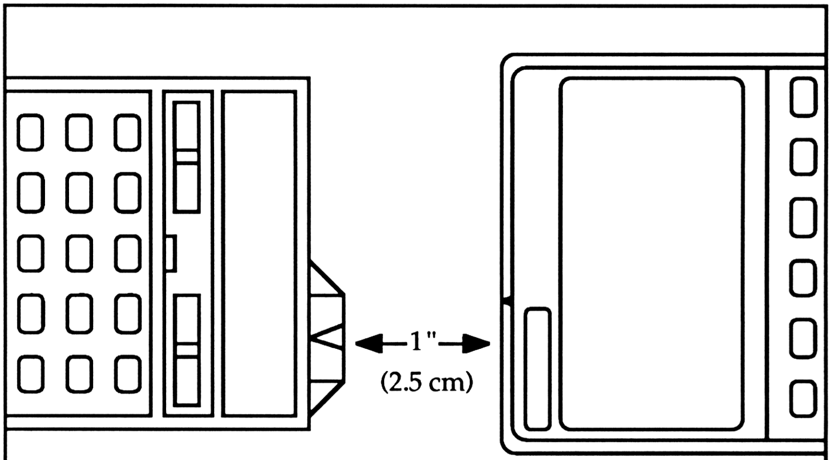
Programs stored on the HP 41 can be sent to the HP 48 using the HP 41 infrared printer module. The HP 48 receives (or captures) the program through its infrared receiver. This process is called capturing a program.

The following program is the first example program in the HP-41CV Owner's Manual:

```
01  LBL ᵀCIRCLE
02  X^2
03  PI
04  *
05  END
```

The following procedure demonstrates the use of the HP 41CV Emulator card:

1. Key the above program into your HP 41.
2. Place both calculators on a flat surface with their top edges opposite each other. The calculators should be no more than one inch (2.5 cm) apart for the transmission.



Program transfer from an HP 41 to an HP 48.

The HP82242A Infrared Printer module should be in one of the two upper ports of your HP 41, and the shiny "IR triangle" in the middle of the module should be aligned between the E and T of HEWLETT on the HP 48 Hewlett-Packard logo.

3. Carry out the transfer by following these steps:

   Access the HP 41 Emulator card and select the HP 41 menu key.

   [↰] [LIBRARY] HP-41

   Turn the HP 41 off and on again to ensure the printer mode is set correctly.

   On the HP 41, start, but do not complete, the key sequence necessary to print the program.

   [XEQ] [ALPHA] P R P [ALPHA] [ALPHA] C I R C L E

   On the HP 48, select the capture mode from the menu.

   CPTR

   On the HP 41, complete the key sequence to print the program through the infrared printer.

   [ALPHA]

The HP 41 prints the CIRCLE program through the Infrared printer module, and the HP 48 captures the program, and automatically compiles it. This should take no more than thirty seconds.

If the transfer seems to take much longer, check if you have set everything up as described above, and try again. If it still does not work, see Chapter 3 of this manual for further advice.

You will see the message `Line` *n* as each line is compiled (*n* counts from 1 to 5). The HP 48 displays the program name `CIRCLE`. The program will be stored in the current HP 48 directory as an object named CIRCLE.

Now run the compiled program.

Enter the HP 41 environment.

`HP41`

Place the HP 41 keyboard overlay on the HP 48 keyboard.

Enter the number 2 in register X.

2

Run the program.

[XEQ] [α] C I R C L E [α]

(Note that [XEQ] is the HP 48 key marked [EVAL].)

The answer 12.5664 should appear in register X. (This is the answer in FIX 4 mode.) You have run an HP 41 program on your HP 48 — well done!

If you want to run the program again, you can type another number into the X register, then press the [R/S] key (the HP 48 [ON] key).

Using the HP 48 Editor on an HP 41 Program

Now that you have captured an HP 41 program, you can modify it. First, you must turn the program into an HP 48 text string:

Exit the HP 41 environment.

`EXIT`

Remove the HP 41 keyboard overlay.

Display the global variables.

VAR

Select the CIRCLE menu key.

CIRCL

Enter the HP 41 control menu.

↰ LIBRARY CTL41

Decompile the program and put it on the HP 48 stack.

→TXT

Use the HP 48 EDIT functions to edit this string. For information on the HP 48 editor refer to your HP 48 Owner's Manual.

For this example, turn the program into one that calculates the surface area of a sphere:

```
01  LBL  ᵀSURFAS
02   X^2
03   PI
04   *
05   4
06   *
```

Replace the name CIRCLE with SURFAS.

▼ ↱ ▶ ◀ ← ← ← ← ← ← α α S U R F A S α

Position the cursor on top of the program's final double quote (use ▼) and add lines 05 and 06. Follow each line with a linefeed (↱ .).

0 5 SPC 4 ↱ . 0 6 SPC × ↱ . ENTER

Turn the string back into an HP 41 program.

→41

The program appears as a Library Data object in level 2, and its label "SURFAS" appears as an identifier in level 1.

Store the program as a global variable.

STO

Return to the HP 41 environment.

[⬅] [LIBRARY] HP-41 HP41

Put the HP 41 keyboard overlay on the keyboard.

Enter 2 in register X and then run the program.

2 [XEQ] [α] S U R FA S [α]

The surface area of a sphere with a radius of 2 is 50.2655.

You can use these steps above to create new HP 41 programs on your HP 48. Instead of capturing a program, you can key in the program as a text string, then compile it. Use the ['] key instead of quotation marks. Simple programs can be entered without line numbers or linefeeds.

To transfer data from the HP 41 to the emulated HP 41 on your HP 48, use the same procedure you used to transfer the CIRCLE program, except use PRREG or PRREGX instead of PRP.

Congratulations! You have captured, compiled, modified and run an HP 41 program on your HP 48. Chapters 2-5 cover all these topics, and others, in more detail.

Terminology Used in This Manual

This manual assumes that you have some experience using the HP 41. You should be familiar with terms used to describe how the HP 41 works. However, a few important words relate only to the HP 41CV Emulator card, or have different meanings depending on whether they refer to the HP 41 or the HP 48.

HP 41

The card is designed for use with programs that work on an HP 41C or an HP 41CV, and many programs that work on an HP 41CX. Unless otherwise specified, "HP 41" refers to all three models.

Modules, Pacs, and Peripherals

Users can extend the features of the HP 41 by plugging extra equipment into the four ports at the top of the HP 41. Most modules provide additional instructions for the HP 41, written in **Read Only Memory** or **ROM**. These modules are called ROMs or **Application Pacs**. Some modules provide additional memory for the user, called

User Memory, or **Random Access Memory (RAM)**. You can also plug **peripherals** into these slots, such as the HP 41 Card Reader or the Optical Wand.

HP 41 Memory
HP 41 User Memory is used to store programs and data. The HP 41 holds data in the **stack**, the **ALPHA register** and **data storage registers**. Each **data storage register** can hold one number or one **alpha string** (six characters).

HP 41 Memory in the HP 48
The HP 48 also has a stack and user memory, but HP 48 user memory is not organized in registers like the HP 41. The HP 41CV Emulator card reserves a special area of HP 48 user memory to be used as the HP 41 user memory. This emulated HP 41 user memory is organized in registers. Thus, in this manual, the term 'HP 41 registers' is used to describe data registers inside an HP 41 and those reserved by the emulator card in the HP 48.

Catalogs (CAT 1, CAT 2, CAT 3)
In the HP 41, CAT 1 is a list of the **global labels** in programs stored in User Memory. CAT 2 is a list of functions and programs provided by modules and peripherals plugged into the HP 41. It also lists the additional HP 41 instructions which are built into the HP 41CX, but not into the HP 41C or HP 41CV. CAT 3 is a list of the functions built into the HP 41.

Operations, Programs, Commands and Functions
The HP 48 User's Manual classifies HP 48 actions into **operations, commands, functions,** and **analytical functions**. This manual classifies HP 41 actions in a similar way.

- Anything the HP 41 can do is called an **operation**. This includes things such as **SHIFT** or **GTO.**, which are not in any of the HP 41 catalogs and cannot be included in programs. It also includes actions such as **DEL** or **SST**, which are in CAT 3 but cannot be included in a program.

- Any operation that can be used in a program is called a **command**. There are four types of commands:

  Functions built into the HP 41.

  Functions in plug-in modules.

  Programs stored in RAM.

Programs in plug-in modules (listed in CAT 2 with the text symbol ^{T} preceding the name).

- Any command that appears in a program as a single word or symbol is called a **function**. The following are examples of functions: **+**, **SIN**, **BEEP**, and **AVIEW**. A function can have a **postfix**. For example, STO 12 is made up of the **function** STO and the **postfix** 12. On the HP 48 the word "function" is reserved only for mathematical functions. Since this manual is for the HP 41 Emulator card, function retains its HP 41 meaning.
- An HP 41 **program** is a list of HP 41 commands stored one after another. Each command is called a **program step**. **User programs** are stored in and execute from RAM. You can modify user programs. **ROM programs** come in plug-in modules. They execute from the module memory and cannot be modified in the module.

Emulation
The card is called an HP 41CV Emulator card because it creates an environment in the HP 48 that **emulates** (behaves like) an HP 41CV. When you first begin to use the card, it provides the operations built into an HP 41CV, a stack, an ALPHA register, flags, and 100 data registers. Later, you will see how you can change the amount of memory available for use by the HP 41CV Emulator card with the SIZE command.

Note

HP 41C OWNERS
An HP 41C is exactly like an HP 41CV except that it has less built-in memory. Programs sent from an HP 41C to the HP 41CV Emulator card will work exactly as they would if you were using an HP 41CV.

The card also emulates the operations built into the HP82242A Infrared Printer Module, except the following: PRPLOT, PRPLOTP, PRAXIS, REGPLOT, and STKPLOT. The Infrared Printer Module provides the print operations that were in the HP82143A printer and the HP-IL module, so the card emulates their printing operations too.

Note

HP 41CX OWNERS
An HP 41CX has a set of built-in Time functions, a set of built-in Extended Functions, and 124 Extended Memory registers. The HP 41CV Emulator card provides some of these (listed below), and Chapter 5 explains how to add the others as extensions to the HP 41CV Emulator card environment.

The following HP 41CX extended functions are built in to the Emulator: CLKEYS, PASN, PSIZE, RCLFLAG, SIZE?, STOFLAG, and ΣREG?. Apart from these, the card does not emulate operations from any other plug-in modules or peripherals. If you need to use commands from HP 41 plug-in modules and peripherals, you can write them yourself using HP 48 functions. Such commands are called **extensions**.

General Principles of the Emulator Card

This section explains the general principles of the card, and tells you where to find full details.

Capturing HP 41 Programs and Data

An HP 41 program can be entered into the HP 48 three ways: transferred through the HP 41 Infrared printer module, typed on the HP 48 keyboard, or read from a computer. Capturing from the Infrared printer module is the easiest method.

HP 41 programs often use data stored in the HP 41. The HP 41CV Emulator card allows you to capture HP 41 data as well as programs. For information on how to transfer programs and data, see Chapter 3.

The HP 41 Environment

The HP 48 uses different levels of operations called **environments**. When you want to use the HP 48 to run HP 41 programs, the HP 41CV Emulator card sets up an **HP 41 environment**. In this environment the display shows you information concerning the HP 41 stack and programs, and the keyboard behaves like an HP 41 keyboard.

Once you have entered the HP 41 environment, you can run compiled HP 41 programs. You can store data in registers and do keyboard calculations as if you were using an HP 41. You can also alter the HP 41 environment. For example, you can set RAD or DEG mode. (These HP 41 modes are similar to, but independent of, HP 48 modes.)

Information about the HP 41 environment is stored as a set of HP 48 objects. Chapter 2 shows how you can modify the environment by changing some of these objects.

Note, however, that if you want to key in HP 41 programs on the HP 48, you must first exit the HP 41 environment. HP 41 program lines are entered as HP 48 text strings, and then compiled into programs for use in the HP 41 environment. Chapter 4 shows you how to compile and decompile programs.

Running HP 41 Programs

The main purpose of the HP 41CV Emulator card is to let you run HP 41 programs on an HP 48. Capturing HP 41 programs and data, setting up an HP 41 environment, and compiling HP 41 programs, are all needed so that you can run HP 41 programs. You can also carry out keyboard calculations to provide numbers needed by the programs, and do keyboard calculations on the results of programs.

Compiled HP 41 programs are kept in an HP 48 directory. This directory is similar to CAT 1 on the HP 41. Programs can call other HP 41 programs as subroutines. They can also call HP 48 programs, and HP 48 programs can call compiled HP 41 programs.

Extending the HP 41 Environment

You can extend the HP 41 environment by calling HP 48 programs. You can add functions to your HP 41 environment by writing HP 48 programs to carry out the tasks you require, and then calling these programs from inside the HP 41 environment. Details of how this is done are given in Chapter 5.

2

The HP 41 Environment

This chapter introduces the following:

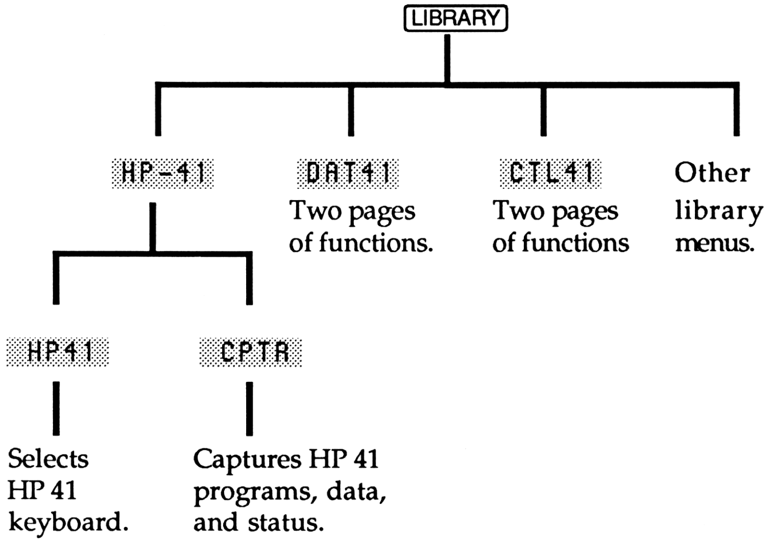
- The HP 41 environment, keyboard, and display.
- Alarms and calculations in the HP 41 environment.
- The HP 41 environment menus.
- Emulator variables and setup options.

Introducing the HP 41 Environment

The HP 41 environment is like a house – you can be inside or outside of the environment. Inside the environment you are working with an emulated HP 41. You can run HP 41 programs, carry out keyboard operations, and make internal changes. Outside the environment you can see its overall structure and can modify the structure, but you cannot make full use of what is inside. This chapter explains how the features of the emulator, inside and outside, can be used.

The menus provided by the emulator let you use the environment from the inside, and from the outside, through sets of commands in each menu. In effect, the Emulator extends the built-in command set of the HP 48 by providing a library of additional menus. The HP 48 Owner's Manual gives more details of libraries in the section called *Using Application Cards and Libraries.*

The Emulator menus are stored inside the Emulator Card's ROM. The following figure shows the layout of the Emulator's menus. If a menu contains more than six entries, it has more than one "page." For example, the DAT41 menu contains two pages of menu labels. You can move between pages of labels with NXT and ⬅ PREV.



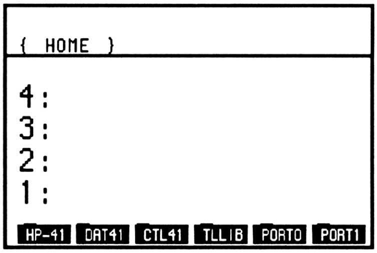
Layout of HP 41 emulator menus

Entering the Environment

This section describes the HP41 menu. Chapter 3 gives details of CPTR .The DAT41 and CTL41 menus are covered in Chapters 4 and 5.

To use the HP 41 Emulator menus, select them from the HP 48 LIBRARY menu.

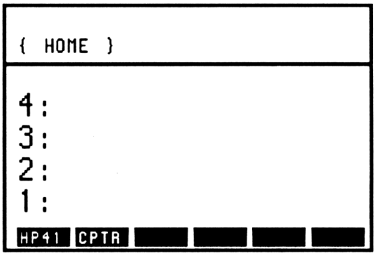
[⬑] [LIBRARY]


```
{ HOME }
4:
3:
2:
1:
HP-41 DAT41 CTL41 TLLIB PORT0 PORT1
```

If you do not see the menu keys shown above, press [NXT] to see if they are further along in the LIBRARY menu. If you can not find HP-41 anywhere in the library menu, make sure the emulator card is correctly plugged into your HP 48 (see Chapter 1). If this does not help, you might have attached the emulator library to a subdirectory. Make sure the subdirectory is in your current path. Chapter 1 explains how the Emulator library is initially attached to the HOME directory, and the HP 48 Owner's Manual explains how you can detach the library and then attach it to a subdirectory.

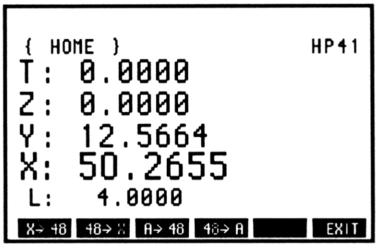
Once you have found the HP-41 menu key, press it to gain access to the HP 41 emulator.

HP-41


```
{ HOME }
4:
3:
2:
1:
HP41 CPTR
```

Enter the interactive HP 41 emulator mode. If you have not used the Emulator before, you will see initialization messages.

HP41



```
T:  0.0000
Z:  0.0000
Y:  12.5664
X:  50.2655
L:   4.0000
```

The HP 41 Environment and Keyboard

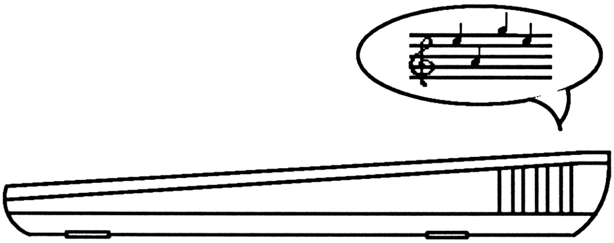
When you pressed the HP41 key of the HP-41 menu it put you in the HP 41 environment. You can see the contents of registers X, Y, Z and T in the display. Below them is L, the LASTX register. Under L is the menu line. The top line displays annunciator information. The HP 41 environment annunciator information is separate from the six symbolic HP 48 annunciators at the very top of the display.

The HP 41 environment provides the same mathematical functions as the HP 41. The range of numbers you can use is larger on the HP 48 (see page 34).

When you are in the HP 41 environment, attach the HP 41 keyboard overlay to the HP 48. The overlay lies on top of the HP 48 keyboard and has six tabs that fit into the slots on the left and right of the keyboard.

As a first example of using the HP 41 environment, try pressing the orange Shift key and then the key marked 4.

[SHIFT] [4]



You will hear the familiar HP 41 BEEP. The function name BEEP is written in yellow above the [4] key, just as on an HP 41. Press the same keys again, but this time hold [4] down. You will see the function name (BEEP) displayed on the bottom line, and then the function will be cancelled, and Null appears. This is exactly the way an HP 41 behaves.

The emulator keyboard lets you carry out calculations and other operations from the HP 48 keyboard, as if you were using an HP 41. Keyboard operations can be used to step through a program, but programs cannot be written from the keyboard while in the HP 41 environment. The capture, translation, and use of HP 41 programs are described in Chapters 3 and 4.

Like the HP 41, the HP 41 environment allows keys to be reassigned for use in USER mode. The Emulator kit does not include spare overlays. If you want to use a customized keyboard, you can purchase HP 48 overlays and mark them for use with your HP 41 key assignments. The overlays do not reach the top row of keys, so it is best to avoid reassigning top row keys. Also, top row keys are automatically used for local HP 41 program labels in HP 41 user mode, so it may be confusing if they are assigned to act in some other way.

The keyboard in the HP 41 environment is similar to a real HP 41 keyboard. A diagram of the HP 41 environment keyboard is shown on the flap in the back cover of this manual, fold it out while you read the next few pages of this section. The keyboard is arranged as follows:

1. **Number entry**. The number entry keys 0 through 9, point, sign change and enter exponent are laid out as on the HP 41, using the HP 48 keys. The [ENTER] key and the backarrow key [←] retain their relative positions and their purpose. If the backarrow is used to delete the contents of the X register then stack lift is disabled (stack lift enable and disable work as they do on the HP 41).

2. **Mathematical functions.** The mathematical functions most often used in scientific and technical calculations use the corresponding HP 48 keys as well. These keys are add, subtract, multiply, divide, the trigonometric and inverse trigonometric functions, [$\sqrt{x}$] [y^x] and [$1/x$].

3. **Other keyboard functions.** As many other functions as possible are positioned in the same way as on the HP 41. For example, the shifted keys in the rows [4] [5] [6], [1] [2] [3] and [0] [.] [SPC]

have the same functions as on the HP 41. Other HP 41 functions are positioned close to their HP 41 locations wherever this is possible.

4. **The SHIFT key.** The HP 41 SHIFT key is replaced by the orange HP 48 [↰] shift key. The blue shift key [↱] is ignored except for three uses: [↰] [↱] puts up the HP 48 interaction menu, which includes the EXIT function to let you leave the HP 41 environment; [↱] [CST] always toggles USER mode, even if you have reassigned the [CST] key; [↱] [ON] turns the HP 48 off.

5. **Arguments.** Functions that require arguments, such as RCL or TONE, work as on the HP 41. Prompts are filled in the same way, with the appropriate number of digits. The [SHIFT] ([↰]) key is used as an "indirect" key — pressing RCL and then [SHIFT] produces `RCL IND __`. In the same way, the [.] key prompts for stack register operations. For example, executing VIEW and then pressing [.] produces the prompt `VIEW ST _`.

   As on the HP 41, the top two rows of keys can be used to provide arguments. For example, press [↰] [PARTS] [NXT] TONE [CST] to get the TONE 9 function. The top rows have 6 keys instead of 5 as on the HP 41, so the keys provide arguments from 01 up to 12. Where a one-digit postfix is expected, the [NXT] key acts as the 0 key. In USER mode the top two rows are automatically assigned to local labels in the current program.

6. **SIZE, STO, and RCL.** The **SIZE** function prompts for three digits which allows SIZE to be set to any value up to 999 registers. Pressing [EEX] will prompt for four digits (as with GTO. on the HP 41). STO and RCL prompt for only two digits, so if you want to access any register beyond 99 you must use indirect addressing (STO IND and RCL IND) just like the HP 41. STO works with the arithmetic keys as on the HP 41 to give ST+, ST-, ST*, and ST/.

   If you attempt to create more registers than can fit in the available HP 48 user memory, you will get the following error message:

   ```
   No  Room
   ```

   If HP 48 memory is very short, you will get the same message even if you try to reduce the SIZE setting. (There has to be enough room to hold both the new registers and the old registers, so that data can be transferred from one to the other.) Delete HP 48 variables to make more room. If the old registers are not needed, delete the variable that contains them (see page 50), or add more HP 48 memory in some other way, then execute SIZE again.

7. **[ON] and [R/S].** The HP 48 [ON] key replaces the HP 41 [ON] toggle key, but it has to be combined with [↱] to turn the HP 48 off. In HP 41 emulator mode, this key acts as the HP 41 [R/S] key. If turned off in the emulator mode, the HP 48 remains in the emulator mode when turned on again. The HP 41 [R/S] key runs, stops, or restarts a program.

8. **ALPHA.** The HP 48 [α] key replaces the HP 41 [ALPHA] toggle key. Below and to the right of each key is its alpha mode action. Pressing [SHIFT] before the keys marked with letters produces lower case. All letters can be written in lower case. The numeric keys produce numbers when they are not shifted, and the symbols marked in yellow below them when shifted. The [ENTER] key acts as the Alpha Append command when it is not shifted and as the caret ^ when shifted.

   ASTO, ARCL, AVIEW, and ASHF are also available. Pressing [SHIFT] and then [α] in alpha mode produces the name CHR and a prompt for a two-digit hexadecimal number (use the top row of keys for A to F). The number you give is treated as a character code letting you type in characters that are not on the keyboard. This replaces the HP 41 XTOA Extended Function. For example, to enter an ampersand (&) whose character code is 38[1] (26 in hexadecimal), press

   [SHIFT] [α] [2] [6]

   Since the Alpha Append symbol "⊢" is not available on the HP 48, the emulator treats the character "→" as an append symbol. If you want to begin an alpha string with"→" , put "→→" at the beginning. If you want a text string beginning with "→" to

[1]For a list of character codes, refer to the appendixes of the HP 48 manual.

replace the current contents of the ALPHA register you will have to use CLA first.

The HP 48 character "▒" (vertical hatched block) can also be used as an Append character. It has the same character code as the HP 41 Append symbol[2].

9. **Special keys.** In the first two rows below the menu keys, the shifted HP 48 [CST] key is used for switching into USER mode. It works like USER on the HP 41. USER mode is entered when this key is pressed. (Press the key again to leave USER mode. If you have assigned something to the [CST] key or there is a LBL I in the current program, then use [↱] [CST] to leave USER mode.) The [VAR] key acts as RCL when it is not shifted and executes SIZE when shifted. The [▼] key acts as RDN when it is not shifted and as R↑ when shifted. Other keys (GTO, STO, XEQ, x<> and x<>y) have their action marked in white above them. Keys that have no special shifted function assigned to them will still carry out their function if [SHIFT] is pressed first. This means that in USER mode, something can be assigned to each of these keys, and they will still carry out their normal action when preceded by [SHIFT]. Conversely, functions can be assigned to a shifted key, and the key will carry out its normal action when unshifted.

   [SST] and [BST] let you step through programs as on the HP 41. Since there is no program entry mode, you cannot use [SST] to move through a program without executing it. You can use [GTO] [.] to go to a particular program line, instead. As with SIZE, you can press [EEX] after [GTO] [.] to extend the prompt to four digits.

   The HP 48 [DEL] key does not correspond to any HP 41 function and is ignored. [SHIFT] [DEL] is used for the percent function in normal mode and for the percent symbol (%) in alpha mode.

   During numeric entry, [SPC] can be used to complete a number *and* to put that number in register X. Unlike [ENTER], this does not copy the number to register Y and does not disable stack lift.

10. **Menus.** Most other functions are accessible from the menu options LOGS, PRGM, PRINT, xTESTS, CATALOG, CLEAR, Σ, ANG, MODES, FLAGS, PARTS, and HP 48. This saves the effort of using [XEQ] and spelling the functions out. Keys that select one of these menus have the menu name printed on a lighter background

[2]Character codes 7F, 8D, and 94 (hexadecimal) are all treated as append characters.

on the keyboard overlay. The list of functions in each menu is given in this chapter in the section called *The HP 41 Menus*. The $\boxed{\text{NXT}}$ and $\boxed{\text{PREV}}$ keys are used to let you move from one page to another in menus with more than six functions. All functions except STK and EXIT can be executed by pressing $\boxed{\text{XEQ}}$ $\boxed{\alpha}$, and spelling out their names (STK can be executed by setting or clearing flag 86).

A few HP 41 functions are not available from the keyboard nor from a menu. They can be executed from programs and by use of $\boxed{\text{XEQ}}$ followed by their names. The reason each function is treated in this way is given with the function name.

END	This function is normally used only in programs, but it can be used from the keyboard to remove a return address from the return stack. This is sometimes required when a program has been stopped with $\boxed{\text{R/S}}$.
OFF	This is normally done by $\boxed{\rightarrow}$ $\boxed{\text{ON}}$, but it can be spelled out.
PROMPT	This is equivalent to AVIEW when executed from the keyboard.
AON AOFF	These are available but it is easier to use $\boxed{\alpha}$ from the keyboard.
LBL PSE STOP	These are available, but they are only meaningful in programs.

The remaining HP 41 functions are not available at all, for the reasons given below.

COPY	There are no HP 41 modules to copy from.
DEL	HP 41 programs can not be edited in the HP 41 environment.
PACK	The Emulator packs HP 41 programs when they are created, and as there is no program editing mode there will be no nulls left in program memory by deletions.
GTO..	Like PACK, GTO.. is not needed to pack programs, since they are already packed. The other use of GTO.., to go to the end of program memory, is not necessary since each

program is stored as a separate HP 48 object. Do not confuse GTO.. with GTO. which is available from the keyboard.

CLP	Programs are created from HP 48 menus, not from within the emulator mode, and they are deleted from the HP 48 too. To delete a program, purge the variable that stores it.
ON	The HP 48 cannot be forced to stay on after 10 minutes of inactivity. An alternative to ON is suggested at the end of this chapter.

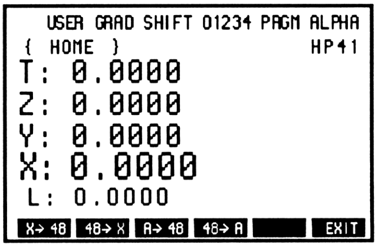
Finally, there is no equivalent for the [PRGM] toggle key, since there is no program entry mode.

The HP 41 Environment Display

When you enter the HP 41 environment, you see the contents of registers X, Y, Z, and T in the display. Below them is L, the LASTX register. X is shown in the large characters used to display the HP 48 stack, Y, Z, and T are in medium size characters, and L is in small characters. Below L is the currently selected menu line, and at the top is an area for HP 41 annunciator information.

All the HP 41 annunciators except BAT are provided, in the same order as on the HP 41. There is no BAT symbol because the HP 48 has its own low battery indicator. The HP 48 annunciators are not used in HP 41 emulator mode, except for the low battery warning, though the shift and busy symbols may turn on momentarily. An extra symbol, to show that the HP 48 is in the HP 41 emulator mode, is shown on the same line as the current HP 48 directory.

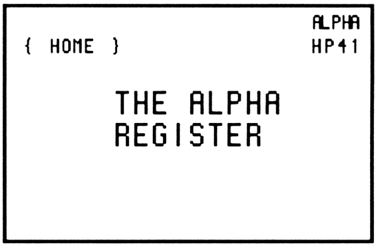
Here are the HP 41 annunciators.➡



```
USER GRAD SHIFT 01234 PRGM ALPHA
{ HOME }                    HP41
T: 0.0000
Z: 0.0000
Y: 0.0000
X: 0.0000
L: 0.0000
```

When you enter alpha mode, the display of the stack is replaced by a display of all 24 characters in the ALPHA register.

The ALPHA register is displayed as two rows of 12 characters each. The menu line is not used in alpha mode. To leave alpha mode press [α].


```
{ HOME }                    ALPHA
                            HP41
        THE ALPHA
        REGISTER
```

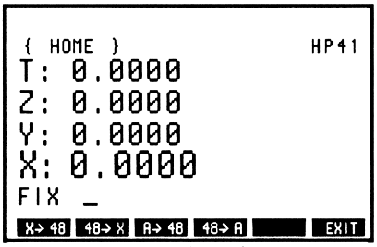
HP 41 error and warning messages are shown below the stack, as are messages shown by VIEW, PROMPT, AVIEW and prompts (for example XEQ__). Up to 22 characters can be shown. Messages of 23 or 24 characters scroll across the display.

You can make the display look like the HP 41 display by pressing STK in the second page of the [MODES] menu. A single line is used to display X, the ALPHA register, program lines and messages. The contents of the ALPHA register are scrolled across the display if the register contains 23 or 24 characters. The HP 41 annunciators are shown below the one-line display, again as on the HP 41, and above the menu keys. Pressing STK a second time returns the display of the full stack. You can also set this mode by clearing flag 86.

The following examples show typical displays.

1. Set the display mode to FIX 2.

   [SHIFT] [FIX] ([←] [1])

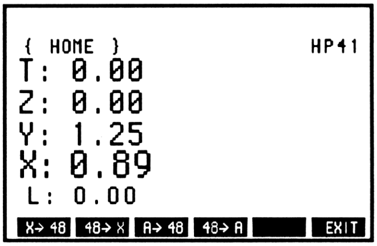

```
{ HOME }                    HP41
T: 0.0000
Z: 0.0000
Y: 0.0000
X: 0.0000
FIX _
X→48 48→X A→48 48→A      EXIT
```

2. Now type 2 to fill the prompt, or press the second key from the left in the top row.

Using this display mode, calculate the value of $\sqrt{16 \times (1.25-0.89)}$

1. Enter 1.25 followed by .89

   1.25 [ENTER] .89

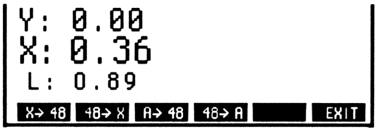


```
{ HOME }                 HP41
T: 0.00
Z: 0.00
Y: 1.25
X: 0.89
 L: 0.00
```

Note that, as on the HP 41, stack lift was disabled when ENTER was pressed, so the number .89 replaces the contents of register X.

3. Subtract .89 from 1.25.

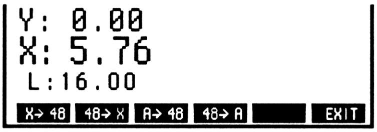
   [-]



```
Y: 0.00
X: 0.36
 L: 0.89
```

4. Multiply 0.36 by 16

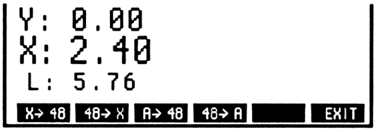
   16 [x]



```
Y: 0.00
X: 5.76
 L:16.00
```

5. Take the square root of the product.

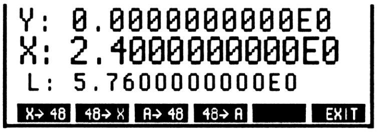
   [$\sqrt{x}$]



```
Y: 0.00
X: 2.40
 L: 5.76
```

6. Change the display to ENG 10 mode.

   [SHIFT] [ENG] [VAR]



```
Y: 0.0000000000E0
X: 2.4000000000E0
 L: 5.7600000000E0
```

Go back to the HP 41 default by pressing [FIX] [4].

Alarms

If an HP 48 control alarm goes off while the HP 41 environment is active, the current function is completed and the HP 48 exits the HP 41 environment. This means that the control alarm can execute HP 48 commands without being affected by the HP 41 environment.

When the control alarm actions are completed, the HP 48 does not return to the HP 41 environment. If you wish to continue using the HP 41 environment, you can return to it from the LIBRARY menu. The stack contents and other HP 41 variables are retained when the HP 48 exits the HP 41 environment, so you can continue to use them after the alarm. However, remember that the control alarm might have changed the current directory.

Note A control alarm can modify HP 48 settings, including the HP 41 environment variables, while you are in the HP 41 environment. Control alarms take you out of the HP 41 environment, so you can determine what they have done. A control alarm variable should not have the same name as an HP 41 environment variable.

The exception to the above note is if you want a control alarm to affect the HP 41 environment. For example, if you are using an HP 48 control alarm to emulate an HP 41 Time module alarm, as described in Chapter 5.

HP 48 appointment alarms do not alter HP 48 variables. Appointment alarms display (optional) messages and sound tones, but leave do not affect the HP 41 environment. You can acknowledge an appointment alarm by pressing a key while the alarm is active. If the alarm is not active, the key action will be executed in the HP 41 environment.

Accuracy in the HP 41 Environment

The HP 41 environment provides the same mathematical functions as the HP 41, so you can do the same calculations. However, there are some differences that should be noted. The HP 41 stores numbers with

10 mantissa digits and 2 exponent digits, whereas the HP 48 uses numbers with 12 mantissa digits and 3 exponent digits, even in the HP 41 environment. Both calculators actually do their calculations using extra digits and then round their results before putting them back on the stack.

A result calculated in the HP 41 environment will have two more digits than the result of the same calculation on an HP 41. For example, an HP 41 gives the square root of 3 as 1.732050808 while an HP 48 gives 1.73205080757. The HP 48 result is more accurate, but does this matter? If you square both these numbers, the HP 41 gives 3.000000001 and the HP 48 gives 3. The difference is very small, but if you use **x=y?** to examine the result then the HP 41 and the HP 41 emulator will give different answers to the test. It might seem that this shows the HP 48 is always more accurate than the HP 41, but if you try the same calculation (square root, then square) with the number 7 then the HP 41 gives 7 and the HP 48 gives 6.99999999998.

Thus you cannot expect either the HP 41 or the HP 48 to give exact results. Both do their best, within the number of digits they use. If you have an HP 41 program which relies on getting an exact result in the HP 41 then the HP 41 emulator might give a different answer.

The HP 41 uses numbers with an exponent between -99 and +99. The HP 48 uses numbers with exponents between -499 and +499. Calculations which the HP 41 cannot manage might come out correctly on the HP 48. When the HP 41 and the emulator give significantly different results, usually the HP 41 program had an undetected error. For example, on an HP 41 the following program lines should leave the contents of X almost unchanged (X is an integer between 0 and 70.)

```
FACT
LASTX
1
-
FACT
/
```

If flag 24 is set, and X is 70, then the result of this calculation will be about 58.4 When the program is transferred to the emulator, the result will be different and the user may suspect a mistake in the translation or in the emulator. In fact, the mistake was in the original HP 41 program. Leaving flag 24 set allowed the HP 41 to accept the factorial of 70 even though it overflowed and gave a result of 9.999999999E99 instead of signalling an error. The wider range of the HP 48 allowed it to calculate the correct answer without any overflow.

Differences might arise with results that underflow to 0 on the HP 41, but give a correct result on the HP 48. Also, many slightly inaccurate results can combine and give a larger cumulative inaccuracy.

Accuracy is discussed in detail in the HP-15C Advanced Functions Handbook (see Appendix B). If you are transferring a calculation from an HP 41 to the emulator then questions of accuracy should already have been dealt with when the calculation was being carried out on the HP 41.

The HP 41 Menus

The HP 41 emulator menus and their functions are listed below. Menus which provide ordinary HP 41CV functions are listed first. They are presented with few comments, since HP 41 users know these functions already. Some of the functions affect HP 41 flags. A table of flags is given in Appendix C.

▨ denotes a menu key which is not used.

LOGS	LN	E^X	LN1+X	E^X-1	LOG	10^X

Note that, unlike the HP 41, the emulator does not have e^x on the keyboard. The E^X menu item is provided instead.

*x*TESTS	X=Y?	X≠Y?	▨	X<Y?	X<=Y?	X>Y?
	X=0?	X≠0?	▨	X<0?	X<=0?	X>0?

CLEAR	CLD	CLST	▨	CLΣ	CLA	CLX
	CLRG	▨	▨	▨	▨	▨

Note that CLRG is on a separate page. CLRG can destroy a large amount of data, so it has been separated so it will not be pressed by mistake.

Σ	Σ+	MEAN	SDEV	CLΣ	ΣREG	Σ-

ANG	D-R	R-D	HR	HMS	HMS+	HMS-

FLAGS	SF	CF	FS?	FC?	FS?C	FC?C

Most of the flags from 12 through 55 have the same purpose in the emulator as on the HP 41. The emulator uses additional flags, numbered 56 through 87. Details are given in Appendix C.

PARTS	RND	ABS	SIGN	MOD	FRC	INT
	DSE	ISG	TONE	FACT	OCT	DEC

Note that this menu includes a few functions which are not, strictly speaking, PARTS functions.

The remaining menus contain functions that are not built into (or function differently on) the HP 41CV.

CATALOG
When you press this key it prompts for a one digit number, as on the HP 41. The numbers 1, 2, 3 and 7 are accepted, all others are treated as if you had pressed 3.

When you press a number, CATALOG displays a menu, instead of running through a list as it would on the HP 41. You can move through the menu with [NXT] and [PREV], and you can execute any item from a menu by pressing the corresponding menu key.

CAT 1 displays a menu of global labels, as on the HP 41. All the global labels in each program in the current directory are displayed, followed by an END at the end of each program. You can press any menu key corresponding to a global label and the program will begin execution at that label. If a program does not begin with a label, you can press the END of that program, then [SST], to get to the first step of the program. Then you can single step through the program or run it by pressing [R/S]. If you have a library of standard HP 41 programs which you use often, you can put these in a special HP 41 program library directory (see the section in this chapter called *Emulator Variables and Setup Options*). CAT 1 displays your library of programs before those in the current directory, and if you type XEQ "name", then the emulator looks for "name" in the library directory, if it does not find it in the current directory.

CAT 2 displays a menu of HP 41 extension commands, like CAT 2 on the HP 41. The extension commands are HP 41 functions that you can add to the HP 41 emulator. Chapter 5 describes how you can write extension commands, also called XROM commands, for use with the emulator. These commands are kept in an HP 48 directory called HP41XROM.

CAT 3 displays a menu (20 pages) of all the HP 41 functions built into the HP 41CV emulator (except for STK and EXIT). This is like CAT 3 on the HP 41, except that a few functions are missing, and a few new functions have been added.

CAT 7 displays a menu of standard printer and extended functions (XROM functions). They are listed separately from CAT 2, which lists functions you can create and add to the emulator. The catalogs are numbered in the order in which they are searched when a command XEQ "*name*" is entered from the keyboard.

PRGM
[PRGM] displays a menu of the names of HP 48 objects that contain HP 41 programs and which are stored in the current HP 48 directory or in the PRG41 directory. These objects are explained in Chapters 3 and 4.

Abbreviated menu keys have their HP 41 function names below them.

MODES	RDX,	SEP■	DEG■	RAD	GRAD	HUSH
	STK■	MAN■	NRM	TRC	PON	POFF■
	STOFL	RCLFL	PSIZE	SIZE?	ΣRG?	PASN
	STOFLAG	RCLFLAG			ΣREG?	
	CKYS					
	CLKEYS					

The first page of the modes menu provides functions to control settings of the HP 41 emulator.

RDX, sets the current decimal mark and digit separators to comma (,) or period(.) (equivalent to CF 28 and SF 28).

SEP toggles a box in the menu label denoting use (or lack) of digit separators in numeric display. When the box is displayed, it is the equivalent of SF 29, lack of a box is the equivalent of CF 29.

DEG , RAD , or GRAD will turn on a box in the selected menu label, turning off a box in either of the other two. This selects the angular mode in use. This is equivalent to using the like named HP 41 functions, and affects flags 42, 43, and the display annunciators.

HUSH toggles a box in the menu label indicating that the audio is disabled, or enabled (Flag 26).

The second page of this menu controls the stack display and printer functions of the HP 41 emulator.

STK toggles the display between showing the stack (X, Y, Z, T, L) and showing X only, as on the HP 41. When stack display is selected, a box is shown in the menu label.
STK is not programmable, and cannot be executed with XEQ, but you can achieve the same effect by setting flag 86 to select stack display, or clearing flag 86 to select X only mode.

MAN , NRM ,or TRC turn on a box in the selected menu label, turning off a box in either of the other two. This selects the print mode (manual, normal, trace) and affects flags 15 and 16.

PON and POFF enable and disable printing.
A box in the PON label denotes printer enabled (flag 55 set). A box in the POFF label denotes printer disabled (flag 55 clear).

The third and fourth pages provide additional functions to let you check and manipulate some of the HP 41 modes. The functions on these pages are not built into the HP 41CV, they are Extended Functions.

STOFL (full name is STOFLAG), lets you restore flags 0 through 43 from a string recalled by RCLFL . Put a RCLFLAG string in register X, then execute STOFLAG. The flags that STOFLAG restores are not selectable; flags 0 through 43 are restored. A more complete version of STOFLAG could be written as an XROM function.

RCLFL (full name is RCLFLAG) recalls the status of all the flags 0 through 43 and puts it into the X register as a text string. This lets you recall the current status of all these flags, then change some of them, and later restore the original modes from this string. The string can be stored in a numbered data register, then later recalled to X, and the flags can be restored by STOFLAG.

PSIZE and SIZE? SIZE? returns the current SIZE setting to X, lifting the stack (unless stack lift is disabled), and leaving register L unchanged. PSIZE is a programmable version of SIZE. You can use SIZE? and PSIZE in a program so that the program can set the SIZE it needs, without stopping and asking the user to do this. PSIZE sets the SIZE to the number given in register X, ignoring the sign and any fractional part. The same limitations and warnings apply to PSIZE as to SIZE. (See the description of SIZE on page 26.)

ΣRG? (full name is ΣREG?), recalls the register number at which the HP 41 statistics registers begin. This lets you recall information from individual statistics registers. For example, to find the number of statistical readings accumulated so far (the number of data points), use the following steps:

ΣREG? 5 + RCL IND X

The sixth menu label is PASN . PASN is a programmable version of ASN. This lets a program make key assignments, instead of asking the user to make them. Put the name of a function or global label in the ALPHA register, put the keycode for the required key in register X, and execute PASN to make the assignment. This deletes any previous assignment made to that key. Keycodes are specified as on the HP 41, as a number *rn*. The row number is specified by *r*, beginning with 1 at the top. The key position within the row is specified by *n*, beginning with 1 for the leftmost key. Remember that the keycode for a shifted HP 41 key is the keycode with a minus sign.

The last of these functions is **CLKEYS**, represented by the menu label CKYS . This function deletes all key assignments. It is on a separate menu page to prevent the key from being pressed by mistake. The function is programmable, and can be used in a program before PASN, to delete all assignments before new ones are made. An alternative to using this function is to hide the key assignments by changing the name of the HP 48 variable that holds them. (See the section in this chapter called *Emulator Variables and Setup Options*.)

The **PRINT** menu has five pages of menu labels. Some HP 41 printer functions have names that do not fit on an HP 48 menu label. Abbreviated menu keys have their HP 41 function names below them.

PRINT	PRP	PREG	PRGX	PRΣ	PRA	PRX
		PRREG	**PRREGX**			
	LIST	PKEY	PSTK	PFLG	PBUF	ADV
		PRKEYS	**PRSTK**	**PRFLAGS**	**PRBUF**	
	ACCHR	SCHR	ACCOL	SCOL	ACA	ACX
		SKPCHR		**SKPCOL**		
	MON	MOFF	STRTU	STOPU	FMT	DELAY
	MAPON	**MAPOFF**	**STARTU**			
	ACSPC	BDSPC		RSETP		
	ACSPEC	**BLDSPEC**		**RESETP**		

Printer functions are described in the manual for the HP 41 Infrared printer module. STARTU and STOPU enable and disable underlining of printed output. The functions are mutually exclusive. MAPON and MAPOFF select mapping enabled or disabled and are mutually exclusive. The default setting is MAPON.

The following printer module functions are not provided by the emulator:

REGPLOT – A user-written version is given in Chapter 5.

STKPLOT – This is a version of REGPLOT, which takes all its arguments from the stack instead of using registers 00, 01 and 02. If you need this, you can write an HP 48 program to do it, by modifying the REGPLOT example in Chapter 5.

PRAXIS, PRPLOT, PRPLOTP – These are HP 41 programs that you can transfer from the infrared printer module by using PRP, as described in Chapter 3.

TESTP – The HP 48 has its own infrared printer tests built into it.

HP 48 functions are used to allow interaction of the HP 41 environment with the HP 48.

HP 48	X→48	48→X	A→48	48→A		EXIT
	XTO48	**48TOX**	**ATO48**	**48TOA**		
	48XQ					

The functions XTO48, 48TOX, ATO48, and 48TOA are used to exchange information between the HP 41 environment and the HP 48, as is 48XQ which treats the contents of the ALPHA register as an HP 48 command to be executed. EXIT is used to leave the HP 41 environment and re-enter the HP 48 environment. See Chapter 5 for more details.

Using the HP 41 Environment and Keyboard

The examples here are designed to show how similar the emulator is to the HP 41, and to point out where there are differences. If you have not used an HP 41 for some time, the examples will help you get up to speed again.

Enter the HP 41 Keyboard Mode

1. Make sure the Emulator card is plugged in, and turn on the HP 48.

   Select the library menu. You may have to press [ATTN] first to get out of a special environment, such as the graphics environment or the alarm catalog.

   [⬑] [LIBRARY]

2. Enter the HP 41 emulator. You may have to press [NXT] one or more times to bring the HP 41 menu key to the display.

   HP-41

3. Enter the HP 41 keyboard mode.

   HP41

4. Put the HP 41 Emulator keyboard overlay over the HP 48 keyboard.

Comparison with the HP 41:
As expected, this is completely different from the HP 41, since the HP 41 Emulator is a specific mode which has to be selected on the HP 48.

Set ENG 10 Display Mode

1. Set the calculator to display 10 digits in engineering mode:

   [↰] [ENG] [VAR]

Comparison with the HP 41:
The first two keys are exactly the same as those used to select ENG mode on the HP 41. The third key is different, the HP 41 has only ten keys on the top two rows and pressing the tenth key selects ENG 0 mode. On the HP 48 more digits can be displayed, and there are twelve keys in the top two rows, so [VAR] provides an argument of 10. As on the HP 41, the rightmost key in the second row used with ENG provides an argument of 0.

Calculate Three Times the Cosine of π Radians

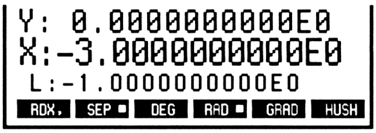
1. Select radians mode in the MODES menu.

   [↰] [MODES] RAD

(Or you could have spelled out the function name [XEQ] [α] R A D [α].)

2. Carry out the calculation 3 x cos(π).

   3 [↰] [π] [COS] [X]



```
Y: 0.0000000000E0
X:-3.0000000000E0
 L:-1.0000000000E0
```

Comparison with the HP 41:
Selection of Radians mode is somewhat different, since it is done through the MODES menu. The calculation is performed exactly as it is on an HP 41.

Execute TONE 9

1. Execute the TONE function

   $\boxed{\text{XEQ}}\boxed{\alpha}$T O N E$\boxed{\alpha}$

   Wait for the prompt `TONE_` to be displayed.

2. Enter 9.

   $\boxed{9}$

Comparison with the HP 41:
This is exactly the same as on the HP 41.

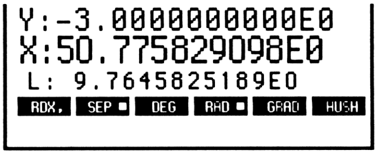
If you want to disable the beeper, and if the MODES menu is still selected, press `HUSH` . Press `HUSH■` again (it has a square block at the right when sounds are disabled), to enable sound again. This is different from the HP 41, but simpler. `HUSH` controls flag 26 (you can also clear flag 26 to disable sounds and set flag 26 to enable them).

Calculate 5.2(7.1+$\sqrt{7.1}$) and Store the Result in Register 12

1. Enter the calculation.

   $\boxed{5}\boxed{.}\boxed{2}\boxed{\text{SPC}}\boxed{7}\boxed{.}\boxed{1}\boxed{\text{ENTER}}\boxed{\sqrt{x}}$
   $\boxed{+}\boxed{\times}$

2. Store the result in register 12.
   $\boxed{\text{STO}}\boxed{1}\boxed{2}$



```
Y:-3.0000000000E0
X:50.775829098E0
 L: 9.7645825189E0
```

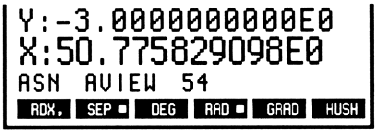
Comparison with the HP 41:
This is the same as on the HP 41 except that the $\boxed{\text{SPC}}$ key was used in place of the $\boxed{\text{ENTER}}$ key to separate two different numbers. Either one of these two keys can be used to separate two numbers when they are being entered, but only $\boxed{\text{ENTER}}$ duplicates the X register in the Y register, so $\boxed{\text{ENTER}}$ had to be used when two copies of the number 7.1 were needed. As on the HP 41, $\boxed{\text{ENTER}}$ **disables stack lift,** so that a

second number typed in or recalled immediately after [ENTER] has been pressed will replace the contents of register X. When [SPC] is used to end the entry of a number, that number is put only in register X and stack lift is enabled. Note that pressing the [NXT] key after [STO] would give the same result as typing the number 12.

Assign AVIEW to the DEL Key

1. Select the ASN function, enter AVIEW and assign it to the [DEL] key.

   [↰] [ASN] [α] A V I E W [α] [DEL]



```
Y:-3.0000000000E0
X:50.775829098E0
ASN  AVIEW  54
RDX, | SEP ■ | DEG | RAD ■ | GRAD | HUSH
```

The keycode for the DEL key is the fifth row, fourth key (54).

Comparison with the HP 41:
This is very similar to the HP 41 ASN function. Keycodes work in the same way as on the HP 41, but there are more keys and they are distributed on the keyboard in different ways. The DEL key does not have an unshifted function, so you can assign commands to it for use in USER mode.

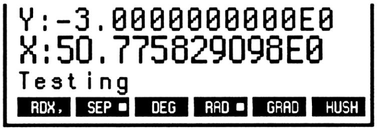
Enter and Display the Alpha String "Testing"

1. Enter the word Testing.

   [α] T [↰] E [↰] S [↰] T [↰] I [↰] N [↰] G [α]

2. Access USER mode and view the ALPHA register.

   [CST] [DEL] (The key to which you have just assigned **AVIEW.**)



```
Y:-3.0000000000E0
X:50.775829098E0
Testing
RDX, | SEP ■ | DEG | RAD ■ | GRAD | HUSH
```

3. Exit USER mode.

   [CST]

Comparison with the HP 41:
The main difference is that users could not display lower case characters (except a through e) on the HP 41. In stack display mode the emulator displays messages in place of the L register, instead of replacing X with them.

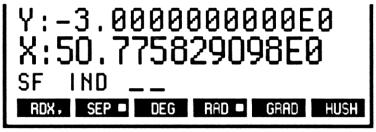
Set the Flag Numbered by Register Y

1. Select the FLAGS menu, and press the set flag menu key.

   [⬑] [FLAGS] SF

2. Select the indirect option.

   [⬑]



```
Y:-3.000000000E0
X:50.775829098E0
SF IND __
RDX. | SEP ■ | DEG | RAD ■ | GRAD | HUSH
```

3. Complete the operation by entering ST Y

   [.] [+/-]

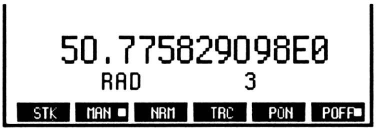
Assuming you still have the result -3 in register Y, you will see a small number 3 appear on the top line of the display, showing flag 3 has been set. You can clear it again by executing CF 03.

Comparison with the HP 41:
The SF command is selected from a menu instead of being on the keyboard, but the Indirect and Stack Register options work exactly as they do on the HP 41. Unlike the HP 48, but like the HP 41, there are no negative flag numbers, and minus signs are ignored by flag operation commands.

Change to a One-Line Display

1. To change to a one-line display, enter the MODES menu and press the stack key.

   [SHIFT] [MODES] [NXT] STK■



```
50.775829098E0
   RAD        3
STK | MAN ■ | NRM | TRC | PON | POFF■
```

Comparison with the HP 41:
There is no corresponding function on the HP 41. This display looks more like that of the HP 41 than does the normal display provided by the emulator. Press the STK menu key again to leave the one-line mode.

Print the Value in Register X

Place an HP82240B printer (or an HP82240A) in line with the infrared printer bulb. Turn the printer on.

1. Turn print mode on.

   [MODES] PON

2. Select the emulator's print menu and print the X register.

   [PRINT] PRX

Comparison with the HP 41:
The PRX command is the same as on the HP 41, but you can select it from the printer options menu. If you wish, you can spell out the command, assign it to a key, or put it in a program. The HP 48 infrared printer works the same as the infrared printer module on the HP 41.

Exit the Emulator

Return to the emulator's HP 48 menu.

[↰] [↱] EXIT

Comparison with the HP 41:
There is no corresponding operation on the HP 41.

Emulator Variables and Setup Options

The first time you use the Emulator Card, it sets up variables to hold HP 41 information. This section describes the variables, and how you can change them when you are not in the HP 41 environment. The variables are given the default names shown in the following table:

HP41PAR
Configuration variable, stores the names of the variables listed below

PRG41	HP41XROM	HP41KEYS	HP41REGS	HP41STACK
Current directory of programs	Current directory of XROM functions	Current key assignments	Current HP 41 data registers	Current HP 41 stack registers

Layout of HP 41 Emulator Variables

These variables are set up as if you were using an HP 41CV. You can alter some of these variables from within the HP 41 emulator, but other settings can only be changed when you are outside the emulator.

For example, if you are short of memory in your HP 48 and want to reduce the amount of memory used by the HP 41 emulator, you can do this using the SIZE function while in the HP 41 environment. On the other hand, if you want to set up a library of commonly used HP 41 programs by putting them in a separate directory, you can only do this while outside the HP 41 environment.

HP41PAR - The Configuration Variable

This variable contains a list of data that provides customization information for the HP 41 emulator. HP41PAR is the starting point whenever the HP 41 emulator needs to find information about the environment. (It is like PPAR which holds information needed when something is to be plotted by the HP 48.) HP41PAR cannot be called any other name. The HP 41 emulator looks for a variable called HP41PAR, and finds the names of other emulator variables there. HP41PAR is a reserved name. If you use this name for other variables in the HP 48, you will cause an error message when the HP 41 emulator tries to use HP41PAR. If HP41PAR is found on the current path at start-up of the emulator, it will be copied to the current directory. If HP41PAR is not found on the current path at start-up, it will be created in the current directory with the following default values:

Variable name:

```
HP41PAR
```

Variable contents:

```
{  PRG41  HP41XROM  HP41KEYS  HP41REGS  HP41STACK  }
```

HP41PAR contains the names of variables that must exist for the emulator to work. Their default names are given below, but you can give them different names so long as you put your chosen name for each one in the right place in HP41PAR.

Default name	Purpose of variable
PRG41	current directory containing a library of HP 41 programs
HP41XROM	current directory containing XROM functions
HP41KEYS	current key assignments
HP41REGS	current working data registers
HP41STACK	current working stack registers

Searching for Emulator Variables

The variables are called current variables because the emulator searches for them in the current path. PRG41 and XROM must be on the current path, but need not be in the current directory. If STACK, REGS, and KEYS variables are found on the current path, but not in the current directory, they are copied to the current directory.

Note You can create a new HP 41 environment simply by creating a new directory, moving to it, and pressing HP41 More details are given under *Configuration*, near the end of this section.

The following is an example of how HP41PAR is used during a search for an emulator variable.

If you have carried out the previous examples, you now have a variable called HP41STACK in the current directory. HP41STACK contains the contents of the stack as it was when you left the emulator. To save these stack contents, you must create a new stack variable with a different name. To do this, edit the list called HP41PAR and change it to:

```
{ PRG41 HP41XROM HP41KEYS HP41REGS HP41XYZ }
```

Store this edited version back into the HP41PAR variable.
The next time you enter the HP 41 emulator mode, the emulator looks for the variable HP41PAR. If it finds HP41PAR on the current path, the emulator then looks for variables called PRG41, HP41XROM, HP41KEYS, HP41REGS, and HP41XYZ. All exist (except HP41XYZ) since they were created the first time the emulator was used. A variable called HP41XYZ does *not* exist though, so the emulator creates it, in the current directory, showing the message:

```
   HP41XYZ  initialized
```

Then the emulator mode becomes active and you see the HP 41 display, including a stack full of zeros. The variable HP41STACK still exists, and has the original stack contents, but the emulator does not use it, instead it now uses the new stack variable HP41XYZ. As this has not been used before, the stack contains all zeros. You can use the emulator, with this new stack variable, as long as you want. When you decide to use the stack variable HP41STACK again, you edit HP41PAR and put the name HP41STACK in place of HP41XYZ.

The HP 41 Program Library Directory PRG41

PRG41 is an HP 48 directory that contains your standard HP 41 programs. Working programs that do not need to be changed can be put here. You can also put any commonly used HP 41 programs in this directory, so that the emulator can find these programs at any time. If you have a lot of HP 41 programs and want to avoid filling your current directory, programs can be put here so the current directory holds fewer variables.

The HP 41 XROM Directory Variable HP41XROM

HP41XROM is an HP 48 directory that contains the definitions of all user-written XROMs. It is an XROM library equivalent to CAT 2 on the HP 41. It is searched whenever a name is not found in CAT 1. (If you do not know what an XROM is, check your HP 41 manual for details of plug-in modules.) For more details, see Chapter 5.

The HP 41 Key Assignments Variable HP41KEYS

HP41KEYS contains the currently-active HP 41 key assignments. These are stored in the form (keycode,object), where keycode is a logical keycode comprising a signed row-column number (for example, row 1 column 2 would be 12), and object is a name which evaluates to an HP 41 global label, XROM, or built-in function. This object also stores information indicating which keys have something assigned to them. HP41KEYS is a library data object whose size depends on the number of key assignments made.

Changing the name of this variable can be very useful. If you have several programs, each of which uses a different set of key assignments, you can create a different key assignment variable for each program. When you run a program, you can either rename its key assignment variable to HP41KEYS, or you can change the third name in HP41PAR to correspond to the name of the required key assignment variable.

The HP 41 Data Registers Variable HP41REGS

HP41REGS stores the current working data registers. The registers are stored sequentially, and implicitly numbered from 0 through SIZE-1. Each register occupies 8 bytes. The size of the array and the position of the statistics registers are stored with the array itself. The overall size of this variable depends on the HP 41 SIZE setting. By default the SIZE is 100. The HP41REGS variable is a library data object, so you must use emulator functions (such as RCL41) to access its contents from HP 48 programs.

The HP 41 Stack Variable HP41STACK

HP41STACK contains the RPN 4-level stack XYZT, the Last X register L, and the ALPHA register. Each register is stored as 8 bytes, except ALPHA, whose size varies, up to a maximum of 24 bytes long when it is full. The variable also contains the HP 41 flags, the current address in the current HP 41 program, and the HP 41 subroutine return stack. The variable is a library data object, which means that it is used by a library, that is, the HP 41 Emulator library.

Library data objects have object type 26. You can copy them, move them on the stack, and store them, but you cannot try to edit them or otherwise alter them with built-in HP 48 operations. Only when you go inside the HP 41 environment can you see and use the information in the stack variable. To exchange information between the HP 48 and the HP 41 environment while you are outside the emulator, use instructions provided by the DAT41 menu of the HP 41 emulator (see Chapter 4). To exchange information between the HP 48 and the HP 41 environment while you are inside the environment, use the functions in the HP 48 menu (see Chapter 5).

Searching for Commands in PRG41 and HP41XROM

When a command XEQ "name" is entered from the keyboard, "name" is looked for in the catalogs. CAT 1 is searched from the end of the catalog backward to the beginning, as on the HP 41. If "name" is not found in the current directory, then the program directory is searched in the same way, since it is part of CAT 1. After this, the XROM (CAT2) directory, and then CAT 3 and CAT 7 are searched. If "name" is not found in any of these directories then the following message appears:

```
Nonexistent
```

This might mean that the name is in a directory that has not been searched. Verify that the current directory is correct. The current directory path is displayed in HP 41 mode. Also, check if you have misspelled the name.

This search mechanism means that you can write a CAT 2 function called "name", to replace a CAT 3 or CAT 7 function with the same name, and your function will be used instead of the one that is built in. Likewise, a test version of a program in the current directory will be found first (before PRG41 programs) and will be used. If you have a particular set of test versions, you can put them all in a separate directory, and make that the current directory, when you want to use the them.

Normally, both your program directory and your XROM directory are always available in the HP 48 HOME directory. You may, however, wish to "hide" one or both, by using the emulator in a directory which is not on the same path as these directories.

Configuration

When the HP 41 Emulator is started up, the environment is configured. The above variables are checked, and created if necessary. If HP41PAR exists but is of the wrong type or cannot be created, configuration fails. If the object or objects named as the program directory or the XROM directory are not in fact directories, configuration fails. If any of the data, key assignment, or stack objects are of the wrong type, configuration fails. When configuration fails, the following message is displayed:

```
Invalid  HP41PAR
```

If you change the names or contents of the variables, then these changes will not be recognized until the next time the environment is configured. For example, you could use the function 48XQ to rename the variable HP41KEYS without leaving the HP 41 environment. The environment would continue to use the key definitions that it found in the key variable at configuration, and would not notice that the variable had been renamed. Only when the environment is configured again will the name change have an effect.

The HP-48 operations described in Chapters 4 and 5 that access the HP 41 Environment will also need to perform a successful configuration before they can proceed. If a variable of the name supplied in the configuration does not exist, but is required by the operation being performed, then it will be created. For example, an attempt to store data in an HP 41 data register will create the data register variable named in HP41PAR if it does not exist, and will create it with 100 registers.

HP41PAR, HP41STACK, HP41REGS, and HP41KEYS must be in the current directory. If they exist anywhere on the current path, they will be duplicated in the current directory. If not, then a new set is created. This means that you can create a new environment, without losing an old one, by creating a new directory, moving into it, and entering the emulator mode. In this new directory, you have a new stack, new data registers, new key assignments, and you can write new programs. The two directories are searched for along the whole current path, so you can still use programs from the program directory and XROM commands.

Limitations

The way in which the environment is held in variables imposes some limitations on the HP 41 Emulator. These limitations, and some others, are described here.

If you use 48XQ to execute an HP 48 command or program, the command or program cannot execute any HP 41 operations. For example, you cannot run a program from the HP 41 emulator that executes an HP 48 program that tries to execute an HP 41 program. Such an operation would mean you would be re-entering the HP 41 environment from outside. If you try to do this, you will get the following error message:

```
        Not  Re-entrant
```

There is no continuous ON mode. The HP 48 always turns off after 10 minutes of inaction, and the HP 41 Emulator does not attempt to override this. If you want to keep the HP 48 permanently on, you can set an appointment alarm to execute every 9 minutes. This will interrupt any HP 48 or HP 41 activity, but the activity will continue after the alarm has executed.

The HP 41 environment allows a maximum of 6 pending subroutine returns. This is exactly the same as on the HP 41, but you should be aware of the limitation if you try to write any new HP 41 programs for use with the Emulator. Execution of XROM functions and 48XQ does not use the subroutine return stack.

The Emulator allows execution of local alpha labels **A** through **L** and **a** through **f** by pressing keys on the top two rows in USER mode. However, for compatibility with existing HP 41 programs, [GTO] or [XEQ] followed by [α] L [α] or [α] K [α] will search for the corresponding global labels. If no such label is found, the following message is displayed:

```
        Nonexistent
```

When you leave the HP 41 environment and re-enter it, and when you turn the calculator off and on, the flags which are preserved or changed are not exactly the same as those on the HP 41. See the flag list in Appendix C for details.

Each data register is 8 bytes long, and can hold 7 alpha characters, but normal alpha operations store 6 characters in data registers, for compatibility with the HP 41.

3

Capturing HP 41 Programs and Data

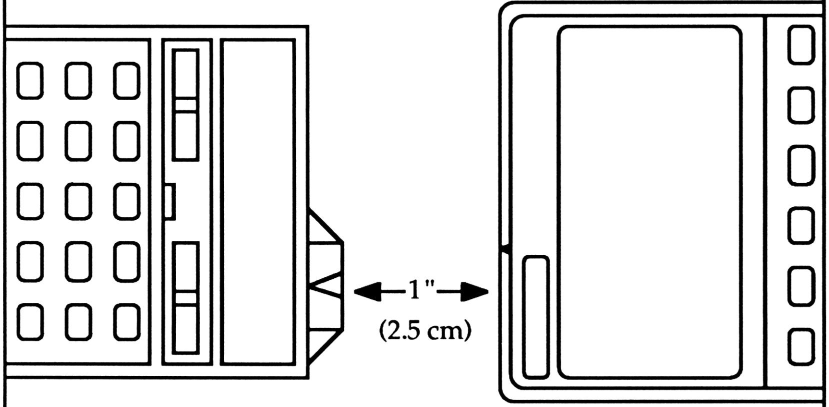
This Chapter describes how the Emulator captures programs, data registers, and status information. It also explains how programs can be run, using captured data.

This chapter covers the following topics:

- General principles of data and program transfer and capture.
- Running captured programs.
- Capturing status and other data.
- Verifying correct capture.
- Compatibility with other calculators.
- Suggestions for handling problems.

Positioning the HP 41 and HP 48

Each time you transfer data or programs from the HP 41 to the HP 48 you have to align them so that the HP 48 can read information sent by the HP 41 Infrared printer module. Position the HP 41 and the HP 48 on a flat surface, so that the HP 41 Infrared printer module and the HP 48 are opposite each other. The base of the triangle at the top of the module should be aligned between the second E and the first T in HEWLETT on the label at the top of the HP 48.



Program transfer from an HP 41 to an HP 48.

The HP 41 Infrared printer module should be in one of the two upper ports of the HP 41. The above figure shows the Infrared printer module in port 1 of the HP 41. If your printer module is in port 2, the module must line up as shown, even though the body of the HP 41 is not in the position shown. Do not try to align the cone on the Infrared printer module with the triangle above the HP 48 display.

The HP 41 printer module and the HP 48 should have a gap of about one inch (2.5 cm).

General Principles of Data and Program Transfer

The CPTR key in the HP-41 Library menu lets the Emulator capture HP 41 programs, data, and status information. The Emulator recognizes what it is receiving, so all three kinds of information are captured in the same way. Below is a simple example of sending one datum from the HP 41 to the HP 48. The general instructions in the example apply to transferring programs, data, and status information. Instructions specific to transferring the datum are boxed, so you can ignore them when you refer to this example for your own work.

1. On the HP 48, select the HP-41 menu from the LIBRARY menu, so you can see the menu key marked CPTR. Do not press this key yet.

2. On the HP 41, plug in an Infrared printer module, if one is not already plugged in.

3. Make sure the data or program to be transferred is in the HP 41.

Key in 41248, and store it in register 0.

4. Be sure that the HP 41 print modes are set correctly. When you plug in the printer module and turn on the HP 41 the modes are automatically set as required. You can also execute RESETP, or otherwise ensure that the print modes are set as follows:

   Single-width characters – clear flag 12.

   Upper-case characters – clear flag 13.

   MAN mode – execute MAN or clear flags 15 and 16.

   Underlining disabled – execute STOPU and ADV.

   Printer activated – execute PRTON to set flag 55.

   If you plan to print from a program, enable printing – set flag 21.

   For fast transfer, set the Infrared printer module's delay time to 0.5 on the HP 41. (Put 0.5 in the HP 41 stack register X, then execute the function DELAY.) There is a small chance that an

HP 48 with a nearly full user memory will fail to capture correctly at this speed. The section at the end of this chapter, *Notes on Possible Problems,* explains this fully.

6. Now you can begin the transfer by using one of the functions PRP, PRREG, PRREGX, or PRFLAGS, depending on the information you are transferring.

> Use the HP 41 printer function PRREGX to copy the number 41248 from a data register in the HP 41 to the same data register in the HP 48. Put 0 in register X, then type [XEQ] [ALPHA] P R R E G X, but do not press [ALPHA] again.

7. Line up the HP 41 and the HP 48 as described earlier.

8. Press the HP 48 key CPTR . When the HP 48 screen goes blank, the HP 48 is ready and waits for 10 seconds to receive some information.

9. Press the HP 41 key that begins the printing.

> Press the [ALPHA] key, so that the PRREGX instruction is completed.

10. Wait for the HP 41 to finish printing and for the HP 48 to finish capturing the information, and translating it for use by the Emulator.

11. If the capture was successful, the HP 48 displays messages to show what information it has received, and how the information is being translated. After capturing the information, you will see a message in level 1, containing details of what has been received. If the capture does not work, level 1 will contain a list. If the capture has only partially worked, the level 1 list gives extra detail about the captured information that was not translated. If the list contains bad data, it is most likely that there was trouble transmitting or receiving the information. Line up the HP 41 and HP 48 and try the transfer again. (If it fails again, see the list of possible problems at the end of this chapter.)

> You should see the message `{ "Regs" 0 1 }` telling you that the HP 48 has captured register data, beginning at register 0 and that only 1 register has been captured.

12. To confirm that the capture worked, go into HP 41 mode (press HP41) and check that the information has been received.

Press VIEW 00. You should see the number 41248 in the message area.

Note that, when it receives programs or data, the HP 48 capture function makes some changes to handle the differences between the HP 41 and HP-48 character sets:

- Double quotes are changed to single quotes.
- Characters 127 and 148 (the HP 41 append character as sent in MAPOFF and MAPON modes respectively) are treated as the append character.
- The '→' symbol (character 141) is treated as append too, allowing programs written on the HP 48 to use this character which is on the HP 48 keyboard.

Note

When you transfer information from the HP 41 to the HP 48, the HP 48 waits for ten seconds after you have pressed the CPTR key. If nothing has been sent to the HP 48 in that time, the HP 48 stops waiting and displays an error message.

The procedure used in the above example ensures that less than ten seconds pass before transmission begins.

Problems?
If you have everything arranged as described above, then there should be no problem in communication between the HP 41 and the HP 48. If at any time you have problems carrying out the capture operations described in this chapter, check the suggestions at the end of this chapter.

Capturing a Program

To capture a program, carry out the same 12 steps as you did for capturing data, but use PRP (instead of PRREGX), just as if you were printing a program. (You can also capture parts of programs by using the HP 41 LIST function.)

Any program in HP 41 Main memory, or XROM program (in a plug-in module that is plugged in) can be captured using PRP. Programs stored in HP 41 Extended Memory must be copied to User memory before they are transferred.

XROM functions cannot be captured. If you want to use an XROM function, check in Chapter 2 to see if the function is built into the Emulator. If it isn't, Chapter 5 shows how you can create and add functions.

Capturing a large program can take a long time. The HP 41 displays PRP "name" until the printing is finished. If the HP 41 has time functions available, then it displays the time and date and gives no indication of when it finishes printing the program.

After the transfer begins, the HP 48 accepts infrared signals until no signals arrive for a period of 3 seconds.

After the program has been transferred, the HP 48 translates the program and displays the message `Line` *n* . The number *n* changes as each line is translated. Once the translation is complete, the Emulator displays the message:

```
        Compiling GTO/XEQs
```

Below this message, the Emulator displays the value of the program counter it has reached while searching for GTO and XEQ commands. If a GTO or XEQ command calls a local label, the Emulator tries to find that label in the program and stores its location with the GTO or XEQ to speed up program execution. Remember that, besides the two-digit numeric local labels, the Emulator allows local alpha labels with unquoted single letters A through L and lowercase a through f. As usual, only local label GTO and XEQ commands are compiled.

XROM commands are searched for in the HP41XROM directory. If the XROM command is found, it is compiled into the program using its XROM and function number. If an XROM such as XROM 'name' is not found in the .i.HP41XROM ;directory, it is converted to XEQ 'name' and is searched for like other global labels at run time. (The name is searched for in CAT 1, CAT 2, CAT 3 and CAT 7, in the order described in *Searching for commands in PRG41 and HP41XROM* in Chapter 2.)

Program steps that are not in the Emulator's CAT 3 are assumed to be names of XROM functions. These names are searched for in the Emulator's CAT 2 and CAT 7 during translation. If a name is not found, the following warning is displayed:

```
Unknown:name
```

In the above message, `name` is the missing function. The unknown name is replaced by XEQ 'name' in the translated program. If the name is longer than seven letters, it is truncated to its first seven letters. If the name (or truncated name) is still not found during program execution, the program stops and displays the following message:

```
Nonexistent
```

Again, if an error is detected during receipt of infrared data, retry the operation. If it still fails, look at the list of problems at the end of this chapter.

If the transfer works correctly and the captured program is successfully translated, then the program is stored as a variable in the current directory and the name of this variable is put on level 1. The variable name is the first global label found in the program.

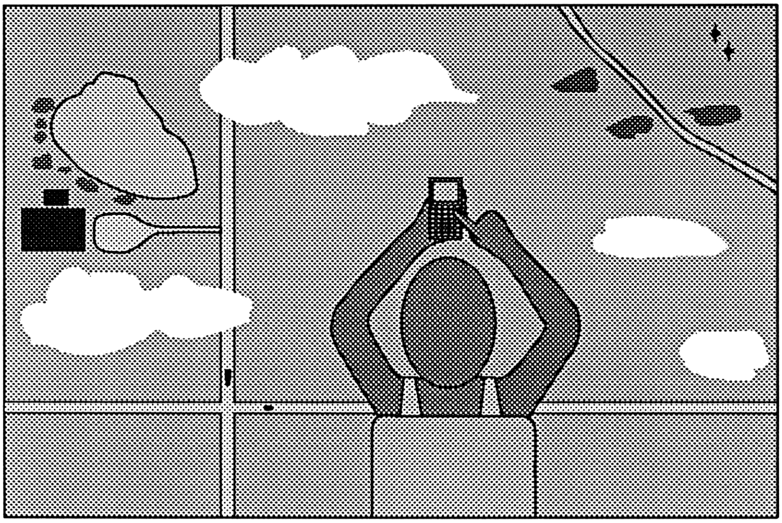
Caution If another program object with the same name already exists in the current directory then its contents will be replaced by the newly translated program.

If no global labels exist in the program then the default name FOCAL[1] is used.

[1]Focal is a name used for the programming language of the HP-41. (It stands for Forty-one calculator language.) An unnamed program written in this language is therefore called a Focal program by the Emulator.

Here is a programming example for you to capture.

Ian Jumpoff enjoys skydiving in Corvallis, Oregon. In his calculator, he keeps a record of the last 10 years' rainfall to see how much the weather interferes with his jumping each year. Since 1979 he has kept a record in his trusty HP 41C, now he wants to move the program and data over to his new HP 48.



The rainfall figures are stored in ten consecutive data registers in the HP 41. The program displays a message, then asks which register contains the first rainfall figure. The user must type in the register number, and press [R/S]. The program calculates the average rainfall for the selected 10 years, then displays the result, and the last rainfall figure of the period for comparison.

Here is the HP 41 program Ian has been using to display the average rainfall for ten years.

Program steps	Explanation
01 LBL "RAIN"	Global label.
02 "RAIN AVERAGE"	A message to say what the program is.
03 AVIEW	Show the message to the user,
04 PSE	and pause so the user can read it.
05 FIX 1	Set FIX 1 display mode.
06 ΣREG 11	Put statistics registers in the right place,
07 CLΣ	and clear them.
08 "FIRST REG?"	Ask which register has the first rainfall
09 PROMPT	number, user must type it in and press R/S.
10 ABS	Make sure the register number is positive,
11 INT	and an integer.
12 1.001	Put register number in fractional part of
13 *	this number and add .009 to it so as to
14 .009	make an ISG counter which counts through
15 +	ten registers, beginning with the first.
16 LBL 01	Label to begin loop over 10 rainfall counts.
17 RCL IND X	Recall rainfall from next register.
18 Σ+	Add it to the statistics data.
19 RDN	Get ISG counter back to stack register X.
20 ISG X	Increment the counter,
21 GTO 01	if not finished then go to repeat the loop.
22 LASTX	If finished, get the last rainfall from L.
23 MEAN	Calculate the average rainfall.
24 "AV:"	Set up a message showing the average,
25 ARCL X	put the average in the ALPHA register,
26 "⊢ LST:"	append a title for last year's rainfall, and
27 ARCL L	append last rainfall, getting it from L.
28 AVIEW	Display the message.
29 FIX 4	Set the default display mode.
30 END	Finish the program.

Key this program into your HP 41 and check that you have made no mistakes. To test the program on your HP 41 before transferring it, use the set of data in the next section (*Capturing Data*). Then transfer the program:

1. On the HP 48, enter the HP 41 emulator.

   [⬅] LIBRARY HP-41

2. On the HP 41, prepare to transfer the program.

   [XEQ] [ALPHA] P R P [ALPHA] [ALPHA] R A I N

3. Initiate transfer of the program to the HP 48.

   CPTR

4. Transfer the program from the HP 41.

   [ALPHA]

When the HP 48 successfully captures the RAIN program, it will translate it. You will see the message `Line` *n* as each line is translated (*n* will go from 1 to 30). The translator will compile GTO and XEQ instructions, and store the program in a variable named RAIN. The name RAIN appears on level 1 to let you know where the program was stored. To run the program, you will need some data.

Capturing Data

The HP 41 printer has two functions that print data registers. PRREG prints *all* data registers, and PRREGX prints a block whose first and last registers are specified by a number in stack register X.

As you saw in the first example in this chapter, PRREGX expects a number *bbb.eee* to be in X on the HP 41. *bbb* is the number of the first register to be printed, and *eee* is the last register to be printed. For example, to print registers 10 through 20, use the numbers 010.020 or 10.02. The SIZE setting of the emulated HP 41 must be at least as large as *eee* .

When the data registers have been captured, translated, and stored, the Emulator displays the following in level 1 of the HP 48 stack :

```
{ "Regs" mm nn }
```

This tells you that CPTR has captured data registers, beginning with register *mm*, and that a total of *nn* registers were captured.

If CPTR has found data it could not translate, it displays the following:

```
{ "Regs" mm nn "Remainder" "xxxxxxxx" }
```

Here, *mm* is the first register, *nn* is the number of registers successfully translated and stored, "Remainder" tells you that the remaining information could not be handled, and "*xxxxxxxx*" is the remainder of the data received, beginning at the point where translation failed.

As an example, transfer the data used by Ian Jumpoff for his program in the previous section. The following table shows the rainfall data collected by Ian.

Year	Rainfall in inches	Data register
1979	42.06	21
1980	42.13	22
1981	47.08	23
1982	46.75	24
1983	55.53	25
1984	48.57	26
1985	39.79	27
1986	43.75	28
1987	37.71	29
1988	37.52	30

Annual Rainfall data for Corvallis, Oregon from 1979 to 1988

Key in the rainfall data and store them in the HP 41 registers specified.

Run the rainfall program on your HP 41 to confirm that you have entered the program and data correctly. When the program displays the question `FIRST REG?`, key in 21 (the register where the first rainfall number is) and press [R/S]. The program should display the message:

```
AV:44.1  LST:37.5
```

This tells you that the average rainfall for the past 10 years was 44.1 inches, and that in the last year for which Ian has data it was 37.5 inches. The numbers are displayed to only one decimal place so the whole message fits in the display, but the program resets the display of the HP 41 to FIX 4.

Now, copy the data to your HP 48:

1. Prepare to transfer the data from the HP 41.

   21.030 [XEQ] [ALPHA] P R R E G X

2. Initiate transfer on the HP 48.

   CPTR

3. Transfer the data.

   [ALPHA]

Assuming the capture was successful, you will see a list in level 1:

```
{ "Regs" 21 10 }
```

This tells you that the first register captured was register 21, and 10 registers were captured.

If you want to capture all the data from your HP 41 user memory, use PRREG. Use the SIZE command to ensure that your emulated HP 41 has a SIZE at least as large as that in your HP 41. Then set up the HP 41 and HP 48 for transfer, press the CPTR menu key, and execute PRREG on your HP 41. The HP 48 will capture the data from each data register, translate it, and store it in the same register in the emulated HP 41, displaying each register number in turn. The following error occurs if the emulated HP 41 has a smaller SIZE than your HP 41:

```
CPTR  Error:Nonexistent
```

All the registers that do exist will contain captured data.

Running Captured Programs

Now you can run the translated example program. Press the HP41 menu key to enter the environment. Position the HP 41 keyboard overlay over the HP 48 keyboard, so that you will be able to use the keys provided by the Emulator.

Find and run the RAIN program.

[←] [CATALOG] 1 RAIN

When you run the program it displays the message "RAIN AVERAGE", then asks you for the number of the first register containing Ian's data. Type 21 and press the [ON] key, which acts as [R/S] on the emulator. You should see the same message as on the HP 41:

```
AV:44.1 LST:37.5
```

Now that Ian has the program in his HP 48, he can use it when he gets new data. He determines that the rainfall in Corvallis for 1989 was 29.64 inches.
To run the program with the new data, first store the data in the HP 48.

29.64 [STO] 31

Now you have data for 11 years, with the last ten years beginning at register 22. Run the program again. When the program asks for the first register, type 22 and press [R/S]. The result should be:

```
AV:42.8 LST:29.6
```

It looks like 1989 was an unusually dry year and Ian should have been able to do a lot of jumping.

The Flying Goose

As on the HP 41, the Emulator moves a program execution annunciator one step across the message area every time a LBL is encountered during a running program. The symbol used is similar to the HP 41 "flying goose" symbol.

Starting and Restarting Program Execution

As on the HP 41, you can start a program at any global label by entering XEQ and typing the label name. If the label is a global label, you can also begin execution at that label by executing CAT 1, finding the menu key with that global label, and pressing the menu key. If the label is a local alpha label, make the program the current program, set USER mode, and press the appropriate key in the top two rows. An easy way to make a program the current program is to press [PRGM] and select the program's menu key.

You can start execution at any program step in any program by going to that program, then going to the step with GTO. or BST, and pressing [R/S]. You can also restart a halted program from its current position by pressing [R/S].

SST and BST

SST and **BST** let you step through programs as on the HP 41. When you press [SST] at a program step, that step is executed. If you hold [SST] down a short time and release it then the step is displayed and executed. If you hold [SST] down longer the step is displayed, and then cancelled and replaced by Null. If you press [BST] you step back one step in a program, but the step is not executed. If you keep [BST] pressed down, the step is displayed.

Since there is no program entry mode, you cannot use [SST] to move through a program without executing it. Use [BST] or GTO. to do this instead. You can press [EEX] after GTO. to extend the prompt to four digits.

If you do not want to press [SHIFT] every time to execute SST or BST then you can assign them to unshifted keys and use them in USER mode.

Alarms

If an alarm goes off, a running HP 41 program continues, but the HP 48 alert annunciator is displayed. The alarm carries out its action once the program stops. If you see the alert annunciator and suspect it is an alarm, then you can press [R/S] to stop the program. Once the program has stopped, a control alarm takes you out of the HP 41 Environment, and an appointment alarm executes its action (sound and/or message). See the description of alarms in Chapter 2 for more details.

The alert annunciator is the same as the low battery warning. You cannot see whether it has come on because the battery is low, or because an alarm has come due. Do not ignore an alert.

Program execution errors

Most program execution errors are the same as on the HP 41, such as incorrect use of data, numerical errors, or nonexistent registers, flags, or labels. You can identify the step giving the error by pressing SST and holding it down. (Since there is no program entry mode, you cannot enter PRGM mode to check the step.)

Capturing the Status

In the example used above, the program set the display mode to FIX 4 and positioned the first statistics register at data register 11. The HP 41 printer function PRFLAGS can be used to send this and other information directly to the Emulator.

PRFLAGS sends the following:

- The number of data storage registers (SIZE).
- The location of the statistics registers (ΣREG).
- The angular mode (DEG, RAD or GRAD).
- The display format (FIX *n*, SCI *n* or ENG *n*).
- The status of all flags (Flag *nn* clear or set).

CPTR captures all this data and sets the HP 41 environment to the same status. Only flags 00 through 43 are changed to correspond to the state captured from PRFLAGS.

To capture this information, position the HP 41 and the HP 48 for data transfer. Press CPTR and execute PRFLAGS on the HP 41. The process takes some time, since each flag is sent separately.

When the status and flags have been captured, translated, and stored, the Emulator leaves a list in level 1 of the HP 48 stack:

```
        { "Status" "Flags" mm nn }
```

This tells you that CPTR has captured status data, and flags beginning with flag *mm*, and that a total of *nn* flags were captured. Normally *mm* will be 00 and *nn* will be 44.

If CPTR found data that it could not translate it leaves one of the following lists in level 1:

```
        {"Status" "Remainder" "xxxxxxxx" }

{"Status" "Flags" mm nn "Remainder" "xxxxxxxx"}
```

The first message means that some or all status information was received correctly, but that the text beginning with the first character in the string after "*xxxxxxxx*" could not be translated. The second message means *nn* flags were successfully translated (starting with *mm*) , but "*xxxxxxxx*" is the remainder of the flag data received, beginning at the point where translation failed. In either case, all the information up to the beginning of the string "*xxxxxxxx*" has been translated and copied to the HP 41 environment.

As an example, set your HP 41 to ENG 9 display mode, to GRAD mode, clear flags 1 and 3, and set flags 0, 2 and 4. Then use CPTR on the HP 48 and PRFLAGS on the HP 41 to transfer the status data. You should see

```
        { "Status" "Flags" 0 44 }
```

in level 1. When you enter the HP 41 environment the status will have changed to the new status, with X, Y, Z, T and L displayed in ENG 9 mode, and the GRAD, 0, 2 and 4 annunciators turned on.

The HP 41 pauses after printing status information before it begins to print the flag values. If DELAY is set at the HP 41 default (1.9 seconds), the HP 48 may time out before it receives the flag values. To avoid this, set DELAY to 1 second.

Capturing Other Information, and Verifying Correct Capture

The emulator provides ways to capture other information, and verify captured data.

Capturing Other HP 41 Information

CPTR does not capture output from the printer commands PRX, PRSTK, PRA, PRΣ and PRKEYS. For numbers and strings in X or ALPHA, simply type the number or string directly on the HP 48. In the case of PRSTK or PRA when the ALPHA register has a long or complicated string you can transfer the contents to four data registers on the HP 41, then use PRREGX to send those four registers to the HP 48, and then extract the information from those registers on the HP 48.

In place of PRΣ you can use PRREGX to send the contents of the statistics registers to the HP 48. If you do not know where the statistics registers are, do one of the following:

- If you have not used ΣREG to change the default position of the statistics registers then they are registers 11 through 16.
- If you have an HP 41CX you can use the function ΣREG? to find the first statistics register.
- If you have used ΣREG from within a program, look through that program to find where it has moved the statistics registers.
- You can use the output of PRFLAGS to tell you the location of the first statistics register.

Using INPRT

The Emulator provides an additional capture command, called INPRT, to let you capture any data from the HP 41, including output from the commands PRX, PRSTK, PRA, PRΣ and PRKEYS. INPRT is described in Chapter 4. It accepts any printer information and puts it on the HP 48 stack as a text string, from which you can then extract the information you want.

Verifying Correct Capture

All the automatic CPTR operations check for errors in receiving data, though there is a very small chance that some errors may not be detected by the CPTR operation. INPRT also checks for errors. An alternative way to check if information is being received correctly is to put an HP82240A or HP82240B printer next to your HP 48, and position your HP 41 so that both the printer and the HP 48 are receiving the signals from the HP 41 Infrared printer module. Then you can get a printout of all the data being sent to the HP 48 *as it is being sent*. If you suspect any mistakes are occurring you can compare the printout with the information captured by the HP 48.

Suggestions for Handling Problems

The list below covers problems that can affect transmission, and suggests what you can do about them. It is followed by a list of other possible problems.

- Are the HP 41 and the HP 48 lined up with each other correctly. (See the diagram at the start of this chapter.)
- Are the HP 41 Infrared Printer module and the Emulator card plugged in correctly?
- Have the correct print modes been set on the HP 41? Flags 12, 13, 15 and 16 must all be clear so that programs will print in single width, upper case, in MAN mode. Underlining must be disabled, the printer must be activated by the PRTON command, and the printer module's DELAY setting should be no less than 0.5 seconds. When capturing PRFLAGS data, DELAY should be set to less than 1.4 seconds. If an HP 41 program is run to carry out the transfer then flag 21 must be set.
- Check the batteries.
- Make sure there is a clear path for the infrared light going from the HP 41 to the HP 48.
- Does the HP 48 have sufficient free user memory to hold the information being transferred and to carry out the required translation? If this is the problem you will see a warning

message. If the SIZE setting of the HP 41 Emulator is too small for the number of data registers being transferred, make SIZE larger.

- The HP 48 stores temporary data in user memory, and when it runs short of memory it stops other processes and clears out unnecessary temporary data. The clearing process takes time. Delete anything the HP 48 has received, and any other unnecessary information to make more space. Execute MEM to clear out unnecessary temporary data.
- The buffer should be cleared before you begin transferring programs or data to the HP 48. Execute ADV to clear the buffer. In most cases simply repeat the transfer, since the transfer that failed should have cleared the buffer.
- Is a second printer connected to the HP 41? Disconnect an HP 82143A printer if one is attached, and disable the print function on an HP-IL module if one is attached.
- Strong light, including sunlight, may have some infrared in it, so avoid carrying out the transfer near strong lights, or near infrared heaters.
- Finally, check if all the equipment is working. If your HP 41 or the receiver or the transmitter are not working, see the instructions on sending the device for repair in the User manual. If nothing seems to be damaged, turn the HP 48 off and on again. A ROM card test is carried out automatically every time the HP 48 is turned on. If a ROM card problem is detected then a message is displayed. If the Emulator card fails the test, send it for repair. If the card passes the test, carry out the HP 48 self-test described in Appendix A of the HP 48 Manual.

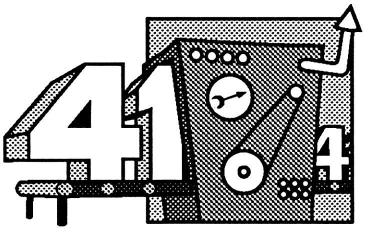
Problems can occur when the information is being translated, stored, or used. Most error messages explain the problem.

- Programs can contain commands that the Emulator does not recognize. Check what the unknown command is. If it is a reference to a program, make sure the program is in the HP 48, and is in the current directory or in the program library directory. If it is an HP 41CX or a command from a HP 41 plug-in module, see Chapter 5 for advice on writing replacement commands.

- Programs can refer to data registers that do not exist because the SIZE setting is too small. Captured data might also require data registers that do not exist. In either case, make sure the SIZE setting is sufficiently large.
- HP 41 programs that try to use a nonexistent flag to generate an error intentionally might not work, since the Emulator provides additional flags. For example, a program that has the step FS? 56 to cause an error will work on the Emulator, which has flags numbered up to 87.
- Some advanced HP 41 programming methods will not work on the Emulator, or might have unexpected consequences. See Appendix D for details.

4

Modifying HP 41 Programs



This chapter gives details of the commands that let you control the capture process. The commands let you check the steps of the capture and translation process, edit captured data and programs, and write complete HP 41 programs on the HP 48.

This chapter covers the following topics:

- The DAT41 and CTL41 menus.
- Exchanging data between the HP 41 environment and the HP 48.
- Translating HP 41 programs and data.
- Capturing programs under user control.
- Modifying and writing programs.
- Setting up a library of programs.

Commands in the DAT41 and CTL41 menus of the emulator interact with the HP 41 environment from the outside, and can be used from the keyboard or in HP 48 programs.

The DAT41 and CTL41 Menus

The DAT41 and CTL41 menus are accessed from the LIBRARY menu. DAT41 has twelve commands that let you send data to the HP 41 Environment and obtain information from it. CTL41 has eight commands that let you control the Environment.

DAT41 has two pages of menus. Both pages are shown below. If the full name of the command does not fit in the menu, it is shown below the menu key:

STO41	RCL41	PSH41	SF41	CF41	FS?41

A41→	→A41	MSG4	ERR41	SKIP4	CLRG
		MSG41		SKIP41	

CTL41 has the two pages of menus:

XEQ4	FCN4	→41	→TXT	INPRT	CAT41
XEQ41	FCN41				

PRIVA	VERS4				
PRIVATE	VERS41				

Exchanging Data Between the Environment and the HP 48

The commands described in this section let the user exchange information between the HP 41 emulator environment and the HP 48. This allows HP 41 and HP 48 programs to interact.

STO41 stores a real (or string) object from level 2 of the HP 48 stack into a specified HP 41 register number, or one of the following register letters: X, Y, Z, T, or L. The register number or letter must be given in level 1. The objects in levels 1 and 2 are dropped. Objects put on the HP 41 stack do not lift the stack. If the object in level 2 is a text string with more than seven characters, only the first seven are put in the HP 41 registers.

RCL41 does the opposite of STO41. It recalls a real number or a text string from a specified HP 41 register number, or one of the following register letters: X, Y, Z, T, or L. The register number or letter must be in level 1 of the HP 48 stack. RCL41 drops the register identifier and replaces it with the register contents.

PSH41 takes a real (or string) object from level 1 of the HP 48 stack, drops it from the HP 48 stack, and pushes it into register X of the HP 41 stack. If stack lift is enabled (HP 41 flag 56 is set), the HP 41 stack is lifted, otherwise the object replaces the contents of register X. If the object in level 1 is a text string with more than seven characters, only the first seven are put in register X. See the description of →A41 for other ways to copy text strings, or parts of text strings, to the HP 41 Environment. PSH41 is particularly useful if you want to put several objects onto the HP 41 stack from the HP 48.

A41→ recalls the contents of the ALPHA register as a text string which is pushed into level 1 of the HP 48 stack. The contents of ALPHA are unchanged.

→A41 is the opposite of A41→. It stores a string from level 1 into the ALPHA register. The string replaces the contents of ALPHA. If you want to append the string to the contents of ALPHA, use the following procedure:

A41→ [←] [SWAP] [+] →A41

This brings the contents of ALPHA to the HP 48 stack, appends the extra string which was in level 1 to the previous contents, and puts the new string in the ALPHA register.

If the string is more than 24 characters long, the first 24 characters are put in ALPHA. If you want to store the last 24 characters of a string in level 1, use the following:

[PRG] STK [NXT] DUP [PRG] OBJ [NXT] [NXT]

SIZE [PRG] STK [NXT] DUP 23 [-] [←] [SWAP]

[PRG] OBJ [NXT] [NXT] SUB →A41

This takes the last 24 characters of a string in level 1, or all of the string if it is less than 24 characters long, and stores it in ALPHA.

CLRG clears all HP 41 data registers. This command is equivalent to storing 0 in all the numbered data registers, so it is included in the DAT41 menu. Be careful when using this command. CLRG can do a lot of damage, destroying all the data in all your HP 41 data registers. The stack and the ALPHA register are not affected.

Errors

If you try to store something other than a real number or a text string in the HP 41 Emulator, the following error is displayed:

```
        Data  Error
```

If you try to store to a nonexistent register, or recall from a nonexistent register, the following error is displayed:

```
        Nonexistent
```

The above message is also displayed in the following cases:

A register has been given in the form of an object which does not specify a data or stack register.

A register number has been given that is greater than allowed by the current SIZE setting.

A string has been given that is not a stack register name.

An object has been given that is not a real number.

If an error occurs and LAST ARG is enabled, the command argument or arguments are left on the HP 48 stack.

Negative and fractional register numbers have their signs and fractional parts ignored. Register numbers must be real numbers.

Flag Commands

The DAT41 menu provides three commands that allow you to set, clear, and test HP 41 flags from the HP 48. In all cases a real number identifying the flag to be used must be given in level 1 of the HP 48 stack, and the command drops the flag number from level 1, unless an error has occurred. A negative sign, or any fractional part, is ignored.

SF41 sets the specified HP 41 flag. All HP 41 flags from 0 through 29, and the flags 85, 86, and 87 (used by the Emulator) are allowed.

CF41 clears the specified HP 41 flag. All HP 41 flags from 0 through 29, and the flags 85, 86, and 87 are allowed.

FS?41 tests the specified HP 41 flag. All HP 41 flags, including the additional Emulator flags 56 through 87, are allowed. 1 indicates the specified flag is set, 0 indicates the flag is not set.

Executing HP 41 Commands and Functions

The menu labels XEQ4 and FCN4 correspond to the two commands XEQ41 and FCN41, which can be used to execute HP 41 programs, commands, and functions from outside the HP 41 Environment.

XEQ4 (full name is XEQ41) takes a string argument from level 1 of the HP 48 stack and attempts to execute this as an alpha label. For example:

```
        "ABC" XEQ4
```

from the HP 48 keyboard or from an HP 48 program does the same as

```
        XEQ "ABC"
```

inside an HP 41 program or from the HP 41 keyboard.

ABC will be found if it is a global label or an XROM, otherwise the following error will be generated:

```
Nonexistent
```

If level 1 contains a null string (the string "") then XEQ41 makes the current HP 41 program continue from its current position. This is the same as pushing [R/S].

FCN4 (full name is FCN41) takes a string object from level 1 of the 48 stack and tries to interpret it as an HP 41 command. For example, to copy the contents of the X register to the ALPHA register, without entering the HP 41 Environment, you would use:

```
"ARCL X" FCN4
```

Commands should be written as they would appear in a program listing. If the command is not programmable, it should be entered as it would appear in the HP 41 display before being replaced by Null. FCN41 may be used to start a program running from a label, by placing a string such as "XEQ 'ABC'" or "XEQ 05" on the stack. The first can be done by XEQ41, but the second cannot.

FCN41 does not work with XROMs. For example,

```
"PRP" FCN4
```

produces the error:

```
Parse Failed
```

Use "XEQ 'PRP' "instead.

Using HP 41 Programs

CAT41 takes a translated HP 41 program from level 1 of the HP 48 stack as input, and returns a list of the global labels, and the END, contained in the program. For example, if you have forgotten the global labels in the object 'RAIN' (the example program in Chapter 3), you would do:

Action	Explanation
'RAIN' RCL	Recall the contents of the variable called 'RAIN' to the stack. Since it is a translated program, you will see it displayed as `Library Data`.
CAT41	Create a catalog list of all the global HP 41 labels in the object. You will see the list `{ "RAIN" "END" }`. This tells you that the program object 'RAIN' contains the HP 41 global label "RAIN" and an END.

The next three commands, MSG41, ERR41 and SKIP41 are designed to be used by HP 48 programs called from the HP 41 Environment. The commands help you write your own extensions to the HP 41 Environment.

`MSG41` takes a string from level 1 of the HP 48 stack, and displays it in the HP 41 message area. If printing is enabled and trace mode is set, then the message is printed as well.

MSG41 uses HP 48 error handling routines. When MSG41 is executed, it exits the HP 48 program that it is part of (including XROM programs). MSG41 should be used to give the user that a message, and to leave the HP 48 program at once. Any tidying up should be done before MSG41 is executed.

If an HP 41 program calls an HP 48 program and the HP 48 program executes MSG41, then the HP 41 program continues execution from its next program step after the message is displayed.

`ERR41` causes an error to be signalled to the HP 41 Emulator. If it is called from outside the HP 41 Emulator, it acts like the HP 48 command DOERR, displaying an error message and setting the error number to #70000h. Note that while DOERR will accept real numbers and binary integers, ERR41 only accepts a text string from level 1. If you want to make use of a built-in error number and message (one of the Emulator errors or one of the standard HP 48 errors), then call DOERR with the number.

If ERR41 is called by a program executed from within the HP 41 Emulator, then it takes a text string from level 1 of the HP 48 stack. The text string is displayed in the HP 41 message area. As with MSG41, the rest of the program which contains ERR41 is ignored. Control returns to the HP 41 program, which had stopped at the program step that executed the HP 48 program containing ERR41.

If HP 41 flag 25 is set, then ERR41 does not display the error message, flag 25 is cleared, and the HP 41 program continues.

Error messages are printed if printing is enabled and trace mode is set.

SKIP4 (full name is SKIP41) allows you to skip one line of an HP 41 program. It skips to the next line of the current program if it is executed as part of an HP 48 program called from the HP 41 Emulator, otherwise it does nothing at all. If the current HP 41 program is at its END then SKIP41 has no effect.

SKIP41 can be executed several times to let you skip over several steps of a program.

PRIVA (full name is PRIVATE) makes an HP 41 program object private. The program can still be executed and copied, but it cannot be viewed using SST and BST. It cannot be printed with PRP or LIST, nor turned into a text string with →TXT.

To use PRIVATE, put a program object in level 1 of the HP 48 stack and press the menu key. You can still copy the object, store it, and use catalog commands (including CAT41) to see the global labels in the program.

INPRT - Capture Without Translation

INPRT lets the HP 48 capture anything sent to the HP 48 infrared receiver as a text string, without translation. You can use INPRT to capture programs or data without automatic translation.

INPRT can also be used to capture other information from an HP 41, or to capture information from other sources. One example would be programs from HP 42S calculators.

When INPRT is executed, the HP 48 clears its screen and waits for 10 seconds for any input from the infrared receiver. Information should be sent to the HP 48 in the same way as is described in Chapter 3. If anything is received during the wait time, INPRT then continues to accept input until 3 seconds pass with no input. When the transfer is complete, INPRT returns the received data to the stack in the form of a string in level 2 and a number in level 1. If the number is 1, each byte that was detected was received correctly (although it is possible entire bytes may not have been received). If the number is 0, uncorrectable transfer errors were detected. Each byte that was not correctable is replaced by character number 127 in the level 2 string. If HP 41 information is sent with MAPON mode set on the HP 41 then any real character 127 (the HP 41 append character) is changed to character 148 before it is sent, so you will be able to recognize any character 127 as a bad character.

If strings, names, or expressions more than 24 characters long are transmitted, they may have had a linefeed added to them during the transfer. This will not happen in HP 41 programs, but you should check other transmissions.

If nothing is received by INPRT then it puts an empty string in level 2 and the number 1 in level 1.

Translating HP 41 Programs and Data

→41 takes a text string from level 1 of the HP 48 stack and attempts to translate it into an HP 41 program or HP 41 data. You can use INPRT to capture an HP 41 program as a text string, then make some changes by editing the string, then use →41 to translate the edited string into an HP 41 program to be used by the Emulator. Ways to use this are described later in the chapter. As with CPTR , →41 replaces quotes with apostrophes. →41 finishes by leaving the translated program as a Library Data object in level 2, with a suggested name in level 1.

If the string passed to →41 is a real HP 41 program captured by INPRT or read in from RS232, then it will be made up of several program steps, each beginning with a step number and ending with a linefeed. An example would be:

```
01 LBL "SHORT"
02 SIN
03 END
```

The string can also be a program typed in from the HP 48 keyboard. In this case, →41 accepts a program all on one line with no step numbers, or END, for example:

```
"LBL 'BRIEF' COS"
```

Note that the label BRIEF has been typed in with single quote marks around it. If the string was typed in as:

```
"LBL "BRIEF" COS END"
```

then the HP 48 would interpret it as the text string "LBL " followed by the unquoted name BRIEF and a second text string " COS END". An alternative way to enter this program is to type it in as a counted string by putting the following in the command line:

```
C$ $ LBL "BRIEF" COS
```

and then pressing ENTER. Counted strings are described in Chapter 4 of the HP 48 Owner's Manual.

→41 will also translate a program that is a mixture of the two types of program described. This means that you can edit a captured HP 41 program without having to add line numbers and line feeds.

If the string in level 1 is not an HP 41 program, then →41 will try to translate it as a set of data registers or status flags sent to the HP 48 by one of the following HP 41 commands: PRREG, PRREGX, or PRFLAGS. The translated information is stored in the Emulator's data registers or flags. This works just like the automatic capture (by CPTR)of data sent to the HP 41.

→TXT takes a translated HP 41 program (an HP 48 Library Data object) from level 1 of the stack and turns it back into a text string. This does not work if the object in level 1 is not a translated HP 41 program, or if it is a PRIVATE program. In effect, →TXT is the opposite of →41.

Once a program has been turned back into a text string, you can examine it, edit it, and translate it back into a program by using →41.

VERS41 - Emulator Version

VERS4 returns to level 1 a text string giving the version number of your HP 82210A Emulator card. The string includes a Copyright notice.

Capturing Programs Under User Control

In Chapter 3 you saw how the CPTR key allows the HP 48 to capture an HP 41 program in one operation. Actually CPTR uses INPRT to read the program sent by the HP 41. If INPRT reports no problems in reading the text from the HP 41, then CPTR uses →41 to translate the program.

Automatic operation	Steps that make up the operation	Explanation
CPTR	INPRT	Reads the text string which contains the information sent by the HP 41. Checks that the string is correct. If a problem has occurred, stops and displays a message.
	→41	Decides whether the string is a program, data registers, or output from PRFLAGS. Translates the string accordingly, stores data, sets status and flags, or sets up an HP 41 program that the emulator can use, then stores the jump distance in each local GTO and XEQ so they go directly to the corresponding labels, and finally stores the translated program, and leaves its name on the stack.

The commands that make up CPTR

You can use INPRT and →41 instead of CPTR whenever automatic capture and translation would be inappropriate. Four such cases are suggested below.

Case 1. You want to capture a program, but you wish to modify it before translating it. In such a case you first use INPRT to capture the program as a text string. If the capture was successful, there should be a 1 in level 1. Edit the text string to make any changes to the program that you want. Finally use →41 to translate the edited HP 41 program.

Case 2. You want to capture some data and change it. Use CPTR to capture the data, then use RCL to recall the contents of individual registers, change those contents, and finally use STO to store the replacements in the registers. It may be simpler to capture everything as a text string with INPRT, edit the data while they are in the form of a single text string, then use →41 to transfer the edited data to the HP 41 data registers.

Case 3. You want to verify that you have captured a program correctly before you translate it. CPTR and INPRT give you considerable protection against incorrect signals, but under bad conditions such as bright sunlight, there is a chance that incorrect information might still be captured. One way to double-check is to capture *two* copies with INPRT and to check if they are the same.

First use INPRT to capture one copy of the program. Reject any copy which is definitely bad because there is a 0 in level 1. If level 1 contains the number 1, then drop this number. Now capture a second copy, rejecting any copies with 0 in level 1.

When you have two acceptable copies in levels 1 and 2, use the HP 48 command SAME (or ==) to check if they are the same. If the copies are not the same, then at least one was captured incorrectly even though INPRT did not detect an error. You cannot know *which* copy was incorrect, so it is simplest to capture two more copies for which INPRT does not detect any errors, then compare again. If you have a lot of trouble getting two copies that are the same then check through the advice given at the end of Chapter 3.

Once you have satisfied yourself that you have two copies that SAME reports to be the same, you use →41 to translate one of the copies. If you did not make a spare copy of one of the captured copies then use [LAST STACK] or [LAST ARG] to recover the captured strings.

Case 4. You want to capture several HP 41 programs, or several blocks of data registers, using an HP 48 program. CPTR is programmable, but if a problem occurs, it is not obvious whether the error occurred in capture or during translation. Write an HP 48 program that uses INPRT to capture what you print. The program can check what has been received and warn you if an error has occurred. If there are no errors, then the HP 48 program can make a duplicate copy of what has been captured, and use →41 to translate the captured program. If an error occurs in translation, then the HP 48 program can warn you, and let you edit the second copy of the HP 41 program. If no error occurs, then the HP 48 program can save the translated program, drop the duplicate copy of the original, and ask if you want to capture another program.

The book *Extend Your HP 41* (see Appendix B) shows how a programmable version of PRP can be achieved. It gives one version that works on any HP 41 and an extended version that requires an Extended Functions module or an HP 41CX. (Both programs are in Chapter 16, the shorter program is in the examples at the end of that chapter.) You could write an HP 41 program that sends all of your programs to an HP 48, provided the HP 48 is programmed to capture them.

Modifying and Writing Programs

Once you have captured a program and used it, you may want to modify it to use some of the special features of the emulator, or of the HP 48 itself. You would do this by turning the program back into a text string with →TXT, then editing the resulting text string, then translating the program again with →41. You might also want to capture an HP 41 program with INPRT and modify it at once, without first using the original version. You might even want to write a new HP 41 program or subprogram entirely on the HP 48. This section explains how you can do these things, and also describes problems that might occur.

Editing Programs

As mentioned earlier, →41 lets you make changes to programs without having to renumber all the lines and having to put in linefeeds. For example, a user might have the following HP 41 program:

```
01 LBL "OLD"
02 DEG
03 "TYPE NO 0-90"
04 PROMPT
05 TAN
06 END
```

On an HP 48 it is possible to replace step 3 with a more meaningful prompt, since the HP 48 lets you see more than 12 characters of the ALPHA register. Text strings in HP 41 programs are still limited to a maximum of 15 characters though, so a text string such as "TYPE A NUMBER 0 TO 90" must be entered as two HP 41 program steps.

Note

Before trying this example on your HP 48, read this entire section first, so you will know what to expect.

The program shown above could be edited to produce the following version:

```
01 LBL "OLD"
02 DEG
03 "TYPE A NUMBER " "→0 TO 90"
04 PROMPT
05 TAN
06 END
```

Step 03 has been replaced by two steps, the second step appends seven characters to the ALPHA register. Note that the character "→" is used in place of the HP 41 append character, since this is not provided by the HP 48. The character "▒" is also treated as an append symbol, because it is character 127, the same number as that used by the HP 41 for its append symbol.

Although step 03 has been replaced by two steps, →41 will still accept the step numbering in this edited program. In fact, →41 does not check step numbers for consistency at all, it simply looks for a step number at the beginning of each line, and removes the number if one is found. This allows you to edit programs in various ways, deleting any number of program steps or copying blocks of steps from one place in a program to another.

To simplify program editing further, →41 does not require an END at the end of a program (it adds an END if it does not find one). If you edit an HP 41 program so it has more than one END in it, →41 stops translating at the first END.

Linefeeds in a string representation of an HP 41 program are shown in the display as a small square block. When you edit such a string, the square blocks disappear, so each step of the program is on a separate line in the display. This is normal HP 48 behavior. If you are unfamiliar with text string editing on the HP 48 then read the HP 48 Owner's Manual.

Quotes and Counted Strings

If you edit a program with quote symbols in it, then the HP 48 puts it in the command line as a counted string. When you finish editing the string and press [ENTER], the HP 48 counts the number of characters in the string and, if this is different from the number in the original string, then the HP 48 adjusts the string size. If the edited string has fewer characters than the original, the number that specifies the size is adjusted. If the edited string is longer than the original, then the extra characters are excluded from the string and are treated as additional objects to be put on the stack. These may be treated as unquoted names, but if any are the same as the name of an HP 48 command or the name of an object then the HP 48 might try to evaluate them. For example, if the step END is dropped from the program string, then the HP 48 produces the following error message:

```
Invalid Syntax
```

To avoid these problems, go to the beginning of the counted string and replace the character count with a dollar sign ($). If you forget to do this and see something unusual on the stack after you press [ENTER], then you can recover the edited string by pressing [LAST CMD]. (Remember to remove the unwanted objects from the stack.)

Edited Commands

CPTR and →41 expect individual commands to be in their printed form. This means that you must be careful when editing a program. Particular points to watch for are:

- There must be a space between any function and its argument. For example, STO 99 cannot be written as STO99. Similarly, TONE 6 must not be typed as TONE6. Particularly confusing is the fact that X<>Y is a special function that is recognized, but the general function X<> expects a space before the argument, so that X<> Z is correct, whereas X<>Z is not recognized.
- Arguments must be positive numbers with no fractional parts. On the HP 41, and in the Emulator, you can put -3.1416 in register X, then execute ENG IND X just as if you were executing ENG 3. You cannot, however, get →41 to translate ENG -3.1416 successfully.
- Strings and arguments must be followed by a space or linefeed.
- Extra spaces are acceptable, and leading zeros can be dropped, or extra leading zeros can be added. For example XROM 2, 004 is acceptable, although it will be printed by the HP 41 as XROM 02,04.

Special Characters to Note When Editing

When you are editing a program and adding symbols that the HP 48 enters in pairs, such as apostrophes (' '), brackets ({ }), or quotes (" "), then you may need to remove one member of the pair. Use the [DEL] or [←] keys to remove the unwanted symbol.

Step Numbers

If a program step has no number at the beginning, then →41 will try to translate the step as if it was an ordinary HP 41 program step with the number missing. If you type a program with a line that contains no step number but which begins with an integer, then →41 will remove that number, assuming it was intended as a step number. On the other hand, if you type two steps with step numbers but without a linefeed between them, then →41 will assume the second step number is a number that the program is putting on the stack. Remember that the

Emulator accepts two numbers separated by a space as two separate numbers to be entered onto the HP 41 stack.

If the first line of a program has a step number, →41 looks for a step number immediately after every linefeed and removes anything that looks like a step number at those positions. For example, the following program is shown with some mistakes for the sake of the example. The numbers in bold are treated as step numbers, for the reasons given.

Program	Comments
01 LBL 'COUNT'	01 is the first step.
02 100 1000	02 is a step number at the beginning of the line, the other numbers are left alone,
STO 17	as is STO 17.
03 LBL 01	03 is a step number, 01 is not.
04 SIN	04 is clearly a step number.
05 LASTX	05 is a step number
06 DSE X	06 is a step number
07 GTO 01 'DONE'	07 is a step number. 'DONE' is a new step but with no number.
AVIEW	AVIEW is the same.
9 TONE IND X	9 is treated as a step number!
08 END	08 is a step number. The sequencing is ignored.

Numbers at the beginning of a line with a negative sign and with a fractional part are recognized as real numbers, not step numbers.

A program captured from an HP 41 with a Time module plugged in or from an HP 41CX will have a time and a date at the beginning. →41 looks for these at the beginning of a program and removes them in the same way as it removes step numbers.

To add a piece out of a program into the middle of another program, copy the complete program to the HP 48 stack by using →TXT. Split the program into two text strings, by editing the text string, going to the point at which to split it, pressing ↱ "" and ENTER. Recall another program object to the stack. Use →TXT, and edit the second program, keeping the part you need as a text string in level 1. Swap the text strings at the bottom of the stack and use + twice to add the extra piece of program into the middle of the original program. You can also duplicate pieces of a program by splitting the program up into several text strings, duplicating some parts, and reassembling the parts into a new version.

Re-entering HP 41 Mode after Editing Programs

If you leave the HP 41 Emulator mode, edit the current program, and re-enter HP 41 mode, then the Emulator will recognize that the current program has been changed and will display the message:

```
        name  missing|diff
```

where *name* is the name of the program that was edited. Normally this is only as a reminder. If you have not changed the current program and see this message, then it means that the current program cannot be found in the current directory or in the program library directory. This might mean that you have deleted the current program or you have changed the current directory. The current directory is displayed in the upper left portion of the HP 41 Emulator display. The message might also mean that you have changed the current path and the program library directory cannot be found on the new current path.

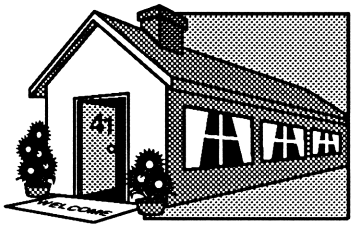
Setting up a Library of Programs

After you have edited a program and made a new version of it, you can store the program back in the variable containing the original program or you can give it a new name and store it in a new variable. At some stage you may wish to put some programs in your program library, not in the current directory. Chapter 2 describes the program library, normally kept in a directory called PRG41.

This program library can be used as a "core" library of HP 41 programs that you have checked on the HP 48 and now want to leave for use without making further changes. You can even make all the programs in this library private to avoid making unwanted changes to them unintentionally at a later stage. The next chapter describes how you can build up a library of XROM commands.

5

Extending the Environment



This chapter explains the reasons and the methods used to extend the emulator environment. You can extend the HP 41 Environment by adding functions that HP 41 plug-in modules and peripherals provide. Ways in which the HP 48 can use programs from the HP 41 Environment are also described.

This chapter covers the following topics:

- Extending the emulator.
- Simple extensions.
- Executing HP 48 operations from the emulator.
- CAT 2 commands.
- The XROM command.
- Time, extended functions, and extended memory.

Extending the Emulator: How and Why

An ideal calculator can perform any calculation you need at the touch of a button. The HP 41 provides over 120 built-in (CAT 3) functions, so there is a fair chance that the calculation you want is already provided. Plug-in modules and peripherals carry out additional operations (CAT 2), and you can write your own programs (CAT 1).

You can carry out any operation from CAT 1, 2, or 3 with a single keystroke.

The Emulator is designed to let you transfer all three kinds of operations to the HP 48. This chapter shows you how to add XROM (or CAT 2) commands to your HP 41 environment. The first section shows extensions that do not need to be XROM commands.

Some Simple Extensions

The CIRCLE program in Chapter 1 is a good example of a simple HP 41 extension. This is like using a built-in HP 41 function, and the program can be run at the touch of a button (R/S).

The CIRCLE program is a CAT 1 command, and with all the extra work involved, there would be little advantage in making it an XROM command.

Another example of a simple extension is the following stack comparison. The HP 41 has the tests X=Y?, X≠Y?, X<Y?, X>Y?, and X<=Y?, but X>=Y? is missing. Users who need this last test could add it as an extra CAT 2 function, but there is a much simpler way. If you need the step X>=Y? in an HP 41 program, use the two steps X<Y? X=Y? instead. The effect is exactly the same; the next program step after these two is executed only if X is greater than or equal to Y.

You can put the two steps X<Y? X=Y? into a program every time you need the test X>=Y? There is no need to write a special CAT 2 function, nor to use a separate HP 41 program.

Executing HP 48 Operations from the Emulator

Now look at another example. You want to display the date with the results of a program. DATE is not provided by the Emulator. Finding the date is a simple task on the HP 48, but there is no simple way to find the date on the HP 41CV. You must get the date from the HP 48.

The Emulator provides functions to let you send information between the HP 41 Environment and the HP 48. You can use two such functions (from the Emulator's HP 48 menu) to get the date from the HP 48.

48XQ takes the contents of the HP 41 ALPHA register and carries them out as an HP 48 operation.

48TOX takes the object from level 1 of the HP 48 stack and pushes it into the HP 41 Emulator's X register.

Therefore, the HP 41 steps:

```
        "DATE"   48XQ   48TOX
```

tell the Emulator to put the text string "DATE" into the ALPHA register and then to send it to the HP 48 to be carried out. The result is that the date is put in level 1 of the HP 48 stack. 48TOX then takes the date off the HP 48 stack and puts it into register X. You can use these three program steps to replace the DATE function in a program captured from an HP 41. The three steps can be used directly in a program to replace the single step DATE, or they can be turned into a CAT 1 program, or into an XROM command.

The following is a list of the commands provided to let the HP 41 Emulator communicate with the HP 48. These commands are in the HP 48 menu of the HP 41 Environment.

Menu key	Function	Description
X→48	XTO48	Copy the current contents of X to level 1 of the HP 48 stack. (Push X to 48 stack.)
48→X	48TOX	Copy the current contents of level 1 to X and drop level 1. (Pop the 48 stack to X.)
A→48	ATO48	Copy current contents of ALPHA to level 1 of the HP 48 stack. (Push ALPHA to 48 stack.)
48→A	48TOA	Copy the current contents of level 1 to ALPHA and drop level 1. (Pop the 48 stack to ALPHA.)
48XQ	48XQ	Carry out the HP 48 operation named in ALPHA.

You see the HP 48 menu when you first enter HP 41 mode. In HP 41 mode you can go to this menu from other menus by pressing [SHIFT] [HP 48]. Note that the names on the menu keys are not the same as the names of the functions. If you want to remind yourself of the name of one of these functions, hold down its menu key and you will see the name of the function, and then the message:

```
        Null
```

The commands work in the same way as the corresponding HP 48 commands described in Chapter 4, and have the same limitations. For example, 48TOA moves the first 24 characters of a text string to the ALPHA register. See Chapter 4 for details.

The Emulator provides a special command, called XROM, to let you write HP 48 programs that behave like CAT 2 functions.
For example, instead of writing the date command as a program (a CAT 1 command) you could write it as an XROM command by writing an HP 48 program:

```
        «XROM  "2612"  DATE  PSH41»
```

You would then store this program under the name 'xDATE' in your XROM directory (described in Chapter 2). The number 2612 tells the Emulator that this is a replacement for an HP 41 CAT 2 command with the XROM identifier 26,12.

CAT 2 Commands During Program Translation and Execution

This section shows how CAT 2 commands appear in HP 41 programs, and how they are dealt with by the Emulator during translation and execution.

The only XROM commands that the Emulator provides are the Printer module functions and the functions in the MODES menu (described in Chapter 2). These XROM commands are shown in CAT 7 of the emulator (CAT 2 is reserved for user written XROMs). If a program captured by the Emulator includes any other XROM command, you must provide instructions to carry out that command. If no instructions are provided, the captured program will stop and display an error message when it comes to the command.

There are six ways an XROM command can be referenced. The six subsections below give examples and descriptions of how the different XROM references are handled during translation and execution.

CAT 2 Commands Appearing in Programs as "Name"

Examples:

```
DATE
PRA
```

Explanation: These are CAT 2 functions. They behave the same way as CAT 3 functions, such as SIN or AVIEW, but they are not built into the HP 41CV. Instead, they are provided by plug-in modules, or are built into the HP 41CX as extra CX functions.

Translation: When a name is encountered during translation, the Emulator looks for the name in CAT 3, CAT 2, and CAT 7 in that order. If the name is found in CAT 2 or CAT 7, it is stored in the program as the corresponding XROM identifier, and is displayed (by SST, BST, PRP, and LIST) as "name". (If the name is found in CAT 3, it is stored in the program as an HP 41 token, and is displayed as "name".) If the name is not found at all, the translator displays a warning message and translates it as XEQ 'name'.

Execution: If the name has been translated as XEQ 'name', the name is looked for in CAT 1, CAT 2, and CAT 7. If the name has been translated as an XROM command, CAT 2 and CAT 7 are searched for an identifier. If the command is found, it is executed, otherwise the following error message is displayed:

```
Nonexistent
```

As with other errors, this error will be ignored if flag 25 is set.

CAT 2 Commands Appearing in Programs as XROM "name"

Examples:

```
XROM "PRPLOT"
XROM "ACOSH"
```

Explanation: These are CAT 2 programs. They behave the same as CAT 1 functions, but they are not written by the user. Instead they are provided in plug-in modules.

Translation: When XROM "name" is encountered during translation, the Emulator looks for "name" in CAT 2 and CAT 7. If "name" exists in the current XROM directory or CAT 7, XROM "name" is translated as XROM *mm,nn,* where *mm,nn* is the command's XROM identifier, and the step is displayed as "name" in the program. If "name" is not found during translation, the step is turned into XEQ 'name' in the translated program.

Execution: If the command was translated as XROM *mm,nn,* this identifier is searched for in catalogs 2 and 7. Otherwise "name" is searched for in CAT 1, CAT 2, and CAT 7. An error occurs if "name" cannot be found.

CAT 2 Commands Appearing in Programs as XROM *mm,nn*

Examples:

```
XROM 26,12
XROM 01,37
```

Explanation: If the plug-in accessory or module to which an XROM command belongs is removed from the HP 41, the command name cannot be found so the command is displayed and printed as XROM *mm,nn*. When the plug-in is replaced, the command reappears with its full name. The command can be a function or a program.

Thus, if you print an HP 41 program that includes DATE but without a Time Module in the HP 41, the program step is printed as XROM 26,12, not DATE. Since the module is not plugged in, the HP 41 cannot find the name of the function, and displays the function's identifier number instead.

Every CAT 2 function and program has an XROM number. XROM numbers are permanent. A list of HP 41 plug-ins and their identifier numbers is given in Appendix C.

You can transfer programs from an HP 41 to the HP 48 Environment with XROM steps in those programs, even if the relevant module is not plugged into the HP 41.

Translation: An XROM *mm,nn* command will be recognized as an XROM command during translation and will be stored as such in the translated program. If the XROM *mm,nn* command exists in your current XROM directory or in CAT 7, the command will show up in the translated program with its name as given in the XROM directory or in CAT 7. No message is given if XROM *mm,nn* is not found during translation (it is assumed that you will provide the XROM when the program is run). Just like an HP 41, the Emulator will try to find the XROM in CAT 2 when the program is executed. You can add the instructions that execute the XROM at a later time, in the same way as you can plug a module into the HP 41 only when you need it.

Execution: If XROM *mm,nn* is not found in the XROM directory when the program is run, the program stops and displays the following error:

```
Nonexistent
```

If you find a step that is displayed as XROM *mm,nn* in a captured program, look in Appendix C to see which module number *mm* this step comes from. Refer to the manual for that module to identify the command by its number *nn* and to read what it does. If the command is a program, you can try to capture it on the HP 48, but you will have to replace XROM *mm,nn* with XEQ 'name'. If the command is a function, you will have to replace it.

CAT 2 Commands Appearing in Programs as XEQ "name"

Examples:

```
XEQ "DATE"
XEQ "ACOSH"
```

Explanation: If the required plug-in is not connected to the HP 41 while a program is being entered from the keyboard, the command appears as XEQ "name". In such a case, when the program is run on an HP 41, the HP 41 looks for "name" in CAT 1 first, then in CAT 2. The command remains in the program as XEQ "name" even when the module is plugged in.

Translation: The Emulator stores the step as XEQ 'name' in the translated program.

Execution: "name" is looked for in CAT 1, CAT 2, and CAT 7. If the command is found, it is executed, otherwise the Emulator stops and displays:

```
Nonexistent
```

Note that by turning any unrecognized command (except XROM *mm,nn*) into an XEQ command, the Emulator gives you the option of implementing the command as a CAT 1 operation or as a CAT 2 operation. You can write an HP 41 program to carry out the command in the same way as on an HP 41. Alternatively, you can use the Emulator's XROM command to write an HP 48 program using HP 48 features that are not otherwise available to the Emulator.

CAT 2 Commands Appearing in Programs as GTO "name"

Examples:
```
GTO  "PRPLOT"
GTO  "ACOSH"
```

Explanation: GTO "name" is different from XEQ "name". In this case "name" can only be the "name" of a global label in CAT 1, or of a global label in a program in CAT 2. This command is used to let an HP 41 program jump to another program, including one in a plug-in module or accessory. There is no special XROM form of GTO.

Translation: GTO "name" is translated as GTO 'name' by the Emulator.

Execution: During execution this command looks for a global label name in the Emulator's CAT 1. The Emulator does not allow you to put HP 41 programs in CAT 2 (they are all in CAT 1), so there is no need to search CAT 2 during execution of GTO.

Non-programmable CAT 2 Commands

Examples:
```
PRP
LIST
```

Explanation: HP 41 CAT 2 commands that prompt for an argument cannot normally be included in programs. The Emulator, however, allows you to write prompting XROM commands (shown later in this chapter).

Translation and Execution: If you do succeed in including the commands PRP or LIST in an HP 41 program and capture (or edit the commands into) the program on an HP 48, the commands will be translated according to the rules in the above subsections. The commands PRP and LIST are ignored when a program containing them is executed by the Emulator.

Note An HP 41 program that has been translated can be turned back into a text string by →TXT. When →TXT turns a program into a text string it behaves like PRP; if a program step "XROM *mm,nn*" is found in the current XROM directory or in CAT 7, it is replaced by its command name.

The Emulator's XROM Command Described in Detail

A CAT 2 command is written on the Emulator as an HP 48 program that begins with the command "XROM" and is stored in the XROM directory. The command identifier is put immediately after XROM in the HP 48 program, followed by its instructions.

The DATE example can be carried out by the program shown earlier:

```
«XROM   "2612"   DATE   PSH41»
```

This program should be stored in your current XROM directory with the name 'xDATE'. The XROM directory has the default name HP41XROM and can be accessed from the HP 48 by pressing [VAR] HP41X .

When the program step DATE or XEQ 'DATE' is found during program translation or execution, the Emulator will look for the name DATE in CAT 1 and then for the name 'xDATE' in the XROM directory. The Emulator automatically changes the name DATE to xDATE when searching the XROM directory because DATE is an HP 48 reserved word.[1] If the program step XROM 26,12 is found, the Emulator will look through the XROM directory, searching for a program that begins with XROM "2612". The program step XROM 26,12 will be displayed as the name of the program, and will be executed as the instructions that follow XROM "2612".

The HP 48 DATE command brings the date to level 1 of the HP 48 stack as a number. Then PSH41 takes this number off the HP 48 stack and pushes it into register X of the HP 41 stack. The number is removed from the HP 48 stack.

XROM must be at the very beginning of a program to identify that program as an XROM program used by the HP 41 Emulator. XROM must be followed immediately by a text string that identifies the command. The text string must have the form "*mmnnp*". The module identifier is *mm.* The command number in the module is *nn,* and *p* is an optional code that defines what information the command prompts for if it is a prompting command The module identifier *mm* must be a number between 00 and 31. The command number *nn* must be a number between 00 and 63. If *p* is given, it must be a letter between A and G. Appendix C contains a list of accessories, modules, and their corresponding module numbers *mm.*

If you write an XROM program that does not follow these rules, when you try to enter the program, you get the error message:

```
        Invalid Syntax
```

If you write an HP 41 program to extend the features of the Emulator, you must put it in CAT 1 (the current directory or the program library directory). The Emulator translates XROM "name" into XEQ 'name' so that "name" will be searched for in CAT 1.

Here is another example of an XROM command. The plug-in Math module for the HP 41 contains programs to calculate hyperbolic and inverse hyperbolic functions. If you transfer a program with XROM "ACOSH" to the Emulator with a Math module plugged into the HP 41, the translated program will contain the step

[1]To discover how the Emulator changes other global labels that are HP 48 reserved words, use →41 to compile a test program consisting solely of the global label. The name that appears in level 1 is the legal HP 48 label.

XEQ 'ACOSH'. You can write an ACOSH command for the Emulator as follows:

```
«XROM "0137" "X" RCL41 ACOSH "X" STO41»
```

You would store this program in the XROM directory under the name 'xACOSH'. This program takes the value from stack register X, calculates its arc hyperbolic cosine, and puts this new value in X.

If you transfer a program from your HP 41 to your HP 48 while there is no Math module plugged in, the Emulator captures XROM 01,37 instead of XROM "ACOSH". If you always use the correct XROM identifier when writing XROM commands, the Emulator will translate them properly.

Caution There are 4 versions of the Math module. The version letter comes at the beginning of a CAT 2 listing of the module's commands. Only versions A and B define ACOSH as XROM 01,37. Other commands have also been renumbered in version C and D. You should always check the name of any command in programs using Math modules. The Math module is the only HP 41 module made by Hewlett-Packard where this occurs. See Appendix C for a list of XROMs.

Prompting XROM Commands

The HP 41 allows you to write XROM commands that behave like HP 41 XROM commands that cannot be included in programs. These commands usually prompt for a parameter, then execute the required command. For example, the command PRP prompts for the name of a global label, then prints the whole program that contains that label.

The Emulator provides an additional parameter code to let you write prompting XROM commands. If *p* is included in *mmnnp* then the Emulator recognizes the command as one that prompts for a parameter. To determine what sort of prompt is produced, *p* must be a letter from A to G.

A prompts for a register. The prompt can be filled with a two-digit number, a stack register name, or IND, followed by a two-digit number or by a stack register name.

B prompts for a flag number. Any two-digit number can be given, or IND followed by a two-digit number or by a stack register name.

C prompts for a single digit. The Emulator lets you supply the numbers 10 and 11 by pressing keys in the top two rows. Indirect parameters can be supplied by IND followed by a two-digit register number or by a stack register name.

D prompts for a two-digit number, or a letter from A to J and a to f. An alpha parameter of up to seven characters is also allowed, but the single letters just listed will be treated as local labels. Indirect parameters are also allowed.

E prompts for a three-digit number. The prompt is turned into a four-digit prompt if [EEX] is pressed.

F prompts for a four-digit number.

G prompts for an alpha parameter of 0 through 7 characters. The alpha key must be pressed twice, even if no characters are pressed in between.

The response given to the prompt is passed to level 1 of the HP 48 stack and then the XROM command is executed. Execution begins with the first command after XROM and the string that follows it. The prompt is passed to the HP 48 as a real number or a text string. If the prompt is given an indirect parameter, such as IND 17, 128 is added to the value. IND 17 would be put on the HP 48 stack as the number 145 (17+128). Registers X, Y, Z, T, and L are passed to the HP 48 as numbers, as follows:

L=123, X=124, Y=125, Z=126, T=127

Indirect stack registers are passed as the above numbers with 128 added to them. Thus the parameter IND Y given in response to a prompt causes the number 253 (125+128) to be put in level 1 of the HP 48 stack.

Here is an example. Create an exponentiation function ST^ that would raise the contents of a register to the power of the number given in register X. To do this you could write the HP 48 program:

```
«XROM "0146A" DUP RCL41 "X" RCL41 ^ SWAP STO41»
```

XROM "0146A" means that this is to be treated as an XROM command, with the identifier 01,46; the A prompts for a register. Type this program into your HP 48 and go into the XROM directory. The program cannot be given the name ST^, because ^ cannot be used in names. Instead, give it the name STE:

['] S T E [STO]

Now go back to the previous directory. Enter HP 41 mode and store the number 2 in register 21, then put the number 3 in the X register. Type [XEQ] [α] S T E [α]. You will see the prompt:

```
STE __
```

Enter 21, in response to the prompt. Recall the contents of register 21. The value will be 8, which is 2^3. The XROM program did the following:

- The prompt operation took the argument and passed it to the XROM program by putting the number 21 in level 1 of the HP 48 stack.
- The program duplicated this number (DUP), then recalled the value from the register (RCL41), putting it in level 1 of the HP 48 stack. In this case the number 2 was recalled from register 21 and put in level 1.
- The program recalled the number from HP 41 register X ("X" RCL41), putting the number 3 in level 1.
- The program calculated 2 to the power 3 (^), leaving 8 in level 1.
- Finally, the program stored this result back in register 21 (SWAP STO41).

Prompting XROM Commands in HP 41 Programs

A prompting XROM command will prompt for a parameter if it is executed from the keyboard. The Emulator also lets you include prompting XROM commands in HP 41 programs. A prompting XROM command does not prompt for a parameter when used in a program. Instead, it assumes the parameter has already been put in level 1 of the HP 48 stack.

Note Ordinary HP 41 functions take their arguments from the HP 41 Environment, most often from register X. This includes ordinary CAT 2 functions. Only the prompting XROM commands expect their parameters to be on the HP 48 stack. If you include a prompting XROM command in an HP 41 program, you must make sure that the parameter needed by the command is in level 1 of the HP 48 stack.

XROM Examples and Advice

This section shows three examples of XROM commands written by a user. Each example is followed by notes describing the techniques used.

Example: Random Numbers

Charlotte New has many HP 41 programs she wants to use on her new HP 48. Some of her game programs use random numbers. These programs contain the command XROM "RNDM" that generates a random number and puts it in stack register X. Charlotte needs a similar command for use with the HP 41 Emulator in her HP 48. She writes the program:

```
«XROM   "0001"   RAND   PSH41»
```

Assuming she is using the HP 41 Emulator from her home directory, she does the following to store this program in her XROM library under the name RNDM, then go back to her home directory.

HP41X ['] R N D M [STO] [←] [HOME]

Charlotte can now capture any HP 41 program that contains the step XROM "RNDM" and the step will be translated into RNDM by the Emulator.

The XROM program uses the HP 48 command RAND to generate a random number and put it in level 1 of the HP 48 stack. Then it uses the command PSH41 to put this number in register X of the HP 41 stack and to remove it from the HP 48 stack.

This RNDM command does not change LASTX.

Notes

1. The command "RNDM" exists in several Applications modules. Charlotte had a choice of using any of those XROM identifier codes for her RNDM command. Rather than choose one and be wrong in some cases she chose a completely different number, XROM 00,01.
2. Using module number 00 for your own XROM commands eliminates the chance that your commands will be mistaken for XROM commands from some HP 41 plug-in accessory or module.

   The command number 00 is usually reserved for a special function that has the name of the module, but does nothing. The Emulator does not use real modules, so it does not need a special name for each module.
3. Charlotte has used the built-in HP 48 command RAND to generate a random number. This works differently from the "RNDM" programs in the Standard, Games, and PPC modules.
4. Charlotte could have written an HP 41 "RNDM"program, or she could even have captured the "RNDM" program from one of the modules. She chose not to do so for two reasons. First, executing an HP 41 program uses one level of the HP 41 return stack. Executing an XROM command does not use the return stack. Since the return stack has only 6 levels, avoid using it, if possible. The second reason to avoid using the "RNDM" program is that it uses an HP

41 data register to store the current random number as a "seed" for the next random number.

Some of Charlotte's HP 41 programs check the status of two user flags. Charlotte wants to have a new HP 41 command, similar to FS?, namely F=? that would compare two flags and carry out the next step only if their status was equal. Unfortunately, Charlotte's name for this command, F=?, will not be accepted by the HP 48 because it contains the symbol =. Instead, Charlotte chooses to use the name FEQ?.

FS? or FC? require only one flag number. They prompt for the flag number and include it in the command. The command FEQ? needs two flag numbers, so Charlotte wants it to take those flag numbers from the stack. Charlotte decides that FEQ? should take the flag numbers from stack registers Z and T. Here is a program that takes a number from register 20, then takes its sine and adds it to the number originally in register X, but only if both flags 05 and 06 are set. Otherwise, the program adds the number itself to register X without taking its sine:

Program step	Explanation
`01 LBL "ADSIN"`	Program that adds a number or its sine to X.
`02 5`	Put flag number 5 in register X.
`03 RCL 20`	Recall the number from register 20.
`04 X<>Y`	Put this number in Y, and get 5 back in X.
`05 6`	Put 6, the other flag number, in X. 5 is now in Y.
`06 RDN`	Move 6 to T, 5 to X, the number to Y.
`07 RDN`	Move 6 to Z, 5 to T, the number to X.
`08 FEQ?`	Are the two flags equal?
`09 SIN`	If so, take the sine of the number in X.
`10 +`	Add the number, or its sine, to the value originally in register X that is now in register Y.
`11 END`	

The XROM command requires a fairly complicated program. The XROM command must compare the status of two HP 41 flags, and check for errors. If an error is detected, the right number of arguments must be dropped from the stack, a message must be displayed, and the program must stop. The command must also check whether it is being carried out from the keyboard or from a program that is being single stepped or run. If executed from the keyboard, the command must display the message `Yes` or `No`. Here is what Charlotte wrote:

```
Program steps                           Explanation
«XROM   "0010"                          Charlotte's 10th XROM.
  "T"  RCL41                            Get flag # from register T.
  "Z"  RCL41                            Get flag # from register Z.
  IFERR                                 Allow for errors.
   FS?41                                Get the flag status of the
   SWAP  FS?41                          flag numbers in T and Z.
  THEN                                  If this caused an error,
                                        drop
    IF  -55  FC?  THEN  DROP  END       bad argument if LAST
                                        ARG set, and drop the
   DROP                                 other argument, show an
    "Data  Error"  ERR41                error message and quit.
  END                                   Else no error, end IFERR
                                        structure.
IF  51  FS?41  58  FS?41  OR  NOT       Check if SST or program
                                        running.
   THEN                                 If not, this is keyboard
     IF                                 execution, check if the
       ==                               two HP 41 flags are equal,
       THEN                             and if flags equal then
         "Yes"                          set up a Yes message,
       ELSE                             otherwise, set up
         "No"                           a No message,
     END                                end the message setup,
     MSG41                              display the message, and
   ELSE                                 quit.
     IF                                 Get here if SST or running
       ≠                                program, if the two HP 41
                                        flags are unequal,
       THEN                             skip the next HP 41
                                        program step
         SKIP41                         using SKIP41, otherwise
                                        do nothing so next 41 step
     END                                is executed. End flag test.
  END                                   End test on program/
                                        keyboard execution.
»                                       End program .
```

Having written the program, Charlotte stores it in her XROM directory under the name FEQ?. Then she goes back to the directory from which she runs her HP 41 programs. Then she tests it by running the program ADSIN, as shown above.

To see whether a program has been correctly entered, check its length and its checksum. After you have typed the program in, compared it with the original version printed here, and stored it in a variable, you should put the program name on the stack again and then execute the BYTES command by pressing ⬅ MEMORY BYTES . For 'FEQ?' the length should be 238.5 and the checksum should be #39762d (the small d means decimal).

Notes

1. The program can go wrong if either register T or register Z contain an alphabetic value or a reference to a nonexistent flag. Both values are tested within one IFERR. If either value causes an error, it is necessary to drop the value that caused the error, but only if LAST ARG is enabled (flag -55 is clear); otherwise, the bad value will already be lost. The other flag number must be dropped too. When both of these have been removed from the HP 48 stack, an error message can be set up. ERR41 displays this message and immediately exits from the XROM program.
2. The instructions following the IFERR...THEN...END can only be reached if the flags were tested successfully. There is no need for an ELSE after the IFERR.
3. If the XROM is being carried out from a running program, and if the two flags tested were not equal, SKIP41 is used to skip over the next HP 41 program step. By default, if the two flags are equal, nothing is done, so the next HP 41 program step is carried out. This is the simplest option, but it has been left to the end of the XROM program, since it is a do-nothing option.

After Charlotte has written a lot of XROM commands, she finds that it takes a while for the Emulator to find those commands that are a long way down the XROM directory. She writes an HP 41 equivalent of the HP 48 ORDER command with a prompting XROM. The command prompts for the name of a single XROM and then moves that XROM to the front of the XROM directory. Any XROM command near the front of the XROM directory is found sooner than a command far down in the directory.

Program steps	Explanations
`«XROM "0060G"`	60th XROM, with alpha prompt for name.
`"'" SWAP OVER + + OBJ→`	Turn the string into a name.
`1 →LIST`	Put the name into a list.
`PATH SWAP`	Get current path so we can return to it.
`HP41XROM`	Move to the XROM library to ORDER it.
`IFERR ORDER`	ORDER, but allow for an error, and
`THEN`	if an error occurred, then
`IF -55 FC?`	if LAST ARG enabled (flag -55 clear),
`THEN DROP END`	drop the bad value from level 1,
`EVAL`	EVAL path to return to current directory,
`"Undefined Name" ERR41`	set up error message, display it, and quit.
`END`	End of IFERR
`EVAL`	It worked, return to current directory.
`»`	Program ends.

Store the program in the XROM directory with the name XORDER. It should have the length 143.5 and the checksum #767d. Now return to the directory in which you do your HP 41 work.

Now Charlotte can go to CAT 2, press the key marked XORDE , then fill the prompt with the name of an XROM command she uses often. The command will be moved to the front of the XROM directory.

Notes

1. The change in order of CAT 2 will not be seen until you execute CAT 2 again.
2. Since the current directory is unlikely to be the XROM directory, this program saves the current path, moves to the XROM directory, does the reordering there, then uses EVAL to evaluate the path list and return to the current directory.

3. This XROM is intended for use as a prompting command, but Charlotte can use it in an HP 41 program too, if she wishes. In that case, though, the name of the XROM command to be put at the front of the XROM directory must be in level 1 of the HP 48 stack.
4. XROM commands are HP 48 programs, so their names are not limited to 7 characters. HP 41 prompts can only accept arguments of 7 characters or less.

Time, Extended Functions, and Extended Memory

The Extended Functions module adds functions that let the HP 41CV handle text, move blocks of registers, and use data files stored in Extended Memory. The HP-IL module allows the HP 41 to communicate with other devices and control instruments. The Time Module allows the HP 41CV to provide general timekeeping and appointment capabilities.

Using similar features built into the HP 48 is better than trying to emulate these features inside the HP 41 Environment. You can write XROM commands to emulate most Time functions and Extended Functions if you need them. This section provides some examples. HP-IL functions are more difficult to emulate (see Appendix D).

Time Module

To write an XROM that behaves like XYZALM, you need to create HP 48 message alarms. XYZALM expects the alarm time in register X, the alarm date in register Y, and the repeat interval in register Z. The date must be set according to the current mode; if the date is zero, the alarm is set to execute on the same date as it was set. If the ALPHA register contains any text, that text is displayed as a message.

The following XROM program emulates these actions.

Program steps	Explanation
`« XROM "2629"`	XROM 26,29 that is the XYZALM XROM identifier.
`"Y" RCL41`	Get register Y.
`IF DUP 0 ==`	If it is not, type 0,
`THEN DROP DATE`	then replace it with the current date,
`END`	otherwise leave it as is.
`1 →LIST`	Put the date in a list.
`"X" RCL41`	Get register X.
`+`	Append it to the list, whatever its value.
`A41→ +`	Get the ALPHA register and append it to the list.
`"Z" RCL41`	Get the Z register.
`IF DUP TYPE`	If it is not type 0,
`THEN DROP2`	drop it and drop the list,
`"Bad Data" ERR41`	signal an error, and quit via ERR41,
`END`	otherwise end the IF and carry on.
`HMS→ 29491200 * +`	Convert Z to ticks and append to the list,
`IFERR STOALARM`	Now use the list to set up an alarm,
`THEN`	if this causes an error,
`IF -55 FC?`	first check if LAST ARG is enabled,
`THEN DROP`	and in that case drop the list,
`END`	end LAST ARG check,
`"Bad Data" ERR41`	signal an error, and quit via ERR41.
`ELSE`	If no error, alarm is set,
`DROP`	drop the alarm number left by STOALARM.
`END`	End of IFERR.
`»`	End of program.

Store the program in the XROM directory with the name XYZALM. The program length should be 228.0 bytes and its checksum should be #24568d.

Notes

1. The program sets up the following list:

    { date time message repeat }

It uses the values in Y (date), X (time), alpha (message), and Z (repeat) to create the list. The repeat value is converted from HH.MMss format to decimal hours, then to HP 48 "ticks." A repeat value of zero is left as zero, since this is treated as no repeat by the HP 48.

2. The HP 48 command STOALARM uses the above list to set up an alarm. The contents of the ALPHA register are always copied to the HP 48 stack as a text string, so they are automatically treated as a message, making this a message alarm.
3. Unlike the HP 41 function, this program does not treat a negative number between 0 and -12 as a p.m. time. A negative date is treated as an error, as on the HP 41, and a negative repeat time is treated as a positive one.

Extended Functions

The Extended Function XTOA takes a number in register X and appends the corresponding character to the ALPHA register. This allows for the construction of alpha displays that could not be made using only the ordinary HP 41 commands.

To include characters such as quote marks and brackets in the display of HP 41 program results, you must write an XROM .

The program has to handle the contents of X when X is a text string, a number, or an error. If X is a number less than 256, it will be converted into a character.

Program steps	Explanation
`« XROM "2547"`	XTOA is XROM number 25,47
`"X" RCL41`	Get value from X.
`IF DUP TYPE NOT`	"If the value is a number..."
`THEN ABS IP`	
`IF DUP 255 >`	If the value is greater than 255,
`THEN DROP`	drop the value and
`"Data Error" ERR41`	signal an error via ERR41.
`END`	End of the character number test
`CHR`	Convert the value to a string.
`END`	End of numeric value conversion.
`A41→ SWAP +`	Append string to ALPHA.
`DUP SIZE DUP 23 -`	Do not use more than 24
`SWAP SUB`	characters,
`→A41`	and put result in ALPHA.
`»`	End of program.

Store this under the name 'XTOA' in the XROM directory.
Size: 164.5 bytes, checksum: #8737d.

Notes

1. ERR41 forces the XROM program to quit at once, so the last part of the program is reached only if there was no error. The text string in level 1 is appended to anything that is already in the HP 41 ALPHA register. The last 24 characters of the string are extracted (if it is more than 24 characters long) and the result is put back in ALPHA.
2. Not all numbers produce the same display character on the HP 41 as on the HP 48. HP 41 character numbers are given in the printer manuals, and in Volume 2 of the HP 41CX Owner's Manual. HP 48 character codes are given in Appendix C of the HP 48SX Owner's Manual.

Extended Memory

The Extended Functions module allows you to store collections of related data in files kept in Extended Memory. The HP 48 does not provide files, but it keeps collections of related objects in lists, so lists can be used instead of files to emulate Extended Memory.

There are three kinds of HP 41 Extended Memory files: data files, text files and program files. Data files are the most similar to HP 48 lists. A data file can hold one HP 41 data object in each position, just as an HP 48 list can hold one HP 48 object in each position. The pointer in an HP 41 data file can be replaced by a pointer used by GETI and PUTI on an HP 48. Text files can be turned into lists as well, since a list can be used to contain a collection of text strings and nothing else. Text file pointers are more difficult on an HP 48, because a record pointer and a character pointer are needed.

Extended Memory keeps a pointer to the current files. Some Extended Memory commands specify which file is to be used, and make it the current file. Other commands use the file that is already the current file. One way for the HP 48 to provide a pointer to the current file is to have an additional Emulator variable that contains the name of the current file. In the examples that follow, this variable is called XM41.

Program files on an HP 41 are similar to the Library Data Objects on the HP 48 used to store HP 41 programs.

CRFLD is a command used to create a data file whose size is given in register X and whose name is given in the ALPHA register.

Program steps	Explanation
`« XROM "2511"`	CRFLD is XROM 25,11.
`"X" RCL41 DUP`	Get the value from X.
`IF TYPE`	Check if it is a real number,
`THEN DROP`	If not, drop it,
`"Bad Data"`	signal an error, and quit via ERR41.
`ERR41`	
`END`	End the check.
`[ 0 ]`	Create an array, to be used later.
`SWAP 1 →LIST`	Make a list containing required size,
`RDM OBJ→`	redimension array to this size, split it up,
`OBJ→ DROP →LIST`	use array size to make a list of this size,
`1 2 →LIST`	put in list with a pointer and original list.
`A41→ SIZE 0`	Get name from ALPHA, check its length.
`IF ==`	If name is zero characters long,

```
      THEN DROP               drop the list,
        "Bad Name"            complain, and quit via ERR41.
ERR41
    END                       Otherwise, name is acceptable.
   A41→ "'" SWAP OVER         Get name string from ALPHA, turn it
+ +                           into a
    OBJ→ SWAP OVER            name, dup name, and store the list
STO                           under it.
   'XM41' STO                 Store dup file name in the pointer
                              variable.
»                             End of program.
```

Store this with the name CRFLD in your XROM directory.
Size: 225.0, checksum: #23222d.

Notes

1. This creates a list of *n* objects, where *n* is the number in register X. The list initially contains *n* zeros. The list is then stored, together with a pointer, in another list. The pointer is initially set to point to the first item in the data list. The list looks like this:

   { { 0 0 0 0 ... *n* zeros } 1 }

   The next example will show how this can be used.

2. The program creates a short array, redimensions this array to be of the required size, then expands the array into *n* zeros on the stack and turns this into a list of *n* zeros.

3 Like other variables, the list and the pointer are stored in the current directory. They will only be available if they are on the current path.

4. The list name is stored in 'XM41' as a quoted name. If XM41 is evaluated, it evaluates to this name, which itself evaluates to the list. To obtain the name, you have to use the quoted name 'XM41', then use RCL to recall the file name.

5. If there is not enough free RAM in the HP 48 for the data file to be created, this version of CRFLD causes an Insufficient Memory error that aborts the HP 41 Environment. A more elaborate version of CRFLD might use the IFERR...THEN...ERR41 END construct to trap this error and return the "No Room" error inside the HP 41 Environment.

To use this data file you need to be able to put data in it and to remove data from it. SAVEX puts a data item from register X into the list, and moves the pointer one place.

Program steps	Explanation
`« XROM "2541"`	SAVEX is XROM 25,41.
`  IFERR`	Use IFERR to check if the pointer exists,
`    'XM41' RCL`	recall the current list name to the stack
`   THEN`	if an error occurs,
`    IF -55 FC?`	check if LAST ARG enabled and, if so,
`     THEN DROP END`	drop bad value from stack,
`    "No Current File"`	tell user current file pointer is bad
`   ERR41`	and quit via ERR41.
`   END`	End test on pointer to file.
`   DUP TYPE`	Get pointer value type.
`   IF 6 ≠`	If it is not 6,
`    THEN`	then the pointer is not a name,
`     DROP`	so drop the object,
`     "Not a Name"`	set up a message,
`      ERR41`	and quit via ERR41
`   END`	End test.
`   RCL DUP TYPE`	Get the object, get its type.
`   IF 5 ≠`	If it is not 5,
`   THEN`	the object pointed to is not a list,
`    DROP`	so drop the object,
`    "Not a File"`	set up a message,
`     ERR41`	and quit via ERR41.
`    END`	End test to see if it is a list.
`   OBJ→ DUP`	Split the list and see
`   IF 2 ≠`	if it contains 2 objects (pointer and contents)
`   THEN →LIST DROP`	If not, reform the list and drop it,
`   "Not a Data File"`	set up a message,
`    ERR41`	and quit via ERR41.
`   END DROP`	End list size check, drop list size.
`  DUP2 SWAP SIZE`	Get list pointer and list size,
`   IF >`	if pointer greater than size,
`    THEN DROP2`	pointer past end of file, drop pointer
`     "End of Fl"`	and list, set up error message,
`        ERR41`	and quit via ERR41.

```
    END                    End of test on pointer position.
   "X" RCL41               Get the value to be put in the list.
    IFERR                  Use IFERR to check for any errors in PUTI.
      PUTI                 Put value in list and advance pointer.
      THEN                 If this caused an error,
        IF -55 FC?         check if LAST ARG enabled and, if so,
         THEN
          DROP2 DROP       drop 3 objects (object, pointer, list),
        END                end LAST ARG test,
        "File Error"       tell user something is wrong with current
         ERR41             file, and quit via ERR41.
    END                    End test on PUTI.
   IF -64 FS?              If wraparound flag set so index is 1,
    THEN
      OVER SIZE +          replace the 1 with list size + 1 as on
                           HP-41.
    END                    End wraparound test.
  2 →LIST                  Combine list and pointer into a list,
   'XM41' RCL              get the file name from the file pointer,
    STO                    and store the updated list.
»                          End of program.
```

Store this program in your XROM directory under the name 'SAVEX'. Size: 441.5 bytes, checksum: #7717d.

Notes

1. This is a long program, and nearly all of it is concerned with possible errors or unusual conditions.
2. The program has to use quoted names and RCL to recall the name of the data file from the pointer XM41. Otherwise the object in XM41, or the object in the name, might come to the stack. As the program stands, the first error message occurs if there is no pointer at all, the second if the pointer exists but does not contain a name, and the third if the name does not point to a list object.
3. The pointer inside the file is updated automatically by PUTI, but if the end of the list is reached, the HP 48 resets the pointer to 1, so that it points to the beginning of the file. This program checks flag -64 to find if that has occurred. If it has occurred, the size of the list is added to by 1, so that the pointer points past the end of the data file, as is done by the HP 41 Extended Function SAVEX.

SAVEX assumes that the current file is the correct one and that the pointer inside the file points to the correct position. You can write other Extended Memory control functions to select files, move pointers, and recall data, using the above two programs as models.

More About Extensions, Including PRPLOT

You can write additional commands using the methods described. If you still have trouble, consult the book "HP 41/HP 48 Transitions." It provides many examples of ways to replace HP 41 commands with HP 48 commands or programs. See Appendix B for details of this book.

If you want to write an XROM command that behaves in the same way as an XROM command on your HP 41, start by giving this command the same XROM identifier. An easy way to find the XROM identifier of a command is to include the command as a step in a program, then turn off the HP 41 and remove the module that provides the command. Turn on the HP 41 again, set PRGM mode, and you will see the program step replaced by XROM *mm,nn*. *mm* and *nn* are the numbers you need for your Emulator XROM command. Also, you can count the command numbers *nn* by running through CAT 2. The first command in each module is the module name, which has the number 0.

Appendix C includes a list of the module identifiers (*mm*) for most of the modules available. If you have a mysterious XROM number in your program, look for its module identifier in this list.

Interaction with the HP 48

One important way to use the Emulator with the HP 48 is to let the HP 48 call programs from the HP 41 Environment. The Emulator commands described in Chapter 4 let an HP 48 program send information to the HP 41, take results from the HP 41, and execute HP 41 programs.

Suppose you have a reliable HP 41 subroutine for pipe sizing. After you capture this on an HP 48, you can write an HP 48 program that prompts for pipe parameters using the larger HP 48 display. You can send these parameters to the HP 41 environment with STO41 and

PSH41. This allows you to execute the HP 41 subroutine from the HP 48 program with XEQ41. You can get the results back to the HP 48 by using RCL41, or the HP 41 program can print the results from within the HP 48.

The HP 41 Environment is not re-entrant. This means that an HP 41 program cannot execute an HP 48 operation that in turn executes an HP 41 operation.

Writing Extension Commands in the HP 41 Language

You should be aware that some extensions cannot be carried out using the XROM commands. The printer module command REGPLOT is one example. REGPLOT needs to use HP 41 style printer operations, such as ACCHR, that are provided by the Emulator, but not by the HP 48. You would have to use the Emulator and write an HP 41 program (or an HP 41 program with one or more XROMs) to carry out the required operation. As a final example, here is an HP 41 program that provides the Emulator with a simplified REGPLOT command.

Program steps	Explanation
`01 LBL "REGPLOT"`	
`02 RCL 00`	Get the lower Y limit.
`03 -`	X=val-lowlimit.
`04 RCL 01`	Get the upper Y limit.
`05 RCL 00`	Get the lower Y limit.
`06 -`	X=(uplimit-lowlimit).
`07 RCL 02`	Get the number of columns +-NNN.AAA.
`08 ABS`	
`09 INT`	
`10 /`	X=Scale factor.
`11 /`	X=Col. no. to plot on inside plot area.
`12 3.5`	
`13 -`	X=No. of cols. to skip to reach plot symbol position.
`14 FIX 0`	
`15 RND`	
`16 X<0?`	
`17 0`	Pegs to lower limit.
`18 SKPCOL`	Accounts for space before plotting symbol.
`19 120`	Lower case x.
`20 ACCHR`	Plotting symbol.

```
21 RDN          X=Cols before plotting symbol, again.
22 7
23 +            Add cols used by plotting symbol.
24 RCL 02       Plotting width.
25 ABS
26 INT
27 X<>Y
28 -            X=Cols after plotting symbol.
29 SKPCOL
30 ADV          Print right justified.
31 END
```

The program is similar to the HP 41 REGPLOT command, but if you run it on an HP 41 the ADV in step 30 makes the print buffer overflow and causes a left justified print. The Emulator's larger printer buffer does not overflow when this program is run.

The REGPLOT program uses one subroutine return level, whereas the REGPLOT function is an XROM that does not use any return levels.

This program uses only a lower case "x" as a plotting symbol. The contents of register 03 are ignored. If you want to use your own plotting symbols you can rewrite the program. RCL the contents of register 03, use SIGN to check if they are numeric. If SIGN returns 0, register 03 contains a BLDSPEC value and this can be appended to the plot string. This program does not print the X-axis.

A

Support and Service

Calculator Support

You can obtain answers to questions about using your application card from our Calculator Support department. Our experience has shown that many customers have similar questions about our products, so we have provided the following section, "Answers to Common Questions." Also, the HP 48 owner's manual includes similar information about the calculator itself. If you don't find the answer to your question below or in Appendix A of the HP 48 owner's manual, contact us at the address or phone number on inside back cover.

Answers to Common Questions

Q: *I'm not sure whether the application card is malfunctioning or if I'm doing something incorrectly. How can I verify that the card and the calculator are operating properly?*
A: You can check the application card by turning on the calculator and pressing [⬅] [LIBRARY]. The calculator checks application cards whenever it turns on. If the message `Invalid Card Data` or `Port Not Available` is displayed at turn-on, the card requires service. If the LIBRARY menu doesn't include the application names shown in Chapter 1, the card may require service. Check that you have installed the card properly. To check the calculator, see Appendix A of the HP 48 owner's manual.

Q: *The calculator beeps and displays a message I do not recognize. How do I find out what's wrong?*
A: See appendix E in this manual for application card messages. See the HP 48 owner's manual for calculator messages that occur while using the application card.

Q: *Can I remove the Emulator Card after I have captured my HP 41 programs and data?*
A: If you take out the Emulator Card, your HP 41 data and programs will stay in your HP 48 but you can only use them when the Emulator Card is plugged into your HP 48.

Q: *When I run an HP 48 program that includes Emulator commands I get the error message* `Undefined XLIB Name` *– why is this?*
A: You have removed the Emulator card, or it has come loose in the calculator. Plug the card back in.

Q: *Capture operations are not working the way they are described in Chapter 3. What can be wrong?*
A: See the list of possible problems at the end of Chapter 3. If your program contains non-standard program steps (also known as *Synthetic Instructions*), then the Emulator might not be able to translate them. Rewrite your program or capture it on the HP 48 using INPRT, then edit it to use other instructions. See Appendix D for more information.

Q: *I am using an HP 41 that has been modified to run faster than normal. I thought this should speed up capture but instead capture is not working correctly. What can I do?*
A: The Infrared Printer module cannot keep up with a "faster than normal" HP 41. Slow down your HP 41 to normal speed. Otherwise, copy the programs to an HP 41 that runs at normal speed and use that for the capture.

Q: *I can't find some HP 41 information, programs, or XROM commands that I used earlier. Where did they go?*
A: You may have been using them in a different directory, or on a directory that is not on the current path. Make sure you are in the right directory and try again.

Q: *I have captured and translated a program, but it stops during execution and displays* `Nonexistent`. *What is happening?*
A: If the program has come from an HP 41CV, this may be an ordinary HP 41 NONEXISTENT error. The program is trying to use a register, a flag, or a label that does not exist. Make sure the SIZE setting is sufficient. Otherwise, hold down $\boxed{\text{SST}}$ to see the step causing the error message. If the step shows up as XEQ or XROM followed by a name or a number, then your program is trying to execute a command that it cannot find in any of the catalogs. You may be in

the wrong directory, so the Emulator cannot find the program or function. Or maybe your HP 41 program is trying to execute a command or function that you have not transferred to the HP 48. Identify the missing command and copy it to the HP 48 or provide a replacement using the methods described in Chapter 5. If the missing command is built into a peripheral device (such as a Card Reader) see Appendix D for suggestions.

Q: *I have typed in an HP 41 program as a text string on the HP 48 but it does not enter the HP 48 stack correctly, nor can it be translated. What should I do?*
A: If you type an HP 41 program on the HP 48 you have to be careful with your spelling and with quote marks. Check to see if you have spelled all commands correctly and have included spaces where they are needed. Use apostrophes (') instead of quotes (").

Q: *I have captured an HP 41 style program from an HP 41CX, an HP 42S, or some other source that is not an HP 41C or HP 41CV. Now I am having trouble running the program; it displays* `Nonexistent` *or some other error message. What should I do?*
A: The program may be trying to use memory that is not available by default to the Emulator. Check the bad program step, and increase SIZE if necessary. Problems can also arise because other calculators use commands that the Emulator does not provide. Use [SST] to identify the command that is causing the trouble.

Q: *I have captured a program that has numbers like E6 in it. The Emulator has translated these numbers into commands such as XEQ 'E6'. What are these numbers, and what should I do with them?*
A: Such numbers are allowed by the HP 41 for compatibility with the earlier HP 67 and HP 97 calculators. See the section on the Card Reader in Appendix D for details and advice.

Q: *Sometimes my HP 48 seems to pause for a few seconds during an operation. Is anything wrong?*
A: Nothing is wrong. The calculator does some system cleanup from time to time to eliminate temporary objects created during normal operations. This cleanup process frees memory for current operations, but it pauses the calculator. If it occurs during a capture operation, you will have to repeat the capture operation.

Environmental Limits

To maintain the reliability of HP 48 plug-in cards, observe the following temperature and humidity limits:

- Operating temperature: 0° to 45°C (32° to 113°F).
- Storage temperature: -20° to 60°C (-4° to 140°F).
- Operating and storage humidity: 90% relative humidity at 40°C (104°F) maximum.

Limited One-Year Warranty

What Is Covered. The card is warranted by Hewlett-Packard against defects in materials and workmanship for one year from the date of original purchase. If you sell your unit or give it as a gift, the warranty is automatically transferred to the new owner and remains in effect for the original one-year period. During the warranty period, we will repair or, at our option, replace at no charge a product that proves to be defective, provided you return the product, shipping prepaid, to a Hewlett-Packard service center. (Replacement may be made with a newer model of equal or better functionality.)

This warranty gives you specific legal rights, and you may also have other rights that vary from state to state, province to province, or country to country.

What Is Not Covered. This warranty does not apply if the product has been damaged by accident or misuse or as the result of service or modification by other than an authorized Hewlett-Packard service center.

No other express warranty is given. The repair or replacement of a product is your exclusive remedy. **ANY OTHER IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS IS LIMITED TO THE ONE-YEAR DURATION OF THIS WRITTEN WARRANTY.** Some states, provinces, or countries do not allow limitation on how long an implied warranty lasts, so the above limitation may not apply to you. **IN NO EVENT SHALL HEWLETT-PACKARD COMPANY BE LIABLE FOR CONSEQUENTIAL DAMAGES.** Some states, provinces, or countries do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

Products are sold on the basis of specifications applicable at the time of manufacture. Hewlett-Packard shall have no obligation to modify or update products, once sold.

Consumer Transactions in the United Kingdom. This warranty shall not apply to consumer transactions and shall not affect the statutory rights of a consumer. In relation to such transactions, the rights and obligations of Seller and Buyer shall be determined by statute.

If the Card Requires Service

Hewlett-Packard maintains service centers in many countries. These centers will repair a card, or replace it with the same model or one of equal or better functionality, whether it is under warranty or not. There is a service charge for service after the warranty period. Calculators and accessories normally are serviced and reshipped within 5 working days.

- **In the United States:** Send the card to the Corvallis Service Center listed on the inside of the back cover.
- **In Europe:** Contact you Hewlett-Packard sales office or dealer, or Hewlett-Packard's European headquarters (address below) for the location of the nearest service center. Do not ship the card for service without first contacting a Hewlett-Packard office.

  Hewlett-Packard S.A.
  150, Route du Nant-d' Avril
  P.O. Box CH 1217 Meyrin 2
  Geneva, Switzerland
  Telephone: 022-780.81.11

- **In other countries:** Contact your Hewlett-Packard sales office or dealer or write to the Corvallis Service Center (listed above) for the location of other service centers. If local service is unavailable, you can ship the card to the Corvallis Service center for repair.

All shipping, reimportation arrangements, and customs costs are your responsibility.

Service Charge. Contact the Corvallis Service Center for the standard out-of-warranty repair charges. This charge is subject to the customer's local sales or value-added tax wherever applicable.

Calculator products damaged by accident or misuse are not covered by the fixed charges. These charges are individually determined based on time and material.

Shipping Instructions. If your card requires service, ship it to the nearest authorized service center or collection point.

- Include your return address and a description of the problem.
- Include proof of purchase date if the warranty has not expired.
- Include a purchase order, check, or credit card number plus expiration date (VISA or MasterCard) to cover the standard repair charge.
- Ship your card postage prepaid in adequate protective packaging to prevent damage. Shipping damage is not covered by the warranty, so we recommend that you insure the shipment.

Warranty on Service. Service is warranted against defects in materials and workmanship for 90 days from the date of service.

Library Numbers. The Emulator consists of four HP 48 library objects, numbered 1402, 1403, 1404 and 2046. Library number 1405 is reserved for extension of the Emulator. Do not use other unrelated libraries with any of the above numbers while you are using the Emulator. You can see the library numbers by pressing [↰] [LIBRARY] and then pressing the menu key that corresponds to the port that the Emulator card resides in.

If you remove the Emulator Card, then Emulator command names will be replaced by `XLIB` *xxxx nnn*. Here *xxxx* is the library number, one of the four numbers above, and *nnn* is the command number.

B

Bibliography

You can find detailed information about the HP 41, the HP 48, their peripherals, and their uses in the manuals, books, and other sources listed in this Appendix. The manuals that accompany each product are the primary sources of information for the HP 41 and its peripherals.

HP 41 General Reference

Extend Your HP-41 by W.A.C. Mier-Jedrzejowicz, Ph.D. (1985). Published by SYNTHETIX, c/o Interfab, 27955 Cabot Road, Laguna Niguel, CA 92677, USA.

This is probably the most complete source of HP 41 information (particularly useful if you do not have all the HP 41 manuals). This book explains HP 41 keyboard operations, programming methods, and information on peripherals and plug-in modules. The Time Functions, Extended Functions and Extended Memory, and the additional HP 41CX functions are covered in particular detail. Advanced programming techniques are dealt with in the last chapters. One of the appendices provides a detailed list of how system flags are used by the HP 41 and peripherals (important information if you are transferring programs that use system flags from the HP 41 to the HP 48).

The HP-41 Synthetic Quick Reference Guide by Jeremy Smith (1983). Published by CodeSmith, 301 NE Byron Place, Corvallis, OR 97330, USA.

The QRG is a pocket-sized collection of reference information. This book shows how HP 41 functions are organized in memory, how main memory and Extended Memory are arranged, how flags are used, and much more. Of particular interest to users transferring programs to the

HP 48 is the extensive listing of XROM functions by module number and command number in the module.

Specific HP 41 Topics

Synthetic Programming on the HP-41C by William C. Wickes, Ph.D. (1980). Published by Larken Publications, 4517 NW Queen's Avenue, Corvallis, Oregon 97330, USA.

This is the original book about the branch of advanced programming called Synthetic Programming. This book explains how HP 41 programs work and how program instructions are laid out. The information might be useful if you are transferring complicated programs from the HP 41 to the Emulator.

HP-41 Extended Functions Made Easy by K. Jarett, Ph.D. (1983). Published by SYNTHETIX, c/o Interfab, 27955 Cabot Road, Laguna Niguel, CA 92677, USA.

This book explains the Extended Functions and Extended Memory in more detail than provided by the HP manual.

HP-41 Synthetic Programming Made Easy by K. Jarett, Ph.D. (1982). Published by SYNTHETIX, c/o Interfab, 27955 Cabot Road, Laguna Niguel, CA 92677, USA.

This book begins at a simpler level and provides additional information not available in *Synthetic Programming on the HP-41C*.

HP 48 Information

The HP 48 Handbook by James Donnelly (1990). Published by Armstrong Publishing Company, 3135 NW Ashwood Drive, Corvallis, Oregon, USA.

This book of reference information for HP 48 users is much smaller than the HP 48 manuals, fitting into a large pocket. HP 48 information, including items not in the manuals, is provided in a compressed form with many tables.

Armstrong Publishing Company also produces the *HP 48 Programmer's ToolKit*, a disk of additional HP 48 commands with a manual explaining how the Toolkit commands can be used to extend the HP 48. The commands comprise a library called **TLLIB** (as shown in the picture of the display in *Entering the Environment*, Chapter 2).

HP 41/HP 48 Transitions William C. Wickes, Ph.D. (1990). Published by Larken Publications, 4517 NW Queen's Avenue, Corvallis, Oregon 97330, USA.

This is a detailed guide for users transferring HP 41 programs to the HP 48 by rewriting them in the HP 48 programming language. This book provides an alternative to capturing an HP 41 program in its original form and running it by using the functions of the Emulator card. In practice, you will probably find that a combination of the two approaches is best. Some programs can be translated easily, while others are best executed using the Emulator (at least while you are still learning to use the HP 48). A table shows how HP 41 functions and commands, including HP 41CX functions, can be carried out on the HP 48. Once you have become experienced with the HP 48, this book should prove an invaluable guide in rewriting complicated HP 41 programs so they will run in the HP 48 language. The author of this book also wrote *Synthetic Programming on the HP-41C* and was the software development team leader for the HP 48, so the book is written from a standpoint of great experience with both calculators.

Another book that might be useful is the *HP-15C Advanced Functions Handbook*. This book includes sections relevant to all calculator work, especially the Appendix on the accuracy of numerical calculations. The HP 41 and the HP 48 carry out internal calculations in the same way as the HP-15C, so the contents of this book apply to calculations and operations. (HP part number 00015-90011)

Other Sources of Information

You can find information from sources other than books. Some are listed here.

An HP Calculator Bulletin Board System is available for the exchange of software and information among HP calculator users, developers, and distributors. The bulletin board operates at 300/1200/2400 baud, full duplex, no parity, 8 bits, 1 stop bit. The telephone number is (503) 750-4448. The bulletin board is a free service (you pay only for the long-distance telephone charge). Users outside the United States can also call this bulletin board, but HP acknowledges that this may be prohibitively expensive and is searching for ways to make the same information available in other countries.

An English language journal is published eight times a year by HPCC, Geggs Lodge, Hempton Road, Deddington, Oxford OX5 4QG, United Kingdom. Membership in HPCC, and its journal, are available to people outside the UK.

A library of HP 41 programs is available from Solve and Integrate Corporation, P.O.Box 1928, Corvallis OR 97330 USA. If you need an HP 48 program to do a specific job in a hurry, but cannot write a suitable program yourself, check if Solve and Integrate can provide an HP 41 program to do the job, then run it on the HP 48 using the HP 41 Emulator.

C

Tables of Flags and XROM Identifiers

Flags

The HP 41 flags provided by the Emulator can be classed in four groups. The groups are described below.

The HP 41 initializes the values of certain flags when it is turned on. The Emulator initializes these flags when the HP 48 is turned off while you are in the Emulator environment.

User Flags (00 through 10). These flags are not designated for any specific operations by the HP 41 nor by the Emulator. In general they are available for any purpose the user chooses. (Be careful if you use commands from the HP 41 Advantage module, as they use all these flags.) These flags are clear when the HP 41 is initialized. Their status is maintained when you leave the Emulator and re-enter it, and when you turn the HP 48 off.

Control Flags (11 through 29). These flags control specific HP 41 operations. See an HP 41 manual for a description, or look in one of the HP 41 reference books listed in Appendix B for details. Flags 11 through 13, 15, 16, and 21 through 29 work as on the HP 41. The status of these flags is maintained if you exit the HP 41 Emulator and then turn off the HP 48.

If flag 11 is set then the HP 41 Emulator will start running the current program from the current step when you next turn on the HP 48 in Emulator mode. If you exit from the HP 41 Emulator and then enter it again, flag 11 will be cleared and the current program will not be restarted. Flag 14 which is used by the Card Reader, and flags 17 and

18 which are used by HP-IL operations, are treated as general purpose flags by the Emulator.

System flags (30 through 55). The system flags are used mainly for operations internal to the HP 41. You cannot directly set or clear them with flag commands, but you can test them. Some can be set and cleared by HP 41 commands or by commands in plug-in modules. The uses made by plug-in modules are not provided by the Emulator, for example flag 31 does not specify whether date format is set to MDY or DMY (The HP 48 flag -42 is used for this purpose by HP 48 date commands.)

HP 41 manuals and manuals for plug-in devices describe how these flags are used. For a full list of details in one place look in *Extend Your HP 41* (see Appendix B). Flags 36-43, 47, 48, 50, 51, and 55 are used as on the HP 41.

If you have a program which uses HP 41 system flags to modify the behavior of programs or to test certain actions then you should check whether the Environment provides the corresponding option, for example there is no permanent ON option (HP 41 flag 44). Users of older HP 41 printers should note that flag 55 can be set or cleared by the commands PON and POFF, to enable or disable printer operations.

Emulator flags (56 through 87). These flags do not exist on the HP 41. The Emulator uses them for its own additional features, as listed below:

Flag No.	Meaning when set	Comments
56	Stack lift enabled	
57	Executing an XROM outside the environment	
58	An HP 41 program is running	
59	Update L register display	
60	Update X register display	
61	Update Y register display	
62	Update Z register display	
63	Update T register display	
64	Unused	
65	Program is Private	
66	Unused	
67	Update annunciator display	
68	Update message display area	
69	Parsing in progress	
70	Negative mantissa sign entered	
71	Exponent entered	
72	Radix mark (point or comma) entered	
73	Negative exponent sign entered	
74	Executing interactive Emulator environment	
75	Program step being printed is a label	
76	Update Alpha register display	
77	Print in Map On mode	
78	Print in Underline On mode	
79	Update menu key label display	
80	Print in right justified mode	
81	Print in centered mode (set when FMT command is used with one string)	
82	Print in right-left split mode (set when FMT is used with two strings)	
83	Escape sequence for double width mode has been sent to the printer	
84	Previous character sent to printer was ESC	
85	Translate special characters in →41 and CPTR	*You can set and clear these 3 flags*
86	Full stack display is enabled	
87	HP 41 Environment exiting	

Flags 26, 28, 29, 37, 40, 77, 85, and 86 are set when the HP 41 Environment is first initialized, all the other Emulator flags are clear.

Many of these flags are used internally by the Emulator and will always be clear when you test them. For example, all the "update" flags are used during internal operations to tell the Emulator that it must update the display before it completes the current command and lets you use the keyboard again.

XROM Identifiers

Every function and command in CAT 2 is identified by an XROM number of the form *mm,nn*. The module or peripheral which provides the command is defined by *mm,* and *nn* is the command number in that module. The table that begin on this page tells you which modules or peripherals provide which XROM numbers.

In some cases more than one module or peripheral use the same XROM identifier.

Some modules and peripherals come in two parts and use two XROM numbers.

All modules designed by Hewlett-Packard and other generally available modules are listed. If you cannot find a module in this list then it may be a "custom" module made for a specific organization. Two identifiers were assigned for use with "custom" modules, and are marked in the list.

XROM number	Modules or peripherals which used this XROM number
1	Math, lower part of Math/Stat module. Note that some function numbers in revisions 1C and 1D differ from those in 1A and 1B
2	Statistics module, upper part of Math/Stat module.
3	Surveying
4	Finance, CMT 200
5	Standard Pac, ZENROM, Paname lower part
6	Circuit analysis, TOMS ROM 1 Survey instrument module
7	Structural analysis lower part, SKWIDBC and SKWIDBC+ Barcode modules, HEPAX module
8	Stress analysis

9	Home Management, Surveyor's Co-op module lower part, Paname upper part, CCD module lower part
10	Autostart/Duplication module, Games, Market Forecaster, Surveyor's Co-op module upper part, PPC ROM lower part
11	Real Estate, CCD module upper part
12	Machine Design
13	Thermal and Transport Science
14	Navigation
15	Petroleum Fluids lower part
16	Petroleum Fluids upper part
17	Plotter module lower part
18	Plotter module upper part, AEC ROM
19	Aviation, Clinical Lab & Nuclear Medicine, Securities, Structural upper part. See Note 1 below
20	PPC ROM upper part
21	Data logger lower part, Daly Oilwell module lower part, US Postal ROM lower part, SUP-R-ROM Survey module lower part, RaceTrack module, two-part custom modules
22	HP-IL Development module lower part, Advantage lower part
23	Extended I/O
24	HP-IL Development module upper part, Advantage upper part
25	Extended Functions, including HP 41CX Extended Functions
26	Time module, including HP 41CX Time functions
27	Wand, Extended IL ROM
28	HP-IL Mass Storage and Control commands
29	Printer, HP-IL printer commands and Infrared printer module
30	Card Reader
31	Data logger upper part, Daly Oilwell module upper part, US Postal ROM upper part, SUP-R-ROM Survey module upper part, Astro*ROM, Handy Compact (antenna structure), one-part and two-part custom modules

Notes.

1. All the modules made by HP with the XROM identifier 19 had an X on their label, warning you to avoid using two of them at the same time.
2. You should never plug two modules with the same XROM identifier into an HP 41 at the same time. This can lead to unpredictable results.
3. The Daly Oilwell module, the US Postal ROM and the SUP-R-ROM are listed here as examples of "custom" modules.
4. The words ROM and module mean the same thing in the above list. Most Hewlett-Packard modules are called Application Pacs.
5. A way to deal with modules that have the same XROM identifiers is to have two separate XROM libraries. You could give them different names, or put them on different paths.

Chapter 5 explains how you can determine the XROM identifier and command number of most CAT 2 commands. The method does not work for CAT 2 functions in the HP 41CX. For this reason, the CAT 2 functions of the HP 41CX are listed on the next page.

As was noted in Chapter 5, different revisions of the Mathematics module have their commands in two different arrangements. All the commands which have different numbers are marked with an asterisk in the table.

EXTENDED FUNCTIONS

XROM number and command number	Command name	XROM number and command number	Command name
25,00	EXT FCN 2D	25,32	RCLFLAG
25,01	ALENG	25,33	RCLPT
25,02	ANUM	25,34	RCLPTA
25,03	APPCHR	25,35	REGMOVE
25,04	APPREC	25,36	REGSWAP
25,05	ARCLREC	25,37	SAVEAS
25,06	AROT	25,38	SAVEP
25,07	ATOX	25,39	SAVER
25,08	CLFL	25,40	SAVERX
25,09	CLKEYS	25,41	SAVEX
25,10	CRFLAS	25,42	SEEKPT
25,11	CRFLD	25,43	SEEKPTA
25,12	DELCHR	25,44	SIZE?
25,13	DELREC	25,45	STOFLAG
25,14	EMDIR	25,46	X<>F
25,15	FLSIZE	25,47	XTOA
25,16	GETAS	**HP 41CX only:**	
25,17	GETKEY	25,48	-CX EXT FCN
25,18	GETP	25,49	ASROOM
25,19	GETR	25,50	CLRGX
25,20	GETREC	25,51	ED
25,21	GETRX	25,52	EMDIRX
25,22	GETSUB	25,53	EMROOM
25,23	GETX	25,54	GETKEYX
25,24	INSCHR	25,55	RESZFL
25,25	INSREC	25,56	ΣREG?
25,26	PASN	25,57	X=NN?
25,27	PCLPS	25,58	X≠NN?
25,28	POSA	25,59	X<NN?
25,29	POSFL	25,60	X<=NN?
25,30	PSIZE	25,61	X>NN?
25,31	PURFL	25,62	X>=NN?

TIME FUNCTIONS

XROM number and command number	Command name	XROM number and command number	Command name
26,00	-TIME 2C	26,19	RCLSW
26,01	ADATE	26,20	RUNSW
26,02	ALMCAT	26,21	SETAF
26,03	ALMNOW	26,22	SETDATE
26,04	ATIME	26,23	SETIME
26,05	ATIME24	26,24	SETSW
26,06	CLK12	26,25	STOPSW
26,07	CLK24	26,26	SW
26,08	CLKT	26,27	T+X
26,09	CLKTD	26,28	TIME
26,10	CLOCK	26,29	XYZALM
26,11	CORRECT	**HP 41CX only:**	
26,12	DATE	26,30	-CX TIME
26,13	DATE+	26,31	CLALMA
26,14	DDAYS	26,32	CLALMX
26,15	DMY	26,33	CLRALMS
26,16	DOW	26,34	RCLALM
26,17	MDY	26,35	SWPT
26,18	RCLAF		

THE TWO TYPES OF MATH PAC ORDERING

XROM number and command number	Command name	XROM number and command number	Command name
01,00	MATH 1A/1B	01,00	-MATH 1C/1D
01,01	"MATRIX"	01,01	"MATRIX"
01,02	"SIMEQ"	01,02	"SIMEQ"
01,03	"VCOL"	01,03	"VCOL"
01,04	"VMAT"	01,04	"VMAT"
01,05	"PVT"	01,05	"PVT"
01,06	"DET"	01,06	"DET"
01,07	"INV"	01,07	"INV"
01,08	"EDIT"	01,08	"EDIT"
01,09	"SOLVE"	01,09	"SOLVE"
01,10	"SOL"	01,10	"SOL"
01,11	"POLY"	01,11	"POLY"
01,12	"ROOTS"	01,12	"ROOTS"

01,13	"INTG"	01,13	"INTG"
01,14	"DIFEQ"	01,14	"DIFEQ"
01,15	"FOUR"	01,15	"FOUR"
*01,16	"C+"	01,16	"Z^N"
*01,17	"C-"	01,17	"MAGZ"
*01,18	"C*"	01,18	"e^Z"
*01,19	"C/"	01,19	"LNZ"
*01,20	"MAGZ"	01,20	"Z^1/N"
*01,21	"CINV"	01,21	"SINZ"
*01,22	"Z^N"	01,22	"COSZ"
*01,23	"Z^1/N"	01,23	"TANZ"
*01,24	"e^Z"	01,24	"a^Z"
*01,25	"LNZ"	01,25	"LOGZ"
*01,26	"a^Z"	01,26	"Z^1/W"
*01,27	"LOGZ"	01,27	"Z^W"
*01,28	"Z^W"	01,28	"C+"
*01,29	"Z^1/W"	01,29	"C-"
*01,30	"SINZ"	01,30	"CINV"
*01,31	"COSZ"	01,31	"C*"
*01,32	"TANZ"	01,32	"C/"
01,33	"SINH"	01,33	"SINH"
01,34	"COSH"	01,34	"COSH"
01,35	"TANH"	01,35	"TANH"
01,36	"ASINH"	01,36	"ASINH"
*01,37	"ACOSH"	01,37	"ATANH"
*01,38	"ATANH"	01,38	"ACOSH"
01,39	"SSS"	01,39	"SSS"
*01,40	"ASA"	01,40	"SAA"
*01,41	"SAA"	01,41	"ASA"
01,42	"SAS"	01,42	"SAS"
01,43	"SSA"	01,43	"SSA"
01,44	"TRANS"	01,44	"TRANS"
01,45	"*FN"	01,45	"*FN"

* revisions A and B differ from revisions C and D.

D

Advanced Applications

Some users who have exploited special features of the HP 41 system may need further information. This appendix provides details on two advanced topics. The first is the way HP 41 programs are translated by the HP 48 Emulator. The second is the use of HP 41 accessories other than the printer module and other HP plug-in modules.

Translation, Compilation, and Decompilation

When a text string is passed to the function →41 for translation to an HP 41 program, several operations are carried out. First of all, some characters are replaced; for example double quote symbols (") are replaced by apostrophes ('). If a time and date are found at the beginning, they are discarded. Then the string is parsed. This means that the string is split up into one function at a time, with their step numbers ignored. Then each function is translated into one or more bytes.

The byte which represents a function is called that function's token, so translation of a program from a text string into a series of bytes is called tokenization. Each program step is parsed and tokenized, until the whole program has been translated.

When the whole text string has been translated, it is compiled. This means that each local GTO and XEQ function is matched up with the corresponding label, and the distance (or "offset") to that label is stored with the GTO or XEQ. Thus, when the program is executed there is no need for each GTO or XEQ to search for a corresponding label. On the HP 41 this compilation is carried out while a program is running, but on the Emulator it is more convenient to carry out the compilation with the translation.

If you want to see the original program, you can translate it back into a text string in the original User Language by using the →TXT command. The process is called decompilation. (The name detokenization would be more accurate, but is more difficult to pronounce.) The HP 48 command →TXT decompiles an HP 41 program object into Focal (Forty-one calculator language). The same decompilation process is carried out when you view a program with SST and BST or when you print it with PRP or LIST.

Other Details of Implementation

Flags
The Emulator provides all the flags 0 through 55, and its own additional flags. See the flag table and notes in Appendix C for details.

Multiple HP 41s
Note that you can have more than one HP41PAR and more than one set of HP 41 parameter variables, as long as each set is in a different HP 48 directory. The set currently in use is the one that is first found in the standard search used when the HP 48 looks for a variable in the current path. This lets you have more than one emulated HP 41 inside your HP 48. Each one can have its own flag settings, XROMs, User keys, and program libraries.

The HP 41 programs accessible at a given time (the *currently* accessible programs) are those in the current directory and those in the PRG41 directory .

In order to speed up operations, the Emulator keeps a temporary copy of the current HP 41 environment. This means that changes made to the environment are not necessarily reflected in HP41PAR at once, though they are whenever you exit from the HP 41 Environment.

Character Transliteration

Two Emulator operations require particular care with special characters. One is the translation of programs, during capture or with →41. The other is execution of HP 48 names from inside the HP 41 Environment.

Special characters produced by the HP 41 will be displayed in the HP 48 as you would expect them on the HP 41, and will be printed using the emulator's print functions correctly on the HP82240B printer. For this to happen, some HP 41 characters have to be transliterated; the character code in the HP 41 program is replaced by a different number. This different number represents the HP 48 character which looks like the original HP 41 character.

Note that the right arrow character → is displayed rather than the HP 41 append character, since the HP 48 does not have the append character in its display character set. With the default settings, the Emulator prints this character as the HP 41 append symbol, since the HP82240B printer can print this character.

If you do not wish to have characters transliterated, you can disable all the transliteration by putting both the infrared printer module and the Emulator in MAPOFF mode and clearing flag 85. The emulator will still work, but some special characters will not display or print as you might expect.

Emulating HP 41 Peripherals

Programs that use the HP 41 as a calculator can be transferred to the HP 48 without too much trouble. On the other hand, programs which use the HP 41 primarily to control other devices are much more difficult to transfer to the HP 48. When the HP 41 was introduced, there was no standard for desktop computers, and it was reasonable to build up a collection of peripherals for the HP 41, making it a portable computing system. HP-IL allowed the HP 41 to be used as a portable instrument controller and data collector. With the acceptance of common standards for desktop computers it was considered that the HP 48 should use a desktop computer for storage, and that desktop computers would replace portable handheld computers as instrument controllers. This means that the HP 48 can not be used to emulate all the ways in which the HP 41 could be used as a portable computer system. The following sections discuss the individual HP 41 peripherals and the extent to which each can be emulated with the HP 48.

Printers

The HP82162A can print HP 41 barcode and can be set up to print text so that any word which would go beyond the right margin is printed on a new line. Neither of these features is available from the HP 48. If you really need to print HP 41 barcode from an HP 48, then connect the HP 48 with an RS232 to HP-IL interface and then use HP-IL commands to print the barcode on an HP82162A printer. See the section below on HP-IL. If you want to justify printed text, send it to a desktop computer and use a word processor to print the text.

Both the HP82143A and the HP82162A used slightly different character sets than the infrared printer, the HP82240 (A and B).

If you need one of the missing symbols, or if you want to create your own symbols you can do so by using BLDSPEC or the graphics features of the HP82240.

The HP82143A and HP82162A printers can inform the HP 41 when they had problems, such as running out of paper. HP 41 programs on the Emulator that check for such problems will not be able to detect them.

The larger printers, the dot-matrix HP82905B and the HP2225A and B ThinkJet printers use wider carriages and different graphics commands than the smaller printers. The best way to emulate them is to send print commands to a printer attached to the HP 48 via RS232.

Card Reader

The HP 41 Card Reader allows users to save copies of programs, data, and HP 41 status information. These copies are made on low-capacity magnetic cards. The HP 48 does not provide a similar method of storing and exchanging small amounts of data. RAM cards can be used to store data and programs on a removable medium. Programs and data can also be copied to a floppy disk on a desktop computer, or use a portable disk drive which can be controlled from the HP 48.

The Card Reader also provides functions to read in subroutines when a program needs them, and to merge a running program with another read off a card. On the HP 48 there is may be enough memory to hold all programs at once.

The Card Reader can record the complete contents of an HP 41 on a set of cards. This function, WALL, can be replaced by the HP 48 operations used to make a complete record of the HP 48, including the HP 41 environment information, on a RAM card.

Card Readers can read HP-67 and HP-97 program and data cards and translate them for use on an HP 41. If you want to run an HP-67 or HP-97 program in the HP 41 Emulator, you must first translate it into an HP 41 program and then capture that on the HP 48.

Some HP-67 and HP-97 commands are not provided by the HP 41, and the Card Reader provides XROM functions to perform these commands. A few users have included these functions in HP 41 programs too. The Emulator does not provide any of these functions, but they can all be emulated fairly easily.

The HP-67 and HP-97 calculators allow you to enter numbers such as 1E6 or 1E-19 without the leading 1. The HP 41 Card Reader and the HP 41 accepted such commands, so that HP 41 programs can have steps such as E6 or E-19. When the Emulator captures a program it does not know if a step such as E6 is the number 1E6 or a CAT 2 function called E6. The Emulator therefore treats it as an unknown command. It will translate the above two examples as XEQ 'E6' and XEQ 'E-19'. You should edit the captured program and replace these commands with 1E6 and 1E-19 respectively if they are intended to be numbers.

Wand

The HP 41 Wand, or HP82153A Optical Barcode Reader, lets the HP 41 read printed barcodes. The HP 41 Wand can only read a special HP 41 form of barcode, but it is nevertheless a very useful input device for some applications. No equivalent is available for the HP 48, but it would be possible to adapt barcode wands designed for use via RS232. If you do use such a wand with the HP 48 you will have to write HP 48 programs to control the wand, and you will either have to use industry standard barcodes or rewrite the wand control software to read HP 41 barcode.

Cassette Drive and Disk Drive

HP 41 Mass Storage devices are the HP82161A Cassette Drive and the HP9114A and B Disk Drives. They use the same commands, and their use can be extended with commands from the Extended I/O module. Since the HP 48 has more internal memory than the HP 41, it may be sufficient to use HP 48 memory to replace HP 41 mass storage and recall operations. For example, the function WRTR writes all HP 41 storage registers to a cassette or disk. On the HP 48, an HP 41 program could execute an HP 48 program which would use RCL41 to copy all the HP 41 registers to the HP 48 stack.

If necessary, the program could then write the registers to a portable disk drive or a disk drive on a desktop computer. Other Mass Storage reading and writing commands can be rewritten as HP 48 programs in the same way.

Securing, renaming and purging files can be done via Kermit. The two Mass Storage commands in the Extended Functions module, GETAS and SAVEAS are more difficult to implement since they work on text files and the HP 48 does not deal with text files.

Other HP-IL devices

HP-IL provides access to many other devices. Some are specifically designed for use with HP-IL and a portable computer, for example, the HP7470 plotter, Option C. Other devices can be controlled from HP-IL through HP-IL interface units (see below). In general it is possible to write HP 48 programs which would control the same devices, or RS232 versions of them, via Kermit or by direct RS232 commands. Although, it may be simpler to rewrite the whole application to run on a desktop computer.

One feature of HP-IL which can not easily be emulated on a desktop computer is control of many devices at one time. The HP-IL loop allows up to 31 devices to be connected to an HP 41. Rewiring all these so that a portable personal computer could control them without HP-IL may be a considerable problem.

Details of modifying HP-IL systems are beyond the scope of this manual. Some of the books in Appendix B may be of help, otherwise the best approach is to find another user who has relevant experience. Your local HP office may be able to help, or a local user group should be able to find a member who can help. The user groups maintain international contacts, so if no one nearby can help they might find an expert further afield. Ask your local HP office for the addresses of the nearest user groups.

E

Error Messages and Other Messages

Messages produced by the Emulator are listed here alphabetically. The list is followed by descriptions of messages issued during translation and configuration.

Each message in the list is shown as:

`Text of message`
Explanation of message.
Message number (in hexadecimal)

`Alpha Data`
An alpha string has been entered as the argument to a function which expects numeric input.
# 57C01

`Bad GTO/XEQ Arg`
In translating a program line, the Emulator has found that the argument to a GTO or XEQ command is invalid.
# 57C0E

`Bad LBL Arg`
In translating a program line, the Emulator has found that the argument following a LBL command is invalid.
# 57C0F

`Bad Numeric Arg`
In translating a program line, the Emulator has found that the argument passed to FIX, SCI, ENG, TONE or a flag operation is invalid.
# 57C10

`Bad Opcode`
This indicates that the Emulator has detected an HP 41 *opcode* which it does not recognize. *Opcode* is another name for a token. This error should not occur in normal use of the Emulator, because the HP 41 Environment and program translation do not allow bad opcodes to be entered. It means that a program has been corrupted in memory, or that a bad opcode has been entered by use of the HP 48 memory editor. It can also occur in a program read into the HP 48 that has already been translated, if that program has been altered.
# 57C0D

`Bad Reg Op Arg`
During program translation the argument to RCL, STO, or other register command in a program line is not recognizable as numeric, indirect, or a stack register. If this error is encountered, the comment

```
!Bad Reg Op Arg!
```

is inserted into the translated program. The register command is ignored, and the Emulator tries to translate the argument on its own.
# 57C11

`Bad XROM Numbers`
When attempting to translate a program line of the form "XROM *mm,nn*", the Emulator detected an error in the *mm,nn* portion (*mm* >31 or *nn*>63).
# 57C12

`Data Error`
An argument has been given which is outside the function's domain. Also used to indicate a string has been passed to 48XQ which the HP48 is unable to parse.
# 57C02

`Errors Detected; Retry`
When the Emulator was attempting to receive infrared data in CPTR or INPRT, either no data was received or errors were detected in the data received. As the message suggests, you should try the capture again.
# 57C13

`Err: Re-entrant`
An attempt has been made to use the HP 41 Environment anew from within the HP 41 Environment.
# 57C06

`Invalid HP41PAR`
During an attempt to use the HP 41 Environment one of the following problems was found:
HP41PAR is not a list.
HP41PAR is a list but does not contain exactly 5 items.
An item in HP41PAR is not a variable name.
A variable name in HP41PAR is not bound to an appropriate object.
# 57C14

`No`
This message is not an error. It is used to display the result of a flag test or an X test, if that test is executed from the keyboard.
# 57C08

`No Room`
There is not enough free user memory available in the HP 48 to change SIZE. This error can occur when decreasing SIZE as well as when increasing SIZE, because at least one copy of both the old data register array and the new register array occur in memory at the same time during execution of SIZE. If it is okay to lose the data in the register array, it may be possible to get around this error and change SIZE to 100 by exiting from the HP 41 Environment and purging the data register array variable (default name HP41REGS). When you enter the Environment a default register array with 100 registers will be created.
# 57C0C

`Nonexistent`
Indicates that a desired data register, program label, flag, or XROM does not exist or is inappropriate for the function attempted. The message also used when 48TOX or 48TOA is invoked at a time when there is nothing in level 1 of the HP 48's stack. It can also indicate a missing XROM.
# 57C03

`Out of Range`
A numeric overflow has occurred. The result is replaced by the number ±9.99999999999E499.
# 57C04

`Parse Failed`
The input to FCN41 is unrecognizable or contains a numeric or alpha data entry, or an XROM function name.
# 57C15

`Private`
You are attempting to see into a private program.
# 57C09

`Prtoff Mode`
A printing function has been invoked from the keyboard in PRTOFF mode. You should set PRTON mode if you want to use a printing command.
# 57C05

`Yes`
This message is not an error. It is used to display the result of a flag test or an X test, if that test is executed from the keyboard.
# 57C07

# 57C0a is used by the ERR41 function with a user-specified error message.

# 57C0b is used by the MSG41 function with a user-specified message.

Capture and Translation

When the Emulator is capturing and translating programs and data it displays messages to inform you what it is doing.

`At PC=`*`aaaa`*` of `*`bbbb`*
This message is displayed after the "Compiling GTO/XEQs" message, to tell you the status of program compilation. "PC=*aaaa*" tells you that the *Program Counter* has reached byte address *aaaa* in the program. The message "of *bbbb*" tells you that the total length of the program is *bbbb* bytes.

`Compiling  GTO/XEQs`
The Emulator has completed translation of a program and is now matching up GTO and XEQ commands with the corresponding local labels.

`Line` *n*
The Emulator is translating a program and is currently at line *n*, where *n* is a number.

`Loaded R`*nn*
When register data is captured or translated, this message is displayed as each register number *nn* is loaded with its data.

`nnn  xxxx`
When an HP 41 program is being turned back into a text string by →TXT, each program step is displayed in turn, with the step number *nnn* followed by the step *xxxx*.

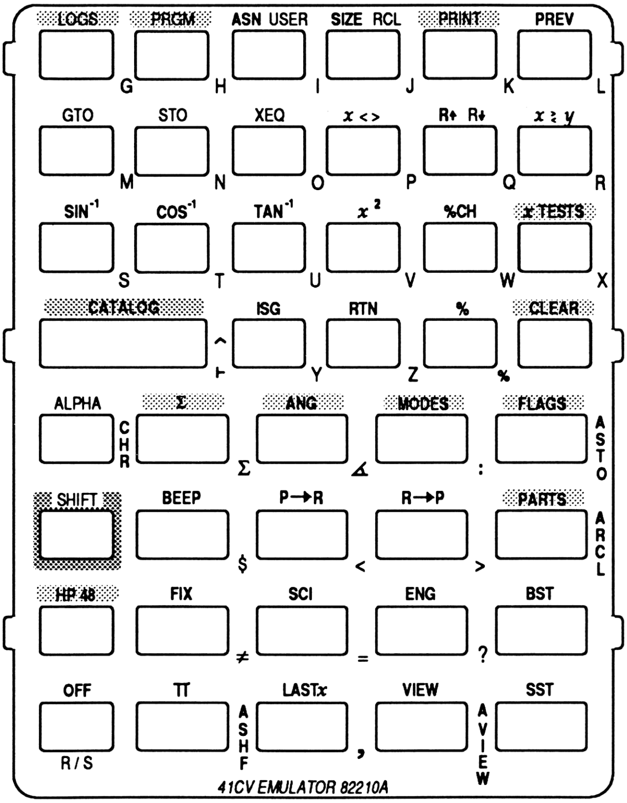
`Unknown` *xxxx*
When an unknown or mistyped command is found during translation, this message is displayed. "*xxxx*" is the unknown command. It is translated into XEQ '*xxxx*'.

Configuration

During configuration the Emulator displays messages if it needs to set up any new Emulator variables. The message displayed is:

xxxx `initialized`
xxxx is the variable being created or altered. It can be HP41PAR or one of the variables named in HP41PAR.

Index


LOGS
PRGM
ASN USER
SIZE RCL
PRINT
PREV
G
H
I
J
K
L
GTO
STO
XEQ
$x<>$
R↑ R↓
$x \gtrless y$
M
N
O
P
Q
R
SIN^{-1}
COS^{-1}
TAN^{-1}
x^2
%CH
x TESTS
S
T
U
V
W
X
CATALOG
ISG
RTN
%
CLEAR
^
⊢
Y
Z
%
ALPHA
CHR
Σ
ANG
MODES
FLAGS
Σ
∡
:
ASTO
SHIFT
BEEP
P→R
R→P
PARTS
$
<
>
ARCL
HP 48
FIX
SCI
ENG
BST
≠
=
?
OFF
π
LASTx
VIEW
SST
R / S
ASHF
,
AVIEW
41CV EMULATOR 82210A

Contacting Hewlett-Packard

For Information About Using the Calculator. If you have questions about how to use the calculator, first check the table of contents, the index, and "Answers to Common Questions" in the Appendix. If you can't find an answer in the manual, you can contact the Calculator Support department:

Hewlett-Packard
Calculator Support
1000 N.E. Circle Blvd.
Corvallis, OR 97330, U.S.A.

(503) 757-2004
8:00 a.m. to 3:00 p.m. Pacific time
Monday through Friday

For Service. If your calculator doesn't seem to work properly, refer to appendix A for diagnostic instructions and information on obtaining service. If you are in the United States and your calculator requires service, mail it to the Corvallis Service Center:

Hewlett-Packard
Corvallis Service Center
1030 N.E. Circle Blvd.
Corvallis, OR 97330, U.S.A.
(503) 757-2002

If you are outside the United States, refer to appendix A for information on locating the nearest service center.

Contents

Reorder Number
82210-90001
82210-90001
Printed in U.S.A. 10/90