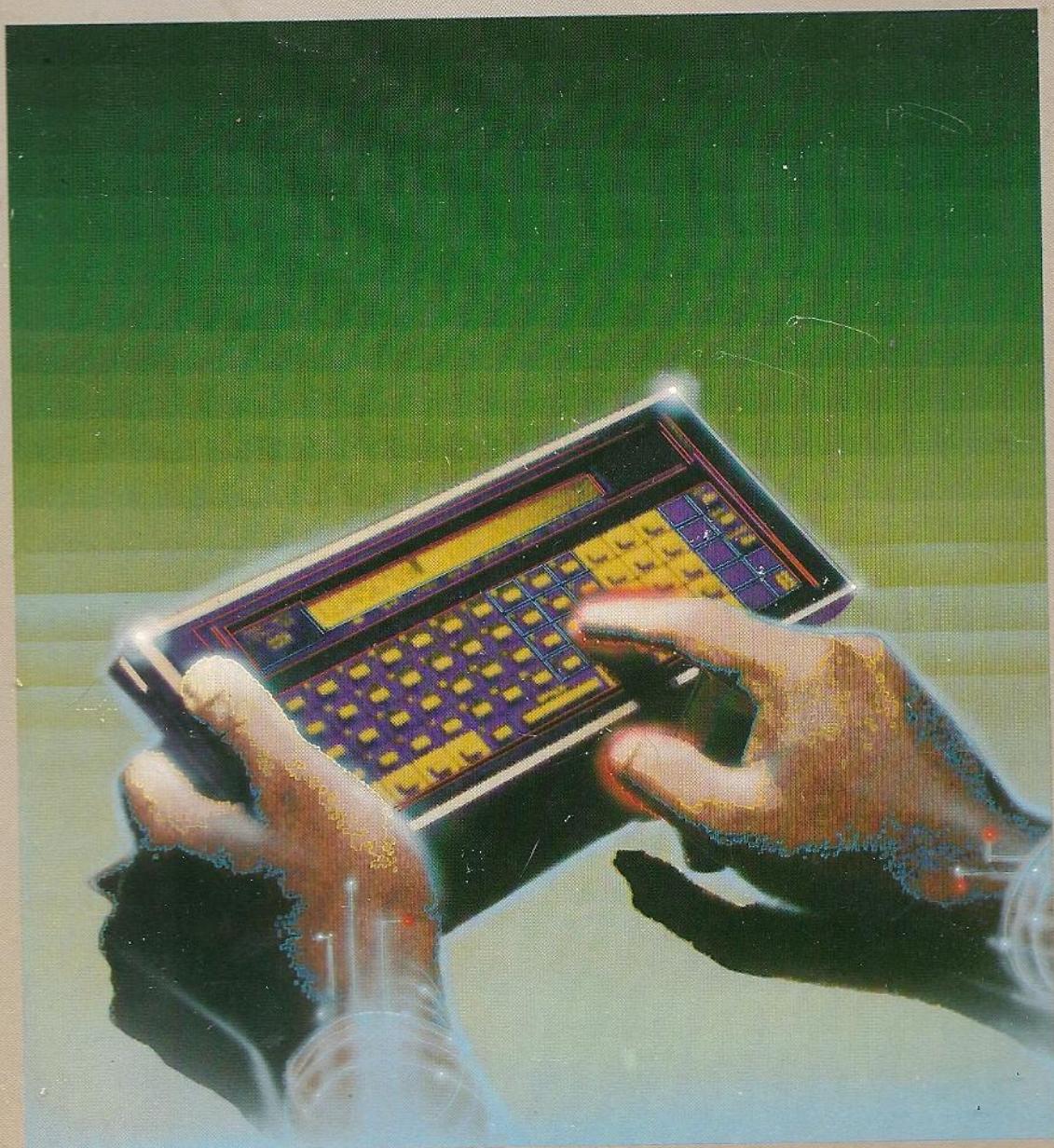
 Osborne/McGraw-Hill

# The **HHC**<sup>TM</sup> User Guide



By Jonathan Sachs, Sand River Software, with Rick Meyer



# **The HHC™ User Guide**



# The HHC™ User Guide

By Jonathan Sachs,  
Sand River Software,  
with Rick Meyer

Osborne/McGraw-Hill  
Berkeley, California



Published by  
Osborne/McGraw-Hill  
2600 Tenth Avenue  
Berkeley, California 94710  
U.S.A.

For information on translations and book distributors outside of the U.S.A., please write to Osborne/McGraw-Hill at the above address.

HHC is a trademark of Matsushita Electric Industrial Company, Ltd. Japan. *The HHC™ User Guide* is not sponsored or approved by Matsushita Electric Industrial Company, Ltd. Japan. All references to HHC in the text of this book are to the trademark of Matsushita Electric Industrial Company, Ltd. Japan.

Panasonic is a registered trademark of Panasonic Company.

Quasar is a registered trademark of Quasar Company.

Apple is a registered trademark of Apple Computer, Inc.

CBM is a trademark of Commodore Business Machines Inc.

MBASIC is a registered trademark of Microsoft Corporation.

#### THE HHC™ USER GUIDE

Copyright © 1983 by McGraw-Hill, Inc. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a data base or retrieval system, without the prior written permission of the publisher.

1234567890 DODO 89876543

ISBN 0-931988-87-X

Cover by Terry Hoff

Photos by John de Groot

Text design by KLT van Genderen

## Contents

Acknowledgments	vii
Preface	ix
1 Introducing the HHC	1
2 Getting Started	7
3 The File System	21
4 The Calculator	41
5 The Clock/Controller	53
6 Special-Purpose Keys and Memory Features	59
7 Running Programs on the HHC	71
8 Programming Language for the HHC	81
9 Using Peripherals	95
A HHC Quick Reference	129
B Cautions	135
C Internal Hardware and Software	137
D Keyboard and LCD Codes	145
E The TV Adaptor	151
F Annotated Bibliography	175
G Glossary	177
Index	189



## ACKNOWLEDGMENTS

Many thanks to Todd Umeda of Quasar Corporation for generous loans of equipment and for technical reviews. Others at Panasonic Corporation, Quasar Corporation, and Matsushita Electric Industrial Co., Ltd. provided information and technical reviews. Hans van der Vuurst of FORTH Team Utrecht, the Netherlands, provided a technical review of Appendix E. American Medical Instruments of Albany, California loaned equipment for last-minute testing and photographs.







System are used in every HHC application, from word processing to data collection and transmission.

Chapter 4 introduces the calculating capabilities of the HHC.

The Clock/Controller discussed in Chapter 5 displays the time and sets and acknowledges alarms. The Clock/Controller can be used as an electronic datebook with times and messages.

The function and program function keys and other special keys covered in Chapter 6 make the HHC more powerful and user-friendly. The Programmable Memory Peripheral expands the storage capabilities of the HHC.

Descriptions of the many applications programs which are available as program capsules for the HHC are contained in Chapter 7. These programs include time-management and analysis, word processing, data collection, telecommunications and others. This chapter includes information on how to use each program for maximum benefit.

The three high-level languages the HHC supports are covered in Chapter 8. SnapFORTH, the HHC's system language compiler, and the two BASIC versions, Microsoft BASIC and SnapBASIC, are compared for their applicability to the different tasks you might wish to use a programming language for.

Using peripherals, including the newest and most exciting peripherals, is covered in Chapter 9. The chapter shows you how to set up each peripheral and suggests the most efficient ways to use the peripheral. With this chapter you can preview peripherals you don't own.

Important words and phrases are printed in italics in the text and are fully defined and described in the glossary, Appendix G. Warnings and cautions for safe and efficient use of the HHC are boxed for easy reference and are gathered into Appendix B. Other appendixes gather information on hardware and software internals, provide a quick reference, explain the character set, and suggest further readings. Appendix E contains additional information for programmers hoping to do sophisticated work with the TV Adaptor peripheral.

# 1

## Introducing the HHC

**D**igital computers have been around for about forty years. During that time, advancing technology has continually made the machines smaller and more powerful. ENIAC, one of the earliest computers, weighed 30 tons and performed about a thousand operations per second. Today a typical desk-top computer weighs a few dozen pounds and performs more than a million operations per second.

Portable computers are the latest step in this evolution. They give you the power of a desk-top computer in a package so small that you can easily slip it into your briefcase.

Arthur C. Clarke's novel *Imperial Earth* introduced the character Duncan Makenzie, a twenty-third century citizen of Titan, and his MiniSec, a handheld computer. Makenzie's MiniSec connected to planet-wide computer systems via phone lines in hotel rooms. It acted as a data collection device, pocket secretary, and electronic datebook for jotting down ideas, formulas, and plans. What Clarke wrote was not wide of the mark. The MiniSec is available today—in the Panasonic/Quasar/Olympia HHC!

Up to now, three types of handheld computers have been in existence. The first type can be programmed in only one built-in, high-level language—usually a limited form of BASIC. As a result, the library of programs that you can run on such a machine is small, and the capability of each program is usually low.

The second type of handheld computer is actually a programmable portable terminal. This type of computer is programmed in a language that is powerful but not easy to use. The computer is usually designed so that once a program is written and installed in the computer, the computer can run that program and no other. As a result, this type of computer is practical only for fixed commercial applications, with many people using the machine for a single purpose.

The third type of handheld computer is the programmable pocket calculator. The programming language for this type of machine consists mostly of the simple operations you perform by pressing the machine's function keys—"add," "display," and so forth. This type of machine usually has less than a full typewriter keyboard and runs rather slowly. It is poorly suited for writing complex programs or writing programs that perform significant amounts of nonnumeric computing (for example, a word processing program).

These three types of handheld computers share certain other limitations. They generally have a small data storage capacity and a limited ability to communicate with other computers and devices such as printers.

The HHC, shown in Figure 1-1, is the first handheld computer that can offer you as much power as desk-top computer systems selling for several thousand dollars. The HHC accepts a large library of plug-in programs such

as specialized calculators, data entry systems, word processors, and more. It can also accept a wide range of peripheral devices, including printers, video terminals, devices for communicating with other computers by telephone—even a four-color plotter.

The HHC is a little larger than a pocket scientific calculator. You can use it as a datebook, a calculator, and an electronic notebook. With it you can record your thoughts and ideas for later incorporation into business plans, memos, or articles. You can use it to run dozens of programs that record data, edit text, and perform "what-if" projections from financial or other numeric data. If no prewritten program is suited to your needs, you can use the HHC to write your own program in any of several different computer languages that you can plug in just as you would a specialized program.

You can find many uses for the HHC in your business. If, for example, you have to keep track of your use of time for billing purposes, the HHC with its TIME/TRAC program can function as a highly automated portable time recorder.

If you sell computers, the HHC can offer you many new marketing possibilities. Since it is easily programmed, it can be readily adapted to fit "vertical marketing" niches in any specialized business that you are conducting or wish to enter.

Telecomputing programs give the HHC data communications capabilities. These programs let you collect data and perform computations while away from your home or office and then transfer the results to another computer by telephone.

## WHAT THE HHC CAN DO

The HHC is a very different device from conventional computers; its strong and weak points are different. If you have definite ideas about how computers should be used, working with the HHC is likely to broaden your outlook.

Roughly speaking, the HHC is appropriate for applications requiring at least one of the following capabilities:

- *A portable device for data collection.* Examples of such applications include sales order entry, survey response recording, meter reading, inventory control, and time-billing recording. The HHC is ideal for these applications, since it combines portability with great flexibility. Furthermore, the HHC can be programmed to perform complex data



Figure 1-1. The HHC



validation and user prompting, thus reducing the amount of attention the user must give to the computer.

- *Telecommunications: rapid, accurate transfer of data from one computer to another.* In sales order entry, for example, a sales person can transfer orders from the HHC in the field to the home office and transfer current inventory data from the office to the HHC in the field. An executive in the field can transfer letters and notes to the office for typing and examine electronic documents kept on file there. A reporter can use the HHC to write a story and then transfer the story directly to the newspaper's typesetting system. These tasks require only an HHC, a telephone, and a *modem*, a peripheral that can transfer computer data by telephone.
- *Portable information processing.* In addition to a built-in, four-function calculator, the HHC can run a scientific calculator and a spread sheet program.
- *Sturdiness and compact size: computing in hostile environments.* The HHC's ruggedness and small size make it especially well suited to work in places where other computers can't go. In some hospitals, for example, the HHC is used to calculate doses of anesthesia needed during surgery. To be used in the operating room, the HHC must be sterilized. Since this would have bad effects on any computer, the HHC is sealed into a plastic pouch before the entire pouch is sterilized. Running on battery power, the computer is used inside the pouch throughout the operation.
- *Process control: automatic control of a machine by a computer.* The HHC's small size and low power requirements make it ideal for process control. At the same time, its powerful system software and input/output facilities make it easy and economical to apply.
- *Custom programming.* Whenever no commercially available program fits your needs, you can develop a custom program (or have one developed for you). The HHC offers you a choice of several programming languages, making it possible to program the computer for a very broad spectrum of tasks.

There are also certain applications that are *not* appropriate for the HHC. These applications tend to have some of the following requirements:

- *Large quantities of data.* The storage capacity of the HHC is necessarily limited. The HHC alone can store about 7000 characters (7K) of data.

Each *programmable memory peripheral* can add another 16K of random-access memory (RAM) storage.

- *Large quantities of input.* The HHC's keyboard is smaller than a standard typewriter keyboard, and most users cannot touch-type on it. This makes the HHC inappropriate for uses that involve large quantities of keyboard input.
- *Large quantities of output.* The HHC's primary output medium is a *liquid crystal display (LCD)* that is 26 characters long. This makes the HHC awkward to use for applications that need to display large quantities of data.

# 2

## Getting Started

**I**n this chapter you will become familiar with the controls on the HHC and will learn how to run an HHC program.

The HHC models sold by Panasonic, Quasar, and Olympia are all manufactured by Matsushita Electric Company and are very similar. The illustrations in this book happen to show only Quasar equipment, but the information applies to all three brands. We will refer to all brands of the computer as "the HHC."

### A GUIDED TOUR OF THE HHC

Hold your HHC so that the front of the machine faces away from you, as shown in Figure 2-1.

The rectangular panel on the back can be lifted off by pulling up on the tab next to the legend OPEN. Lift the panel, as shown in Figure 2-2, and look at the compartment beneath. The compartment has three sockets for HHC capsules containing application programs.

If you have purchased an HHC capsule containing an application program, plug it into any one of these sockets. The capsule and socket are shaped so that you cannot insert the capsule the wrong way. Figure 2-3 demonstrates capsule insertion.

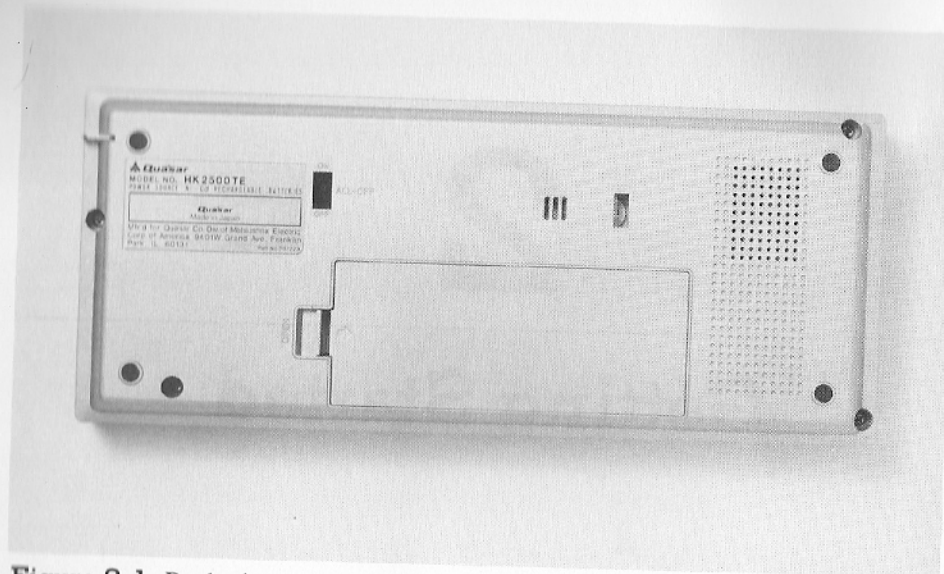


Figure 2-1. Back view of the HHC

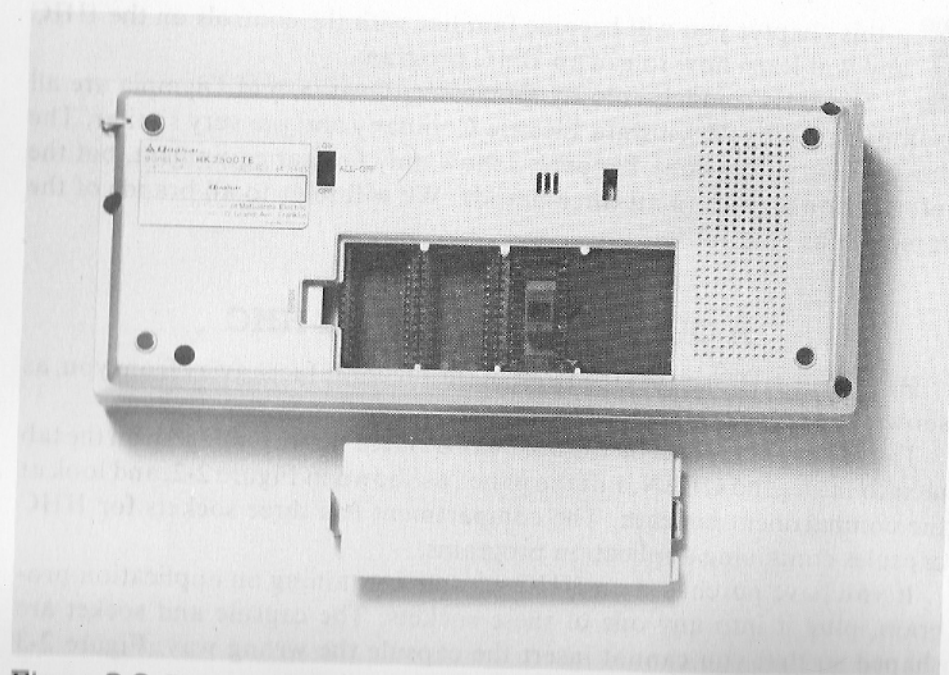


Figure 2-2. Program capsule compartment with one capsule in place

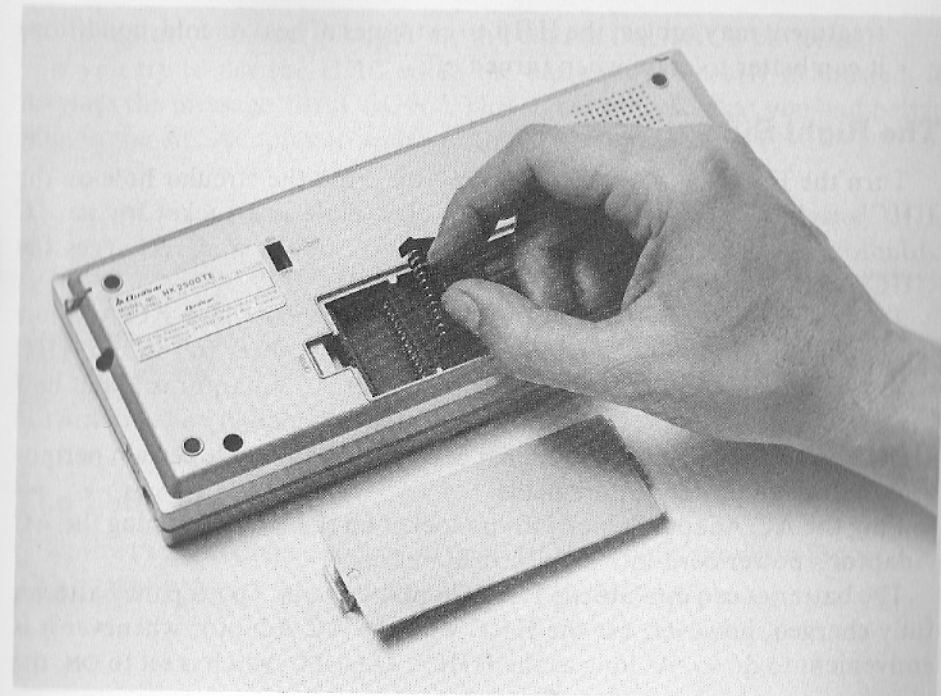


Figure 2-3. Program capsule being inserted into socket

Now reattach the panel that covers the HHC capsule sockets. Above the left end of the panel is a rectangular hole containing a slide switch. This is the *ALL-OFF* switch, so named because it controls the HHC's power supply and can turn off all parts of the HHC.

Use a slender object such as a pencil to move the *ALL-OFF* switch to its *ON* position.

The *ALL-OFF* switch may stay on at all times—even when you put the HHC away at the end of the day. That way the HHC can keep power on its memory and preserve the data stored there. You should turn off the *ALL-OFF* switch, however, in the following two situations:

- *If you are putting the HHC away for a period of weeks or more.* Preserving the contents of the HHC's memory could drain the batteries completely; if this happens repeatedly, the batteries' ability to hold a charge could be reduced. Turning off the *ALL-OFF* switch can prevent such a problem.
- *If you are shipping the HHC by mail, in luggage, and so forth.* Such



treatment may subject the HHC to extremes of heat or cold, conditions it can better tolerate when turned off.

### The Right Side

Turn the HHC so that its front faces you. Find the circular hole on the HHC's right side, shown in Figure 2-4. This hole is a socket for an *AC Adaptor*, which operates the HHC from AC power and recharges the HHC's batteries.

Two AC Adaptors are made for the HHC. The standard AC Adaptor weighs about two pounds and can deliver enough power to run the HHC with any combination of peripherals. A smaller AC Adaptor is about half the size and weight of the standard one and delivers enough power to run the HHC and one peripheral, although it cannot accommodate certain peripherals with high power requirements.

Plug the AC Adaptor's jack into its socket on the HHC and plug the AC Adaptor's power cord into an electrical outlet.

The batteries can operate the HHC for many hours. To keep the batteries fully charged, however, use the HHC with the AC Adaptor whenever it is convenient to do so. As long as the HHC's ALL-OFF switch is set to ON, the

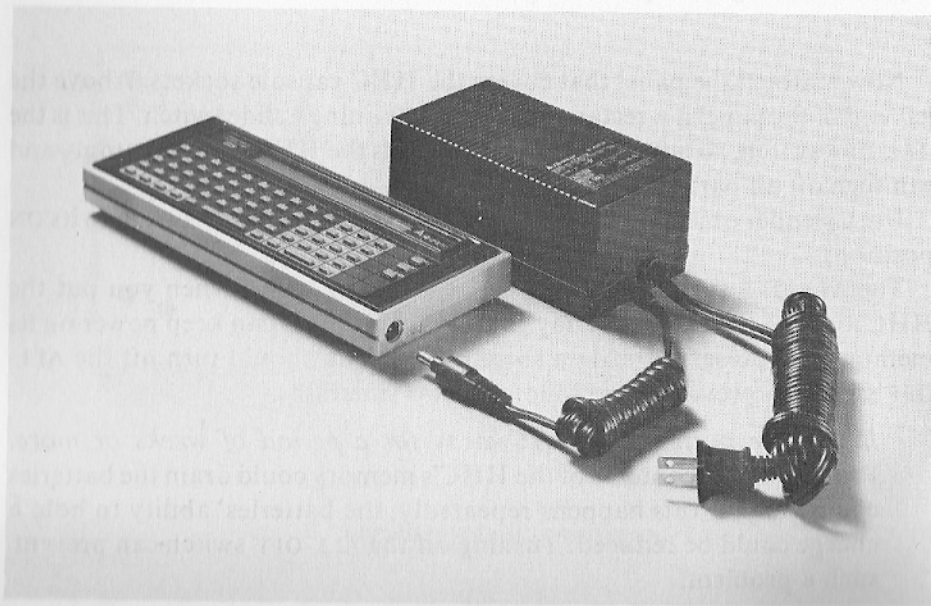


Figure 2-4. Right side of HHC: AC power jack

AC Adaptor, whenever plugged in, will charge the HHC's batteries.

If you try to use the HHC when the batteries are nearly exhausted, it displays the message "BAT LOW". This signal suggests that you had better plug in the AC Adaptor to recharge the batteries.

**Caution:** Many AC Adaptors can be set to operate on either 110-volt or 220-volt power. Be sure your AC Adaptor is set for the voltage that your electrical system provides. If you try to use the AC Adaptor at the wrong setting, you may damage it severely.

### The Left Side

Notice the slot-shaped socket on the HHC's left side, shown in Figure 2-5. This socket is where you can plug in an HHC peripheral.



Figure 2-5. Left side of HHC: bus slot



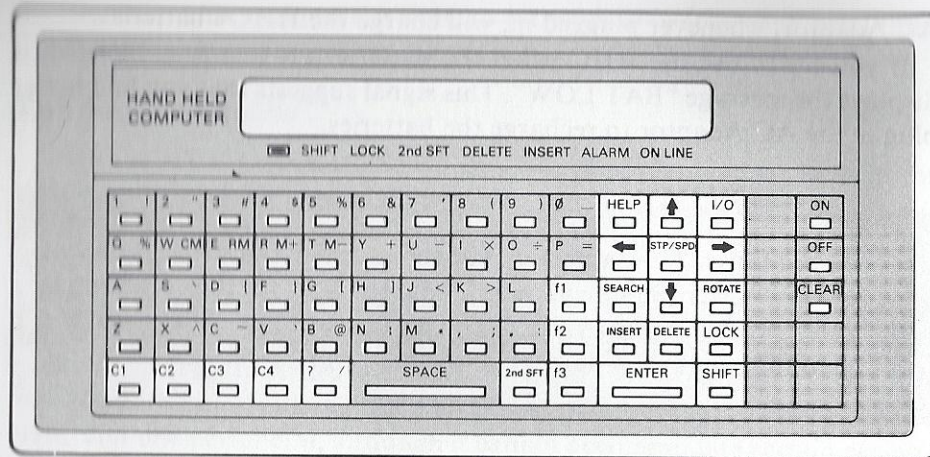


Figure 2-6. The HHC keyboard

## The Keyboard

To the right of the main keyboard shown in Figure 2-6 are the *ON* and *OFF* keys. You use these keys to turn the HHC on and off in everyday use. Unlike the ALL-OFF switch on the back of the HHC, the OFF key does not interrupt the flow of power to the HHC's memory. Thus, when you turn the HHC to OFF, the program it is running and the data that is stored in its memory will not be disturbed.

Below the ON and OFF keys is the *CLEAR* key. Use this key to reset the HHC when you have finished running most programs.

**Caution:** When you have finished running a program, always wait a second or so before pressing CLEAR. Many programs need this time to store their data in an orderly way.

On the left half of the keyboard are the *typing keys*, used to enter such displayable characters as A, Q, 3, and ?.

The *ENTER* key, located near the lower right corner of the keyboard, is similar to a typewriter's RETURN key.

On the right side of the main keyboard are the *editing keys*, used (among other things) to edit data. The editing keys are

1	2	3	4	5	6	7	8	9	0	HELP	↑	I/O	ON										
Q	%	W	CM	E	RM	R	M+	T	M-	Y	+	U	-	I	X	O	÷	P	=	←	STP/SPD	→	OFF
A	S	\	D	{	F	}	G	[	H	]	J	<	K	>	L	f1	SEARCH	↓	ROTATE				CLEAR
Z	X	^	C	~	V	`	B	@	N	:	M	*	,	;	.	:	f2	INSERT	DELETE	LOCK			
C1	C2	C3	C4	?	/	SPACE				2nd SFT	f3	ENTER		SHIFT									

The four *program function* keys are located in the lower left corner of the keyboard. Labeled C1 through C4, these keys are used to control many of the programs that run on the HHC. Sometimes they also function as typing keys, producing the special characters “ä,” “ë,” “ü,” and “ñ.”

1	2	3	4	5	6	7	8	9	0	HELP	↑	I/O	ON										
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>										
Q	%	W	CM	E	RM	R	M+	T	M-	Y	+	U	-	I	X	O	÷	P	=	←	STP/SPD	→	OFF
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
A	S	\	D	{	F	}	G	[	H	]	J	<	K	>	L	f1	SEARCH	↓	ROTATE				CLEAR
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Z	X	^	C	~	V	`	B	@	N	:	M	*	,	;	.	:	f2	INSERT	DELETE	LOCK			
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
C1	C2	C3	C4	?	/	SPACE				2nd SFT	f3	ENTER	SHIFT										
				<input type="checkbox"/>								<input type="checkbox"/>											

The remaining keys perform control functions for the HHC. These keys are

1	2	3	#	4	\$	5	%	6	&	7	'	8	(	9	)	0	—	HELP	↑	I/O		ON	
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Q	%	W	CM	E	RM	R	M+	T	M-	Y	+	U	-	I	×	O	÷	P	=	←	STP/SPD	→	OFF
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
A	S	\	D	{	F	}	G	[	H	]	J	<	K	>	L	f1	SEARCH	↓	ROTATE		CLEAR		
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Z	X	^	C	~	V	`	B	@	N	:	M	*	,	;	.	:	f2	INSERT	DELETE	LOCK			
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
C1	C2	C3	C4	?	/	SPACE					2nd SFT	f3	ENTER	SHIFT									
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

The *function* keys, labeled f1 through f3, let you type three frequently used keystroke sequences by pressing a single key. The *HELP* key lets you display a brief explanation of a particular key's function. The *I/O* key turns peripheral devices on and off and lets you select different memory devices containing programs and data.

The *STP/SPD* key “freezes” the HHC, halting a program run so that you



can inspect the output. The STP/SPD key also lets you control the speed at which the HHC displays data.

The *LOCK* key locks the effect of another key, somewhat like the *SHIFT LOCK* key on a typewriter. The *SHIFT* key and the *2nd SFT* key let you type upper-case letters and special characters.



As you learn more about the HHC, these keys will be discussed in greater detail.

### The Liquid Crystal Display

Above the keyboard is the *liquid crystal display*—LCD for short. The LCD can display one line of 26 characters.

Press the ON key now to turn the HHC on and see how the LCD looks in operation. (You may have to press the CLEAR key once or twice to make the LCD display something.) Press the OFF key to turn the HHC off again.

The LCD can display ordinary characters, inverse image characters—that is, clear characters on a black ground—and flashing characters. The LCD can also display special symbols and graphics, since HHC programs can directly manipulate the dots in LCD's dot matrix.

Just below the LCD is the row of words and symbols  SHIFT LOCK 2nd SFT DELETE INSERT ALARM ON LINE. Along the bottom of the LCD, the HHC displays a row of triangular *blips*, images that point to the row of words below the LCD and display information about the status of the HHC. When you press the SHIFT key, for example, the SHIFT blip goes on. The  blip has no predefined meaning, so an HHC program is free to use this blip for any purpose.

Turn the HHC on again. Press the SHIFT key to see what a blip looks like. Press the SHIFT key again to turn the blip off. Now turn the HHC off.

### INSIDE THE HHC

The “brain” of the HHC is a 6502 microprocessor—the same microprocessor that is used in such popular computers as the Apple II and Commodore CBM series.

The HHC's overall operation is managed by the *operating system*, a program that is permanently stored in the HHC's *read-only memory* (ROM). Three commonly used application programs are also stored in the ROM.

The HHC contains read/write *random-access memory* (RAM) for storing data. Depending on your model of the HHC, there may be approximately 4000 or 8000 *bytes* (characters) of RAM.

The HHC's batteries deliver power to RAM at all times; thus, even when turned off, the HHC can preserve its memory. The HHC uses a special low power type of RAM that can run off the batteries for several months without exhausting their charge.

These internal components of the HHC, and their interrelationships, are shown in Figure 2-7.

### THE HHC ACCESSORIES

Let's take a quick look at some of the accessories available for the HHC. Every HHC comes with

- A soft plastic *carrying case* that you can use to protect your HHC when it is not in use
- A *carrying strap* that you can clip to a small lug mounted underneath the HHC's upper right corner
- A *plastic sleeve* that slides into the groove around the sides of the HHC. You can use this sleeve to hold together the HHC and a peripheral. You can also slide the sleeve over the face of the HHC to protect the keyboard from accidental pokes.

The Programmable Memory Peripheral (PMP) contains up to 16K of additional RAM. You can use it to store HHC programs and data; it is shown in Figure 2-8.

Several other peripheral devices are available for the HHC including an RS-232C Interface for printers, video terminals, and similar equipment; a Modem for telecommunications; a TV Adaptor for display of text and graphics on a TV or video monitor; a battery-powered portable printer; and

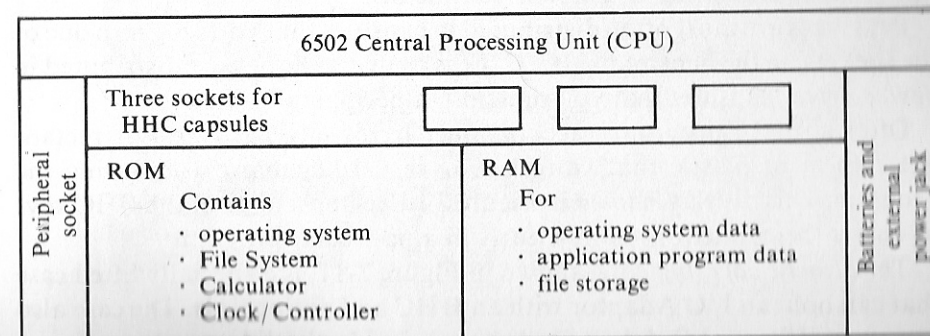


Figure 2-7. Internal components of the HHC



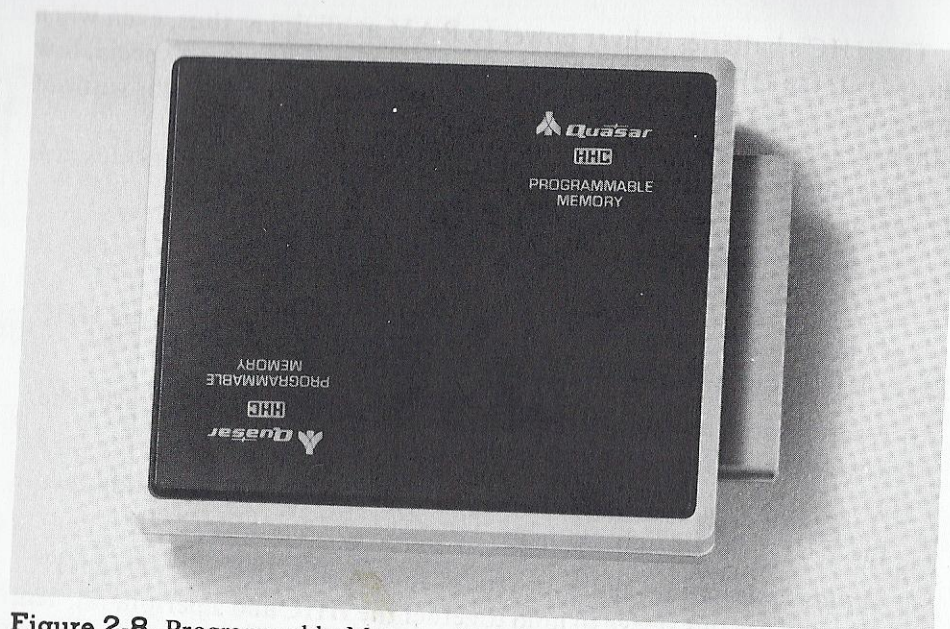


Figure 2-8. Programmable Memory Peripheral or PMP

a Four Color Plotter. Shown in Figure 2-9 are the RS-232C Interface, Modem, Micro Printer, TV Adaptor, PMP, and I/O Adaptor.

The *I/O Adaptor* is a plastic tray that can connect the HHC to as many as six peripherals at once. The HHC slides into the lower right channel of the I/O Adaptor, and its peripheral socket connects to a plug on the Adaptor. Five peripherals slide into the other channels, and their plugs fit into sockets on the Adaptor. On top of the Adaptor is a sixth socket; a cable inserted into this socket connects an additional peripheral.

Programs are most often distributed in capsules that you plug into one of the sockets on the back of the HHC. Less frequently, they are distributed in *device driver capsules* that you plug into a peripheral device.

Often an HHC program is accompanied by a *keyboard overlay*, a rectangular sheet of plastic that you place over the keyboard when using the program. This overlay, shown in Figure 2-10, relabels some of the HHC keys to reflect their functions more clearly in a particular program.

The *attache carrying case*, shown in Figure 2-11, is a specially fitted case that can hold an I/O Adaptor with an HHC and peripherals. The case also has space for an AC Adaptor. A pouch inside the lid can store several instruction manuals and keyboard overlays.

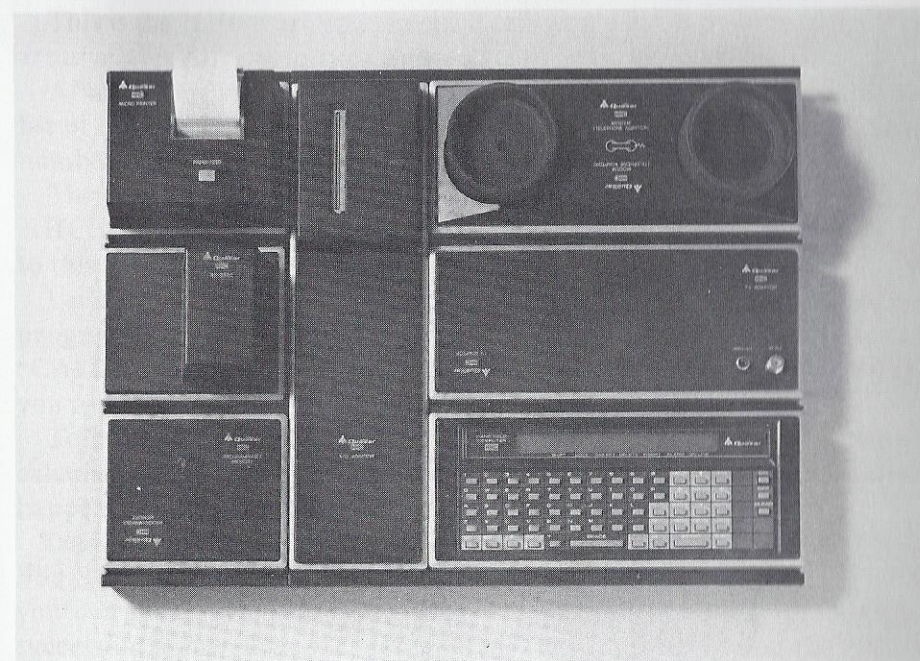


Figure 2-9. The RS-232C Interface, Modem, Micro Printer, I/O Adaptor, TV Adaptor, and PMP



Figure 2-10. HHC with overlay





Figure 2-11. Attache case, holding the HHC and peripherals

## THE PRIMARY MENU

To run a program on the HHC, turn the HHC on. Then press the CLEAR key, if necessary, to make the LCD display start changing. You should see the following LCD display:

1 = CALCULATOR

2 = CLOCK/CONTROLLER

3 = FILE SYSTEM

4 = RUN SNAP PROGRAMS

This is the HHC's *primary menu*. It tells you what programs in the HHC are available for you to run.

A "menu" is so named because—just like a restaurant menu—it contains a list of options that you can choose. You choose an option by pressing the number key that the menu lists for a particular program.

The primary menu is the starting point for everything you do with the HHC. If you press the CLEAR key repeatedly, it will always return the HHC to this menu.

Let's try the menu to see how it works. Press the 1 key, which chooses the program entitled Calculator. After you press 1, the HHC displays "CALCULATOR" on the LCD. It then displays the digit "0" and waits for you to enter the calculation you want it to perform.

To leave the Calculator, press the CLEAR key twice: once to clear any calculation you may be performing (you aren't doing one, so nothing happens) and once to return the HHC to the primary menu.

Let's try another program. Press 2 for the Clock/Controller or 3 for the File System. Each of these programs displays another menu from which you can make a selection. Many HHC programs work by presenting a succession of menus that let you specify exactly what you want the HHC to do.

Again, you must press CLEAR twice to return to the primary menu. Do so now and turn the HHC off.

## THE AUTO-OFF FEATURE

If you leave the HHC on without pressing any key for a period of about ten minutes, the HHC will turn itself off. This *auto-off* feature protects you from draining the HHC's batteries should you forget to turn off the HHC.

## CARING FOR THE HHC

The HHC's extreme portability does not remove your responsibility to care for it properly. You must protect the HHC from impact and other physical hazards, and you must not abuse the batteries.

### Protection

Your HHC is designed to be rugged and reliable, but it is still a complex electronic instrument and contains delicate parts. Protect it from being dropped on concrete, struck with heavy blunt instruments, and so forth.



The most delicate part of the HHC is the faceplate of the LCD. Made of fairly soft plastic, the faceplate is liable to be scratched if the HHC is thrown loose into a drawer or an attache case. Protect the HHC by slipping it into its soft plastic carrying case or sliding the rigid plastic sleeve over its face.

### Temperature Extremes

The operating range of the HHC and its peripherals is 0°C to 40°C (approximately 32°F to 104°F).

The HHC may be stored or shipped safely at somewhat lower and higher temperatures if the ALL-OFF switch is moved to the OFF position.

### Caring for the Batteries

Your HHC is delivered with rechargeable nickel-cadmium batteries, which will last indefinitely with proper care.

When your HHC is delivered, the batteries will probably have no charge. Leave your HHC connected to an AC Adaptor to charge the batteries. An overnight connection should charge the batteries fully.

To charge the batteries quickly, leave your HHC turned off. You can also charge the batteries while the HHC is in use, but the charging process will take longer. How much longer depends on how heavily the HHC (or any connected peripheral device) is being used.

The first few times you charge the HHC's batteries, you should expect them to run down quickly. After several recharges, the batteries' ability to hold a charge will increase to normal.

Since it takes some power to preserve the contents of the HHC's memory, the batteries gradually lose their charge when the HHC is turned off. Even if the ALL-OFF switch is off, the batteries will gradually lose their charge spontaneously.

Do not allow the batteries to remain uncharged for long periods of time; they may permanently lose their ability to hold a charge and have to be replaced. Recharge the batteries about every six months, even if your HHC is unused.

Should your HHC's batteries become exhausted or damaged, take your HHC to a qualified maintenance service to have the batteries replaced.

# 3

## The File System

**T**his chapter discusses the role of files in the HHC. You will learn to use the HHC File System, which enables you to create and modify text files through the keyboard.

### WHAT IS A FILE?

File is a computer term for a collection of data grouped together under one name. The HHC File System is analogous to a conventional filing system; one file folder usually contains a specific collection of data. For example, one file might contain information about orders and payments from a particular customer.

Just as a conventional file is stored in a folder, a computer file must be stored on some particular medium. Reels of magnetic tape, magnetic disks, and punched cards are examples of media that can be used to store computer files. On the HHC, the most important medium for file storage is *intrinsic RAM*. The HHC's intrinsic RAM is used for storing files and programs.



A file can contain any sort of data that the HHC can process. Some common uses for files are

- *Storing text.* For example, a file may contain notes that you write to yourself; documents that you have created with a text editor; or reports generated by application programs.
- *Storing data.* A file may contain any sort of data that you create for input to a program. A program may also create a file for input to another program.
- *Storing a program.* If you write your own programs for the HHC, you will generally keep them in files.
- *Storing system data.* Some parts of the HHC's operating system use files to store data that they must preserve. For example, the operating system component that controls the RS-232C Interface stores that device's configuration parameters in a file. These files are "invisible" to you, since only the computer uses them.

## CREATING A TEXT FILE

The *File System*, option 3 on the HHC's primary menu, allows you to create, modify, and manage text files by typing on the HHC's keyboard.

The File System has an *editor* that allows you to create or modify one file at a time. The file being edited is called the *current* file. Within the current file, the editor displays one line at a time for you to inspect and modify. This line is called the *current* line.

Turn on your HHC and choose option 3. The HHC displays "FILE SYSTEM" to tell you what program it is running and then displays a menu that looks like the following:

1=NEW FILE

2=COPY FILE

NO FILES

Choose option 1. Now the File System prompts you to type a file name and press ENTER. Let's name this file "mary had a little lamb". Type that

name and press the ENTER key to get

mary had a little lamb

When you press ENTER, the File System's editor starts running and then displays the first line of the file. Because this file is empty, the LCD will be blank.

The *cursor*, a solid blinking rectangle at the left edge of the display, indicates where the next character you type will go in the current line.

Study the following information, which you will soon enter—errors and all—in your file:

mary had a little lamb,  
its fleece was white as snoe,  
an everywhere that mary went  
the lambb was sure to go

In this chapter's section on "Correcting Errors" you'll learn how to correct the preceding errors for practice in using the editor. (If you make some additional errors on your own when you enter the poem, so much the better.)

To place the first line of the nursery rhyme in the first line of the file, simply type the data on the HHC's keyboard. When you finish typing a line, press ENTER. The File System will move the cursor to the start of the next line, just as if you had pressed the RETURN key on a typewriter.

Since the second line of the poem is more than 26 characters long, it won't fit on the LCD. When you type the 27th character, the editor *scrolls* the display and the beginning of the line moves off the LCD so that you can see the characters you type at the end. The File System lets you create a line up to 80 characters long.

## Naming a File

A file name may be any length from one to 80 characters, and it may include blanks, punctuation, or any other character that you can type on the HHC keyboard. Every character is significant, and upper or lower case of every letter is significant. If you name a file "March AR", for example, that name is distinct from "March ar", "March A.R.", or "MARCH AR".



Because the LCD holds 26 characters, keep the length of the name to 24 characters or fewer. The number and “=” take up two characters.

### Saving a File

After you have ended the last line of your file by pressing ENTER, press the CLEAR key. This saves the file and returns you to the File System menu. Now the menu displays

1=NEW FILE
2=COPY FILE
3=mary had a little lamb

Notice how completely your file name describes its contents. Most other computers do not allow a file name that is this long or that consists of several words.

### Retrieving a File

To edit the file again, choose option 3, which the File System menu now tells you is the “mary had a little lamb” file. The File System displays the file name, first in ordinary characters and then in inverse image characters, with the cursor immediately after it. And so you see

mary had a little lamb
mary had a little lamb █

Press the ↓ key, which is located near the right side of the keyboard. The File System’s editor moves “down” from the file name to the first line of the file.

Notice that you can press ENTER in place of ↓ to move the cursor to the next line. The reverse is not true: you cannot press ↓ to create a new line at the end of the file. In that case you must use ENTER. If you use ↓ and no line exists after the current line in the file, the editor beeps to indicate that you have tried to move beyond the end of the file.

### Cursor Control for Moving Through a File

Using the four arrow keys, try moving the cursor around within the file. Notice that you can move the cursor in all four directions. If you try to move the cursor to a part of a line that is not on the LCD, the File System obligingly scrolls the line for you. You can’t move the cursor before the first line of the file or more than one line past the last line, nor can you move the cursor before the beginning of a line or more than one character past the end of a line. When you first enter the editor, the ↓ key moves you from the file name down to the first line of the file; but the ↑ key will not move you up from the first line of the file to the file name. If you try to move the cursor “out of bounds” the editor will not move the cursor, but will beep.

When you display a line that is more than 26 characters long, the File System scrolls through the entire line. You can halt it at any point by pressing any typing key or editing key once.

### CORRECTING ERRORS

Now let’s correct those errors that you introduced into the “mary had a little lamb” file. In the passages that follow you will learn how to substitute characters, insert or delete text, and use the File System’s capabilities to do more extensive editing.

#### Change a Character

The first error is located at the end of the second line, where “snow” is spelled “snoe”. You can correct this kind of error simply by placing the cursor over the incorrect character and typing the correct character.

Move the cursor to the second line with ↓ or ENTER. Then move it over the “e” in “snoe” with →. When you press w, the character “w” replaces the “e” and the cursor moves forward one character.

#### Insert a Character



The next error is located on the third line, where the “d” is missing in “and”. Here you must insert a character within a line. You can insert a character by placing the cursor immediately after the insertion point. Press the INSERT key and then enter the character you want to insert.

Move the cursor down to the third line by pressing ↓, and move it back to the space following “an” with ←. Here’s a hint—hold down the ← and, after a





brief pause, the key will *auto-repeat* until you release it. All of the HHC's typing keys and most of the editing keys do this.

Notice the four bars at the right edge of the LCD. The editor displays this symbol when the right end of the line is off the LCD.

an  everywhere that many we 

Pressing INSERT causes the following: The cursor changes from a solid rectangle to a checkerboard pattern, and the INSERT blip on the LCD goes on.

an  everywhere that many we 

Press the d key. The cursor moves one position to the right and returns to its normal solid form. The space that was under the cursor and all the characters following it move one position to the right. A "d" is inserted where the cursor was, and the INSERT blip goes off.



The INSERT key applies to one character at a time. If you want to insert a whole word into a line, you must press INSERT before each letter in the word. (You'll learn a way to avoid that dull chore in a later section.)

It's very easy to extend a line. Position the cursor at the end of the line and type. You don't have to press INSERT to do this.

## Delete a Character

The next error is located in the fourth line, where "lamb" is spelled with two "b"s. You must delete one "b". You can delete a character by placing the cursor over that character, pressing the DELETE key, and pressing → or ←.

Place the cursor over either "b" in "lambb". Pressing DELETE changes the cursor from a solid rectangle to a hollow one and turns on the DELETE blip on the LCD.

the lamb  was sure to go. 

Now press →. Notice that the cursor remains in place but returns to its normal solid form. The "b" that was under the cursor is deleted, all the characters following it are moved one position to the left, and the DELETE blip goes off. Like INSERT, DELETE applies to one character at a time.







Before:	its fleec  was white as sn 
	-----
After:	it  was white as snow
Before:	its fleec  was white as sn 
	-----
After:	its  was white as snow

Figure 3-1. "Delete Left" and "Delete Right"

## Delete "Right to Left"

You can delete a character by pressing DELETE ←, as well as by pressing DELETE →. But DELETE ← produces a slightly different effect: Instead of leaving the cursor in place, it moves the cursor one position left.

Let's try this. Reinsert the "b" that you just deleted from "lamb"—remember, position the cursor, press INSERT, and then type the letter you want to insert. Now place the cursor on the extra "b" and press DELETE ←.

You can think of DELETE ← as "delete left." If you enter this sequence several times, it deletes a string of characters running left from the character under the cursor, as shown in Figure 3-1.

Similarly, you can think of the DELETE → sequence as "delete right." If you enter this sequence several times, it deletes a string of characters running right from the character under the cursor (Figure 3-1).

## Cancel a DELETE or INSERT

If you press INSERT and then decide you don't want to insert a character after all, just press INSERT again. The second INSERT cancels the effect of the first. Similarly, if you press DELETE, a second DELETE will cancel the effect of the first.

You can also move directly from INSERT to DELETE by pressing DELETE, or from DELETE to INSERT by pressing INSERT.



## SHIFT AND 2nd SFT

Now you're going to learn how to enter capital letters and special characters. To practice these techniques, you'll correct the capitalization and punctuation in your file.

You type a capitalized letter by pressing the SHIFT key, but the HHC's SHIFT key works a little differently from a typewriter's SHIFT key. On a typewriter you hold the SHIFT key down *while* you press a typing key. On the HHC, you press the SHIFT key *before* you press a typing key. Let's try it. Place the cursor over the "m" at the start of the first line. Press SHIFT and then release it. Notice that the SHIFT blip went on. Now press M. The File System replaces the "m" in "mary" with a shifted—capitalized—"M", and turns the SHIFT blip off again.

You may shift several characters in a row by holding down the SHIFT key while you type, but you must release the SHIFT key *before* you type the last character that you want to shift.

For practice, capitalize the first letters on the remaining three lines and the "m" in the nursery rhyme's second occurrence of "mary."

Now what about punctuation? Most of the HHC's punctuation characters are located in an additional case called *second shift*. To get a semicolon, colon, or any other second shift character, press the 2nd SFT key before you press the typing key.

The second line of your file should end with a semicolon, not a comma. The semicolon is the second shift character located on the "comma" key.

Place the cursor over the comma at the end of the second line. Press 2nd SFT and then release it. Notice that the 2nd SFT blip went on. Now press the "semicolon" (or "comma") key. The File System replaces the comma with a semicolon and turns the 2nd SFT blip off again.

The keys SHIFT and 2nd SFT are like INSERT and DELETE in the following ways:

- They act on only one character at a time.
- If you press SHIFT or 2nd SFT and then decide you don't want a shifted character, you can press the same key again and the second action will cancel the effect of the first.
- You can move directly from SHIFT to 2nd SFT by pressing 2nd SFT or from 2nd SFT to SHIFT by pressing SHIFT.

Notice the following unusual properties of SHIFT and 2nd SFT:

- In the second row of the keyboard the symbols CM to ÷ (on the w

through O keys) look like second shift characters but are not. They are special symbols used by the Calculator program. If you use any one of these keys with 2nd SFT in the File System, you will get a blank.

- You can type the symbols ! to \_ (on the 1 through 0 keys) with either SHIFT or 2nd SFT.
- The symbols ? and / are on the same key; but ? is an unshifted character and / is a second shifted character. This is the opposite of the usual typewriter arrangement.

## The LOCK Key

Suppose you want to CAPITALIZE A GROUP OF LETTERS LIKE THIS ONE. Pressing SHIFT before each letter would be quite a bore; the HHC solves this problem with the LOCK key.

To type a series of shifted characters, press LOCK and then SHIFT. Notice that the LOCK and SHIFT blips go on. Now *everything* you type will be shifted until you press SHIFT again, which will return you to unshifted mode.

Try using LOCK SHIFT to add some shifted characters to your file.

Similarly, to type a series of second shifted characters, press LOCK and then 2nd SFT. After the LOCK and 2nd SFT blips go on, everything you type will be second shifted until you press 2nd SFT again to return to the unshifted mode.

You can also cancel LOCK SHIFT by pressing 2nd SFT. This action puts the HHC in second shift for one character and then returns it to unshifted mode. Similarly, you can cancel LOCK 2nd SFT by pressing SHIFT, and at this point the HHC will shift for one character and then return to unshifted mode.

## The LOCK Key with INSERT and DELETE

You can use LOCK in conjunction with INSERT and DELETE, as well as with SHIFT and 2nd SFT.

To illustrate this, delete the word "fleece" from the second line of the file. Begin by placing the cursor on the first letter of "fleece" and pressing LOCK DELETE. Notice that the cursor changes to its "delete" form and the DELETE blip goes on. (The LOCK blip does not go on; it is displayed with SHIFT and 2nd SFT only.)

Now press → once for each character you want to delete. When you have finished deleting, press DELETE once more to cancel LOCK DELETE. Notice that the DELETE blip goes off.

To reinsert the word "fleece", place the cursor on the first character of



"was", the place where the word "fleece" should begin. Now press LOCK INSERT. At this point, the cursor changes to its "insert" form and the INSERT blip goes on. The LOCK blip does not. Now type the word "fleece" and then a space, and press INSERT once more to cancel LOCK INSERT. Notice that the INSERT blip goes off again and the cursor returns to its regular solid form.

**Caution:** Remember to cancel LOCK DELETE or LOCK INSERT when you have finished deleting or inserting. If you press ENTER or any of the arrow keys while DELETE or INSERT is in effect, you will delete or insert characters or lines in your file.

### Open Up Space In a Line

The File System allows you to create space within a line and type characters into the space. This is sometimes more convenient than typing text with INSERT.

To insert a space, press INSERT and then ← or →. Each time you press →, the cursor is moved one position to the right and a space is inserted in the line at the cursor. Each time you press ←, the cursor does not move, but a space is inserted at the cursor.

Use ← and → with LOCK INSERT to insert several spaces. Press LOCK INSERT ← ← ..., and the HHC inserts several spaces and leaves the cursor on the first space. Press LOCK INSERT → → ..., and the computer inserts several spaces and leaves the cursor on the last space. Remember to cancel LOCK INSERT when you have finished inserting spaces.

### DELETE/INSERT and SHIFT/2nd SFT In Other Programs

The DELETE, INSERT, LOCK DELETE, and LOCK INSERT operations that you have learned are part of the File System. Most other programs apply these keys the same way for entering text, but many programs use the keys for different functions in different contexts. Such programs are accompanied by keyboard overlays that relabel the keys to reflect their functions.

On the other hand, the SHIFT, 2nd SFT, LOCK SHIFT, and LOCK 2nd SFT operations that you have learned are part of the HHC's operating system. Thus, these keys work the same way for every program that runs on the HHC.

## LINES

You can use the INSERT and DELETE keys to insert and delete whole lines in a file, as well as characters in a line.

### Delete a Line

To delete the second line of the poem, place the cursor anywhere on the second line and press DELETE and ↓. The File System deletes the line and moves the cursor, making the next line (formerly the third) the current line. (Figure 3-2.)

Pressing DELETE and ↑ deletes the current line and moves the cursor up to make the preceding line the current line.

### Insert a Line

To reinsert the deleted line, move the cursor up to the first line of the file. Press INSERT, then ↓ (or ENTER), as shown in Figure 3-3. The File System creates a new, empty line after the first line and moves the cursor to the empty line. You may now type in the line.

To create a new, empty line *before* the current line, press INSERT and ↑, a sequence depicted in Figure 3-3. Notice that the cursor moves to the empty line.

Before:	Mary had a little lamb, Its fleece was white as snow; And everywhere that Mary went The lamb was sure to go.
After DELETE ↓:	Mary had a little lamb, And everywhere that Mary went The lamb was sure to go.
After DELETE ↑:	Mary had a little lamb, And everywhere that Mary went The lamb was sure to go.

Figure 3-2. "Delete Up" and "Delete Down"



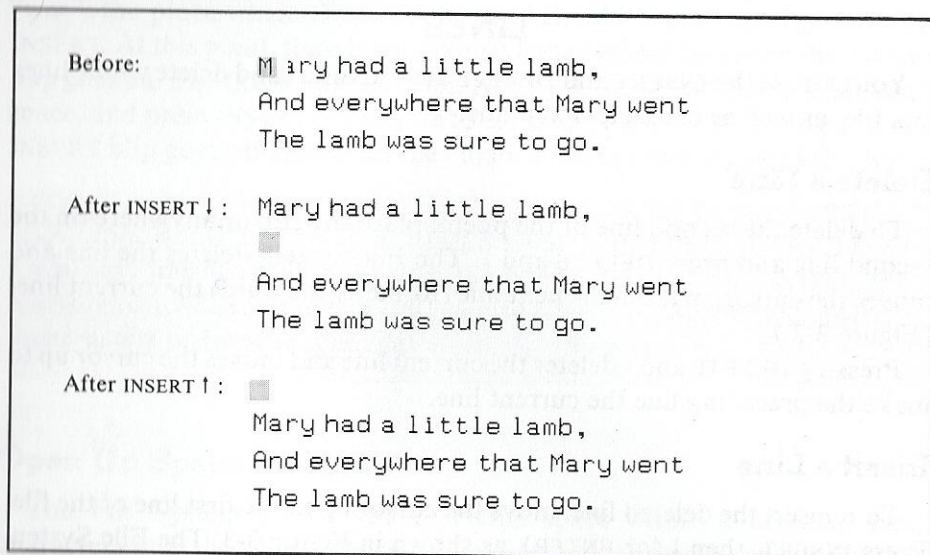


Figure 3-3. "Insert Up" and "Insert Down"

### LOCK DELETE and LOCK INSERT for Lines

The keys LOCK DELETE and LOCK INSERT affect lines as well as individual characters.

For example, to delete several lines, place the cursor on the first line to be deleted and then press LOCK DELETE ↓ ↓ ↓. Each ↓ deletes the current line and moves the cursor to make the following line the current line. After you have deleted the desired lines, press DELETE again to cancel LOCK DELETE.

To insert several lines, place the cursor on the line that the inserted lines will follow and then press LOCK INSERT ↓ (text) ↓ (text) ↓ (text). After you have entered the last line of text, press INSERT again to cancel LOCK INSERT.

### Locking Arrow Keys

You can press LOCK before an arrow key to make any editing command auto-repeat. The command will simply execute continuously until it reaches a "natural" stopping point, or until you interrupt it by pressing any typing or editing key.

For example, to move the cursor left one character, press ←; to move the cursor to the beginning of the current line, press LOCK ←. The beginning of the line is the "natural" stopping point for this command. You can stop the cursor's movement by pressing a typing or editing key.

Similarly, to move the cursor to the end of the current line, press LOCK →; to move the cursor to the first line of the file, press LOCK ↑; and to move the cursor to the last line of the file, press LOCK ↓.

To delete all characters from the immediate left of the cursor to the start of the line, press LOCK DELETE LOCK ←. To delete all characters from the cursor to the end of the line, press LOCK DELETE LOCK →.

### The ROTATE Key

To review the entire contents of a line that is too long to be displayed on the LCD, press the ROTATE key, located on the right side of the keyboard. The ROTATE key continuously scrolls the current line across the LCD from right to left, separating the end of the line from the beginning by several blanks. Press any typing or editing key to make the LCD stop rotating.

## SEARCH

To search a file for a particular string of characters, press the SEARCH key while editing the file. The File System prompts you with the phrase "SEARCH FOR?" Enter the string you want to search for. (It may be up to 12 characters long.) Now press ENTER.

If SEARCH finds the string you entered, it displays the line containing the string with the cursor on the first character of the string. If SEARCH does not find the string, it beeps, briefly displays the message "NOT FOUND", and redisplay the current line. Now let's try SEARCH.

Move the cursor to the beginning of the file and then press the SEARCH key. When the File System prompts you with "SEARCH FOR?", type in the string "white" and press ENTER. The File System displays the second line of your file and leaves the cursor on the "w" at the beginning of the string "white".

### Where SEARCH Starts

When pressed, the SEARCH key starts searching at the position of the cursor and searches to the end of the file. If it hasn't found the string by then, it goes to the beginning of the file and searches as far as the cursor. Let's demonstrate this.

With the cursor still on the string "white", press SEARCH, type in the string "Mary", and press ENTER. The File System finds "Mary" in the third line, since that is the first occurrence of "Mary" at (or after) the cursor.



Without moving the cursor, repeat the search. Press SEARCH, type in "Mary", and press ENTER. The File System leaves the cursor right where it is. Do you see why?

Remember, SEARCH looks for the first occurrence of the string at—or after—the cursor. If the cursor is on the first character of "Mary", SEARCH will find "Mary" right there.

Now move the cursor one position to the right and repeat the SEARCH again. This time SEARCH does not find the string "Mary" anywhere before the end of the file, so it returns to the beginning and finds "Mary" there.

### Strings Rather Than Words

Note that SEARCH looks for a string, not a word or a group of words. A string is any consecutive group of characters on the same line. It does not necessarily begin and end with a blank.

To demonstrate this point, move the cursor to the start of the file and search for the string "as". Now the SEARCH key finds the string in the second line—not in the word "as", but in the word "was". Move the cursor to the right at least one position and search again. This time SEARCH finds the word "as", which is the next occurrence of the string "as". Move the cursor to the right and search again; now SEARCH finds "as" in the fourth line, again in the word "was".

You can find a word by searching for a string that consists of the word surrounded by blanks. To find the word "as" in our poem, search for the string "basb" (where "b" represents a blank). However, if you search for a string that begins and ends with a blank, you will miss words that begin or end a line. Thus, the "surrounded by blanks" technique is most useful when you are referencing a printout of the file and you want to move the cursor quickly to a specific word that you can see is (or is not) surrounded by blanks. The technique works less well if you want to find the next occurrence of a word but don't know where it happens to fall in a line.

### Notes on SEARCH

Remember, SEARCH pays attention to case; if you search for "mary", you won't find "Mary" or "MARY". You can use the character editing functions of INSERT, DELETE, and LOCK while entering the SEARCH string. The SEARCH key is a function of the File System. Don't assume that SEARCH will perform similarly in other programs.

### When to Press CLEAR

The editor does not store a new or modified line in the current file until you press ↓, ↑, or ENTER. Thus, you can undo any error you may make while editing a line by pressing CLEAR before you leave the line. The editor will return to the File System menu, and the changes you made to the last line will be discarded. Then you can reselect the same file and resume editing.

*Caution:* When you have finished working on a file, you must leave the last line you edited before you press CLEAR. If you do not do so, the changes that you made in the line will not be preserved. Be careful not to press CLEAR immediately after pressing ENTER, ↓, or ↑; the File System needs time to store the updated line in the file. Waiting a second or so is usually enough.

## OTHER FILE OPERATIONS

The File System provides three file operations in addition to editing text files: It lets you copy, rename, and delete files.

The File System can perform these additional operations on any file, including a file containing a program that was not created by the File System editor. (If you try to edit a program file, the File System beeps and displays the message "CAN'T EDIT".)

### Copy a File

You can copy a file with the COPY FILE option of the File System menu. Let's try copying the "mary had a little lamb" file.

Return to the File System menu and choose option 2, COPY FILE. The File System displays the instruction "SELECT FILE", followed by a menu of files you can copy. Unless you've done some experimenting on your own, the only item on this menu is "mary had a little lamb".

1=NEW FILE

2=COPY FILE

3=mary had a little lamb



Select "mary had a little lamb" by choosing option 3 (or whichever option number represents the file in your menu).

Now the File System displays the message "SELECT DESTINATION RAM", followed by the one-item menu

```
1=INT RAM, 2223 FREE
```

The phrase "INT RAM" means *intrinsic RAM*. This menu is part of a feature that lets you copy files back and forth between intrinsic RAM and a Programmable Memory Peripheral. For the moment, just press the 1 key; you'll learn how to use the menu in Chapter 6.

After a moment, the File System displays the message "COPY DONE" and returns to the RAM menu. Notice that the menu now shows two files named "mary had a little lamb". The first one is your original file, and the second one is the copy.

## Rename a File

Now you have two files with the same name. If you modify one of them, how will you be able to tell which is which? You'll be able to tell them apart if you change the name of one of the files.

To change the name of a file, edit the inverse image "line" containing the name of the file. Let's do this.

Choose one of the "mary" files from the File System menu—since they're identical so far, it doesn't matter which one. Wait for the File System to display the file name in inverse image, leaving the cursor at the end of the name.

Now edit the file name to be whatever you want, using the INSERT, DELETE, and LOCK keys just as you would use them to edit a line in the file.

When you have finished, press ↓ to make the File System save the edited file name. Then press CLEAR. Watch the file names on the File System menu; the file you chose now appears with its edited name.

## Delete a File

To delete a file, delete the entire line containing its name. For practice, let's delete the file you just renamed.

Choose the file from the File System menu. Wait for the File System to display the file name in inverse image. Now press DELETE↑. The File System

returns to the File System menu. Look at the file names on the menu; the file you chose is no longer there.

## Print a File

Procedures for using HHC peripherals are discussed fully in Chapter 9, but printing a file is such a common operation that we'll describe it briefly here.

There are two ways to print a file. You can use the HHC's Micro Printer, Mini Printer, or Plotter. Or you can use a standard computer printer connected to the HHC's RS-232C Interface.

If you are using an HHC printer or plotter, you need only load the device with paper and plug it into the HHC's bus socket. If you are using a standard printer with the RS-232C Interface, you must also wire a cable to connect the Interface to the printer and match several configuration parameters on the Interface to those of the printer. Consult the instructions for the printer and the Interface's device control capsule, or ask an experienced person or the dealer who sold the equipment to you for assistance.

When you are ready to use the printer, observe the following procedures:

**Step 1.** Connect the printer (via the RS-232C Interface, if any) to the HHC *before* selecting the File System from the main menu. Always turn the HHC off while connecting or disconnecting any peripheral.

**Step 2.** Select the File System, then select the file from the File System menu.

**Step 3.** Press the I/O key. Now the HHC will display an *I/O menu*, including an item similar to the following item:

```
1=PRINTER OUT, OFF, SLOT=0
```

Pressing 1 selects the printer and changes the word "OFF" to "ON" in the menu. If you are using an RS-232C Interface, the HHC displays

```
1=RS-232C OUT, OFF, SLOT=0
```

```
2=RS-232C IN, OFF, SLOT=0
```

Pressing 1 and 2 selects the RS-232C Interface and changes "OFF" to "ON".



Press the I/O key again. This returns you to the File System.

Press LOCK ↓. This causes the File System to display each file line from the first to the last. As each line is displayed, it is also printed.

To interrupt the display and printout, press any key. If you don't press a key, the File System displays and prints the entire file.

To turn the printer off, press the I/O key, press the key for the menu number that corresponds to the printer, and press the I/O key again.

## FREE SPACE FOR FILE STORAGE

To find out how much free space remains in intrinsic RAM, press the I/O key. Again, you'll see the following I/O menu:

1=INT RAM, 2223 FREE

This menu item says, "There are 2223 characters of free space in intrinsic RAM." This free space can be used to add new files or to expand any of the files you have already created. Certain operating system functions also use this space.

To resume what you were doing, press the I/O key again.

## The I/O Menu

Unless an application program disables the I/O menu, you can look at it at any time. If you look at the I/O menu while you are running a program, the program will simply continue running—performing as though nothing has happened when you leave the I/O menu by pressing the I/O key a second time. While you are looking at the I/O menu, however, the program is "asleep." This can be undesirable if, for example, your Telecomputing program should be watching a modem for incoming data.

Most programs and peripheral devices take over some part of intrinsic RAM while they are running in order to hold internal data. For example, if you look at the I/O menu while a program is running, the amount of free room is reduced by whatever amount the program has taken. If you look at the I/O menu while no program is running, expect the amount of free room to be somewhat larger.

## When Free Space Is Exhausted

When the free space in intrinsic RAM is exhausted, the HHC presents you with the message

NO ROOM

or

NO ROOM

DELETE FILES

You must delete some files from intrinsic RAM to free up enough room to continue your work. Press CLEAR to return to the primary menu and select the File System. Some programs display the "NO ROOM, DELETE FILES" message and then display files to select for deletion. (When the Programmable Memory Peripheral is plugged into the HHC, the DELETE FILE option does not always work; Chapter 6 describes this peripheral in detail.)

## FILE TYPES

Every HHC file has a *file type* that is stored in RAM with the file but is invisible to you when you examine the file (for example, with the File System editor).

File type indicates file contents and determines what operations the HHC will perform on the file.

A file created with the File System editor has the file type *text*. It consists of some number of lines, each of which may contain from 0 to 255 characters. All text files can be edited by the File System editor. If you try to edit non-text files, the File System beeps and displays the message "CAN'T EDIT".

HHC non-text files are *binary* files. One type of binary file, an *executable SNAP* file, contains a runnable SNAP program. All executable SNAP files appear in the secondary menu presented by the RUN SNAP PROGRAMS

selection from the primary menu (this is described fully in Chapter 7). Programs also use binary files to store "private" information. For example, the Telecomputing program series, which allows you to connect your HHC to other computers through the modem, stores information in binary files used for configuring the modems so that it can communicate with your computer.

Other types of binary files you may encounter are

- Microsoft BASIC program files
- SnapBASIC program files
- SnapFORTH dictionary files (files produced from SNAP programs by the SnapFORTH compiler; they contain the compiled program plus additional information that is useful for debugging).

In addition to its regular file type, any file may be *invisible*. An invisible file does not appear in the file menu presented by the File System editor (or in any other menu of files). Files that programs use to store "private" information are often invisible. For example, the files that the Telecomputing programs use to store their configuration parameters are invisible.

# 4

## The Calculator

**T**he Calculator program makes the HHC operate much like a four-function pocket calculator because it computes the four elementary arithmetic functions and percentages and stores an intermediate result. Certain HHC capsules provide much more powerful calculating facilities than the HHC's intrinsic calculator. These capsules are described in Chapter 7.

To start the Calculator, choose option 1 from the HHC's primary menu. On the LCD, the Calculator displays a blinking cursor over the number 0.

### THE KEYBOARD

The Calculator lets you use the keys in the top row to enter numbers and the keys in the second row to enter such operations as +, ÷, and =. The fourth row's . key, shown in Figure 4-1 along with the first and second row keys, is used as a decimal point.

The second row keys that produce the characters Q, W, E, R, and so on, also have distinct Calculator functions. On each one of these keys appears a symbol—from %, CM, RM, and M+ to =. You do not need to press the 2nd SFT key to type these characters into the Calculator. In fact, you should not use the SHIFT or 2nd SFT keys with the Calculator at all.



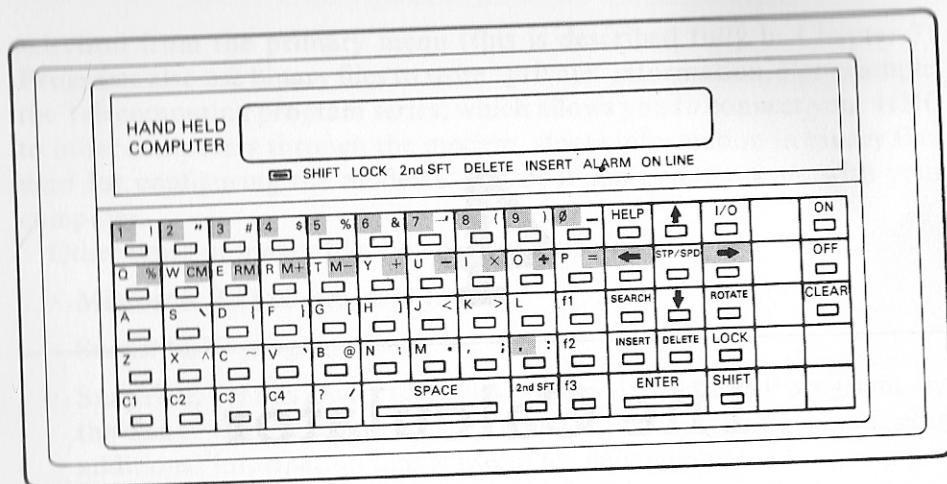


Figure 4-1. Valid keys for the Calculator

If you press a key that the Calculator does not use, the program will just beep. The Calculator will respond in the same manner if you press a valid key at the wrong time; for example, you'll hear a beep if you press + before entering a number.

## SIMPLE ARITHMETIC

Press the four keys on the HHC's keyboard required to display the following operation:

2+2=

As you enter "2 + 2," the HHC displays the numbers and the + sign. The Calculator, you'll notice, displays both numbers and the + simultaneously. A pocket calculator, on the other hand, would display one number at a time.

When pressed, = replaces "2+2" with "4"—the answer.

The Calculator clears itself when you enter the first digit of a new number. If you want to clear the display, however, you can do so by pressing the CLEAR key once.

Subtraction, multiplication, and division function like addition. Just enter the appropriate sign between the two numbers you want to calculate.

At this point, you should try entering some arithmetic problems in order to become familiar with the Calculator.

## Correcting Errors

Most pocket calculators provide only one way to correct a keying error: You must clear the number you are entering and enter it again. The HHC offers an easier way: Just press the ← key. After the cursor moves left one space, you correct the mistake and proceed.

You can press the ← and → keys to correct errors buried within the information you have typed, just as you did while using the File System. The Calculator does not support the INSERT, DELETE, and LOCK keys, however.

Unlike other Calculator operations, = acts on the entire contents of the LCD, no matter where the cursor happens to be. Suppose, for example, you enter the expression

12.5+5.11

But then you change your mind. What you really want to compute is  $12.5 \times 50$ . Here's what you do. Press ← five times to get

12.55.11

Then press ×, →, and 0 to get

12.5×5011

Now, if you press =, the Calculator does not multiply  $12.5 \times 50$ , choosing to ignore the characters under and after the cursor; instead, it will multiply  $12.5 \times 5011$ . To make the Calculator multiply  $12.5 \times 50$ , you must press . and 0 to get

12.5×50.0

and then press =.

## Sequences of Operations

To find the sum of a sequence of three or more numbers, type the first two numbers separated by a + sign. Enter, for example,

2+3

and then press + again. The Calculator displays the sum of the two numbers you have entered and the second + sign, so that you get

5+

When you enter the third number, the Calculator displays

5+4

When you press + again, the Calculator displays the new sum and the next + sign, so that you get

9+

and so forth. Compute the final sum by pressing =.

You can perform any sequence of operations the same way. Try entering the following sequence of operations:

2+3×4

In conventional arithmetic the result would be 14, because multiplication takes precedence over addition. In other words, you would get the following:  $3 \times 4 = 12$  and  $2 + 12 = 14$ . On the HHC, however, entering

2+3×

makes the Calculator display

5×

and then entering 4 and = makes it display

20

The Calculator performs sequences of operations in strict left-to-right order because it performs each operation as soon as the two numbers are entered.

## Ranges and Rounding

The Calculator can operate on numbers as long as ten digits. Internally, however, the Calculator stores numbers that are 13 digits long. Thus, you can perform long sequences of operations and be confident that your final result will be accurate to ten digits.

When the Calculator displays a number that cannot be represented exactly in ten digits, it rounds the display up or down to the closest ten-digit number. If, for example, you multiply

3.333333333×33

the HHC's internal 13-digit result is 109.999999890. This amount rounds up to the ten-digit result 110.000000, and displays

110

But subtract 109 from the product, and you will see

.99999989

Though it has not displayed them, the Calculator retained all 13 digits of the product and used them in the subtraction.

## Percentages

The Calculator can also compute percentages.

Suppose your company's sales were \$150,000 last year. If this year's sales are 120% of last year's, what will they be? To answer that question, enter

150000×120%

When you press %, the Calculator computes the result and displays

180000



You would get the same result if you entered

150000×1.20=

and that makes sense, because 120% of 150,000 is the same as  $150,000 \times 1.20$ .

The % key works the same way with division as with multiplication. For example, let's reverse the multiplication problem. If your sales are \$180,000 this year and if they are 120% of last year's sales, what were your sales last year? When you enter

180000÷120%

the Calculator displays

150000

And you will arrive at that same correct answer when you enter

180000÷1.20=

The % key works a little differently with addition and subtraction. It increases or decreases a quantity by a given percentage. This approach is equivalent to adding a percentage of the quantity to itself or subtracting a percentage of the quantity from itself and is useful for computing discounted or marked-up prices, sale price plus tax, and so forth.

If, for example, the sales tax rate is 6%, what is the price plus tax on a sale of \$57.50?

To find the answer, enter

57.50+6%

and the Calculator will display the following answer:

60.95

To take a 10% discount of \$57.50, enter

57.50-10%

and the Calculator will display

51.75

## MEMORY OPERATIONS

In an area called *memory*, the Calculator stores an intermediate result while calculating the next operation. The HHC keyboard keys CM, RM, M+, and M- manipulate the Calculator's memory. The Calculator's memory is distinct from the HHC's memory.

The HHC's memory (RAM and ROM) is all the area that stores the HHC programs and data. The Calculator's memory is an area set aside to hold one number.

To use the Calculator's memory feature, you must remember the following simple rules:

**Rule 1:** The M+ (add to memory) key adds the last number on the LCD to memory. The M- (subtract from memory) key subtracts the last number on the LCD from memory. Neither key affects the contents of the LCD.

**Rule 2:** The RM (restore memory) key replaces the last number on the LCD with the contents of memory. This key does not affect the contents of memory.

**Rule 3:** The CM (clear memory) key sets memory to zero. The CM key is the *only* thing that sets memory to zero. The Calculator "remembers" the value stored in memory when you turn the HHC off and back on; when you reset the Calculator by pressing the CLEAR key once; and even when you return to the primary menu and then reselect the Calculator.

Let's see how these rules apply in practice.

Suppose you want to calculate the average speed of a car during a crosstown trip. At the start of the trip, the time is 10:18 and the car's odometer reads 112.0 miles. At the end of the trip, the time is 10:45 and the odometer reads 125.5 miles.

The average speed is equal to the distance traveled divided by the time elapsed, or

$$\frac{125.5 - 112.0}{10:45 - 10:18}$$



First, let's compute the elapsed time. Since the hours are the same, we don't have to do any conversion of hours to minutes; we can just subtract

45-18=

to get the elapsed time, which is 27 (minutes).

Before computing the distance traveled, clear memory by pressing the CM key. Now add the intermediate result to memory by pressing the M+ key. This operation stores 27, the intermediate result, in memory.

Now you can compute the distance traveled by entering

125.5-112.0=

and you'll get 13.5 (miles).

Finally, you'll have to divide 13.5 by the quantity in memory, which is the elapsed time. To do this, press the ÷ key. Now press the RM (restore memory) key, and the LCD will display

13.5÷27

The Calculator has restored 27, the contents of memory, after the ÷.

Now you can press = and get your result, which is 0.5 (0.5 miles/minute, or 30 miles/hour).

## Restore Memory

Let's repeat the demonstration above but with one difference. Before pressing RM, you'll enter a second number.

Press CLEAR once and then press the following keys:

45-18=CM M+ 125.5-112.0÷6 RM=

In this case you might expect the Calculator to divide 13.5 (the difference between 125.5 and 112.0) by 627 (the 6 you pressed and the 27 restored by RM). Instead, it divides 13.5 by 27, just as the first demonstration did. The Calculator does this because the RM key *replaces* the last number of the display line with the contents of memory. (Compare this with Rule 2 mentioned earlier.)

How does this rule apply to the first demonstration, where you pressed RM immediately after ÷?

After you entered ÷, the Calculator decided there was a number after the ÷, even though you had entered no numeric digits. Then the Calculator replaced that zero-digit "number" with the contents of memory.

To demonstrate this principle again, press CLEAR and enter

13.5÷8

Now, instead of pressing the = key, press RM. The RM key replaces the last number on the display with the contents of memory, giving you

13.5÷27

## Clear Memory

In the average speed problem mentioned earlier, you cleared memory before adding the elapsed time to it. This was an essential step, even though you were using memory for the first time since selecting the Calculator from the primary menu. Remember, memory isn't cleared when the display is cleared. The only way to clear memory is to press the CM key.

## Store a Constant

When you are performing a series of simple calculations, you can use memory to hold a constant value. This saves you the trouble of entering the constant over and over.

Suppose, for example, you want to convert a list of volumes from gallons to liters. Since there are approximately 3.785306 liters in a gallon, you must divide liters by 3.785306 to perform each conversion.

You can set up the Calculator for this conversion by storing in memory

CM 3.785306 M+ CLEAR

What is the gallon equivalent of 15 liters? To find the answer, enter

15÷RM

The answer is 3.962691523 gallons.



What is the gallon equivalent of 22 liters? You should enter

22÷RM

to get the answer, 5.811947568 gallons.

Each time you use the RM key in this calculation, you save seven key-strokes. You also reduce potential keying errors.

### Multiple Adds

You can use the Calculator's memory to accumulate the sum of a series of intermediate calculations.

Suppose, for example, you want to calculate the area of an office containing rectangular rooms that are  $10 \times 13$  feet,  $11 \times 9$  feet,  $10 \times 10.5$  feet, and  $12 \times 21$  feet.

Clear memory, compute the area of the first room, and add the answer to memory.

CM 10×13=M+

Compute the areas of the second, third, and fourth rooms and add the answers to memory.

11×9=M+

10×10.5=M+

12×21=M+

Now restore the sum from memory with RM, and you have the result.

Figure 4-2 summarizes the rules governing the Calculator's memory operations.

### RETURN TO MAIN MENU

To return from the Calculator to the primary menu, press the CLEAR key twice in a row. Remember, *one* CLEAR returns the Calculator to its initial state (except that memory is not zeroed); *two* CLEARS return the HHC to the primary menu.

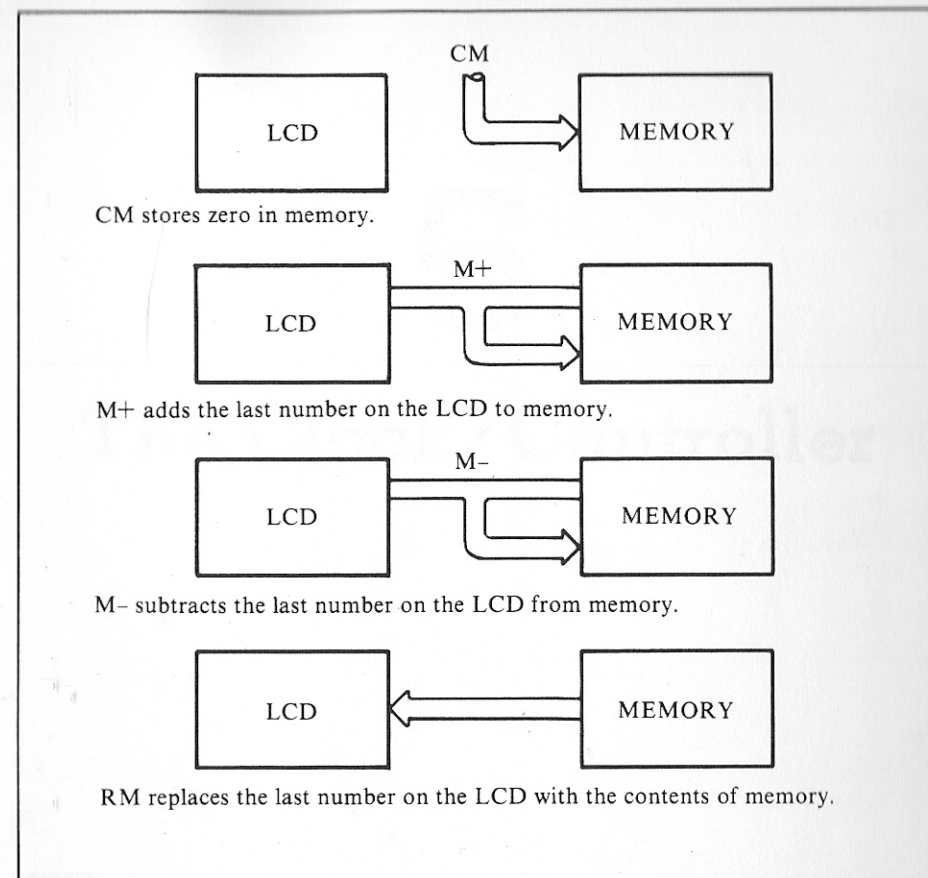


Figure 4-2. Rules governing the Calculator's use of memory

# 5

## The Clock/Controller

**T**he Clock/Controller keeps time with a highly accurate internal clock even when the HHC is turned off. The Clock/Controller lets you

- Display the current date and time
- Set the current date and time
- Set an *alarm* that will “ring” at a specified time, even if your HHC is turned off
- Review and acknowledge an alarm that is “ringing”
- Review all the alarms that are currently set
- Cancel any alarm before it “rings.”

You can use the Clock/Controller as an electronic datebook to keep track of all your appointments and tasks, friends’ birthdays, and other noteworthy events.



## ENTER THE CLOCK/CONTROLLER

To enter the Clock/Controller, choose option 2 from the HHC's primary menu. A secondary menu will display

1=SET ALARM

2=REVIEW

3=ACKNOWLEDGE

4=TIME

5=SET TIME

## DATE AND TIME

If your HHC is new, its internal clock-calendar probably has the wrong date and time. To set the current date and time, choose SET TIME, option 5, from the Clock/Controller's menu.

The Clock/Controller displays the date and time in the following format:

SAT 10:17:16#10/16/82

According to this display, the time is 17 minutes, 16 seconds after 10 in the morning, and the date is Saturday, October 16, 1982.

To change the date and time, enter corrections in the same format, revising only the parts you want to change and using the ← and → keys to skip over the correct parts.

The cursor skips over such constant parts of the display as the colons in the time and the "M" in "AM" or "PM." Since the Clock/Controller automatically computes the correct day when you set the date, the cursor also skips over the day of week.

Find the telephone company's "time of day" number or any other accurate source for the current time. Set the Clock/Controller's time of day a few seconds in the future. Wait until the actual time matches the time you have set, and then press the ENTER key. At this point the Clock/Controller sets its internal clock and returns to its menu.

As long as the HHC's batteries are charged, the clock—once set—should gain or lose no more than a second per day.

If you wish to return to the secondary menu without changing the date or time, press CLEAR.

## Viewing the Time

To view the current time, choose TIME, option 4, from the Clock/Controller's menu. The Clock/Controller displays the time in the following format:

SAT 10:17:16# OCT 16 1982

The Clock/Controller increments the time, second by second, as you watch it.

To return to the Clock/Controller's menu, press CLEAR.

## CLOCK/CONTROLLER ALARM SYSTEM

To set a Clock/Controller alarm, choose SET ALARM, option 1, from the Clock/Controller's menu. The Clock/Controller displays

SET ALARM, TYPE ENTER

and then displays the current date and time.

Set the date and time when you want the alarm to "ring," and then press ENTER. At this point the date and time disappear, and the Clock/Controller displays

MESSAGE :

Now you can type a message as long as 26 characters.

When you have finished typing your message, press ENTER again. The Clock/Controller plays a musical phrase, sets the alarm, and returns to the Clock/Controller menu.

If you wish to return to the Clock/Controller menu without setting an alarm, press CLEAR.

## Reviewing Alarms

To review the alarms that are set, choose REVIEW, option 2, from the Clock/Controller's menu. The Clock/Controller displays

REVIEW, USE ARROWS

and then displays the message associated with the most recently entered alarm.

To see when this alarm will ring, press the ← key. To see the message again, press the → key.

To see the message associated with the alarm set just before the most recently entered alarm, press the ↓ key. To see the corresponding date and time, press the ← key.

You can continue reviewing the alarms by pressing ↓. When you pass the last alarm in the HHC's memory, the Clock/Controller beeps instead of displaying another alarm message.

You can return to a more recently entered alarm at any time by pressing the ↑ key. When you reach the most recently entered alarm, pressing ↑ again makes the Clock/Controller beep. Figure 5-1 summarizes cursor control for reviewing and acknowledging alarms.

To return to the Clock/Controller's menu, press CLEAR.

## Deleting Alarms

To delete an alarm before it starts "ringing," display the alarm's message or its date and time. After you press the DELETE key, the Clock/Controller displays the message

DELETED

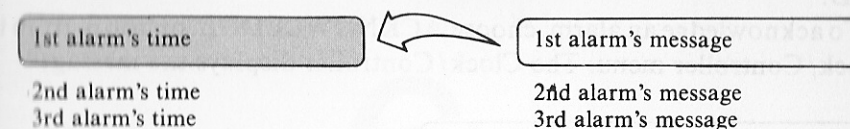
and waits for you to press a key.

After you press ↓, ↑, or CLEAR, the Clock/Controller moves to the next ("↓") or previous ("↑") message or returns to the secondary menu (CLEAR).

## When an Alarm Rings

When an alarm "rings," the Clock/Controller plays a short musical phrase—even if another program is running or the HHC is turned off. Until you acknowledge the alarm, the HHC will repeat the musical phrase about

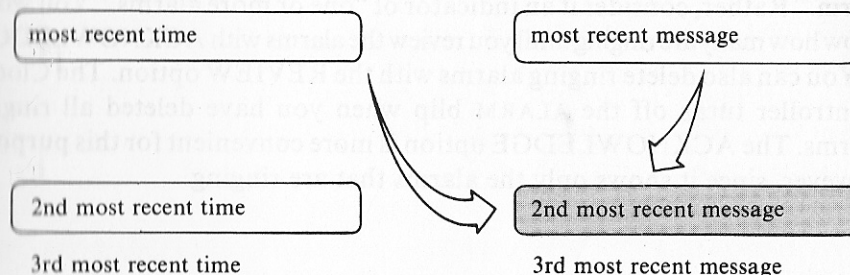
Press ←, the "left arrow", to move from an alarm message to its time.



Press →, the "right arrow", to move from an alarm's time to its message.



Press ↓, the "down arrow", to move from the first alarm time or message to the second alarm's message.



Press ↑, the "up arrow", to move from the second alarm's time or message to the first alarm's message.

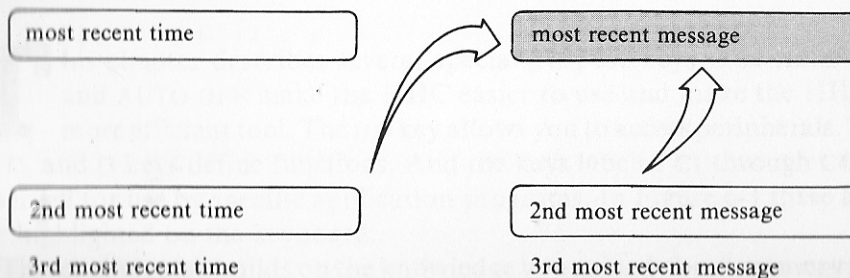


Figure 5-1. Viewing alarm times and messages with cursor control keys (annotated flow chart)



every ten minutes and, when turned on, will display the ALARM blip on the LCD.

To acknowledge an alarm, choose ACKNOWLEDGE, option 3, from the Clock/Controller menu. The Clock/Controller displays the message

ACKNOWLEDGE, USE ARROWS

It then displays the message associated with the most recently set alarm that is ringing.

If more than one alarm is ringing, you can navigate among them with the arrow keys, just as you would in the REVIEW option. When you're ready to acknowledge an alarm, follow the same procedure you used to delete an alarm with REVIEW—press DELETE and then ↑, ↓, or CLEAR.

When two or more alarms are ringing, you must acknowledge all of them to stop the repetition of the musical phrase. Don't think of the phrase as "an alarm." Rather, consider it an indicator of "one or more alarms." You won't know how many are ringing until you review the alarms with ACKNOWLEDGE.

You can also delete ringing alarms with the REVIEW option. The Clock/Controller turns off the ALARM blip when you have deleted all ringing alarms. The ACKNOWLEDGE option is more convenient for this purpose, however, since it shows only the alarms that are ringing.

# 6

## Special-Purpose Keys And Memory Features

**T**his chapter describes several special-purpose keys. The HELP key and AUTO-OFF make the HHC easier to use and make the HHC a more efficient tool. The I/O key allows you to access peripherals. The f1, f2, and f3 keys define functions. And the keys labeled C1 through C4 are reserved for use by specific application programs. In Figure 6-1 these keys are highlighted on the keyboard.

This chapter also builds on the knowledge you gained about memory and file manipulation while reading Chapter 3. The *Programmable Memory Peripheral* adds a new area of RAM to your HHC, and the *persistent memory* feature gives you stop-and-go computing capability.

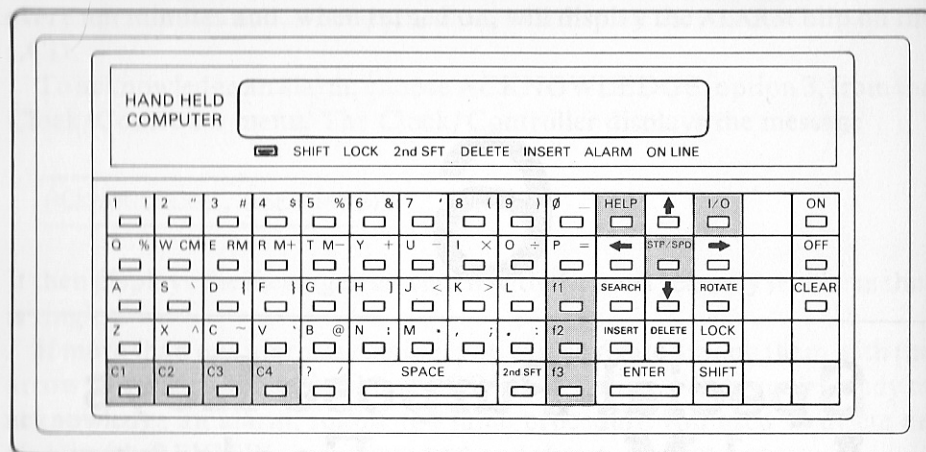


Figure 6-1. Special-purpose keys

## THE HELP KEY

The HELP key reminds you of the functions of other keys on the keyboard. When you press HELP, the HHC displays the message

PRESS KEY FOR DEFINITION

Now press ←, →, I/O, or any other special-purpose key, and the HHC displays a brief message explaining the function of that key.

To get a short explanation of the INSERT key, for example, press HELP and then INSERT, and the HHC will display

INSERT TEXT

You can use the HELP key while you are running most HHC programs (Microsoft BASIC is one of the exceptions) or while you are using the HHC primary menu. However, HELP displays the same text message for each key, no matter what program is running. Many programs use keys like INSERT and DELETE for distinct functions, and HELP does not reflect that fact.

You cannot use the HELP key while you are using the I/O key or the STP/SPD key.

## THE I/O KEY

Chapter 3 describes one application of the I/O key—determining how much free memory remains in intrinsic RAM. The I/O key has two other functions: It switches the HHC's file space from intrinsic RAM to a Programmable Memory Peripheral and back, and it turns peripheral devices on and off. These functions are discussed later in this chapter and in Chapter 7.

You can use the I/O key while you are running most HHC programs, except Microsoft BASIC, or while you are using the HHC primary menu.

## THE STP/SPD KEY

The STP/SPD suspends the HHC's operation while you examine the contents of the LCD and controls the speed of the LCD display.

To suspend the HHC's operation while you examine the LCD, press STP/SPD. To resume the operation, press STP/SPD again.

To control the speed of the LCD display, press STP/SPD and then a number key such as 1, 2, ..., 9, or 0. At this point the HHC immediately displays information at a different rate. The "STP/SPD 1" combination produces the slowest display, "STP/SPD 9" the second fastest, and "STP/SPD 0" the fastest.

Experiment with the STP/SPD key to find a comfortable display speed.

You can use STP/SPD at any time. However, no processing can take place when the HHC is halted by STP/SPD. Such a feature can cause problems. If, for example, you are running a Telecomputing program and another computer sends your halted HHC a message, the HHC won't detect the message's existence.

The display speed setting is not affected by the CLEAR key. Barring accidents—batteries running down, for example—the display speed should remain the same until you change it, or until you turn the HHC off with the ALL-OFF switch.

## THE AUTO-OFF FEATURE

Whenever the HHC is on but not used for approximately ten minutes, the AUTO-OFF feature automatically turns off the HHC.

The AUTO-OFF feature does not work when the STP/SPD key has stopped the HHC, when the I/O menu is being displayed, or when the HELP key has just been pressed. If left on in one of these states, the HHC will remain on until you turn it off or until the batteries run down.



## THE FUNCTION KEYS

Each function key represents a series of as many as 15 keystrokes. Whenever you press one of the function keys, the HHC will respond as if you had entered the series of keystrokes that the function key represents.

Function key definitions are not erased by the CLEAR key.

### Define a Function Key

To define a function key you have to perform the following steps:

**Step 1.** Press the HELP key. At this point the HHC displays the message "PRESS KEY FOR DEFINITION".

**Step 2.** Press the function key you want to define. The HHC displays "DEFINE FUNCTION" and then the current definition (if any) of the key. After the last character of the definition, the HHC leaves a cursor consisting of a nonblinking underscore. Although the definition may be 15 keystrokes long, only the first 14 will be displayed.

**Step 3.** Enter the series of keystrokes this function key will represent. The keystrokes may include any key except ON, OFF, CLEAR, or another function key. It may, for example, include the ENTER key. The I/O, STP/SPD, and HELP keys do not perform their normal functions when they are part of a function definition. The HHC uses inverse-image characters to represent the special keys it can store in a function key definition. Shown in Table 6-1 are the characters displayed when special keys are used in a function definition.

After you enter the last character of the definition, press the function key again. The HHC will complete the function key definition and return to its previous task. You may also press CLEAR to complete the function key definition.

If you enter a function key definition that is 15 characters long, the HHC flashes the message "FULL" on the LCD when you enter the 15th character. Then it completes the function key definition and returns to its previous task.

Since a function key definition may include →, ←, INSERT, or DELETE, you can't use those keys to edit a function definition. If you make a mistake while entering a definition, you must complete the function key definition and then reenter it.

To demonstrate, let's define a function key that chooses the File System from the primary menu, selects the first file (option 3) from the File System menu, and displays the last line of the file.

To begin the definition, press HELP. At this point the HHC displays the message






















PRESS KEY FOR DEFINITION

Press f1 and the HHC displays the current definition of f1. If f1 is undefined, the LCD displays only an underline cursor.

Now press the keys

3 3 LOCK ↓ f1

**Table 6-1.** Special Keys and What They Display

Key	Symbol Displayed	Key	Symbol Displayed
		ROTATE	
		INSERT	
		DELETE	
		ENTER	
HELP		C1	
I/O		C2	
STP/SPD		C3	
SEARCH		C4	
LOCK			

The function key is defined to simulate the keystroke sequence 3 3 LOCK ↓. The f1 ends the definition.

Now, if you press f1 while the HHC is displaying the primary menu

- The first 3 selects the File System.
- The second 3 selects the first file (option 3) from the File System's menu (the usefulness of the definition depends on whether or not this file is always the one you want; since files are added to the end of the menu, your first file will remain in the first position).
- LOCK, which is stored in the function key definition as a separate keystroke, is displayed as "M".
- ↓ moves the display down one line in the file, and the sequence LOCK ↓ moves the display to the end of the file, just as if you entered it directly through the keyboard.

### Display a Function

To display the definition of a function key, press HELP and then the function key. The LCD displays "DEFINE FUNCTION" again and then the definition of the function key.

When you have finished looking at the definition, press CLEAR. If you have pressed no other keys since selecting the function you wanted to display, the HHC will return to its previous task without changing the function definition.

### Erase a Function

To erase a function key definition, press HELP and then press the appropriate function key *twice*. You *cannot* erase a function key definition by pressing HELP, the function key, and then CLEAR.

### Use of Function Keys

You can use a function key at almost any time. The 3 3 LOCK ↓ example showed how to use the function key definition to select a program from the primary menu, select a program from a secondary menu, and then perform a repetitive operation within the program.

Use function keys whenever you need to enter the same sequence of keystrokes repeatedly because they "abbreviate" the sequence, reducing it to a single key. For example, use function keys to store user names and passwords for Telecomputing; keywords for commands in Microsoft

BASIC or SnapBASIC; constant values for a calculator; or words that appear frequently in the File System editor.

## THE PROGRAM FUNCTION KEYS

The four *program function keys* are used in various ways by many application programs. Many applications let you use these keys to enter commands when the main keyboard is being used for typing. When a Microsoft BASIC program is running, for example, the main keyboard is used to type input data and the C1 key is used to interrupt execution of the program. In the COMMUNICATE mode of a Telecomputing program, the main keyboard is used as a computer terminal and the four program function keys are used to enter control characters and perform other special functions.

To the HHC's hardware, the program function keys are just like main keyboard keys: They generate unique input codes that programs interpret in distinct ways. When sent to the LCD, these codes display their representatives—all of the characters shown in Table 6-1. When sent to a peripheral device that does not recognize the HHC's extended character set—for example, a printer connected to the HHC through an RS-232C Interface—the codes produce unpredictable effects. (Peripherals are discussed in detail in Chapter 9.)

## THE PROGRAMMABLE MEMORY PERIPHERAL (PMP)

A Programmable Memory Peripheral (PMP) is a device that contains 4096 (4K), 8192 (8K), or 16,384 (16K) bytes of RAM to augment the HHC's intrinsic RAM. Though not a true peripheral device—a program sees the PMP as an extension of the HHC's memory—it is called a peripheral because it plugs into the HHC's peripheral socket or I/O Adaptor.

### The Peripheral Plug Cap

A plastic cap covers the peripheral plug to protect the PMP's circuitry from damage by static electricity. The cap must remain on the plug whenever the PMP is disconnected from the HHC.

The PMP uses special low-power chips that are particularly sensitive to static electricity.



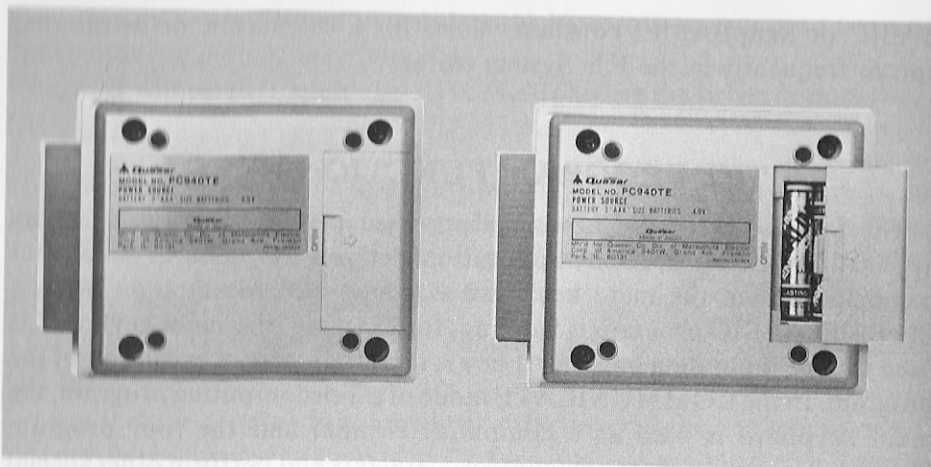


Figure 6-2. PMP battery compartment

## The PMP's Batteries

The 4K and 8K PMP's sliding back panel, shown in Figure 6-2, covers a compartment for three AAA size batteries. These PMPs do not need batteries to operate. When disconnected from the HHC, however, they need batteries to preserve their file storage.

You should equip your 4K or 8K PMP with "long life" alkaline batteries, which last for about a year. When you insert the batteries in the battery compartment, be sure to orient them correctly, placing them as shown in Figure 6-2.

To replace batteries without giving the PMP amnesia, connect it to your HHC. The PMP will draw power from the HHC's batteries while its own batteries are being changed. If you remove the PMP's batteries while it is disconnected from the HHC, all of the PMP's files will be lost.

To prevent the PMP's bus plug from being disconnected or strained while you change the batteries, slide the HHC's rigid plastic sleeve over the back of the HHC and the PMP.

The 16K PMP has built-in lithium batteries with a useful life of about four years. When your PMP's batteries near the end of their life span, take the entire PMP to an HHC service center to have the batteries replaced.

## Intrinsic and Extrinsic File Space

When you connect a PMP to your HHC, you introduce an area of *extrinsic RAM* that is separate from the HHC's intrinsic RAM.

When a PMP is connected, *intrinsic RAM* functions the same as before: It holds data used by the system and by a running application program and stores files. Extrinsic RAM may only be used to store files.

At any time, the HHC has access to files stored in only one RAM area. Called the *current file space*, this area is either intrinsic RAM or in a PMP's extrinsic RAM.

Using the I/O menu, you can switch the current file space from one RAM area to another.

## Connecting the PMP

Let's go through the steps necessary to connect a PMP to your HHC and switch the current file space to the PMP.

You cannot change the current file space while a program is running. If you try, the HHC will terminate your program and return to the primary menu. Always return to the primary menu before you plug in a PMP.

**Caution:** Turn your HHC off before connecting or disconnecting a PMP or any other peripheral. This precaution eliminates any risk of damaging the HHC or the peripheral.

Turn your HHC off and plug the PMP into the HHC's peripheral socket. As you plug in the PMP, the HHC beeps. When the beeping stops, the PMP plug is properly seated.

Slide the plastic sleeve over the the back of the HHC and the PMP to hold them together and protect the peripheral plug from strain.

Turn your HHC on again. Now press the I/O key, and you'll see a display that resembles

```
1=INT RAM, 1665 FREE
```

```
2=EXT RAM, 16374 FREE, SLOT=0
```

This I/O menu is explained in the following statements:

- The words "INT RAM", displayed in inverse characters, identify the current file space as intrinsic RAM. Intrinsic RAM has 1665 characters of free space.
- "EXT RAM" (extrinsic RAM) tells you that one PMP is connected to

the HHC. Extrinsic RAM has 16,374 characters of free space. (This is the amount of free space in a completely empty 16K PMP. A 4K or 8K PMP would show 4086 or 8182 characters—always ten characters less than the PMP's total capacity.)

"SLOT = 0" means that the PMP is plugged directly into the peripheral socket of the HHC.

To switch the current file space to the PMP, choose option 2 from the I/O menu, and the HHC will display the message

EXT RAM, 16374 FREE, SLOT=0

Now the HHC will continue displaying the I/O menu, but this time "EXT RAM" is displayed in inverse characters, which tells you the current file space is now located in the PMP's extrinsic RAM.

Press the I/O key again to return to the primary menu.

You can only change the current file space when no program is running (or when you are willing to cancel the program that is running); but you can use the I/O key at any time to check the amount of free space in each file space, extrinsic and intrinsic.

## Copying a File

To copy a file from one file space to another, switch to the file space you want to copy from. Enter the File System, select COPY FILE (option 2) from the File System menu, and then select the file you want to copy from the COPY FILE menu.

At this point the File System displays the message

SELECT DESTINATION RAM

The File System will display a menu of the file spaces available to accept the copy of the file. You can tell which file space is the current file space because its name is in inverse characters. Choose the file space that you want for the duplicate file the File System creates. The File System then copies the file, displays the message "COPY DONE", and returns to the File System menu.

If there is not enough room to create a copy of the file, the File System displays the message "NO ROOM, DELETE FILES" and does not make the copy.

## A Caution

In the chapter on the File System, you discovered that many programs follow the message "NO ROOM, DELETE FILES" with a menu of files that you can delete. However, the DELETE FILES menu does not always work when you are using a PMP. The DELETE FILES menu only allows you to delete files from the current file space, that is, the PMP. In some cases the space shortage is located in intrinsic RAM, where a program—even a program stored in the PMP—must get space for its internal working storage. You can delete all the files in the PMP, yet fail to provide the space you need in intrinsic RAM.

To solve this problem, first return to the primary menu. After using the I/O key to switch the current file space to intrinsic RAM and the File System to delete one or more files, switch the current file space back to the PMP and resume work.

## Other Uses

Expanding your file storage is just one use for the PMP. Since it has its own batteries, the PMP can preserve a file when disconnected from the HHC. This feature makes the PMP a convenient medium for *back-up storage*. If you want to preserve a copy of a file, duplicate it onto a spare PMP and put the PMP in a safe place.

The PMP is also a convenient vehicle for transporting files. If you want to create a duplicate file on another HHC, you can copy the file onto a PMP, transport the PMP to the second HHC, and copy the file from the PMP to the second HHC.

To create larger files than can fit in the HHC's intrinsic RAM, you should create files in a PMP since it has a larger storage capacity.

Use a PMP to increase the amount of intrinsic RAM that a program can use as temporary storage. If you move a file from intrinsic RAM to extrinsic RAM, programs no longer have to compete with that file for working space in intrinsic RAM.

## THE PERSISTENT MEMORY FEATURE

The HHC can recall what it was last doing even after you turn it off because of its *persistent memory feature*.

Suppose, for example, you are using the Calculator. In the middle of a calculation, you turn the HHC off by pressing the OFF key. After a few



seconds (or days), you turn the HHC on again. Your partially entered calculation reappears on the LCD, exactly as it was displayed before you turned off the HHC. When you finish it, the calculation will be as correct as if you had not turned off the HHC.

Turning the HHC off and on without losing information is not unique to the Calculator. You can do it with any program—intrinsic or not—that runs on the HHC. However, a program that uses a peripheral device sometimes reads or writes a spurious character when the HHC is turned off or on. Peripherals are discussed in detail in Chapter 9.

# 7

## Running Programs On the HHC

**M**uch of the HHC's versatility arises from the wide range of programs available as separate products. This chapter explains how to run programs on the HHC and surveys the programs that are currently available, but it does not describe each program in detail. You will find detailed information on specific programs in the manufacturer's documentation.

### HOW TO RUN HHC PROGRAMS

The HHC runs programs in three basic forms—HHC capsules, device capsules, and SNAP files.

#### HHC Capsules

An *HHC capsule* stores a program on a chip of read-only memory (ROM). The chip is mounted on a special plastic carrier that protects it from damage.

## Using an HHC Capsule

To install an HHC capsule, return your HHC to the primary menu and press the OFF key. Detach it from any peripherals. Now remove the panel on the back of the HHC. With the HHC capsule positioned so that the flat face is down, insert it into any one of the three sockets behind the panel, as shown in Figure 2-3. Now replace the panel.

To use a program, turn the HHC on and watch the program appear as an additional item in the primary menu. If, for example, you have installed HHC capsules for the HHC programs Portaflex and Time/Trac, you'll get

1=CALCULATOR

2=CLOCK/CONTROLLER

3=FILE SYSTEM

4=RUN SNAP PROGRAMS

5=PORTAFLEX

6=TIME/TRAC

Now press the key corresponding to the desired program's menu number.

## Device Capsules

A *device capsule*, which plugs into a socket in a peripheral device, provides the HHC's operating system with special software required to operate the peripheral. It may also contain an application program intended specifically for use with the peripheral. One example, the Telecomputing 2 program, enables you to transfer files back and forth between a modem-equipped HHC and another computer. Telecomputing 2 is a device capsule that plugs into the modem. A *modem* is a peripheral device that can transmit data from one computer to another over ordinary telephone lines. Chapter 9, *Using Peripherals*, discusses modems in detail.

Physically, a device capsule is the same as an HHC capsule. You can plug a device capsule into the back of the HHC or an HHC capsule into a device; no damage occurs, but they simply won't work. Figure 7-1 shows a device capsule.

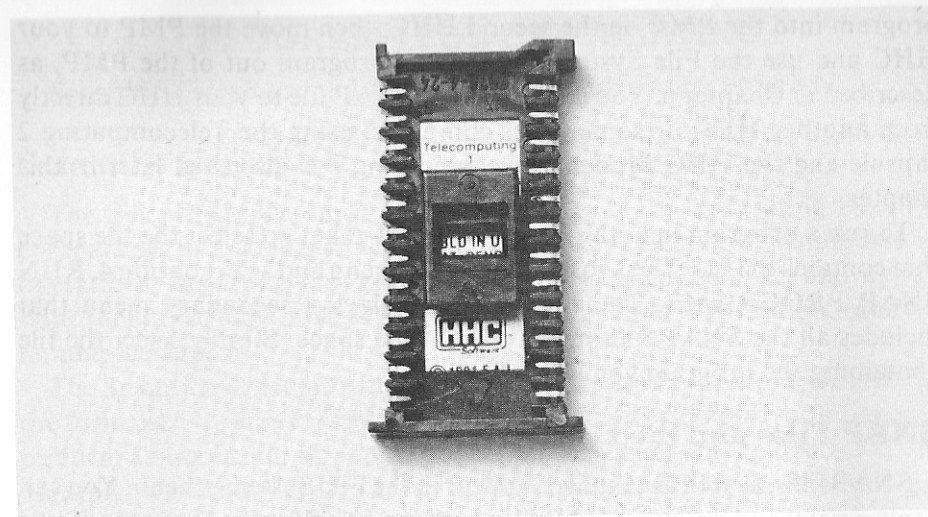


Figure 7-1. The Telecomputing 1 capsule

## Using a Device Capsule

If the device capsule contains an application program, the program's name appears in the primary menu whenever the peripheral is connected to the HHC. You run a device capsule program the same way you run a program in an HHC capsule: Once connected, the program appears on the primary menu for your selection. Device capsule programs follow HHC capsule programs in the list.

## SNAP Files

A *SNAP file* contains an executable program that you write in SnapFORTH to perform a specific task. If your programs are generally used for a short time and then discarded, or if they are constantly used but require frequent updating, they are good candidates for SNAP files.

To create a SNAP file, write a computer program in the SNAP programming language and process it with the SnapFORTH compiler, which is available as an HHC capsule. A SNAP file is always written in SNAP but must be compiled in SnapFORTH. Chapter 8 clarifies this relationship and discusses compilers in detail.

You can also acquire a SNAP file by copying it from a second HHC using a Programmable Memory Peripheral. Use the File System to copy the



program into the PMP on the second HHC; then move the PMP to your HHC and use the File System to copy the program out of the PMP, as described in Chapter 6. You can transfer a SNAP file to your HHC directly from another HHC or some other computer, using the Telecomputing 2 capsule and the HHC Modem. Telecomputing 2 is discussed later in this chapter.

To run a program in a SNAP file, use the I/O key to select the file space that contains the file. Then go to the primary menu and select option 4, RUN SNAP PROGRAMS. The HHC now displays a secondary menu that includes all the SNAP files in the current file space. Simply select the file containing the program you want to run.

### SNAP Files and the File System

SNAP files, as well as text files, appear in the File System's menu. You can copy, delete, and rename SNAP files with the File System. However, if you try to edit a SNAP file, the File System displays the message "CAN'T EDIT" and returns to its menu. The File System doesn't allow you to edit a SNAP file because the file contains non-text data.

## TYPES OF HHC PROGRAMS

Most HHC programs can be divided by function into the following groups: calculating, data entry and editing, communicating, and language processing.

The Scientific Calculator, an example of a program that calculates, has log/exponential, trigonometric, and hyperbolic functions, and many other features of a multifunction scientific calculator.

Portaflex is a program used to enter and edit data. It prompts you for information one field at a time, checking your responses for validity and consistency, and enters them into a customized file.

Some programs let the HHC communicate with other computers. Telecomputing 2 lets you use the HHC as a terminal for communicating with another computer and allows you to transfer files back and forth between the other computer and the HHC.

Programming language processors let you develop your own application programs or run programs that you obtain in human-readable source format from other HHC users.

This chapter surveys the calculating, editing, and communicating programs. Chapter 8 surveys the language processors.

## PROGRAMS FOR CALCULATING

Scientific Calculator and Portabudget are calculating programs.

### Scientific Calculator

The Scientific Calculator capsule gives the HHC calculating facilities comparable to a sophisticated scientific pocket calculator. It supports 41 operations, including the standard arithmetic, trigonometric, and hyperbolic functions, logs, powers and roots, and a normal distribution, random number generator.

The Scientific Calculator accepts input in reverse Polish notation (RPN), the form used by many scientific calculators. In reverse Polish notation you perform an operation by entering two numbers and then the operation. To multiply, for example, you enter  $12\ 23\ \times$  instead of  $12 \times 23$ . Once you get used to this form of notation, you can use it to enter complicated expressions without the parentheses that ordinary notation would require. Figures 7-2 and 7-3 show examples of RPN calculations.

The Scientific Calculator has 12 locations for storing intermediate results.

### Scientific Calculator Functions In Other Programs

A SNAP program can use many of the Scientific Calculator's functions if the Scientific Calculator capsule is plugged into the HHC. The Scientific Calculator capsule is then called a *library capsule*. In effect, the SNAP program can use the library capsule's functions as though they were additions to the HHC's built-in numeric functions.

You enter	This happens	Calculator contains
505 ENTER	enters 505	505
3	enters 3	505, 3
$\times$	multiplies $505 \times 3$	1515
384 ENTER	enters 384	1515, 384
96	enters 96	1515, 384, 96
$\div$	divides $384 \div 96$	1515, 4
+	adds $1515 + 4$	1519

Figure 7-2. RPN: Computing  $505 \times 3 + 384 \div 96$



You enter	This happens	Calculator contains
505 ENTER	enters 505	505
9	enters 9	505, 9
+	adds 505 + 9	514
505 ENTER	enters 505	514, 505
3	enters 3	514, 505, 3
-	subtracts 505 - 3	514, 502
×	multiplies 514 × 502	258028
11	enters 11	258028, 11
+	adds 258028 + 11	258039
219	enters 219	258039, 219
÷	divides 258039 ÷ 219	1178.260274

Figure 7-3. RPN: Computing  $[(505 + 9) \times (505 - 3) + 11] \div 219$

## Portabudget for Personal Budget Management

The Portabudget program helps you manage your personal or family budget. It is currently under development and is expected to be released sometime in early 1983.

Portabudget can keep track of activity in your checking, savings, and credit card accounts, and help you reconcile statements with your records at the end of each month. It can also keep track of your expenses in the various categories that you define, and can keep track of your total monthly expenses in each category so that you will know when your expenses have exceeded your budget.

## PROGRAMS FOR ENTERING AND EDITING DATA

Portaflex and Time/Trac let you enter and edit data.

### Portaflex for Customized Data Entry

Portaflex is a program for interactive data entry. Although its specialized programming language creates data entry applications tailored to your specific needs, you need not be a programmer to use Portaflex.

Put your Portaflex application into a text file, using the File System. Then

run Portaflex and identify the file that contains your application. You can keep several Portaflex applications in different files if you wish.

Figure 7-4 shows a simple Portaflex program.

Taking inventory in a warehouse, a typical Portaflex application, requires recording each inventory item in response to a series of prompts for stock number, quantity on hand, warehouse location, and so forth. The Portaflex program would accumulate this information for you in a text file, with one line representing each item. After completing the inventory, you could process the resulting file with another program on the HHC or transfer it to another computer.

A Portaflex application can perform several kinds of computations and check the data for validity and consistency. If a Portaflex application is supposed to prompt you for a state name abbreviation, the application could check your response for the correct standard abbreviation (NY for New York, IL for Illinois, and so on). If a stock number contains a fixed number of digits, a Portaflex application could check the number of digits you type and ask you to reenter the stock number if you typed too few.

## Time/Trac for Time Accounting and Billing

Time/Trac, which records work activities for time billing, is designed for attorneys, accountants, consultants, and other professionals who bill by the hour.

For each activity, Time/Trac can record start and stop times, client names, and a short descriptive comment. The accuracy of the start and stop times depends on the accuracy of the HHC's internal clock.

Time/Trac records can be *up-loaded* to a larger computer for further processing by a comprehensive billing system. With Time/Trac you can also review your activities and generate several kinds of printed reports with the HHC and a printer.

## PROGRAMS FOR COMMUNICATING WITH OTHER COMPUTERS

Telecomputing 1 and Telecomputing 2 are device capsules for the Modem. Each capsule contains an application program that allows your Modem-equipped HHC to communicate with another computer. With these capsules, you can use your HHC as a terminal to control another computer.



```

*Example inventory application.

*Define format of data collection file.
record item
field stocknum 6
field quantity 3
field location 3

*Jump to this label to begin collecting a new data item.
rlabel get-item item

*Stock number.
prompt Stock #?
format xxxxxxxxxxxxxxxnnnn
alphas
numerics
query
save stocknum

*Quantity in stock.
prompt Quantity in stock?
format xxxxxxxxxxxxxxxxxxxnnnn
numerics
query
save quantity

*Location.
prompt Location?
format xxxxxxxxxxxnnnn
alphas
numerics
query
save location

*Write data item to file & jump to collect next item.
write
jump get-item

```

**Figure 7-4.** A simple Portaflex application

## Telecomputing 1

Telecomputing 1 presents a menu with the options COMMUNICATE and CONFIGURE.

COMMUNICATE lets your HHC communicate with another computer that has a modem. Anything you type into the HHC is transmitted through its Modem and over the telephone to the other computer. Anything received

over the telephone by the HHC's Modem is displayed on the LCD and on a Printer, TV Adaptor, or other attached output peripheral.

The "review" feature of Telecomputing 1's COMMUNICATE mode lets you recall information that has scrolled off the LCD. Thus, on the 26-character LCD, you get to see more than the last 26 characters of your Telecomputing session.

CONFIGURE lets you set a number of options that control the way the Modem sends and receives data. To communicate with each other, two RS-232C devices—the HHC's Modem and another computer—must be configured with matching options.

## Telecomputing 2

Telecomputing 2 has all of the features of Telecomputing 1 except "review" and also lets you transfer files between the HHC's File System and another computer.

To transfer a file from the HHC to another computer, find, on the other computer, a program that will record typed data in a file. Most text editors will do the job. Then use the HHC as a terminal and start the program. Finally, command Telecomputing 2 to transmit the file. While Telecomputing 2 sends the data, the program on the other computer will record the file just as if you were typing information on the HHC's keyboard.

To transfer a file from another computer to the HHC, prepare Telecomputing 2 to receive a file. Then find, on the other computer, a program that will send and display the contents on the LCD. Telecomputing 2 records the data while the other computer sends it.

Telecomputing 2 has a special mode of operation that lets you transfer a binary file. For each byte in a binary file, Telecomputing 2 transmits two bytes. This feature enables the program to transfer all 256 values that a byte in a binary file may have while transmitting only the 100 or so printable characters that can be sent reliably over a modem. At the receiving end, you must set up Telecomputing 2 or an equivalent program to combine each pair of transmitted bytes into one file byte.

## Future Telecomputing Products

A more advanced Telecomputing program is under development for release sometime in early 1983.

As yet unnamed, this program will have all the important capabilities of Telecomputing 2. It will also detect any data errors that occur during transmission (because of a noisy telephone line, for example) and retransmit

the part of a file that contains the error.

This new program will let you record a sequence of keystrokes in an *auto file*. You can then send the sequence to another computer on demand. If an auto file is divided into a series of *messages*, these messages may be sent one at a time by pressing the ENTER key. Auto files are useful for logging on to a time sharing system and other procedures that you must perform over and over.

Because this new program will encode data in a special way so as to detect and correct transmission errors, you will need two versions. One program will run on the HHC and one will run on the other computer.

### RS-232C Capsule for Communications

The RS-232C capsule is an HHC device capsule that lets you configure communications options and use the RS-232C Interface. The RS-232C lets you communicate with another computer over a data cable, much as Telecomputing 1 lets you communicate with another computer over the telephone system.

Capsules designed for the Modem may also be used in the RS-232C Interface, since the circuitry in the two devices is very similar. The future Telecomputing program capsule will be particularly useful in the RS-232C Interface, since it will allow you to perform error-free file transfers between two computers at higher data rates than the Modem can handle.

# 8

## Programming Languages For the HHC

**C**ommercially available programs can perform many different tasks. If you have a unique task that no existing program addresses, you may have to either write a new computer program or change an existing one to meet your needs.

If you want to learn to write computer programs, this chapter will identify the programming languages that are available on the HHC and will tell which tasks each language is appropriate for. If you are not interested in learning to write programs, you may skip this chapter.

### COMPUTER PROGRAMMING

The computer and the nonprogrammable calculator have different functions. A calculator stores only the data that it uses in calculations. A computer, on the other hand, stores the instructions that define the calculating process itself. These instructions are called a *computer program*.

#### Machine Language

The most straightforward way to create a computer program is to use the computer's own *machine language* to write the actual instructions that will be executed.



Ironically, the most straightforward way is also the most time-consuming and susceptible to error. Machine language is a difficult medium for people to use to express computing processes. Although the instructions may be quite simple, using simple instructions to describe a process can be very tedious.

Furthermore, since a machine language program is composed of numbers only, people find it difficult to read. Machine language instructions that perform addition might, for example, look like the following:

```
25100 31101 20100
```

### Assembly Language

*Assembly language*, a form of notation for machine language instructions, assigns each instruction an alphabetic name. This name is easier for us to remember than the machine language series of numbers that represent the same instruction.

An assembly language program section that adds two numbers and is equivalent to the preceding machine language example might appear as follows:

```
LDA X
ADC Y
STA X
```

To run assembly language programs, you need an *assembler*. This is a program that translates a *source program* written in assembly language into an *object program* written in machine language, allowing the computer to read the instructions.

### High-Level Languages

A *high-level language* is a more sophisticated kind of programming tool. It expresses instructions in a notation that is closer to English.

The source program written in a high-level language is translated into an object program written in machine language by another program called a *compiler*. A compiler performs the same basic function as an assembler; but the compiler is more sophisticated, since a high-level language differs more from machine language than assembly language.

If you substitute a high-level language for the preceding assembly language program you might get

```
X=X+Y
```

As a rule, each kind of microcomputer has its own machine language and its own assembly language, depending on the microprocessor used. But a high-level language is invented to solve a certain type of problem and is not necessarily tied to a single type of computer. The same high-level language program may be translated by compilers running on many types of computers. This capability gives high-level languages an additional advantage over assembly languages—they are *portable*. Portability is the freedom to move programs from one computer type to another without rewriting them.

### Interpreters

Some compilers do not directly translate a source program into machine language. They employ an extra step in the process, translating the instructions into a special internal language. This internal language may be considered the machine language of an imaginary *pseudo-machine*. A compiler can more easily translate a high-level language into this pseudo-machine language than into the actual machine language.

When the pseudo-machine language program is run, an interpreter program examines and performs each pseudo-machine instruction as if the pseudo-machine were real. Called *interpretive execution*, this method of executing a program simplifies the task of a compiler program. Because one pseudo-machine language instruction may replace several machine language instructions, interpretive execution compacts the code and makes the program run more efficiently.

Unfortunately, interpretive execution adds a step when running a program. Instead of being executed directly by the computer, each pseudo-machine instruction must be “executed” by the interpreter, which in turn is executed by the computer. Depending on the pseudo-machine language used, interpretive execution is anywhere from two to hundreds of times slower than direct machine execution.

### SnapFORTH

SnapFORTH is a compiler and execution support package for the programming language SNAP. A variant of FORTH, SNAP is the most powerful and flexible language available on the HHC. It gives you full access to HHC operating system facilities such as file I/O, peripheral I/O, and file space switching. Much of the HHC’s operating system and intrinsic application programs are written in SNAP.

```

: on.lcd (S N --- [N] B )
  dup dbuf1 <      \ N B(N<DBUFL)
  IF              \ IF N not too big; N
    dup 0<        \ N B(N<0)
    IF drop 0     \ IF N too small, B=0
    ELSE 1        \ ELSE N is OK; N B=1
    ENDIF
  ELSE drop 0     \ ELSE N too big; B=0
  ENDIF ;

```

(P If N is a valid LCD character position, return N & B=TRUE.  
Else return B=FALSE.)

Figure 8-1. A portion of a SnapFORTH program

SNAP offers you most of the power normally associated with assembly language programming, yet like a high-level language it relieves you of many of the burdensome details of assembly language programming.

SNAP programs are compiled into an interpretive code. The SNAP interpreter is built into the HHC's operating system. More compact—but slower—than an equivalent program written in assembly language, a compiled SNAP program is more efficient than most high-level languages.

The SnapFORTH compiler has a built-in assembler, so you can write a program partly in SNAP for ease of programming and partly in assembly language for ultimate efficiency or fine program control.

Figure 8-1 shows a portion of a program written in SNAP. In this sample, the shaded portion represents the actual source code; comments are shown in unshaded type.

## When to Use SNAP

SNAP is an appropriate language for programs that are complex and more than several pages long; that must employ I/O concurrently on several peripherals, use library capsules, or make other heavy demands of the HHC's features; or that must run as efficiently as possible.

SNAP is inappropriate for programs that will be used a few times and then discarded; that must be developed rapidly, or developed or modified by

inexperienced programmers; or that must be transportable to computers other than the HHC.

## How to Use SnapFORTH

If you're planning to write a lengthy SnapFORTH program, you should consider a *host computer* that has a full-size keyboard, a multi-line display, and disk storage. The host's facilities make it much easier to develop large programs.

When you write a program on a host computer, you must *download* the source code to the HHC (now called a *target computer*) through the RS-232C Interface (Figure 8-2). The SnapFORTH capsule on the HHC compiles the program and stores the object code in a file, just as if the source code were read from a file.

Once you have compiled a SnapFORTH program on the HHC, check it for errors or *debug* it by typing SNAP code on the HHC keyboard. Each line of code that you type is executed as soon as you press ENTER. You can set up data for a program test, execute the program or any part of it, and study the results of the test.

After running a series of tests, you can recompile part of your program and continue testing immediately. In other languages you would have to recompile the entire program from the start.

When your program is complete, you can make it into a SNAP file, which you can then execute through the RUN SNAP PROGRAMS option on the HHC's primary menu.

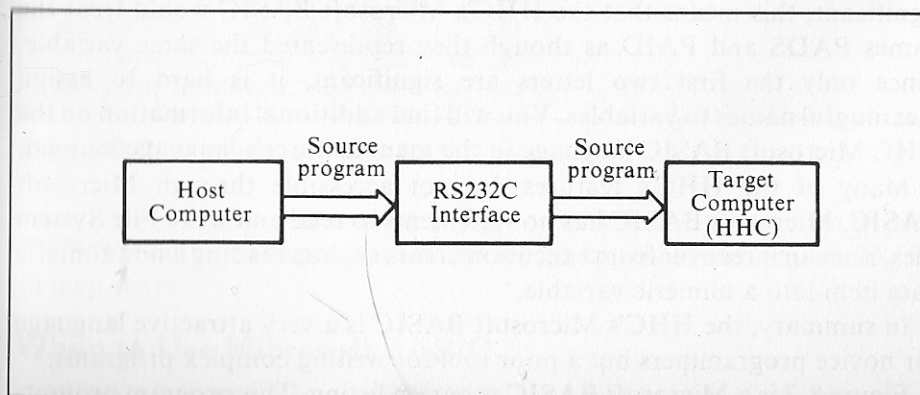


Figure 8-2. Downloading a source program from a host computer to a target computer



## MICROSOFT BASIC

Microsoft BASIC, a product of Microsoft, Inc. of Bellevue, Washington, is a dialect of the programming language BASIC. The most widely used programming language on small computers, BASIC features simple syntax, a highly interactive nature that makes it easy to learn, and the ability to perform many kinds of tasks, including numeric computation and character processing.

Since Microsoft BASIC is available on many computers, computer programs treating a variety of subjects have been developed in this dialect and can be run on the HHC with little modification. Most of these programs are part of the public domain or are available for a small charge (\$5 to \$50).

Programs developed in Microsoft BASIC can be transported easily from the HHC to many other computers. Applesoft, one popular implementation of Microsoft BASIC, is distributed with Apple II and Apple II Plus computers. In fact, most application programs that run on the Apple II or Apple II Plus are written in Applesoft BASIC and can be modified to run on the HHC.

In addition, many books and courses of instruction for beginning programmers teach Microsoft BASIC as a first language. This minimizes the difficulty you might encounter because of small differences between the BASIC you are using and the BASIC your book describes.

The bibliography in Appendix F contains sources of BASIC programs that will run on the HHC with little modification.

The version of Microsoft BASIC that is available on the HHC has several limitations. Only the first two characters of a program variable name are significant; this means that the HHC's Microsoft BASIC would treat the names PADS and PAID as though they represented the same variable. Since only the first two letters are significant, it is hard to assign meaningful names to variables. You will find additional information on the HHC Microsoft BASIC language in the manufacturer's language manual.

Many of the HHC's features are not accessible through Microsoft BASIC. Microsoft BASIC has no statements to read and write File System files, nor can it recover from execution errors such as reading a nonnumeric data item into a numeric variable.

In summary, the HHC's Microsoft BASIC is a very attractive language for novice programmers but a poor tool for writing complex programs.

Figure 8-3 is a Microsoft BASIC program listing. This program prompts you for a date in "month, day, year" form—for example, "feb,1,1973"—and displays the day of year. February 1, 1973, was the 32nd day of that year, so

```

110 REM Convert a date to day-of-year.
120 REM Input: month, day, & year. A null month means "end run."
130 REM Output: day of year.
140 REM Variables: MN$=month of year (3 chars).
142 REM DM=day of month. YR=year.
150 REM DR=day of year, the result.
170 REM
180 REM MA= array of days in year before each month.
185 REM MO$(I)=name of month I+1.
190 DIM MO$(12),MA(12)
200 FOR I=0 TO 11
210 READ MO$(I),MA(I)
220 NEXT I
230 REM Prompt user for month, day, year.
240 INPUT "Month,day,year";MN$,DM,YR
250 IF MN$="" THEN END
252 FOR I=0 TO 11
254 IF MN$=MO$(I)GOTO 280
256 NEXT I
258 PRINT "Month name invalid!":GOTO 240
260 REM Calculate DR.
280 DR=MA(I)+DM
290 REM Allow for leap years.
300 IF I<=1 GOTO 350
310 IF YR/400=INT(YR/400)GOTO 340
320 IF YR/100=INT(YR/100)GOTO 350
330 IF YR/4<>INT(YR/4)GOTO 350
340 DR=DR+1
350 PRINT "Day of year=";DR
360 GOTO 240
370 DATA "jan",0,"feb",31,"mar",59
380 DATA "apr",90,"may",120,"jun",151
390 DATA "jul",181,"aug",212,"sep",243
400 DATA "oct",273,"nov",304,"dec",334

```

**Figure 8-3.** A simple Microsoft BASIC program

"32" is the output. The program allows for the fact that February has 29 days in leap years.

### When to Use Microsoft BASIC

Microsoft BASIC is an appropriate language for programs that are simple and small; that are used by a limited number of people; that must be

developed rapidly or by inexperienced programmers; that have already been written in Microsoft BASIC on some other computer; or that must be transportable to computers other than the HHC.

Microsoft BASIC is a good language for programs that do a lot of floating-point arithmetic with a preference for speed over precision. SnapBASIC uses the HHC's intrinsic arithmetic routines that are accurate to 13 decimal places, while Microsoft BASIC uses its own routines that are accurate to only 9 places. Microsoft BASIC's less precise routines run faster, and this can make its programs faster overall if they do a lot of arithmetic.

Microsoft BASIC tends to be inappropriate for programs that are complex, longer than a few pages, or that make heavy use of the HHC's features.

## How to Use Microsoft BASIC

You enter a Microsoft BASIC program through a special program editor built into the Microsoft BASIC capsule. This editor converts the program into Microsoft BASIC's interpretive code and stores the code in a file. You can examine a Microsoft BASIC program with the Microsoft BASIC editor, but not with the File System editor.

You run a Microsoft BASIC program by running Microsoft BASIC, selecting the file containing your program from a menu, and entering the Microsoft BASIC command RUN.

## SnapBASIC INTERPRETER/COMPILER

SnapBASIC is another dialect of the BASIC programming language. Like SnapFORTH, the SnapBASIC capsule compiles a BASIC program to an internal code. Thus, SnapBASIC programs tend to be compact and to run fast, although they are generally not quite so compact or fast as programs written in SNAP.

Because it is a dialect of BASIC, SnapBASIC allows you to debug programs in a highly interactive manner. Its combination of fast, compiled code and interactive debugging is unique among microcomputer BASICs.

In Figure 8-4 an example of a short program written in SnapBASIC demonstrates several attributes of good programming style. The ONERR statement on line 250 and the ONERR routine beginning on line 700 handle a nonnumeric response to the "ZIP code" prompt, trapping errors before they cause the program to fail. Line 260 determines how much free memory

```

110 REM ZIP code recording program.
120 REM Input: first, name for a file to store ZIP codes in.
130 REM Then, each city, state, & code.
140 REM Output: data is stored in file.
150 REM Variables: fname$=filename; city$=city name; state$=state;
160 REM zip=ZIP code; stlist$=valid states; errcode$=error code.
170 REM
200 stlist$ = "AL AK AR AZ CA CO DE DC FL GA HA ID IL IN IA KA "
210 stlist$ = stlist$ + "KY LA ME MD MA MI MN MS MO MT NB NV NH "
220 stlist$ = stlist$ + "NJ NM NY NC ND OH OK OR PA RI SC SD TN "
230 stlist$ = stlist$ + "TX UT VT VA WA WV WI WY "
240 REM Set up error recovery & 1000 bytes free space for file.
250 ONERR GOTO 700
260 zip=free(free(0)-1000)
300 REM Get file name & create file.
310 INPUT "Enter storage file name: ",fname$
320 IF exist(fname$) THEN PRINT "Already exists!" : GOTO 310
330 FOPEN fname$
400 REM
410 REM Take & record next input.
420 INPUT "Enter city name: ",city$
430 INPUT "Enter state code: ",state$
440 IF len(state$) <> 2 THEN GOTO 500
450 IF search(stlist$,state$+" ".1)=0 THEN GOTO 500
460 INPUT "Enter ZIP code: ",zip
470 IF zip < 10001 OR zip > 99999 THEN GOTO 600
480 FINS freq(0), city$+" "+state$+" "+left$(str$(zip), 5)
490 GOTO 400
500 REM
510 REM Error, invalid state.
520 PRINT "Invalid state code!" : GOTO 430
600 REM
610 REM Error, invalid ZIP code.
620 PRINT "Invalid ZIP code!" : GOTO 460
700 REM
710 REM ONERR recovery point.
720 errcode$ = chr$(peek(852)) + chr$(peek(853))
730 IF errcode$ = "TM" GOTO 620
740 PRINT errcode$ + " error!"
750 STOP

```

Figure 8-4. A simple SnapBASIC program

SnapBASIC has appropriated and instructs it to keep all but 1000 bytes, which are returned to the File System to give it space to create the output file.



When a SnapBASIC program starts running, it appropriates most of the free memory in the current file space for its own use.

"EXIST( )" in line 320 determines whether the file name given by the user is the name of an existing file. "SEARCH" in line 450 checks the validity of the state name abbreviation.

Capitalization and spacing enhance the program's readability. This sample program will be stored in a File System file; if it were stored in a SnapBASIC file, the capitalization and spacing would be lost.

## SnapBASIC and Microsoft BASIC Compared

With its ability to read and write RAM files and to recover from errors, SnapBASIC is a more powerful language than Microsoft BASIC. In addition, SnapBASIC has more extensive facilities for peripheral I/O and possesses several major features that Microsoft BASIC lacks.

With SnapBASIC you can store a source program in a File System file. To compile the program, you enter the LOAD command with the name of the file containing the program. You can edit a program with the HHC File System editor or with SnapBASIC's built-in editor.

With SnapBASIC you can store and edit SnapBASIC programs on a separate host computer and download them to the HHC.

SnapBASIC produces compiled SNAP code, which makes a SnapBASIC program faster and more compact than an equivalent Microsoft BASIC program.

SnapBASIC has an integer arithmetic facility. Since integer arithmetic is much faster than floating-point arithmetic, SnapBASIC is much faster than Microsoft BASIC for integer calculations. Without an integer facility, Microsoft BASIC must use its slower floating-point routines for all calculations.

SnapBASIC supports many additional built-in functions, including numeric functions LN, SIN, COS, and TAN, integer functions ABS%, MAX%, and MIN%, and bitwise logical operations functions BAND (and), BOR (or), and BXOR (exclusive or).

SnapBASIC supports DATE\$, TIME\$, SEARCH, STRF\$, and string functions that convert a numeric value to a string with extensive control over the way the value is formatted.

In SnapBASIC you can use variable names as long as will fit on a source file line. Microsoft BASIC allows you to write long variable names, but uses only the first two letters of a variable name.

A SnapBASIC program can read and write text or binary File System files. With SnapBASIC's ONERR command, you can write a program that recovers automatically from an error such as nonnumeric input into a numeric variable.

## When to Use SnapBASIC

SnapBASIC is an appropriate language for programs that must be developed rapidly or modified by inexperienced programmers, or that do not require heavy use of the HHC's features.

SnapBASIC is an inappropriate language for programs that require heavy use of the HHC's features or that must be as efficient as possible.

## How to Use SnapBASIC

Like SnapFORTH, SnapBASIC can compile a source program that is stored in a File System file or is downloaded from a host computer.

Instead of converting your program into a special interpretive code, SnapBASIC compiles the program into SNAP code. Furthermore, SnapBASIC preserves your comments as well as information about variable names and line numbers so that the compiled program can be translated to its original form when you list or edit it.

Both the compiled SNAP code and the other preserved information are stored in a SnapBASIC program file. You can list and edit this file only with the SnapBASIC editor, not the File System editor or Microsoft BASIC. To run the program, run SnapBASIC, select the SnapBASIC file that contains the program, and enter the command RUN.

With SnapBASIC you can inspect and change the values of variables by typing BASIC commands on the HHC keyboard and then executing all or part of your program. This sequence lets you debug the program interactively.

SnapBASIC's ability to compile a program that is downloaded from a host computer is a great convenience if your program is more than a few pages long. If, on the other hand, you use the SnapBASIC editor to modify your program while you debug it, you will have to upload or duplicate the changes to the host system before downloading the program for additional work.

## Summary

Each of the HHC's three programming languages is best suited for certain applications. You will have to decide which language to use for each

**Table 8-1.** HHC Programming Languages

Characteristic	SnapFORTH Assembly language	SNAP	Microsoft BASIC	SnapBASIC
Access to File System	yes	yes	no	yes
Run-time error recovery	yes	yes	no	yes
HHC operating system features accessible to a program	all	all	few	some
Programmer experience needed for effective use	much	much	little	little
Ease of use for experienced programmer	poor	moderate	good	good
Suitability for large complex programs	very poor	good	poor	moderate
Execution efficiency	very high	high	very low	moderate
Programs can be placed in HHC capsules	yes	yes	no	yes
Programs can be placed in SNAP files	yes	yes	no	no

application on a case-by-case basis. Table 8-1 summarizes the characteristics of the HHC's three programming languages.

## PROGRAM DEVELOPMENT FOR THE HHC

This section provides information on ROM chip types and sophisticated program development. If you are working only in Microsoft BASIC, you may wish to skip this section.

### Types of Program Capsules

Several of the HHC language processors can produce object programs suitable for "burning" into HHC capsules. To produce programs that may be used in capsule form, you should know about the two types of ROMs used in HHC capsules.

A ROM chip's appropriateness for a given program depends on the

number of program copies you expect to produce, not the function of the program or the programming language used. Most HHC capsules come on *masked ROM* chips. This type of ROM is mass-produced with a built-in program and is suitable for production runs of a few hundred capsules or more. Small-volume and custom programs developed by HHC distributors come on *erasable programmable ROM* (EPROM) chips. When purchased from HHC distributors, these chips are blanks premounted on capsule carriers.

A program can be "burned" into EPROM chips, one or a few at a time, with an *EPROM Burner*. The EPROM Burner is a computer peripheral that records HHC programs on EPROM chips. EPROM Burners are available for many kinds of host computers. Since some burners are made with RS-232C interfaces, you can use them on a variety of computers, including the HHC itself.

ROM capsules are manufactured in 2K, 4K, 8K, and 16K byte sizes. Although 16K bytes is the largest size capsule the HHC can accommodate, EPROM capsules are manufactured only in 2K and 4K byte sizes.

Table 8-2 shows the characteristics of the two types of ROM chips.

### The HHC System Development Tools

HHC distributors offer a development system called the HHC System Development Tools (HHC/SDT) for HHC programs. HHC/SDT runs on the Apple II computer and offers several major tools for developing program products, including a complete FORTH-based operating system, a SNAP compiler that runs on the Apple and produces object code for execution on the HHC, and an HHC simulator that runs on the Apple and permits you to test SNAP code. The simulator includes a SNAP interpreter

**Table 8-2.** Characteristics of ROM chip types

Characteristic	Masked ROM	EPROM
Burned during manufacturing process	yes	no
Created on EPROM Burner	no	yes
Per-program setup cost	\$10,000+	none
Per-capsule cost	relatively low	relatively high
Economic production run	thousands	up to hundreds



so you can test and debug programs with the Apple's full-size keyboard.

HHC/SDT also has a utility for making HHC EPROMs with an EPROM burner card plugged into the Apple; utilities for downloading source programs or compiled programs from the Apple to the HHC and uploading them from the HHC to the Apple; and a screen-oriented text editor that can be used for SNAP program development and word processing. The word-processing function is essential for writing documentation.

Finally, HHC/SDT has a utility that makes the Apple function like a "dumb terminal" when connected to the HHC's RS-232C Interface. Many HHC programs, including SnapFORTH and SnapBASIC, allow you to use a "dumb terminal" as a full-size keyboard and multi-line display in place of the HHC's keyboard and LCD.

Though not essential for developing application programs in SNAP, HHC/SDT makes a useful adjunct to SnapFORTH, especially if you already plan to do your source code editing on an Apple. HHC/SDT enables you to test most functions of your programs with the Apple's full-size keyboard while running under the HHC simulator, and it offers you a number of utility programs that are useful for HHC program development.

### Products for Developers

In addition to the HHC System Development Tools, HHC distributors sell several products that are useful to software and hardware developers.

If you have an EPROM burner designed for a host computer, you can purchase an adaptor for burning HHC EPROMs. HHC distributors sell blank EPROM chips premounted on HHC capsule carriers.

An HHC EPROM Burner peripheral, currently under development, will be a convenient tool for making EPROMs on the HHC. Watch for announcements of this peripheral's commercial availability.

Contact your HHC dealer or distributor for more information about these products.

# 9

## Using Peripherals

**P**eripheral devices expand the capabilities of the HHC immensely, enabling it to read input from sources other than the keyboard and write output to destinations other than the LCD.

In this chapter you will learn about the various HHC peripherals that are available and the applications for which they are suited. You will also learn procedures that will help you use the peripherals.

All the information that you will need to use any peripheral with prewritten software is covered in this chapter. If you want to learn to write programs that exercise a peripheral's special features, however, you must also consult the HHC owner's manual or the literature that comes with the peripheral.

You may also use this chapter to evaluate peripherals that you are considering purchasing.

Peripheral devices are divided into three categories: those that read data (*input* peripherals), those that write data (*output* peripherals), and those that read and write data (*input/output* peripherals). Included in the latter category are the Programmable Memory Peripheral (PMP), the RS-232C

Interface, Modems, and the I/O Adaptor. Examples of output peripherals are the Four-Color Plotter, Mini Printer, Micro Printer, and TV Adaptor. No current HHC devices serve strictly as input peripherals.

The first section of this chapter presents information common to all the HHC peripherals, using the Mini Printer as a specific example. The second section describes each individual peripheral in detail. After reading the first section you should read the second section to learn about the peripherals you own or those you wish to preview.

### Connecting a Peripheral

To connect a peripheral to the HHC follow the same procedure used to connect the Programmable Memory Peripheral (PMP). First, turn off your HHC. This precaution eliminates any risk of damaging the electronic components in the HHC or the peripheral.

Now plug the peripheral into your HHC's peripheral socket. As you connect the peripheral, the HHC beeps. When the beeping stops, the peripheral plug is properly seated.

If you plan to move the HHC and peripheral while they are connected, use the HHC's rigid plastic sleeve to hold them together.

Finally, turn your HHC on again. If you forget to return the HHC to the primary menu before attaching the peripheral, the HHC's operating system, operating as if you had pressed the CLEAR key twice, automatically returns to the primary menu.

Now press the I/O key, and the I/O menu will tell you what devices are available. Each device configuration presents a unique menu. If an RS-232C Interface, Modem, and Mini Printer are attached, your I/O menu will look like this.

1=RS232C, IN, ON, SLOT=3

2=RS232C, OUT, ON, SLOT=3

3=MODEM IN, OFF, SLOT=4

4=MODEM OUT, OFF, SLOT=4

5=PRINTER OUT, OFF, SLOT=5

6=INT RAM, 7145 FREE

To turn the Mini Printer on, press the key corresponding to the Mini Printer's number—in this case, 5—on the I/O menu. At this point the HHC displays

PRINTER OUT, ON, SLOT=5

Now that the peripheral is turned on, the HHC resumes displaying the I/O menu.

To return to the primary menu, press the I/O key again.

Once a peripheral is on, pressing CLEAR will not turn it off. You must turn it off with the I/O key by disconnecting it from the HHC or connecting another peripheral.

Application programs as well as the I/O menu can turn on peripherals. When you run them, some application programs turn on peripherals automatically.

While displaying the I/O menu, the HHC cannot execute the program (if any) that was running when you pressed the I/O key. As a result, peripheral input intended for that program may be lost. Most peripherals can save, or *buffer*, incoming characters until you leave the I/O menu, but their buffering capacity is usually limited to no more than ten characters.

The I/O menu displays the key that must be pressed to turn on each peripheral and displays each peripheral's status. For example,

1=PRINTER OUT, ON, SLOT=0

describes the status of the Mini Printer. According to this I/O menu, the Mini Printer is an output device, is currently turned on, and is plugged directly into the HHC's peripheral, or zero, socket.

For the peripheral capable of input as well as output, the I/O menu contains two items—one labeled "OUT" and one labeled "IN." Because the HHC treats an input/output peripheral as two separate devices—one for input and one for output—you should turn on both "devices" when you want to use an input/output peripheral.

When a peripheral is turned on, it uses a few dozen bytes of the free space in internal RAM. If you want to calculate the exact amount, note the amount of free space that the I/O menu shows in internal RAM before and



after you turn on the peripheral. Once this space is taken, the space remains tied up until the device is turned off *and* the HHC is returned to the primary menu. Simply turning the device off again will not free the space.

Some programs, including Microsoft BASIC, disable the I/O key. Before running such a program, you must turn on all the peripherals that you need.

## Peripherals as Slaves

The File System editor from the HHC's primary menu demonstrates the use of an output peripheral.

The File System editor automatically writes to any output device connected to the HHC and turned on with the I/O key. Whatever the File System displays on the LCD, it also writes to the peripheral. In other words, the File System *slaves* the peripheral to the LCD.

Select the File System from the primary menu. As soon as the program starts running, output appears on the peripheral.

To turn a peripheral off while running a program, invoke the I/O menu by pressing the I/O key and then press the key corresponding to the peripheral's menu entry. At this point the HHC will display a message like

```
PRINTER OUT, OFF, SLOT=0
```

and then display the I/O menu. If you press the I/O key again, the HHC returns to the File System. But since the peripheral is turned off, it is not slaved to the LCD output.

Practice turning the peripheral on and off again with the I/O key. Whenever the peripheral is on, it is slaved to the File System's LCD output. Some peripherals generate a spurious output character when you turn them off with the I/O menu or when you turn the HHC off with the OFF key.

Not all HHC programs simply slave peripherals to the LCD. Since many programs use peripherals for other functions, you may have to enter special commands to operate them.

## Disconnecting a Peripheral

Turn your HHC off before you disconnect a peripheral, just as you would before connecting one. When you disconnect a peripheral, the HHC operating system displays the primary menu and turns off any peripherals that are still attached. Use the I/O key to turn the remaining peripherals on again if you wish to use them.

## Using Peripherals on Battery Power

Many of the HHC peripherals drain substantial power from the HHC's batteries. The TV Adaptor draws so much power that you cannot run it on batteries at all. If you are using a peripheral for any extended period of time, use it with an AC Adaptor.

Some HHC peripherals have built-in rechargeable batteries similar to those in the HHC main unit. These peripherals can run off one charge of their batteries for a few hours.

Recharge a peripheral's batteries by attaching the peripheral to the HHC and the HHC to an AC Adaptor. The batteries will recharge more slowly while the HHC is turned on than while it is turned off.

All the instructions you read in Chapter 2 about the HHC's batteries apply as well to rechargeable batteries in peripherals.

## THE I/O ADAPTOR

The I/O Adaptor is an extension to the HHC's peripheral socket. Not strictly a peripheral, it lets you connect several peripherals to your HHC at once.

The I/O Adaptor, shown in Figure 9-1, is a plastic tray with three long slots on the right side and three short slots on the left side. The lowest right-hand slot has a plug that fits into the HHC's peripheral socket. Peripherals use the other five slots. A sixth peripheral socket is mounted on top of the I/O Adaptor.

The left side of the I/O Adaptor's tray is attached to the rest of the Adaptor by two spring-loaded clips. If you do not need the sockets on the left side of the tray, you can detach it. Once the tray is detached, the I/O Adaptor is somewhat smaller and easier to handle.

To use the I/O Adaptor, slide the HHC all the way into the lower right slot. Your HHC beeps while you do this, but the beeping stops when the HHC is properly seated.

Although you can plug any peripheral device into any socket, for the sake of convenience plug only long peripherals such as the Modem into the long slots. The top, sixth socket is reserved for future peripherals that cannot fit in the I/O Adaptor's tray. To avoid damaging a peripheral or the I/O Adaptor, use this socket only with an I/O bus cable like the one shown in Figure 9-2.

When you use the I/O key with the HHC, a Printer, a PMP, the RS-232C

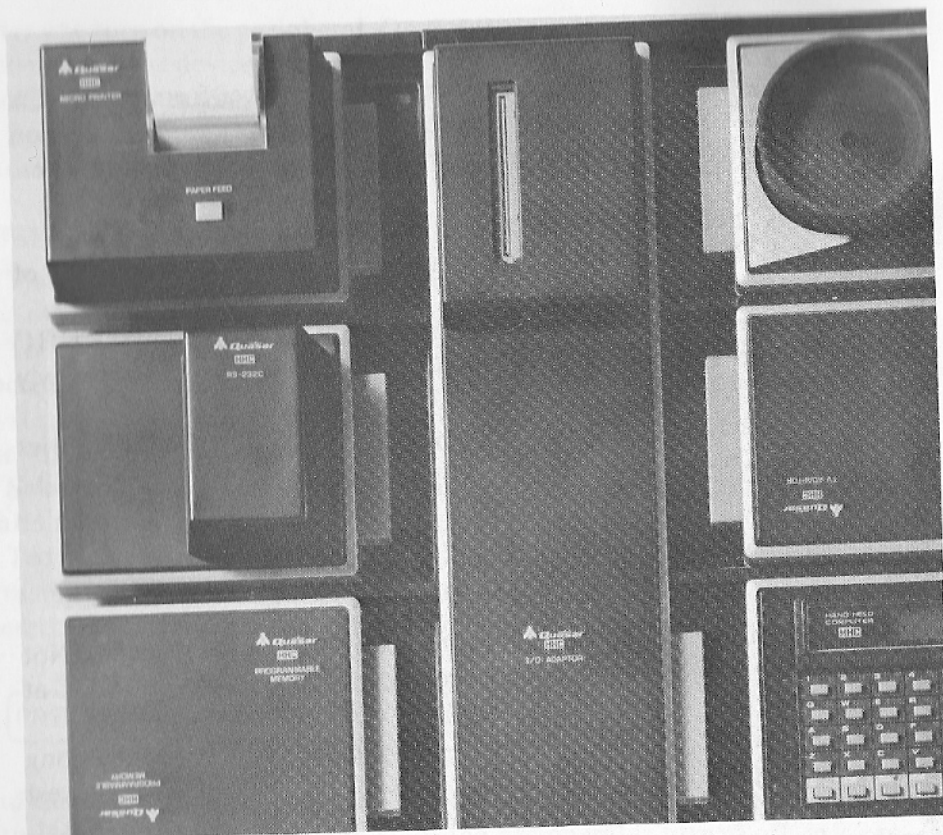


Figure 9-1. The I/O Adaptor peripheral, showing the HHC socket and six peripheral sockets

Interface, and a Modem mounted in the I/O Adaptor, the I/O menu shows the following list:

1=PRINTER OUT, OFF, SLOT=3

2=RS232C, IN, ON, SLOT=4

3=RS232C, OUT, ON, SLOT=4

4=MODEM IN, OFF, SLOT=5

5=MODEM OUT, OFF, SLOT=5

6=INT RAM, 1317 FREE

7=EXT RAM, 1211 FREE, SLOT=1

Use the I/O menu just as you do when you plug a peripheral directly into the HHC—in other words, select a menu item representing a peripheral to turn that peripheral on or off. If you want to select a menu item representing a Programmable Memory Peripheral (PMP) to switch that PMP into the current file space, select item 7. A PMP represents *extrinsic RAM*, displayed as “EXT RAM” in the menu. If you want to choose intrinsic HHC RAM, you would choose item 5.

The “SLOT=x” at the end of each I/O menu entry identifies the I/O

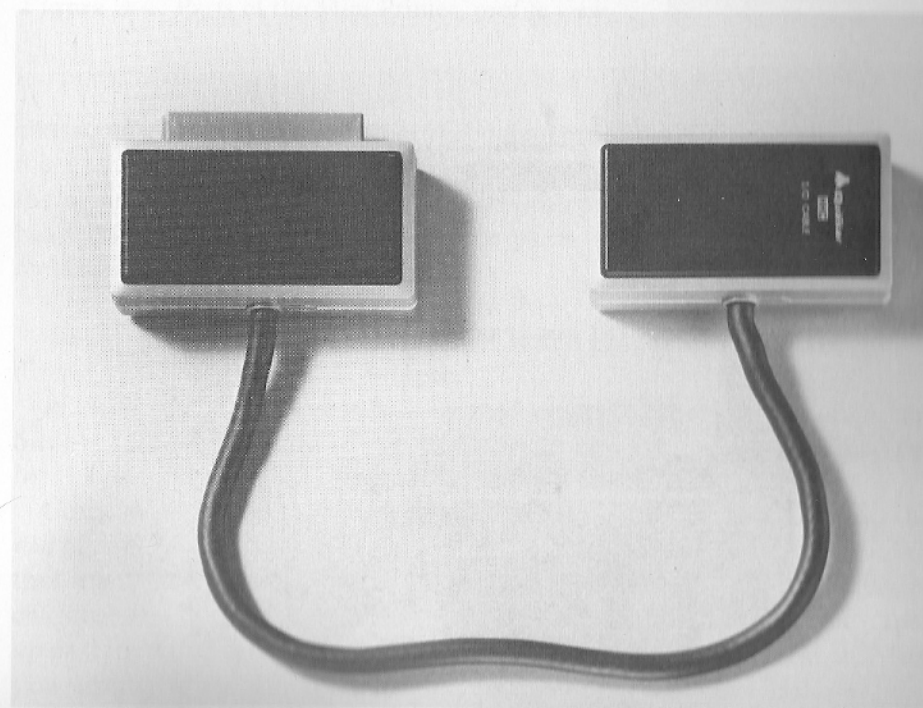


Figure 9-2. The bus cable that is used to connect a peripheral to the sixth peripheral socket on the I/O Adaptor



Adaptor slot or socket into which each peripheral is plugged.

You may want to stick small adhesive tabs next to your I/O Adaptor's sockets to distinguish one slot from another. The slot numbers increase counter-clockwise from the lower left, where the first slot is 1.

## PRINTING AND PLOTTING DEVICES

In order to give you permanent records of your work sessions, HHC printing devices include two printers and a color plotter.

### The Mini Printer

The Mini Printer, shown in Figure 9-3, prints 40-character lines on paper rolls that are 80 mm (about 3-1/2 inches) wide, making it a convenient, portable device for getting hard copy from your HHC. Figure 9-4 shows its parts.

The Mini Printer produces output with a  $5 \times 7$  dot matrix for each

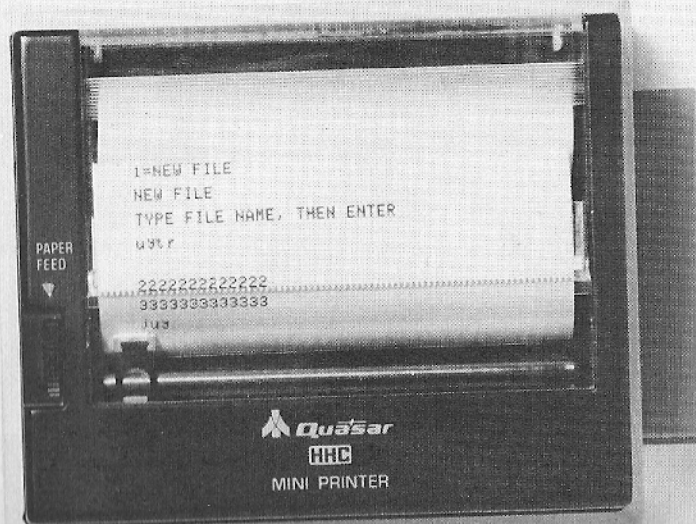


Figure 9-3. The Mini Printer

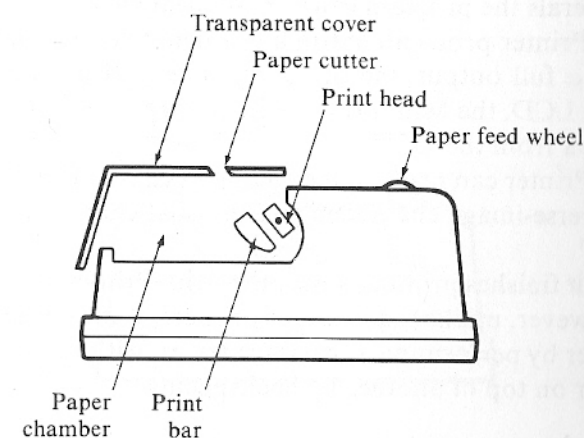


Figure 9-4. Parts of the Mini Printer, profile view

character. Its "thermal" printing mechanism produces characters by activating a row of tiny heating elements as they pass over the paper. This makes the printer very quiet and reliable. The only moving parts move the print head back and forth and advance the paper between lines. The thermal mechanism requires special paper.

Complete specifications for the Mini Printer are shown in Table 9-1.

To use the Mini Printer, first unpack it and plug it into the HHC, either directly or through the I/O Adaptor.

A transparent plastic cover is located on the printer's paper chamber. Slide the cover backward and then remove it. You are now ready to load the Mini Printer with paper.

Consult Figure 9-5 while you load paper into your printer. Unwrap the roll of paper that comes with the printer. If the paper is ragged, cut the end so that it is square. Feed the paper through the slot at the front of the paper chamber and turn the "paper feed" wheel to make the printer pull the paper up past the print bar. Now drop the paper roll into the paper chamber. After you slip the cover over the chamber, the printer is ready for use.

To operate the Mini Printer connect it to your HHC and select it from the I/O menu. Now it is ready to be used by any program that writes to a Mini

Printer. The documentation accompanying each program capsule identifies which peripherals the program can communicate with.

The Mini Printer prints an entire line when it receives an "end of line" character or a full output line of 40 characters. If a program slaves the printer to the LCD, the Mini Printer will not *begin* to print a line until that line *disappears* from the LCD.

The Mini Printer can print all the characters in the HHC's character set, including inverse-image characters and unusual characters like "ä," "ö," "ü," and "ñ."

Each time it finishes printing a line, the Mini Printer advances the paper one line. However, unlike many larger printers, it cannot print one line on top of another by performing a "carriage return without line feed" or print one character on top of another by backspacing.

Table 9-1. Printer and Plotter Characteristics

	Micro Printer	Mini Printer	Four Color Plotter
Printing mechanism	Thermal dot matrix	Thermal dot matrix	Turrent holding 4 ballpoint pens
Resolution	5 × 7 dots/char	5 × 7 dots/char	0.2mm/step, horizontal and vertical
Maximum line length characters	15 char/line	40 char/line	96mm (4"); 16 character sizes, 13 to 80 chars/line (40 chars/line standard)
Paper type	Thermal, roll, 38mm × 8m (1-1/2" × 26')	Thermal, roll, 80mm × 8m (3-1/2" × 26')	plain, roll, 114mm × 8m (6-1/2" × 26')
Approximate printing speed	1.5 lines/sec regardless of line length	20 char/sec; depends on line length	6 char/sec with 40 char line; smaller chars print faster
Word wrap	Yes	Yes	No
Size (W × H × D)	113.5 × 53 × 95mm (4-1/2 × 2-1/16 × 3-3/4")	113.5 × 61 × 95mm (4-1/2 × 2-13/32 × 3-3/4")	227 × 58 × 95mm (8-15/16 × 2-9/32 × 3-3/4")
Weight	450g (1 lb)	550g (1 lb 3.4 oz)	920g (2 lb 1 oz)
Power source	Built-in batteries recharged by AC adaptor	Built-in batteries recharged by AC adaptor	Built-in batteries recharged by AC adaptor; draws 350 mW

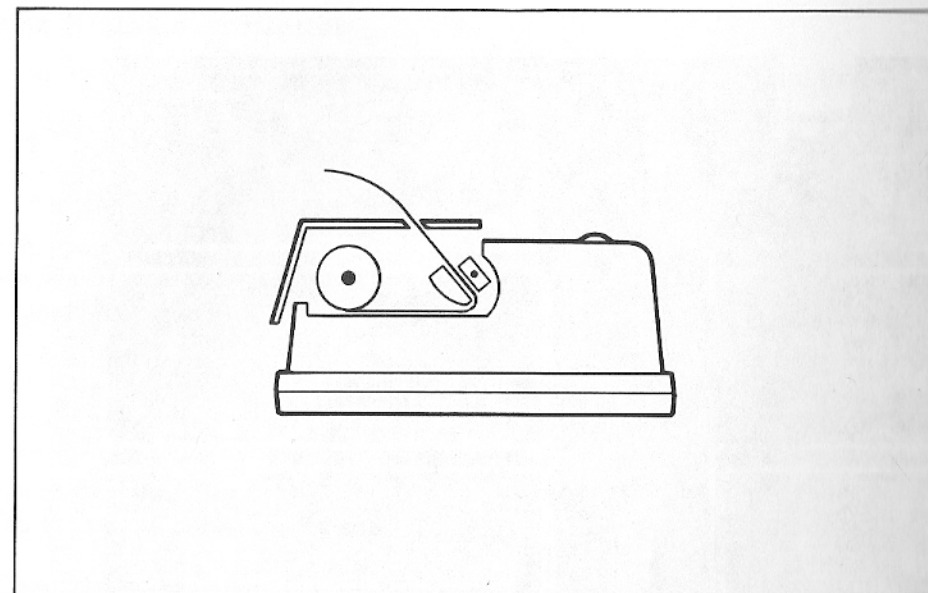


Figure 9-5. Loading paper on the Mini Printer

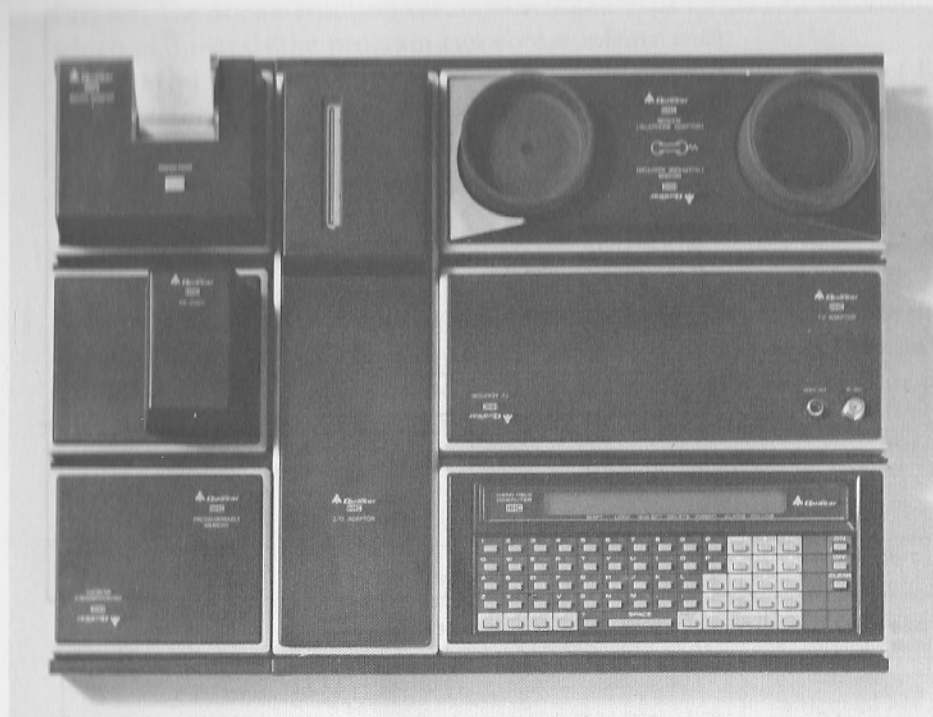
A special feature of the Mini Printer and of many other HHC peripherals is *word wrap*. If a program tries to print a line that is more than 40 characters long, the printer automatically breaks the line into two shorter lines after printing as many complete words as will fit on the first line. The printer "wraps" the entire word that does not fit on the first line onto the beginning of the next line. This line division may take place before the printer produces 40 characters.

### The Micro Printer

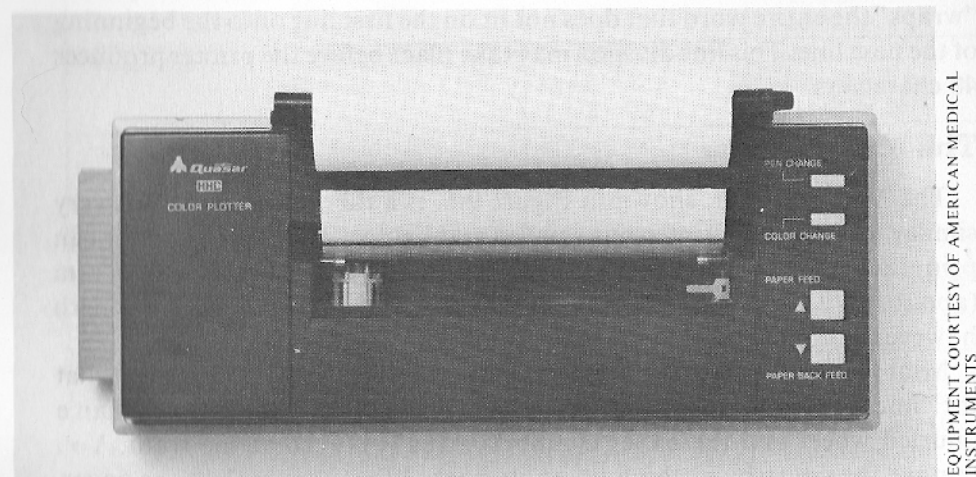
The Micro Printer, shown in Figure 9-6, is a thermal printer. It is very similar to the Mini Printer but smaller and lighter. The Micro Printer can print lines with as many as 15 characters on thermal paper that is 38 mm (about 1-1/2 inches) wide. Its usefulness is limited by its short line length however.

Mini Printer instructions apply equally to the Micro Printer, except that the "line feed" control for the Micro Printer is a push button rather than a knurled wheel and the paper chamber cover is hinged at the front. You remove the cover by pushing up on its back. When you replace the cover, you must feed the end of the paper through the slot in its top.





**Figure 9-6.** The Micro Printer (upper left corner) with the HHC and I/O Adaptor



**Figure 9-7.** The Four Color Plotter

## The Four-Color Plotter

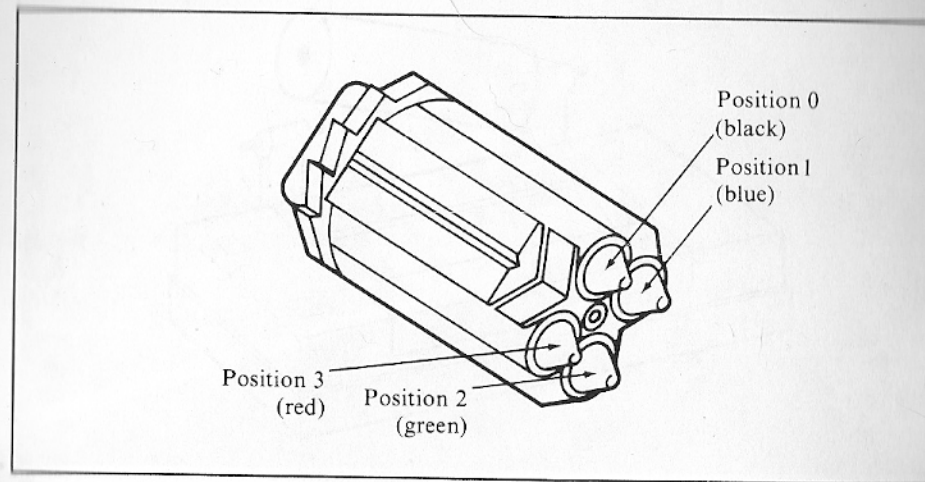
Although it prints more slowly than the Micro Printer and Mini Printer, the Four-Color Plotter, shown in Figure 9-7, can produce charts, graphs, and other kinds of diagrams on a roll of plain, nonthermal paper. You can use the Plotter as a printer with several different sizes of characters.

The Plotter both prints and plots with a turret containing four ballpoint pens. Shown in Figure 9-8, the turret moves along the paper to produce horizontal motion and rotates so that the pen colors can be changed, the paper rolls up and down to produce vertical motion, and the pen lifts off the paper to move the turret without drawing.

Horizontal and vertical motion take place in steps of 0.2 mm (about 1/125 inch). This step is small enough to produce many types of accurate, pleasing graphics yet large enough to be visible so, as a result, it is inappropriate for exacting applications. Figure 9-9 is an example of graphics rendered by the Plotter.

## The Plotter Paper

To install a roll of paper, remove the cover from the Plotter and consult Figure 9-10. Gently press down on the grooved tabs at the cover's back corners and then press forward. Use only paper rolls supplied for the HHC Plotter by an authorized HHC distributor. Other types of paper may produce unsatisfactory results.



**Figure 9-8.** Close-up of Plotter pen turret

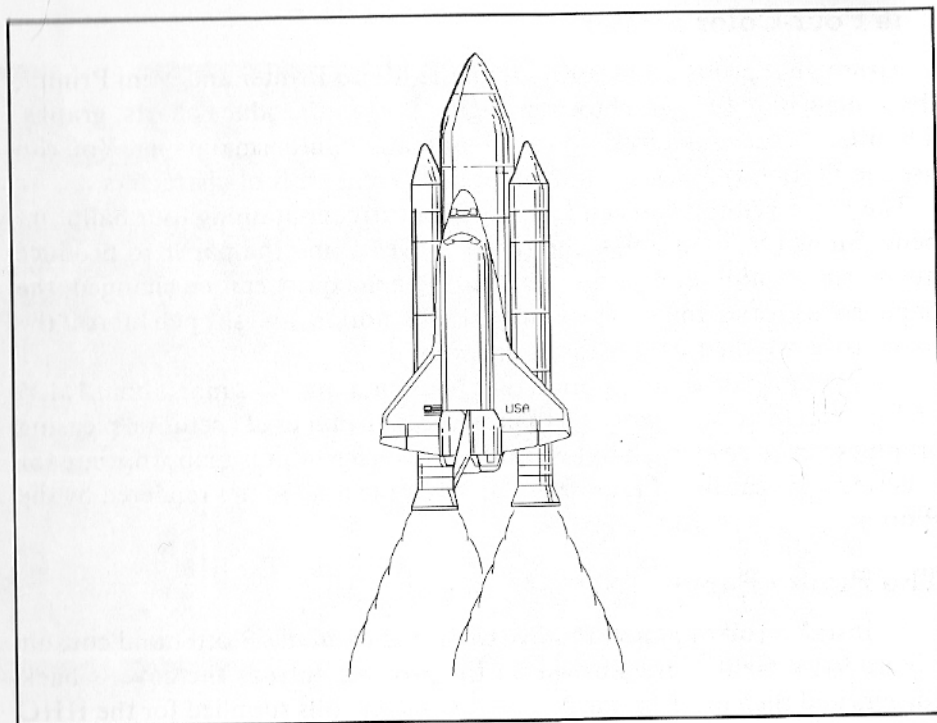


Figure 9-9. Plotter output sample

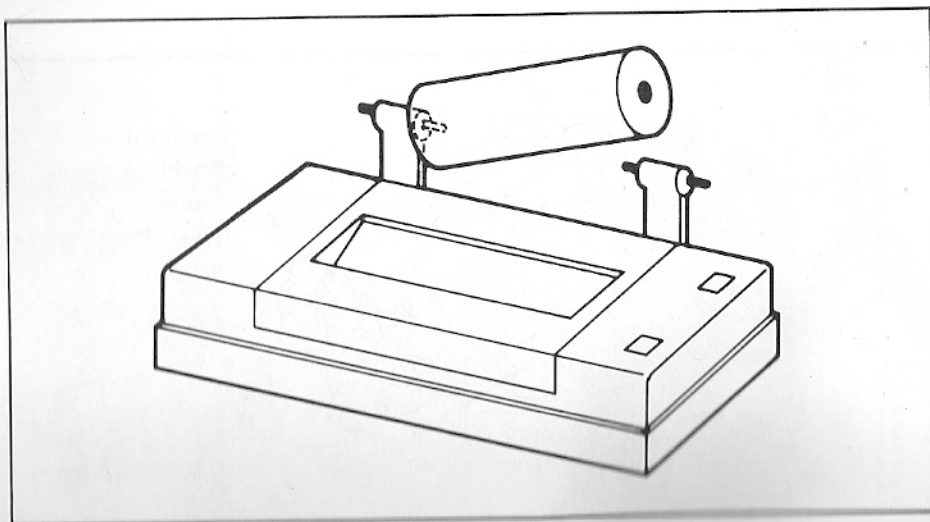


Figure 9-10. Loading paper on the Plotter

Attach the Plotter to the HHC and connect the HHC to an AC Adaptor. Raise the arms of the Plotter's paper holder and slip the core of the paper roll over the pins at the ends of the arms. You can do this more easily by pulling outward on the right-hand pin.

Now cut the end of the paper roll so that its edge is even and insert the end in the paper inlet at the back of the Plotter. When the paper stops, press the PAPER FEED button long enough to feed an inch or so of paper past the pen turret and then replace the Plotter's cover.

### The Plotter Pens

The HHC Plotter uses ballpoint pens that are designed to fit the plotting turret. Use only pens supplied for the HHC Plotter by an authorized HHC distributor.

Before installing the pens, remove the cover from the Plotter and attach the Plotter to the HHC and AC Adaptor, just as you do when you install paper.

The turret holds four pens filled with black, blue, green, and red ink. To mount the pens in the correct slots, turn on the HHC and press CLEAR. This makes the Plotter return the pen turret to the left margin and position the turret so that the "black" pen position is uppermost. Figure 9-11 shows the turret in its "home" position.

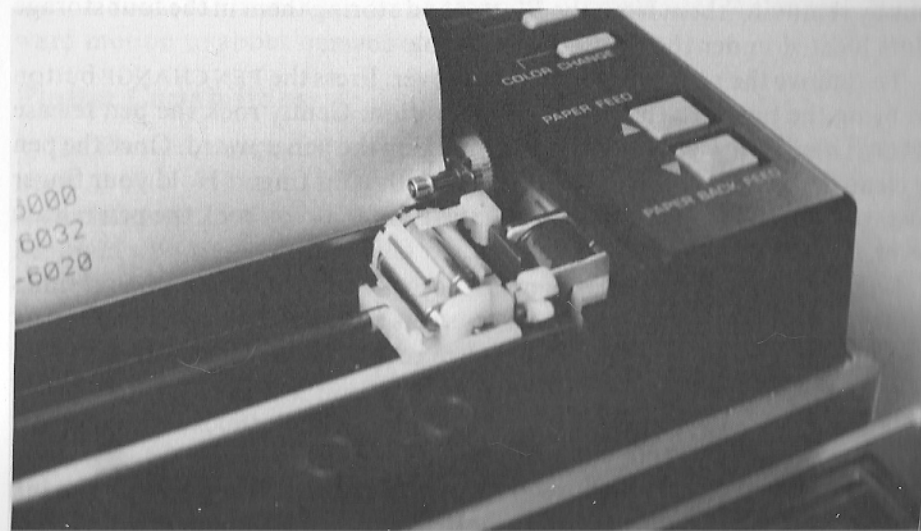


Figure 9-11. Plotting turret in home position



After you press **CLEAR**, you should see a small metal bar in the plastic divider located to the left of the uppermost pen. This bar is a magnet that the Plotter uses to sense the position of the turret.

Now press the **PEN CHANGE** button and the plotting turret will move to its "pen change" position at the right end of the platen. Slip the pen with black ink into the turret slot, writing tip first.

Press the **COLOR CHANGE** and **PEN CHANGE** buttons again. Now insert the pen with blue ink. Repeat this procedure for the third and fourth pens (green and red). Now replace the cover and you are ready to use the Plotter.

*Caution:* Do not rotate the turret by hand or touch any part of the mechanism in ways not described here. This part of the Plotter is rather delicate and easily damaged.

Never operate the Plotter without all four pens in place. If you break this rule, you may damage the Plotter's mechanism.

## Changing or Removing Pens

You must remove the pens to change them when they run dry. If you do not use the Plotter for more than a few days, protect the pens from drying out by removing them from the Plotter and storing them in the four storage slots located under the Plotter's removable cover.

To remove the pens, lift the Plotter's cover. Press the **PEN CHANGE** button to bring the turret to the pen change position. Gently rock the pen release lever. This lifts a wire below the pen, nudging the pen upward. Once the pen is clear of the turret, you can slide it out with your finger. Hold your finger over the pen to prevent it falling into the Plotter as you rock the pen release lever.

If the pen slips under the wire, press the **COLOR CHANGE** button four times and the **PEN CHANGE** button once. This gets the wire back under the pen by turning the turret through one complete revolution and returning it to the pen changing position.

If the pen pops out of the turret and falls into the Plotter, turn the Plotter upside-down and let it drop out.

Before you install a new pen, run it over a sheet of paper so that the ink will begin to flow. This action ensures that the pens will begin drawing as soon as the Plotter starts.

## Using the Plotter

Built-in rechargeable batteries can operate the Plotter for at least an hour. When the Plotter is not running, you can recharge its batteries with either the small or the large AC Adaptor. To recharge the Plotter while it is running, you must use the large AC Adaptor.

If several peripherals are in use, even the large AC Adaptor may be unable to recharge the Plotter while it is running. In such a situation, the AC Adaptor will extend the batteries' life to about four hours of plotting time.

To use the Plotter, simply turn it on with the I/O menu and run any program that operates the Plotter or a Printer. The Plotter operates as a printer until a program sends it a command that shifts it to plotting mode. Thus, any program that slaves output devices to the LCD can operate the Plotter because the program does not have to "know" about the Plotter's graphic abilities.

The Plotter has four control buttons: **COLOR CHANGE**, **PEN CHANGE**, **PAPER FEED**, and **PAPER BACK FEED**. You have already used the **COLOR CHANGE** and **PEN CHANGE** buttons.

The **PAPER FEED** button advances the paper continuously in plotting increments at a speed of about one inch per second.

The **PAPER BACK FEED** button moves the paper backward in plotting increments. When first pressed, it moves the paper slowly, allowing you to position the paper quite accurately. If you hold the **PAPER BACK FEED** button down for more than three seconds, it accelerates the paper's backward motion to about one inch per second.

## Plotter Capabilities

The Plotter can operate in either of two modes: character mode or graphic mode.

In character mode, the Plotter functions as a conventional printer with backspace and reverse line feed commands, enabling it to move its print head in all four directions. The character set is "programmed" by two built-in ROMs.

The Plotter normally draws characters right-side-up, in black ink, and the standard character size produces a 40-character line. You can choose an orientation that is upside-down and backward, sideways reading up the page, or sideways reading down the page. A program can at any time switch to red, green, or blue ink with any of several character sizes from 13 to 80 characters per line.

In graphic mode, the Plotter has extensive plotting capabilities. You can

draw lines in any position and direction in several line types, including solid, dotted, and dashed. Lines are defined by the coordinates of each end point or by the coordinate of one end point and the relative position of the other. You can move the pen to any point without drawing a line.

Circles, ellipses, radial lines, and Cartesian axes are all possibilities.

You can use hatching of specified angle and density to fill in squares and arcs, and draw sine and cosine curves of specified amplitude and frequency.

If you print a string of characters, you can choose one of 16 different character sizes and change the orientation of characters.

You can write your own plotting software in any programming language supported by the HHC. The Plotter's user's manual explains in detail how to use the Plotter to draw graphics.

## COMMUNICATION DEVICES

The HHC has two communication devices: the RS-232C Interface and the Modem. The Modem is made with an optional Cassette Interface. All expand the HHC's capabilities by allowing communications with other computers and data storage media.

### The RS-232C Interface

Sometimes called the *serial interface*, the RS-232C Interface enables your HHC to communicate with any device that follows the RS-232C communication standard.

Since most computers conform to the RS-232C communication standard, you can use the RS-232C Interface to transfer data directly between your HHC and another computer.

Using the RS-232C Interface, you can connect your HHC to printers, video and printing terminals, disk and tape recording devices, and modems that conduct computer communications by telephone.

The RS-232C Interface can read and write data at data rates as high as 9600 bits per second, a rate that corresponds to about 960 characters per second or—as it is more commonly known—9600 *baud*.

If you wish to connect your RS-232C Interface to the RS-232C device, you must use a cable with a 25-pin DB-25 connector at each end. You can purchase these cables from your HHC dealer or from most computer supply dealers, or get a service technician to wire one for you.

Since different devices require cables wired in different ways, HHC distributors sell three types for use with the RS-232C Interface.

The *straight cable* connects every required pin of one plug to the corresponding pin of the other plug and is used to connect the RS-232C Interface to a serial interface modem. This cable is described in the RS-232C section of your HHC owner's manual under the heading "Connection To the Modem." The HHC's RS-232C Interface is configured as data terminal equipment, not data communication equipment. Unlike other RS-232C interfaces it is connected to modems with a straight cable and to devices such as terminals and printers with a crossed cable.

A *crossed cable* cross-connects certain pairs of pins in the two plugs and is used to connect the RS-232C Interface to a printer, display terminal, or similar serial interface device. It is described in the RS-232C section of your HHC owner's manual under the heading "Connection Between HHC's."

The third type of cable is wired to every pin of a plug at one end but is completely unwired at the other end. Since you can wire this cable any way you want, you can use it with a device that has special cabling requirements.

Unless you have some prior experience with computer hardware, however, leave cable wiring to an experienced technician. It is a demanding task, and few people do a respectable job the first time they try it.

To prepare the RS-232C Interface for use, you must have a device capsule intended for the RS-232C Interface, such as the RS-232C Communications capsule. Although intended primarily for the Modem, the Telecomputing 1 and 2 capsules also work.

Remove the sliding panel on the bottom of your RS-232C Interface. Beneath it you will find a capsule socket and a row of eight slide switches. Insert your device capsule in the socket.

### Configuring the Interface

You must decide at what *baud rate*, or data transmission rate, you want the RS-232C Interface to run. To transfer information correctly, the RS-232C Interface on your HHC and on the other device must run at the same rate.

Some devices operate at a fixed baud rate, while others let you choose a rate. In general, choose the highest rate that the HHC and the other device can handle.

The highest baud rate at which you can operate the RS-232C Interface may be limited by some HHC software programs. If you are not sure what baud rate to use, establish communication at 300 baud and then raise the baud rate until you find the highest rate at which your HHC and program work reliably.



The slide switches next to the capsule socket control the RS-232C Interface's baud rate. Table 9-2 will help you choose the proper slide switch settings.

RS-232C parameters must be set properly on the RS-232C Interface and the other device before the RS-232C Interface can communicate successfully. All of these parameters except baud rate are set through software, generally by a CONFIGURE menu item in an application program in the device capsule.

For information about the parameters that you can set and the procedure used to determine their values, see the user's manual for the specific equipment you are employing. Most of these manuals contain lists of configuration parameters that are appropriate for common types of devices.

If you are not sure how to configure the Interface, don't be afraid to experiment. You cannot hurt the Interface or the other device by choosing incorrect parameters.

To use the RS-232C Interface, connect it to the HHC. Then

*Step 1.* Connect the data cable to the RS-232C Interface's DB-25 socket and to the device or other computer.

*Step 2.* Turn on the device or other computer, if you have not already done so.

*Step 3.* Turn on the RS-232C Interface with the I/O menu.

*Step 4.* Run a program that reads, writes, or both reads and writes to the RS-232C Interface.

## Troubleshooting

Configuring an RS-232C Interface to a peripheral device or computer is, unfortunately, one of the more complicated procedures in computing. Many problems can occur, and most of them result in the same symptom: No data are transferred.

If you connect your RS-232C Interface and it doesn't work, you have to follow a troubleshooting checklist.

First, make sure your cable is wired as directed. If your cable has an "HHC end" and an "other end," try reversing the cable. Try this particularly if your HHC is connected to another computer and if the other computer "freezes" when it should send a character to the HHC.

Next check the baud rate settings and other parameters on the RS-232C Interface and the peripheral device to see if you set one of them incorrectly.

**Table 9-2.** RS-232C Baud Rate and Switch Settings

Baud Rate (Bits/Sec)	Approximate Data Rate (Char/Sec)	Set These Switches off	All Others on
50	5	1,2	
75	7.5	1	
110	11	2,4,6,7,8	
150	15	2	
200	20	3,4	
300	30	3	
600	60	4	
1200	120	5	
2400	240	6	
4800	480	7	
9600	960	8	

This is particularly likely if data get through the connection but are garbled, or if everything received by the HHC looks like an inverse-image "@" or an "a."

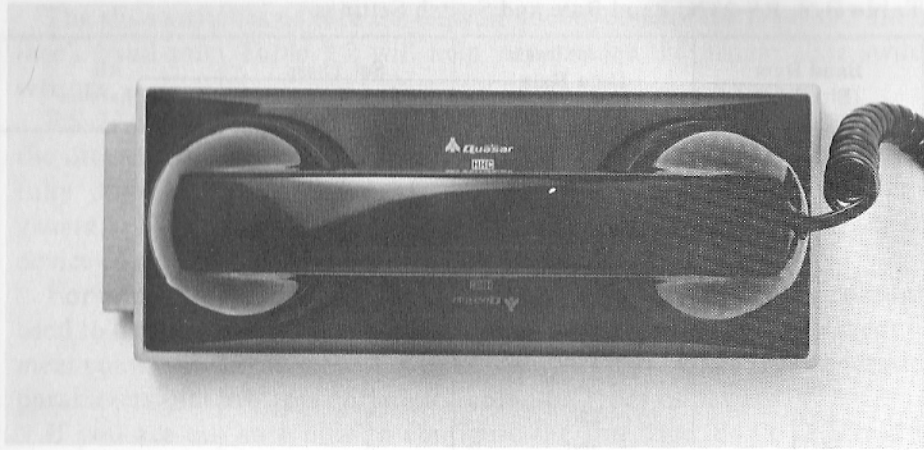
After rereading the instructions very carefully, make sure the proper software is running on the HHC and on your companion computer and then make sure you are using the software correctly. When everything seems to work right, but no data cross the RS-232C Interface, checking your software is a particularly important task.

If these rules of thumb don't help, you will need experienced technical assistance.

## The Modem

The Modem enables your HHC to communicate with another modem-equipped computer over an ordinary telephone line. A modem can be an *acoustic coupler* or a *direct-connect modem*. The HHC's Modem peripheral is an acoustic coupler. Shown in Figure 9-12, the HHC's Modem is often useful in portable computing applications because it can connect the HHC to a larger host computer. With this setup you can *upload* information from the HHC to the host or *download* information from the host to the HHC. Uploading and downloading are discussed in Chapter 8.

The Modem is available with a Cassette Interface that lets you transfer HHC files to and from magnetic tape using an audio cassette recorder. The Cassette Interface peripheral is described later in this chapter.



**Figure 9-12.** Acoustic Coupler Modem with phone handset

To prepare the Modem for use, you must have a device capsule intended for the Modem—for example, the Telecomputing 1 and Telecomputing 2 capsules.

Remove the sliding panel on the bottom of the Modem and insert your device capsule in the socket beneath the panel.

### Configuring the Modem

On the bottom of the Modem is a slide switch with settings labeled TEST and NORMAL.

The TEST setting connects the Modem's output circuit directly to its input circuit. To test whether the Modem's digital circuitry is in operating condition, set the TEST/NORMAL switch to TEST, connect the Modem to your HHC and turn it on, and then run any communications program that uses the Modem in its "terminal mode." Telecomputing 2 is a good example. The HALF/FULL DUPLEX configuration option must be set to FULL.

When you are ready to use the Modem, the switch should be set to NORMAL.

Finally, type on your HHC's keyboard. Whatever you type should appear on the LCD, just as though the HHC had received it over the Modem. Unfortunately, because the TEST switch does not test the Modem's speaker, microphone, or filter circuits, there is no absolute proof that the Modem is working properly. The ultimate test is use.

The Modem has a number of parameters that must be set properly if it is to communicate successfully. All of these parameters are set through software, generally by a CONFIGURE menu item in an application program in the device capsule.

For lists of configuration parameters appropriate for communicating with most common types of computer systems, see the user's manual for the capsule you are using.

If you are not sure how to configure the Modem, don't be afraid to experiment. You cannot hurt the Modem or your companion computer by using incorrect parameters. The worst thing that can happen is that the other computer will fail to "hang up the phone" after one of your attempts to communicate, and you will have to ask a host-site operator to make it "hang up" so that you can try again.

The Modem, unlike the RS-232C Interface, has a baud rate parameter that is established by software. Although most modern telecommunications systems run at 300 baud, the Modem gives you a choice of 110 or 300 baud.

The exact procedure for using the Modem may vary with the device capsule you are using. This section shows you how to use a Modem with the Telecomputing 2 capsule, the one most commonly used with this device.

First, connect the Modem to the HHC. Then use the I/O menu to turn on the INPUT and OUTPUT sides of the Modem. At this point the Modem will produce a continuous, high-pitched whine called a *carrier signal*. By modulating the tone of this signal, a Modem transmits data.

Now let's prepare the HHC program to communicate over the Modem. In Telecomputing 2, start the program so that you see the CONNECT/CONFIGURE menu and select the CONNECT item. The carrier signal should stop.

Now dial the telephone number of the other computer on any telephone with a standard handset. If the other computer has an *auto-answer modem*, a device that "answers the phone" automatically, you should hear another carrier signal over the phone after a few rings. If the other computer does not have auto-answer, a person at the other end must pick up the phone when you call and push the handset into the modem, just as you are about to do at your end.

Push the telephone handset into the rubber cups (top of the Modem). The cord end should face away from the HHC or the I/O Adaptor, as shown in the diagram (top of the Modem). The HHC's ON LINE blip should go on when it detects the other computer's carrier signal. The word "CONNECT", displayed by Telecomputing 2 when you selected CONNECT from the "CONNECT/CONFIGURE" menu, should disappear, clearing the LCD.



You have now established communications between your HHC and your companion computer.

## Troubleshooting

There are some test procedures to try if your Modem does not work as you expected.

If the other computer never answers the phone or gives no response to the data you send, it may be out of order.

If the ON LINE blip does not go on when you press the telephone handset into the Modem's rubber cups, then you did not receive a carrier signal from the other computer, the TEST/NORMAL switch is not set to NORMAL, you inserted the handset backward, or your Modem is set to ANSWER, not ORIGINATE.

If the HHC displays a message like "CONNECTION LOST", or if nothing happens and you cannot hear a carrier signal over the telephone when you lift it out of the Modem, the other computer was probably unable to make connections with your Modem and "hung up the phone." Try again. Practice inserting the phone in the Modem so that you can do it promptly when you hear the carrier signal; make sure the HHC's Modem is turned on and in CONNECT mode before you start.

Check the configuration parameters that you set for the Modem—especially if you make the connection but the data you send are garbled in transmission.

Make sure you are running the proper software on the HHC and the companion computer and that you are using the software correctly.

If these test procedures don't help, you will need experienced technical assistance.

## Modem with Cassette Interface

The Modem can be purchased with an optional Cassette Interface, as shown in Figure 9-13, that lets you save HHC data on magnetic tape with an ordinary cassette recorder. Like the standard Modem, the Cassette Interface connects to the telephone system, as shown in Figure 9-14. The Cassette Interface provides a cheap, compact medium for backing up or transporting HHC files.

The operating principle of the Cassette Interface is simple: The modulated carrier signal normally fed to the Modem's "send" cup is instead fed to the cassette recorder. The cassette recorder saves this carrier signal—and the



Figure 9-13. Modem with Cassette Interface

information contained in it—on tape, just as it would save any other type of sound. Later, the cassette recorder plays back the carrier signal through the Modem, and the Modem detects the data just as if the carrier signal were coming over a telephone.

In order to save files on tape, you must have an HHC and a Modem with a Cassette Interface. You also need Telecomputing 2 or any other HHC

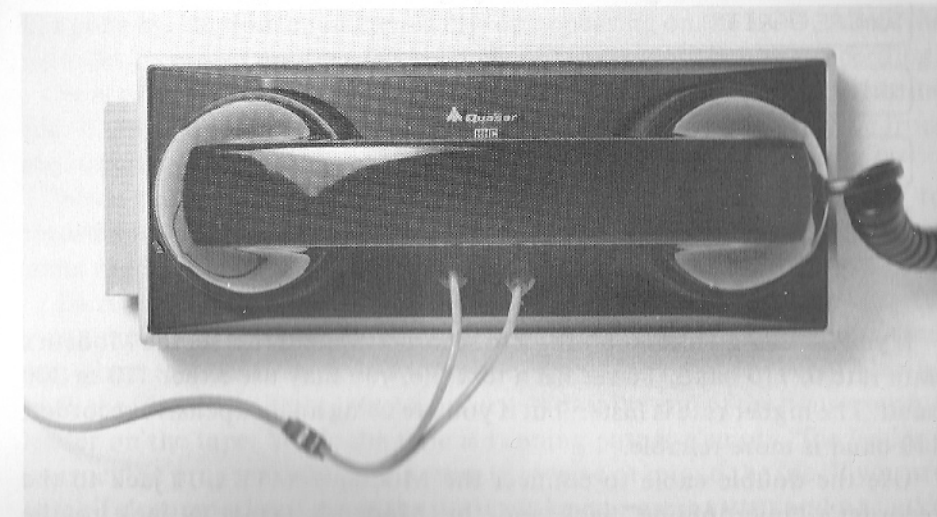


Figure 9-14. Modem with phone handset

program that can transfer a file from the HHC's File System to the Modem and back without expecting any sort of response. You need this kind of program because a cassette recorder by itself cannot respond to incoming data.

Finally, you need a cassette recorder with a jack for an external microphone and a jack for earphones. A double cable that is supplied with the Modem connects the two jacks on the Modem to the cassette recorder's microphone and earphone jacks.

To prepare the Modem for use with the Cassette Interface, connect the Modem to the HHC, and turn it on with the I/O menu.

Now set the following configuration parameters:

HALF DUPLEX

300 or 110 BAUD

8 DATA BITS

MASK 1ST BIT

2 STOP BITS

IGNORE PARITY

ORIGINATE MODE

NO XON/XOFF

VIDEO LF OFF

If you are using Telecomputing 2 to record a nontext file, set the Modem's data rate to 110 baud. To record a text file, you may use either 110 or 300 baud. The higher rate is faster; but if you are using an inexpensive recorder, 110 baud is more reliable.

Use the double cable to connect the Modem's SAVE OUT jack to the recorder's "microphone" jack and the Modem's LOAD IN jack to the recorder's "earphone" or "monitor" jack. When you plug the cable into both

jacks, the Modem's audible carrier signal stops. At this point the Modem directs its output to the SAVE OUT jack.

Now turn on the recorder's POWER switch and adjust the recorder's controls. Set the recorder's "volume" or "playback volume" control to a level you would use to play back normal conversation. If the recorder has a separate "record volume" control, set it to a level that you would use to record normal conversation. If the recorder has a MUSIC/TALK switch, set it to MUSIC.

## Saving and Loading Files

To save a file with the Cassette Interface, insert a tape cassette in the recorder and rewind the tape to its beginning. Place the HHC in CONNECT mode. At this stage the HHC may begin receiving random characters from the recorder and displaying them on the LCD.

Prepare the program you are running to save a file on the recorder—that is, perform all steps necessary to save a file except the last.

If you are using Telecomputing 2, press the SEND key to save a text file or press SHIFT and SEND to save a nontext file. The SEND key is actually the ROTATE key relabeled with the Telecomputing 2 keyboard overlay. Inspect the menu of sendable files that SEND displays and verify the presence of the file you want to save.

Make the recorder begin recording. Wait about ten seconds while the tape's nonrecording leader is going by the recording head. When the leader has gone by, random characters will stop appearing on the LCD. Allow the recorder to record another few seconds of "file leader" before proceeding.

Now perform the last step necessary to make the program begin saving the file. If you are using Telecomputing 2, select the file you want to save from the program's menu.

When the program has finished saving the file, allow the recorder to record another ten seconds or so of "file trailer." Then stop the recorder and press the HHC's CLEAR key.

To reload a file through the Cassette Interface, insert the tape cassette containing the file in the recorder and rewind the tape to its beginning. After you place the HHC in CONNECT mode, make the recorder begin playing back the file and wait until the recorder gets past the end of the nonrecording leader on the tape. While the tape is running past the empty "file leader," instruct the program you are running to receive or reload the file. If you are using Telecomputing 2, press the RECEIVE key; or press SHIFT and RECEIVE to reload a nontext file. The RECEIVE key is actually the → key, relabeled



with the Telecomputing 2 keyboard overlay. Type the name you want to give the reloaded file, and press ENTER. The LCD cursor disappears and the program begins waiting for the file.

When the LCD cursor reappears, signaling that the program has finished reloading the file, press the HHC's CLEAR key and stop the recorder. You should perform these steps while the recorder is playing back the empty "trailer" that you recorded at the end of the file.

You may have to experiment with the recorder's volume setting to make the Cassette Interface work reliably. To get the most reliable operation, experiment until you find the highest and lowest setting at which the Cassette Interface works and then set the volume midway between those points.

When you record important data, always play back the recording to check it for correctness. Like telecomputing, cassette recording of data is somewhat prone to errors.

### Several Files on One Tape

The Cassette Interface needs very little tape to save a typical HHC file. At 300 baud the Interface takes about 2 minutes and 15 seconds to record a 4000-byte text file, while most cassette tapes can record for 30 or 45 minutes per side. If you record only one file per tape, you are wasting a lot of tape.

You can make more economical use of your tapes by recording several files on each tape. Simply wind the tape to a position after the last saved file when you are ready to save another file.

Use your recorder's tape meter to keep track of the location of each file. To identify each file you may wish to record an introductory voice message that describes the file's contents, the date the file was saved, and other important information.

Table 9-3 summarizes characteristics of the RS-232C Interface and Modem communications devices.

## DISPLAY DEVICES

The TV Adaptor is a display device designed to give you large-screen editing, text processing, and graphics capabilities.

### The TV Adaptor

The TV Adaptor, also known as a *video adaptor*, enables your HHC to display information on a television set or a *video monitor* like the set shown

**Table 9-3.** Communication Devices' Characteristics

	RS-232C Interface	Modem
Communicates with	Other RS-232C devices, e.g., printers terminals, modems	Other computers or devices, by phone; cassette recorders
Device capsule slots	1	1
Speed	50, 75, 110, 150, 200, 300, 600, 1200, 2400, 4800, and 9600 baud (approximately 5 to 960 char/second), switch selected	110 or 300 baud (approximately 11 or 30 char/second) software selected
Word wrap	No	No
Other parameters	Full/half duplex 5 to 8 data bits 1 or 2 stop bits Odd/even/no parity	Full/half duplex 5 to 8 data bits 1 or 2 stop bits Odd/even/no parity Originate/answer mode
Other features		TEST/NORMAL switch
Size (W × H × D)	113.5 × 61 × 95mm (4-1/2 × 2-13/32 × 3-3/4")	227 × 56.5 × 95mm (8-15/16 × 2-7/32 × 3-3/4")
Weight	350g (12.35 oz)	550g (1 lb 3.4 oz)
Power source	HHC main unit; draws 650 mW	HHC main unit; draws 130 mW

in Figure 9-15. The TV Adaptor produces an instantaneous, noiseless display and consumes no paper. It can display as many as 16 lines of 32 characters, including all of the HHC's special LCD characters. The TV Adaptor can also duplicate all of the LCD's blinking and inverse-image characters and other special display modes and can display a complete set of Katakana (Japanese alphabetic) characters. Appendix D describes some of these characters. With appropriate software and a color television set or monitor, the TV Adaptor can display *graphics* and other color images. Text and graphic output can be combined in some of the graphic modes.

Table 9-4 summarizes the characteristics of the TV Adaptor.

### Television Set or Monitor

You can use the TV Adaptor with either an ordinary television set or a video monitor designed specifically to display computer images. Since most

people already own television sets, they have the advantage of economy.

The TV Adaptor modulates its image into a carrier signal, just as though it were a television transmitter. The TV Adaptor feeds the signal into your television set through its antenna connection.

A video monitor offers a clearer image than a television set, since it feeds the TV Adaptor's image-forming signal directly into its display tube's driver circuits. This sequence eliminates two intermediate steps—modulating the image into a carrier signal in the TV Adaptor and then demodulating it in the display device.

Video monitors are made in color, black-and-white, and "green screen" versions. A "green screen" is like black-and-white, but displays a green image instead of a white one. Many people find green screens sharper and easier to look at for long periods.

Most popular video monitors have 9- or 11-inch screens and are sold at



Figure 9-15. The TV Adaptor and HHC with a video monitor

Table 9-4. Memory and Display Devices' Characteristics

	Programmable Memory Peripheral	I/O Adaptor	TV Adaptor
Function	Expands HHC's RAM	Connects up to 6 peripherals to HHC	Communicates with television set or video monitor
Characteristics	4K, 8K, or 16K non-volatile RAM storage	Large tray holds HHC and 2 peripherals. Detachable small tray holds 3 more peripherals. Top socket connects 1 peripheral by cable.	Text mode: 16 lines $\times$ 32 characters, 5 $\times$ 7 dots/char, has word wrap Graphics mode: 32 $\times$ 64, 48 $\times$ 64, or 64 $\times$ 128 pixels  Colors: green, yellow, blue, red, buff, cyan, magenta, and orange
Weight	260 (9oz) with batteries	1100g (2 lb 6.8 oz) with small tray attached	227 $\times$ 30.5 $\times$ 95mm (8-15/16 $\times$ 1-3/16 $\times$ 3-3/4")
Size (W $\times$ H $\times$ D)	113 $\times$ 30.5 $\times$ 95mm (4-1/2 $\times$ 1-3/16 $\times$ 3-3/4")	425.5 $\times$ 51 $\times$ 318 mm (16-3/4 $\times$ 2 $\times$ 12-7/32")	500g (1 lb 1.64 oz)
Power source	HHC main unit; draws 20 mW. In 4K and 8K PMPs, 3 replaceable AAA cells maintain memory when device is disconnected or turned off. 16K PMP's use long-life lithium batteries that must be replaced at an HHC service center.	HHC main unit; draws 20 mW	Large AC Adaptor only; draws 1.5W

computer stores and television stores. They cost about as much as television sets of similar size.

Video monitor signals are completely standardized, so you need not be concerned about TV Adaptor and monitor incompatibility.

Some modern television sets have image input jacks that let them double as video monitors.



To use the TV Adaptor with a television set, turn your TV Adaptor so that the bottom faces you and then locate the SLIDE switch labeled CH3/CH4. This switch selects the television channel on which the TV Adaptor will "transmit." If one of these channels is used by a television station in your area, choose the other one.

Connect the TV Adaptor to your HHC.

Packed with your TV Adaptor is a cable with a coaxial-cable plug at one end and a pair of spade lugs at the opposite end. Screw the plug end of this cable into the TV Adaptor jack labeled RF OUT. Attach the opposite end to the television set's antenna terminals.

Switch on the television set and turn the selector to either Channel 3 or Channel 4 to match the TV Adaptor's channel selector switch.

To use the TV Adaptor with a video monitor, connect the TV Adaptor to your HHC.

Packed with the TV Adaptor is a cable with an RCA pin plug at each end like the ones used to hook up stereo components. Plug one end of this cable into the TV Adaptor jack labeled VIDEO OUT and plug the opposite end into the input jack on your video monitor. If your monitor's only input is a coaxial-cable socket, you will need an adaptor to connect the pin plug to the coaxial socket.

If your monitor has a DATA/PICTURE switch, set it to DATA. If your monitor has an IMPEDANCE switch, set it to the HIGH position. Your monitor has an instruction book with additional information.

## Power Requirements and Operation

The TV Adaptor runs only with the HHC's large AC Adaptor. It has no built-in batteries, and the small AC Adaptor cannot meet its power requirements.

If you try to run the TV Adaptor without the AC Adaptor, the TV Adaptor will not appear in the I/O key's menu.

To use the TV Adaptor, turn on the monitor or television set, then your HHC, and finally the TV Adaptor with the I/O menu. The background color lights up on your TV or monitor.

Select the File System editor or any other program that uses the TV Adaptor for output. An image of the program's output should appear on the screen. Adjust the controls of the monitor or television set to get the best possible image. If you are using a television set, first adjust the fine tuning control.

Information required to write a program that uses the TV Adaptor's graphics capabilities is presented in Appendix E.

## NEW DEVELOPMENTS

An EPROM Burner peripheral that plugs directly into an HHC peripheral socket is under development and should soon be available commercially.

This EPROM Burner will accept 2K and 4K EPROM capsules without the adaptor required by other EPROM burners and will be accompanied by a utility program for burning, reading, verifying, and modifying HHC capsules or device capsules.

Keep in touch with your HHC dealer for information about this and other new products.

## SUMMARY

To use a peripheral, you must

- Turn off the HHC
- Connect the peripheral to your HHC's peripheral socket
- Turn on the HHC
- Turn on the peripheral with the I/O key and the I/O menu
- Run a program that can operate the peripheral
- Enter the specific commands (if any) needed to make the program use the peripheral.

To connect or disconnect a peripheral you must observe the following rules and suggestions:

- Turn your HHC off before connecting or disconnecting anything.
- Return your HHC to the primary menu first, or the operating system will automatically return it to the primary menu. Since you cannot connect a peripheral while you are running a program, connect all the peripherals you will need before you start the program.
- Since your HHC's operating system turns all peripherals off whenever you connect or disconnect one, turn them back on with the I/O key if you still need them.

Observe the following rules when you turn a peripheral on or off:

- Turn peripherals on and off with the I/O key and the I/O menu. You can turn peripherals on or off at any time without disturbing the program that is running.
- If a peripheral can perform both input and output tasks, the I/O menu lists it twice—once for input and once for output. Remember to turn both menu items on or off.
- When you connect or disconnect one peripheral, the HHC's operating system turns all peripherals off.
- The CLEAR, ON, and OFF keys do not affect the on or off status of a peripheral. If you turn the HHC off, all the peripherals will become inactive. When you turn the HHC on again, the peripherals that were on before will go on again.

# A

## HHC Quick Reference

This appendix is a quick reference to such File System program editing features as cursor motion, inserting, deleting, canceling, reviewing, interrupting, and searching (Table A-1), to the Calculator (Table A-2), and to the Clock/Controller (Table A-3). For tutorials on these subjects, consult Chapters 3, 4, and 5, respectively.



**Table A-1.** File System Editing

Cursor Motion	
To move cursor one character/line:	Press:
left	←
right	→
up	↑
down	↓
To move cursor to the extreme:	Press:
left	LOCK ←
right	LOCK →
up	LOCK ↑
down	LOCK ↓
Insert and Delete	
To insert character(s) or space(s):	Press:
one character	INSERT x
one space	INSERT ←
one space, after advancing the cursor characters	INSERT →
three spaces left from cursor	LOCK INSERT xxxxx INSERT
three spaces right from cursor	LOCK INSERT ← ← ←
an indefinite number of spaces left from cursor	LOCK INSERT → → →
an indefinite number of spaces right from cursor	LOCK INSERT LOCK →
To delete:	Press:
the character under the blinking cursor and move the cursor one space to the left	DELETE ←
the character under the blinking cursor	DELETE →
three characters left from cursor	LOCK DELETE ← ← ←
three characters right from cursor	LOCK DELETE → → →
an indefinite number of characters left from cursor	LOCK DELETE LOCK ←
an indefinite number of characters right from cursor	LOCK DELETE LOCK →
To insert lines:	Press:
one, above current line	INSERT ↑
one, below current line	INSERT ↓
three, above current line	LOCK INSERT ↑ ↑ ↑
three, below current line	LOCK INSERT ↓ ↓ ↓

**Table A-1.** File System Editing (continued)

To insert lines (continued):	Press (continued):
an indefinite number, above current line	LOCK INSERT LOCK ↑
an indefinite number, below current line	LOCK INSERT LOCK ↓
To delete lines:	Press:
one, line above becomes current	DELETE ↑
one, line below becomes current	DELETE ↓
three lines, from current line up	LOCK DELETE ↑ ↑ ↑
three lines, from current line down	LOCK DELETE ↓ ↓ ↓
an indefinite number, from current line up	LOCK DELETE LOCK ↑
an indefinite number, from current line down	LOCK DELETE LOCK ↓
Cancel, Review, Interrupt, and Search	
To:	Press:
Cancel effect of INSERT	INSERT or DELETE
Cancel effect of DELETE	INSERT or DELETE
Review a line longer than LCD	ROTATE
Cancel effect of ROTATE	any typing or editing key
Interrupt a "locked" operation	any typing or editing key
Find "string" (up to 12 characters long)	SEARCH string ENTER

Table A-2. The Calculator

Valid Keys	
Key	Function
0 to 9	Digits
.	Decimal point
%	Percent
CM	Clear memory
RM	Restore memory to display, replacing last displayed number
M+	Add last displayed number to memory
M-	Subtract last displayed number from memory
+ - × ÷	Add, Subtract, Multiply, Divide
=	Equals; completes calculation
← →	Move cursor left or right
↑	Clear entry or result

Calculations	
If you press:	The result is:
$a + b =$	Add $a$ and $b$
$a - b =$	Subtract $b$ from $a$
$a \times b =$	Multiply $a$ by $b$
$a \div b =$	Divide $a$ by $b$
$a + b \%$	$a$ increased by $b \%$
$a - b \%$	$a$ decreased by $b \%$
$a \times b \%$	$b \%$ of $a$
$a \div b \%$	$a$ is $b \%$ of what?

CM	Clear memory
RM	Restore memory; replace last number on display
M+	Add last number on display to memory
M-	Subtract last number on display from memory

Clear		
To clear:	Enter this:	To avoid clearing, enter this:
Last entry or result	↑	Anything else
=	CLEAR, or next number	← →
+ - × ÷	CLEAR	Next number
Memory	CM	Anything else

Table A-3. The Clock/Controller

Function:	Procedure:	To cancel function:
Set Alarm	Choose option 1 Set time; press ENTER Type message; press ENTER	Press CLEAR
Review Alarms	Choose option 2 For next message, press ↓ For preceding message, press ↑ For time, press ← To redisplay message, press →	Press CLEAR
Delete Alarm	Choose option 2 Display alarm's message or time Press DELETE, then ↓, ↑, or CLEAR	Don't press DELETE at this point
Acknowledge Alarm	Choose option 3 Display alarm's message or time Press DELETE, then ↓, ↑, or CLEAR	Don't press DELETE at this point
Display Time	Choose option 4 When done, press CLEAR	Press CLEAR
Set Time	Choose option 5 To move cursor, press ← or → key Type in desired time Press ENTER	Press CLEAR



# B

## Cautions

This appendix summarizes guidelines for avoiding damage to the HHC and its peripherals. Some sources of damage are physical and electrical, while others involve software.

### PHYSICAL/ELECTRICAL HAZARDS

The printers and the Four Color Plotter are particularly susceptible to damage from improper handling.

To avoid damaging its mechanism, never operate the Plotter until all four pens are in place. When you are changing pens or paper, never rotate the turret by hand or touch any part of the mechanism except in accordance with the procedure described in Chapter 9. The owner's manual also has instructions for safe and careful handling of the Plotter turret mechanism.

The Mini Printer and Micro Printer also have delicate moving parts, particularly in the paper loading area.

The Plotter and TV Adaptor have high power requirements. Do not attempt to operate the TV Adaptor without the large AC Adaptor attached. Although this will not damage it, it will not work.

A major source of electrical damage to the HHC is improper use of AC Adaptors. Many AC Adaptors can be set for either 110 or 220 volt power. Be sure that you set your AC Adaptor to match the voltage your electrical

outlet provides. If you set the AC Adaptor to the wrong voltage, you may damage it severely.

Although you need not purchase the AC Adaptor designed specifically for the HHC, your adaptor must deliver 9 volts at 300 milliamperes with negative voltage on the center pin. Some inexpensive adaptors have poor voltage regulation, which can damage the HHC and its peripherals. Damage caused by using the HHC with an AC adaptor not designed for it will not be covered by the HHC's warranty.

Remember to turn off the HHC before physically connecting or disconnecting any peripheral. You may damage the HHC or a peripheral if the HHC is on when a peripheral is connected or disconnected.

## SOFTWARE

Your files can be endangered during three main periods: while you are editing, while you are running programs, and while you are storing programs and files in the PMP.

Since LOCK INSERT and LOCK DELETE stay on until you cancel them, remember to cancel LOCK DELETE and LOCK INSERT when you have finished using the File System editor. If you leave them on, you may inadvertently add or delete characters or whole lines in your file.

Except in two cases, you press the CLEAR key to reset the HHC after running a program. When you have Microsoft BASIC or SnapBASIC capsules installed in the HHC, however, do not press the CLEAR key or you will damage the files in the current file space.

With other programs, you should wait a few seconds after entering the last program operation before pressing the CLEAR key to allow the program to store its data.

The files stored in a Programmable Memory Peripheral (PMP) may be lost when you change batteries. Remember, the PMP must have some source of power for its memory chips at all times. When it is not receiving power from its own batteries, the PMP must be connected to the HHC.

First, connect the PMP to the HHC. Make sure that the HHC/PMP combination is held securely in a plastic sleeve or other holder. Then remove the PMP's batteries and insert new batteries.

If you lose the HHC and PMP connection and the PMP is without batteries, all the files you stored in the PMP will be gone forever.

# C

## Internal Hardware And Software

This appendix contains technical information primarily of interest to programmers and hardware engineers who may develop new software and peripherals for the HHC.

### HHC HARDWARE

The HHC includes a 6502 microprocessor with a 1 MHz clock rate, intrinsic RAM, bus interface, keyboard, LCD display, and batteries in one integral unit.

### The HHC Memory Map

The HHC uses bank switching to address more than 64K of memory through its 64K address space. Table C-1 presents a memory map of the HHC. The capsule space (4000H to 7FFFH) is bank-switched among the three HHC capsule sockets on the back of the HHC. The HHC's operating system also uses this space to store data for memory-mapped I/O. The extrinsic RAM space (8000H to 0BFFFH) is bank-switched to the peripheral socket containing the currently selected PMP, if any. The device control space (2000H to 3FFFH) is bank-switched to a device control capsule



**Table C-1.** Memory Map of the HHC

Start	End	Amount	Type	Purpose
0H	1FFFH	8K	Intrinsic	Operating system and program data storage; File System storage
2000H	3FFFH	8K	Intrinsic and extrinsic	Device control capsules for peripherals; memory-mapped I/O
4000H	7FFFH	16K	Extrinsic	HHC capsules
8000H	0BFFFH	16K	Extrinsic	Programmable memory peripherals
0C000H	0FFFFH	16K	Intrinsic	Operating system and intrinsic applications

during I/O operations and during execution of an application program that resides in a device control capsule.

Normally, all bank-switching operations are handled by the operating system. This process of bank-switching is transparent to the application program.

### Proprietary Parallel Bus

The HHC uses a proprietary parallel bus with a 44-pin edge connector. Since the HHC's peripheral socket connects directly to the bus, all HHC peripherals run directly off the bus.

The HHC's processor signals are not buffered; they connect directly to bus pins. This arrangement lowers cost, size, and power consumption, but it means that only one peripheral can be connected directly to the HHC main unit.

The eight data bus lines from the 6502 microprocessor are connected directly to the bus. Of the 6502's 16 address lines, 14 (A0 to A13) are connected to the bus and are sufficient to address 16K, the largest contiguous region of RAM or ROM that appears in the HHC's address space.

Most of the 6502's control signals appear on the bus. They include the two interrupts, the clock,  $R/\overline{W}$ ,  $\overline{PS}$ , and SYNC.

The HHC has a sophisticated power supply and power control circuit. Four separate power lines present on the bus include unregulated supplies from the main unit's battery and AC Adaptor, a regulated supply for CMOS ICs, and a +5 volt regulated supply present only when the CPU is running.

The three bus pins used to protect the bus from damage make contact first because their connector pins are longer than the others. These three pins activate the 6502's RESET line, suspend all bus activity, and cause a warning beep. When the connector is fully seated, the protection is released so that bus activity can resume. The bus protection mechanism is also activated if a connection becomes loose or a peripheral is disconnected.

HHC distributors make complete bus specifications available on a non-disclosure basis to qualified parties interested in developing new HHC hardware. Service manuals for the HHC and each HHC peripheral are available on the same basis.

### The HHC Keyboard

The HHC keyboard has 65 positive action keys, 53 of which are readily usable by application programs. The SHIFT and 2nd SFT keys provide access to the entire printable ASCII character set and are software-lockable. Most of the keyboard keys auto-repeat at a rate controlled by the STP/SPD (stop/speed) key.

All of the keyboard's accessible keys may be software-redefined to generate any 8-bit character code.

### The LCD

The LCD displays 26.5 characters on a  $159 \times 8$  dot matrix. With each character displayed within a  $5 \times 7$  matrix, the top row of dots is reserved for floating accent marks.

The operating system software handles character generation. You can define a 256-character alternate character set through software. Once it is defined, an application may select the standard or the alternate character set for each character of output. The standard character set may also be redefined by software.

You can display any character from either character set with any combination of flashing, inverse image, and invisible attributes.

Floating accent marks may be displayed over any character and may be defined by software.

The LCD's standard cursor may be software-redefined and may also be suppressed.

The LCD may be used as a sequential device in which characters rotate off the left edge of the LCD as they are displayed at the right, or as a random device in which any character may be displayed in any position.

You can use the LCD as a  $159 \times 8$  dot-addressable graphics device.

The eight LCD blips are software-controllable, although five of them are used by the HHC's operating system in ways that are likely to conflict with applications use.

The "squaker," a sound generator associated with the LCD, can generate tones at 36 different pitches and of any desired duration.

### Battery Power

The five rechargeable nickel-cadmium AA size batteries that power the HHC are intended to be permanent; only a service technician should replace these batteries.

Low-power HHC peripherals draw power from the HHC batteries through the HHC bus. Higher-power peripherals contain their own rechargeable batteries that recharge from an AC Adaptor through the HHC bus. The TV Adaptor has no batteries and requires an AC Adaptor to operate.

Designed to maximize power conservation, the HHC's operating system makes extensive use of hardware interrupts to power each hardware component only when necessary. When an application program is waiting for the user to type input on the keyboard and the cursor is not flashing, for example, power is applied to the LCD and to RAM, but the non-CMOS, high-power CPU and all ROMs are turned off. Each keystroke generates an interrupt that turns on the CPU. After it processes the keystroke, the CPU turns itself off again.

## SOFTWARE

Most of the HHC's operating system and application software is written in SNAP, a FORTH variant designed specifically for the HHC. SNAP is the language of choice for demanding applications—the large, complex programs that must run efficiently and that use many of the operating system's more esoteric features.

A SNAP inner interpreter, which interprets compiled code, is built into the nucleus of the HHC's operating system. An outer interpreter, which compiles source code, is built into the HHC SnapFORTH capsule.

Compiled SNAP code is typically about 30% more compact than equivalent machine code and 20 to 25% as fast. SnapFORTH contains a 6502 assembler that can be used to rewrite selected parts of SNAP programs in machine language for more speed.

For more general information on SNAP and SNAP programmers' tools, see the description of the SnapFORTH capsule in Chapter 8.

### The Operating System

When the HHC is powered up with the ALL-OFF switch and you press the CLEAR key, the HHC's operating system initializes itself, remaining in control until the ALL-OFF switch is turned off again or until a program bug causes a crash.

The OFF key does not cause the operating system to lose control of the HHC. In fact, it does not even turn off the HHC in the conventional sense. Rather, software directs the operating system to turn off power to the LCD and CPU. Then interrupts can turn the CPU on again.

The HHC's operating system makes extensive use of interrupts, and all peripheral I/O and keyboard I/O are interrupt-driven.

A software control block, called an *event control block* (ECB), manages interrupts. The initiating routine assigns an ECB to an event and, directly or indirectly, calls the operating system's event handler. After the event handler clears the first byte in the ECB and places the ECB in a queue, the initiating routine can continue with its processing or turn the CPU off.

When the event associated with the ECB occurs, it generates an interrupt. The interrupt handler is executed, associates the event with the proper ECB, and *posts* the ECB by turning on the high-order bit in the first byte. The initiating routine can then take whatever post-event action is appropriate.

A character-input operation, for example, is initiated by a RIP (Request Input) call to the operating system with an ECB as a parameter. Immediately after the RIP, control returns to the initiating routine, which can do other processing until it needs the character that is being read. When the routine can proceed no further without the input, it makes a WAIT call to the operating system, which turns off the CPU if no character is ready. Whenever an interrupt occurs, the operating system turns on the CPU and the event manager processes the event. When the requested character arrives, the event manager posts the ECB and returns control to the operation following WAIT in the initiating routine.

The operating system maintains a 5-byte "time of century clock" that is based on a 3-byte timer in the HHC hardware. At 18°C the clock is accurate to  $\pm 15$  seconds per month. Over its rated temperature range of 0°C to 40°C, it is accurate to  $\pm 28$  seconds per month.

Application programs can query the time of century, and can also set timed delays with various operating system calls.



## I/O Operations

Most I/O operations are performed through a pair of system calls. The first call initiates the operation—either RIP (Request Input) or ROP (Request Output). A second call, WAIT, halts processing until the operation is complete. With programs written in SNAP, you can easily program a WAIT on multiple ECBs. Thus, you can, for example, wait for the completion of an I/O operation on any of several devices.

I/O is largely device-independent. A system call addresses a device by a *logical unit number* (LUN) bound to the device at run time, so that a program may be written to send output to a Printer, a Modem, or a TV Adaptor without concern for which device will actually be attached at run time. Physical differences among devices are accommodated by routines that are located in each peripheral's device driver capsule. The operating system's use of these routines is transparent to the application program.

Like peripherals, the keyboard and LCD may be manipulated through ECBs and LUNs. When device independence is not needed, these devices can be addressed with special I/O operations that are more convenient to use. These operations support all of the devices' special functions—for example, special display attributes and dot matrix graphics on the LCD.

Programs that need line-oriented input may use EXPECT, a very powerful line-input system call that supports all the line editing functions of the HHC's File System editor. EXPECT not only provides a convenient way for a programmer to implement powerful line input, but also presents the user with a consistent user interface for all HHC programs that read lines of data from the keyboard.

The operating system maintains a 16-character buffer for the keyboard, making it possible for the user to type ahead of any program. Most input peripherals have similar buffers.

It is possible, though with some restrictions, for a program to redirect keyboard input and LCD output to a peripheral by binding the keyboard's or LCD's LUN to another device. Redirected I/O cannot support the devices' special functions, and the redirection is canceled the next time you press the CLEAR key.

## The File System

The HHC's File System application program is based on a file management routine in the operating system that is also referred to as "the File System."

The File System routine simulates a random access file structure with non-volatile RAM. Application programs can call it to create, delete, and locate files; to read, write, insert and delete records; to compute the amount of free RAM in the file space; and for various other purposes.

Files may be stored in intrinsic RAM, extrinsic RAM, or both. Extrinsic RAM means a Programmable Memory Peripheral or PMP. One RAM area, intrinsic or extrinsic, is accessible to a program at a time. You switch from one RAM area to the other with the I/O key or through a program.

## Floating Point Operations

The operating system includes a complete set of system calls for floating point operations, including the four arithmetic functions, formatted output, and rounding. The internal floating point format uses a 52-bit, decimal-base mantissa that provides 13 decimal digits of precision. The range of the exponent is approximately  $10^{\pm 1024}$ .

The Scientific Calculator capsule contains the trigonometric functions and several other additional floating point operations that can be called as library routines if the calling program and the Scientific Calculator capsule are in the HHC at the same time.

# D

---

## Keyboard And LCD Codes

This appendix lists and discusses the HHC's character set, a collection of printable and control characters. The HHC's character set is an expanded version of the American Standard Code for Information Interchange (ASCII), a standard used by virtually all computers manufactured today. HHC variations from ASCII—almost all of them extensions—are shaded in the tables that follow.

Since a computer represents all data as numbers, each character in the HHC character set has a numeric value. This number is used to represent the character inside the computer.

The character set is divided into three major groups, each with separate numeric value ranges.

### Control Characters

*Control characters* have control functions such as ringing a bell or beginning a new line on an output device. Control characters have numeric values from 0 to 31.



Table D-1 lists each control character with its numeric value, keyboard key, and LCD symbol.

If a character has no entry in the LCD symbol column, the character displays an inverse image symbol on the LCD or TV. To determine which symbol, add 32 to the numeric value and consult Table D-2, "Printable Characters."

A character with an entry in the LCD symbol column may also display an inverse image symbol in certain special output operations.

### Printable Characters

*Printable characters* represent such symbols as A, 3, or !. They can be displayed on most output devices and entered on most input devices. Printable characters have numeric values from 32 to 126.

The printable characters are letters, numbers, punctuation, and common symbols found in English text. Table D-2 lists the numeric values for each keyboard key.

### Unique Characters

*Unique characters* are unique to the HHC's character set. Most unique characters are "printable," meaning that they can be displayed on the LCD and on some HHC peripheral devices, such as the TV Adaptor. Unique characters have numeric values from 127 to 142.

Table D-3 lists the numeric values and keyboard keys for the HHC's unique characters.

In standard ASCII, the value 127 represents the control character "delete" or "rubout," instead of the HHC's "insert" cursor. The ← key backspaces the cursor and the → key advances the cursor. Neither disturbs the character that was under the cursor. The values 137 and 138 represent the "standard" and "delete" cursors, respectively.

The value 136 represents the multiplication operator.

Table D-1, Control Characters

Numeric Value	HHC Keyboard Key	HHC LCD Symbol	ASCII Name and Function
0			NUL, Null
1			SOH, Start of heading
2			STX, Start of text
3			ETX, End of text
4			EOT, End of transmission
5			ENQ, Enquiry
6			ACK, Acknowledge
7	ROTATE	Bell	BEL, Bell. When sent to the LCD, makes the HHC "beep"
8		Backspace	BS, Backspace. Backspaces cursor and erases last character written to LCD
9			HT, Horizontal tab
10			LF, Line feed
11	I/O*		VT, Vertical tab
12			FF, Form feed
13	ENTER	New line	CR, Carriage return. Ends a line of input or output
14	STP/SPD*		SO, Shift out
15			SI, Shift in
16			DLE, Data link escape
17			DC1, Device control 1
18			DC2, Device control 2
19			DC3, Device control 3
20			DC4, Device control 4
21	HELP*		NAK, Negative acknowledge
22	f1*		SYN, Synchronous idle
23	f2*		ETB, End of transmission block
24	f3*		CAN, Cancel
25			EM, End of medium
26			SUB, Substitute
27			ESC, Escape
28			FS, File separator
29			GS, Group separator
30			RS, Record separator
31			US, Unit separator

\*Normally these keys have special control functions for the HHC. They cannot be read from the keyboard by an application program.

Table D-2. Printable Characters

Numeric Value	HHC Keyboard Key	Name	Numeric Value	HHC Keyboard Key	Name
32	Space		80	P	
33	!	Exclamation mark	81	Q	
34	"	Quotation mark	82	R	
35	#	Pound sign	83	S	
36	\$	Dollar sign	84	T	
37	%	Percent sign	85	U	
38	&	Ampersand	86	V	
39	'	Apostrophe	87	W	
40	(	Left parenthesis	88	X	
41	)	Right parenthesis	89	Y	
42	*	Asterisk	90	Z	
43	+	Plus sign	91	[	Left bracket
44	,	Comma	92	\	Backslash
45	-	Hyphen; minus sign	93	]	Right bracket
46	.	Period	94	^	Caret
47	/	Slash	95	_	Underscore
48	0		96	`	Grave accent
49	1		97	a	
50	2		98	b	
51	3		99	c	
52	4		100	d	
53	5		101	e	
54	6		102	f	
55	7		103	g	
56	8		104	h	
57	9		105	i	
58	:	Colon	106	j	
59	;	Semicolon	107	k	
60	<	Less than	108	l	
61	=	Equal sign	109	m	
62	>	Greater than	110	n	
63	?	Question mark	111	o	
64	@		112	p	
65	A		113	q	
66	B		114	r	
67	C		115	s	
68	D		116	t	
69	E		117	u	
70	F		118	v	
71	G		119	w	
72	H		120	x	
73	I		121	y	
74	J		122	z	
75	K		123	{	Left brace
76	L		124		Vertical bar
77	M		125	}	Right brace
78	N		126	~	Tilde
79	O				

Table D-3. Unique Characters

Numeric Value	HHC Keyboard Key	HHC LCD Symbol
127		☐
128	↑	↑
129	←	←
130	→	→
131	↓	↓
132	INSERT	A M
133	DELETE	P M
134		☐
135		÷
136		×
137	SEARCH	■
138		□
139	C1	ä
140	C2	ø
141	C3	ü
142	C4	ñ



# E

## The TV Adaptor: A Programmer's Guide

This appendix contains information for programmers who want to write software for the HHC's TV Adaptor.

To understand this appendix you must know how to use a programming language and know how programs in that language perform I/O on the HHC. You must also be familiar with hexadecimal (number base 16) notation.

The information contained here has not been previously published.

The TV Adaptor operates in one of five modes. When you attach it to your program it is in *alphanumeric* mode. In this mode you can display ASCII characters on the screen and move the cursor with HHC cursor control characters, much as if the display were a multi-line LCD. Device-specific commands can change the display colors and perform editing for other functions.

The other four modes allow the TV Adaptor to display graphics. The *semigraphic 4*, *semigraphic 6*, *graphic 64*, and *graphic 128* modes differ both in the number and size of the graphic image elements they display on the screen and in the types of control they allow you to exercise over the contents of the display.

## COMMUNICATING WITH THE TV ADAPTOR

If you are running a BASIC program, the device code for the TV Adaptor is 67. Before running your program, connect the TV Adaptor to the HHC and turn it on with the I/O menu to establish communication with the TV Adaptor. In your program, use ATTACH to associate a logical unit number (LUN) with device code 67, and then write output to the TV Adaptor.

With PRINT, you have the usual choice between performing I/O by character, with PUT, or by line. PRINT is more convenient for writing alphanumeric output, but PUT is easier to use for graphics and for control operations that involve outputting nonprintable characters.

### In SNAP

If you are running a SNAP program, the device code for the TV Adaptor is 43H. Before running your program, connect the TV Adaptor to the HHC and turn it on with the I/O menu to establish communication with the TV Adaptor. In your program, perform an ATTACH to associate a logical unit number (LUN) with device code 43H, and then write output to the TV Adaptor.

You can do I/O in several ways. You can write one character at a time with ROP, the HHC's standard way of doing output. You can also write an entire block of ASCII data in a single operation. The block may be any length from 1 to 253 characters. This is an efficient way of changing the contents of the display. You set up an ECB as shown in Table E-1 and perform an RCTL to initiate the operation. Perform a WAIT or WAITM to complete it. Writing a block of ASCII data is equivalent to doing a ROP on each character in the block.

You can write a block of raw data directly to the TV Adaptor's display. Set up an ECB as shown in Table E-2 and perform an RCTL to initiate the operation. Then perform a WAIT or WAITM to complete it. When you write a raw data block, you bypass the TV Adaptor's usual filtering for command characters and escape sequences. This method is slightly more efficient than writing a block of data without bypassing the Adaptor's usual filtering. The format of a raw data block is quite different from the format of an ASCII data block. Raw data formats are discussed in the next section of this chapter.

You can also read a block of raw data directly from the TV Adaptor's display. First, set up an ECB as shown in Table E-2 and perform an RCTL

Table E-1. ECB for Writing a Block of ASCII Data

Byte	Contents
0	Traffic byte
1	Set to 01H
2	Set to length of block in bytes ( $1 \leq n \leq 253$ )
3 to 2+n	Data bytes 0 through n-1

Table E-2. ECB for Reading or Writing a Block of Raw Data

Byte	Contents
0	Traffic byte
1	Set to 2
2	Specifies the number of RAM pages to read from the video display RAM into the buffer or to write from the buffer to the video display RAM. A RAM page is 100H bytes long. Use a positive value to read; use a negative value to write.
3-4	Address of the buffer that receives the data (for a read) or contains the data (for a write).

to initiate the operation; then perform a WAIT or WAITM to complete it.

You can read the TV Adaptor's configuration parameters. Set up an ECB as shown in Table E-3 and perform an RCTL to initiate the operation. Perform a WAIT or WAITM to complete it. All fields following 0 and 1 are filled in, and initial values, if any, are lost.

All possible ECB return codes are standard device-independent ones with values 0 to 6.

## ALPHANUMERIC MODE

When you attach the TV Adaptor to your program, it runs in alphanumeric mode until you shift it to another mode with an escape sequence.

In alphanumeric mode the display consists of 16 lines, each 32 columns wide. In cursor positioning operations the lines are identified by numbers from 0, at the top, to 15, or 0FH, at the bottom. The columns are identified by numbers from 0, at the left, to 31, or 1FH, at the right. Each character is displayed with an  $8 \times 5$  matrix in a  $12 \times 8$  cell.



**Table E-3.** ECB for Reading TV Adaptor Configuration Parameters

Byte	Contents
0	Traffic byte
1	Set to 0
2	X position of cursor. In alphanumeric mode, by character; in semigraphic and graphic modes, by pixel
3	Y position of cursor, by character or pixel
4	Control nybble at cursor position (see section on "Raw Data")
5	Data byte at cursor position
6	Currently selected page (alphanumeric and semigraphic modes only) 0=page 1; 1=page 2
7	Default character
8	Control flags (meaningful only in alphanumeric mode) See Table E-7
9	Default control byte, as set by SETPRMS
0A	Current mode (alphanumeric, semigraphic 4, and so on) Meaning is the same as for the SETMODE escape sequence, see Table E-7
0B	Vertical measure of display. In alphanumeric mode, in lines; in semigraphic and graphic modes, in pixels
0C	Horizontal measure of display, in lines or pixels

The TV Adaptor maintains two *pages* of memory designated "page 1" and "page 2." Each page describes one screenfull of information. At any time, one of these pages is selected. Thus, output to the TV Adaptor affects its contents, and its contents are displayed on the screen. Page 1 is automatically selected when the TV Adaptor is first attached.

A program can select either page at any time, thereby enabling it to flip back and forth between two different displays by writing a single control sequence.

The TV Adaptor initially displays data with certain default attributes. However, you can change them with appropriate control operations. Data characters you send to the TV Adaptor are displayed in green on a black background. When the display or part of the display is erased, each affected position is filled with a blank, the default character. After displaying a character, the TV Adaptor moves the cursor one column to the right.

After displaying a character in the last column on a line, the TV Adaptor moves the cursor to the first position of the next line down. This is called *automatic line feed*.

When you attempt to move the cursor down from the last line of the display, the TV Adaptor scrolls the entire display up one line. The former top line is lost, and the bottom line is filled with the default character, a blank. This is called *automatic scrolling*. Word wrap is on, and the word delimiter is the "blank" character.

## Raw Data

Several of the TV Adaptor's I/O operations involve *raw data*. In order to handle raw data you must first be familiar with the way the TV Adaptor stores display information in alphanumeric mode.

The TV Adaptor stores display information in RAM that is addressed only by the TV Adaptor's device driver. There are 4K of this display RAM to store two pages.

Each page of the display RAM consists of 1024 pairs of bytes. Each pair stores information about one position on the display. The pairs are stored in RAM in English reading order: the top line from column 0 to column 31, then the second line from 0 to 31, and so forth.

The first, even-addressed byte in each pair is the *control byte*. This byte contains information about how data is to be displayed. The second, odd-addressed byte is called the *data byte*, and it contains information about what is to be displayed. The format of each byte depends on the current display mode.

The format for a pair of bytes in alphanumeric mode is shown in Table E-4, where 0 is the control byte and 1 is the data byte.

When an ASCII data byte is written to the display, the TV Adaptor's control routine translates it into a pair of bytes that is stored in the display RAM. The current foreground color, invert mode, and flash mode are used. The 8-bit ASCII data byte is translated into seven bits of raw data, an "internal/external" bit and an "invert/no-invert" bit, as shown in Table E-5.

## Foreground and Background

Many of the video display operations deal with "foreground" and "background" attributes of the display. "Foreground" refers to the attributes of a character or graphic display element written to the screen. "Background" refers to the attributes of the rest of the screen.

For example, when the screen is cleared, the entire display is given the background attributes. Similarly, when a line feed in alphanumeric mode makes an empty line appear at the bottom of the screen, the empty line is

**Table E-4.** Structure of a Display RAM Byte Pair Representing One Display Position in Alphanumeric Mode

Byte	Bits	Meaning
0		Control Byte
	80-10	Unused (actually not supported by hardware)
	08	0 = no flash, 1 = flash
	04	Unused
	02	0 = use internal character generator; 1 = external
1	01	Color: 0 = display green or black; 1 = display orange or yellow
		Data Byte
	80	Color inversion: 0 = display green or orange, 1 = display black or yellow
	40-01	Data

**Table E-5.** The TV Adaptor Translates ASCII Data to Raw Data

Character	ASCII Value	0 = Internal 1 = External	0 = No Invert 1 = Invert	Notes
^ @ - ^	00H - 1FH	0	1	
blank - _	20H - 5FH	0	0	Raw = ASCII - 20H
\ - end	60H - 0DFH	1	0	Raw = ASCII - 60H

shown with the background attributes. When a character is written to the screen, it is shown with the foreground attributes.

As an example, if the background color is green and the foreground color is orange, the display will show orange characters on a green background.

## Types of Output

The TV Adaptor recognizes (and treats differently) several types of output.

Printable characters, which are most of the characters from 20H up, are treated as *data*. Each data character written to the TV Adaptor displays one character on the screen in alphanumeric mode or one group of picture elements in any of the graphic modes.

Nonprintable characters, which are 01H or 1FH and the cursor control

characters (80H to 83H), are treated as *control commands*. Each valid control command performs a control operation on the TV Adaptor. These operations include carriage return, cursor movement, and inserting or deleting lines on the display.

Many control commands must be followed by one or more *operands*. Each operand is a byte of data that influences the way the control command is performed. Operands that specify numeric quantities, like the line and column numbers in a cursor positioning command, are always expressed in binary form. The three-byte sequence of binary values "29 10 12" puts the cursor in column 10, line 12.

Control sequences that begin with code 27 (1BH or ESC) are called *escape sequences*. These sequences set various configuration parameters of the TV Adaptor and, in this way, affect the display indirectly. Some escape sequences are device-independent. These are discussed in the chapter on peripheral I/O in the *SNAP Programmer's Reference*.

Every escape sequence consists of

1. Control code 27 (1BH or ESC)
2. A one-byte binary *opcode*, which determines the function of the sequence
3. One or more operands, which influence the way the operation is performed. The value of the opcode determines the number and meaning of the operand bytes.

At least one operand byte must always be present. In escape sequences that do not use any operands, the operand byte is ignored.

## Character Generators

The TV Adaptor generates alphanumeric characters from patterns stored in two special ROMs called *character generators*.

The *internal character generator* produces ASCII characters 20H through 5FH, which are numerals, capital letters, and most special characters. It is permanently installed in the TV Adaptor.

The *external character generator* produces ASCII characters 60H through 0DFH, which are lower-case letters, Japanese Katakana characters, and some other special characters. It may be replaced by a custom-programmed EPROM, making it possible to change this part of the TV Adaptor's character set.

The standard pair of character generators conforms to HHC Extended ASCII except that character 96 or 60H, the grave accent, displays as a blank.



## Control Commands and Control Sequences

Table E-6 lists control commands and control sequences for the TV Adaptor.

The column labeled "symbols" lists words that are used to refer to the control commands in the text. Some of these words are predefined to represent the control commands in SnapFORTH. Others may be defined in your program with the symbolic constant word, "`=`".

The symbol "`@`" represents the cursor; the cursor's initial position is (X,Y). In a multicharacter control sequence, the first character, the control character, is "`c0`". The second character is "`c1`", and so forth.

Table E-7 lists the SETMODE escape sequences for the TV Adaptor in alphanumeric mode.

The "symbol" column in Table E-7 has the same significance as it had in Table E-6, but it refers to the second character, the opcode, rather than to the first character, which is always ESC. The "operand bytes" column gives the number of operand bytes in the sequence. An "x" means that the sequence has no operands. The opcode must be followed by one operand byte, whose value is ignored. In the "meaning" column, the first operand is denoted by "`p0`", the second by "`p1`", and so forth.

## SEMIGRAPHIC AND GRAPHIC MODES

You enter any of the semigraphic and graphic modes by issuing the SETMODE escape sequence from alphanumeric mode or from another graphic or semigraphic mode.

Table E-8 summarizes the characteristics of the graphic and semigraphic modes.

The semigraphic modes support two display pages, just as alphanumeric mode does. The graphic modes support only one display page and use all of the TV Adaptor's display RAM to give a high-resolution display.

Alphanumeric, semigraphic 4, and semigraphic 6 data can be intermixed freely on the display. Graphic 64 or graphic 128 data must be displayed by itself; these modes may not be mixed with alphanumeric or semigraphic modes or with each other.

In any graphic or semigraphic mode, the display consists of a grid pattern of picture elements called *pixels*. Pixels are grouped into larger elements called *pixel matrixes*, which also form a grid pattern. In each mode, the

Table E-6. Control Commands in Alphanumeric Mode

Decimal Value	Hex Value	Symbol	Control Sequence Length	Meaning
8	08	BACKSPACE	1	Replaces the character at (X-1, Y) with the default character (usually "blank") and moves @ to (X-1,Y).
9	09	HTAB	1	Moves @ right to the next tab stop. If no tab stops are set to the right of column X, moves @ to (31,Y). (Tab stops are set and cleared with escape sequences.)
10	0A	LF	1	Moves @ to (X,Y+1). If @ is at (X,15), the display is scrolled or erased.
11	0B	CLREOP	1	Fills the display with the default character, beginning at (X,Y) and ending at (31,15).
12	0C	FORMFEED	1	Moves @ to (0,0). The entire display is filled with the default character.
13	0D	CR	1	If automatic line feed is enabled, moves @ to (0,Y+1); if @ is at (X,15), the display is scrolled or erased. If automatic line feed is disabled, moves @ to (0,Y).
14	0E	DUPLI	1	Copies the character displayed at (X,Y-1) to (X,Y). If Y=0, copies from (X,15).
27	1B	ESC	varies	Begins an escape sequence. Followed by an opcode and one or more operand bytes. (See Table E-7 for information about escape sequence opcodes and operands.)
29	1D	POSCUR	3	Moves @ to (c <sub>1</sub> , c <sub>2</sub> ). Does not affect the contents of the display.
30	1E	IDATA	2	c <sub>1</sub> is interpreted as a raw data byte with the current default control byte; it is displayed regardless of whether or not it is a data character. Note that the symbol a raw data character displays is not the same as the one it represents in HHC extended ASCII notation. (See the text, Table E-4, and Figures E-1 through E-4 for more information.)
31	1F	XYD	4	Moves @ to (c <sub>1</sub> , c <sub>2</sub> ), displays c <sub>3</sub> as raw data, and moves @ forward (or backward) one character in the usual manner. (See IDATA for notes on raw data.)
128	80	&U	1	Moves @ up to (X,Y-1). If @ is at (X,0), scrolls the display down one line; line 0 is filled with the default character.
129	1	&<	1	Moves @ left to (X-1,Y). If @ is at 0,Y moves it to (31,Y-1). Unlike BACKSPACE, does not modify the display.
130	82	&>	1	Moves @ right to (X+1,Y). If @ is at (31,Y), moves it to (0,Y+1). Unlike "blank," does not modify the display.
131	83	&D	1	Moves @ down to (X,Y+1). If @ is at (X,15), scrolls the display up one line; line 15 is filled with the default character.

Table E-7. Escape Sequences in Alphanumeric Mode

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning																								
0	00	SETCTRL	1	86H	Each bit of $p_0$ sets one configuration parameter. Note that many of these parameters can be set separately with other escape sequences.																								
<table><tr><th>Bit</th><th>Value</th><th>Parameter</th></tr><tr><td>0</td><td>80H</td><td>Same as SETCLEOP</td></tr><tr><td>2</td><td>20H</td><td>Same as SETTRANS</td></tr><tr><td>3</td><td>10H</td><td>1 = uninvert foreground color; 0 = invert. Same function as SETINV and UNSETINV</td></tr><tr><td>4</td><td>08H</td><td>1 = enable foreground flash; 0 = disable. Same function SETFLASH and UNSETFLH</td></tr><tr><td>5</td><td>04H</td><td>Same as SETSCRL</td></tr><tr><td>6</td><td>02H</td><td>Same as SETLF</td></tr><tr><td>7</td><td>01H</td><td>Same as SETCOL</td></tr></table>						Bit	Value	Parameter	0	80H	Same as SETCLEOP	2	20H	Same as SETTRANS	3	10H	1 = uninvert foreground color; 0 = invert. Same function as SETINV and UNSETINV	4	08H	1 = enable foreground flash; 0 = disable. Same function SETFLASH and UNSETFLH	5	04H	Same as SETSCRL	6	02H	Same as SETLF	7	01H	Same as SETCOL
Bit	Value	Parameter																											
0	80H	Same as SETCLEOP																											
2	20H	Same as SETTRANS																											
3	10H	1 = uninvert foreground color; 0 = invert. Same function as SETINV and UNSETINV																											
4	08H	1 = enable foreground flash; 0 = disable. Same function SETFLASH and UNSETFLH																											
5	04H	Same as SETSCRL																											
6	02H	Same as SETLF																											
7	01H	Same as SETCOL																											
1	01	SETLF	1	1	1 = Enable automatic linefeed on CR; 0 = disable.																								
2	02	SETSCRL	1	1	1 = Enable automatic scrolling; 0 = when @ tries to move down from (X, 15), it is moved to (0,0) and display is filled with default character.																								
3	03	SETBACK	1	0	1 = write backward (cursor moves left after a character is displayed, and wraps up); 0 = write forward.																								
4	04	SETCLEOP	1	1	1 = enable clear display to end of page on LF; 0 = disable.																								
5	05	SETPAGE	1	0	0 = select page 1; 1 = select page 2.																								
6	06	SETMODE	1	0	Select display mode: 0 (00H) = Alphanumeric 4 (04H) = Semigraphic 4 6 (06H) = Semigraphic 6 64 (40H) = Graphic 64 128 (80H) = Graphic 128																								
7	07	SETPRMS	1	0	Sets background display parameters. Each bit in the $p_0$ sets one parameter. Note that many of these parameters can be set separately with other escape sequences.																								

Table E-7. Escape Sequences in Alphanumeric Mode (Continued)

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning																											
<table><tr><th>Bit</th><th>Value</th><th>Parameter</th></tr><tr><td>0</td><td>80H</td><td>Bits 80, 40, 04, and 02 determine the current display mode. Their functions and values are the same as SETMODE. "OR" other required bits into these values.</td></tr><tr><td>1</td><td>40H</td><td>See bit 80H</td></tr><tr><td>2</td><td>20H</td><td>Not used; should be 0</td></tr><tr><td>3</td><td>10H</td><td>Same as SETPRINV</td></tr><tr><td>4</td><td>08H</td><td>Same as SETPRFLH</td></tr><tr><td>5</td><td>04H</td><td>See bit 80H</td></tr><tr><td>6</td><td>02H</td><td>See bit 80H</td></tr><tr><td>7</td><td>01H</td><td>Same as SETPRCOL</td></tr></table>						Bit	Value	Parameter	0	80H	Bits 80, 40, 04, and 02 determine the current display mode. Their functions and values are the same as SETMODE. "OR" other required bits into these values.	1	40H	See bit 80H	2	20H	Not used; should be 0	3	10H	Same as SETPRINV	4	08H	Same as SETPRFLH	5	04H	See bit 80H	6	02H	See bit 80H	7	01H	Same as SETPRCOL
Bit	Value	Parameter																														
0	80H	Bits 80, 40, 04, and 02 determine the current display mode. Their functions and values are the same as SETMODE. "OR" other required bits into these values.																														
1	40H	See bit 80H																														
2	20H	Not used; should be 0																														
3	10H	Same as SETPRINV																														
4	08H	Same as SETPRFLH																														
5	04H	See bit 80H																														
6	02H	See bit 80H																														
7	01H	Same as SETPRCOL																														
8	08	SETPRCOL	1	0	Background color. If SETPRINV = 0, then 1 = orange, 0 = green. If SETPRINV = 1, then 1 = yellow, 0 = black.																											
9	09	SETPRINV	1	0	Invert background color. See SETPRCOL.																											
10	0A	SETPRFLH	1	0	1 = enable background flash; 0 = disable. Background alternates between its regular and inverted color.																											
11	0B	SCRCOL	1	—	Sets the entire display to a specified color. Both foreground and background are affected. 1 = orange; 0 = green.																											
12	0C	SCRINV	1	—	Sets the entire display to a specified invert mode. Both foreground and background are affected. 1 = invert enabled; 0 = disabled.																											
13	0D	SCRFLH	1	—	Sets the entire display to a specified flash mode. Both foreground and background are affected. 1 = flash enabled; 0 = disabled.																											
14	0E	SCRLUP	1	—	Scrolls entire display up $p_0$ lines. The vacated lines are filled with the default character.																											



Table E-7. Escape Sequences in Alphanumeric Mode (Continued)

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning
15	0F	SCRLDN	1	—	Scrolls entire display down $p_0$ lines. The vacated lines are filled with the default character.
16	10	SCURX	1	—	Moves @ horizontally to column $p_0$ .
17	11	SCURY	1	—	Moves @ vertically to line $p_0$ .
18	12	CURRT	x	—	Moves @ horizontally to (31,Y).
19	13	SETTAB	1	—	Sets a tab stop at column $p_0$ .
20	14	CLRTSTOP	1	—	Clears tab stop at column $p_0$ .
21	15	CLRTABS	x	—	Clears all tab stops.
22	16	SETBYTE	1	20H	Sets ASCII default character to $p_0$ .
23	17	FILLCHRS	1	—	Sets part of current display to the current default character (see SETBYTE). $p_0$ characters are affected, beginning at (X,Y) and extending toward the end of the display.
24	18	FILLINES	1	—	Sets part of current display to the current default character (see SETBYTE). $p_0$ lines are affected, beginning at @, extending toward the bottom of the display, and ending at the end of the line $p_0-1$ lines below @.
25	19	FILLSCR	1	—	Sets entire display to the ASCII character $p_0$ .
26	1A	CHRFLH	1	—	Sets part of current display to the current foreground flash mode (see SETFLASH / UNSETFLH). $p_0$ characters are affected, beginning at (X,Y) and extending toward the end of the display.
27	1B	CHRCOL	1	—	Sets part of current display to the current foreground color (see SETCOL). $p_0$ characters are affected, beginning at (X,Y) and extending toward the end of the display.

Table E-7. Escape Sequences in Alphanumeric Mode (Continued)

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning
28	1C	CHRRINV	1	—	Sets part of current display to the current foreground invert mode (see SETINV). $p_0$ characters are affected, beginning at (X,Y) and extending toward the end of the display.
30	1E	FLIPINV	1	—	Reverses invert mode of part of current display (see SETINV). $p_0$ characters are affected, beginning at (X,Y) and extending toward the end of the display.
31	1F	FLIPCOL	1	—	Reverses color of current display (see SETCOL). $p_0$ characters are affected, beginning at (X,Y) and extending toward the end of the display.
64	40	DUNESC	1	—	No-operation. All HHC devices other than the TV Adaptor display or print $p_0$ as though it had been written as an ordinary data byte; thus, it can be used to insert "except on TV" data into a common data stream being written to several devices.
65	41	EOSINSERT	1	—	Inserts ASCII character $p_0$ at (X,Y). The rest of the display is displaced to the right. The last character of each line is displaced to the next line. The character at (31,15) is lost.
66	42	EOSDELETE	x	—	Deletes character at (X,Y). The rest of the display is displaced to the left. The first character of each line is displaced to the preceding line. The position at (31,15) is filled with the default character.
67	43	SETINV	x	—	Inverts foreground color. See SETCOL.

Table E-7. Escape Sequences in Alphanumeric Mode (Continued)

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning
68	44	UNSETINV	x	—	Uninverts foreground color.
69	45	SETFLASH	x	—	Enables foreground flash. Foreground flips between its regular and inverted color.
70	46	UNSETFLH	x	—	Disables foreground flash.
73	49	SETTRANS	1	0	1 = display control commands as data; 0 = execute control commands as usual.
74	4A	CHOME	x	—	Moves @ to home position, (0,0).
75	4B	WBREAK	1	20H	Sets ASCII delimiter character for word wrap feature to $p_0$ . $p_0 = 0FFH$ disables the feature.
76	4C	SETCOL	1	0	Sets foreground color. If foreground color is inverted (see SETINV and UNSETINV), then 1 = orange, 0 = green. If not, then 1 = yellow, 0 = black.
77	4D	EOLINSERT	1	—	Inserts ASCII character $p_0$ at (X,Y). The rest of the line is displaced to the right. The character at (31,Y) is lost.
78	4E	EOLDELETE	x	—	Deletes character at (X,Y). The rest of the line is displaced to the left. The position at (31,Y) is filled with the default character.
79	4F	INLINE	1	$\pm 1$	An empty line (filled with the default character) is inserted in the display at line Y. If $p_0 = +1$ , lines 1 through Y are moved up one line; line 0 is lost. If $p_0 = -1$ , lines Y through 14 are moved down one line; line 15 (the bottom line) is lost.
80	50	DELLINE	1	$\pm 1$	Line Y is deleted from the display. If $p_0$ equals +1, lines 0

Table E-7. Escape Sequences in Alphanumeric Mode (Continued)

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning
82	52	MULDATA	Varies	—	through Y-1 are moved down one line, and line 0 is filled with the default character. If the operand $p_0$ equals -1, lines Y+1 through 15 are moved up one line, and line 15 is filled with the default character. Displays one or more raw data bytes. $p_0$ indicates the number of raw data bytes given. Each following byte contains one raw data byte. The display begins at (X,Y) and proceeds in the current direction of cursor motion, wrapping at line boundaries in the usual way.
83	53	SVCTL	1	—	Stores $p_0$ into the control byte representing position (X,Y) and updates the display. The data byte at (X,Y) is not changed. The cursor moves in the usual direction.
84	54	SVDATA	1	—	Stores $p_0$ into the data byte at (X,Y) (i.e., displays it as raw data). The control byte at (X,Y) is not changed. The cursor moves in the usual direction.

dimensions of a pixel matrix, measured in pixels, are fixed. The dimensions of the display, measured in pixels or pixel matrixes, are also fixed.

Every pixel on the display has a Cartesian (X,Y) coordinate. Pixel rows are numbered from the top of the display to the bottom, the top row being number 0. Pixel columns are numbered from the left of the display to the right, the leftmost row being number 0. You control the cursor by moving it to a pixel with a specified coordinate.

There are two ways to display graphic information: in pixel mode and in matrix mode.

When you write data bytes to the TV Adaptor, you are in *pixel mode*. In this mode, each data byte turns an individual pixel on or off and determines



**Table E-8.** Summary of the Characteristics of Semigraphic and Graphic Modes

Mode	Screen Size (Matrices)	Matrix Size	Screen Size (Pixels)	Display Pages	Notes
Semigraphic 4	16 × 32	2 × 2	32 × 64	2	Any pixel on or off; one color/matrix.
Semigraphic 6	16 × 32	3 × 2	48 × 64	2	Any pixel on or off; one color/matrix.
Graphic 64	64 × 16	1 × 4	64 × 64	1	All pixels on; any pixel any color.
Graphic 128	64 × 16	1 × 8	64 × 128	1	Any pixel on or off; one color/matrix.

its color. When you write data with the control command `IDATA` or with the escape sequence `MULDATA`, you are in *matrix mode*. In this mode each byte affects all the pixels in a matrix.

Pixel mode is easier to program and is considered the normal mode of displaying graphic information. Matrix mode is more efficient and can display data more rapidly.

### Writing Data in Pixel Mode

The TV Adaptor recognizes the same three types of output in semigraphic modes as in alphanumeric mode, but it defines the three types somewhat differently.

Binary values 00H through 08H are considered pixel mode data. Each data byte sets the pixel at the cursor to a specified color or turns it off. Writing a pixel affects the other pixels in the same pixel matrix in a manner dependent on the current mode.

After a pixel mode data byte has been written, the TV Adaptor moves the cursor one pixel in its current direction of motion. The cursor does not wrap when it has reached its limit of motion in any direction, nor does the display scroll.

Binary values greater than 08H are either control commands, as listed in Table E-9, or are ignored.

Table E-9 lists control commands for the TV Adaptor in semigraphic and graphic modes. The symbol "@" represents the cursor, and (X,Y) represents the cursor's position. All addressing is done by pixel matrix.

**Table E-9.** Control Commands in Semigraphic and Graphic Modes

Decimal Value	Hex Value	Symbol	Control Sequence Length	Meaning
15	0F	MOVECUR	1	Moves cursor one pixel in its current direction of motion. The display is not changed.
16	10	GO.UP	1	Set cursor direction to "up."
17	11	GO.BACK	1	Set cursor direction to "left."
18	12	GO.FORW	1	Set cursor direction to "right."
19	13	GO.DOWN	1	Set cursor direction to "down."
20	14	GO.UP,BACK	1	Set cursor direction to "up and left."
21	15	GO.UP,FORW	1	Set cursor direction to "up and right."
22	16	GO.DOWN,BACK	1	Set cursor direction to "down and left."
23	17	GO.DOWN,FORW	1	Set cursor direction to "down and right."
24	18	GO.NOWHERE	1	Set cursor direction to "stay in place."
25	19	DRAWLINE	4	Draws a line from @ to coordinate( $c_1$ , $c_2$ ). The line is one pixel thick. Its color is determined by $c_3$ (see Table E-11 for the color each value of $c_3$ produces). The cursor is left at ( $c_1$ , $c_2$ ).
27	1B	ESC	Varies	Begins an escape sequence. Followed by an opcode and one or more operand bytes. (See Table E-10 for information about escape sequence opcodes and operands.)
29	1D	POSCUR	3	Moves @ to coordinate ( $c_1$ , $c_2$ ). Does not affect the contents of the display.
30	1E	IDATA	2	Writes $c_1$ to the data byte representing the pixel matrix at @. The corresponding control byte is set to the current default, as set by SETPRCOL or SETPRMS. (See text for more information.)
31	1F	XYD	4	Moves @ to coordinate ( $c_1$ , $c_2$ ); writes $c_3$ to the data byte representing the character at @. The corresponding control byte is set to the current default, as in IDATA. The cursor moves one pixel in the current direction.

Table E-10 lists escape sequences for the TV Adaptor in semigraphic and graphic modes. The "symbol" column has the same significance as in Table E-11, but refers to the second character, the opcode, rather than to the first character, which is always ESC. The "operand bytes" column gives the number of operand bytes in the sequence. An "x" means that the sequence has no operands; the opcode must be followed by one operand byte, whose value is ignored. In the "meaning" column, the first operand is denoted by " $p_0$ ", the second by " $p_1$ ," and so forth.

Table E-10. Escape Sequences in Graphic and Semigraphic Modes

Decimal Value	Hex Value	Symbol	Operand Bytes	Default Value	Meaning
5	05	SETPAGE	1	0	0 = select page 1; 1 = select page 2. (Meaningful only in semigraphic 4 and semigraphic 6 modes.)
6	06	SETMODE	1	0	Select display mode: 0 (00H) = alphanumeric 4 (04H) = semigraphic 4 6 (06H) = semigraphic 6 64 (40H) = graphic 64 128 (80H) = graphic 128
7	07	SETPRMS	1	Varies with Mode	See description of SETPRMS for alphanumeric mode (Table E-7).
8	08	SETPRCOL	1	0	Sets the 1's bit in the default control byte value. The significance of this bit varies with mode. (See Figures E-1 through E-4.)
64	40	DUNESC	1	—	No-operation. All HHC devices other than the TV Adaptor display or print $p_0$ as though it had been written as an ordinary data byte; thus, it can be used to insert "except on TV" data into a common data stream being written to several devices.
74	4A	CHOME	x	—	Moves @ to home position, (0,0).
81	51	FILLGRAF	1	—	The entire display is filled with $p_0$ , interpreted as a data byte (see Figures E-1 through E-4). Every control byte is set to the default control byte.
82	52	MULDATA	Varies	—	Outputs one or more data bytes. $p_0$ indicates the number of data bytes given. The equivalent of a 1DATA is performed for each data byte in turn.
83	53	SVCTL	1	—	Writes $p_0$ to the control byte for the pixel matrix at @. The cursor moves in its current direction.
84	54	SVDATA	1	—	Writes $p_0$ to the data byte for the pixel matrix at @. $p_0$ is the data byte. The cursor moves in its current direction.

Data byte functions in semigraphic 4 mode are listed in Table E-11. Whenever a pixel is turned on, the other "on" pixels in the same pixel matrix are set to the same color.

Data byte functions in semigraphic 6 mode are shown in Table E-12. Whenever a pixel is turned on, the other "on" pixels in the same pixel matrix are set to the same color.

As of this writing, a "bug" in the TV Adaptor ROM causes color changes. If a pixel is turned off, the colors of the other "on" bits change as follows: green becomes buff, yellow becomes cyan, blue becomes magenta, and red becomes orange. You can avoid this problem if you write all of the "off" pixels in a matrix before you write the "on" pixels.

Table E-11. Data Byte Values in Semigraphic 4 Mode (Pixel Mode)

Data Byte Value	Turns Pixel	And Sets It to Color
0	On	Green
1	On	Yellow
2	On	Blue
3	On	Red
4	On	Buff
5	On	Cyan
6	On	Magenta
7	On	Orange
8	Off	(None)

Table E-12. Data Byte Values in Semigraphic 6 Mode (Pixel Mode)

If SETPROCOL=1, Data Byte Value	If SETPRCOL=0, Data Byte Value	Turns Pixel	And Sets It to Color
	0	On	Green
4	1	On	Yellow
5	2	On	Blue
6	3	On	Red
7	4	On	Buff
0	5	On	Cyan
1	6	On	Magenta
2	7	On	Orange
3	8	Off	(None)
8			



Data byte functions in graphic 64 mode are shown in Table E-13.

The eight possible colors are divided into two ranges: 0-3 (green, yellow, blue, and red) and 4-7 (buff, cyan, magenta, and orange). Whenever a pixel is turned on, the other "on" pixels in the same pixel matrix are shifted into the same range. For example, if a matrix contains buff pixels and a yellow pixel is written, the buff pixels (color 4) are changed to green (color 0). But if a cyan pixel is written, the buff pixels are not affected, because cyan (color 5) is in the same range as buff (color 4).

Data byte functions in graphic 128 mode are shown in Table E-14.

**Table E-13.** Data Byte Values in Graphic 64 Mode  
(Pixel Mode)

If SETPRCOL = 1, Data Byte Value	If SETPRCOL = 0 Data Byte Value	Sets Pixel to Color
4	0	Green
5	1	Yellow
6	2	Blue
7	3	Red
0	4	Buff
1	5	Cyan
2	5	Magenta
3	7	Orange
8	8	Equivalent to 4

**Table E-14.** Data Byte Values in Graphic 128 Mode  
(Pixel Mode)

If SETPRCOL = 1, Data Byte Value	If SETPRCOL = 0, Data Byte Value	Sets Pixel to Color	And Sets Rest of Matrix to Color
2	0	Black	Green
3	1	Green	Black
0	2	Black	Buff
1	3	Buff	Black
4-7	4-7	Equivalent to 0-3	
8	8	Equivalent to 0	

## Writing Data in Matrix Mode

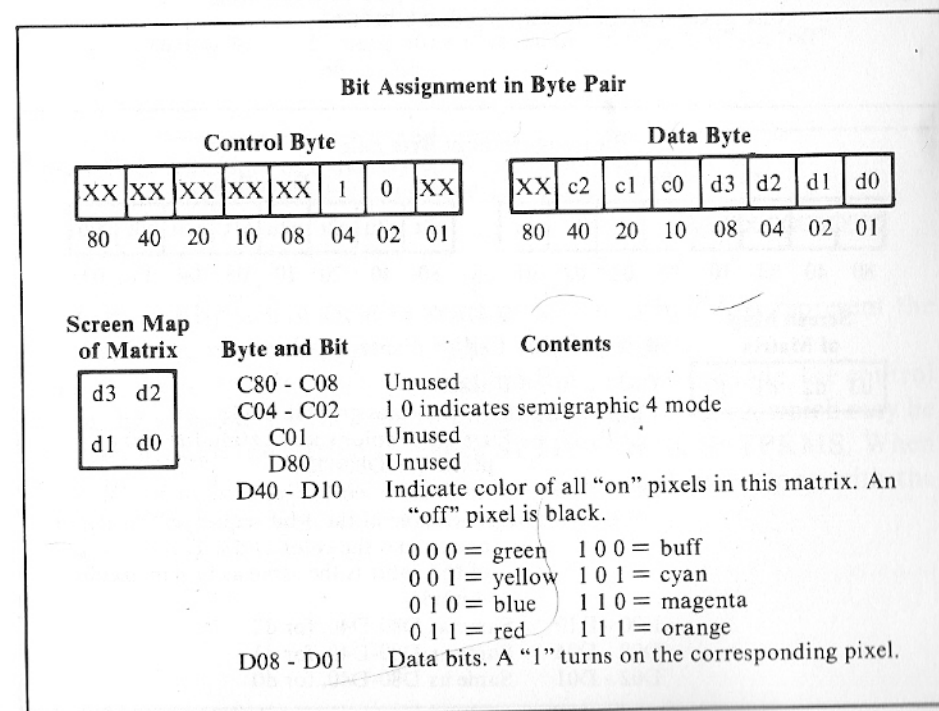
You write data in matrix mode with the control command 1DATA, which writes a single data byte, or with the escape sequence MULDATA, which writes multiple bytes.

When you write data in matrix mode, each data byte changes the contents of the entire pixel matrix containing the pixel at the cursor. Then the cursor moves one matrix in its current direction of motion.

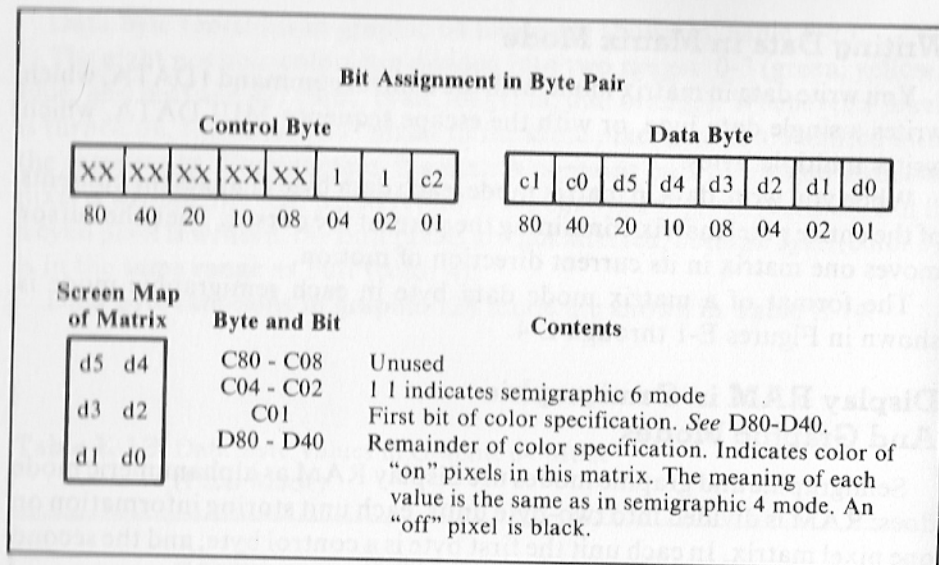
The format of a matrix mode data byte in each semigraphic mode is shown in Figures E-1 through E-4.

## Display RAM in Semigraphic And Graphic Modes

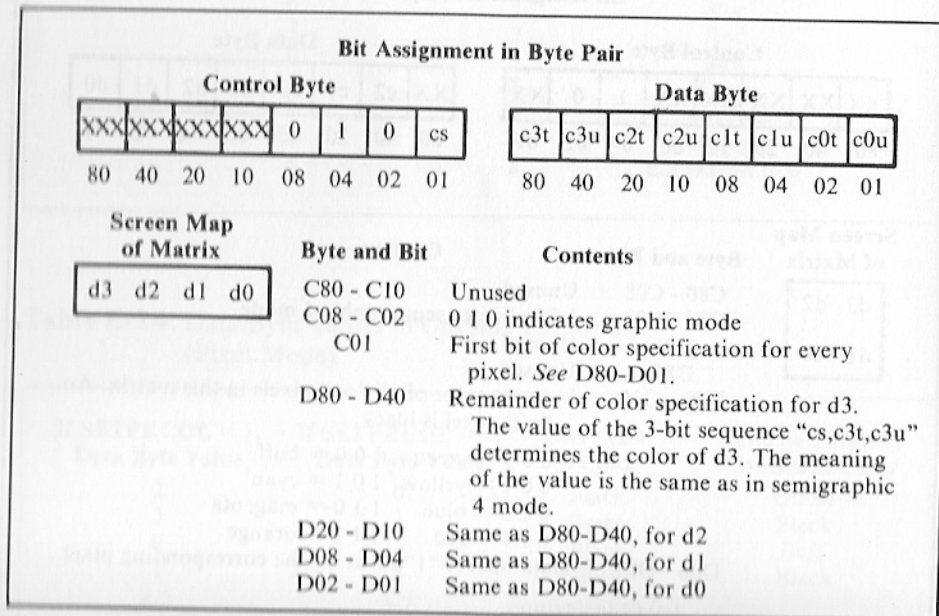
Semigraphic and graphic modes use display RAM as alphanumeric mode does: RAM is divided into two-byte units, each unit storing information on one pixel matrix. In each unit the first byte is a control byte, and the second



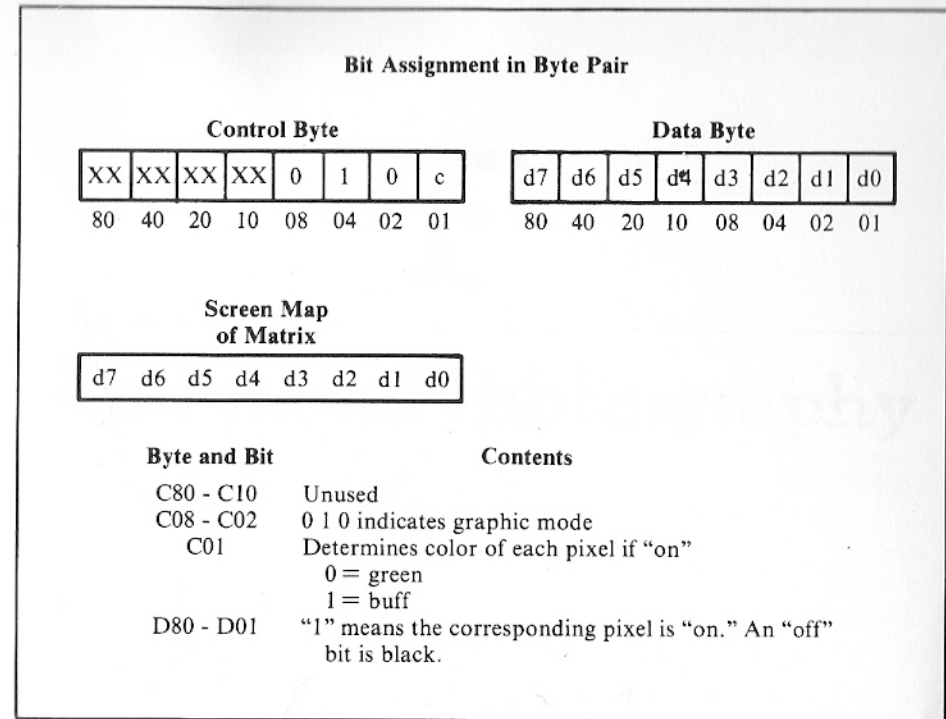
**Figure E-1.** Structure of a display RAM byte pair representing one pixel matrix in semigraphic 4 mode



**Figure E-2.** Structure of a display RAM byte pair representing one pixel matrix in semigraphic 6 mode



**Figure E-3.** Structure of a display RAM byte pair representing one pixel matrix in graphic 64 mode



**Figure E-4.** Structure of a display RAM byte pair representing one pixel matrix in graphic 128 mode

byte is a data byte. Consecutive pairs of locations in RAM represent the pixel matrixes on the display in English reading order.

When you write a data byte in pixel mode or in matrix mode, the control byte of the affected matrix is set to the current default value, which may be changed through the escape sequence SETPRCOL or SETPRMS. When you write raw data, of course, you set each control byte along with the corresponding data byte, as in alphanumeric mode.



# F

## Annotated Bibliography

### BASIC PROGRAMS AND PROGRAMMING

Osborne/McGraw-Hill is publishing two BASIC tutorials. *Armchair BASIC* by David and Annie Fox is designed for the nonprogrammer. *Microsoft BASIC Tutorial* by Walter Ettlin and Greg Solberg, to be published in 1983, is an introduction to the popular Microsoft BASIC language.

Two books on programming style should be read in tandem with any language tutorial. *The Elements of Programming Style* by Brian Kernighan and P.J. Plauger (McGraw-Hill Book Co.) and *BASIC with Style* by Paul Nagin and Henry Ledgard (Hayden Book Co.) will teach you how to write well-constructed, correct programs and introduce you to the elements of programming style. They are "must" reading for any BASIC programmer, from beginner to expert.

Osborne/McGraw-Hill publishes three collections of BASIC programs: *Some Common BASIC Programs*, *Practical BASIC Programs*, and *Science and Engineering Programs*.

Many computing magazines publish BASIC programs in each issue. Check current and back issues of *Compute!*, *Popular Computing*, and

*BYTE* for a start. Review these magazines and others to see which one is best suited to your background and interests.

The user's manuals for Microsoft BASIC and SnapBASIC contain a number of examples of complete BASIC programs, as well as excellent tutorials in BASIC programming.

## INFORMATION ABOUT SNAP AND FORTH

The SnapFORTH tutorial guide, distributed with the SnapFORTH capsule, provides an introduction to programming in SNAP. The SnapFORTH reference manual, also distributed with the capsule, provides complete information on the SNAP language and the architecture of the HHC from a programmer's point of view.

SNAP is based on the widely available language FORTH. If you would like to learn FORTH in an easy, tutorial fashion, read *Discover FORTH* by Thom Hogan (Osborne/McGraw-Hill).

## HHC HARDWARE

Service manuals and design specifications for the HHC are available to qualified parties interested in developing new HHC hardware products. Contact Panasonic, Olympia, or Quasar for information.

# G

## Glossary

This appendix defines the words and phrases that are printed in italics in the text.

Arranged for easy reference, each entry lists the chapter or appendix in which the term is introduced in italics.

*AC Adaptor*: a device that powers the HHC from AC power and recharges the HHC's batteries (Chapter 2).

*Acoustic coupler*: a type of modem that enables your HHC to communicate with another modem-equipped computer through a telephone handset. See "modem" (Chapter 9).

*Alarm*: an entry in the HHC's Clock/Controller. At a time that you specify, the alarm will "ring." It will continue to ring at approximate ten-minute intervals—even if your HHC is turned off—until you enter the Clock/Controller and acknowledge the alarm (Chapter 5).

*ALL OFF switch*: a slide switch in the back of the HHC, so named because it controls the HHC's power supply and can turn off all parts of the HHC (Chapter 2).

*Assembler*: a computer program that translates assembly language programs into machine language code (Chapter 8).



**Assembly language:** a convenient, human-readable notation used to write programs in machine language (Chapter 8).

**Attache Carrying Case:** a specially fitted case that can hold an HHC and peripherals (Chapter 2).

**Auto-answer modem:** a modem capable of responding, without human assistance, to a telephone call from another modem. Auto-answer modems are customarily used on time-sharing computer systems (Chapter 9).

**Auto-dial modem:** a modem capable of "dialing" a telephone call under software control (Chapter 9).

**Auto-off feature:** a feature of the HHC operating system that protects you from draining the HHC's batteries should you forget to turn off the HHC. If you leave the HHC on for about ten minutes without pressing any key, the HHC will turn itself off (Chapter 2).

**Auto-repeat:** a keyboard feature that lets you enter a character many times while holding down its key. Most of the HHC's keyboard keys auto-repeat, but the auto-repeat speed depends on the setting of the STP/SPD key (Chapter 3).

**Back-up:** a file copy kept in a safe place in case the working copy is lost or damaged (Chapter 6).

**Baud rate:** the maximum rate at which data can be transmitted over a serial interface. Two serial devices must be set to the same baud rate to communicate with each other. If you use a rate of 300 baud—the most common setting for telecommunications and low-speed printers—the maximum rate at which data can be transmitted is 300 bits, or about 30 characters, per second. "Baud" is a contraction of "Baudot," the name of an early system for encoding character data in digital signals (Chapter 9).

**Binary file:** any file that is not a text file. SNAP files, Microsoft BASIC and SnapBASIC, and Portaflex program files are all binary files. Programs generally use binary files to store "private" information. For example, the Telecomputing series of programs, which allow you to connect your HHC to other computers through the Modem, use binary files to store information that is used to configure the Modem so that it can communicate with your type of computer connection (Chapter 3).

**Bit:** the most elementary unit of digital information. A bit can have only two values—zero or one. Digital computers represent all their data and

programs as strings of bits or strings of larger units of data (either bytes, characters, or words) composed of bits (Chapter 9).

**Blip:** an indicator that gives you information about the HHC's status. Along the bottom of the LCD, the HHC can display the eight blips "SHIFT," "LOCK," "2nd SFT," "DELETE," "INSERT," "ALARM," "ON LINE," and "▢." The rectangular blip has no fixed definition (Chapter 2).

**Buffer:** (1) To save data temporarily until they can be accepted by their destination. Input is buffered between the time it comes in from a peripheral and the time a program reads it; output is buffered between the time it is written by a program and the time it goes out to a peripheral. (2) When used as a noun, the word "buffer" refers to an area in RAM where data are temporarily stored (Chapter 9).

**Byte:** a unit of computer memory that holds one character of information. RAM, ROM, and file sizes are customarily measured in bytes. On the HHC and most other computers, a byte consists of eight bits (Chapter 2).

**Carrier signal:** a continuous signal that can be modulated to make it carry information. Modulation is a change in frequency, amplitude, or phase. The radio frequency signal that the HHC's TV Adaptor outputs through its "RF out" jack is a carrier signal modulated to carry a video image. The high-pitched tone that you hear when you turn on the HHC's Modem or dial up a modem-equipped computer on your telephone is a carrier signal modulated to carry a stream of digital information (Chapter 9).

**CLEAR key:** a key on the HHC's keyboard that you use to reset the HHC when you have finished running most programs. See Appendix B for cautions on use of the CLEAR key (Chapter 2).

**Compiler:** a computer program that translates from a high-level language into machine language or into a pseudo-machine language that may then be interpreted (Chapter 8).

**Computer program:** a set of instructions stored in a computer and used to control the computer's operation (Chapter 8).

**Control character:** a "character" with a control function, such as ringing a bell or beginning a new line on an output device. This differs from the "printable character" that causes the device to print or display symbols like "a," "3," or "!" (Appendix D).

**Crossed cable:** an RS-232C cable in which certain pairs of pins are cross-connected from one end to the other. This cable is used to connect the RS-232C Interface to a printer, display terminal, or other serial-interface device (Chapter 9).

**Current file:** the file that the File System editor is creating or modifying at any given time (Chapter 3).

**Current file space:** the file space currently "visible" to the HHC's File System and to other programs that use files. The current file space is in either intrinsic (HHC) or extrinsic (PMP) RAM. *See also* "intrinsic," "extrinsic," and "PMP" (Chapter 6).

**Current line:** the line in the current file that the File System editor is creating or modifying at any given time (Chapter 3).

**Cursor:** a symbol that marks the place where the next typed character will go on a display device such as the LCD. The HHC's "normal" cursor is a flashing solid rectangle. The "insert" cursor is a rectangle filled with a checkerboard pattern. The "delete" cursor is an empty, outlined rectangle (Chapter 3).

**Debugging:** the process of diagnosing and correcting errors in the logic of a computer program. For all but the most trivial programs, debugging is an unavoidable part of programming (Chapter 8).

**Device:** an abbreviation of "peripheral device." *See* "peripheral" (Chapter 9).

**Device capsule:** ROM mounted on a special plastic carrier. A device capsule contains control software for an HHC peripheral such as the Modem; it may also contain an application program. Plug the device capsule into a socket on the peripheral device it is designed to control (Chapter 7).

**Direct-connect modem:** a modem that interfaces directly with the telephone network. An acoustic coupler, in contrast, operates through the handset of a telephone (Chapter 9).

**Downloading:** the process of transferring information (most often a source program or object program) from a host computer to a target computer. The opposite of downloading is "uploading," that is, transferring information from the target computer to the host (Chapters 8 and 9).

**ECB:** *see* "event control block" (Appendix C).

**Editing keys:** a group of keys on the right side of the HHC's keyboard. Used primarily to edit data, the editing keys are ←, →, ↑, ↓, INSERT, DELETE, ROTATE, and SEARCH (Chapter 2).

**Editor:** a computer program that can be used to create and modify text files. *See* "File System" (Chapter 3).

**ENTER key:** a key located near the lower right corner of the HHC's keyboard. Its action is similar to a typewriter's RETURN key (Chapter 2).

**EPROM:** *see* "erasable programmable read-only memory" (Chapter 8).

**EPROM burner:** a peripheral device that records information, usually computer programs, on EPROMs (Chapter 9).

**Erasable programmable read-only memory (EPROM):** a type of ROM that can be "burned" with a program, erased, and reburned. EPROMs are used to produce experimental, small-volume, custom programs in HHC capsules (Chapter 8).

**Event control block (ECB):** a software control block that the HHC's operating system uses, along with internal processing, to synchronize events such as the startup and completion of I/O operations (Appendix C).

**Extrinsic:** added to the HHC as an option or accessory, or attached to the HHC temporarily. The RAM in a Programmable Memory Peripheral is called extrinsic RAM (Chapter 6).

**Extrinsic RAM:** RAM contained in a Programmable Memory Peripheral (Chapter 6).

**File:** a collection of information grouped together under one name. A computer program processes stored data by reading and writing the file in which the data are stored. You would work in much the same way with a conventional file—reading and writing your documents after retrieving a file folder from the filing cabinet (Chapter 3).

**File space:** an area in RAM that the HHC uses to store files. This area may be the intrinsic file space in intrinsic RAM or an extrinsic file space in any Programmable Memory Peripheral (PMP) attached to the HHC (Chapter 6).

**File System:** (1) One of the HHC's three intrinsic applications, which can be used to locate, rename, and delete files. The File System contains an editor that may be used to create and modify text files. (2) The part of the



**HHC's operating system** that maintains HHC files in RAM. It is used by the File System application to manipulate files (Chapter 3).

**File type:** the characteristics of an HHC file. They are stored with the file but are not visible when you look at the file with the File System. *See* "text file," "invisible file," "SNAP file," and "binary file" (Chapter 3).

**Floating point:** a way of storing numbers in a computer that can represent real numbers like 0.01 and 10.99 as well as integers like 0, 1, and 10. The HHC's operating system contains software that operates on floating point numbers with 13 decimal digits of precision. Microsoft BASIC has its own set of floating point software that operates on floating point numbers with nine decimal digits of precision (Chapter 8).

**Function keys:** three keys labeled f1 through f3. They let you enter frequently used sequences of keystrokes by pressing a single key. You set the function keys' definitions with the HELP key (Chapter 2).

**Graphics:** computer output that takes the form of a picture rather than text. The HHC can display limited graphics on its LCD, but its primary graphics devices are the Plotter and TV Adaptor (Chapter 9).

**HHC capsule:** a ROM chip mounted on a special plastic carrier. An HHC capsule containing an HHC application program is plugged into one of the three sockets on the back of the HHC (Chapter 7).

**HELP key:** a keyboard key that displays a brief explanation of another key's function. You also use the HELP key to define the function keys (Chapter 2).

**High-level language:** a programming language that is easy to use because it expresses programs in terms of the problem to be solved rather than the operations the computer hardware must perform to solve them (Chapter 8).

**Host computer:** (1) A computer that is used to develop programs that will ultimately be run on another computer. The host computer is selected because of its speed or advanced capabilities. A computer with a full-size keyboard and fast disk-based file storage is often used as a host computer for developing HHC programs in SnapFORTH or SnapBASIC. (2) A larger computer that you communicate with through a modem. With a Modem peripheral, you can transfer the data that your HHC collects in the field to a larger database. *See also* "modem" (Chapter 8).

**Input:** (1) The process of reading information into a computer. It is done through a peripheral or through an intrinsic device such as a keyboard. (2) The information that is read into a computer (Chapter 9).

**Interpreter:** a computer program that executes programs expressed in pseudo-machine language. An interpreter examines the pseudo-machine instructions one by one and performs the operations those instructions would perform directly if the pseudo-machine were real. *See also* "compiler" (Chapter 8).

**Interpretive execution:** the process of executing a pseudo-machine language program with an interpreter (Chapter 8).

**Intrinsic:** built into the HHC. The term is used to describe components and programs that are part of the HHC's main unit. Its opposite is "extrinsic" (Chapter 3).

**Intrinsic RAM:** RAM that is built into the HHC. Its opposite is "extrinsic RAM" (Chapter 6).

**Invisible file:** a type of file that does not appear in the file menu presented by the File System editor. Any file may have the "invisible" file type in addition to its other types. Files that programs use to store "private" information are often invisible. The files that the Telecomputing programs use to store their configuration parameters, for example, are invisible (Chapter 3).

**I/O:** means "input and/or output." I/O is any process that transfers information into a computer (input), out of a computer (output), or both into and out of a computer (Chapter 9).

**I/O Adaptor:** a plastic tray that connects the HHC to as many as six peripherals at once. The HHC slides into the lower right channel of the I/O Adaptor, and its peripheral socket connects to a plug on the Adaptor (Chapter 2).

**I/O key:** a key on the HHC keyboard that displays the status of the HHC's peripherals and RAM and allows you to change the current file space and turn peripherals on and off (Chapters 2 and 3).

**Keyboard overlay:** a rectangular sheet of plastic that you place over the HHC's keyboard when you are using a particular application program. The keyboard overlay relabels the keyboard keys to reflect the functions they have in a particular program (Chapter 2).

**LCD:** see "liquid crystal display" (Chapter 1).

**Library capsule:** an HHC capsule that may be used as a "library" of subroutines to be called by programs in other capsules or SNAP files. The Scientific Calculator capsule is a library capsule (Chapter 7).

**Liquid crystal display (LCD):** the HHC's primary output medium. Located above the keyboard on the main unit, the LCD is an electronic device consisting of specially treated material placed between two glass plates. An electric field applied to the plates can make the material between them change from transparent to opaque, creating a visible image. The HHC's LCD displays a matrix of square dots, 159 dots long and 8 dots high. The LCD can display as many as 26.5 characters (Chapter 1).

**LOCK key:** somewhat like the SHIFT LOCK key on a typewriter. The LOCK key lets you "lock" the effect of another key (Chapter 2).

**Logical unit number (LUN):** a number used internally by HHC programs to address a specific device, that is, to tell the operating system which device an I/O operation is meant for. Logical unit numbers are associated with devices when the program is executed, making it easy for a program to operate different devices at different times (Appendix C).

**LUN:** see "logical unit number" (Appendix C).

**Machine language:** the numerical "language" that a computer uses for its internal operations. All programs written in other computer languages are ultimately translated into machine language before the computer actually acts on the instructions. Machine language is expressed in binary notation—that is, in combination of 0's and 1's. Each type of computer has its own unique machine language (Chapter 8).

**Masked ROM:** a type of ROM produced in large quantities with a pre-recorded program. The one-time cost of preparing to produce a program in masked ROM is high (typically several thousand dollars), but the cost of producing each HHC capsule in masked ROM is low (Chapter 8).

**Memory:** (1) Any part of a computer that stores information, particularly RAM or ROM. (2) An area of RAM reserved for use by the HHC's intrinsic Calculator and used to store an intermediate result while the Calculator is calculating something else. Four of the HHC's keyboard keys (CM, RM, M+, and M-) manipulate the Calculator's memory (Chapter 4).

**Menu:** a list of options similar to the menu found in a restaurant. The HHC presents its menus on the LCD, one option at a time. From this menu you can choose one item. To select your option, press the key indicated before the equal sign (Chapter 2).

**Modem:** a peripheral device through which a computer communicates with another computer over a telephone line. A direct-connect modem connects directly to the telephone system through the modular jack in the wall, bypassing the usual telephone. A modem that operates through the telephone handset is called an acoustic coupler; it typically has a pair of cups that hold the two ends of a telephone's handset. An acoustic coupler is what most people mean when they say "modem." The HHC's Modem peripheral is an acoustic coupler (Chapters 1, 7, and 9).

**Object program:** an executable program produced from a source program by a compiler or assembler (Chapter 8).

**OFF key:** a key on the HHC's keyboard that turns off the HHC. See also "ON key" (Chapter 2).

**ON key:** an HHC keyboard key that turns on the HHC. See also the "ALL OFF switch," which controls power for the HHC's memory, processor, and LCD (Chapter 2).

**Operating system:** a program that manages the HHC's overall operation. It is permanently stored in the HHC's intrinsic ROM (Chapter 2).

**Output:** (1) The process of writing information out from a computer. The information is transmitted through a peripheral or through an intrinsic device such as an LCD. (2) Information that is written out from a computer (Chapter 9).

**Peripheral:** a device that is attached to a computer for reading information into the system (input), writing information out from the system (output), or both reading and writing (input/output or I/O) (Chapter 9).

**Persistent memory:** a feature that enables the HHC to preserve its File System files and "remember" its location in a program when turned off. With persistent memory, you can turn off the HHC while you are running a program. Later, when you turn it on, the program will run as though nothing had happened. Be careful, however. Some peripherals generate a spurious character when the HHC is turned off or on (Chapter 6).

**PMP:** see "Programmable Memory Peripheral" (Chapters 1 and 6).



**Portability:** the degree to which a computer program can be transported from one kind of computer or one environment to another and still work (Chapter 8).

**Posting:** the process of setting a bit in an ECB to indicate that the event represented by that ECB has occurred. ECBs are normally posted by the operating system (Appendix C).

**Primary menu:** the HHC's master menu. It tells you what programs are available in the HHC (Chapter 2).

**Printable character:** a character that represents a number, letter, or punctuation symbol. *See also* "unique character" and "control character" (Appendix D).

**Program function keys:** four keys in the lower left corner of the HHC keyboard. Labeled C1 through C4, they are used to control many of the programs that run on the HHC. They also sometimes function as typing keys and produce the special displayed characters "ä," "ö," "ü," and "ñ" (Chapters 2 and 6).

**Programmable Memory Peripheral (PMP):** a peripheral device that contains 4096, 8192, or 16,384 (4K, 8K, or 16K) characters of RAM to augment the HHC's intrinsic RAM (Chapters 1 and 6).

**Pseudo-machine language:** an imaginary computer's machine language; an easier target for translating a high-level programming language than the machine language of a real machine. Once a compiler has translated a source program into pseudo-machine language, the program may be executed by an interpreter (Chapter 8).

**RAM:** random-access memory. RAM is used to hold data for the HHC's operating system and programs and to store data files that programs operate on (Chapter 2).

**Random-access memory:** *see* "RAM" (Chapter 2).

**Read-only memory:** *see* "ROM" (Chapter 2).

**ROM:** read-only memory, a type of computer memory that is "burned" with a program or other data and then cannot be changed. ROM holds the HHC's operating system and intrinsic applications. In device capsules and HHC capsules, it is used to hold device control routines and other application programs (Chapter 2).

**RS-232C Interface:** a type of connection often used for communication between computers or between a computer and a peripheral. It is suitable for communicating data over distances as great as a few dozen feet and at speeds as high as a few thousand characters per second. It is also called a "serial interface" (Chapter 9).

**Scrolling:** the process of moving information into and out of a computer display so as to show more information than will fit on the display at one time. Think of the display as a small window. Think of scrolling as a process that moves a large page of paper around behind the window, revealing different parts of the page (Chapter 3).

**2nd SFT key (second shift):** a keyboard key that lets you type special characters. The HHC has a SHIFT key that lets you type capital letters and punctuation symbols; the 2nd SFT key gives you access to a third complete set of characters (Chapter 2).

**Serial interface:** a type of connection often used for communication between computers or between a computer and a peripheral. The serial interface transmits information one bit at a time over a single wire. The most common type is the RS-232C Interface. In fact, the term "serial interface" is often used loosely as a synonym for "RS-232C Interface" (Chapter 9).

**SHIFT key:** a keyboard key that is somewhat like the SHIFT key on a typewriter. The SHIFT key lets you type capital letters and punctuation symbols (Chapter 2).

**Slaving:** the process of writing output to another device as well as to the usual recipient. We say the second device is "slaved" to the first. You can slave a Printer, Plotter, or TV Adaptor to the LCD (Chapter 9).

**Sleeve:** a rigid plastic object that slides into the groove around the sides of the HHC. You can use the sleeve to hold together the HHC and a peripheral. You can also slide the sleeve over the face of the HHC to protect the keyboard from accidental pokes (Chapter 2).

**SNAP file:** a type of file that contains a runnable SNAP program. SNAP files appear in the secondary menu presented by the RUN SNAP PROGRAMS selection from the primary menu (Chapters 3 and 7).

**Source program:** a program that can be compiled or interpreted. The result is an object program (Chapter 8).

**STP/SPD key:** a keyboard key that lets you set the HHC's LCD display

speed and keyboard auto-repeat speed or "freeze" the HHC while it is running a program so that you can inspect the output (Chapter 2).

**Straight cable:** an RS-232C cable in which each wired pin of one plug is connected to the corresponding pin of the other plug. This cable is used to connect the RS-232C Interface to a serial-interface modem (Chapter 9).

**Text file:** a type of file that consists of lines of text, each of which ranges from zero to 255 characters in length. Text files with lines no more than 80 characters long may be created and modified with the File System editor (Chapter 3).

**Typing keys:** the keys on the left side of the HHC's keyboard, including the number and letter keys (Chapter 2).

**Unique character:** a character that is unique to the HHC's character set. Most of these characters are "printable" in the sense that they can be displayed on the LCD and some HHC peripheral devices such as the TV Adaptor (Appendix D).

**Uploading:** the process of transferring data from a target computer to a host computer. *See also* "downloading" (Chapter 9).

**Video adaptor:** a device for displaying digital information on a conventional television set. The HHC's TV Adaptor is a video adaptor (Chapter 9).

**Video monitor:** a device that is similar to a television set but is designed specifically to display computer images. A video monitor accepts an image-forming signal used directly to deflect the electron beam in the display tube and is not encoded on a carrier signal (Chapter 9).

**Word wrap:** a feature of many word processing programs, and one that also appears in several of the HHC's peripherals. When a line of input or output is too long to fit on a peripheral's output line or too long to fit between a word processing program's left and right margins, word wrap begins a new line at the beginning of the word that does not fit. When no word wrap feature is present, the new line begins with the first character that does not fit on the old line (Chapter 9).

## Index

### A

AC Adaptor, 10, 99, 126, 138, 140  
     Caution on use of nonstandard  
         adaptors, 135  
         Color Plotter, 111  
 Acoustic coupler, 115  
 Alarm in Clock/Controller, 55, 133  
 ALL-OFF switch, 9, 141  
 Alphanumeric mode, TV Adaptor, 151, 153,  
     159, 160  
 American Standard Code for Information  
     Interchange, *see* ASCII  
 Arrow key, 25, 31, 32, 43, 56, 58, 130, 132,  
     133, 146, 149  
 ASCII, 139, 145, 155, 156  
 Assembler, 82  
 Assembly language, 82, 84  
 Attache carrying case, 16  
 Auto file, 80  
 Auto-off feature, 19, 61  
 Auto-repeat feature, 26, 139  
 Automatic line feed, 154  
 Automatic scrolling, 155

### B

Back-up file storage, *see* File  
 Background, 155  
 Back-up, 121  
 BASIC, 152, 175  
     Caution on use of CLEAR key, 136

Battery, 138, 140  
     Care of, 20  
     Caution on changing PMP batteries, 136  
     Color Plotter, 111  
     HHC, 10, 15, 19  
     Peripherals, 99  
     PMP, 66  
 Baud, 112, 114, 117, 120  
 Binary file, 39, 79  
 Blip, 14  
     2nd SFT, 28  
     DELETE, 26  
     INSERT, 26  
     LOCK, 29  
     ON LINE, 117, 118  
     Software control of, 140  
 Buffer, 142  
 Burning, *see* EPROM  
 Bus, 138  
 Byte, 14

### C

Cable, RS-232C, 112, 114  
 Calculator, 41  
     Quick reference, 132  
 Capsule  
     Device, 16  
     HHC, 7  
     Library, 75  
 Carrier signal  
     Cassette interface, 118



## Carrier signal (continued)

- Modem, 117
- TV Adaptor, 124
- Carrying case, 15, 16, 20, *see also*
  - Attache carrying case
- Carrying strap, 15
- Cassette interface of Modem, 118
- Cassette recorder, 120
- Character generator in TV Adaptor, 157
- CLEAR key, 12, 19, 35, 64, 128, 132, 133, 136, 142
- Clock, Internal, *see* Time of century clock
- Clock/Controller, 53
  - Quick reference, 133
- Color Plotter, 107
  - Caution on handling mechanism, 135
  - Caution on use with AC Adaptor, 135
- Compiler, 82
- Computer program, 81
- Constant, in Calculator, 49
- Control byte, 155
- Control character, 145, 147
- Control command, 157, 159, 167
- CPU, 137
  - Power for, 141
- Current file, 22
- Current line, 22
- Cursor, 23, 139, 149
- Cursor control, *see* Arrow key

**D**

- Data entry, 76
- DB-25 connector, 112, 114
- Debugging, 85
- Default character, 154
- DELETE key, 26, 27, 29, 31, 32, 130, 133, 149
- Device capsule, 16, 72, 127
- Direct-connect modem, 115
- Display RAM, 155, 156, 171
- Downloading, 85, 91, 115

**E**

- ECB, 141, 142, 152, 153
- Editing key, 13
- Editor
  - File System, 22
  - Microsoft BASIC, 88
  - SnapBASIC, 91
- ENTER key, 12, 147
- EPROM, 93, 94
- EPROM Burner, 93, 94, 127
- Erasable programmable ROM, *see* EPROM
- Escape sequence, 157, 160, 168
- Event control block, *see* ECB
- Executable file, 39
- EXPECT for program input, 142
- Extrinsic RAM, 66, 67, 143

**F**

- File, 21, 143
  - Auto file, 80
  - Back-up on cassette tape, 118
  - Back-up on PMP, 69
  - Copying, 35
  - Copying to or from a PMP, 68
  - Deleting, 36
  - Leader, 121
  - Nontext, 120
  - Printing, 37
  - Renaming, 36
  - Rules for naming, 23
  - Saving on cassette tape, 121
  - SNAP, 73
  - Text, 120
  - Trailer, 121
  - Transferring from computer to computer, 79
- File name, 23, 36
- File space, 67
- File System, 21, 142
  - Editor, 22, 142
  - Editor quick reference, 130
  - SNAP files, 74
- File type, 39
- Floating point numbers
  - Accuracy of, 45, 143
  - Microsoft BASIC, 88
  - SnapBASIC, 90
- Foreground, 155
- FORTH, 83, 93, 140, 176
- Function key, 13, 147
  - Defining, 62

**G**

- Graphic mode, TV Adaptor, 151, 159, 166, 167, 168, 170
- Graphics
  - Color Plotter, 111
  - LCD, 140
  - TV Adaptor, 123

**H**

- HELP key, 13, 60, 62, 64, 147
- HHC capsule, 7, 71, 92, 127, 137
- HHC/SDT, 93
- High-level language, 82
- Host computer, 85

**I**

- I/O, 95, *see also* Input, Output, Peripheral
- I/O Adaptor, 16, 99, 125
- I/O key, 13, 38, 61, 96, 97, 98, 99, 147
- I/O menu, 37, 38, 67, 96, 97, 100
- Input, 95, 142

Input/output, *see* I/O

- INSERT key, 26, 27, 29, 31, 32, 130, 149
- Internal RAM, 97
- Interpreter, 83
- Interpretive execution, 83, 84, 140
- Intrinsic RAM, 21, 36, 67, 143
- Inverse image, 146
- Invisible file, 40

**K**

- Katakana on TV Adaptor, 157
- Keyboard, 12, 139, 142
  - Auto-repeat feature, 26
  - Calculator's use of, 41
  - Overlay, 16
- LCD, 14, 139, 142, 146
  - Power for, 140
  - Protection of, 20
  - Scrolling, 23
  - Slaving peripheral to, 98
- Library capsule, 75, 143
- Liquid crystal display, *see* LCD
- LOCK key, 14, 29, 32, 130
- Logical unit number, *see* LUN
- LUN, 142, 152

**M**

- Machine language, 81, 140
- Matrix mode, 171
- Matrix, TV Adaptor, 158
- Memory, 47, 48, 49, *see also* RAM, ROM
- Memory map, 137
- Menu
  - I/O, 37, 38, 67
  - Primary, 18
- Message, 80
- Micro Printer, 104, 105
- Mini Printer, 102, 104
- Modem, 72, 77, 115
  - Auto-answer, 117
  - Cassette interface, 118

**O**

- Object program, 82
- OFF key, 12, 128, *see also* ALL-OFF key
- ON key, 12, 128
- Opcode, 157
- Operand, TV Adaptor control command, 157
- Operating system, 14, 137, 140, 141
- Output, 95

**P**

- Page of display RAM in TV Adaptor, 154
- Pen, Color Plotter, 109, 110

- Peripheral, 15, 95, 139, 142
  - Characteristics of, 104, 125
  - Connecting, 96, 136
  - Disconnecting, 98, 136
  - Effect on charging batteries, 20
  - Plastic sleeve holds in place, 15
- Peripheral socket, 11, 16, 99, 138
- Persistent memory feature, 69
- Pixel, 158, 169, 170
- Pixel matrix, 166
- Pixel mode, 166
- Plastic sleeve, 15, 20, 67, 96
- Plotter, *see* Color Plotter
- PMP, 15, 65, 125, 136, 137, 143
- Polish notation, reverse, 75
- Portability of programs, 83
- Primary menu, 18
- Printable character, 146, 148
- Printing a file, 37, 96
- Program, 81
- Program function key, 13, 65, 149
- Programmable Memory Peripheral, *see* PMP
- PROM, *see* EPROM
- Pseudo-machine, 83

**R**

- RAM, 14, 138, 143, *see also* Memory map
  - Extrinsic, 66, 67
  - Intrinsic, 22, 36, 67
  - Power for, 140
  - TV Adaptor, 155, 156, 171
- Random access memory, *see* RAM
- Raw data, 155, 156
- Read-only memory, *see* ROM
- Reverse Polish notation, 75
- ROM, 14, 92, 138
  - TV Adaptor, 157
- ROTATE key, 33, 131, 147
- Rounding in calculator, 45
- RS-232C
  - Cable, 112
  - Capsule, 80
  - Interface, 79, 112

**S**

- Scientific Calculator, 75, 143
- SEARCH key, 33, 131, 149
- 2nd SFT key, 14, 28, 29, 139
- Second shift key, *see* 2nd SFT key
- Semigraphic mode, TV Adaptor, 151, 158, 166, 167, 168, 169
- Serial interface, *see* RS-232C
- SHIFT key, 14, 27, 29, 139
- Slaving peripheral to LCD, 98
- SNAP, 83, 88, 91, 140, 152, 176
- SNAP file, 73

SnapFORTH, 83, 176  
 Source program, 82  
 Squeaker, 140  
 STP/SPD key, 13, 61, 139, 147  
 System Development Tools, *see* HHC/SDT

## T

Target computer, 85  
 Telecomputing, 3, 77, 116, 119, 120  
 TEST/NORMAL switch, 116  
 Text file, 39  
 Thermal printer, 103  
 Time of century clock, 54, 141  
 Transmission rate, *see* Baud  
 Turret of Color Plotter, 107, 109  
   Care of, 110  
   Changing pens, 110

TV Adaptor, 122, 125, 146, 151  
   Use only with AC Adaptor, 126  
 Typing key, 12

## U

Unique character, 146  
 Uploading, 77, 94, 115

## V

Video adaptor, *see* TV Adaptor  
 Video monitor, 122

## W

Word wrap, 105

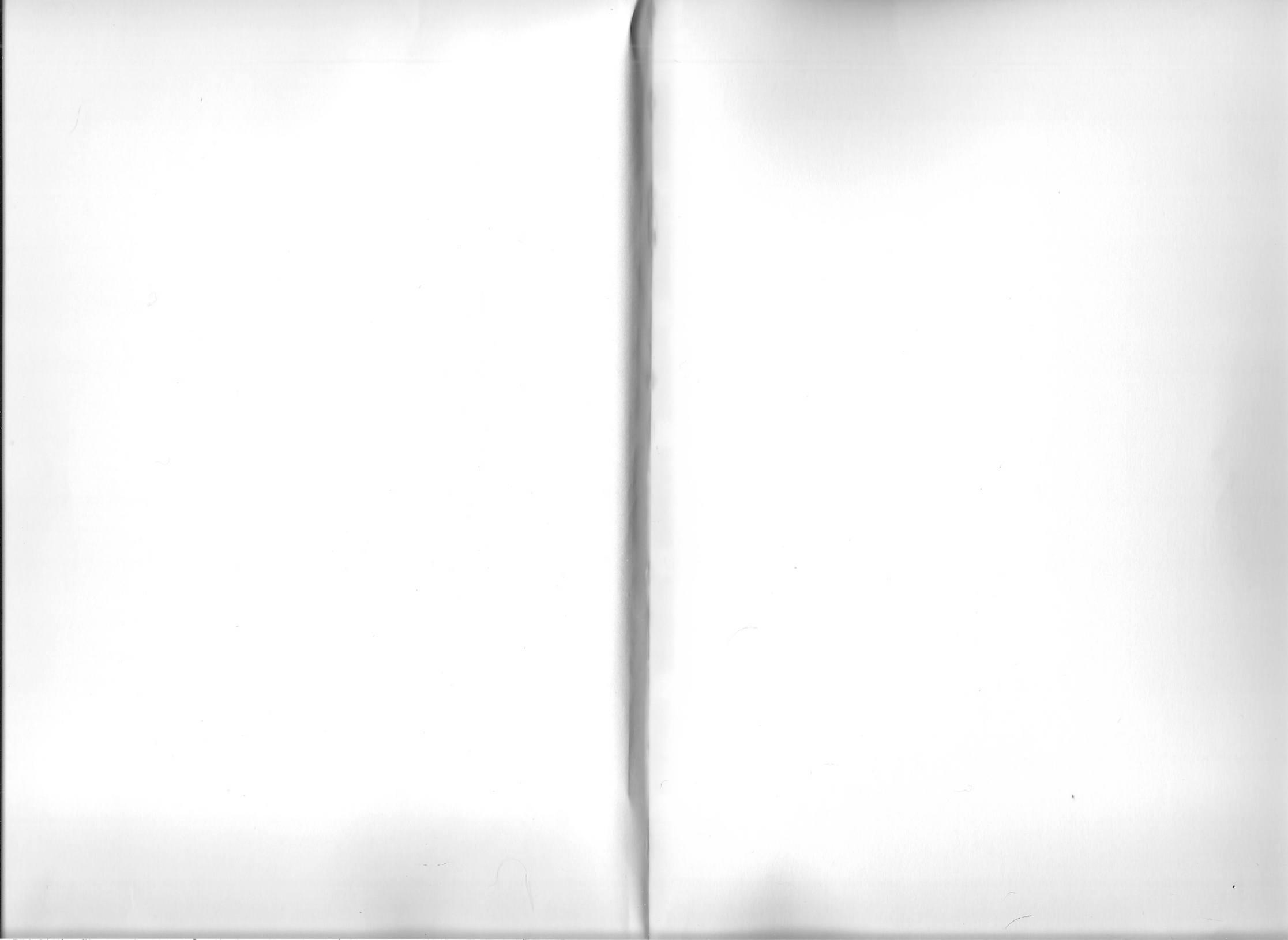


### Other Osborne/McGraw-Hill Publications

An Introduction to Microcomputers: Volume 0—The Beginner's Book, 3rd Edition  
An Introduction to Microcomputers: Volume 1—Basic Concepts, 2nd Edition  
Osborne 4 & 8-Bit Microprocessor Handbook  
Osborne 16-Bit Microprocessor Handbook  
8089 I/O Processor Handbook  
CRT Controller Handbook  
68000 Microprocessor Handbook  
8080A/8085 Assembly Language Programming  
6800 Assembly Language Programming  
Z80 ® Assembly Language Programming  
6502 Assembly Language Programming  
Z8000 ® Assembly Language Programming  
6809 Assembly Language Programming  
Running Wild—The Next Industrial Revolution  
The 8086 Book  
PET ®/CBM™ and the IEEE 488 Bus (GPIB)  
PET ® Personal Computer Guide  
CBM™ Professional Computer Guide  
Business System Buyer's Guide  
Osborne CP/M ® User Guide, 2nd Edition  
Apple II ® User's Guide  
Microprocessors for Measurement and Control  
Some Common BASIC Programs  
Some Common BASIC Programs—PET™/CBM™ Edition  
Some Common BASIC Programs—Atari ® Edition  
Some Common BASIC Programs—TRS-80™ Level II Edition  
Some Common BASIC Programs—Apple II ® Edition  
Some Common BASIC Programs—IBM ® Personal Computer Edition  
Some Common Pascal Programs  
Practical BASIC Programs  
Practical BASIC Programs—TRS-80™ Level Edition  
Practical BASIC Programs—Apple II ® Edition  
Practical BASIC Programs—IBM ® Personal Computer Edition  
Practical Pascal Programs  
Payroll with Cost Accounting  
Accounts Payable and Accounts Receivable  
Accounts Payable and Accounts Receivable CBASIC  
General Ledger  
CBASIC™ User Guide  
Science and Engineering Programs—Apple II ® Edition  
Interfacing to S-100/IEEE 696 Microcomputers  
A User Guide to the UNIX™ System  
PET ® Fun and Games  
Trade Secrets: How to Protect Your Ideas and Assets  
Assembly Language Programming for the Apple II ®  
VisiCalc ®: Home and Office Companion  
Discover FORTH  
6502 Assembly Language Subroutines  
Your ATARI ® Computer  
The HP-IL System  
WordStar ® Made Easy, 2nd Edition  
Armchair BASIC









# The HHC<sup>TM</sup> User Guide

This is the most complete and authoritative source of information on the Panasonic<sup>®</sup> HHC<sup>™</sup>, Quasar<sup>®</sup> HHC<sup>™</sup>, and Olympia<sup>®</sup> HHC<sup>™</sup> portable computers.

- Gives easy-to-use operating instructions for this multipurpose system.
- Thoroughly describes the HHC peripherals including: thermal printers, programmable memory extender, video adapter, modem, color plotter, serial interface, and others.
- Discusses many of the HHC packaged application programs: Electronic Mail, Financial Forecasting, Time Management, Custom Sales Order Entry, and more.
- Covers the HHC programming languages: MBASIC<sup>®</sup>, SnapBASIC, SnapFORTH, the high-level language of the HHC's operating system.

Whether you are using your HHC in the field to collect data or at your desktop as a full-fledged computer system, **THE HHC<sup>™</sup> USER GUIDE** shows you how to get maximum results quickly and easily.

**Jonathan Sachs** has been involved with the HHC since its development at Friends Amis Inc. He wrote a substantial part of the development team's documentation as well as the two-volume user's guide which accompanies the HHC's Microsoft BASIC language capsule. He has also written software for the HHC. Sachs is now an independent software developer and technical writer. He holds a B.S. in Physics from Oberlin College and a J.D. from De Paul Law School.

**Rick Meyer** helped create the HHC capsule programs and development tools while he was Applications Software Manager at Friends Amis Inc. He has published an article on the Quasar/Panasonic HHC that appeared in **BYTE Magazine**. Meyer is now a microcomputer consultant.

Panasonic is a registered trademark of the Panasonic Company.

Quasar is a registered trademark of the Quasar Company (a Division of Matsushita Electric Corporation of America).

HHC is a trademark of the Matsushita Electric Industrial Company, Ltd., JAPAN.

MBASIC is a registered trademark of the Microsoft Corporation.

ISBN 0-931988-87-X

