

AD-A139 438 DATA TRANSFERS AMONG THE HP-75 HP-86 AND HP-9845

1//

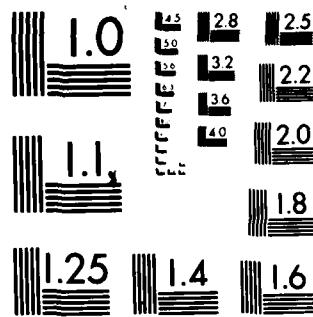
MICROCOMPUTERS(U) AIR FORCE INST OF TECH  
WRIGHT-PATTERSON AFB OH D P CONNOR 1983

UNCLASSIFIED AFIT/CI/NR-83-91T

F/G 9/2

NL


END  
DATE  
FILMED  
18-4  
DTIC



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS 1963 A

AD A139438

UNCLASS

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

612200

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CI/NR-83-91T	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Data Transfers Among the HP-75, HP-86, and HP-9845 Microcomputers		5. TYPE OF REPORT & PERIOD COVERED THESIS/DISSERTATION
7. AUTHOR(s) Daniel Paul Connor		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: University of Central Florida		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (If different from Controlling Office)		12. REPORT DATE 1983
		13. NUMBER OF PAGES 56
		15. SECURITY CLASS. (of this report) UNCLASS
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES APPROVED FOR PUBLIC RELEASE: IAW AFR 190-17 do March 84 LYNN E. WOLAVER Dean for Research and Professional Development AFIT, Wright-Patterson AFB OH		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) ATTACHED		

DTIC  
ELECTE  
MAR 27 1984  
EDD FORM 1473  
1 JAN 73

EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASS

DTIC FILE COPY

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DATA TRANSFERS AMONG THE HP-75,  
HP-86, AND HP-9845 MICROCOMPUTERS

BY

DANIEL PAUL CONNOR  
B.S.E., University of Central Florida, 1981

RESEARCH REPORT

Submitted in partial fulfillment of the requirements  
for the degree of Master of Science in Engineering  
in the Graduate Studies Program of  
the College of Engineering  
University of Central Florida  
Orlando, Florida

Fall Term  
1983



Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

84 03 26 067

# ABSTRACT

↓ This report details procedures for transferring data between the HP-75 and HP-9845 microcomputers, and between the HP-75 and HP-86 microcomputers. Specific guidance is given on the mechanics of interfacing the computers. Software is provided for each computer that will enable the user to transfer data between computers. ↑

A

UNIVERSITY OF CENTRAL FLORIDA

Office of Graduate Studies


RESEARCH REPORT APPROVAL

DATE October 27, 1983




I HEREBY RECOMMEND THAT THE RESEARCH REPORT PREPARED UNDER MY  
SUPERVISION BY Daniel P. Connor

ENTITLED "Data Transfers Among the HP-75, HP-86, and  
HP-9845 Microcomputers"

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE  
DEGREE OF Master of Science in Engineering  
FROM THE COLLEGE OF Engineering

  
Richard N. Miller  
Supervisor of Research Report

Recommendation Concurred In:

  
  
  
R. C. Harlen

Committee on Final Examination

Bruce E. Mathews  
Coordinator of Degree Program

Louis M. Trefonas  
Dean of Graduate Studies

#### ACKNOWLEDGEMENTS

Many people were of considerable help during the preparation of this report. Firstly, I would like to thank Mr. Carl Bishop whose assistance saved me countless hours of research. Secondly, I would like to thank my father, Mr. Joseph P. Connor whose suggestions tremendously reduced the amount of time spent on the manuscript. I would also like to thank International Calculator/Computer Corporation for their assistance. Finally, I would like to thank Dr. Richard N. Miller for his help with this report.

## TABLE OF CONTENTS

INTRODUCTION . . . . .	1
Chapter	
ONE. COMPARING THE THREE COMPUTERS . . . . .	4
TWO. INTERFACE CONCEPTS AND THE HP-IL AND HP-IL/HP-IB SYSTEMS . . . . .	11
THREE. PREREQUISITES FOR USING THE PROGRAMS . . . . .	19
FOUR. PROGRAMS FOR DATA TRANSFERS BETWEEN THE HP-75 AND HP-86 . . . . .	26
FIVE. PROGRAMS FOR DATA TRANSFERS BETWEEN THE HP-75 AND HP-9845 . . . . .	39
SIX. CONCLUSIONS AND POSSIBILITIES FOR FURTHER RESEARCH . . . . .	50
Appendix	
A. LIST OF COMMANDS USED . . . . .	54
B. GLOSSARY . . . . .	57
REFERENCES . . . . .	58



the size of the memories in the 86 and the 9845; so there is a limitation to how much data can be transferred between computers. Another limitation is that data can only be transferred as characters in a string variable. This means that numeric data must always be converted to string data, a change that must be incorporated in the software.

Fortunately, the computers are compatible in many ways. For example, all three use the language BASIC. Almost all of the commands are the same for the three computers. Hence, the lack of proper software for data transfers is the only significant problem; and that is the purpose of this report--to close the "software gap" so that the three computers will be compatible.

Moving data from one machine to another requires a separate program for each computer involved in the transfer. This report provides the necessary software for data transfers between the 75 and the 9845, and between the 75 and the 86 microcomputers.

Several assumptions are made in this report. Among these, familiarity with basic computer and electrical engineering concepts is assumed. A glossary is provided for those not acquainted with some of the terms used in this report. Further, familiarity with all three computers and all of the *fundamental keyboard executable* commands associated with them is assumed.

## INTRODUCTION

The University of Central Florida South Orlando Campus (SOC) recently acquired two desktop computers from Hewlett-Packard (HP) Corporation: the HP-75 (hereafter called the "75") and the HP-86 (hereafter called the "86"). The computers are to be used for classroom instruction and research at SOC. On the main campus another Hewlett-Packard desktop computer, the HP-9845 (hereafter called the "9845"), is already in use; it controls and processes data from a complex electronic network-analysis system that is used for research and classroom instruction. The question arose, Would it be possible to transfer data among the three computers?

There was a problem in that each computer used a different mass-storage system; the 86 used a disc, the 75 used program cards, and the 9845 stored its data on cassette tapes. Finally the question was asked, Could data from either the 86 or the 9845 be stored in the 75, handcarried to the appropriate campus, and then be transferred to the appropriate computer? This is the problem this report addresses.

The problem of transferring data between different models of computers poses certain limitations on the user. For example, the memory in the 75 is less than one-quarter

Since this report is concerned with moving data which is stored in one- and two-dimensional arrays from one machine to another, no attempt will be made to explain how to use the computers to measure data or how to store data. It is assumed that data is already stored in a numeric or string variable or a numeric array in either the 86 or the 9845. String arrays are allowed in the 9845 and the 86, but not in the 75 (HP-75 Owner's Manual, 1982). Hence, they will not be considered. Furthermore, it is assumed that the data will be transferred between the 9845 and the 86 computers using the 75 as the intermediary.

## CHAPTER ONE

### COMPARING THE THREE COMPUTERS

This report is concerned with three microcomputers from Hewlett-Packard Corporation: the HP-75, HP-86, and HP-9845. While the computers perform the same basic functions, they were designed for different purposes; and this manifests itself in the way each computer differs from the other in its external features. These similarities and differences will now be examined.

#### Similarities

How are the computers alike? Each computer uses BASIC language. All three can perform the same scientific calculations. Each computer processes data and commands in the form of 8-bit bytes. With the correct HP interfaces, they can all use the same peripherals. However, there are also major differences in these computers.

#### External Differences

The computers differ in a number of ways, but the first obvious way is external--the computers are not the same size. The 9845 is the largest of the three computers in terms of physical dimensions and memory size. It comes with two tape-storage systems and a printer. The 86 is

smaller than the 9845 both in size and in memory (the 86 features 64 Kbytes of memory while the 9845 has 187 Kbytes). The 86 has no built-in storage system nor a printer. The 75 is smallest in size, has 16 Kbytes of memory, uses magnetic cards for storage, and displays information on an external screen.

Externally, the computers also have different interfaces. The 9845 comes with the IEEE-488 standard HP-IB interface which allows it to easily interact with more than 1000 products developed by over 170 manufacturers in 14 countries (Tutorial Description of the Hewlett Packard Interface Bus, 1980). The 86 has a built-in interface that allows it to function with a HP disc drive and a HP printer. The optional HP-IB interface gives the 86 the ability to function with all of the devices the 9845 can work with. Additionally, the optional HP-IL interface allows the 86 to interact with up to 30 HP-IL devices such as the 75.

The 75 can also work with all of the systems the 9845 can work with through the optional HP-IL/HP-IB interface. In this report, the HP-IL/HP-IB interface is used so that the 75 and the 9845 may transfer data. The HP-IL interface is used with the 75 and the 86 to facilitate data transfers.

### Precision and Memory Requirements

The most important differences among the three computers are those related to precision and memory requirements, and those differences will be examined in detail. First, however, a brief understanding of the variables used in these machines is necessary; a more detailed explanation will be given in Chapter Three, "Prerequisites for Using the Programs."

The computers allow for three types of variables:

1. A simple numeric variable, which is assigned a number value, is permitted.
2. A simple string variable, which is assigned a sequence of valid characters (letters, numbers, or symbols), is allowed.
3. Finally, numeric arrays, which are collections of numbers all having the same precision, can be used.

With an understanding of these variables, the problem of "precision versus memory" can be considered.

When using the 9845, 86, and 75 microcomputers, there is a constant trade-off between number precision and available memory. Number precision is related to the number of digits after the decimal point. Each computer uses three programming statements which set computational precision: the DIM statement, the SHORT statement, and the INTEGER statement. Table 1 lists the precision associated with each statement in each machine.

TABLE 1  
COMPUTER PRECISION

Computer	Precision
9845	DIM: Positive numbers: 1.00000000000E-99 to 9.99999999999E99  Negative numbers: -9.99999999999E99 to -1.00000000000E-99
9845	SHORT: Positive numbers: 1.00000E-63 to 9.99999E63  Negative numbers: -9.99999E63 to -1.00000E-63
9845	INTEGER: ranges from -32768 to +32767
86	DIM: Positive numbers: 1.00000000000E-499 to 9.99999999999E499  Negative numbers: -9.99999999999E499 to -1.00000000000E-499
86	SHORT: Positive numbers: 1.0000E-99 to 9.0000E99  Negative numbers: -9.9999E99 to -1.0000E-99
86	INTEGER: ranges from -99999 to +99999
75	DIM: Positive numbers: 1.00000000000E-499 to 9.99999999999E499  Negative numbers: -9.99999999999E499 to -1.00000000000E-499
75	SHORT: Positive numbers: 1.0000E-99 to 9.9999E99  Negative numbers: -9.9999E99 to -1.0000E-99
75	INTEGER: ranges from -99999 to +99999

Memory requirements for each of the three computers will vary and depend upon the type of precision that is desired and the type of data transaction. Table 2 summarizes how each computer allocates memory.

TABLE 2

## MINIMUM MEMORY REQUIREMENTS FOR THE THREE COMPUTERS

Computer	Precision	Minimum Memory Requirements in Bytes		
		String Variable	Numeric Variable	Numeric Array
9845	DIM	25	10	10 plus 4/dimension plus 8/element
9845	SHORT	19	6	10 plus 4/dimension plus 4/element
9845	INTEGER	13	4	10 plus 4/dimension plus 2/element
86	DIM	30	12	11 plus 8/element
86	SHORT	23	8	11 plus 4/element
86	INTEGER	18	7	11 plus 3/element
75	DIM	27	12	10 plus 8/element
75	SHORT	20	8	10 plus 4/element
75	INTEGER	15	7	10 plus 3/element



### How Many Data Points Can Be Taken?

When using the 75 as an intermediary between the 86 and the 9845, the limitation on the number of data points transferred is determined by the 75 since it is the smallest machine. There are some special cases where the number of data points that can be processed is limited by the software, but these cases will not be considered here. A few examples of calculating memory requirements will now be considered. It is impossible to predict how many data points can be taken (based on available memory) for every case because different applications will require different amounts of memory, but it is hoped that these examples will give some guidance to the user.

Example 1: Suppose many measurements are to be made and stored as data points. If it is assumed that 14 Kbytes of memory are available in the 75, and if the program that will control the measuring process and data storage requires 300 bytes, how many points can be stored for full precision? It is assumed the data will be stored in a one-dimensional numeric array. The memory available after the storage of the control program is 14026 bytes. One numeric array requires 10 bytes plus 8 bytes per element. Hence, the total number of data points that could be stored is found to be 1753. If SHORT precision is used, the total number of points would be 3506.

Example 2: Suppose one desires to store the sample points of a sine wave in a two-dimensional numeric array. Since a sine wave requires two variables to describe it (for example, the sample magnitude and the time the sample was taken), each "data point" or sample would require 16 bytes (8 bytes per variable) if full precision is desired. The total number of samples would be 876. If SHORT precision is used, the total number of samples would be 1753. Either a 876-by-2 element array or a 1753-by-2 element array would be required, depending on precision.

The preceding examples are optimistic because it is a rare occasion when the amount of available memory in the 75 microcomputer exceeds 11 Kbytes. (In examples 1 and 2, this means that the number of data points that could be stored is reduced by 28 percent.) Precision and available memory should always be considered before storing data in any of the three computers considered in this report. Failure to consider memory storage requirements could lead to a loss of essential data.

## CHAPTER TWO

### INTERFACE CONCEPTS AND THE HP-IL AND HP-IL/HP-IB SYSTEMS

What is an interface and what does it do? An interface is a common boundary between two systems or between parts of a system through which information is conveyed. An interface provides compatibility between two systems in four areas: mechanical, electrical, data, and timing (BASIC Language Interfacing Concepts, 1981). These four areas will be examined in some detail. Then the operation of the two systems used in this report, the HP-IL and the HP-IL/HP-IB interfaces, will be summarized.

#### The Purpose of the Interface

The basic compatibility requirement for the interface is that of providing mechanical compatibility. This means the interface must provide the appropriate connectors to the two systems involved. Fortunately, standardized parts are available for this compatibility. For example, the 25-pin RS-232-C connector is used extensively as part of the Electronic Industry Association (EIA) recommended standard for interfacing between data communications

equipment using serial binary data interchange. (This particular interface is optional on both the 86 and 9845.) Further, in Europe about 90 percent of the parallel-bus compatible products use IEEE-488/ANSI MC1.1 connectors which are equivalent to the HP-IB interface (Tutorial Description of the Hewlett-Packard Interface Bus, 1980). In general, most systems are not "plug-to-plug" compatible; but the interfaces used in this report are.

The second purpose of the interface is to match the electrical characteristics (current and voltage levels, sometimes called logic levels) of each system. The 75, 86, and 9845 microcomputers and all associated interfaces use compatible electronic transistor-transistor logic (TTL) levels; these are commonly used in computer hardware. The use of TTL logic means that the interfaces can be designed using prepackaged integrated circuits that will generate and detect voltage levels within a certain range and perform certain logic operations, thus allowing for a smooth transfer of data between computers. The voltage levels commonly used in TTL hardware are dictated by the properties of the transistors themselves and fall into three ranges:

High-level logic:	3 to 5 volts
Indeterminate:	0.7 to 3 volts
Low-level logic:	0 to 0.7 volt

Computers work entirely with high and low logic levels. It is the function of the interface to provide the necessary electrical compatibility between systems. The HP-IL and HP-IL/HP-IB interfaces provide this compatibility.

The third function of an interface is to provide data compatibility. Computers exchange information as electrical signals over wires called data lines. But just as two people who do not speak the same language need a translator, messages between computers require a translator. Sometimes the computer does the translating and sometimes the interface does it. In the case of the HP-IL and HP-IL/HP-IB interfaces, the computer does the translation and such data translation must be incorporated within the program.

The last general function of an interface is to provide timing compatibility. Computers have such a wide range of operating speeds that information could easily be "lost" or "misinterpreted" during a data transfer. The interface must insure that information is conveyed between computers in an orderly and efficient manner. Hence, the interface may slow down the data transfer for a slow computer and speed it up for a fast one. Each interface used in this report has a internal microprocessor that controls the timing process. With this background, the operation of the HP-IL and HP-IL/HP-IB interfaces will now be examined.

### The HP-IL Interface Operation

The HP-IL system (hereafter called the IL system) is a serial-loop system. The word "serial" means data is transferred one bit at a time on a single electrical line. This is in contrast to a bus system where many bits of information may be transferred over many lines at any given time. The word "loop" means that devices are configured in a circular loop with the computer. This is illustrated in Figure 1.

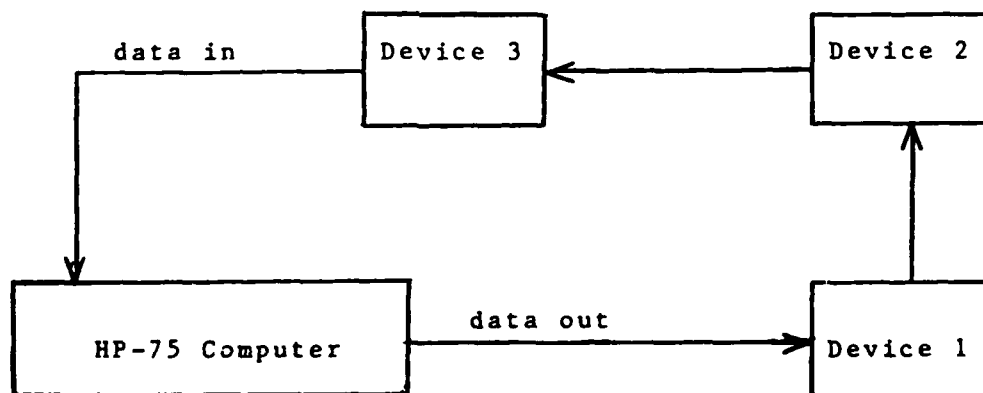


Figure 1. The HP-IL serial-loop system

There is always a "system controller" in the loop, and Hewlett-Packard has designed the 75 microcomputer so that it is always the system controller. In this report, the

only other device on the loop is the 86, and it cannot act as a system controller when the 75 is in the loop. And so the 75 sources all commands and controls the transfer of data. The 86 can only receive or transmit data when told to do so.

In any data-transfer configuration, there are such things as "talkers" and "listeners." A talker is a machine that sends out data to other devices in the system when it is told to do so by the system controller. A listener is a machine that accepts data from some device in the system (the device could be the system controller) at the command of the system controller. There can be only one talker at a time in the IL configuration. The 75 can instruct the 86 to send it some data. Or, the 75 can instruct the 86 to listen while it sends it data or commands.

The system controller, the 75, assigns an address to each device in the loop; in this case, the 86 is the only other device in the loop. Every device in the loop must receive a different device select code. This unique code is incorporated in the software and allows the system controller to communicate with each device on an individual basis. This code can be a letter, two letters, a letter and a digit, or a digit and a letter.

Information is sent from the 75 to the 86 in 11-bit sections called "frames." The first 3 bits identify the frame as either a command frame or a data frame. The last 8 bits are the actual content of the frame. In Figure 1, frames are always sent counterclockwise around the loop--from the 75, to device 1, device 2, device 3, and then back to the 75 completing the loop. The 75 is the source and sink of all frames. With this brief overview of the IL system operation, the IL/IB interface system will now be examined.

#### The HP-IL/HP-IB Interface Operation

The IL/IB interface system will be examined as it concerns this report; that is, with respect to the 75 and the 9845. The IL/IB interface can be operated in any one of three modes:

1. Translator mode with control on the IB side, or the side that has the 9845 as the controller.
2. Translator mode with control on the IL side, or the side that has the 75 as the controller.
3. Mailbox mode where there can be controllers on either side.

The only mode used in this report is the mailbox mode and this is the one that will be examined in detail.



Consider an overall view of IL/IB system in the mailbox mode. In this configuration the 75 is isolated from the 9845. Data can be transferred between computers and each computer can be a active controller. The interface uses two 110-byte buffers (a buffer is a temporary storage location in memory) for bidirectional, independent data transfers. Each computer "thinks" that the interface is the only other device in the system. The interface is assigned two device select codes--one for each computer. Then each computer may store and retrieve data in the buffers in the interface independently of the other computer. Figure 2 illustrates the concept.

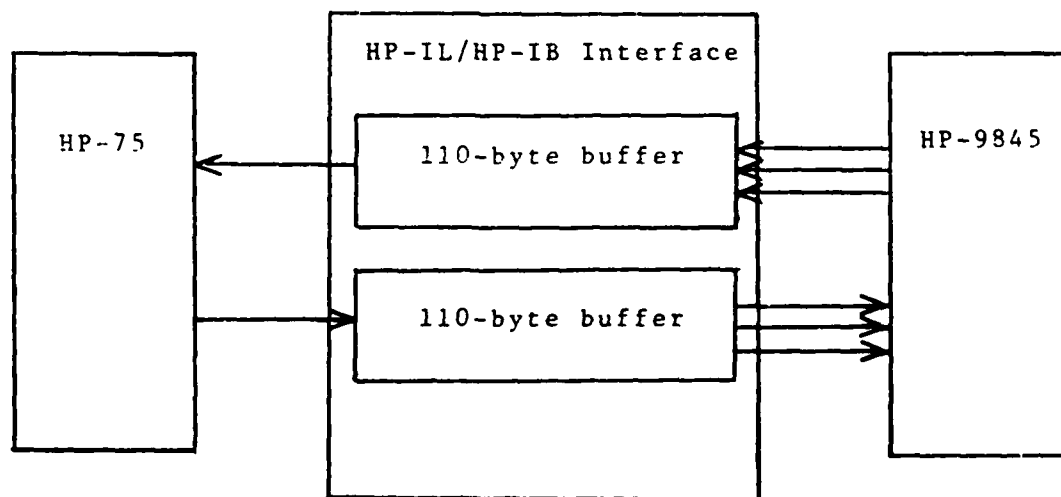


Figure 2. How data is transferred between the 9845 and the 75 through the HP-IL/HP-IB interface.

Operation on the IL side of the interface will now be considered; the system operation will be very similar for the IB side. The ASSIGNIO command is used to assign the interface a device select code through the 75. Now the flow of data can be directed to and from the interface. To send data to the 9845, the SENDIO command is used. The interface stores the data in its buffer, signals the 9845, and then waits for the 9845 to read the data. When the 110-byte buffer is full, the interface suspends operation until after the 9845 has removed some data; then the operation is allowed to continue. The ENTIO\$ command is used to remove data from the interface. This command, used with the "send data" mnemonic--SDA and the "talker active" mnemonic--TAD#, allows the 75 to retrieve data that has been placed there by the 9845. When all of the data has been retrieved from the interface, a "transmission ended" message is sent to the 75. Operations for sending and receiving data on the IB side are the same; only the commands are slightly different.

## CHAPTER THREE

### PREREQUISITES FOR USING THE PROGRAMS

Before the programs documented in this report can be used, certain concepts and procedures must be understood. How do the variables and arrays, introduced in Chapter One, relate to data transfers between the computers? How are arrays dimensioned? Which computers use which interfaces? How are the computers connected together? All of these questions will be answered in this chapter.

#### Variables and Numeric Arrays

As mentioned previously, three types of variables are used in this report--numeric variables, string variables, and numeric arrays. Most data (such as measurements from electronic instruments) is represented in the computer as numbers. However, data is transferred as characters between the computers used in this report. Therefore, a measurement stored as a number must be converted to a set of characters for transfer through the interface. Basically, three types of transactions can occur between the computers:

1. The value of a numeric variable can be changed to a character string, sent through the interface, converted back to its original value, and stored in a numeric variable in the receiving computer.

2. The value of a string variable can be sent to the receiving computer as is and stored in a string variable.

3. Elements in a numeric array can be converted to string characters, sent through the interface, changed back into numeric data, and stored in a numeric array.

A variable can be thought of as a one-dimensional array (that is, a one-by-one array); so anything that applies to numeric arrays, applies to numeric variables as well.

Arrays are dimensioned using the DIM, SHORT, or INTEGER statements. In the following statement, four arrays or variables are declared:

```
10 DIM A(9), B(2,4), C$(19), D$(25)
```

The computer performs the following steps:

1. It allocates enough memory for a 1-dimensional 9-by-1 numeric array, A.
2. It allocates enough memory for a 2-dimensional 2-by-4 numeric array, B.
3. It allocates enough memory for a 19-character string variable (Recall that every string variable is automatically dimensioned to accept up to 18 characters, unless the user specifies otherwise.).
4. It allocates enough memory for a string variable of 25 characters in length.

The next topic is how to interface one computer with another.

### Interfacing the Computers

Before data can be transferred, the computers must be properly connected and the correct programs for each computer must be loaded, ready for use. When transferring data between the 75 and the 86, the IL interface is used. The IL/IB interface is used between the 75 and the 9845. Procedures for setting up all three machines will now be covered.

#### Choosing the Correct Programs

All of the programs documented in this report have been checked for proper operation. Table 3 has been provided to help the user quickly select the proper software for his application.

TABLE 3

#### PROGRAMS FOR DATA TRANSFERS

Type of Transfer	Sending Computer	Receiving Computer	Programs Required	Remarks
string variable	75	86	75: PRG175 86: PRG186	
	86	75	75: PRG275 86: PRG286	
	75	9845	75: PRG175 9845: PRG145	warning, Chap. Five
	9845	75	75: PRG275 9845: PRG245	

TABLE 3 (Continued)

Type of Transfer	Sending Computer	Receiving Computer	Programs Required	Remarks
numeric variable	75	86	75: PRG375 86: PRG386	
or	86	75	75: PRG475 86: PRG486	
numeric array	75	9845	75: PRG375 9845: PRG345	warning Chap. Five
	9845	75	75: PRG475 9845: PRG445	

The same programs used for transferring numeric arrays can be used for transferring numeric variables because the variables can be thought of as "one-by-one" arrays. All of the software documented in this report has been verified for proper operation. Nevertheless, the following steps should be performed before initiating any data transfer:

1. Check to see if all of the variables and arrays are the proper size.
2. Check precision. These programs have been written using full precision and must be changed if SHORT or INTEGER precision is desired.
3. Have the necessary datafiles been created in mass-storage memory in the 86 and the 9845? (The 75 automatically creates the needed files.)
4. What type of numeric arrays are being transferred? These programs have been written assuming two-dimensional

arrays. The nested FOR-NEXT loops must be removed if one-dimensional arrays are used.

5. Verify that the program lines that open and close data files are correct. Does the name of the file to be opened (or closed) agree with the name of the file to be transferred?

Once these checks have been performed, the data transfers should take place smoothly, assuming, of course, that the computers are properly connected. The following procedures explain how to properly interface the computers.

#### The 75, 86, and the IL Interface

The IL interface is used for data transfers between the 75 and the 86 microcomputers. The following procedure is a prerequisite for proper system operation.

1. First, make sure both machines are turned off. Then connect the two long, black, thin IL wires provided with the 75 to the IL interface plugged into the back of the 86. They can only be connected one way.
2. Turn on the 86 first, then the 75 (the system controller is always turned on after the peripherals).
3. Load the proper programs into the appropriate computers and check the software as explained in the previous section, "Choosing the Correct Programs."

4. The correct select codes are already included in the software for the 86; however, the 86 must be assigned a device select code with respect to the 75. Use the ASSIGNIO command in the 75 and assign the 86 a device select code, "8x." (The display on the 75 will read, "Device #1=':8x'.") WARNING: THE DEVICE SELECT CODE MUST BE RE-ASSIGNED EVERY TIME A SYSTEM RESET IS EXECUTED.

Keying in the RUN command (or pressing RUN on both machines) will initiate the data transfer.

The 75, 9845, and the IL/IB Interface

The following procedure is used for data transfers between the 75 and the 9845 via the IL/IB interface:

1. Verify that both machines are turned off. Then connect the supplied AC adaptor to the interface. Do not plug in the interface yet.
2. Connect an IB cable from the network analyzer to the interface. It can only be connected one way.
3. Connect the two long, black, thin IL wires provided with the 75 to the interface. They can only be connected one way.
4. Check to see if the mode address switch on the side of the interface is set to "mailbox" (m=1). The IB device select code switches are preset to binary 27 (11011).  
WARNING: THE PROGRAMS FOR THE 9845 ASSUME A BINARY SELECT CODE OF 27. CHANGING THE SELECT CODE SWITCHES ON THE



INTERFACE WILL REQUIRE APPROPRIATE SOFTWARE CHANGES.

5. Plug in the interface. Then turn on the 75 and the 9845.

6. The interface must be assigned a device select code with respect to the 75. Use the ASSIGNIO command to assign the interface the code "8x." (The display will read, "Device #1=':8x'.") WARNING: THE DEVICE SELECT CODE MUST BE RE-ASSIGNED EVERY TIME A SYSTEM RESET IS EXECUTED.

7. Load the proper programs into the appropriate computers and check the software as explained in the previous section, "Choosing the Correct Programs."

Keying in the RUN command (or pressing the RUN button on both computers) will initiate the data transfer.

WARNING: SPECIAL CONDITIONS APPLY WHEN ANY DATA IS BEING MOVED FROM THE 75 TO THE 9845. REFER TO THE APPROPRIATE SECTIONS OF THIS REPORT AND BE SURE THESE CONDITIONS ARE UNDERSTOOD.

This chapter covered the prerequisites for using the programs this report documents. A "quick-reference" table was provided of the software along with a procedure for checking and adapting the software. Procedures for proper interfacing of the computers were also provided.

## CHAPTER FOUR

### PROGRAMS FOR DATA TRANSFERS BETWEEN THE HP-75 AND HP-86

The following four sets of programs allow for data transfers between the 75 and the 86 microcomputers. Certain instructions and assumptions should be understood before using any of these programs, and these are listed at the beginning of each program set. There are always two programs involved--one for the 75 and one for the 86. These programs have been tried and do work within the limitations of this report. No attempt will be made to explain any errors that might be encountered. Any special warnings or suggestions will be included at the appropriate time. Each program should be thoroughly reviewed and understood before any attempt is made to use it. Chapter Three should be reviewed before the present chapter. WARNING: THE DEVICE SELECT CODE MUST BE RE-ASSIGNED, VIA THE ASSIGNIO COMMAND IN THE 75, EVERY TIME EITHER THE 86 OR THE 9845 IS RESET.

#### The First Program Set

These programs are used to move string data from the 75 to the 86. A file for the data must be created on the disc that the 86 uses for mass storage. The following instructions and assumptions should be carefully read and understood before attempting to use the programs.

### Instructions and Assumptions

1. Read through the sample programs provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.
2. Be sure the machines are connected together properly. (In Chapter Three, see the section: "The 75, 86, and IL Interface.")
3. Load the necessary programs into each machine and key in the appropriate changes.
4. Be sure there is a data file already created in the 86 disc mass-storage system.
5. Run the programs concurrently.

### Sample Program for the 75

```

6   ! Dimension a full-precision
8   ! string variable
10  DIM AS[20]
12  ! Open the data file, "DATA1" in line 20.
20  ASSIGN #2 TO "DATA1"
25  ! In line 30, read the data into AS.
30  READ #2; AS
35  ! Send the data to the 86.
40  SENDIO ':8X','LAD#',AS&CHR$(10)
42  ! Note that in line 40, "8x" is the device select code,
44  ! "LAD#" tells the 86 to prepare for data, AS is that
46  ! data, and "CHR$(10)" signals the end-of-transmission
48  ! and line feed.
49  ! Line 50 will close the data file.
50  ASSIGN #2 TO *
60  END

```

## Sample Program for the 86

```

10  OPTION BASE 1
20  ASSIGN# 2 TO "DATA1"      ! Opens the data file
30  DIM AS[20]
40  ENTER 9;AS
44  ! In line 40 the data is received from the 75
46  ! microcomputer.
50  PRINT# 2; AS              ! Stores the data in the file
60  ASSIGN# 2 TO *            ! Closes the data file
70  END

```

## General Program for the 75 (PRG175)

```

6    ! Dimension a full-precision
8    ! string variable
10   DIM AS[at least 20 characters]
12   ! Open the desired data file in line 20
20   ASSIGN #2 TO "file name"
25   ! Read the data into AS
30   READ #2; AS
35   ! Send the data to the 86
40   SENDIO ':8X','LAD#',AS&CHR$(10)
42   ! Note that in line 40, "8x" is the device select code,
44   ! "LAD#" tells the 86 to prepare for data, AS is that
46   ! data, and "CHR$(10)" signals the end-of-transmission
48   ! and line feed.
49   ! Line 50 will close the data file.
50   ASSIGN #2 TO *
60   END

```

## General Program for the 86 (PRG186)

```

10  DIM AS[at least 20 characters]
18  OPTION BASE 1
20  ASSIGN# 2 TO "file name"  ! Opens the data file
30  ENTER 9; AS               ! Inputs the data from the 75
40  PRINT# 2; AS              ! Stores the data in the file
50  ASSIGN# 2 TO *            ! Closes the data file
60  END

```

### The Second Program Set

These programs are used to move string data (characters) from the 86 to the 75. The 75 will automatically create a file for storing the data; or it will open an existing file of the same name. The following instructions and assumptions should be carefully read and understood before attempting to use the programs.

#### Instructions and Assumptions

1. Read through the sample programs provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.
2. Be sure the machines are connected together properly. (In Chapter Three, see the section: "The 75, 86, and IL Interface.")
3. Load the necessary programs into each machine and key in the appropriate changes.
4. Be sure there is a data file already created in the 86 disc mass-storage system.
5. Run the programs concurrently.

## Sample Program for the 75

```

6  ! Dimension a full-precision string variable in line 8.
8  DIM A$(20)
9  ! Create a data file, "DATA2", in line 10.
10 ASSIGN #4 TO 'DATA2'
15 ! In line 20, the data is obtained from the 86 and
17 ! stored in A$.
20 A$=ENTIOS(':8X','TAD#',SDA')
22 ! In line 20 note that "8x" is the device select code,
24 ! and "TAD#" and "SDA" are commands that tell the 86
26 ! to send its data.
28 ! In line 30 the data is stored in the data file.
30 PRINT #4; A$
35 ! In line 40 the data file is closed.
40 ASSIGN #4 TO *
50 END

```

## Sample Program for the 86

```

7  OPTION BASE 1
10 DIM A$(20) ! Dimension a string variable.
15 ! In line 20, the data file, "DATA2", is opened.
20 ASSIGN# 4 TO "DATA2"
30 READ# 4; A$ ! Read the data into A$.
35 ! In line 40, the data is sent to the 75.
40 OUTPUT 9;A$
50 ASSIGN# 4 TO * ! This closes the data file
60 END

```

## General Program for the 75 (PRG275)

```

6  ! In line 8, a full-precision string variable is declared.
8  DIM A$(at least 20 characters)
9  ! In line 10, a data file is created or opened.
10 ASSIGN #4 TO 'file name'
15 ! Obtain the data from the 86 and store it in
17 ! A$ temporarily.
20 A$=ENTIOS(':8X','TAD#',SDA')
22 ! In line 20 note that "8x" is the device select code,
24 ! and "TAD#" and "SDA" are commands that tell the 86
26 ! to send data to the 75.
28 ! In line 30 the data is stored in the data file.
30 PRINT #4; A$
35 ! In line 40 the data file is closed.
40 ASSIGN #4 TO *
50 END

```

## General Program for the 86 (PRG286)

```

7  OPTION BASE 1
10 DIM A$(at least 20 characters)
15 ! In line 20, the data file is opened.
20 ASSIGN# 4 TO "file name"
30 READ# 4; A$
35 ! In line 40 the data is sent to the 75.
40 OUTPUT 9; A$
50 ASSIGN# 4 TO *           ! This closes the data file
60 END

```

## The Third Program Set

These programs are used to move data stored in numeric variables or arrays from the 75 to the 86. A file for the data must be created on the disc that the 86 uses for mass storage. A FOR-NEXT loop must be used to transfer the data. This is because the data must be converted from numbers to characters for transfer through the interface. This conversion must be done one element (of the array) at a time. The following instructions and assumptions should be carefully read and understood before attempting to use these programs to transfer data.

## Instructions and Assumptions

1. Read through the sample programs provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.

2. Be sure the machines are connected together properly. (In Chapter Three, see the section: "The 75, 86, and IL Interface.")

3. Load the necessary programs into each machine and key in the appropriate changes. Though these programs have been written for two-dimensional arrays, instructions concerning software changes for numeric variables and one-dimensional arrays are included as "comment statements" in the listing of each program.

4. Run the programs concurrently.

#### Sample Program for the 75

```

2   ! Line 10 dimensions a full-precision, two-dimensional
4   ! array. For a one-dimensional array, there is only
6   ! one number (the dimension) in the parenthesis.
8   ! For a variable, only the variable name is given.
10  DIM A(2,2), AS[20]
15  ! Line 20 opens or creates the data file, "DATA4"
20  ASSIGN #6 TO "DATA4"
22  ! In line 30, list only the variable name after the
24  ! semicolon for a variable transfer. The comma between
26  ! the parenthesis is removed for one-dimensional arrays.
30  READ # 6; A(,)
32  ! The following FOR-NEXT loop is eliminated for a
34  ! variable. This means lines 40, 50, 80, and 90
36  ! must be deleted. For a one-dimensional array, delete
38  ! lines 50 and 90. Do not delete lines 60 and 70.
40  FOR I=1 TO 2
50  FOR J=1 TO 2
52  ! Line 60 will convert each number (array element)
54  ! to a string character. For a one-dimensional array,
56  ! the "J" would be deleted. For a variable, the line
58  ! would read, "AS=STR$(variable name)."
60  AS=STR$(A(I,J))

```



```

65 ! Each element is sent to the 86 in line 70.
70 SENDIO ':8X','LAD#', A$&CHRS(10)
72 ! Note in line 70 that "8x" is the device select code.
80 NEXT J
90 NEXT I
100 ! The data file is closed in line 110.
110 ASSIGN #6 TO *
120 END

```

#### Sample Program for the 86

```

1  OPTION BASE 1
2  ! Line 10 dimensions a full-precision, two-dimensional
4  ! array. For a one-dimensional array, there is only
6  ! one number (the dimension) in the parenthesis.
8  ! For a variable, only the variable name is given.
10 DIM A(2,2), A$(20)
15 ! Line 20 opens or creates the data file, "DATA4".
20 ASSIGN #6 TO "DATA4"
22 ! The FOR-NEXT loop is eliminated for a variable
24 ! transfer. Lines 50 and 90 are deleted for a one-
26 ! dimensional array. Do not delete lines 60 and 70.
40 FOR I=1 TO 2
50 FOR J=1 TO 2
60 ENTER 9; A$
62 ! Line 70 will convert the string data to numeric
64 ! data. For a variable, the parenthesis are deleted.
66 ! The "J" is deleted for a one-dimensional array
70 A(I,J)=VAL(A$)
90 NEXT J
100 NEXT I
102 ! In line 110 the data is stored on disc. For a
104 ! one-dimensional array, there is no comma between
106 ! the parenthesis. For a variable, only the variable
108 ! name is given after the semicolon.
110 PRINT# 6; A(,)
120 ASSIGN# 6 TO *
130 END

```

## General Program for the 75 (PRG375)

```

2   ! Line 10 dimensions a full-precision, two-dimensional
4   ! array. For a one-dimensional array, there is only
6   ! one number (the dimension) in the parenthesis.
8   ! For a variable only the variable name is given.
10  DIM A(dimension 1,dimension 2), AS[20]
15  ! Line 20 opens or creates the data file.
20  ASSIGN #6 TO "file name"
22  ! In line 30, delete the comma between the parenthesis
24  ! for a one-dimensional array. For a variable transfer,
26  ! the line should read, "READ #6,1; variable name".
30  READ # 6; A(,)
32  ! The following FOR-NEXT loop is eliminated for a
34  ! variable. This means lines 40, 50, 80, and 90
36  ! must be deleted. For a one-dimensional array, delete
38  ! lines 50 and 90. Do not delete lines 60 and 70.
40  FOR I=1 TO dimension 1
50  FOR J=1 TO dimension 2
52  ! Line 60 will convert each number (array element)
54  ! to a string character. For a one-dimensional array,
56  ! the "J" would be deleted. For a variable, the line
58  ! would read, "AS=STR$(variable name)".
60  AS=STR$(A(I,J))
65  ! Each element is sent to the 86 in line 70.
70  SENDIO ':8X','LAD#', AS&CHRS(10)
72  ! Note in line 70 that "8x" is the device select code.
80  NEXT J
90  NEXT I
100 ! Line 110 closes the data file.
110 ASSIGN #6 TO *
120 END

```

## General Program for the 86 (PRG386)

```

1   OPTION BASE 1
2   ! Line 10 dimensions a full-precision, two-dimensional
4   ! array. For a one-dimensional array, there is only
6   ! one number (the dimension) in the parenthesis.
8   ! For a variable, only the variable name is given.
10  DIM A(dimension 1,dimension 2), AS[20]
15  ! Line 20 opens or creates the data file.
20  ASSIGN #6 TO "file name"
22  ! The FOR-NEXT loop is eliminated for a variable
24  ! transfer. Lines 50 and 90 are deleted for a one-
26  ! dimensional array. Do not delete lines 60 and 70.
40  FOR I=1 TO dimension 1
50  FOR J=1 TO dimension 2
60  ENTER 9; AS

```

```

62 ! For a variable transfer, line 70 should read,
64 ! "variable name=VAL(AS)". For a one-dimensional array,
66 ! the "J" is deleted.
70 A(I,J)=VAL(AS)
90 NEXT J
100 NEXT I
102 ! In line 110 the data is stored on disc. For a
104 ! one-dimensional array, there is no comma between
106 ! the parenthesis. For a variable, only the variable
108 ! name is given after the semicolon.
110 PRINT# 6; A(,)
120 ASSIGN# 6 TO *
130 END

```

#### The Fourth Program Set

These programs are used to move data stored in numeric variables or arrays from the 86 to the 75. The 75 will automatically open or create the necessary data file. The data must be converted to string characters, one element (number) at a time, for transfer through the interface. The following instructions and assumptions should be carefully read and understood before attempting to use these programs.

#### Instructions and Assumptions

1. Read through the sample programs provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.

2. Be sure the machines are connected together properly. (In Chapter Three, see the section: "The 75, 86, and IL Interface.")

3. Load the necessary programs into each machine and key in the appropriate changes. Though these programs have been written for two-dimensional arrays, instructions concerning software changes for numeric variables and one-dimensional arrays are included as "comment statements" in the listing of each program.

4. Run the programs concurrently.

Sample Program for the 75

```

4  C$="&"
5  OPTION BASE 1
6  ! In line 10 a two-dimensional array is declared.
7  ! For a one-dimensional array, there would only be
8  ! one number in the parenthesis. When declaring a
9  ! variable, only the variable name is listed.
10 DIM A(2,2), AS[20]
30 ASSIGN #8 TO "DATA4"
32 ! The following FOR-NEXT loop would be eliminated for
34 ! a variable transfer. For a one-dimensional array,
36 ! lines 50 and 100 would be deleted. Do not delete
38 ! lines 60 or 70.
40 FOR I=1 TO 2
50 FOR J=1 TO 2
60 AS=ENTIO$(':8X','TAD#',SDA')
62 ! In line 70 each number is converted from a character
64 ! to a number. For a one-dimensional array, the "J"
66 ! would be deleted. For a variable, the line would
68 ! read, "variable name=VAL(AS)".
70 A(I,J)=VAL(AS)
80 WAIT 0.1
90 SENDIO ':8X','LAD#',C$
100 NEXT J
110 NEXT I
112 ! In line 120 the data is stored in the data file.
114 ! For a numeric variable, only the variable name
116 ! would appear after the semicolon. There would be no
118 ! comma in between the parenthesis for a one-dimen-
119 ! sional array.
120 PRINT #8; A(,)
130 ASSIGN #8 TO *
140 END

```

## Sample Program for the 86

```

5  OPTION BASE 1
6  ! In line 10, a two-dimensional array is declared.
7  ! For a one-dimensional array, there would only be
8  ! one number in the parenthesis. When declaring a
9  ! variable, only the variable name is listed.
10 DIM A(2,2), A$(20)
30 ASSIGN #8 TO "DATA4"
32 ! In line 40 the data is read from the file into the
34 ! array. For a one-dimensional array, there is no
36 ! comma between the parenthesis. For a variable,
38 ! only the variable name appears after the semicolon.
40 READ# 8; A(,)
42 ! The following FOR-NEXT loop would be eliminated for
44 ! a variable transfer. Lines 60 and 110 would be
46 ! deleted for a one-dimensional array. Do not delete
48 ! lines 70 and 100.
50 FOR I=1 TO 2
60 FOR J=1 TO 2
61   CS="0"
62   ! In line 70, the numbers are converted to characters.
64   ! For a one-dimensional array, the "J" would be deleted.
66   ! When converting a variable, line 70 would read,
68   ! "AS=VAL$(variable name).".
70   AS=VAL$(A(I,J))
100  OUTPUT 9; AS
102  ENTER 9; CS
104  IF CS <> "&" THEN 102
110  NEXT J
120  NEXT I
140  ASSIGN# 8 TO *
150  END

```

## General Program for the 75 (PRG475)

```

4  CS="&"
5  OPTION BASE 1
6  ! In line 10 a two-dimensional array is declared.
7  ! For a one-dimensional array, there would only be
8  ! one number in the parenthesis. When declaring a
9  ! variable, only the variable name is listed.
10 DIM A(dimension 1,dimension 2), A$(20)
30 ASSIGN #8 TO "file name"
32 ! The following FOR-NEXT loop would be eliminated for
34 ! a variable transfer. For a one-dimensional array,
36 ! lines 50 and 100 would be deleted. Do not delete
38 ! lines 60 or 70.
40 FOR I=1 TO dimension 1
50 FOR J=1 TO dimension 2

```

```

60 AS=ENTIOS(':8X','TAD#,SDA')
62 ! In line 70 each number is converted from a character
64 ! to a number. For a one-dimensional array, the "J"
66 ! would be deleted. For a variable, the line would
68 ! read, "variable name=VAL(AS)".
70 A(I,J)=VAL(AS)
80 WAIT 0.1
90 SENDIO ':8X','LAD#', CS
100 NEXT J
110 NEXT I
114 ! For a numeric variable, only the variable name
116 ! would appear after the semicolon. There would be no
118 ! comma in between the parenthesis for a one-dimen-
119 ! sional array.
120 PRINT #8; A(,)
130 ASSIGN #8 TO *
140 END

```

General Program for the 86 (PRG486)

```

5  OPTION BASE 1
7  ! In line 10, there would be only one number in the
8  ! parenthesis for a one-dimensional array. When declar-
9  ! ing a variable, only the variable name is listed.
10 DIM A(dimension 1,dimension 2), AS[20]
30 ASSIGN #8 TO "file name"
32 ! In line 40 the data is read from the file into the
34 ! array. For a one-dimensional array, there is no
36 ! comma between the parenthesis. For a variable,
38 ! only the variable name appears after the semicolon.
40 READ# 8; A(,)
42 ! The following FOR-NEXT loop would be eliminated for
44 ! a variable transfer. Lines 60 and 110 would be
46 ! deleted for a one-dimensional array. Do not delete
48 ! lines 70 and 100.
50 FOR I=1 TO dimension 1
60 FOR J=1 TO dimension 2
61 CS="0"
62 ! In line 70, the numbers are converted to characters.
64 ! For a one-dimensional array, the "J" would be deleted.
66 ! When converting a variable, line 70 would read,
68 ! "AS=VAL$(variable name)."
70 AS=VAL$(A(I,J))
100 OUTPUT 9; AS
102 ENTER 9; CS
104 IF CS <> "&" THEN 102
110 NEXT J
120 NEXT I
140 ASSIGN# 8 TO *
150 END

```

## CHAPTER FIVE

### PROGRAMS FOR DATA TRANSFERS BETWEEN THE HP-75 AND HP-9845

The following four sets of programs allow for data transfers between the 75 and the 9845 microcomputers. Certain instructions and assumptions should be understood before using any of these programs, and these are listed at the beginning of each program set. There are always two programs involved--one for the 75 and one for the 9845. The programs for the 75 have already been listed and explained in detail in Chapter Four. The pages where the appropriate programs can be found will be listed with the instructions for each program set. Each program set will consist of a sample and a general program for the 9845.

These programs have been tried and do work within the limitations of this report. No attempt will be made to explain any errors that might be encountered. Any special warnings or suggestions will be included at the appropriate time. Each program should be thoroughly reviewed and understood before any attempt is made to use it. Chapter Three and the sections in Chapter Four concerning the 75 should both be read before this chapter is read. **WARNING: THE DEVICE SELECT CODE MUST BE RE-ASSIGNED, VIA THE ASSIGNIO COMMAND IN THE 75, EVERY TIME THE 9845 IS RESET.**

### The First Program Set

These programs are used to move string data from the 75 to the 9845. A file for the data must be created on the tape that the 9845 uses for mass storage.

The speed of program execution of the 9845 is much higher than that of the 75. Because of this special care must be taken when moving data from the 75 to the 9845. The 75 must be allowed to fully execute its program before the 9845. Basically, this means that the user must press the RUN button on the 75 first, wait until the "PRGM" prompt vanishes from the one-line video screen on the 75, and then execute the program in the 9845. WARNING: THE 75 MUST BE ALLOWED TO FULLY EXECUTE ITS ROUTINE BEFORE RUNNING THE PROGRAM IN THE 9845. ANY ATTEMPT AT RUNNING BOTH ROUTINES SIMULTANEOUSLY WILL RESULT IN AN IMPROPER DATA TRANSFER.

### Instructions and Assumptions

1. Read through the sample program provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.

2. Be sure the machines are connected together properly. (See the section in Chapter Three: "The 75, 9845, and the IL/IB Interface.")



3. Load the necessary programs into each computer and key in the appropriate changes. The correct program for the 75 is "PRG175" and the instructions for using it can be found in Chapter Four, along with the general program.

4. Be sure there is a data file already created in the 9845 tape mass-storage system.

5. DO NOT RUN THE PROGRAMS CONCURRENTLY. Execute the program in the 75 first, then run the program in the 9845.

#### Sample Program for the 9845

```

5   ! In line 10, the string variable is dimensioned for
6   ! 20 characters--19 characters for the full-precision
7   ! number and 1 character for the line feed. This is
8   ! the minimum allowable size for a string variable.
10  DIM A$(20)
15  ! In line 20, the data file, "DATA1" is opened.
20  ASSIGN #2 TO "DATA1"
24  ! In line 30, the data is accepted from the 75 and
26  ! stored in the variable, A$. The number 727 is the
27  ! device select code for the IL/IB interface. It
28  ! should not be changed.
30  ENTER 727; A$
40  DISP A$
44  ! In line 50 the data is printed in the data file.
50  PRINT #2; A$
60  ASSIGN #2 TO *
70  END

```

#### General Program for the 9845 (PRG145)

```

5   ! In line 10, the string variable is dimensioned for
6   ! 20 characters--19 characters for the full-precision
7   ! number and 1 character for the line feed. This is
8   ! the minimum allowable size for a string variable.
10  DIM A$(20)
15  ! In line 20, the data file is opened.
20  ASSIGN #2 TO "file name"

```

```

24 ! In line 30, the data is accepted from the 75 and
26 ! stored in the variable, A$. The number 727 is the
27 ! device select code for the IL/IB interface. It
28 ! should not be changed.
30 ENTER 727; A$
40 DISP A$
44 ! In line 50 the data is printed in the data file.
50 PRINT #2; A$
60 ASSIGN #2 TO *
70 END

```

### The Second Program Set

These programs are used to move string data from the 9845 to the 75. The 75 will automatically create a file for storing the data; or it will open an existing file. The following instructions and assumptions should be carefully read and understood before attempting to use the programs.

#### Instructions and Assumptions

1. Read through the sample programs provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.

2. Be sure the machines are connected together properly. (See the section in Chapter Three: "The 75, 9845, and the IL/IB Interface.")

3. Load the necessary programs into each computer and key in the appropriate changes. The correct program for the 75 is "PRG275" and the instructions for using it can be found in Chapter Four, along with the general program.

4. Do not run the programs concurrently. Execute the program in the 9845 first, then run the routine in the 75.

#### Sample Program for the 9845

```

10 DIM A$(20)
20 ASSIGN #4 TO "DATA1"
24 ! In lines 30 and 40, the data is read from the file
26 ! and sent to the 75. The code number 727 should not
28 ! be changed either on the interface or in the program.
30 READ #4; A$
40 OUTPUT 727; A$
50 ASSIGN #4 TO *
60 END

```

#### General Program for the 9845 (PRG245)

```

10 DIM A$(20)
15 ! In line 20, the data file is opened.
20 ASSIGN #4 TO "file name"
24 ! In lines 30 and 40, the data is read from the file
26 ! and sent to the 75. The code number 727 should not
28 ! be changed either on the interface or in the program.
30 READ #4; A$
40 OUTPUT 727; A$
50 ASSIGN #4 TO *
60 END

```

### The Third Program Set

These programs are used to move data stored in numeric variables or arrays from the 75 to the 9845. A file for the data must be available on the tape that the 9845 uses for mass storage. A FOR-NEXT loop must be used for an orderly transfer of data. This is because the data must be converted from numbers to characters for transfer through the interface. This conversion is done one element (of the array) at a time.

The speed of the 9845 is much higher than that of the 75. Extra care must be taken when attempting to transfer data between the two computers. The routine in the 75 must be initiated first. The user should then wait several seconds before running the program in the 9845. It is not necessary to wait until the 75 has completed program execution; only a short pause is needed. WARNING: THE PROGRAM IN THE 9845 IS EXECUTED ONLY AFTER THE ROUTINE IN THE 75 HAS BEEN RUNNING FOR SEVERAL SECONDS. ATTEMPTING TO RUN BOTH PROGRAMS CONCURRENTLY COULD LEAD TO AN IMPROPER DATA TRANSFER.

### Instructions and Assumptions

1. Read through the sample program provided first; then substitute the necessary parameters (that is, variables, size of arrays, data file names--parameters that will vary from application to application) into the general programs. These substitutions should be made on paper first and then keyed into the machines later.

2. Be sure the computers are connected together properly. (See the section in Chapter Three: "The 75, 9845, and the IL/IB Interface.")

3. Load the necessary programs into each computer and key in the appropriate changes. The correct program for the 75 is "PRG375" and the instructions for using it can be found in Chapter Four, along with the general program.

4. Be sure there is a data file already created in the 9845 tape mass-storage system.

5. DO NOT RUN THE PROGRAMS CONCURRENTLY. Instead, wait a few seconds after pressing the RUN button on the 75, and then execute the program in the 9845.

### Sample Program for the 9845

```

10 OPTION BASE 1
12 ! In line 20 a two-dimensional array is declared. For
14 ! a one-dimensional array, only one number would be
16 ! within the parenthesis. For a numeric variable, only
18 ! the variable name would be declared.
20 DIM A(2,2), AS[20]
22 ! In line 30 the data file, "DATA2" is opened.
30 ASSIGN #6 TO "DATA2"
```

```

32 ! The FOR-NEXT loop in lines 40 through 90 are for
34 ! two-dimensional arrays. For one-dimensional arrays,
36 ! delete lines 50 and 80. For variable transfers,
38 ! delete lines 40, 50, 80, and 90. Do not remove
39 ! lines 60 and 70.
40 FOR I=1 TO 2
50 FOR J=1 TO 2
55 WAIT 299
60 ENTER 727; A$
62 ! In line 70, the data is converted from a number to
64 ! a character. For a one-dimensional array, the "J"
66 ! would be deleted. For a variable, only the variable
68 ! would be within the parenthesis.
70 A(I,J)=VAL(A$)
80 NEXT J
90 NEXT I
92 ! In line 100 the data is stored in the data file.
94 ! To store data from a variable, place the variable
96 ! name after the semicolon.
100 PRINT #6; A(*)
110 ASSIGN #6 TO *
120 END

```

General Program for the 9845 (PRG345)

```

10 OPTION BASE 1
12 ! In line 20 a two-dimensional array is declared. For
14 ! a one-dimensional array, only one number would be
16 ! within the parenthesis. For a numeric variable, only
18 ! the variable name would be declared.
20 DIM A(dimension 1,dimension 2), A$(20)
30 ASSIGN #6 TO "file name"
32 ! The FOR-NEXT loop in lines 40 through 90 are for
34 ! two-dimensional arrays. For one-dimensional arrays,
36 ! delete lines 50 and 80. For variable transfers,
38 ! delete lines 40, 50, 80, and 90. Do not remove
39 ! lines 60 and 70.
40 FOR I=1 TO dimension 1
50 FOR J=1 TO dimension 2
55 WAIT 299
60 ENTER 727; A$
62 ! In line 70, the data is converted from a number to
64 ! a character. For a one-dimensional array, the "J"
66 ! would be deleted. For a variable, line 70 would
68 ! read: "variable name=VAL(A$)".
70 A(I,J)=VAL(A$)
80 NEXT J
90 NEXT I

```

```
92 ! In line 100 the data is stored in the data file.  
94 ! To store data from a variable, place the variable  
96 ! name after the semicolon.  
100 PRINT #6; A(*)  
110 ASSIGN #6 TO *  
120 END
```

#### The Fourth Program Set

These programs are used to move numeric variables and arrays from the 9845 to the 75. The 75 will automatically create or open a file for the data. The following instructions and assumptions should be read and understood before any attempt is made to use these programs.

#### Instructions and Assumptions

1. Read through the sample program provided first; then change the general-program parameters as necessary.
2. Be sure the computers are connected together properly. (See the section in Chapter Three: "The 75, 9845, and the IL/IB Interface.")
3. Load the necessary programs into each computer and key in the appropriate changes. The correct program for the 75 is "PRG475" and the instructions for using it can be found in Chapter Four, along with the general program.
4. Do not run the programs concurrently. Start the program in the 9845 first. Wait until the first number is displayed on the 9845, then start the program in the 75.

## Sample Program for the 9845

```

10  OPTION BASE 1
12  ! In line 20 a two-dimensional array is declared.  For
14  ! a one-dimensional array, there will be only one number
16  ! within the parenthesis.  For a variable, only the
18  ! variable name is declared.
20  DIM A(2,2), A$(20)
30  ASSIGN #8 TO "DATA2"
32  ! In line 40, the data is read from the file into the
34  ! array.  To read data into a variable, place the
36  ! variable name after the semicolon.
40  READ #8; A(*)
42  ! The FOR-NEXT loop is written for a two-dimensional
44  ! array.  When using a one-dimensional array, delete
46  ! lines 60 and 90.  When transferring data in a
48  ! variable, delete lines 50, 60, 90, and 100.
49  ! Do not delete lines 70 and 80.
50  FOR I=1 TO 2
60  FOR J=1 TO 2
61  C$="0"
62  ! In line 70 the data is converted into characters.
64  ! For a one-dimensional array, the "J" is deleted.
66  ! To convert the data in a variable to string data,
68  ! use: "A$=VAL$(variable name)".
70  A$=VAL$(A(I,J))
80  OUTPUT 727; A$
82  PRINT A(I,J)
84  ENTER 727; C$
86  IF C$ <> "&" THEN 84
90  NEXT J
100 NEXT I
110 ASSIGN #8 TO *
120 END

```

## General Program for the 9845 (PRG445)

```

10  OPTION BASE 1
12  ! In line 20 a two-dimensional array is declared.  For
14  ! a one-dimensional array, there will be only one number
16  ! within the parenthesis.  For a variable, only the
18  ! variable name is declared.
20  DIM A(dimension 1,dimension 2), A$(20)
30  ASSIGN #8 TO "file name"
32  ! In line 40, the data is read from the file into the
34  ! array.  To read data into a variable, place the
36  ! variable name after the semicolon.
40  READ #8; A(*)

```



```
42 ! The FOR-NEXT loop is written for a two-dimensional
44 ! array. When using a one-dimensional array, delete
46 ! lines 60 and 90. When transferring data in a
48 ! variable, delete lines 50, 60, 90, and 100.
49 ! Do not delete lines 70 and 80.
50 FOR I=1 TO dimension 1
60 FOR J=1 TO dimension 2
61 C$="0"
62 ! In line 70 the data is converted into characters.
64 ! For a one-dimensional array, the "J" is deleted.
66 ! To convert the data in a variable to string data,
68 ! use: "A$=VAL$(variable name)".
70 A$=VAL$(A(I,J))
72 ! In line 80, the number 727 is the device select
74 ! code and should not be changed, either on the
76 ! interface or in the program.
80 OUTPUT 727; A$
82 PRINT A(I,J)
84 ENTER 727; C$
86 IF C$ <> "&" THEN 34
90 NEXT J
100 NEXT I
110 ASSIGN #8 TO *
120 END
```

## CHAPTER SIX

### CONCLUSIONS AND POSSIBILITIES FOR FURTHER RESEARCH

In the Introduction the question was asked, Would it be possible to transfer data among the HP-75, HP-86, and HP-9845 microcomputers? It was suggested that such an idea was indeed feasible. The purpose of this report was to provide the necessary computer programs that would make data transfers among the three computers possible, because a lack software for each machine was seen as the only significant problem.

Chapters Three, Four, and Five document and explain the computer programs necessary for ordinary data transfers between the Hewlett-Packard machines, and guidance is given on the mechanics of interfacing the computers. All of the programs have been tested and have displayed acceptable performance. The programs for data exchanges between the 75 and the 9845 are slow. However, many hours of experimentation went into trying to increase the speed of the data transactions, and it was concluded that there could be no significant speed increase without a major change in either the computer software or hardware. For example, when data was being moved from the 9845 to the 75, there was a time delay between the time the 9845 placed the data in the

110-byte buffer and the moment when the data was available for storage in the 75 microcomputer. This time delay was found to be at least one-tenth of a second. The delay was a function of how fast the data was shifted through the buffer; and it could not be controlled.

All of the programs are limited in at least two ways.

1. The programs are limited in that they allow single variable or single numeric array data transfers. Multiple variable or multiple array transfers will require more complex software.

2. As mentioned previously, the amount of data that can be transferred between computers is limited by the amount of available memory in the 75.

Because of these limitations, several avenues of new research are suggested in the following section.

#### Future Research Topics

Four possibilities for research on the subject of information transfers between computers are suggested below. Whenever possible, cautions, ways to limit the problem, or suggested solutions are provided.

Case 1: The programs in this report provide for the transfer of data stored in either single variables or single numeric arrays. However, this is not the general case. It is not uncommon in most scientific analyses to have data stored in many variables and arrays. Therefore, it would be

nice if one could transfer many data arrays and variables at the same time. Using the programs provided herein as a basis, this problem can be solved through the use of FOR-NEXT loops and nested FOR-NEXT loops.

Case 2: Occasionally, one may wish to transfer entire computer programs between the machines. The limitation on the size of the routine that can be transferred will be determined by the amount of available memory in the 75, the amount of space on the 86 disc, or the amount of room left on the cassette tape for the 9845. The HP-IL interface will allow only data transfers, so any program will have to be converted to some type of data for transfer through the interface. Furthermore, all data that is sent to either the 86 or the 9845 is stored in a data file, not in a program file. In like manner, the 75 stores all of the data it receives in a data file, not in a program file. One possible solution to this problem would be to rename a program file (already stored in one of the computers) as a data file, and then transfer the data file to the appropriate machine. The program file would have to be treated as a numeric array, and each programming statement would make up one "element" of the array. For example, a software routine with 20 programming statements in it would be considered a

20-by-1 array. Once the routine is in the proper form, the programs listed in this report could be used to transfer the "data." The last step would be converting the data file back into a program file.

Case 3: One may wish to plot data as it is being transferred between computers; particularly since both the 9845 and the 86 have plotting capabilities. The programs in this report only allow for the transfer of data between machines. There is no processing done with the data. However, a program could be written that would plot the data even as it is being stored in mass-storage memory.

Case 4: Data is frequently stored in string arrays when working with the 9845 and the 86 microcomputers. Unfortunately, the 75 can not handle the string array function. Rather than change all of the software for the 9845 and the 86, it would be nice if string arrays could be simulated on the 75 so that data transfers could take place. The concept of simulating string arrays is documented in the owner's manuals for both the 9845 and the 75.

## APPENDIX A

### LIST OF COMMANDS USED

Command/Statement	Computer	Reference
ASSIGNIO	75	HP-75 User's Library Solutions I/O Utilities
ASSIGN#	86	HP-86/87 Operating and BASIC Programming Manual
ASSIGN #	75	HP-75 Owner's Manual
ASSIGN #	9845	Operating and Programming Manual for the 9845
DIM	75	HP-75 Owner's Manual
DIM	86	HP-86/87 Operating and BASIC Programming Manual
DIM	9845	Operating and Programming Manual for the 9845
DISP	75	HP-75 Owner's Manual
DISP	86	HP-86/87 Operating and BASIC Programming Manual
DISP	9845	Operating and Programming Manual for the 9845
ENTER	86	HP-86/87 Operating and BASIC Programming Manual
ENTER	9845	Operating and Programming Manual for the 9845
ENTIOS	75	HP-75 User's Library Solutions I/O Utilities

Command/Statement	Computer	Reference
FOR-NEXT	75	HP-75 Owner's Manual
FOR-NEXT	86	HP-86/87 Operating and BASIC Programming Manual
FOR-NEXT	9845	Operating and Programming Manual for the 9845
INTEGER	75	HP-75 Owner's Manual
INTEGER	86	HP-86/87 Operating and BASIC Programming Manual
INTEGER	9845	Operating and Programming Manual for the 9845
LAD#	75	HP-75 User's Library Solutions I/O Utilities
OUTPUT	86	HP-86/87 Operating and BASIC Programming Manual
OUTPUT	9845	Operating and Programming Manual for the 9845
PRINT#	86	HP-86/87 Operating and BASIC Programming Manual
PRINT #	75	HP-75 Owner's Manual
PRINT #	9845	Operating and Programming Manual for the 9845
READ#	86	HP-86/87 Operating and BASIC Programming Manual
READ #	75	HP-75 Owner's Manual
READ #	9845	Operating and Programming Manual for the 9845
SDA	75	HP-75 User's Library Solutions I/O Utilities
SENDIO	75	HP-75 User's Library Solutions I/O Utilities

Command/Statement	Computer	Reference
SHORT	75	HP-75 Owner's Manual
SHORT	86	HP-86/87 Operating and BASIC Programming Manual
SHORT	9845	Operating and Programming Manual for the 9845
STR\$	75	HP-75 Owner's Manual
TAD#	75	HP-75 User's Library Solutions I/O Utilities
VAL	75	HP-75 Owner's Manual
VAL	86	HP-86/87 Operating and BASIC Programming Manual
VAL	9845	Operating and Programming Manual for the 9845
VAL\$	86	HP-86/87 Operating and BASIC Programming Manual
VAL\$	9845	Operating and Programming Manual for the 9845



## APPENDIX B

### GLOSSARY

bit: A bit is a binary digit, the smallest part of a binary character that contains intelligible information.

buffer: A buffer is a location in memory, or a block of memory locations used for temporary storage of information.

characters: Characters are common typewriter symbols.

data file: A data file is a block of mass-storage memory where data can be permanently stored.

file: A file is a block of mass-storage memory where binary information, such as data or a program, can be stored for later use.

frame: A frame is a grouping of information into 11-bit sections. The first 3 bits identify the type of frame (that is, command or data), and the last 8 bits are the actual content of the frame.

interface: An interface is a common boundary between two systems, or two parts of a system, through which information is conveyed.

listener: A listener is a device which has been addressed to receive data or instructions from another device. A listener can also be called a "receiver."

prompt: A prompt is a symbol that appears at the left edge of the computer display that indicates the computer is ready for the next keyboard statement or command.

source: A source is a device transmitting data or commands to other devices in the system.

talker: A talker is a device that has been addressed to send data or instructions to other devices in the system. A talker can also be called a "sender."

## REFERENCES

### Manuals for the HP-75

"HP-75 Owner's Manual," Hewlett-Packard Corvallis Division, Corvallis, Oregon, 1982.

"HP-75 User's Library Solutions I/O Utilities," part no. 00075-90122.

### Manuals for the HP-86

"HP-86/87 Operating and BASIC Programming Manual," Hewlett-Packard Personal Computer Division, Corvallis, Oregon, 1982.

"HP 82169A HP-IL/HP-IB Interface Owner's Manual," Hewlett-Packard Corvallis Division, Corvallis, Oregon, 1983.

"HP 82938A HP-IL Interface Owner's Manual--Series 80," Hewlett-Packard Personal Computer Division, Corvallis, Oregon, 1982.

"I/O ROM Owner's Manual--Series 80," Hewlett-Packard Personal Computer Division, Corvallis, Oregon, 1983.

### Manuals for the HP-9845

"BASIC Language Interfacing Concepts," Hewlett-Packard Desktop Computer Division, Ft. Collins, Colorado, 1981.

"Operating and Programming Manual for the 9845," Hewlett-Packard Desktop Computer Division, Ft. Collins, Colorado, 1981.

"Tutorial Description of the Hewlett-Packard Interface Bus," Hewlett-Packard Corp., 1980.

### Textbook

Cassell, Douglas A., "Microcomputers and Modern Control Engineering," Reston Publishing Co. Inc., Reston, Virginia, 1983.

DA  
FILM