HEWLETT PACKARD

PCInstruments

System Owner's Guide

for IBM PC, PC/XT and PC/AT
Personal Computers

Using the HP 61061A System Interface

HP Part No. 61061-90001
Microfiche No. 61061-90002

Printed in U.S.A.
April, 1985

Reader Comment Sheet

PC Instruments System Owner's Guide (61061-90001)

We welcome your evaluation of this manual. Your comments and suggestions will help us to serve you better by improving our publications. Please explain your answers and identify specific page numbers under comments, below.

■ Is this manual technically accurate?	Yes	No
■ Are the concepts and wording easy to understand?	Yes	No
■ Are instructions complete?	Yes	No
■ Is the organization logical?	Yes	No
■ Is it convenient in size and readability?	Yes	No

Comments: ______________________________

____

____

____

____

____

____

Name: ______________________________

Title: ______________________________

Company: ______________________________

Address: ______________________________

City: ______________________________

State: ______________ Zip: ______________

Please tear out and mail in.

HEWLETT PACKARD

Printed in U.S.A. 4/84

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 63 ROCKAWAY, N.J.

POSTAGE WILL BE PAID BY ADDRESSEE

Hewlett-Packard Company
Attn: Publications Dept.
150 Green Pond Rd.
Rockaway, N.J. 07866

Fold here

Tape Please do not staple Tape

Printing History

New editions of this manual will incorporate all material since the previous edition. Update packages, which may be issued between editions, contain replacement and additional pages to be merged into the manual by the user.

The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.)

The interface card prefix number adjacent to the date refers to the first part of the serial number located on the solder side of the card. This number indicates the version of the card that was available at the time that this manual was issued. Similarly, the software revision letters below the prefix number indicate the versions of each disc that were available at time of issue.

However, some changes to the hardware or software product do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between hardware/software changes and manual updates.

Edition 1.... April 1985....Interface Card Prefix 2511A
...System Master Diskette Rev. A
...System Utilties Diskette Rev. A

Notice

Safety Summary

The following safety precautions must be observed during all phases of installation and operation of your HP PC Instruments System. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of this system. Hewlett-Packard assumes no liability for the customer's failure to comply with these requirements. The PC Instrument Owner's Guides include additional safety statements that apply to specific instruments.

Check the Input Voltage

Make sure that the input voltage required by each instrument's Power Pack corresponds to the voltage source at the ac outlet.

Ground the System

To avoid potentially hazardous electrical shock, establish a safety ground before connecting user's circuits. To ground the computer, connect its line cord to the ac line. To ground each instrument, connect the output cable from its Power Pack to the instrument, and then connect the line cord from the Power Pack to the ac line. Detailed instructions are in Chapter 2 of this guide.

Ensure Equipment Status

Instruments in the HP PC Instruments System are under user's manual or program control. Equipment failure, power failure, or program error may result in a hazardous situation. Any application requiring a fail-safe method of ensuring equipment status must be provided by the installer. This includes devices such as interlocks, thermostats, limit switches, or overpressure or overspeed sensors.

Safety Symbols

WARNING

The WARNING sign calls attention to a procedure, practice, or the like, which, if not correctly performed or adhered to could result in personal injury. Do not proceed beyond a WARNING sign until the indicated conditions are fully understood and met.

CAUTION

The CAUTION sign calls attention to an operating procedure, or the like, which, if not correctly performed or adhered to could result in damage to or destruction of part or all of the product or any equipment connected to it. Do not proceed beyond a CAUTION sign until the indicated conditions are fully understood and met.

Contents

Chapter 5 Programmed Instrument Control

Introduction

About this Guide

This is a system-level guide for the HP PC Instruments and is the first document that you should read after opening your PC Instruments package.

Relation to Other Documents

Figure 1 is a documentation map that shows how PC Instruments documentation relates to some of the documentation that you received with your personal computer. Although the computer is shipped with comprehensive software documentation, the documents shown in the map are those most important for using PC Instruments.

This guide covers PC Instruments system hardware and software. Except for instructional examples, this guide does not cover detailed operation of any PC Instruments. Each PC Instrument module is shipped with an Instrument Owner's Guide that provides complete operating information for that instrument. You must consult both this guide and the appropriate Instrument Owner's Guide in order to operate an instrument.

How this Guide is Organized

Table 1 gives a synopsis of what you will find in each chapter of this guide. Refer also to ''Contents'' (at the front of this guide) for a more detailed list of topics.

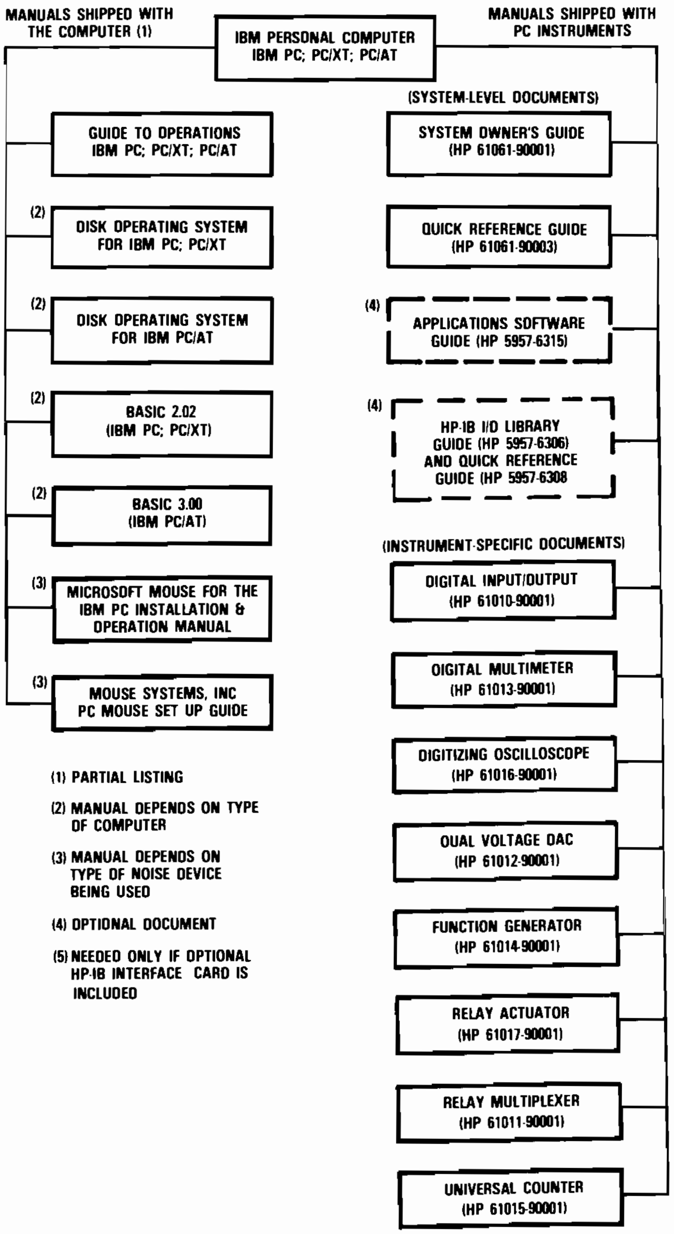

MANUALS SHIPPED WITH THE COMPUTER (1)
IBM PERSONAL COMPUTER IBM PC; PC/XT; PC/AT
MANUALS SHIPPED WITH PC INSTRUMENTS
GUIDE TO OPERATIONS IBM PC; PC/XT; PC/AT
(2) DISK OPERATING SYSTEM FOR IBM PC; PC/XT
(2) DISK OPERATING SYSTEM FOR IBM PC/AT
(2) BASIC 2.02 (IBM PC; PC/XT)
(2) BASIC 3.00 (IBM PC/AT)
(3) MICROSOFT MOUSE FOR THE IBM PC INSTALLATION & OPERATION MANUAL
(3) MOUSE SYSTEMS, INC PC MOUSE SET UP GUIDE
(SYSTEM-LEVEL DOCUMENTS)
SYSTEM OWNER'S GUIDE (HP 61061-90001)
QUICK REFERENCE GUIDE (HP 61061-90003)
(4) APPLICATIONS SOFTWARE GUIDE (HP 5957-6315)
(4) HP-IB I/O LIBRARY GUIDE (HP 5957-6306) AND QUICK REFERENCE GUIDE (HP 5957-6308
(INSTRUMENT-SPECIFIC DOCUMENTS)
DIGITAL INPUT/OUTPUT (HP 61010-90001)
DIGITAL MULTIMETER (HP 61013-90001)
DIGITIZING OSCILLOSCOPE (HP 61016-90001)
DUAL VOLTAGE DAC (HP 61012-90001)
FUNCTION GENERATOR (HP 61014-90001)
RELAY ACTUATOR (HP 61017-90001)
RELAY MULTIPLEXER (HP 61011-90001)
UNIVERSAL COUNTER (HP 61015-90001)
(1) PARTIAL LISTING
(2) MANUAL DEPENDS ON TYPE OF COMPUTER
(3) MANUAL DEPENDS ON TYPE OF NOISE DEVICE BEING USED
(4) OPTIONAL DOCUMENT
(5) NEEDED ONLY IF OPTIONAL HP-IB INTERFACE CARD IS INCLUDED

Figure 1. PC Instruments Documentation Map

Table 1. A Synopsis of this Guide

Introduction

- How this guide relates to other PC Instruments documents.
- What's in this guide and how to use it.
- What you are assumed to know.

Chapter 1 - Fundamental System Concepts

- A brief description of, and specifications for, the PC Instruments system.

Chapter 2 - Installing Your System

- Checking your shipment for damage or missing items.
- Installing the PC Instruments Interface Card.
- Connecting the instrument modules to the system.

Chapter 3 - Getting Started

- Making your PC Instruments work diskettes.
- Configuring your PC Instruments System.
- Getting the default system configuration.
- Loading PC Instruments software and trying it out.
- What to do in case of trouble.

Chapter 4 - Manual Instrument Control

- What the Soft Front Panel is and what it can do.
- A step-by-step example of manual instrument control.
- The system menus explained in detail.

Chapter 5 - Programmed Instrument Control

- An outline of how to develop an application program.
- A simple example of an application program.
- An explanation of PC Instruments system programming statements.
- More details on how to develop an application program.
- All about system error handling.
- How to measure the performance of your application program.

Table 1. A Synopsis of this Guide (cont.)

Chapter 6 - Stimulus/Response Testing

- A complete example of a stimulus/response test application.
- Techniques for stimulus/response testing.
- Adding the optional PC Instruments graphics utility to your application program.

Chapter 7 - Using PC Instruments with Other Applications Software

- How to make use of some non-PC Instruments data handling software.

Appendices

- A synopsis of related computer information.
- Detailed PC Instruments system verification procedures.
- PC Instruments system error messages.

Glossary

- Explains many terms used in this manual.

Supplement

- Using HP-IB Instruments with PC Instruments. How to install and use the optional HP-IB I/O Library software.

PC Instruments Application Newsletter

This guide includes a reply card that entitles you to a free subscription to the PC Instruments Application Newsletter. The newsletter has valuable information on all aspects of PC Instruments, including hardware, software, and training. Be sure to reutrn the reply card promptly.

How to Begin

How you approach this guide depends on the type of user you are. Table 2 shows the general reading requirements for four typical types of users. Table 3 is a more detailed step-by-step recommended reading sequence for a first-time user with a new system. That table includes other applicable documents.

NOTE

If you would like a synopsis of how to get your PC Instruments system operational, turn to "QuickStart Guide" (Table 3-1 in Chapter 3)

Table 2. Recommended Reading for Typical PC Instruments Users

	Chapter							Appendix			
Type of User	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**A**	**B**	**C**	**Glossary**
First-time user with a new system	x	x	x	x	x	x		x	x	x	x
First-time user with an existing system	x		x	x	x	x			x	x	x
*Experienced user with a new system		x	x	x	x	x	x			x	
*Experienced user adding to an existing system		x	x			x	x			x	

*Refer also to *PC Instruments Quick Reference Guide.*

Table 3. Recommended Reading Sequence for First-Time Users (1)

Procedure	Guide*	Chapters
Install the personal computer.	1	As req'd
Learn to operate computer; become familiar with DOS	1,2,3	As req'd
Become familiar with the basics of the PC Instruments System.	4	1
Install the PC Instruments Interface Card and instrument module(s).	4	2
Get Started -		
Install the PC Instruments Software	4	3
Configure the PC Instruments System (2).	4	3
Load the PC Instruments Soft Front Panel.	4	3
Try out your instruments (3) -		
Familiarize yourself with your instrument by operating it from the Soft Front Panel.	5	2
Exit from the Soft Front Panel.	5	2

Table 3. Recommended Reading Sequence for First-Time Users(1) (Continued)

Procedure	Guide*	Chapters
Learn the details of manual-mode operation -		
Follow an example of Soft Front Panel control.	4	4
Control your instrument from the Soft Front Panel.	5	3
Learn programmed-mode operation -		
Follow development of a BASIC program that performs the same functions as the manual-mode example.	4	5
Write a sample program that controls your specific Instrument.	5 6	5
Follow more sophisticated examples of programming and manipulation of the resultant data.	4 6	6,7
Connect the instrument(s) to your application (4)	5	4

Notes: (1) Other users may skip steps, as appropriate.
(2) Need be done only once, except that configuration must be repeated if PC Instruments hardware is changed.
(3) Specifically designed for the first-time user.
(4) You may also want to learn about the programmed mode before connecting your application.

*** Guides (See Figure 1):**
(1) Guide to Operations for your specific computer.
(2) Disk Operating System Manual for your specific computer.
(3) Instruction manual for your mouse device.
(4) PC Instruments System Owner's Guide (this manual).
(5) PC Instrument Owner's Guide (for each instrument).
(6) BASIC manual for your specific computer.

What You Should Already Know

The HP PC Instruments system software runs on your IBM personal computer under the DOS operating system. Your computer Guide to Operations explains the basics of operating the keyboard and the Disk Operating System manual fully describes how to use the operating system. The manual supplied with the ''mouse'' accessory explains how to use that device (Appendix A of this manual summarizes some important things you must know). Before proceeding further, please be sure you understand the following things concerning your computer:

Using DOS. The IBM personal computer uses the PC-DOS (or just DOS) operating system, which is an IBM version of the Microsoft Operating System (MS-DOS). DOS is fully documented in the IBM Disk Operating System manual. Familiarity with DOS will make the HP PC Instruments System operation that much easier for you.

Making a Selection. You can point to and select an active area on the screen either with a mouse device or via the computer keyboard. Appendix A has a summary of typical mouse device operation. PC Instruments allows full use of the mouse device. If you do not have a mouse, you can point to the active field by operating the cursor control keys and then selecting the file with the Enter [↵] key.

Use of Softkeys. PC Instruments has softkeys on the screen that can be selected with the mouse. The first eight keyboard function keys duplicate the functions of these screen softkeys.

Diskette Handling Care. The diskettes supplied with your system can be damaged by improper handling or storage. *Keep diskettes away from strong magnetic fields, such as those around the PC Instrument module Power Packs.* Diskette operating life is long, but not unlimited. Some diskettes may have a write-protect lable to prevent accidental erasure. Details are given in your computer documentation.

Backing Up Diskettes. You should not use the master diskettes supplied to you as regular working diskettes. This manual recommends how to make "work diskettes" for actual use.

BASIC. The programmed control mode of PC Instruments requires you to write programs in BASIC. PC Instruments has features that greatly simplify this for you, but you must know the fundamental material in the BASIC instruction manual supplied for your computer.

NOTE

BASICA is an advanced Microsoft BASIC that operates on your PC under DOS. Unless otherwise specified, the word "BASIC" means BASICA in this manual.

---

1 Fundamental System Concepts

NOTE

If you have the optional demonstration diskette ''Discovering HP PC Instruments'', this is a good time to load and run it. The diskette covers much of the material given in this section. You do not need any extra hardware or software to use that diskette. Just follow the directions supplied on it.

What is HP PC Instruments?

HP PC Instruments is a combination of hardware and software that allows you to use your personal computer as an integrated instrumentation system (see Figure 1-1). PC Instruments consists of:

- Hardware (individual instrument modules, such as the Digital Multimeter and the Function Generator) that performs instrumentation functions. You can mix and match any combination of PC Instruments modules.
- One or more PC Instruments Interface Cards that connect the instrument modules to your personal computer via a ribbon-cable bus.
- Soft Front Panel and instrument control programs that allow you to manually operate your PC Instruments from your personal computer.
- A library of powerful instrument programming statements that help you create programs for automatically controlling your PC Instruments in the programmed mode.

- An optional library of routines to allow you to write programs to control instruments that are not PC Instruments modules, but are equipped with the HP-IB (IEEE-488) Interface (refer to the HP-IB documentation).
- Optional applications software (in BASIC) that run combinations of PC Instruments in the programmed mode to solve specific measurement problems, such as data acquisition.
- Optional software packages that perform mathematical and/or scientific/engineering graphical operations on the data gathered by the PC Instruments system.

The HP PC Instruments Soft Front Panel program and BASIC programming are mutually exclusive software applications. Therefore, at any given time your system operates in either the manual (Soft Front Panel) or the programmed (BASIC) mode.

The Manual Mode of PC Instruments System

Figure 1-2 shows the basic concept of PC Instruments manual-mode operation. With conventional instruments, you interact with each one via its controls and indicators. With the PC Instruments System, the instruments are replaced by modules that perform the instrumentation function but do not have operating controls or indicators.

Your computer screen becomes the means by which you monitor and operate your instruments. This is made possible by the PC Instruments Soft Front Panel program that controls each instrument and organizes the screen into four fixed areas:

System View Window - This window allows you to select the instrument you wish to operate. When you do, a front panel view of that instrument appears in the Interactive Instrument Window. The System View Window also gives you a short summary of every instrument in the system.

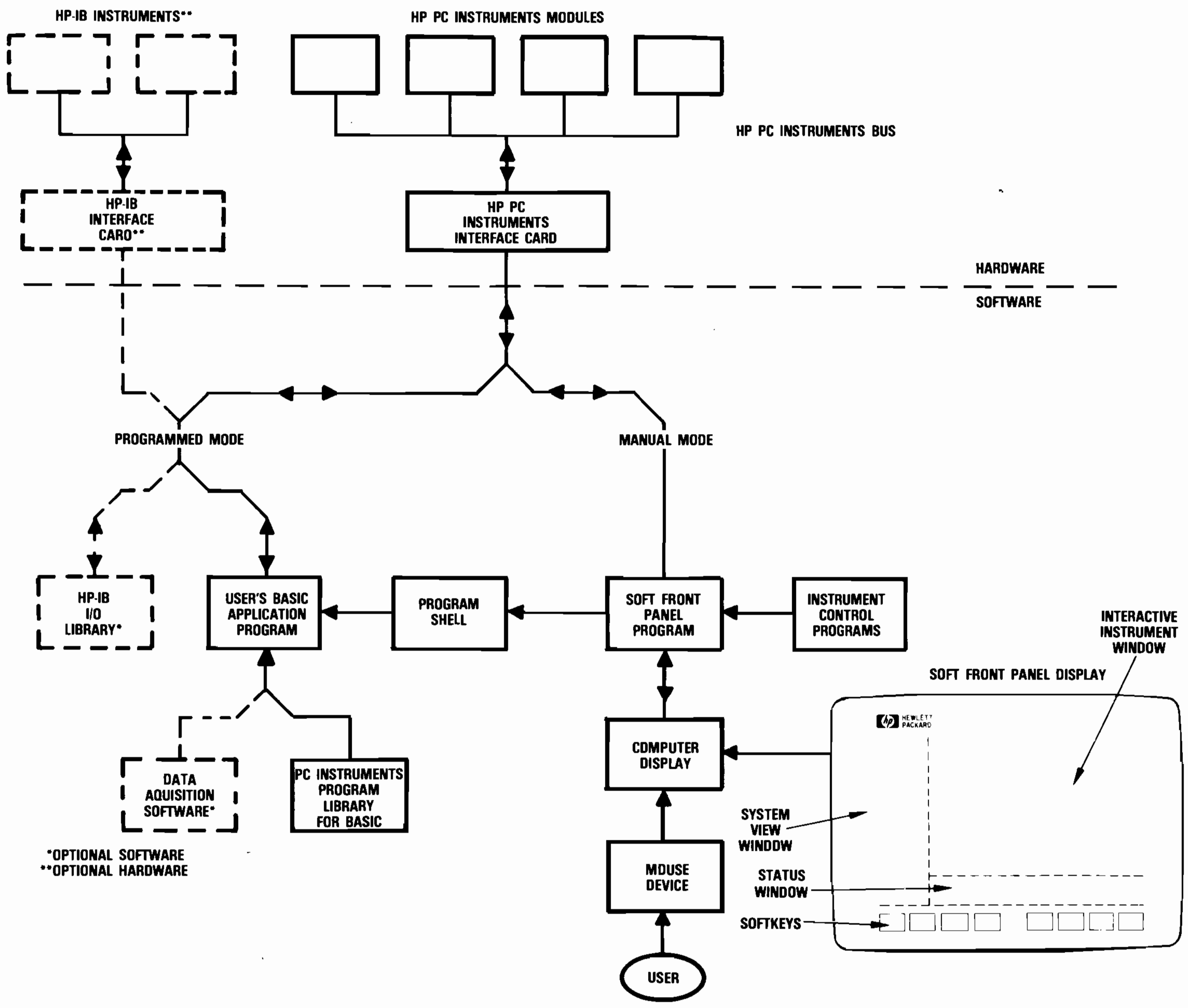
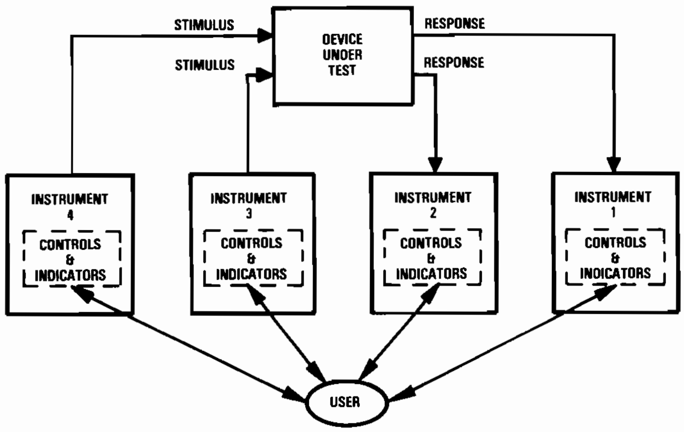
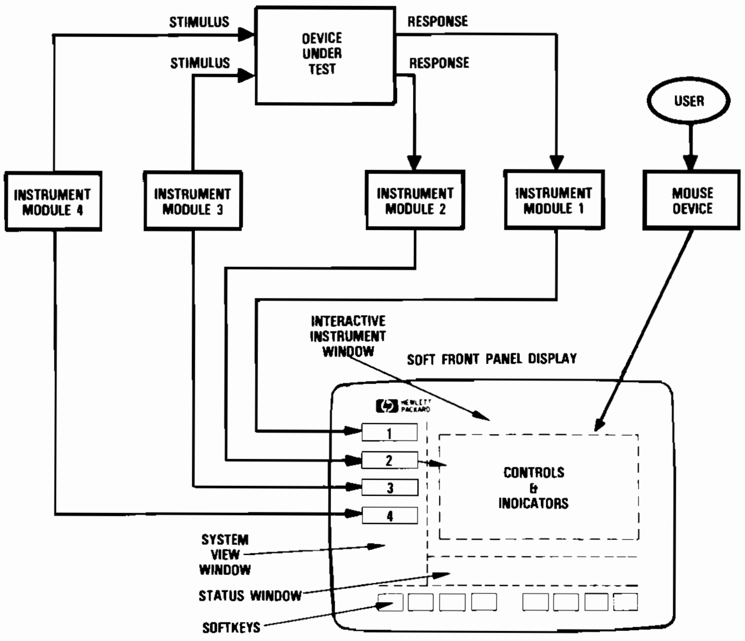


Figure 1-1. Functional Diagram of the HP PC Instruments System



a) Conventional Manual-Mode Measurement System



b) HP PC Instruments Manual-Mode Measurement System

Figure 1-2. Concept of the HP PC Instruments System

Interactive Instrument Window - This shows the operable instrument (the one you are controlling manually) in detail. The window has graphic representations of familiar controls and indicators. By means of the computer, you can interact with the instrument just as if an actual hardware panel were there in front of you.

Status Window - This is a message area that gives you feedback (in the form of prompts and error messages) about both the instrument you are operating and the system. The window also provides a place for you to make entries for file storage and retrieval.

Softkeys - These comprise a multi-level menu that simplifies setting up your system and helps you manipulate system files without having to return to DOS. The menus also allow you to quickly disable all output instruments in an emergency.

The Programmed Mode of PC Instruments System Operation

To help you create application programs, the HP PC Instruments software has a library of instructions, or ''statements'', that perform specific instrument functions. Most of these statements are generic and apply to all the PC Instruments in the same way. You combine these statements in your own BASIC program to produce instrument control programs. These programs may be as simple, or as complex, as the measurment task demands and your knowledge of BASIC permits.

Executable Applications Software

There are optional BASIC software packages available that eliminate the need for you to write programs for the more common applications. If your system has the required instrument modules, you simply enter BASIC from DOS and run the program. You can obtain a list of available applications programs from your local HP Sales Office.

Graphing Scientific/ Engineering Results

The PC Instruments software allows you several options for graphing the results of data acquired by its programmed mode of operation. For example, the optional Data Acquisition Software has a graphics utility that you can merge with your instrument control program. Or, you can use the Convert Utility (included with the system software) to change a BASIC data file into an ASCII or DIF format suitable for use with Lotus 1-2-3, NWA STATPAK, or other third-party software packages.

System Specifications

Table 1-1 gives some technical data about your HP PC Instruments System. Please note the ''Computer Configuration Requirements.'' Your computer must meet these requirements in order for the PC Instruments System to operate properly. You will find the specifications for each instrument in its owner's guide.

Table 1-1. HP PC Instruments System Specifications

Computer Configuration Requirements

Supported Computers: IBM PC; PC/XT; PC/AT

Minimum Required RAM: 640 K Bytes Total

IBM PC - 256 K internal plus one 384 K Quadram Quadboard.

IBM PC/XT - 256 K internal plus one 64/256 K Memory Expansion Card with 256 K loaded and one 64/256 K Memory Expansion Card with 128 K loaded.

IBM PC/AT - 512 K internal plus one 128 K Memory Expansion Card. (The basic AT model requires an additional 256 K of internal memory to bring the total internal to 512 K).

Table 1-1. HP PC Instruments System Specifications (Cont.)

Accessory Slots Required:

IBM PC - 3 (1 each for PC Instruments Interface Card, 384 Quadram Quadboard, and Color/Grahpics Monitor Adapter).

IBM PC/XT - 4 (1 for PC Instruments Interface Card, 2 for 64/256 K Memory Expansion Cards, and 1 or Color/Graphics Monitor Adapter).

IBM PC/AT - 3 (1 for PC Instruments Interface Card, 1 for 128 K Memory Expansion Card, and 1 for Color/Graphics Monitor Adapter. The basic Model AT requires an additional slot for the Serial/Parallel Adapter needed for the mouse device).

General Hardware Specifications

PC Instruments Interface

Maximum Number of PC Instruments Interface Cards:
Determined by number of available accessory card slots

Maximum Number of Instruments per Interface Card: 8

****Instruments Supported:**

HP 61010A Digital I/O
HP 61011A Relay Multiplexer
HP 61012A Dual Voltage DAC
HP 61013A Digital Multimeter
HP 61014A Function Generator
HP 61015A Universal Counter
HP 61016A Digitizing Oscilloscope
HP 61017A Relay Actuator

** For instrument specifications, refer to applicable Instrument Owner's Guide.

Table 1-1. HP PC Instruments System Specifications (Cont.)

Diskette Drives: 2 - Double-sided diskette (standard)
or
1 - Double-sided diskette (high capacity)
or
1 - Fixed disk + 1 diskette (either standard or high capacity).

Monitor: 1 - IBM Color Monitor or Princeton Graphics System HXC2, or Sanyo Dm8112CX (monochrome).

Graphics Adapter: 1 - IBM Color Graphics Monitor Adapter (required for all monitors).

Mouse Device: 1 - Microsoft Mouse* or 1 Mouse Systems Model 1B-1.
*Requires special adapter cable when used with the IBM-PC/AT.

Compatible Peripherals:

Printer - IBM Graphics Printer

Plotter - HP 7470A Two-Pen Plotter
or
HP 7475A Six-Pen Plotter

Software Media: 5¼ inch double-sided diskettes (standard capacity; may be copied to high-capacity diskettes)

Operating System: DOS 3.00

Required Software: BASICA 2.0 (included with DOS Master Diskettes)

Optional Software: HP 14856A Data Acquisition Software
HP 61062A HPIB I/O Package (includes diskette, interface card & cable, and documentation)
HP 14851-10001 PC Instruments Demo Diskette

Table 1.1 HP PC Instruments System Specifications (Cont.)

General Hardware Specifications (Cont.)

Cables:

PC IB Bus Cable - 1 supplied (1.8 m or 5.9 ft)

Adjacent Instrument Interconnect Cable - 1 supplied

Instrument Interconnect Cable - 1 supplied with each instrument

General Software Specifications

Manual Instrument Control:

- Up to 16 PC Instrument modules supported in System View Window
- System View Window update rate is 3/sec (typical)
- 1 user-definable label per instrument
- Any combination of PC Instrument modules supported
- Multiple state files supported

Programmed Instrument Control:

- Up to 16 PC Instrument modules supported in any combination
- Allows user-defined labels from state files
- Allows BASIC simple variable or array variables
- Error handling compatible with BASICA
- Optional Data Acquisition Software utility calls may be used within the same user application program
- Optional HP-IB I/O Library calls may be used within the same user application program

Table 1-1. HP PC Instruments System Specifications (Cont.)

PC Instruments Files:

System Software Master (61061-10001)

(Root Directory)

AUTOEXEC.BAT
AUTOEXEC.HD
CONFIG.SYS
LOADPCIB.BAT
MKDRPCIB.BAT
MAKEPCIB.BAT

PCIB<DIR>

HPERR.HPE
PANELS.EXE
PCIBAS.BAT
PCIB.PLD
PCIBILC.BLD
PCIBILC.EXE
PCIBILC.HPE
PCIB.SYN
SFPMSG.HPE

Utilities & Verification (61061-10002)

(Root Directory)

CONFIGUR.BAS
CONFIGUR.BAT
CONVERT.EXE
CONVERT.HPE
SYS_VER.BAT

PCIB_VER <DIR>

VERIFY.BAS
ACUTATOR.BAS
COUNTER.BAS
DIGIO.BAS
DMM.BAS
FUNCCAL.BAS
FUNCGEN.BAS
RELAYMUX.BAS
SCOPE.BAS
VDAC.BAS

2 Installing Your System

Inspecting Your Shipment

It is important that you inspect your HP PC Instrument shipment promptly and report any missing items.

Your PC Instruments equipment is divided into two major groups. First there is the system-level equipment contained in the interface product package. This hardware and software is identified in Table 2-1 and Figure 2-1. Optional software items that may be included, if purchased, are listed in Table 2-2. The second group of equipment consists of the PC Instrument module packages. A typical instrument module package is identified in Figure 2-2 and Table 2-3. The types and quantity of instrument packages should agree with those specified in the packing list. The inventory for each instrument package appears in Chapter 2 of its owners's guide.

Table 2-1. Standard PC Instruments System Components

Qty	Description	HP Part No.
	Hardware	
1	PC Instruments Interface Card	61061-60022
1	PCIB Interface Cable, 1.8 metre (5.9 ft.)	8120-4632
1	Adjacent Instrument Interconnect Cable, 0.35 metre (13.8 in.)	8120-4633
	Software	
2	Diskette holders containing:	
1	System Software Master Diskette	61061-10001
1	System Utilities & Verification Disc	61061-10002
2	Blank unformatted diskette (double-sided)	9164-0175
1	System Owner's Guide package containing:	
1	Manual binder & slip case	5080-2064
1	System Owner's Guide page packet	61061-90001
1	Quick Reference Guide	61061-90003
1	PC Instruments Support Guide	5957-6317
1	PC Instruments software update package (Included as required)	

Table 2-2. Optional System Items

Qty.	Description	HP Part No.
	Software	
1	Demo Disc	14851-10001
1	Data Acquisition Software Package	14856A
1	HP-IB Interface and I/O Library	61062A
1	Box of 10 blank, double-sided diskettes	92190A
1	Extra slipcase and binder	5080-2064
	Hardware	
1	System Power Unit	61001A
1	PC Instruments Rack Shelf	14801A

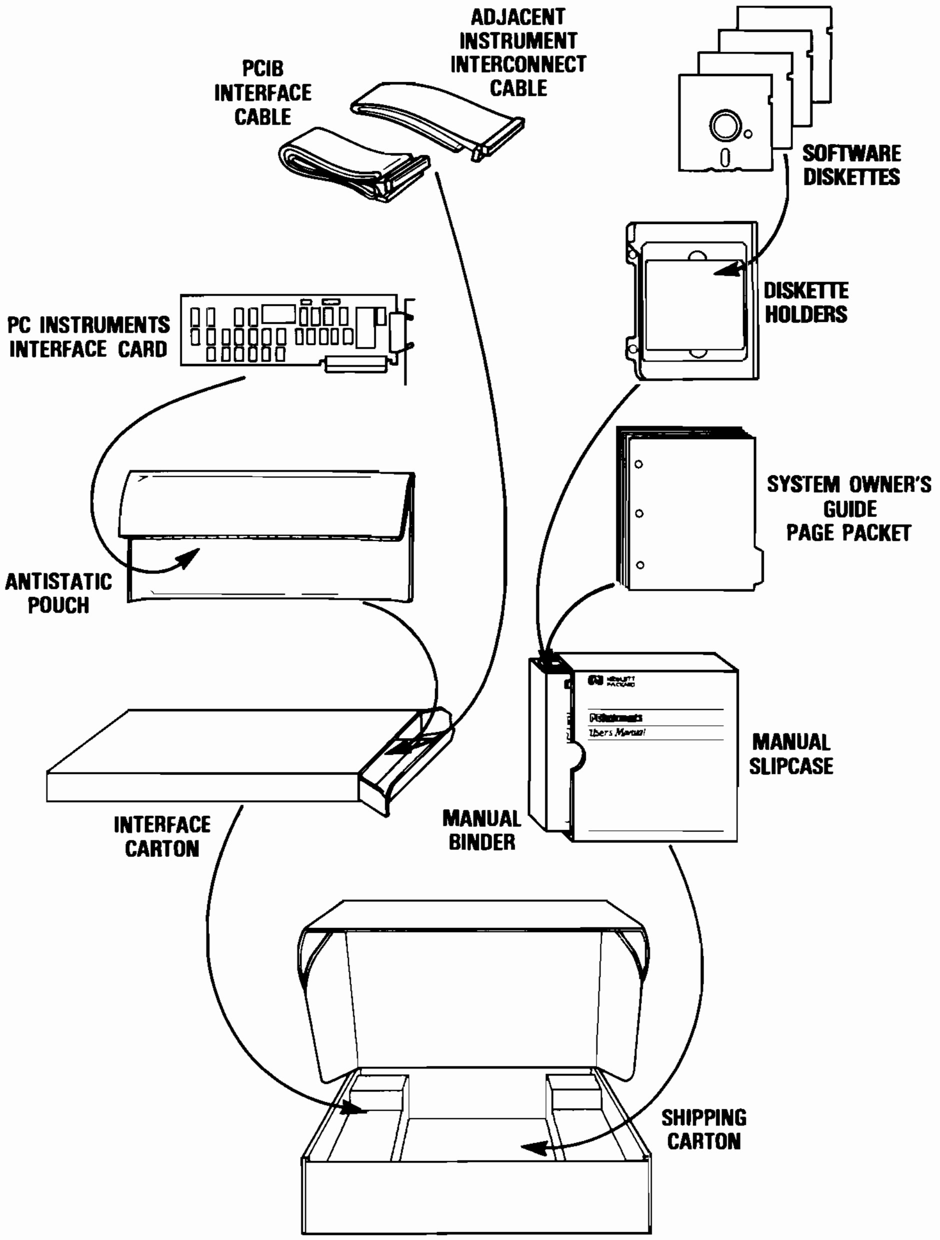
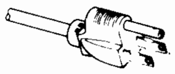
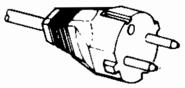
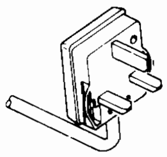
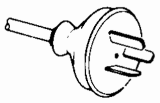
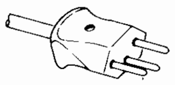


Figure 2-1. Basic HP PC Instruments System Components

Table 2-3. Typical HP PC Instrument Module Package *

Qty.	Description	HP Part No.
1	Instrument Module	*
1	Power Pack (Depends on Mains Power)	
	100 Vac (International)	5080-2056
	120 Vac (U.S.A./Canada)	5080-2057
	220 Vac (International)	5080-2058
	240 Vac (International)	5080-2059
1	Line Cord (Depends on Country)	
		8120-1378
		8120-1689
		8120-1351
		8120-1369
		8120-2104
1	Instrument Interconnect Cable, 0.11 metre (4.3 in)	*
1	Accessories (if applicable)	*
1	Instrument Owner's Guide page packet	*

* Consult Chapter 2 of the applicable Instrument Owner's Guide.

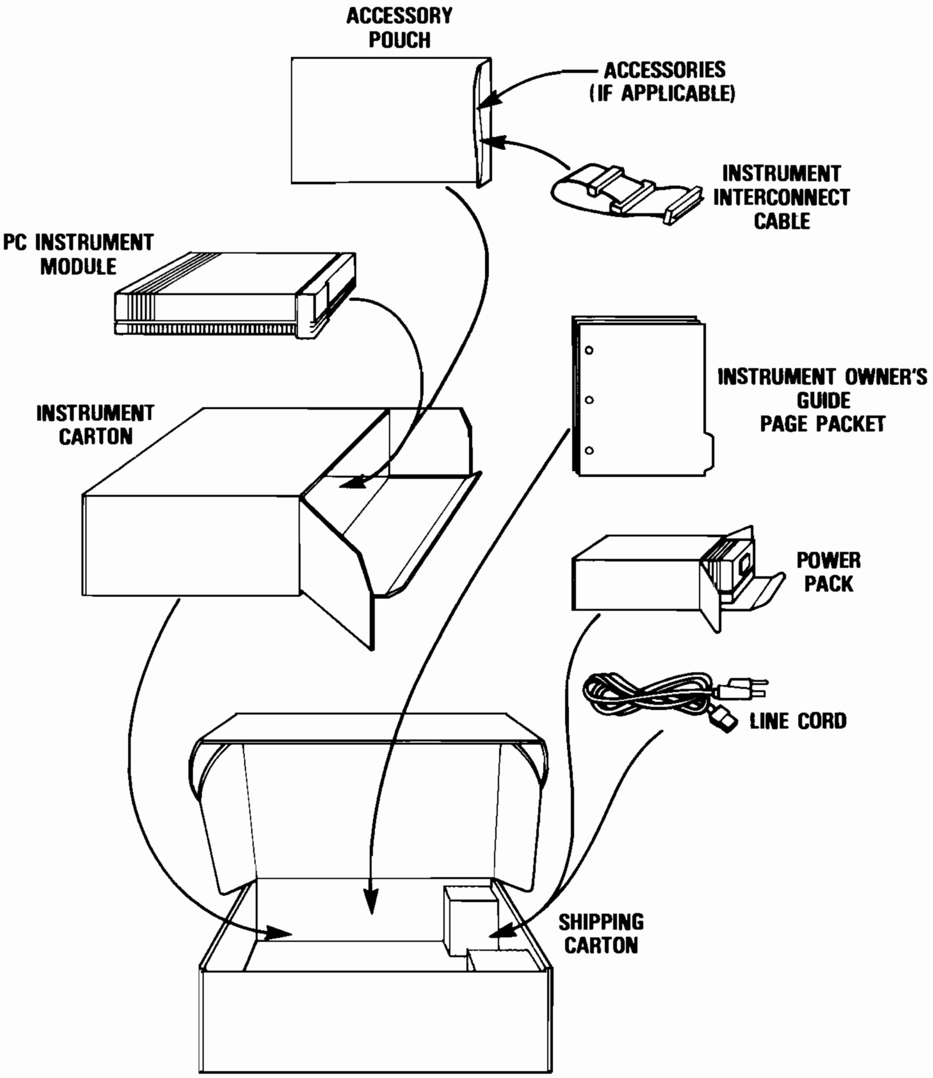

ACCESSORY POUCH
ACCESSORIES (IF APPLICABLE)
INSTRUMENT INTERCONNECT CABLE
PC INSTRUMENT MODULE
INSTRUMENT CARTON
INSTRUMENT OWNER'S GUIDE PAGE PACKET
POWER PACK
LINE CORD
SHIPPING CARTON

Figure 2-2. Basic HP PC Instrument Package Components

Is Something Missing

If you are missing one or more HP PC Instruments items, contact the place where you purchased them. This could be either your local Hewlett-Packard Sales Office or your dealer.

Returning HP PC Instruments Items

You may have to return an item because it was damaged during shipment or because it requires servicing. Before returning an item, consult the *HP PC Instruments Support Guide* for warranty and service policy information. The guide will identify the HP Service Center nearest you. The Customer Repair Card, located in the rear of each manual, *must* be filled out and returned with any item that you return. Be sure to pack the item in its original (or equivalent) packaging material. This is important because an item damaged in reshipment may not be accepted under terms of the warranty.

Connecting the Hardware

Hardware installation consists of installing the PC Instruments Interface Card into your computer, connecting the interface to your instrument module, and connecting each module to your application.

The Operating Environment

The hardware and software components of your HP PC Instruments System will operate in any environment that is suitable for the computer as described in your computer documentation (refer also to Table 1-1 of this manual). In general, the best environment is within the 20°C to 30°C (68° F to 86° F) temperature range at 40% to 60% relative humidity. Further environmental restrictions for instrument modules are as follows:

- Leave at least 25 mm (1 inch) clearance on each side of an instrument module.
- Leave at least 50 mm (2 inches) clearance behind an instrument module.

• Do not place anything on top of an instrument module that could obstruct its convection cooling.

NOTE

The PC Instruments modules may be stacked one on top of another because they are designed for proper cooling in this configuration (see Figure 2-8).

Installing the PC Instruments Interface Card

CAUTION

The HP PC Instruments Interface Card has solid state components that can be damaged by static electricity. Leave the card in its protective package until you are ready to install it and then handle it only by its edges. Drain off any static charges on your body by touching an exposed metal part of the computer chassis (such as a screw) before you unpack the interface card.

Where to Install the PC Instruments Interface Card. The IBM personal computer has accessory card ''slots'' located inside the system unit (the unit with the diskette drives). If your system includes an expansion unit, that has slots for additional accessory cards. You may install the interface card in any empty slot that is available; the physical position of the slot does not matter. The computer ''knows'' which slot that card is in because of an address switch on the card. This switch must be set to a unique address that is not used by any other accessory card.

CAUTION

If the address of the PC Instruments Interface Card is already being used by another accessory card in your computer, your program will either not run at all, or else continue with unpredictable results.

Address Switch Settings. The address switch on the PC Instruments Interface Card is set at the factory to the address reserved for the IBM Prototype Card, which is hexadecimal 300 (0300H), as shown in Figure 2-3. This address was selected because it is most likely to be unused. The only time it should be necessary for you to change this address is:

- If your computer already has an IBM Prototype Card.
- You are installing more than one PC Instruments Interface Card.
- Another accessory card is already using the 0300H addresses.

If any of these conditions apply to your installation, refer to Appendix A for instructions on changing the address of the PC Instruments Interface Card.

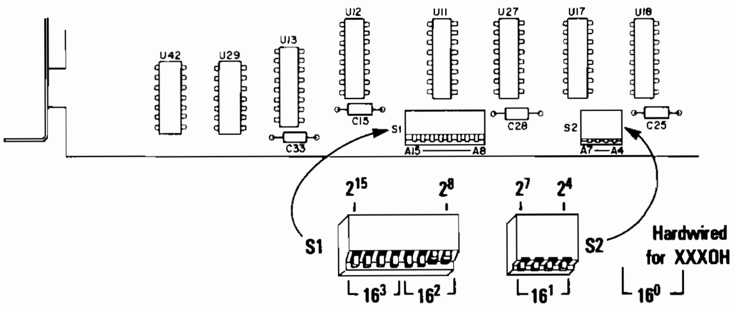


Figure 2-3. Example of PC Instruments Interface Card

Installing the Card. You can install the interface card in any empty expansion slot in the computer or expansion unit. The procedure is the same in either case. The only tool you will need is a medium-sized, flat-bladed screwdriver. If available, a pair of nutdrivers (1/4-inch and 3/16-inch) can make the work easier. These instructions will vary slightly with the model of IBM PC being used. In case of difficulty, refer to the options installation section of your computer manual.

WARNING

This procedure requires removal of the rear cover. Hazardous voltages are inside the unit. Turn off all power switches in your system and disconnect the line cord from the computer (and the expansion unit).

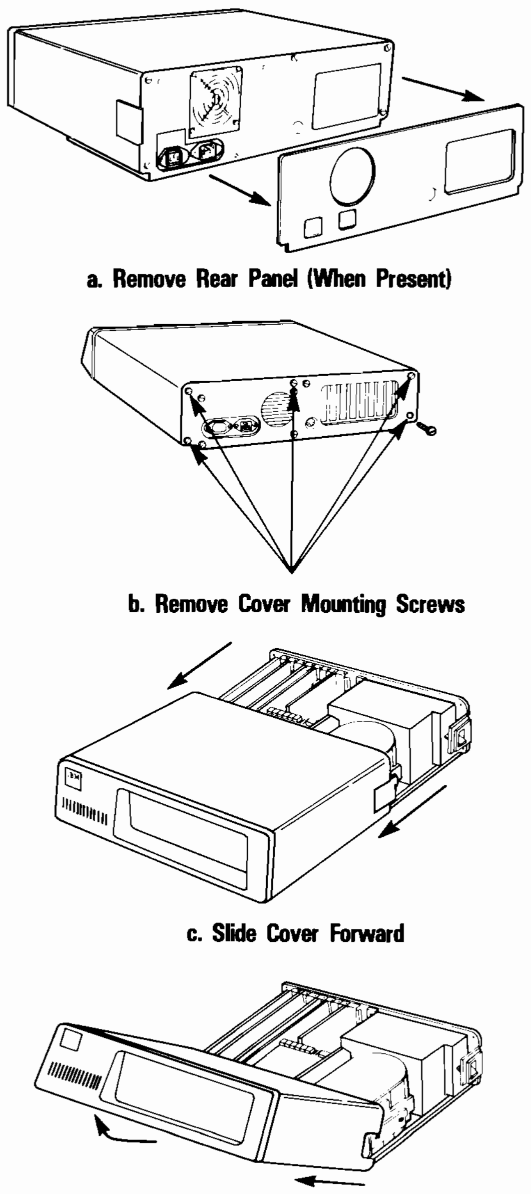
---

1. See Figure 2-4b. Using the screwdriver or the 1/4-inch nutdriver, remove the five screws holding the rear cover. If the unit has a rear panel covering the screws, first remove the panel by pulling it free of the plastic hook and loop fasteners (Figure 2-4a).

2. If the unit has a front panel key lock, turn the key to the ''off'' position and remove it from the lock. Carefully remove the rear cover by sliding it forward (Figure 2-4c). With some models, you will also have to tilt the cover slightly upward (Figure 2-4d).

3. Locate an unused expansion slot inside the rear of the unit. You may install the HP PC Instruments Interface Card in any unused slot (see Figure 2-5).

4. Using the screwdriver or the 3/16-inch nutdriver, remove the cover for the expansion slot (see Figure 2-5a). Save the cover in case you must operate the unit without this interface card at some future time.



a. Remove Rear Panel (When Present)

b. Remove Cover Mounting Screws

c. Slide Cover Forward

d. Tilt Up (Where Required)

Figure 2-4. Removing Computer Cover

5. Before unpacking the interface card, drain off any static charge on your body by touching a ground, such as an exposed screw on the metal part of the computer chassis.

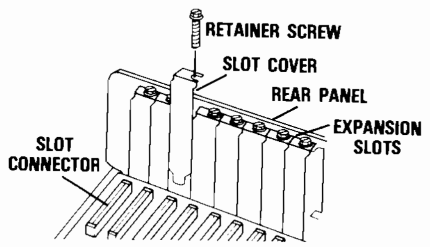
6. Carefully unpack the interface card and *hold it only by its edges* with the components facing up.

7. Compare the address switch settings on the card to make sure they agree with the default settings (see Figure 2-3). If it is necessary to change the switch settings to a different address refer to Appendix A.

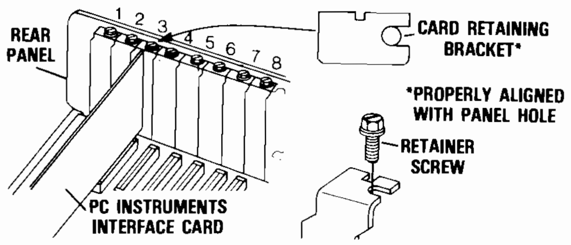
8. Carefully slip the connector end of the card through the rear slot opening and slide the card into the card guides. Be sure that the card retaining bracket hole lines up with the hole in the rear panel (see Figure 2-5b).

9. Using the screw removed in step 4, fasten the card retaining bracket to the rear panel.

10. Carefully replace the rear cover by reversing the procedure you used to remove it. Fasten it to the frame with the five screws removed in step 1 (see Figure 2-6). If the unit has a rear panel, press it in place to secure the hook and loop fasteners.



a. Removing the Expansion Slot Cover


REAR PANEL
1 2 3 4 5 6 7 8
CARD RETAINING BRACKET*
*PROPERLY ALIGNED WITH PANEL HOLE
RETAINER SCREW
PC INSTRUMENTS INTERFACE CARD

b. Installing the PC Instruments Interface Card

Figure 2-5. Installing PC Instruments Interface Card in Expansion Slot

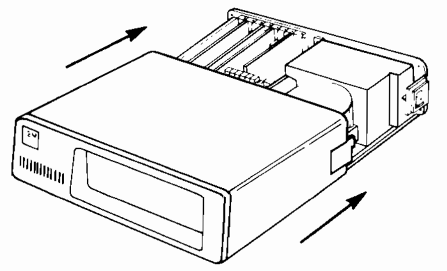
Installing the Instrument Modules

You must complete installation of the hardware by connecting each instrument module to the system, to a power source, and to your application. The first thing to do is make sure that each instrument module's address switch is correctly set.

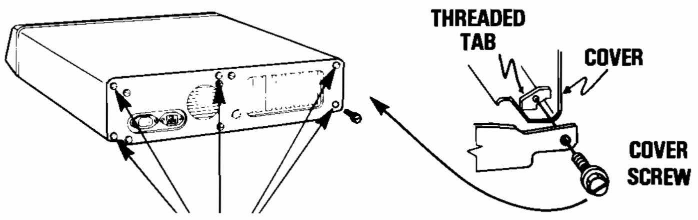
CAUTION

It is important that each instrument module in your system be set to a unique address. Failure to to do this can result in damage to the equipment.

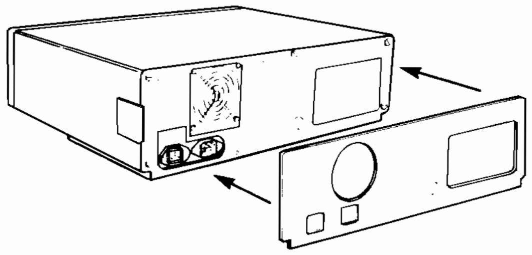
Setting the Instrument Module Address Switch. Before you connect the PC Instruments Interface Card to the instrument modules, you must set the address switch on each of the instrument modules. Since up to eight instruments can be connected to an interface card, you



a. Slide Cover Over Unit



b. Install Cover Mounting Screws



c. Replace Rear Panel (If Unit Has One)

Figure 2-6. Replacing the Unit Cover

have addresses 0 through 7 available. The address switch is located on the rear of each instrument (see Figure 2-7). When an instrument is shipped from the factory, its address switch is set to zero (as shown in Figure 2-7). To change the address setting, insert a small screwdriver into the slotted switch shaft and rotate the arrow to the desired number, as shown in Figure 2-8.

An instrument module's address determines where it will be displayed in the System View Window of the Soft

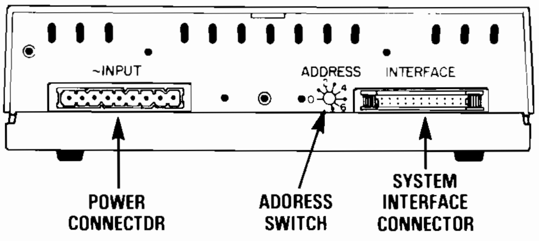


Figure 2-7. Rear View of a Typical Instrument Module

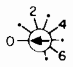
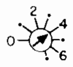
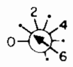
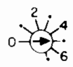
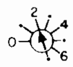
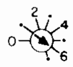
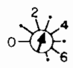
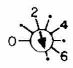
ADDRESS	SWITCH SETTING	ADDRESS	SWITCH SETTING
0		4	
1		5	
2		6	
3		7	

Figure 2-8. Instrument Address Switch Settings

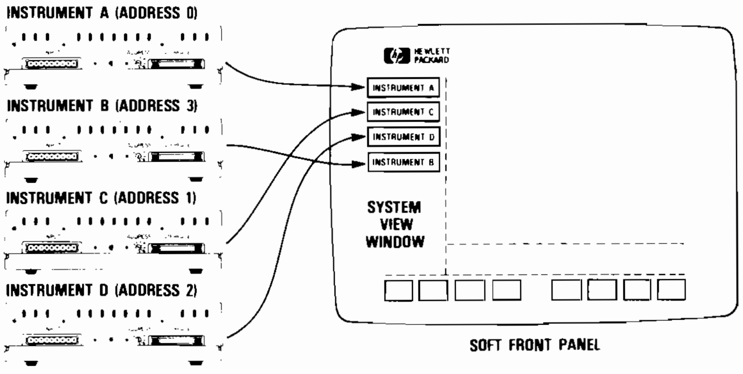


Figure 2-9. Instrument Address and System View Window Order

Front Panel (see Figure 2-9). The instrument with the lowest address is displayed first, followed by the next highest, and so on.

Connecting Instruments to the System. Once its address switch is properly set, an instrument module can be connected to the system. Three ribbon cables (see Figure 2-10) are provided for this purpose. The cable connectors are keyed to help you install them correctly. Start by connecting the PC Instruments PCIB Cable (1) to the PC Instruments Interface Card in the computer expansion slot. Spread the latches on the cover plate of the interface card, and insert the straight connector of the interface cable into the socket. The cable will automatically latch when it is fully inserted into the socket. After that, the connections depend on the configuration of your system.

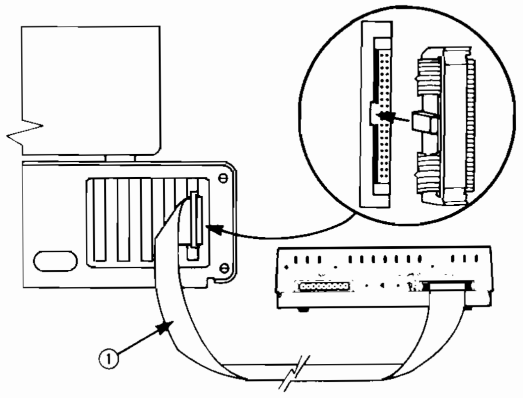
CAUTION

Always install the cables so that there are no unused connectors. Be certain to observe proper key alignment. Failure to do so may result in damage to the equipment.

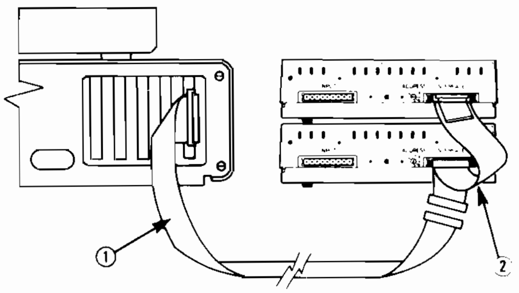
Note that the cable position of an instrument has nothing to do with its position in the System View Window of the Soft Front Panel. This is determined by the setting of the instrument's address switch, as shown in Figure 2-9.

Figure 2-10a shows the simplest connection, that between a single instrument and the PC Instruments Interface Card. Connect the other end of the PCIB Interface Cable (1) to the interface connector on the rear of the instrument module. You do not need the Instrument Interconnect Cable (2) supplied with the instrument module.

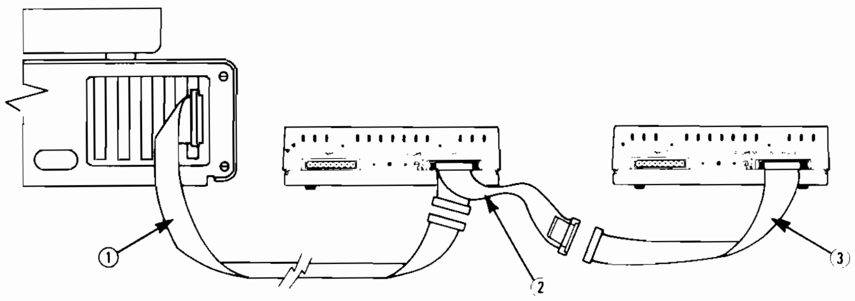
Figure 2-10b shows a simple stacked instrument configuration. Connect the first instrument to the computer and to the instrument above it with one Instrument Interconnect Cable (2). Note that only one such



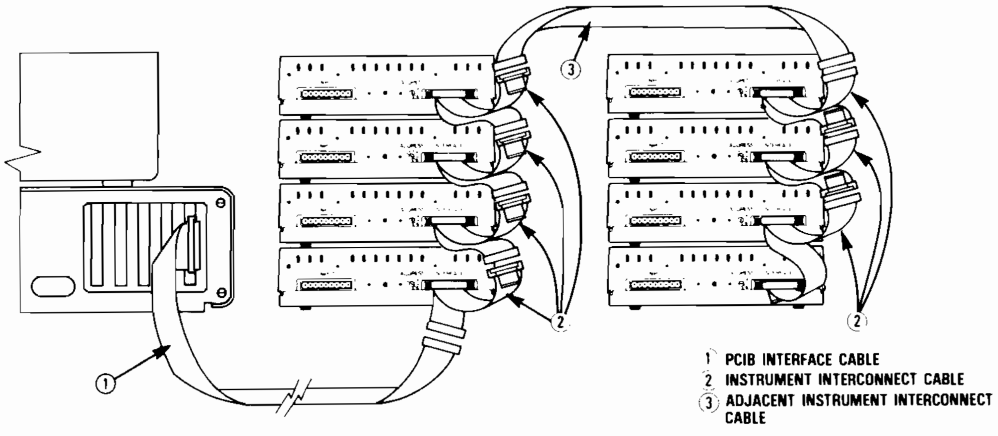
a. Single Instrument



b. Two Instrument Stacked



c. Two Instruments Adjacent



1
2
3
1 PCIB INTERFACE CABLE
2 INSTRUMENT INTERCONNECT CABLE
3 ADJACENT INSTRUMENT INTERCONNECT CABLE

d. Eight Instruments in Two Adjacent Stacks

Figure 2-10. Examples of HP PC Instruments System Configurations

cable is needed for the two instruments.

Figure 2-10c shows a side-by-side installation using two instruments. Here your must use the Adjacent Interconnect Cable (3) between the two instrument modules. You can expand this layout into two stacks (see Figure 2-10d) by using Instrument Interconnect Cables (2). Note that one such cable is needed for each instrument, except at the second stack, where one cable connects two instruments. The Adjacent Interconnect Cable (3) may connect the stacks at either the top or the bottom.

Connecting Power to the Instruments. Each instrument module is powered from a power pack that is shipped with it. Depending on your location, each instrument will have one of the power packs listed in Table 2-3.

CAUTION

Make sure that the input voltage required by your Power Pack corresponds to the voltage source at the a-c outlet. Never place a diskette near a Power Pack.

Figure 2-11 shows a typical power connection. First, connect the female end of the line cord to the power pack. Next, connect the ribbon cable to the power connector on your instrument module. Finally, connect the female end of the line cord to the a-c outlet. When you do this last step, the instruments front panel "ON" indicator will light.

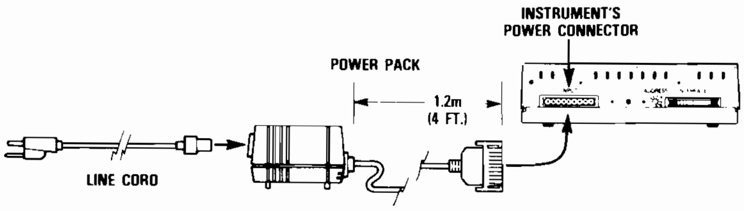


Figure 2-11. Typical Instrument Module Power Connection

Connecting Instruments to Your Application

WARNING

Instruments in the PC Instruments System are under the user's manual or program control. Equipment failure, power failure, or program error may result in a hazardous situation. Any application requiring a failsafe method of ensuring equipment status must be provided by the installer. This includes devices such as interlocks, thermostats, limit switches, or overpressure or overspeed sensors.

Your installation is now complete except for the connections between each instrument module and your application. These connections depend on the type of instrument module and how it is being used. Refer to Chapter 4 of the applicable *Instrument Owner's Guide(s)* for details.

CAUTION

Do not connect the instruments to your application until you have gone through Chapters 3 and 4 of this manual and have read the applicable ***Instrument Owner's Guide.*** *In order to prevent damage to either your equipment or the instrument modules, you must first have an understanding of how the two will interact when they are connected together.*

3 Getting Started

Introduction

Table 3-1 is a synopsis of what you must do to get your PC Instruments system ''up and running'' in the manual control mode. The rest of this section gives the details for loading the software and trying out your system.

Table 3-1. Quick-Start Guide

Step	Action	Refer to Page
1.	Install the PC Instruments Interface Card	2-7
2	Connect the instrument module(s) to the system	2-15
3	Install your PC Instrument System software	3-2
4	Configure the PC Instruments System*	3-16
5	Obtain the default PC Instruments configuration**	3-20
6	Run the PC Instruments Soft Front Panel program	3-21
7	Try out your PC Instruments System (refer also to Chapter 2 of the appropriate Instrument Owner's Guide.	3-23

* You must do this only after the initial hardware installation, or if you should change the address of a PC Instruments Interface Card.

** You may need to do this only if you are learning to use a system that has already been operated before.

The PC Instruments System software is shipped on two double-sided master diskettes that are identified in Table 1-1. You will also need the master diskette for your mouse device and may need the master diskette containing your DOS and IBM Advanced BASIC files.

NOTE

Throughout this guide, the keyboard Enter key [↵] is identified as Enter .

Installing Your PC Instruments System Software

The following procedures show how to combine your software onto one or two work diskettes that you will use for running your PC Instruments System software. The PC Instruments System software has two "batch file" commands (MKDRPCIB and LOADPCIB) to simplify the procedure. You also use the DOS system COPY and DISKCOPY commands (refer to your *Disk Operating System User's Guide*).

Making a Work Disk or Diskette

In order to both protect your master diskettes and to simplify the use of the software, you will construct a "work disk" or "work diskette" by copying files from master diskettes. The actual procedure depends on the type of diskette drives that you have.

NOTE

Typical IBM Personal Computer drive configurations are shown in this chapter.

Fixed-Disk System. Figure 3-1 shows some typical fixed-disk systems. You will put all your necessary files on the fixed disk (see Figure 3-2). It is assumed that you have formatted the fixed disk using the [/S] (copy system) option so that the disk is already "bootable". If you have not done this and cannot reformat the disk, then you will have to boot your computer from a DOS Master diskette. However, you will still want to take advantage of its capacity to make the fixed disk your work disk.

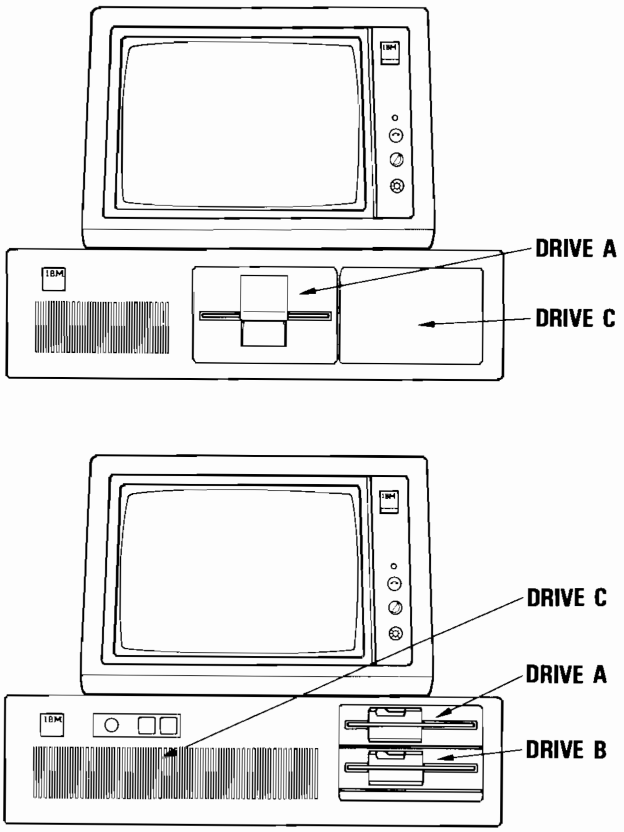


Figure 3-1. Typical Fixed – Disk Systems

1. It is assumed that you have all your DOS Master diskette files already on the fixed disk. This should include advanced BASIC (BASICA). If you do, go to to Step 2. Otherwise, insert your DOS System Master diskette in drive A and type:

```
C> COPY A:\BASICA.COM C:\
```
   and press Enter

2. Insert your mouse device master diskette in drive A. If you have the Microsoft device, type:

```
C>COPY A:\MOUSE.SYS C:\
```
   and press Enter

   If you have the Mouse Systems device, type:

```
C>COPY A:\MSMOUSE.SYS C:\MOUSE.SYS
```
   and press Enter

3. Insert your PC Instruments Software Master diskette in drive A. Create subdirectory "C:\PCIB" on your fixed disk by typing:

```
C> A:
```
   and pressing Enter

```
A> MKDRPCIB C:
```
   and pressing Enter

CAUTION

The next step overwrites existing DOS files. Proceed carefully.

---

4. Copy the PC Instruments files by typing:

```
A>LOADPCIB A: C:
```
   and pressing Enter

   LOADPCIB is a PC Instruments batch command that copies the AUTOEXEC.BAT, CONFIG.SYS, AND CONVERT files under the root directory and the PC

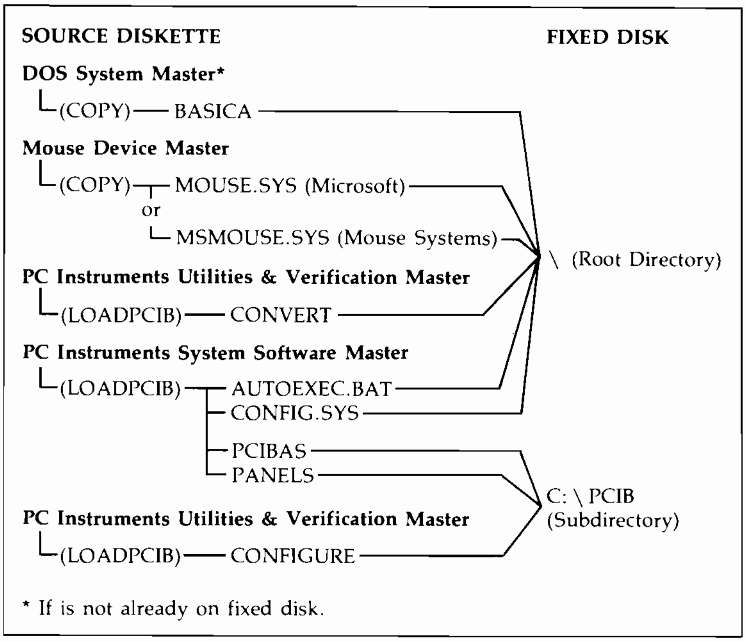

SOURCE DISKETTE
FIXED DISK
DOS System Master*
(COPY) BASICA
Mouse Device Master
(COPY) MOUSE.SYS (Microsoft)
or
MSMOUSE.SYS (Mouse Systems)
\ (Root Directory)
PC Instruments Utilities & Verification Master
(LOADPCIB) CONVERT
PC Instruments System Software Master
(LOADPCIB) AUTOEXEC.BAT
CONFIG.SYS
PCIBAS
PANELS
C: \ PCIB
(Subdirectory)
PC Instruments Utilities & Verification Master
(LOADPCIB) CONFIGURE
* If is not already on fixed disk.

Figure 3-2. Structure of Work Disk as Part of the Fixed Disk

Instruments files (including CONFIGURE) under subdirectory \ PCIB. The command will warn you that any existing AUTOEXEC.BAT and CONFIG.SYS files on your fixed disk will be overwritten. If you do *not* want these files overwritten, follow the procedure given in Step 5. Otherwise, complete the LOADPCIB operation and go to to Step 6.

NOTE

Two of the files that you are copying (CONVERT and CONFIGURE) are on the PC Instruments Utilities and Verification Master Diskette. You will be prompted when to put that diskette into drive A.

---

5. The PC Instruments software includes AUTOEXEC.BAT and CONFIG.SYS files that will be written to your fixed disk. If you already have those files on your disk and they are customized for your system, then you will not want to overwrite them. First, save your files by renaming them:

a. If you want to keep the AUTOEXEC.BAT file on your fixed disk, type:

```
A> REN C:\AUTOEXEC.BAT AUTOEXEC.SAV
```

and press [Enter]

b. If you want to keep the CONFIG.SYS file on your fixed disk, type:

```
A>REN C:\CONFIG.SYS CONFIG.SAV
```

and press [Enter]

After renaming the file or files, complete the LOADPCIB operation. Then, use an editor to combine the new files with your original files. That is:

Load AUTOEXEC.BAT into your editor and merge AUTOEXEC.SAV

and/or

Load CONFIG.SYS into your editor and merge CONFIG.SAV

NOTE

If you are using a serial mouse device, check that it is connected to the computer's COM1 serial port. If it is connected to COM2, then you can either physically connect it to COM1 or you can further edit your CONFIG.SYS file. Change the line in the file from "device=mouse.sys/1" to "device=mouse.sys/2".

You will now have restored your original batch file(s) with the added information from the PC Instruments batch file(s).

6. If you have not already done so, make sure that your mouse device is connected to the computer's COM1 port. If it is not, then you must either physically switch it from COM2 to COM1 or edit the CONFIG.SYS file . To edit the file, change the line from ''device=mouse.sys/1'' to ''device=mouse.sys/2''.

7. Perform a reset [Alt] [Ctrl] [Del] to ''reboot'' the computer. This will load the PC Instruments configuration files and restore DOS operation to the default (C) drive.

Your fixed disk is now your PC Instruments work disk. Put all your master diskettes away in a safe place and proceed to ''Configuring the PC Instruments System.''

Standard Dual-Diskette System. Figure 3-3 shows a typical dual diskette system. You will use the two blank diskettes supplied with your PC Instruments System and copy the required software to them as shown in Figure 3-4. Work diskette 1 will be your ''bootable'' DOS diskette and work diskette 2 will have the most needed PC Instruments files.

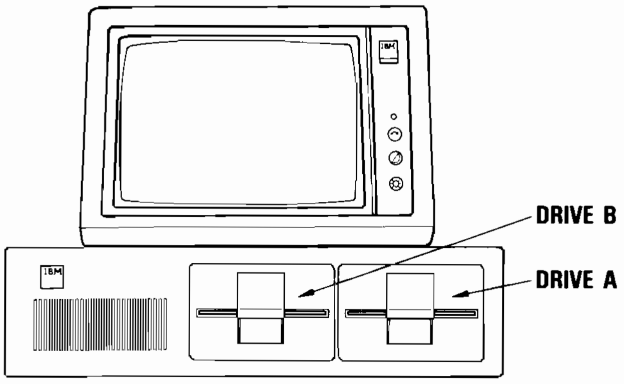


Figure 3-3. Typical Dual – Diskette System

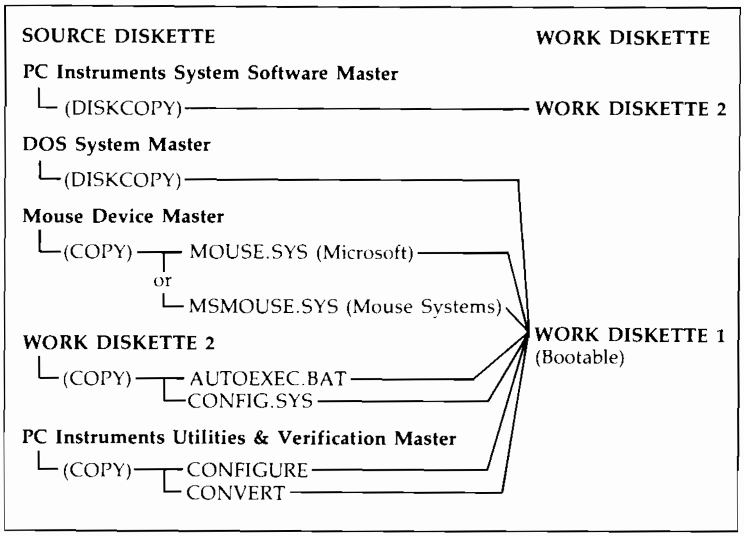


Figure 3-4. Structure of Work Diskettes for Dual – Diskette System.

1. Insert your DOS Master diskette in drive A and one of the blank, double-sided diskettes supplied with your PC Instruments System in drive B. Then type:

   A>DISKCOPY A: B:
   and press $\boxed{\text{ENTER}}$

2. Insert your mouse device master diskette in drive A:
   If you have the Microsoft mouse device, type:

   A>COPY A:\MOUSE.SYS B:\
   and press $\boxed{\text{Enter}}$

   If you have the Mouse Systems mouse device, type:

   A>COPY A:\MSMOUSE.SYS B:\MOUSE.SYS
   and press $\boxed{\text{Enter}}$

3. Load the DISKCOPY program into memory by typing:

   A>DISKCOPY A: B:
   and pressing $\boxed{\text{Enter}}$

4. In response to the ''source'' prompt, insert your PC Instruments Software Master diskette in drive A. In response to the ''target'' prompt, insert your second work diskette (the blank, unformatted one) in drive B.

5. Press any key to copy the PC Instruments Software to your second work diskette.

6. Insert the first (bootable) work diskette in drive A. Leave the second work diskette in drive B.

CAUTION

The next procedure overwrites two existing DOS files (AUTOEXEC.BAT and CONFIG.SYS). If you do ***not*** *want one or both of those files overwritten, perform Step 7. Otherwise, go to Step 8.*

---

7. If your existing AUTOEXEC.BAT and/or CONFIG.SYS files are customized for your application, then you will not want them overwritten by the files existing on the diskette in drive B. Therefore, temporarily save your files by renaming them:

   a. If you want to keep the AUTOEXEC.BAT file on your work diskette, type:
      ```
      A>REN A:\AUTOEXEC.BAT AUTOEXEC.SAV
      ```
      and press Enter

   b. If you want to keep the CONFIG.SYS file on your work diskette, type:
      ```
      A>REN A:\CONFIG.SYS CONFIG.SAV
      ```
      and press Enter

8. Copy the following two files from your work diskette in drive B to the root directory of your bootable work

diskette in drive A by typing:

```
A>COPY B:\AUTOEXEC.BAT A:\
```
and pressing Enter
```
A>COPY B:\CONFIG.SYS A:\
```
and pressing Enter

9. If you renamed a file or files as described in Step 7, perform Step 10. Otherwise, go to Step 11.

10. To restore your original AUTOEXEC.BAT and/or CONFIG.SYS file, use an editor to combine the new files (that you copied from the work diskette in drive B) with your original files that were renamed in Step 8. That is:

Load AUTOEXEC.BAT into your editor and merge AUTOEXEC.SAV

and/or

Load CONFIG.SYS into your editor and merge CONFIG.SAV

NOTE

*If you are using a serial mouse device, check that it is connected to the computer's COM1 serial port. If it is connected to COM2, then you can either physically connect it to COM1 or you can further edit your CONFIG.SYS file. Change the line in the file from "device=*mouse.sys/1" to "device=mouse.sys/2".

You will now have restored your original batch file(s) with the added information from the PC Instruments batch file(s).

11. Remove the PC Instruments Software Master diskette from drive B and insert the PC Instruments Utilities and Verification Master diskette. Copy the CONVERT and CONFIGURE files by typing:

```
A>COPY B:\CONVERT.* A:\
    and pressing [Enter]
A>COPY B:\CONFIGUR.* A:\
    and pressing [Enter]
```

12. If you have not already done so, make sure that your mouse device is connected to the computer's COM1 port. If it is not, then you must either physically switch it from COM2 to COM1 or edit the CONFIG.SYS file. To edit the file, change the line from ''device = mouse.sys/1'' to ''device = mouse.sys/2''.

13. This is an optional, but recommended, step. Remove the work diskette from drive A and type:

```
A> DISKCOPY B: A:
   and press [Enter]
```

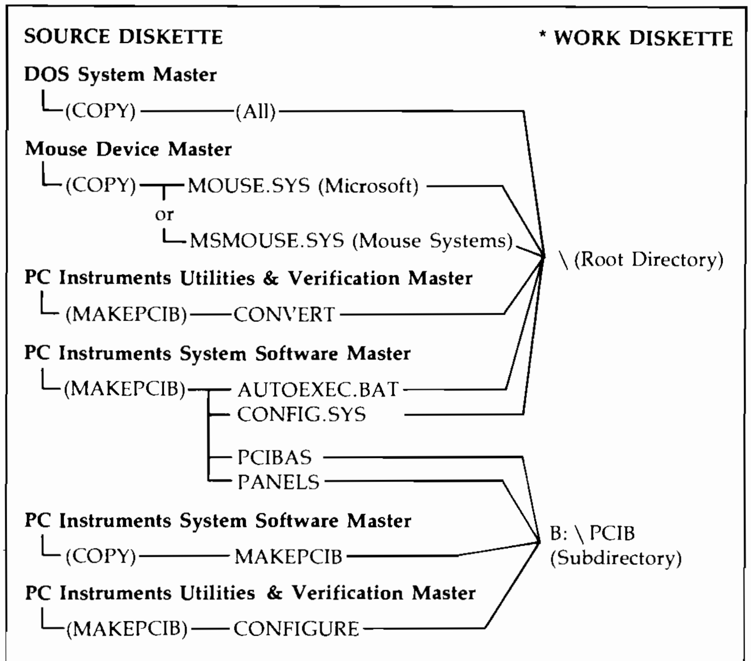
In response to the ''source'' and ''target'' prompts, insert a blank (formatted or unformatted) diskette in drive A. (Leave your PC Instruments Utilities and Verification Master diskette in drive B).

Press any key to make a backup copy of your Utilities and Verifcation diskette.

14. Insert your first (bootable) work diskette in drive A. Perform a reset [Alt] [Ctrl] [Del] to ''reboot'' the computer. This will load the PC Instruments configuration files and restore DOS operation to the default (A) drive.

You now have two PC Instruments work diskettes. Put all your master diskettes away in a safe place and proceed to ''Configuring the PC Instruments System.''

High-Capacity Single Diskette System. The procedure is for a typical high-capacity diskette system having one drive. You will put all your files on one diskette as shown



*Transfer must be made from source diskette to virtual disk and then to work diskette.

Figure 3-5. Structure of High Capacity Work Diskette

in Figure 3-5. (This is the same arrangement as is done with a fixed disk).

NOTE

Because there is only one diskette drive (A), the following procedure will require you to change diskettes in the drive. You will be prompted when it is time to change diskettes.

---

1. Insert your DOS Master Diskette in drive A and type:

```
A>FORMAT A:/S/V
```
and press [Enter]

2. At the prompt, insert a blank, unformatted diskette in the drive. Press any keyboard key to start the FORMAT program.

NOTE

*You **cannot use the blank diskettes supplied with your PC Instruments System**. Instead, use a high-capacity double-sided diskette designed for your drives.*

---

3. At the prompt, enter a suitable label to identify this as your bootable PC Instruments work diskette (such as "PCIB_BOOT").

4. After formatting the diskette, exit from the FORMAT routine.

NOTE

Because of the incompatability of the diskettes, you cannot copy the PC Instruments Software directly to the newly formatted high capacity diskette. The following procedure creates a virtual disk, copies the PC Instruments Software to that disk, and then copies the virtual disk to the high capacity diskette.

---

5. Leaving you newly formatted diskette in the drive, type:

```
A>COPY CON.CONFIG.SYS and press [Enter]
A>DEVICE=VDISK.SYS 400 512 and press [Enter]
A>  [Ctrl][Z] and press [Enter]
```

6. Reinsert the DOS Master Diskette. Type:

```
A>COPY VDISK.SYS B:
```

and press Enter

7. When prompted, reinsert the formatted high capacity diskette in the drive and press Enter .

8. When the copy operation is completed, perform a reset Alt Ctrl Del to reboot the computer. This will load the system from the high capacity diskette.

9. Insert your PC Instruments System Software Master diskette in the drive.

10. Copy the PC Instruments Software to the high capacity diskette via the virtual disk by typing:

A>C: and pressing Enter

C>COPY A:MAKEPCIB.BAT and pressing Enter

C>MAKEPCIB A: C: and pressing Enter

NOTE

Two of the files that you are copying (CONVERT and CONFIGURE) are on the PC Instruments Utilities and Verification Master Diskette. The copy operation requires that you insert four diskettes (the two PC Instruments masters, the DOS System Master, and the work diskette) in the drive. You will be prompted when to put a particular diskette into the drive.

---

11. Insert your mouse device master diskette in the drive. If you have the Microsoft device, type:

```
A>COPY A:\MOUSE.SYS B:\
```

and press Enter

If you have the Mouse Systems device, type:

```
A>COPY A:\MSMOUSE.SYS B:.MOUSE.SYS
```

and press Enter

NOTE

Change diskettes when prompted.

12. When the copy operation is completed, perform a reset Alt Ctrl Del to reboot the computer.

13. If you have not already done so, make sure that your mouse device is connected to the computer's COM1 port. If it is not, then you must either physically switch it from COM2 to COM1 or edit the CONFIG.SYS file. To edit the file, change the line from "device=mouse.sys/1" to "device=mouse.sys/2".

14. This is an optional step requried only if you must use special AUTOEXEC.BAT and/or CONFIG.SYS files. The PC Instruments Software loads these two files under the root directory of your work diskette. If you have your own customized version of either file, then:

    a. Use an editor to add your own information to the files that are now on your work diskette. You will then have your original batch file(s) with the added information from the PC Instruments batch file(s) on the work diskette.

    b. Perform a reset Alt Ctrl Del to reboot the computer.

You now have a bootable PC Instruments System work diskette. Put all your master diskettes away in a safe place and proceed to "Configuring the PC Instruments System."

Configuring the PC Instruments System

NOTE

The procedure given here must be done after PC Instruments hardware is first installed or if the address of a PC Instruments Interface Card is changed. If you are only learning and the system is already operational, skip this procedure and go on to ''Trying Out the PC Instruments System.''

Purpose

The system must be configured in order for PC Instruments software to ''know'' details of the hardware, such as the type of computer being used, the address of the PC Instruments Interface Card(s), etc. You only have to provide this information once for each work diskette, unless you subsequently change the hardware. PC Instruments will ''remember'' everything you enter here, even after the computer has been shut off. You perform the configuration from DOS, which we will assume is loaded in memory.

NOTE

The last menu question is only important if your available computer memory space is very limited and you do not want to allocate a full 64 K bytes of RAM to the PC Instrument data structures. Refer to Appendix A for details on reducing the amount of RAM that the system allocates for this purpose.

Procedure

1. If you have a fixed-disk system (with the PC Instruments software on it), type:

   ```
   C>CONFIGURE
   ```

   and press Enter

2. If you have a high capacity diskette system, insert your work diskette in drive A

and type:

```
A>CONFIGURE
```

and press [Enter]

3. If you have a standard dual-diskette system, insert your bootable work diskette (number 1) in drive B and your PC Instruments work diskette 2 (the one with your PC Instruments Software Master files) in drive A. Then, type:

```
A>CONFIGURE
```

and press [Enter]

4. The System Configuration menu will appear on the screen (see Figure 3-6).

5. Answer the menu questions by making the appropriate entries from the keyboard.

6. After you have answered the last question, you will be automatically returned to the operating system (DOS). The information you entered will be stored in the system configuration file (\ PCIB \ HPPI.HPH), which is stored on either your fixed disk or your PC Instruments work diskette.

ENTER COMPUTER TYPE? (Default is 1)

1-IBM PC
2-IBM PC/XT
3-IBM PC/AT

(Specifies the type of IBM computer you are using. Press the `Enter` key to automatically select the IBM PC)

ENTER KEYBOARD VERSION? (Default is 1)

1-United States

(Specifies the language version of the computer keyboard. Refer to Table 3-2 for options. Press `Enter` to specify the default).

ARE YOU USING A COLOR MONITOR? (Y/N)
(Default in N)

(Specifies the computer monitor you are using. If you have a black and white monitor press `Enter` for the default. If you respond with ''Y'', then you will be prompted to specify the color in accordance with Table 3-3).

ENTER NUMBER OF PC INSTRUMENTS INTERFACE CARDS?
(Default is 1)

(Specifies the number of interface cards. Press `Enter` to specify the default).

ENTER PC INSTRUMENTS INTERFACE ADDRESS?
(Default is 0300)

(Specifies the hexadecimal address of the PC Instruments Interface Card. Press `Enter` to specify the default. If you have more cards, you will be prompted to enter an address for each card. The address is a four-digit number in which the last (least significant) digit must be zero. Your entry is assumed to be a hexadecimal number).

ENTER HP-IB OPTION (Y/N)?
(Default is N)

(Enter ''Y'' if you have the optional HP-IB I/O Software Library. Otherwise, press `Enter` for the default)

ENTER MEMORY SPACE ALLOCATED FOR PC INSTRUMENTS DATA STRUCTURES?
(Default is 64 K)

(Specifies how much memory to set aside for the PC Instrument data structures. Press `Enter` for the default. In rare cases, this amount of memory may have to be reduced. Refer Appendix A for details.)

Figure 3-6. How to Answer the Configure Menu

Table 3-2. Configuration Codes for Computer Keyboards

Keyboard	Version
United States	1*
(Not used)	2
(Not used)	3
(Not used)	4
Danish	5
(Not used)	6
Finnish	7
Flemish	8
French	9
German	10
Italian	11
Norwegian	12
Spanish (European)	13
(Not used)	14
Swedish	15
(Not used)	16
(Not used)	17
United Kingdom	18

*The default entry.

Table 3-3. Configuration Codes for Color Monitors

Screen Color	Version	Screen Color	Version
Black	1	Gray	9
Blue	2	Light blue	10
Green	3	Light green	11
Cyan	4	Light cyan	12
Red	5	Light red	13
Magenta	6	Light magenta	14
Brown	7	Yellow	15
White	8	High-intensity white	16

Loading PC Instruments

You are now ready to load your PC instruments software into the computer memory.

Obtaining the Default System State

NOTE

This procedure is required only if you are a first time user on a PC Instruments System that has previously been used by someone else. If your system has never been operated before, go to ''Entering the Soft Front Panel.''

What is a State File?. Each PC Instrument module has certain operating parameters, such as its name (label), function setting, range setting, etc. When it is shipped from the factory, each instrument's parameters are in a default state determined by its hardware. When an instrument is operated from the Soft Front Panel, its parameters can be changed by the operator. When the Soft Front Panel is exited, the changed parameters are automatically stored in a ''state'' file (in the current directory) called ''HPSTATE.HPC.'' Whenever anyone operates the system and then exits the Soft Front Panel, HPSTATE.HPC is overwritten to store the states of all the instruments on the PCIB bus as they exist *at that time.*

If this is the first time that you are using a previously operated system, then you need to get the system back to its original (default) state. You may also need to save the present state of the system so that it can be restored when you are finished with it.

Getting Back the Default System State. In order to guarantee that the system is in its default state, the state file must be absent (deleted). Then, the system will construct a new HPSTATE.HPC file from the hardware default parameters. However, the existing state file should not be deleted arbitrarily, because it has the last previous operating configuration. You, or someone else, may not want to lose this configuration. Therefore, the safest way to get back to the default condition is to save the current state file by renaming it. For example, assume that your work diskette is in the current drive. You want to save the current configuration in a file called ''LASTCONF'' and to obtain the default state. Do this by typing:

```
REN HPSTATE.HPC LASTCONF
```
and pressing Enter

Now there is no longer any HPSTATE.HPC file. When you load the PC Instruments System Software, the system will initialize the file with the default parameters.

Entering the Soft Front Panel

The procedures given here assume that you have done the following things explained earlier in this chapter:

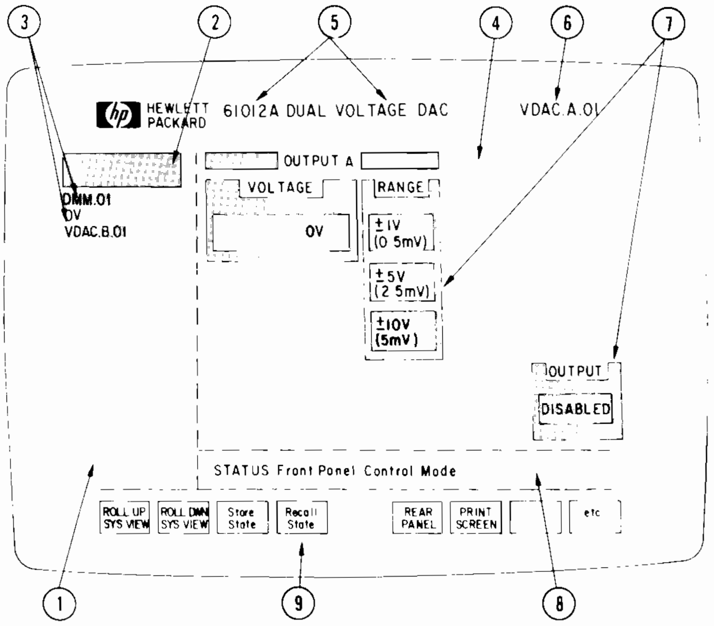
- Loaded DOS into memory.
- Installed and copied the required software to your work disk or diskette.
- Configured the HP PC Instruments System.
- If required, saved the current system state file by renaming it and deleted the state (HPSTATE.HPC) file.

To load PC Instruments, follow the procedure that applies to your system:

1. **For a Fixed-Disk System**, type:
   ```
   C>PANELS
   ```
   and press Enter
2. **For a High Capacity Diskette System**, (assuming that your work diskette is in drive A), type:
   ```
   A>PANELS
   ```
   and press Enter
3. **For a Standard Dual-Diskette System** with work diskette 2 in drive A, type:
   ```
   A>PANELS
   ```
   and press Enter

NOTE

When starting up the computer in the future, ''boot up''' with work diskette 1 in drive A. Then replace work diskette 1 with work diskette 2 and type: PANELS. This will give you the Soft Front Panel and leave drive B free for your own data or programming diskette.



1 - System View Window
2 - Label & status area of operable instrument
3 - Label and status areas of remaining instruments (in the order of their bus addresses)
4 - Interactive Instrument Window
5 - Official model number and name of operable instrument
6 - Default label of operable instrument
7 - Default front panel view of operable instrument
8 - Status area
9 - System Main Menu softkeys *

* The softkeys are duplicated by keys F1—F8 on the keyboard.

Figure 3-7. Typical Soft Front Panel Display

After a few seconds, the Hewlett-Packard logo and PC Instruments copyright display will appear on the screen. This will be quickly followed by the Soft Front Panel (see Figure 3-7). That display will automatically show the front panel of the first instrument (the one with the lowest bus address) in the Interactive Instrument Window. Now go to ''Trying Out the PC Instruments System.''

Trying Out the PC Instruments System

Chapter 2 of your Instrument Owner's Guide explains how to try out your instrument in the manual operating mode. That guide assumes that you have loaded the PC Instruments software with the system in the default configuration, as previously explained in this chapter. Refer now to that document and try operating the instrument. You can point to and select the active Soft Front Panel fields by using either the mouse or the computer cursor controls and the Enter key. If you have a second instrument you wish to try, bring that instrument into the Interactive Instrument Window by selecting its label in the System View Window.

NOTE

The end of Chapter 2 in the Instrument Owner's Guide tells you how to exit the Soft Front Panel. Whenever you leave the Soft Front Panel, the file HPSTATE.HPC will have parameters corresponding to the state of the system as you left it. This is the way the system will ''wake up'' the next time you, or someone else, uses it. If you want to leave the system in the state that you saved previously in this chapter (e.g., to ''LASTCONF''), then you must use the **Recall State** *softkey to return to that state before you exit the Soft Front Panel.*

Learning the Details of Manual-Mode Operation

If you are following the ''Recommended Reading Sequence for First-Time Users'' (given in Table 2 of the Introduction), you are now ready to learn the full use of the manual mode. Go on to Chapter 4 of this guide and Chapter 3 of the applicable Instrument Owner's Guide. The explanations include how to give an instrument a unique ''label'' that is descriptive of its function in the system (when you first use it, each instrument operates with a default label in that field).

Programmed-Mode Operation

If you want to write application programs in BASIC, then read Chapters 5 and 6 of this guide and Chapter 5 of your Instrument Owner's Guide. Refer also to the BASIC manual that is supplied for your computer. You also have several methods for graphing the data returned by your BASIC program. Refer to Chapters 6 and 7 of this guide.

Completing the Installation

After learning the manual mode of operation and gaining familiarity with programmed operation, you are ready to complete the installation of your instrument by connecting it to your application. Chapter 4 of your Instrument Owner's Guide shows you how to do that. Your instrument will then be ready to perform the task that you wish it to do in the system.

In Case of Trouble

Here are some suggestions on what to do if your HP PC Instrument does not seem to operate properly.

Verifying PC Instruments Operation

When you first select the Soft Front Panel, the system does a brief self check of the system hardware. This self check, which you may not even notice, is sufficient to allow you to begin system operation.

Preliminary Self Check Failure

The system can fail the preliminary self test in two ways. If the system loads but fails self test, there will be an error message in the status window (refer to Appendix C). If the system cannot load, you will not get a Soft Front Panel display. At this point, be sure that you have configured the system properly (refer to ''Configuring the PC Instruments System'' in this chapter). If the configuration is o.k., then run the system verification test explained in Appendix B. That test runs as a BASIC application and does not require the Soft Front Panel. The test results should indicate where the problem is.

Is it Your PC Instruments System?

Make sure your computer is working properly before assuming there is something wrong with your PC Instruments System. If other applications work (Wordstar, BASIC etc.), then proceed to look for a fault in you PC Instruments System.

Operational Failure

If your system runs, but not properly, here are some things you can check:

a. An instrument does not operate, but one or more other instruments do -

- Check that the instrument's address switch is set according to the instructions in its guide. Make certain that no two instruments are set to the same address.
- Switch the plug position on the HP PCIB Interface Cable with an instrument that is working.

- Run the instrument verification program as given in Appendix B of it's owner's guide.

b. The system operates in the manual mode but not in the programmed mode -
- Wrong version of BASIC (refer to Table 1-1 in Chapter 1).
- You failed to create and/or use a Program Shell.
- Your PC Instruments configuration does not agree with the one existing when you generated the Program Shell.
- Error in your BASIC program.
- Spelling or parameter errors in your PC Instruments CALL statements.

c. Certain Soft Front Panel menu keys do not respond, or give the wrong response -
- Incorrect operation; read Chapter 4.
- Possible system software problem.

d. Instead of the Soft Front Panel, a system error message followed by a number appears on your screen. This indicates a fault within your PC Instrument Software. Refer to ''If You Need Help.''

NOTE

Your HP PC Instruments software is ''bundled,'' in that the software that controls the instrument modules is combined with the software that creates the Soft Front Panel. Furthermore, the library of programming statements is included in the same software package. If something destroys part of your PC Instruments software, it could create a problem in any of the above operations. If this happens, make a new work diskette from your original master diskettes.

If You Need Help

If you have determined that you have bad hardware or software, or cannot determine the cause of the problem, you can get help during normal business hours by calling the number listed in the *HP Personal Instruments Support Guide* included with this guide.

The person at the service center will help you solve the problem or recommend what steps to take to have it resolved. If some part of your system is defective, consult the guide for warranty and service policy information.

Software from Independent Software Vendors

There are some software products for PC Instruments that are marketed by third-party organizations - referered to as ''independent software vendors,'' or ''ISVs.'' The diskettes from ISVs do not bear the Hewlett-Packard label. To obtain support or updating information for ISV software, contact the vendor or dealer who supplied it.

4 Manual Instrument Control

How This Chapter is Organized

This chapter begins with a brief description of the HP PC Instruments Soft Front Panel (SFP) and how it is organized. Next, there is an example of how the most common menu functions are used for controlling two instrument modules in a simple application. Finally, each of the Soft Front Panel menus is described in detail.

What You Can Do From the Soft Panel

When you type PANELS from DOS (or CALL PANELS from BASIC as explained in Chapter 5), a graphical in Chapter 5), a graphical representation of your PC Instruments System appears on the screen. This representation is called the ''Soft Front Panel'' and it allows you to:

- label your instruments by giving each a unique name that is meaningful within the context of your application.
- view the other instruments in your system via the System View Window.
- control your instruments by:
  - choosing an instrument for manual operation (you control one instrument at a time)
  - operating the instrument by using screen graphics equivalent to actual instrument controls. The graphics on the Soft Front Panel replace the switches, knobs, buttons, and indicators found on traditional instruments.

  - enabling output instruments. Whenever the Soft Front Panel is entered from DOS, all instruments that generate an output have their outputs disabled. You can subsequently enable and disable the outputs at any time.
- print the contents of the screen for a hard copy of the data.
- store or recall the state (set-up conditions) of every instrument in the system in a ''State'' file. You can have more than one state file for the same hardware. By storing set-up conditions for all the instruments, you can restore those same instrument conditions at any time by recalling the file, without having to access the instruments. State files also are needed when you write your applications programs and want to initialize your instruments to certain conditions (refer to Chapter 5).
- store a Program Shell in a file so that you can write a BASIC application program to control your instruments (refer to Chapter 5). The Program Shell contains system information (such as configuration, common variables, etc.) in BASIC.

How the Soft Front Panel is Organized

Figure 4-1 shows a typical Soft Front Panel display. It consists of three fixed-size windows and a softkey area. Some windows have interactive fields that you can point to and select. The Soft Front Panel interactive fields and softkeys give you manual control over the system.

System View Window

Description. The System View Window displays all of the instruments connected to the PC Instruments interface and provides you with a short summary of each (as shown in Figure 4-1). This allows you to monitor the status of all the instruments while you control the one that you select (refer to ''How to Select an Instrument'' further in this chapter). When all instruments cannot fit in the window at one time, you can view more instruments by scrolling the window with softkeys. The instruments are displayed in the order of their addresses on the System Interface Bus (refer to Chapter 2). When it is selected for the Interactive Instrument Window, an instrument disappears from the System View Window.

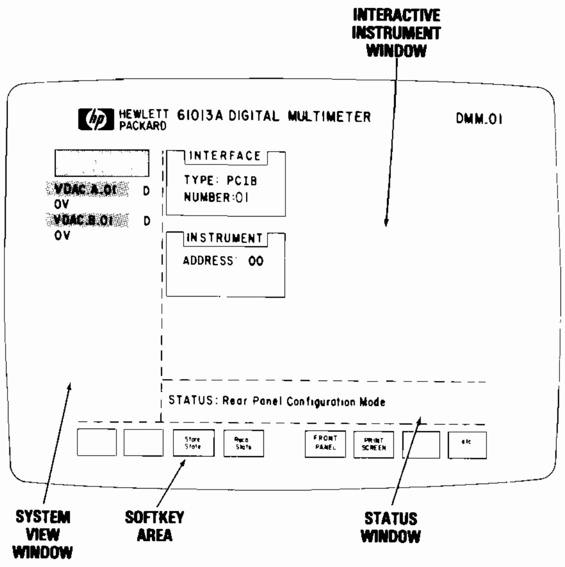
The first line of each instrument summary is the user-defined or factory-default ''label'' that identifies the instrument. It is the same as the label that appears on the instrument's front and rear panel views. Some instruments have more than one label in the System View Window. For example, the Dual Voltage DAC has one for each output. Instruments with more than one label also have more than one rear panel display. For details on how to change this label, refer to ''How To Label and Readdress the Instrument'' further in this chapter.

Next to each instrument label is a system prompt. The meanings of these prompts are:

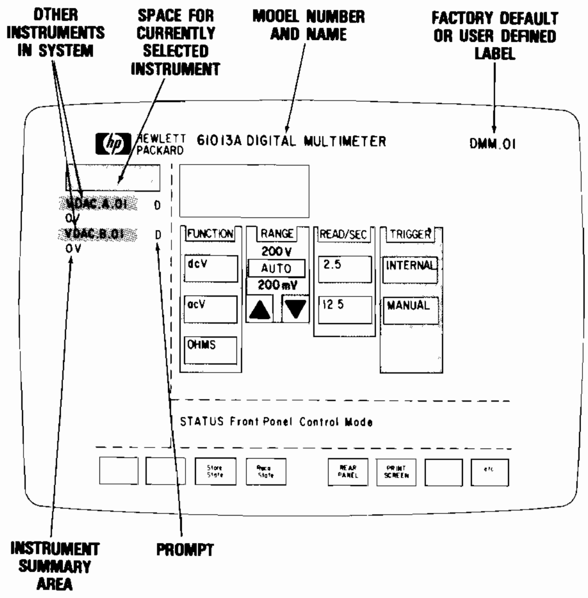
E	An error has occurred with this instrument.
D	The instrument's output is disabled (only for output instruments)
M	The instrument is in manual trigger mode (only for input instruments)
W	The instrument is waiting for an external event.

If there is no prompt next to an instrument, then none of these conditions exists.

Beneath an instrument's label field is a noninteractive tabular field or graphical display. This summarizes the condition of the instrument (how it is set, the value of its output, what value it is reading, etc.) For input instruments set to internal trigger, this field or display is updated automatically according to what changes take place in the system. This permits you to view the instrument's response after it is stimulated by the instrument in the Interactive Instrument Window. You cannot directly change the items listed in this field or display.



a. Typical Instrument Rear Panel View



b. Typical Instrument Front Panel View

Figure 4-1. An Example Soft Front Panel

How to Select an Instrument. To manually control or configure an instrument using the Soft Front Panel, point to and select its label from the list appearing in the System View Window. If the instrument has more than one label in the System View Window, select the label of the particular input or output you want. If the label does not appear in the System View Window, the instrument is either incorrectly installed or has been relabled. If it has been relabled, you will have to point to and select each label in the list until you can identify the instrument you want. To help you identify them, selected instruments display their model name and number in the upper-left corner of their Interactive Instrument Window. Also, the instrument's ACTIVE indicator (on the front panel of the actual hardware), turns on.

Interactive Instrument Window

Description. The front or rear panel view of a system instrument appears in this window when you select that instrument from the System View Window (as shown with the Digital Multimeter in Figure 4-1). When the system is turned on, the front panel view of the first instrument in the System View Window is displayed in the Interactive Instrument Window. Subsequently, you can use a softkey to choose which view of an instrument appears in this window. The Interactive Instrument Window lets you label each instrument via its rear panel view (see Figure 4-1a) and manually control each instrument via its front panel view (see Figure 4-1b). The first line of the Interactive Instrument Window is reserved for the instrument's model number, model name, and user-defined or default label.

How to Label and Readdress the Instrument. You can change the instrument label and assign one specific to your application. Do this by pointing to and selecting the label on the rear panel view. Next, back up the cursor to the first character of the label and enter a new one from the keyboard. The label can have up to 13 characters. The characters allowed are letters, numbers, and the decimal point; the first character must be a letter.

You cannot interact with the configuration information on the rear panel display. To change the address of your instrument, you must exit the Soft Front Panel and reset the address switch on the back of the instrument, as explained in Chapter 2. Remember, this will change the order in which your instrument appears in the System View Window.

How to Control the Instrument. The remaining area of the instrument's front panel view consists of interactive graphical fields that allow you to change the instrument mode, select its function, set its range, etc. When needed, you can also enter numeric quantities from the keyboard. For a complete operating description of an instrument's front panel, refer to Chapter 3 of its Owner's Guide.

Status Window

The Status Window (see Figure 4-1) consists of five lines that display system and instrument control software messages and give you space to enter file information.

Status lines one and two are reserved for information having to do with files. These messages appear only when you are within a menu that requires you to specify a filename or a directory name.

The third status line provides system status information. Status messages are easily recognizable because they are always preceded by "STATUS:". Status messages guide you through manual operation. They show you what the system is doing or what you should be doing when you press the various menu softkeys and interact with the system.

Status lines four and five provide prompts and error messages. System prompts are preceded by "NOTE:", while error messages are preceded by "ERROR:".

System Softkey Area

The last remaining area of the Soft Front Panel (Figure 4-1) is reserved for the system softkeys. You can access a softkey either by pointing to and selecting it on the Soft Front Panel, or by pressing a corresponding keyboard function key that operates as a softkey. The softkeys support the system level menus, which allow you to perform such necessary functions as storing and recalling instrument states. The Main Menu is the first one accessed when entering the Soft Front Panel and the last when leaving it.

The Main Menu includes a **Store State** and a **Recall State** softkey. You can save the present state (e.g., output voltage, range) of all instruments in the system by pressing the **Store State** softkey. You identify this state by storing it in a file under a specified name. Files that contain instrument states are called ''state files''. The same instrument state can be recalled at any time by pressing the **Recall State** softkey and entering the same state file name. State files are also required when initializing the instrument from BASIC (refer to Chapter 5).

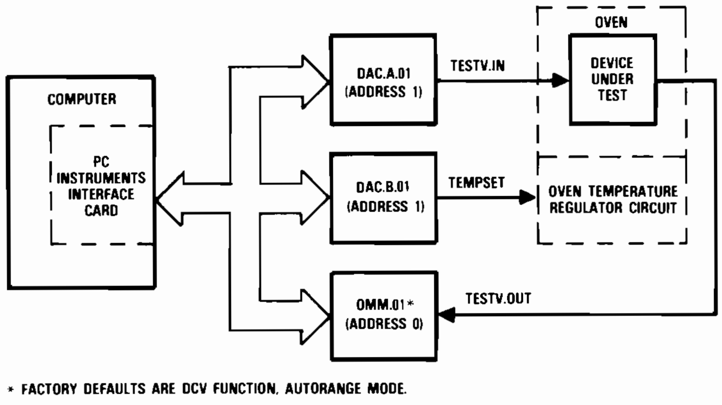
Each softkey menu is explained, in detail, further on in this chapter (refer to ''Softkey Menus''). But first, we will show how the softkeys are used in an example of manual control operation.

Example of Manual Instrument Control

Manual control means operating your HP PC Instruments System from the Soft Front Panel. Here is a specific example of manual control.

Purpose of the Example

This example shows how manual control of PC Instruments can be used in a test procedure that involves two instrument modules to monitor a device under test. The example is intentionally simple and general, and by no means shows everything that you can do in the manual mode.



COMPUTER
PC INSTRUMENTS INTERFACE CARD
DAC.A.01 (ADDRESS 1)
DAC.B.01 (ADDRESS 1)
OMM.01* (ADDRESS 0)
TESTV.IN
TEMPSET
TESTV.OUT
OVEN
DEVICE UNDER TEST
OVEN TEMPERATURE REGULATOR CIRCUIT
* FACTORY DEFAULTS ARE DCV FUNCTION, AUTORANGE MODE.

Figure 4-2. Test Setup for the Example of Manual Control.

Hardware Used

Figure 4-2 is a block diagram of a test setup that uses the PC Instruments Dual Voltage DAC (Digital-to-Analog Converter) and Digital Multimeter (DMM) instrument modules. The DAC has two voltage outputs, or channels, and is treated as two separate instruments. One channel determines the oven temperature by applying a dc voltage to the oven temperature control circuit. The other DAC channel applies an input voltage to the device under test inside the oven. The DMM measures the voltage output of the device under test.

Example Test Procedure

Figure 4-3 illustrates the procedure for this test example, which is shown step-by-step in the series of drawings presented in Figure 4-4. From Figure 4-3, note that PC Instruments allows you to change the original instrument labels to ones that are meaningful to the specific application. Also, each DAC is enabled individually from its front panel view. (If desired, both DACs could be set up and then simultaneously enabled with the system **ENABLE OUTPUTS** softkey.)

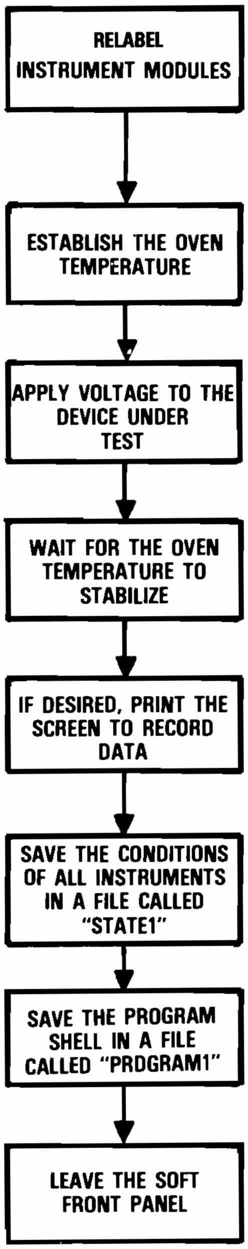
For this example, we named the state file ''STATE1'' and the program file ''PROGRAM1''. The file ''STATE1'' stores a complete description of all three instruments as they are set up for this test. As long as they remain in the system at their same bus addresses (refer to Chapter 2), the instruments can always be returned to this state by recalling file ''STATE1'' (as explained under ''Recall State Menu''). The information stored in Program Shell ''PROGRAM1'' includes the rear panel information of these three instruments. Both ''STATE1'' and ''PROGRAM1'' will be referred to again in Chapter 5.

Entering the Soft Front Panel

Enter the Soft Front Panel from DOS by typing:

A> PANELS and pressing Enter

After the brief welcome screen, the Soft Front Panel display will appear. Its System View Window displays the three instruments in the order of their bus addresses and the front panel view of the first instrument will appear in the Interactive Instrument Window.


RELABEL INSTRUMENT MODULES
ESTABLISH THE OVEN TEMPERATURE
APPLY VOLTAGE TO THE DEVICE UNDER TEST
WAIT FOR THE OVEN TEMPERATURE TO STABILIZE
IF DESIRED, PRINT THE SCREEN TO RECORD DATA
SAVE THE CONDITIONS OF ALL INSTRUMENTS IN A FILE CALLED "STATE1"
SAVE THE PROGRAM SHELL IN A FILE CALLED "PROGRAM1"
LEAVE THE SOFT FRONT PANEL

Change ''VDAC.A.01'' to TESTV.IN''
Change ''VDAC.B.01'' to TEMPSET''
Change ''DMM.01'' to ''TESTV.OUT''

Change ''TEMPSET'' Voltage to 5.0 V
Enable Output

Change ''TESTV.IN'' Voltage to 3.5 V
Enable Output

Read DMM results in System View Window

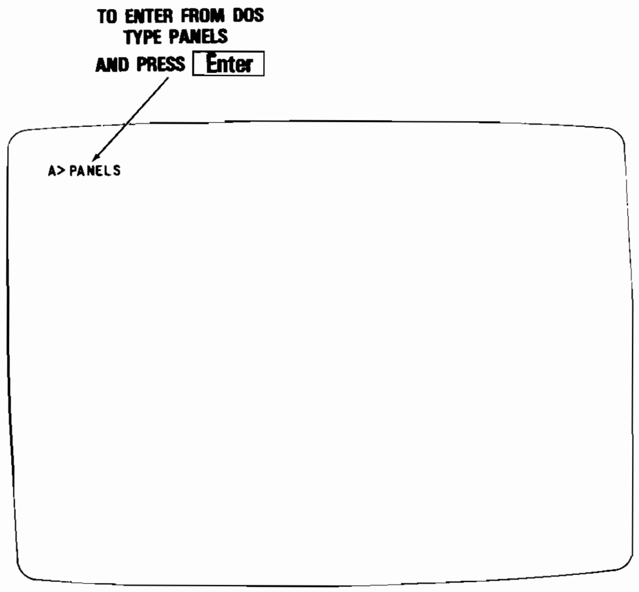
Select the **PRINT SCREEN** softkey.

Use the Store State Menu

Use the Store Program Menu

Press the **etc** and **EXIT** softkeys

Figure 4-3. Procedure for Manual Mode Control.



```
A> PANELS
```

Figure 4-4a. Manual Control Example – Entering SFP from DOS

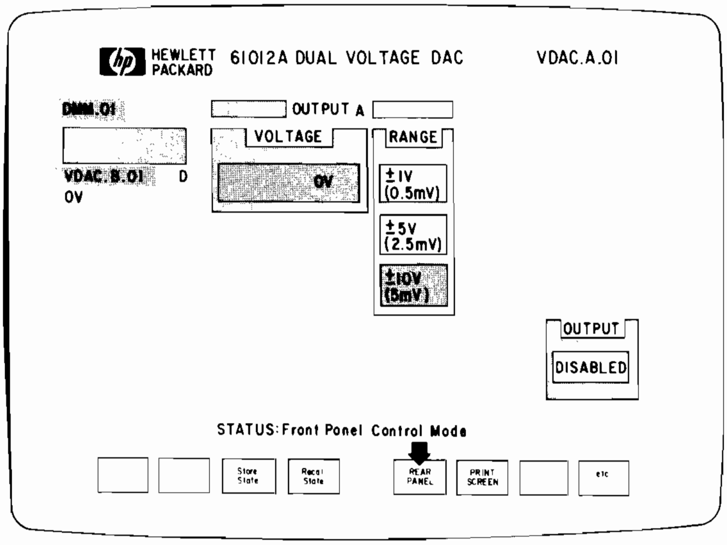
Labeling Your Instruments

Assign labels to your Dual Voltage DAC and Digital Multimeter that are meaningful for this example by:

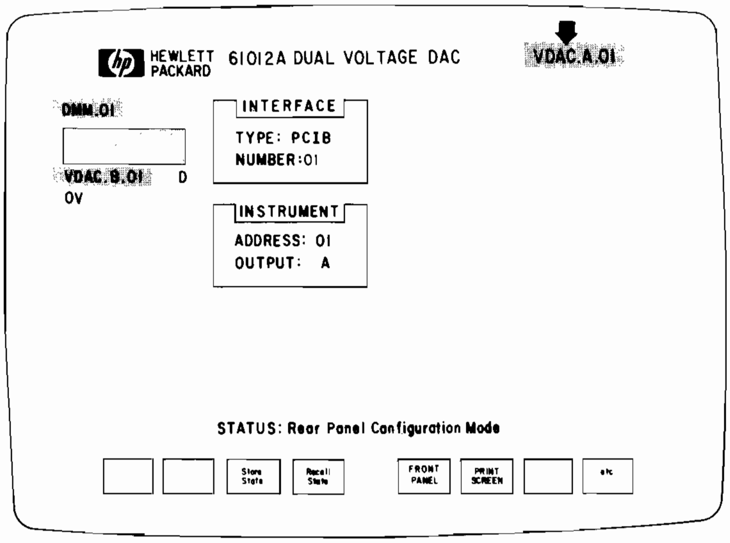
- entering rear panel configuration mode via the the **REAR PANEL** softkey
- pointing to and selecting the label field on the instrument's rear panel
- backing up the cursor to the first character position on the label.
- typing the new label and pressing Enter

NOTE

This process is shown only for the VDAC.A.01. Use the same process for relabeling the VDAC.B.01 to "TEMPSET" and DMM.01 to "TESTV.OUT."



STATUS: Front Panel Control Mode



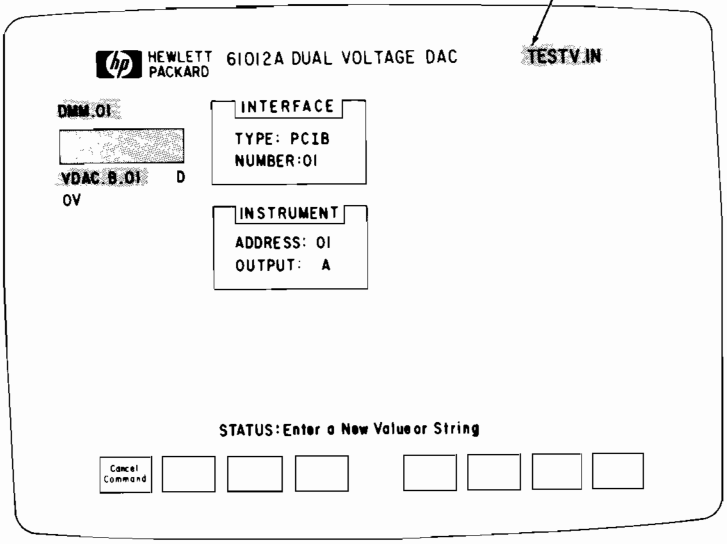
TYPE: PCIB
NUMBER:01

ADDRESS: 01
OUTPUT: A

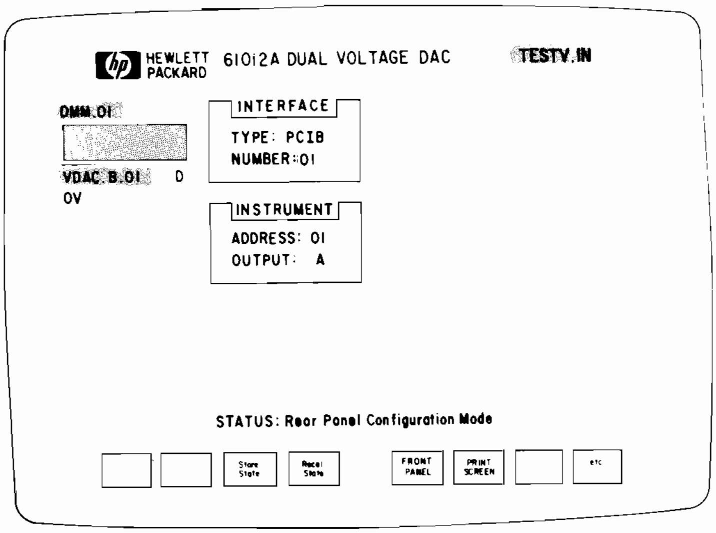
STATUS: Rear Panel Configuration Mode

Figure 4-4b. Manual Control Example – Relabeling VDAC.A.01

STARTING AT THE LEFT,TYPE THE NEW LABEL AND PRESS $\boxed{\text{Enter}}$



STATUS:Enter a New Value or String



STATUS: Rear Panel Configuration Mode

Figure 4-4c. Manual Control Example – Relabeling VDAC.A.01

Controlling Your Instruments

Manually control the Digital Multimeter and the Dual Voltage DAC to generate outputs and to monitor the status of TESTV.OUT in the System View Window. Do this by:

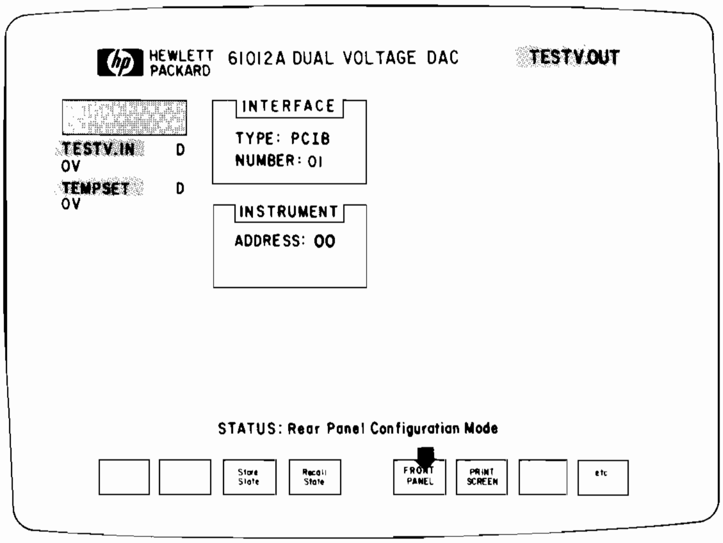
- getting the front panel view (Press the **FRONT PANEL** softkey)
- selecting the label of the instrument to control from the System View Window
- changing the voltage of TEMPSET and enabling its output
- changing the voltage of TESTV.IN and enabling its output

NOTE

The default settings for the DMM are used in this example. With these values, the DMM will continually take d-c voltage readings at a rate of 2.5 per second.

NOTE

After changing the value of TEMPSET, you can recall TESTV.OUT back into the Interactive Instrument Window by pressing the **LAST INSTRMT** *softkey. This is useful if you want to change (or just view) the front and rear panel settings of the DMM. To restore TEMPSET to the Interactive Instrument Window, press* **LAST INSTRMT** *again.*



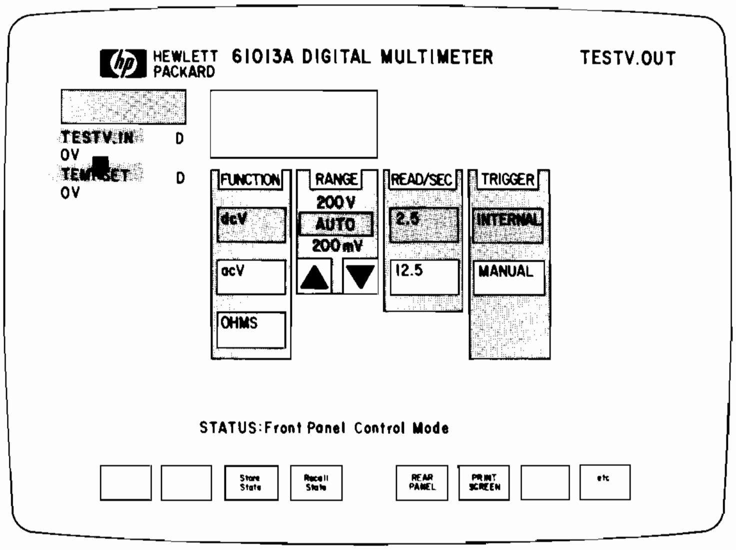
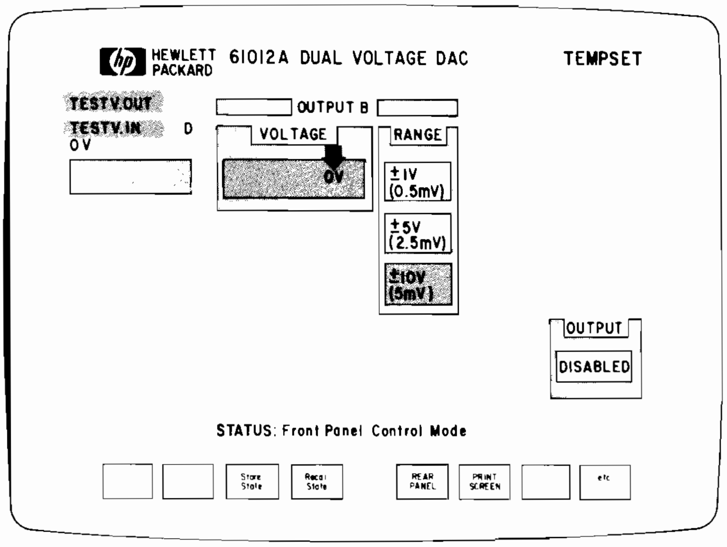
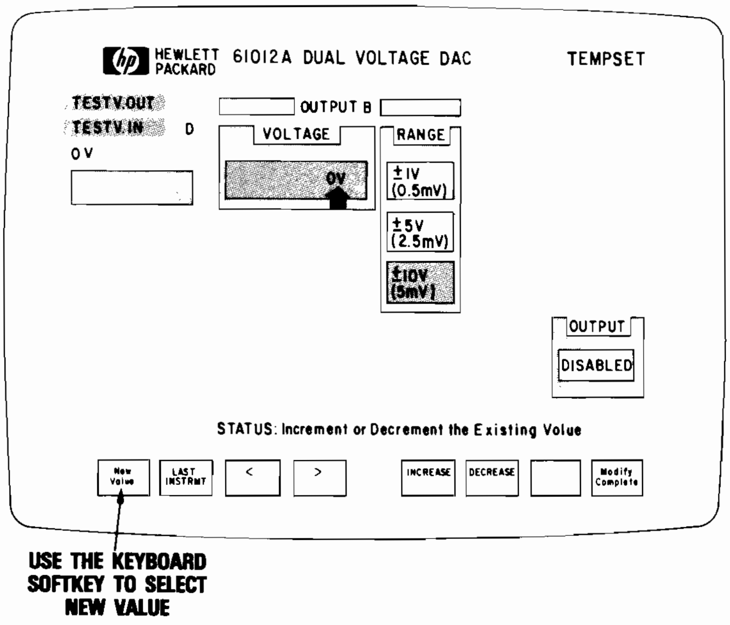


Figure 4-4d. Manual Control Example – Selecting DAC TEMPSET

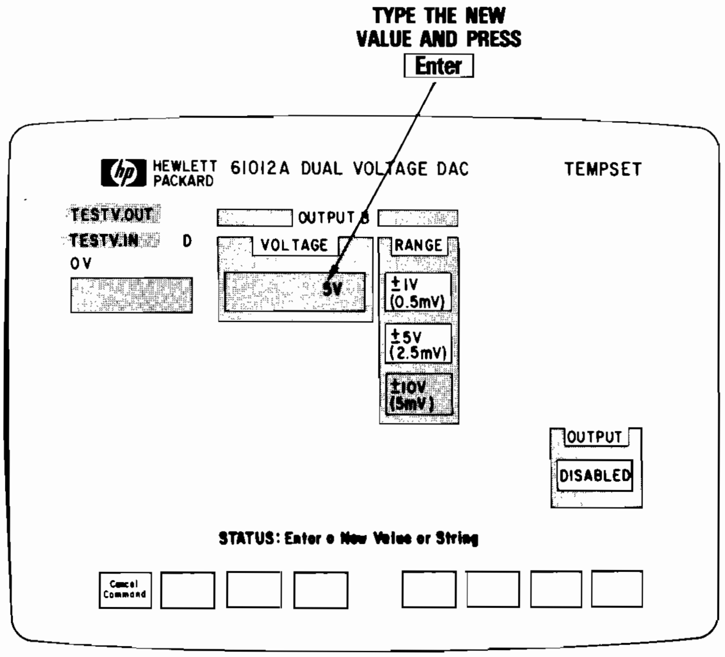


STATUS: Front Panel Control Mode



STATUS: Increment or Decrement the Existing Value

Figure 4-4e. Manual Control Example – Changing the Value of TEMPSET



HEWLETT PACKARD 61012A DUAL VOLTAGE DAC TEMPSET

TESTV.OUT
TESTV.IN D
0 V

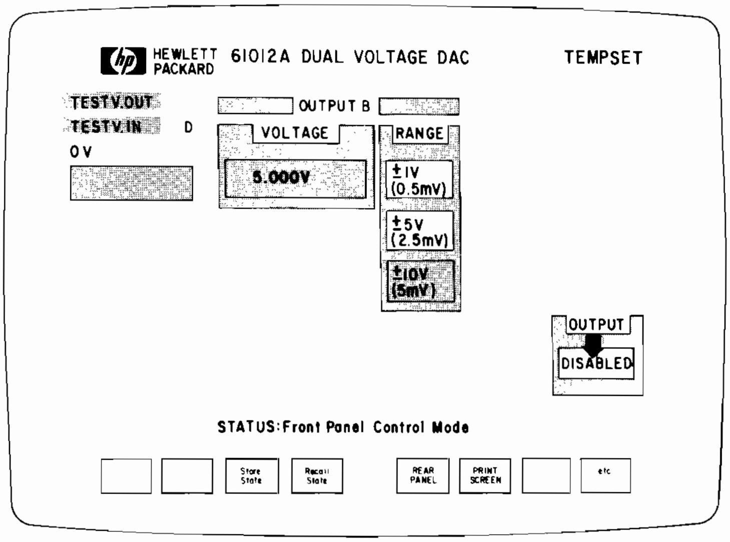
OUTPUT B

VOLTAGE: 5V

RANGE: ±1V (0.5mV) | ±5V (2.5mV) | ±10V (5mV)

OUTPUT: DISABLED

STATUS: Enter a New Value or String



HEWLETT PACKARD 61012A DUAL VOLTAGE DAC TEMPSET

TESTV.OUT
TESTV.IN D
0 V

OUTPUT B

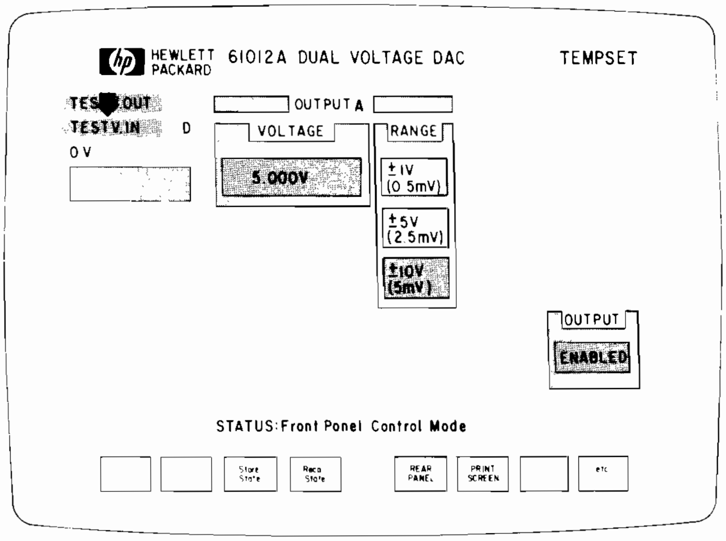
VOLTAGE: 5.000V

RANGE: ±1V (0.5mV) | ±5V (2.5mV) | ±10V (5mV)

OUTPUT: DISABLED

STATUS: Front Panel Control Mode

Figure 4-4f. Manual Control Example – Changing the Value and Enabling the Output of TEMPSET

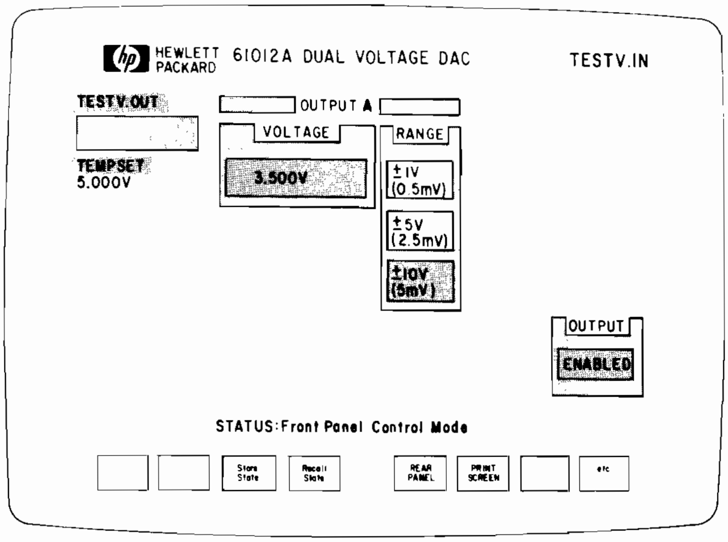


HEWLETT PACKARD 61012A DUAL VOLTAGE DAC TEMPSET

TESTV.OUT
TESTV.IN D
0 V

OUTPUT A
VOLTAGE 5.000V
RANGE ±1V (0.5mV) ±5V (2.5mV) ±10V (5mV)
OUTPUT ENABLED

STATUS:Front Panel Control Mode



HEWLETT PACKARD 61012A DUAL VOLTAGE DAC TESTV.IN

TESTV.OUT
TEMPSET
5.000V

OUTPUT A
VOLTAGE 3.500V
RANGE ±1V (0.5mV) ±5V (2.5mV) ±10V (5mV)
OUTPUT ENABLED

STATUS:Front Panel Control Mode

Figure 4-4g. Manual Control Example – Selecting DAC "TESTV.IN" and Changing its Value

Now change the voltage of TESTV.IN and enable its output. This is the same process as previously shown for TEMPSET, except that only the end results are shown for TESTV.IN.

Printing the Screen

You may record data by printing the entire screen.

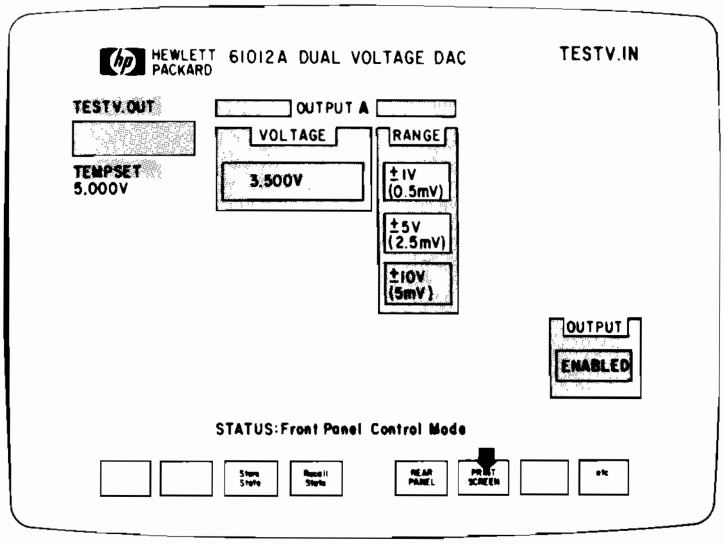
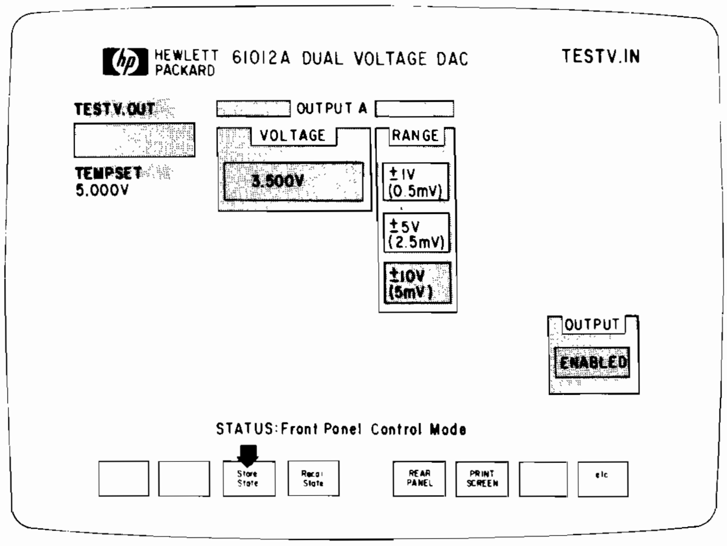


Figure 4-4h. Manual Control Example – Printing the Screen

Storing the Instrument States

Store the state of all the instruments in the system by:

- invoking the Store State Menu
- specifying the filename "STATE1" (name used in this example) for storing the state of TESTV.IN, TEMPSET and TESTV.OUT
- Saving the state file and returning to the Main Menu.

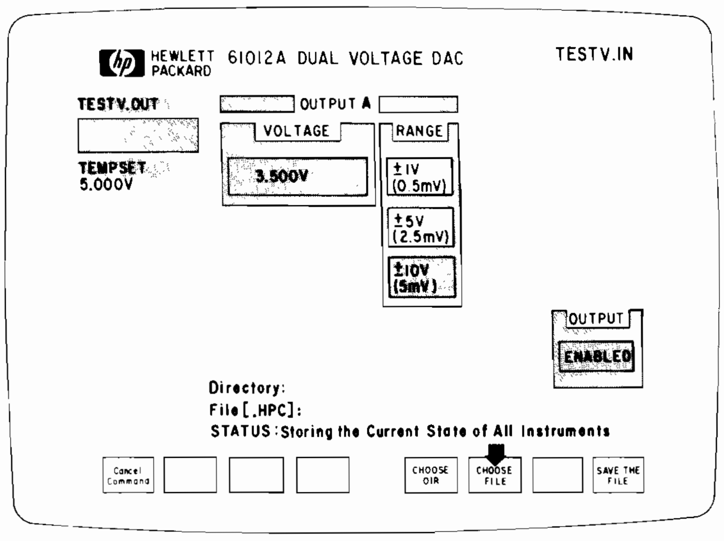


TEMPSET 5.000V

VOLTAGE 3.500V

OUTPUT ENABLED

STATUS:Front Panel Control Mode



TEMPSET 5.000V

VOLTAGE 3.500V

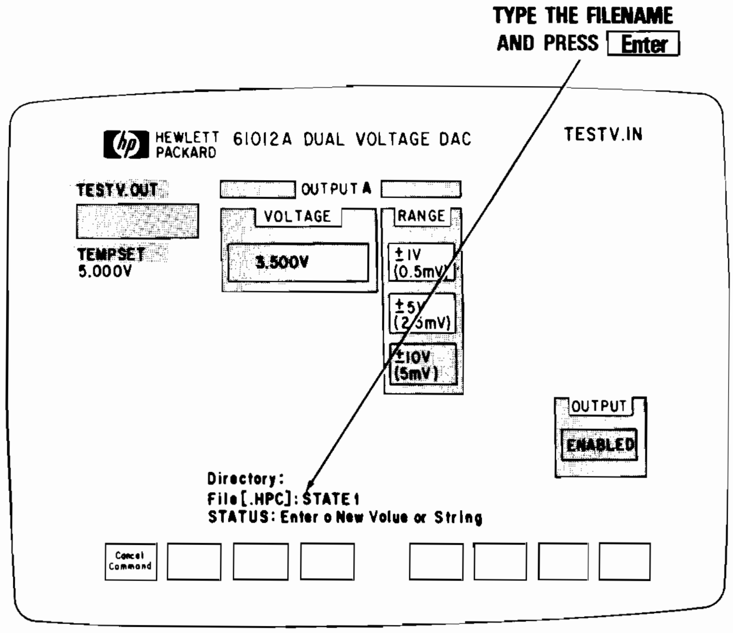
OUTPUT ENABLED

Directory:

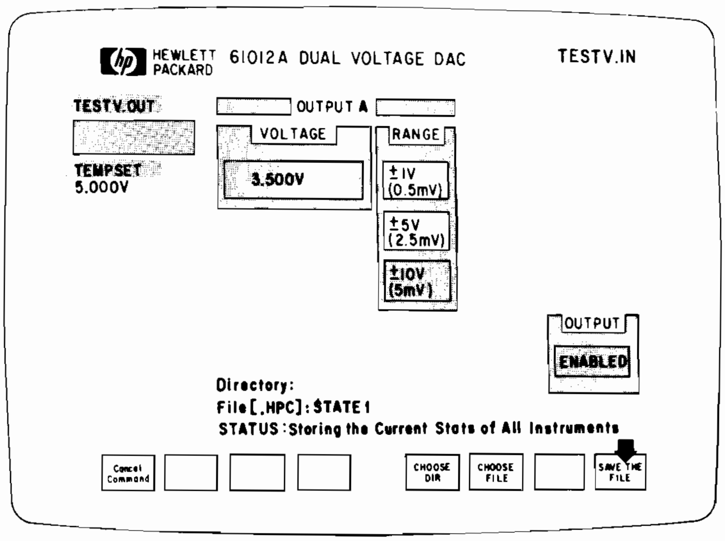
File[.HPC]:

STATUS:Storing the Current State of All Instruments

Figure 4-4i. Manual Control Example – Obtaining the Recall State Menu and Choosing a File



Directory:
File[.HPC]: STATE1
STATUS: Enter a New Value or String



Directory:
File[.HPC]: STATE1
STATUS: Storing the Current State of All Instruments

Figure 4-4j. Manual Control Example – Saving File "STATE1."

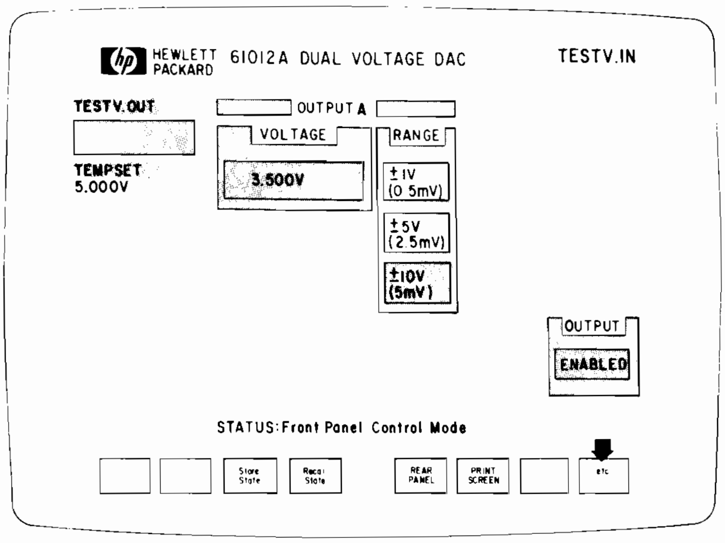
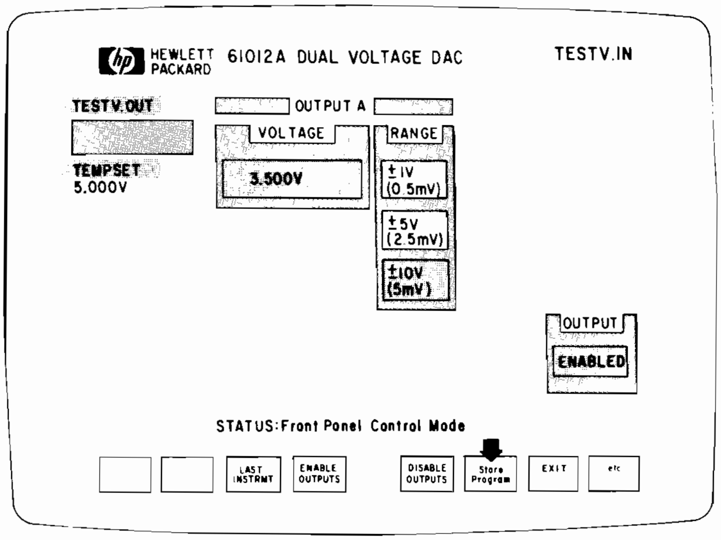


Figure 4-4j. Manual Control Example – Accessing the Store Program Menu.

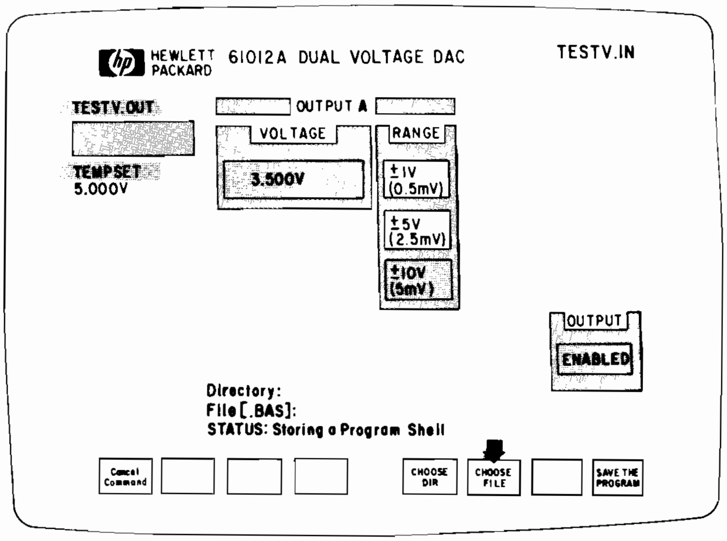
Storing the Program Shell

Generate and store a Program Shell for this test setup by:

- Invoking the Store Program Menu
- Specifying the filename "PROGRAM1" (name used in example) for storing the program specifically for this example
- Saving the program and returning to the Main Menu

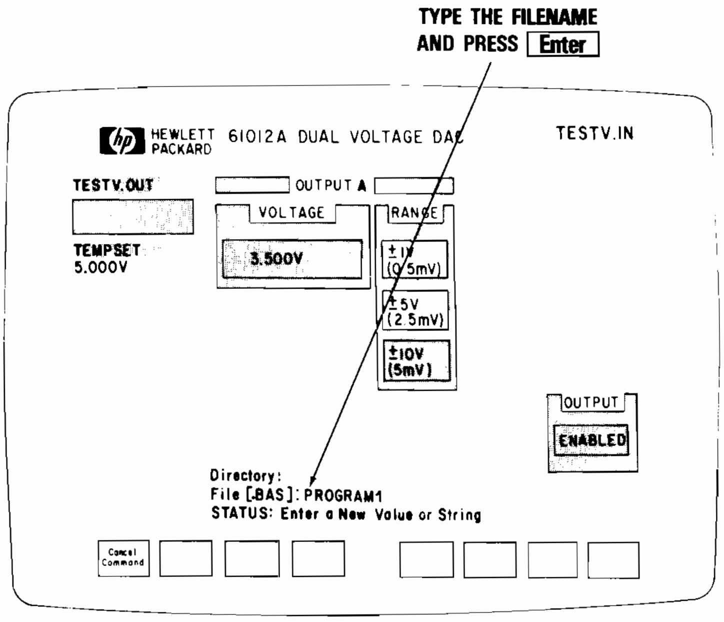


STATUS:Front Panel Control Mode

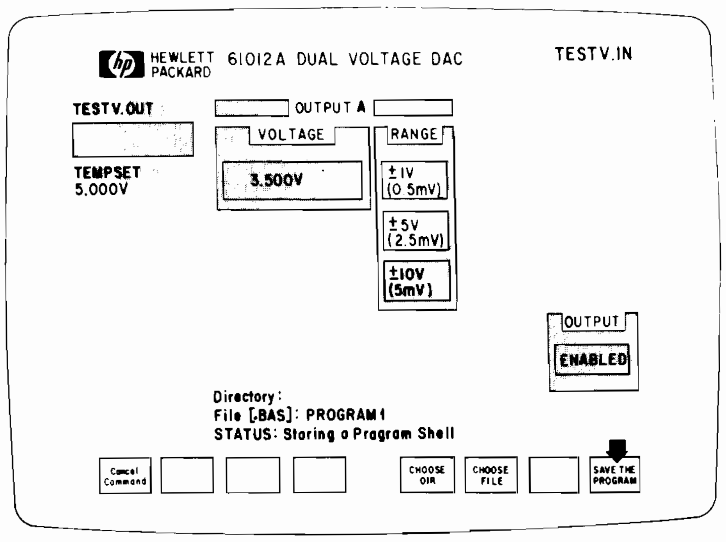


Directory:
File[.BAS]:
STATUS: Storing a Program Shell

Figure 4-4k. Manual Control Example – Invoking Store Program Menu and Choosing a File



Directory:
File [.BAS]: PROGRAM1
STATUS: Enter a New Value or String

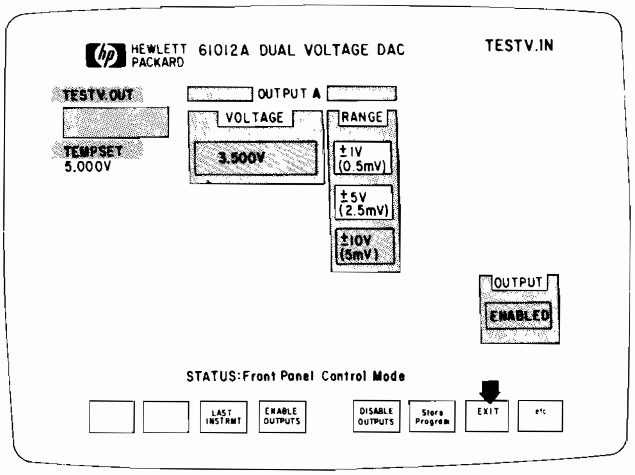


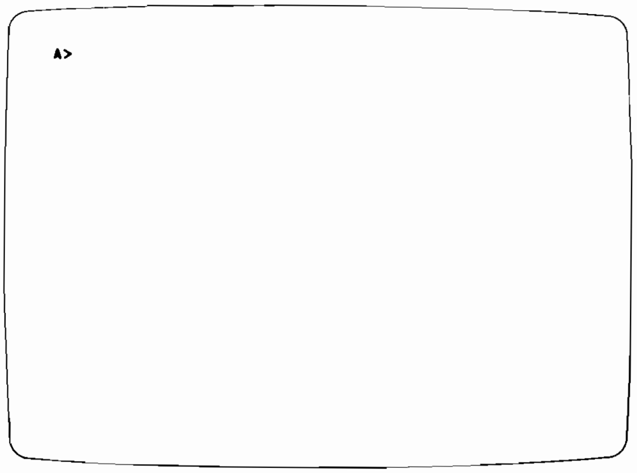
Directory:
File [.BAS]: PROGRAM1
STATUS: Storing a Program Shell

Figure 4-41. Manual Control Example – Saving File "PROGRAM1"

Exiting the Soft Front Panel

Exit the Soft Front Panel and return to DOS. The instrument states are stored in file ''STATE1'' (and automatically in ''HPSTATE.HPC'' as explained at the end of this chapter). The Program Shell is stored in file '' PROGRAM1.'' These files will be used when you develop a program for this test example in Chapter 5.





```
A>
```

Figure 4-4m. Manual Control Example – Exiting the SFP

Softkey Menus

The following text describes how the PC Instruments softkey menus are organized and how each menu functions.

Introduction

The Soft Front Panel softkeys are organized into seven menus (see Figure 4-5). When you enter the Soft Front Panel, the Main Menu is automatically invoked. It is the first menu to appear when you enter the Soft Front Panel and the last to appear when you leave it. The Main Menu gives you access to every other menu.

The Numeric Modify Menu is automatically invoked whenever you select a field that has a numeric value that can be changed. The Keyboard Entry Menu is accessed interactively from the Numeric Modify Menu or whenever you select a field or softkey that requires you to make an alphanumeric entry. Whenever you enter a menu from another menu, you will always be returned to the menu you just came from.

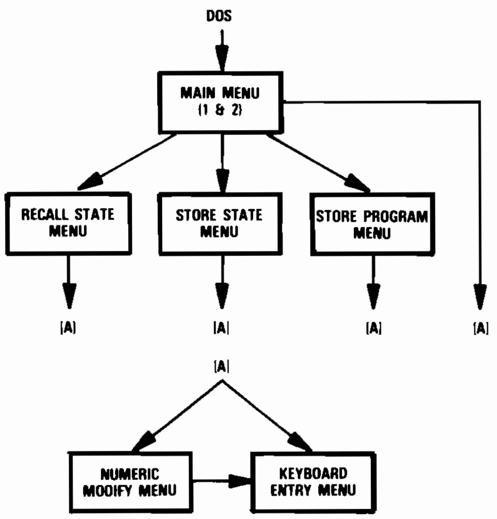


Figure 4-5. Softkey Menu Functions and Organization.

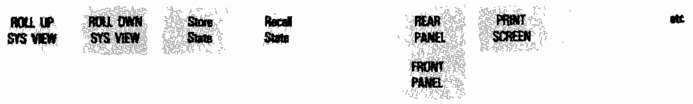
Table 4-1 lists the softkeys alphabetically. The table identifies each softkey function and in which menu or menus it is used. Menu softkey legends are of two forms (refer to Table 4-1). A softkey legend that is all capital letters (e.g., **PRINT SCREEN**) performs some immediate function. A softkey legend with only its first letter capitalized (such as **Store State**) takes you to another menu, as described by the legend.

The rest of this chapter describes each menu in detail.

Main Menu

This is the first level of softkeys to appear when the Soft Front Panel is run. Since only eight softkeys fit on the screen at a time, there are two rows of Main softkeys – Main 1 and Main 2. You access each row by pressing **etc.** The Main level softkeys are:

Main 1 (First row). This is the first menu to appear when you enter the Soft Front Panel.



ROLL UP SYS VIEW **ROLL DWN SYS VIEW** Appear only when there are more instruments in the system than can fit in the System View Window. These softkeys allow you to view all the instruments in the system by scrolling the System View Window upward or downward, instrument-by-instrument. These softkeys are available on both Main softkey levels (Main1 and Main2).

Store State Invokes the Store State softkey menu as described later in this chapter.

Recall State Invokes the Recall State softkey menu as described later in this chapter.

REAR PANEL **FRONT PANEL**

Pressing **REAR PANEL** causes the selected instrument's rear panel view to appear in the Interactive Instrument Window. The softkey legend then changes to **FRONT PANEL** . The rear panel display shows configuration information (model, bus address, interface number, secondary address (where used)), and the instrument label. The interface and bus address information are fixed for the PC Instrument interface; you cannot change these values. If you want to assign a meaningful label to the instrument, you can do so by pointing to and selecting the label field on the rear panel display and typing over the current label. After you have finished with the rear panel view of one instrument, you can either select another rear panel to view (from the System View Window) or exit the rear panel display by pressing **FRONT PANEL** (see Figure 4-4d of the Manual Instrument Control Example).

Pressing **FRONT PANEL** causes the selected instrument's front panel display to appear in the Interactive Instrument Window. This allows you to manually control the instrument. This also causes the softkey's legend to change to **REAR PANEL** so you can obtain the rear panel display of an instrument at any time.

PRINT SCREEN

Prints the contents of the screen (or display) to the system line printer (PRN:device). ''Printing the Current Screen'' appears in the Status Window.

etc

Advances you to the second row (Main 2) of Main level softkeys.

Main 2 – (Second Row). You obtain this menu by pressing **etc** in the Main 1 Menu.

ROLL UP SYS VIEW	ROLL DWN SYS VIEW	LAST INSTRMT	ENABLE OUTPUTS		DISABLE OUTPUTS	Store Program	EXIT	etc

ROLL UP SYS VIEW **ROLL DWN SYS VIEW**

(Same as in Main 1 Menu).

LAST INSTRMT	Allows you to alternate, or ''toggle'' between the instrument presently in the Interactive Instrument Window and the *last state* of the one previously in that window. With **LAST INSTRMT**, you can access the front and rear panel views of the instrument last in the window without having to reselect it from the System View Window. The state of the instrument will be exactly as you left it when you removed it from the Interactive Instrument Window. This is useful if you want to change a numeric value of a stimulus instrument, but must also interact with the response instrument. It saves you time (and keystrokes) when toggling between two instruments in the Interactive Instrument window. Whenever you select a new instrument from the System View Window, **LAST INSTRMT** will toggle between that newly selected instrument and the one that was last in the Interactive Instrument Window.
ENABLE OUTPUTS	Enables the outputs of *all* system instruments that generate outputs.
DISABLE OUTPUTS	Disables the outputs of *all* system instruments that generate outputs. If you exit the Soft Front Panel after having entered it from DOS, the outputs of all the instruments will be disabled. Thus, the instruments will always ''wake up'' disabled when you enter the Soft Front Panel from DOS. This allows you to view the state of an instrument before it generates any output.
Store Program	Invokes the Store Program Menu, as described further in this chapter.
EXIT	Exits you from the PC Instruments Soft Front Panel and returns you to DOS or BASIC. The operations that occur when you exit depend on where you entered the Soft Front from. Refer to ''Exiting the Soft Front Panel'' at the end of this chapter for complete details.
etc	Advances you to the first row (Main 1) of Main Menu softkeys.

Store State Menu

You obtain this menu by pressing the **Store State** softkey on the Main 1 Menu:

The Store State Menu is a lower-level menu that allows you to specify a filename for storing the current state of the system; i.e., the front panel settings and the rear panel information for every instrument in the System View Window. This menu returns you to the Main 1 Menu.

Cancel Command		CHOOSE DIR	CHOOSE FILE	SAVE THE FILE

Cancel Command

Automatically returns you to the Main Menu without accepting any new entry. No modifications to the system are made.

CHOOSE DIR

Invokes the Keyboard Entry Menu (refer to that menu description further in this chapter), which allows you to specify a file directory or pathname for organizing state files on the diskette. You can type a new directory name or modify an existing one. The default is the current directory. Both the directory and filenames will appear in the Status Window. When your entry is complete, press the [Enter] key. You will then be returned to the Store State Menu.

CHOOSE FILE

Invokes the Keyboard Entry Menu (refer to that menu description further in the chapter), which allows you to specify a filename for storing the state information. There is no default filename; you must specify one. You can either type a new filename or modify an existing one. Both the directory and filename will appear in the System Status Window. (State files have the extension [.HPC]). Press the [Enter] key when your entry is complete. This will return you to the Store State menu. Then, press **SAVE THE FILE** to store the file.

SAVE THE FILE

Stores the specified state file on diskette (under the default or specified directory) and returns you to the Main 1 Menu.

Recall State Menu

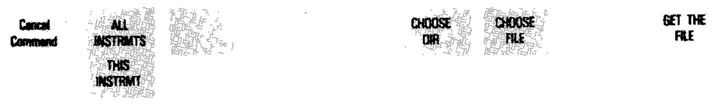
You obtain this menu by pressing **Recall State** on the Main 1 Menu. Most of the Recall State Menu softkeys are identical to those previously described for the Store State Menu.

The Recall State Menu is a lower-level menu that allows you to specify a previously stored state file. This allows you to update either the state of every instrument in the system or just the state of the instrument currently in the Interactive Instrument Window.

The instrument(s) in the file being recalled must have the same model, bus address, interface number and secondary address (if needed) as the instrument(s) currently in the system. If this information does not match for at least one of the instruments (if you are recalling the state of all instruments) or the instrument in the Interactive Instrument Window (if you are recalling only the state of that instrument), you will get an error message.

NOTE

If you entered the Soft Front Panel from DOS, the labels that currently appear in the System View Window will be overwritten by (changed to) those in the file you are recalling. This is **not** *true if you entered the Soft Front Panel from BASIC (refer to Chapter 5).*



Cancel Command Same as in Store State Menu.

ALL INSTRMTS This softkey's legend toggles between **ALL INSTRMTS** and **THIS INSTRMT** . Press **ALL INSTRMTS** to recall the state of all the instruments in the system. Press **THIS INSTRMT** to recall only the state of the instrument currently in the Interactive Instrument Window. When you subsequently press **GET THE FILE** , the appropriate windows will be updated. They are the Interactive Instrument Window for **THIS INSTRMT** and both the Interactive Instrument and System View windows for **ALL INSTRMTS** .

CHOOSE DIR Invokes the Keyboard Entry Menu, which allows you to specify a file directory or pathname for accessing the state file being recalled. You must specify an existing directory name. The default is the current directory. When your entry is complete, press the [Enter] key. You will then be returned to the Recall State Menu.

CHOOSE FILE Invokes the Keyboard Entry Menu, which allows you to specify a filename to obtain the states of all instruments in the system or just the instrument in the Interactive Instrument Window. There is no default filename; you must specify an existing state filename. Both the directory and filename will appear in the Status Area. After specifying the file, press the [Enter] key. You will be returned to the Recall State Menu. Then, press **GET THE FILE** to load the file.

GET THE FILE Retrieves the specified file from the diskette (under the specified or default directory), updates the Soft Front Panel display, and returns you to the Main 1 Menu.

Store Program Menu

You obtain this menu by pressing **Store Program** on the Main 2 Menu.

The Store Program Menu is a lower-level menu that allows you to create a Program Shell and specify a filename for storing it on diskette. You will load this Program Shell file after you have exited the Soft Front Panel and are ready to automate your test by writing a BASIC application program. This menu returns you to the Main 2 Menu.

Cancel Command Same as in the Store State Menu.

CHOOSE DIR Invokes the Keyboard Entry Menu (refer to that menu description further in this chapter), which allows you to specify a file directory or pathname for organizing state files on diskette. Specify the directory that contains the same state file as will be used by this Program Shell. When your entry is complete, press the [Enter] key. You will then be returned to the Store Program Menu.

CHOOSE FILE Invokes the Keyboard Entry Menu for storing the generated Program Shell. There is no default filename; you must specify one. You can either type a new filename or modify an existing one. Both the directory and filenames will appear in the Status Area. Program Shell files have a the extension [.BAS]. When your entry is complete, press the [Enter] key. You will be returned to the Store Program Menu. Then, press **SAVE THE PROGRAM** to store the program file.

SAVE THE PROGRAM Stores the generated Program Shell in the specified file (under the default or specified directory) on diskette and returns you to the Main 2 Menu. The Program Shell uses each instrument's label, model, bus address, interface number, and secondary address (where needed). When you develop an application program using this Program Shell (as explained in Chapter 5), you must make sure that all this information agrees exactly with the system as it will operate under that application program.

Keyboard Entry Menu

The Keyboard Entry Menu appears whenever you are required to make an input from the keyboard (e.g., when you are entering a numeric or alphanumeric field value, or specifying a filename).

When you are done typing your entry, press [Enter] to return to the menu from where you began the keyboard entry. If the Keyboard Entry Menu was invoked because you pressed the **New Value** softkey of the Numeric Modification Menu (refer to that menu), then you will be returned to the menu from where you began the numeric modification. If you make an invalid entry, an error message will appear in the Status Areas.

NOTE

The mouse and the Softkey Area of the Soft Front Panel does not operate with this menu. Use the corresponding keyboard controls.

Cancel Command

Returns you to the menu from where you entered, without accepting any keyboard entry. You can use **Cancel Command** any time before pressing the keyboard [Enter] key . If you press **Cancel Command** while in the middle of an entry, the original field will be restored.

Numeric Modification Menu

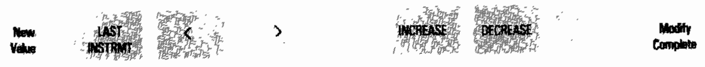
You enter this menu by pointing to and selecting the numeric field that you wish to modify.

When you select a numeric field, it darkens to indicate that you can use the numeric modification softkeys to modify the field. Move the cursor to the digit to be modified by pressing the **<** or **>** softkey. Then, increase or decrease the value of the selected digit with the **INCREASE** or **DECREASE** softkey. As you change a value, you will see the appropriate instrument field increase or decrease.

If you wish to enter a new value, press the **New Value** softkey to invoke the Keyboard Entry Menu. After completing the new value, press the **Modify Complete** softkey. This softkey always returns you to the menu you were using before you entered numeric modification.

NOTE

The mouse and the Softkey Area of the Soft Front Panel does not operate with this menu. Use the corresponding keyboard controls.



New Value Invokes the Keyboard Entry menu so you can enter a completely new value (refer to the previous Keyboard Entry Menu description).

LAST INSTRMT Performs the same function described under the Main 2 Menu. This softkey is useful if you want to change a numeric value of a stimulus instrument while interacting with the response instrument.

< Moves the digit position to the left, allowing you to modify this selected digit.

> Moves the digit position to the right, allowing you to modify this selected digit.

INCREASE Increments the selected digit by one and includes the carry digit (where applicable). For example,

```
1 8 0
  |------ Selected digit
1 9 0
```

DECREASE Decrements the selected digit by one and works with a borrow (where applicable). For example,

```
1 9 0
  |------ Selected digit
1 8 0
```

NOTE

If you are changing the value of the least-significant digit (LSD), remember that an instrument can only respond to changes equal to or greater than its resolution. In the above examples, assume that the numbers represent millivolts and that the instrument's resolution is 8 millivolts. As you **INCREASE** or **DECREASE** the LSD, the instrument will not respond until you have changed the LSD by at least 8 millivolts.

Modify Complete Enters the modified value and returns you to the menu from where you invoked the Numeric Modify Menu. The graphics cursor is released so you can again move it to any position on the screen.

Table 4-1. Summary of Menu Softkeys

SOFTKEY	MENU WHERE USED						SOFTKEY FUNCTION
	Main	Store State	Recall State	Store Program	Numeric Modify	Keyboard Entry	
ALL INSTRMTS			x				Recall states of all system instruments
Cancel Command		x	x	x		x	Ignore all inputs & return to previous menu
CHOOSE DIR		x	x	x			Enter directory (pathname)
CHOOSE FILE		x	x	x			Enter filename
DECREASE					x		Decrement existing value by one
DISABLE OUTPUTS	x						Disable all system instrument outputs
ENABLE OUTPUTS	x						Enable all system instrument outputs
etc	x						Access next set of Main Menu softkeys
EXIT	x						Return to the operating system
FRONT PANEL	x						Operate an instrument
GET THE FILE			x				Recall a(.HPC) state file
INCREASE					x		Increment the existing value by one
LAST INSTRMT	x				x		View the previous instrument
Modify Complete					x		Make the change and return to previous menu
New Value					x		Invoke the Keyboard Entry Menu
PRINT SCREEN	x						Obtain a hard copy of the screen
REAR PANEL	x						Label an instrument; examine its configuration
Recall State	x						Invoke the Recall State Menu
ROLL DWN SYS VIEW	x						Scroll the System View Window down
ROLL UP SYS VIEW	x						Scroll the System View Window up
SAVE THE FILE		x					Store a (.HPC) Program Shell file
SAVE THE Program				x			Store a (.BAS) Program Shell file
Store Program	x						Invoke the Store Program Menu
Store State	x						Invoke the Store State Menu
THIS INSTRMT			x				Recall state of specified instrument
>					x		Select digit by moving right one place
<					x		Select digit by moving left one place

Exiting the Soft Front Panel

If you entered the Soft Front Panel from DOS, the present state of all the instruments in the system is automatically stored in a filenamed ''HPSTATE.HPC''. The next time you run the Soft Front Panel, this state file is automatically recalled (accessed) so that all the instruments will appear just the way you left them. This ''HPSTATE.HPC'' file is not accessed when you enter the Soft Front Panel from BASIC.

NOTE

Your instruments come from the factory set to certain values called factory defaults. Once you make changes to each instrument's front panel, these factory defaults will be lost. To regain these values, rename the HPSTATE.HPC file as previously described in this chapter.

Before you exit the Soft Front Panel, be sure you have done the following if you plan to write a BASIC program to control your instruments (as described in Chapter 5):

1. Generated and saved a Program Shell for your particular application program. This Program Shell is needed as the starting point of your program. The instrument labels in the System View Window are saved with the Program Shell. Configuration information (model number, interface number, bus address and secondary address) is also saved with the Program Shell. This information must agree with your hardware configuration when you run your complete application program.

2. Stored one or more State files so that you can initialize your instrument(s) from your BASIC program. It is a good idea to write down the name of your State file so you can enter it into your program

correctly. As with the Program Shell, configuration information and instrument labels are also saved in the state file. Later on, when you initialize instruments from your application program, this information must agree with the hardware setup.

You are now ready to write a BASIC application program using the Program Shell and state files that were saved using the Soft Front Panel. Proceed to the next chapter for complete programming details.

5 Programmed Instrument Control

How this Chapter is Organized

This chapter explains how to develop your application program from the Program Shell. You are given a brief outline of the procedure to follow for developing your application program, followed by a short example of programmed instrument control. After the example, the HP PC Instruments System statements and program development are described in detail. The chapter concludes with some information on how to evaluate the performance of your application program.

General Information

In order to write a BASIC program to control your instruments, you should already know how to write programs in BASIC. This chapter explains how to develop and run your own application program. You must start by first running the Soft Front Panel, which allows you to:

- Assign a unique label to each instrument in the system.
- Store the state of the instruments in one or more files.
- Generate and save a Program Shell for a specific hardware configuration.

The label that you gave an instrument while using the Soft Front Panel is also the label you must use when programming that instrument. State files are required for the INITIALIZE.SYSTEM and INITIALIZE statements described later in this chapter. The Program Shell is required to allow you to communicate with your instruments. You will write your application program, *beginning with line 1000 of the Program Shell.*

Outline of Program Development

After running the Soft Front Panel (as explained in Chapter 4) you are ready to automate your test. Do this by developing a complete application program that includes the Program Shell. Briefly, the procedure for doing this is:

- Get into BASIC by entering PCIBAS from DOS.
- Load the Program Shell
- Insert your application program segment (the CALLs to the PC Instruments Statement Library for BASIC and the BASIC commands) beginning at line 1000 to form a complete application program.
- Save your application program
- Run your application program

A detailed description of this procedure, along with a description of the Program Shell and the program statements, follows the description of the PC Instruments Statement Library for BASIC.

Example of Programmed Instrument Control

This example illustrates programmed control by expanding the example given in Chapter 4 and the final program performs the same control functions as that example.

Purpose of the Example

This example shows how to control your instruments from an application program after you have first gone through the Soft Front Panel (manual mode). The new example incorporates a program using the state file (STATE1) and Program Shell file (PROGRAM1) that you created in the manual mode. (These names are exclusively for this example; you do not have to use the names ''STATE1'' and ''PROGRAM1'' in your own programs).

Hardware Used

The test setup for this example is identical to that of the previous example in Chapter 4 (see Figure 4-2). It uses the Dual Voltage DAC and the Digital Multimeter.

Example Test Procedure

Figure 5-1 shows the steps taken in this example. The Program Shell that you load and develop into a complete application program is ''PROGRAM1''. PROGRAM1 contains the following information for the Dual Voltage DAC and Digital Multimeter:

User Label	Model No. (Self-Id)	Bus Address	Interface Number	Secondary Address
TESTV.IN	61012A	01	01	A
TEMPSET	61012A	01	01	B
TESTV.OUT	61013A	00	01	—

This same hardware must be in the system, configured and labeled as indicated, or you will get an error message. The complete program for the programmed control example is shown in Figure 5-2.

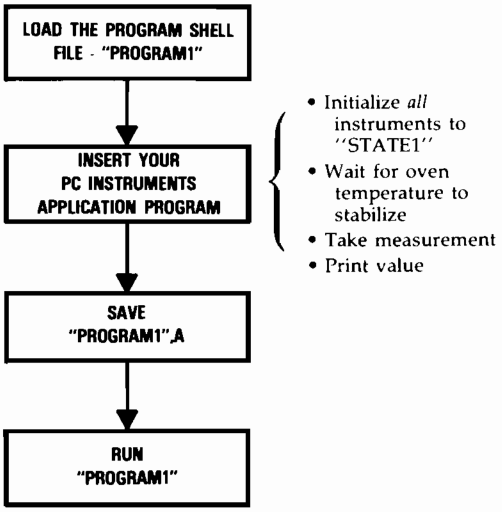

LOAD THE PROGRAM SHELL FILE - "PROGRAM1"
INSERT YOUR PC INSTRUMENTS APPLICATION PROGRAM
SAVE "PROGRAM1",A
RUN "PROGRAM1"
• Initialize *all* instruments to ''STATE1''
• Wait for oven temperature to stabilize
• Take measurement
• Print value

Figure 5-1. Procedure for Programmed Instrument Control Example

```
   1  < Program Shell>
 999

1000  ' User application program code starts at this line
1010  '
1020  FILE$ = ''STATE1''
1030  '
1040  ' Initializes all the instruments in the system to the
1050  ' state file specified by FILE$ and enables the
      ' outputs
1060  ' TESTV.IN and TEMPSET because they were stored
      ' ENABLED
1070  CALL INITIALIZE.SYSTEM (FILE$)
1080  '
1090  ' Wait for the oven temperature to stabilize
1100  PRINT ''Press any key to continue''
1110  '
1120  A$ = INKEY$ : IF A$= ''''THEN 1120
1130  '
1140  ' Take a reading with the Digital Multimeter
1150  CALL MEASURE (TESTV.OUT,VALUE)
1160  '
1170  ' Print the returned value
1180  PRINT VALUE
1190  END
```

Figure 5-2. Program Listing for Programmed Instrument Control Example

PC Instruments Statement Library for BASIC

The PC Instruments Statement Library consists of system statements (that perform system operations) and instrument-specific statements (that perform operations only to the named instruments). The PC Instruments Quick Reference Guide includes a complete listing of the PC Instruments programming statements˙. Only the system statements are described in this chapter. The instrument- specific statements are described in Chapter 5 of the applicable Instrument Owner's Guide.

The PC Instruments library statements are actually supplied routines that you "CALL" from your BASIC application. The functions these statements perform are equivalent to the functions performed when you manually control the instruments from the Soft Front Panel. By inserting these statements into your BASIC Program Shell and running the resultant program, you can perform these functions automatically. This will be fully discussed under "Detailed Program Development".

NOTE

Make sure that the instruments currently in the system match those that were in the system when you generated the Program Shell. These instruments must have the same bus address, model number, interface number and secondary address (if needed).

Notation Conventions

The following notation conventions are used in this guide, and throughout the Instrument Owner's Guides, for the programming statement descriptions. These conventions are the same as those used in most BASIC programming manuals:

CAPITAL LETTERS: Words in capital letters are keywords and must be entered exactly as they are shown.

lower-case letters: Words shown in *italicized,* lower-case letters are words that you must supply.

[square brackets]: Items in square brackets are optional.

punctuation: All punctuation (except square brackets), such as commas, semicolons, parentheses, etc., must be included exactly as shown in the text.

Parameter Types

The numeric variables used in your application program must represent real, single precision numbers.

System Statements

The system programming statements are:

```
INITIALIZE.SYSTEM
ENABLE.SYSTEM
DISABLE.SYSTEM
PANELS
DEF.ERR
GET.MEM
LD.FILE
```

Unlike instrument-specific statements, system statements do not perform functions on named instruments. Instead, they either perform functions on all output instruments (such as ENABLE.SYSTEM) or perform functions that are entirely independent of any instrument (such as PANELS). The system statements are explained in the order listed above.

INITIALIZE.SYSTEM(*statefile$*) - This statement initializes *all* system instruments (including the instrument outputs) to the state specified in *statefile$*. The parameter used in the initialization statement *must* be a string variable. *Statefile$* is the string variable equal to the state filename. Before using this statement, make sure that you have stored *statefile$* via the Soft Front Panel. Before you use this statement, first assign the state file to a string variable, such as:

```
FILE$ = "EXAMPLE"
CALL INITIALIZE.SYSTEM(FILE$)
```

In this case, "EXAMPLE" is the name you specified in the Store State Menu when you pressed the **CHOOSE FILE** softkey on the Soft Front Panel. This string variable assignment is necessary whenever you need to specify a state filename in the system initialization statement. If you only want to initialize one instrument, use the instrument-specific INITIALIZE statement (refer to the Instrument Owner's Guide).

Proper initialization by this statement occurs only if each instrument specified in *statefile$* is currently in the system and is labeled and configured (model number, bus address, interface number, and secondary address) exactly as when the Program Shell was generated. If this information does not match for *every* instrument in the system, you will get an ERROR message.

NOTE

If you add instruments to the system ***after*** *you generate the Program Shell, you will not get an ERROR message. However, you will* ***not be able to initialize or program the added instruments.***

If you do not supply an initialization statement (with a specified state file), the instruments will be set to their factory default (as-shipped) state.

ENABLE.SYSTEM - Enables the outputs of all the output instruments in the system. This statement is equivalent to the ENABLE OUTPUTS softkey of the Soft Front Panel (refer to Chapter 4).

DISABLE.SYSTEM - Disables the outputs of all the output instruments in the system . This statement is equivalent to the DISABLE OUTPUTS softkey of the Soft Front Panel (refer to Chapter 4).

PANELS - This statement allows you to enter the Soft Front Panel from BASIC, either directly from the keyboard or from your application program. PANELS is used mainly for debugging. It enables you to track down errors that might have occurred during the program mode. PANELS allows you to view how the system appears at the time when the Soft Front Panel is entered.

a. *Entering the Soft Front Panel From the Keyboard.* To enter the Soft Front Panel directly from the keyboard while in BASIC, make sure you have first run the Program Shell at least once to initialize variables. Also, you must have the Soft Panels files resident on a diskette drive. After you have ensured this, type:

CALL PANELS
and press Enter

When you are finished and exit the Soft Front Panel, you will be returned to the BASIC prompt.

b. *Entering the Soft Front Panel From a Program.*To enter the Soft Front Panel from your application program, insert the CALL PANELS statement into the program where you want program execution to halt and the Soft Front Panel to appear. When CALL PANELS takes you to the Soft Front Panel, you can check the setting on each instrument's front panel and the configuration information on its rear panel. You can also change instrument settings at this time. When you exit the Soft Front Panel, execution will continue from the next statement in your BASIC program.

NOTE

The instruments that you are controlling from BASIC will be effected by any changes you make to them while you are in the Soft Front Panel. Also, any changes made to a file accessed via the **Recall State** *softkey will be saved if you restore that file via the* **Store State** *softkey. However, these changes are not saved automatically in the HPSTATE.HPC file when you enter the Soft Front Panel from, and return to, BASIC.*

---

c. *BASIC Entry Vs. DOS Entry.* How you enter the Soft Front Panel (whether from DOS or from BASIC) makes a difference in how some operations are performed. These differences are summarized in Table 5-1. Aside from these differences, the Soft Front Panel works the same whether it is entered from DOS or CALLed from BASIC (refer to Chapter 4 of this guide for complete details on the Soft Front Panel).

DEF.ERR(PCIB).ERR,PCIB.ERR$,PCIB.NAME$, PCIB.GLBERR) · This statement must be used after an ERASE, CLEAR, or CHAIN statement is executed. DEF.ERR defines the location of PC Instruments error variables PCIB.ERR, PCIB.ERR$, PCIB.NAME$ and PCIB.GLBERR. Refer to ''PC Instruments Error Handling'' further in this chapter (additional information is also included under ''Chaining Programs'').

NOTE

The next two statements require you to have a good understanding of your system before using them. GET.MEM and LD.FILE are general-purpose statements used to ensure proper memory management. The statements are for allocating a portion of memory that is outside of BASIC's workspace and is unoccupied by other software (free memory).

GET.MEM*(num.bytes,block.seg)* · Reserves a block of free memory for your general use. You specify the size of the block to be reserved (in bytes) by the variable *num.bytes,* which must be entered in a separate statement that preceeds GET.MEM. You are given the segment address of the reserved block of memory in the return variable *block.seg*. If you specify zero number of bytes (*num.bytes* = 0), the *block.seg* address returned is where free memory begins. This enables you to use the BASIC DEF SEG and BLOAD statements for loading memory image files (refer to the LD.FILE statement description).

Table 5-1. Summary of DOS/BASIC Differences

Origin of Soft Front Panel Entry		
Action	**From DOS**	**From BASIC**
Entering the Soft Front Panel	HPSTATE. HPC is accessed; all the instruments "wake up" in the state they were left in (determined by HPSTATE.HPC) or their hardware defaults if HPSTATE.HPC is not found. All outputs are set DISABLED.	HPSTATE is not accessed; the instruments are in the state set by the BASIC program
Recall State Softkey	The instrument labels are recalled along with the state information (the labels in the System View Window are updated). The outputs are set DISABLED (even if the state file has them enabled).	The instrument labels are not recalled along with the state information. The labels in the System View Window *must* be the same as those in the file being recalled. The outputs are set the way they are stored in the state file.
Exiting the Soft Front Panel	HPSTATE.HPC is updated and stored. You are returned to DOS. The outputs are set DISABLED.	HPSTATE.HPC is not updated or stored. No state file is stored unless specified. You are either returned to the BASIC prompt (if you entered the Soft Front Panel directly from the keyboard) or to your program (execution resumes with the statement following the CALL PANELS statement).

LD.FILE(*file$,load.seg*) - Loads one of your own files into the next unoccupied portion of free memory. You can load an .EXE file (with correct relocation) or any non .EXE file directly into memory without alteration. You specify the file being loaded by the variable *file$*. You will be given the segment address where the file was loaded into free memory in return variable *load.seg*. When you use this statement to load files, the PC Instruments System software keeps track of free memory for you. You can also load a memory image file (not a .COM or .EXE file), using GET.MEM to find the starting address of free memory, followed by the BASIC DEF SEG statement and BLOAD command. This is not recommended, however, unless you are familiar with these BASIC statements and with your system.

NOTE

If you use a DEF SEG statement in your program, you must restore it by DEF SEG = PCIB.SEG before any subsequent PC Instruments CALL statement in your program.

Detailed Program Development

This section describes the Program Shell and explains how you develop a complete application program with it. Before you do so, you should be familiar with Microsoft Advanced BASIC (BASICA). Refer to your BASIC Manual for complete details about specific statements and commands .

What is the Program Shell?

The Program Shell consists of BASIC program lines that perform initialization chores to allow you to communicate with your instruments. You generated the Program Shell and stored it on a work diskette (or fixed disk) from the Store Program Menu when you exited the Soft Front Panel (refer to the Store Program Menu in Chapter 4).

Remember that the Program Shell contains the label (user-defined or default name), model number, bus address, interface number, and secondary address (if needed) of each instrument that was in the system when you generated the Program Shell. You must make sure these same instruments are in the system (and have the same label, model number, bus address, interface number and secondary address) before you run your program. If you have altered this configuration in any way, you cannot use this Program Shell and you will be informed of the error when you try to run your program.

In order to control your instruments, you must perform the following steps, each of which is described in detail.

1. Load the Program Shell into the BASIC workspace.
2. Add your application program lines.
3. Save the completed application program on a work diskette.
4. Run your application program.

NOTE

The instructions assume that you are loading from and saving to the default drive. However, you can use any drive by including the approriate drive designator and directory path with your instruction.

Loading the Program Shell

Before beginning, be sure you have all the PC Instruments software resident on fixed disk, or have your work diskette arranged as described in Chapter 3.

The first step is to enter the BASIC programming environment by typing:

```
A> PCIBAS
```

and pressing Enter

NOTE

It is important to enter BASIC this way to ensure that the operating system allocates the extra memory required by the PC Instruments Software.

Next, load the Program Shell by typing:

LOAD "Filename"
and pressing [Enter]

"Filename" refers to the name of the file you used when you stored the Program Shell on the work diskette or fixed disk. It is the same name you assigned in manual mode (via the Soft Front Panel) when you pressed the **CHOOSE FILE** softkey of the Store Program Menu.

After you have loaded and listed the Program Shell, a program similar to Figure 5-3 will appear on the screen (each Program Shell differs slightly depending on what instruments were in the system when you generated the Program Shell).

The Program Shell shown in Figure 5-3 occupies lines 1 through 999. It performs all the system initialization and sets up the system so that you can access state files from your application program and control your instruments. To prevent any system malfunction (errors, incorrect data), it is recommended that you *not tamper with the Program Shell.*

If you use the BASIC "RENUM" command, you will renumber the entire program (including the Program Shell). Therefore, any references made in this chapter to specific line numbers could be inaccurate. To keep the Program Shell line numbers, renumber *only* your own program by using the command "RENUM,1000,1000."

NOTE

If you should accidently renumber the entire program, you can recover with the technique described under "Merging Programs."

The DEFINE statements (beginning on line 124 of Figure 5-3) identify each instrument that was in the system when you generated the Program Shell. Each statement contains an instrument's label and hardware configuration information. You *must* use these same labels and same hardware configurations whenever you run the application developed from this Program Shell.

If you *remove* instruments from the system after generating and storing the Program Shell, you will get an error message. To correct the error, you can either generate a new Program Shell or add the instrument(s) you previously removed and rerun your program. If you do generate a new Program Shell, you can modify your application program code and MERGE it with the new Program Shell, as explained under "Merging Programs."

Inserting your Application Program Lines

NOTE

The PC Instruments software "looks for" your application lines at line 1000. Be sure always to have something (even if only a comment) at line 1000.

You write the application program and insert it beginning at line 1000 of the Program Shell. The program lines you write are a combination of PC Instruments statements with BASIC keywords and BASIC programming statements. You can include the commands for the optional HP-IB interface. Remember to supply the BASIC "END" statement for your program.

```
1 DEF SEG:CLEAR ,&HFE00:GOTO 4 'Begin PCIB Program Shell
2 GOTO 1000 ' User program
3 GOTO 900 ' Error handling
4 I=&HFE00 ' Copyright Hewlett-Packard 1984,1985
5 PCIB.DIR$=ENVIRON$("PCIB")
6 I$=PCIB.DIR$+"\PCIBILC.BLD"
7 BLOAD I$,&HFE00
8 CALL I(PCIB.DIR$,I%,J%):PCIB.SEG=I%
9 IF J%=0 THEN GOTO 13
10 PRINT "Unable to load.";
11 PRINT "   (Error #";J%;")"
12 END
13 '
14 DEF SEG=PCIB.SEG:O.S=5:C.S=10:I.V=15
15 I.C=20:L.P=25:LD.FILE=30
16 GET.MEM=35:L.S=40:PANELS=45:DEF.ERR=50
17 PCIB.ERR$=STRING$(64,32) : PCIB.NAME$=STRING$(16,32)
18 CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB.GLBERR) : PCIB.BASERR=255
19 ON ERROR GOTO 3
20 J=-1
21 I$=PCIB.DIR$+"\PCIB.SYN"
22 CALL O.S(I$)
23 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
24 I=0
25 CALL I.V(I,PRT.ON,PRT.OFF,READ.SELFID,DEFINE)
26 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
27 CALL I.V(I,INITIALIZE.SYSTEM,ENABLE.SYSTEM,DISABLE.SYSTEM,INITIALIZE)
28 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
29 CALL I.V(I,MEASURE,OUTPUT,START,HALT)
30 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
31 CALL I.V(I,ENABLE.INT.TRIGGER,DISABLE.INT.TRIGGER,ENABLE.OUTPUT,DISABLE.OUTP
UT)
32 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
33 CALL I.V(I,CHECK.DONE,GET.STATUS,SET.FUNCTION,SET.RANGE)
34 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
35 CALL I.V(I,SET.MODE,WRITE.CAL,READ.CAL,STORE.CAL)
36 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
37 I=1
38 CALL I.V(I,SET.GATETIME,SET.SAMPLES,SET.SLOPE,SET.SOURCE)
39 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
40 CALL I.C(I,FREQUENCY,AUTO.FREQ,PERIOD,AUTO.PER)
41 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
42 CALL I.C(I,INTERVAL,RATIO,TOTALIZE,R100MILLI)
43 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
44 CALL I.C(I,R1,R10,R100,R1KILO)
45 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
46 CALL I.C(I,R10MEGA,R100MEGA,CHAN.A,CHAN.B)
47 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
48 CALL I.C(I,POSITIVE,NEGATIVE,COMN,SEPARATE)
49 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
50 I=2
51 I=3
52 CALL I.V(I,ZERO.OHMS,SET.SPEED,J,J)
53 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
54 CALL I.C(I,DCVOLTS,ACVOLTS,OHMS,R200MILLI)
55 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
56 CALL I.C(I,R2,R20,R200,R2KILO)
57 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
58 CALL I.C(I,R20KILO,R200KILO,R2MEGA,R20MEGA)
59 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
60 CALL I.C(I,AUTOM,R2.5,R12.5,J)
61 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
62 I=4
63 CALL I.V(I,OUTPUT.NO.WAIT,ENABLE.HANDSHAKE,DISABLE.HANDSHAKE,SET.THRESHOLD)
64 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
65 CALL I.V(I,SET.START.BIT,SET.NUM.BITS,SET.INVERTER,J)
66 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
67 CALL I.C(I,SIGNED,UNSIGNED,OC,TTL)
68 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
69 I=5
70 CALL I.C(I,R1,R5,R10,J)
71 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
72 I=6
73 CALL I.V(I,SET.FREQUENCY,SET.AMPLITUDE,SET.OFFSET,SET.SYMMETRY)
74 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
75 CALL I.V(I,SET.BURST.COUNT,J,J,J)
76 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
77 CALL I.C(I,SINE,SQUARE,TRIANGLE,CONTINUOUS)
78 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
79 CALL I.C(I,GATED,BURST,J,J)
80 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
81 I=7
82 CALL I.V(I,AUTOSCOPE,CALIBRATE,SET.SENSITIVITY,SET.VERT.OFFSET)
83 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
84 CALL I.V(I,SET.COUPLING,SET.POLARITY,SET.SWEEPSPEED,SET.DELAY)
85 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
86 CALL I.V(I,SET.TRIG.SOURCE,SET.TRIG.SLOPE,SET.TRIG.LEVEL,SET.TRIG.MODE)
87 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
88 CALL I.V(I,GET.SINGLE.WF,GET.TWO.WF,GET.VERT.INFO,GET.TIMEBASE.INFO)
89 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
90 CALL I.V(I,GET.TRIG.INFO,CALC.WFVOLT,CALC.WFTIME,CALC.WF.STATS)
```

Figure 5-3. A Sample Program Shell (1 of 2)

```
91 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
92 CALL I.V(I,CALC.RISETIME,CALC.FALLTIME,CALC.PERIOD,CALC.FREQUENCY)
93 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
94 CALL I.V(I,CALC.PLUSWIDTH,CALC.MINUSWIDTH,CALC.OVERSHOOT,CALC.PRESHOOT)
95 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
96 CALL I.V(I,CALC.PK.TO.PK,SET.TIMEOUT,J,J)
97 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
98 CALL I.C(I,R10NANO,R100NANO,R1MICRO,R10MICRO)
99 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
100 CALL I.C(I,R100MICRO,R1MILLI,R10MILLI,R100MILLI)
101 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
102 CALL I.C(I,R1,R10,R20NANO,R200NANO)
103 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
104 CALL I.C(I,R2MICRO,R20MICRO,R200MICRO,R2MILLI)
105 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
106 CALL I.C(I,R20MILLI,R200MILLI,R2,R20)
107 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
108 CALL I.C(I,R50NANO,R500NANO,R5MICRO,R50MICRO)
109 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
110 CALL I.C(I,R500MICRO,R5MILLI,R50MILLI,R500MILLI)
111 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
112 CALL I.C(I,R5,R50,CHAN.A,CHAN.B)
113 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
114 CALL I.C(I,EXTERNAL,POSITIVE,NEGATIVE,AC)
115 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
116 CALL I.C(I,DC,GROUND,TRIGGERED,AUTO.TRIG)
117 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
118 CALL I.C(I,AUTO.LEVEL,X1,X10,J)
119 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
120 CALL C.S
121 I$=PCIB.DIR$+"\PCIB.PLD"
122 CALL L.P(I$)
123 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
124 I$="Dig.in.01";I=4:J=0:K=1:L=1
125 CALL DEFINE(Dig.in.01,I$,I,J,K,L)
126 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
127 I$="VDAC.A.01":I=5:J=0:K=2:L=1
128 CALL DEFINE(VDAC.A.01,I$,I,J,K,L)
129 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
130 I$="Counter.01":I=1:J=0:K=3:L=1
131 CALL DEFINE(Counter.01,I$,I,J,K,L)
132 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
133 I$="DMM.01":I=3:J=0:K=4:L=1
134 CALL DEFINE(DMM.01,I$,I,J,K,L)
135 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
136 I$="Func.Gen.01":I=6:J=0:K=5:L=1
137 CALL DEFINE(Func.Gen.01,I$,I,J,K,L)
138 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
139 I$="Relay.Mux.01":I=2:J=0:K=6:L=1
140 CALL DEFINE(Relay.Mux.01,I$,I,J,K,L)
141 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
142 I$="Scope.01":I=7:J=0:K=7:L=1
143 CALL DEFINE(Scope.01,I$,I,J,K,L)
144 IF PCIB.ERR<>0 THEN ERROR PCIB.BASERR
800 I$=PCIB.DIR$+"\PANELS.EXE"
801 CALL L.S(I$)
899 GOTO 2
900 IF ERR=PCIB.BASERR THEN GOTO 903
901 PRINT "BASIC error #";ERR;" occurred in line ";ERL
902 STOP
903 TMPERR=PCIB.ERR:IF TMPERR=0 THEN TMPERR=PCIB.GLBERR
904 PRINT "PC Instrument error #";TMPERR;" detected at line ";ERL
905 PRINT "Error: ";PCIB.ERR$
906 IF LEFT$(PCIB.NAME$,1)<>CHR$(0) THEN PRINT "Instrument: ";PCIB.NAME$
907 STOP
908 COMMON PCIB.DIR$,PCIB.SEG
909 COMMON LD.FILE,GET.MEM,PANELS,DEF.ERR
910 COMMON PCIB.BASERR,PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB.GLBERR
911 COMMON PRT.ON,PRT.OFF,READ.SELFID,DEFINE,INITIALIZE.SYSTEM,ENABLE.SYSTEM,DI
SABLE.SYSTEM,INITIALIZE,MEASURE,OUTPUT,START,HALT,ENABLE.INT.TRIGGER,DISABLE.IN
T.TRIGGER,ENABLE.OUTPUT,DISABLE.OUTPUT,CHECK.DONE,GET.STATUS,SET.FUNCTION
912 COMMON SET.RANGE,SET.MODE,WRITE.CAL,READ.CAL,STORE.CAL,SET.GATETIME,SET.SAM
PLES,SET.SLOPE,SET.SOURCE,ZERO.OHMS,SET.SPEED,OUTPUT.NO.WAIT,ENABLE.HANDSHAKE,D
ISABLE.HANDSHAKE,SET.THRESHOLD,SET.START.BIT,SET.NUM.BITS,SET.INVERTER
913 COMMON SET.FREQUENCY,SET.AMPLITUDE,SET.OFFSET,SET.SYMMETRY,SET.BURST.COUNT,
AUTOSCOPE,CALIBRATE,SET.SENSITIVITY,SET.VERT.OFFSET,SET.COUPLING,SET.POLARITY,S
ET.SWEEPSPEED,SET.DELAY,SET.TRIG.SOURCE,SET.TRIG.SLOPE,SET.TRIG.LEVEL
914 COMMON SET.TRIG.MODE,GET.SINGLE.WF,GET.TWO.WF,GET.VERT.INFO,GET.TIMEBASE.IN
FO,GET.TRIG.INFO,CALC.WFVOLT,CALC.WFTIME,CALC.WF.STATS,CALC.RISETIME,CALC.FALLT
IME,CALC.PERIOD,CALC.FREQUENCY,CALC.PLUSWIDTH,CALC.MINUSWIDTH
915 COMMON CALC.OVERSHOOT,CALC.PRESHOOT,CALC.PK.TO.PK,SET.TIMEOUT
916 COMMON FREQUENCY,AUTO.FREQ,PERIOD,AUTO.PER,INTERVAL,RATIO,TOTALIZE,R100MILL
I,R1,R10,R100,R1KILO,R10MEGA,R100MEGA,CHAN.A,CHAN.B,POSITIVE,NEGATIVE,COMN,SEPA
RATE,DCVOLTS,ACVOLTS,OHMS,R200MILLI,R2,R20,R200,R2KILO,R20KILO,R200KILO
917 COMMON R2MEGA,R20MEGA,AUTOM,R2.5,R12.5,SIGNED,UNSIGNED,OC,TTL,R1,R5,R10,SIN
E,SQUARE,TRIANGLE,CONTINUOUS,GATED,BURST,R10NANO,R100NANO,R1MICRO,R10MICRO,R100
MICRO,R1MILLI,R10MILLI,R100MILLI,R1,R10,R20NANO,R200NANO,R2MICRO,R20MICRO
918 COMMON R200MICRO,R2MILLI,R20MILLI,R200MILLI,R2,R20,R50NANO,R500NANO,R5MICRO
,R50MICRO,R500MICRO,R5MILLI,R50MILLI,R500MILLI,R5,R50,CHAN.A,CHAN.B,EXTERNAL,PO
SITIVE,NEGATIVE,AC,DC,GROUND,TRIGGERED,AUTO.TRIG,AUTO.LEVEL,X1,X10
919 COMMON Dig.in.01,VDAC.A.01,Counter.01,DMM.01,Func.Gen.01,Relay.Mux.01,Scope.01
999 'End PCIB Program Shell
```

Figure 5-3. A Sample Program Shell (2 of 2)

CAUTION

*Be **very** careful not to make any errors in the CALL statements of your program. Watch for spelling errors, wrong type of parameter, or incorrect number of parameters. Remember that constants cannot be passed as parameters in CALL statements (assign a constant to a variable name **before** the CALL statement). These types of errors in CALL statements cannot be "trapped" and can cause your program to "crash", requiring you to restart your system.*

To begin inserting your application program lines, you can type each line number (beginning with 1000) followed by the statement. Or, you can have the system generate line numbers by typing:

AUTO 1000
and pressing Enter

Saving the Complete Application Program

After you have inserted your application code, you no longer have a Program Shell but a complete application program. You must save it on a work diskette that already has the state file on it. To save the program, type:

SAVE "Filename", A
and press Enter

NOTE

The A option must be used to store the file in ASCII format, which is required for later MERGE operations.

Whatever you specify for "Filename" will be the name of your application program from now on. You may specify a new name or the same name used when you stored the Program Shell from the Soft Front Panel. If you specify the same name you stored the Program Shell under, you will overwrite the original Program Shell file. In this case, after the SAVE operation, you will have a copy of the complete application program but will no longer have the original Program Shell.

Running the Complete Application Program

Once you have developed your program lines, inserted them into the Program Shell, and saved the complete application program on diskette, you are ready to run your program. Type:

RUN ''Filename''
and press Enter

''Filename'' must be the same name you gave when you saved your application program . This loads the program file from the diskette into memory and executes it (refer to your BASIC Manual for details).

Recording the Program Results

You may want to store and record the data acquired when your program is executing . You can do this by setting up internal data arrays in your BASIC program or by writing each data item directly to the work diskette.

For example, let's assume that your program contains the following statement for a Digital Multimeter that is labeled ''DMM.A.01'':

```
1060 CALL MEASURE (DMM.A.01,READING)
```

This statement returns a voltage reading in the variable ''READING''. If you take twenty consecutive readings, you need to record twenty data values. You can have a data array in your program to record these readings. You must dimension the array and use OPTION BASE 1 to set the lowest array subscript to 1. An example of BASIC lines that use a data array loop to return 20 values is:

```
1030 OPTION BASE 1
1040 DIM DATA.ARRAY(20)
1050 FOR I = 1 TO 20
1060 CALL MEASURE
       (DMM.A.01,DATA.ARRAY[I])
1070 NEXT I
```

NOTE

You have limited (64 K) BASIC workspace for your application program. This restricts the size of the arrays you can DIMmension. You can use the BASIC FRE function to determine the number of unused bytes within BASIC's data space.

After you have recorded your results, you can create a data file by wiring the data to diskette. This is useful for integrating the acquired data with applications software packages. You must write the data in the format compatible with the application package(s) being used. Chapter 7 explains why and how to create a formatted data file.

In Case of Trouble

If your application program doesn't work, check your program's CALL statements as follows:

- Restart DOS
- Re-enter PCIBAS
- Load your program (''Filename'')
- Enter the trace mode (by typing TRON)
- Type RUN and examine all line numbers with CALL statements
- Refer to the information under ''PC Instruments Error Handling'', further in this chapter.

Merging Programs

Merging a New Shell with Your Application Program

The MERGE command (refer to your BASIC Manual) allows you to incorporate statements from one program file to another. This is helpful if you wish to save all (or part) of your application program lines (statements above line 999) but need to generate a new Program Shell. You might want to do this if you have altered the hardware in the system and need an updated Program Shell (remember that when you generate a Program Shell you are committing yourself to the current hardware in the system and the instrument labels in the System View Window).

The following lines show how to add an updated Program Shell to a complete application program by using the MERGE command. Assume that the new Program Shell is in file ''SHELL2'' and the complete application program is in file ''APPLIC1''. Proceed as follows:

```
LOAD ''APPLIC1'' (if APPLIC1 is not already in
                  memory)
DELETE 1-999 (Program Shell of ''APPLIC1'')
MERGE ''SHELL2''
SAVE ''APPLIC2'',A
```

The merge will give you a complete application program (with the newly generated Program Shell) in the file ''APPLIC2''. This saves you from having to retype all the application program lines after the new Program Shell.

NOTE

The line numbers in the DELETE statement only apply if ''RENUM 1000,1000'' was the only renumbering command ever used. Otherwise, find the line equivalent to 999 by looking at the second line of your renumbered program (see Figure 5-3). That line will have a GOTO statement to a line identified as ''User program.'' Your old Program Shell terminates at the line number that immediately precedes the one specified in that GOTO statement and is identified as ''End PCIB Program Shell.'' Delete up to and including that line.

Recovering From a RENUM Command. If you forgot to use ''RENUM 1000,1000'' during the insertion of your application code, then the straight ''RENUM'' command will have changed all the line numbers in your Program Shell. For example, assume you have an application program ''APPLIC1'' that includes ''SHELL1''. Also assume that (because of a RENUM command), the Program Shell runs from lines 10 to 2140 and that your application code starts at line 2150. You can correct this as follows:

- Delete all lines before your program (e.g., DELETE 10-2140).
- Renumber your application code starting at line 1000 (RENUM 1000)
- Merge your original (not renumbered) Program Shell (MERGE ''SHELL1'')

Since the Program Shell, as created from the Soft Front Panel, is always numbered from line 1 to line 999, the above procedure will restore the correct numbers to the program currently in memory. To keep it permanently, SAVE it (with the ''A'' option).

NOTE

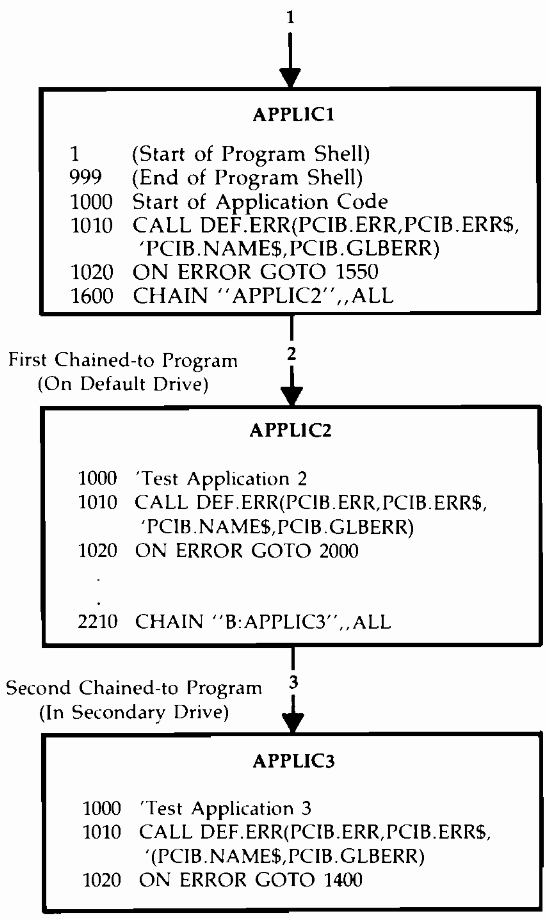
Whenever a program on diskette is merged with one in memory, the diskette program has precedence with respect to line numbers. If both programs have one or more lines with the same number, the merged program will have the lines from the diskette program.

Chaining Programs

Chaining Two or More Applications Programs. The CHAIN command (refer to your BASIC Manual for details) allows you to transfer control (including passing variables) from the current program to another program. CHAIN is useful when you have large PC Instruments application programs to run but not enough memory available. The CHAIN command executes programs in sections, each time erasing the chained-from program and leaving only the chained-to program in memory. Figure 5-4 illustrates two CHAIN statements. When the first one is executed, control is transferred to ''APPLIC2'' and APPLIC1'' is erased from memory. When the second CHAIN is executed, control transfers to ''APPLIC3'' and ''APPLIC2'' is erased from memory.

Important Points Concerning Chaining. There are requirements and restrictions that apply to chaining operations. Please refer to your BASIC manual and be sure you understand them. Note the following points concerning this use of CHAIN:

- The first executable statement in any program that you chain to must be CALL DEF.ERR (refer to the preceeding description of the DEF.ERR statement). This is needed to define the location of PC Instruments error variables for each chain application.
- Common system variables must be retained from the chained-from program to the chained-to program. This can be done by including the ALL parameter (as shown in Figure 5-4), or by retaining the COMMON statements (located in the Program Shell) in all your applications programs. This last method is the recommended practice.
- The chained-from program is always in memory, while the chained-to program must be on the default diskette. If not, then use a drive designator in your command, such as CHAIN "B:APPLIC2",,ALL
- Chaining will turn off any ON ERROR error checking function in the chained-to program. Therefore, that program code must turn the error handling back on again after the chaining operation.
- Each CHAIN command causes a RESTORE function before beginning the chained-to program. Therefore, the next READ operation will be from the first item in the first DATA statement of the chained-to program, not from where it left off in the chained-from program.



APPLIC1

```
1      (Start of Program Shell)
999    (End of Program Shell)
1000   Start of Application Code
1010   CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,
        'PCIB.NAME$,PCIB.GLBERR)
1020   ON ERROR GOTO 1550
1600   CHAIN "APPLIC2",,ALL
```

APPLIC2

```
1000   'Test Application 2
1010   CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,
        'PCIB.NAME$,PCIB.GLBERR)
1020   ON ERROR GOTO 2000
  .
  .
2210   CHAIN "B:APPLIC3",,ALL
```

APPLIC3

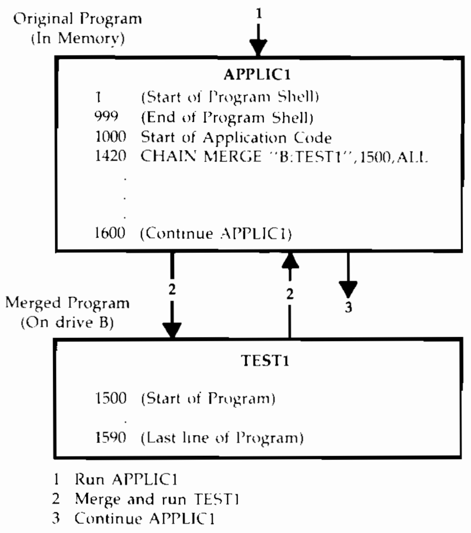
```
1000   'Test Application 3
1010   CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,
        '(PCIB.NAME$,PCIB.GLBERR)
1020   ON ERROR GOTO 1400
```

1 Run APPLIC1

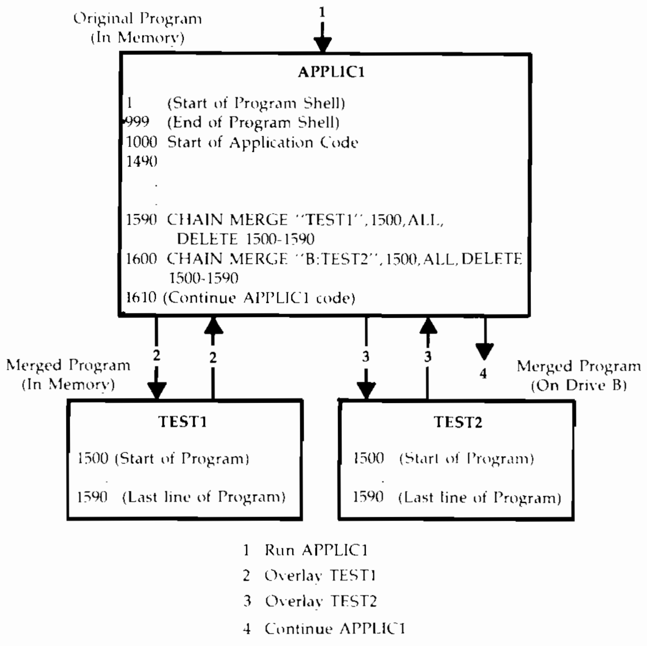
2 Erase APPLC1; run APPLIC2

3 Erase APPLC2; run APPLIC3

Figure 5-4. Example of Chained Application Program



a. Chain Merge of TEST1 with APPLIC1


Original Program
(In Memory)
1
APPLIC1
1 (Start of Program Shell)
999 (End of Program Shell)
1000 Start of Application Code
1490
1590 CHAIN MERGE "TEST1",1500,ALL,
DELETE 1500-1590
1600 CHAIN MERGE "B:TEST2",1500,ALL,DELETE
1500-1590
1610 (Continue APPLIC1 code)
2
2
3
3
4
Merged Program
(In Memory)
Merged Program
(On Drive B)
TEST1
1500 (Start of Program)
1590 (Last line of Program)
TEST2
1500 (Start of Program)
1590 (Last line of Program)
1 Run APPLIC1
2 Overlay TEST1
3 Overlay TEST2
4 Continue APPLIC1

b. Overlaying APPLIC1 with TEST1 and TEST2

Figure 5-5. Examples of CHAIN MERGE Commands

Chaining and Merging Programs

The MERGE command may be included as an option to the CHAIN command. This allows you to transfer control to another program without erasing the current program. For example:

```
1110  'Application program code
1120  CHAIN MERGE "B:TEST1",1450,ALL
1600  'Continue with application program code
```

Statement 1120 loads program TEST1 into a space, beginning at line 1450, of the application program currently in memory. TEST1 then takes control and runs. If TEST1 has an END statement, then the combined program will terminate with the end of TEST1. Otherwise, control will be returned to the the application at line 1600. The merged program must be able to fit between lines 1110 and 1600 without conflict. If there are duplicate line numbers, the lines from TEST1 will overwrite those in the appliciation program.

Instead of replacing one program with another (as is done with the standard CHAIN command), CHAIN MERGE overlays part of your existing program with another program (see Figure 5-5). The two programs then form one new complete program in memory. If you wish, you can subsequently erase the overlay program and replace with another program, as illustrated in Figure 5-5. This allows you to run many programs within the same memory space.

Be careful to observe all the rules that apply to the CHAIN and MERGE commands. These include:

- The merged program must be in ASCII format.
- All user-defined functions in a program must precede any CHAIN MERGE statement in that program.
- Ensure that needed variables are passed from the merged program. If you retain the Program Shell, then COMMON statements will be available to all your programs. Otherwise, use the ALL option, or insert the COMMON statements into each program.

- The same rules concerning DEF.ERR and ON ERROR, previously explained in ''Chaining Programs,'' apply when chaining and merging programs.

PC Instruments Error Handling

CAUTION

If there are no error-handling lines in your application program, errors will be undetected. This can cause unpredictable results from your program.

If you want PC Instruments errors to be detected, then there must be error handling lines in your application program. You can use the program segment provided on line 900 of the Program Shell or you can write your own program lines using the error variables described here.

Using Program Shell Error Handling

If you do not want to write your own error handling routine, you can use the one provided in line 900 of the Program Shell. That is the line number *before* any renumbering. If the Program Shell should be renumbered, the start of the error routine will be identified by the GOTO in the third statement of the Program Shell.

Writing Your Own Error Handling Routines

If you want to write your own error handling routines, the following error variables are available:

PCIB.ERR · Allows you to determine if an error has been detected in the last exited CALL statement. The value of PCIB.ERR should be checked *after every CALL statement.* If PCIB.ERR = 0, then no error was detected in the last CALL statement. (To determine the error message corresponding to the error number, print the PCIB.ERR$ variable.)

PCIB.ERR detects PC Instrument System errors (errors numbered 1-99) and instrument-specific errors (errors numbered 100 and above). However, it *does not* detect spelling errors or parameter passing errors (such as providing the wrong number or type of parameters) in the CALL statement. *Be careful* not to make these mistakes. They cause unpredictable results and will probably force you to restart your system.

PCIB.ERR$ - Allows you to determine the error message corresponding to the error number in PCIB.ERR.

PCIB.NAME$ - Allows you to print the label of the instrument that caused the error (for instrument-specific errors only).

These error detecting variables are updated after each CALL to an external routine. Therefore, the variables *must be checked after each CALL statement* to detect errors. Since your program will probably contain multiple CALL statements, you should develop an error handling routine that can be entered every time an error occurs. This routine should handle both PC Instruments and BASIC errors.

NOTE

The variables described here do not detect BASIC programming errors. To include these, use the BASIC ON ERROR and ERROR statements together with the ERR and ERL variables (refer to your BASIC manual).

If you do not want to include complete error handling using the three error handling variables just described, you can still get overall error detection by using the following variable:

PCIB.GLBERR - Allows you to determine if an error has been detected after *any* CALL statement in your program. If PCIB.GLBERR = 0, no error was detected in your program. Otherwise, PCIB.GLBERR contains a number that reflects the condition of the most recent CALL statement with an error. (To determine exactly which CALL statement caused the error, use a PCIB.ERR variable check after each CALL statement.)

You can use PCIB.GLBERR to determine if an error has occurred *anywhere* in the program. An example of this is:

```
3000  IF PCIB.GLBERR < >0 THEN PRINT ''An error
      has been detected in the user program.''
3010  STOP
```

Check PCIB.GLBERR at the end of the program. If PCIB.GLBERR is equal to any value other than zero, further error checking code must be included to determine which CALL statement caused the error.

If you wish to use the error handling routine provided on line 900 of your Program Shell, you *do not* have to include any ON ERROR GOTO statement in your program. That error trapping has already been enabled for you. To disable that error trapping, execute an ON ERROR GOTO 0 statement (refer to your BASIC manual for details). Appendix C lists the PC Instrument System error messages. Instrument specific error messages are listed in the applicable Instrument Owner's Guide.

Measuring Software Performance

The remainder of this chapter explains how you can measure the performance of the PC Instruments I/O statements by inserting a benchmark program into your application program.

I/O Throughput

The execution time of any I/O operation can be broken into two independent parts:

1. The time it takes for the software to decode a statement, ''massage'' the data, initiate the I/O operation, detect its completion, and return for the next statement.
2. The time it takes for the I/O hardware and the application to which it is connected to complete the requested operation.

In some cases, the time required by the hardware is very brief compared to the time needed by the software. This is the case, for instance, in an Output statement to a PC Instruments 61012A Dual Voltage DAC.

In other cases, the time for the hardware is a constant that may be comparable to the time for executing the software. This is the case for time required for the relays to settle after an Output statement to the 61011A Relay Multiplexer.

Benchmark Program

You may want to measure the performance of an I/O statement or group of I/O statements in your application. Figure 5-6 is a simple benchmark program you can use to evaluate your application program. The benchmark program uses the BASIC TIMER statement (refer to your BASIC manual for details). You can keystroke this program into your Program Shell. To ensure reasonable accuracy, it is recommended that you execute the loops at least 1000 times, as done in Figure 5-6.

```
1000  ' Start of Application Program
1010  CALL DEF.ERR(PCIB.ERR,PCIB.
       ERR$,PCIB,NAME$,PI.GLBERR) 'This line is
      ' needed only if this program is chained-to from
      ' another program
1005  ON ERROR GOTO 3 'or to line number of your
      'own error handler
1020  '
1130  ' Your PC Instruments state filename here
1140  '
1150  INPUT "ENTER STATE FILENAME";FILE$
1160  CALL INITIALIZE.SYSTEM(FILE$)
1170  '
1180  ' Time the dummy loop, followed by the loop
1190  ' containing your instruction
1200  A = TIMER
1210  FOR I=1 TO 1000
1220  NEXT I
1230  B = TIMER
1240  FOR I=111 TO 1000
1250  ' Put your statement(s) to be timed here
1260  NEXT I
1270  C = TIMER
1280  '
1290  ' Calculate the execution time
1300  '
1310  LPRINT "Time per statement ="; (C-2*B+A)/1000;
      "seconds"
1320  END
```

Figure 5-6. Sample Benchmark Program

6 Stimulus/Response Testing

How This Chapter Is Organized

Chapter 6 begins with a complete programming example of a HP PC Instruments stimulus/response test. This example builds on the concepts of manual instrument control from Chapter 4, programmed instrument control from Chapter 5, and a graphics utility described further in this chapter.

Next, programming details specific to stimulus/response testing are introduced. Here you will learn how to write a FOR...NEXT loop to load a data array with stimulus/response data pairs. Inside the FOR...NEXT loop, the PC Instruments OUTPUT statement generates the test stimulus output, while the MEASURE statement initiates the test response input.

Finally, you will be introduced to the graphics utility portion of the PC Instruments Data Acquisition Software product. The graphics utility has been designed for easy integration into your BASIC stimulus/response application program. The graphics utility can plot linear, semilogarithmic, and logarithmic relationships among up to three independently measured responses to a given stimulus.

A Stimulus/ Response Programming Example

A manufacturer of digital panel meters uses a voltage-to-frequency converter (V/F) IC (integrated circuit) device as the heart of of its product. The manufacturer has asked its engineering group to design an incoming inspection tester for the V/F converter. The test circuit, using nominal circuit values and no external offset adjustment, must verify that a given device will produce the required dynamic range of frequencies within a specified tolerance band.

Block Diagram of the V/F Converter Tester

Figure 6-1 shows the voltage-to-frequency converter test circuit. External resistor R1 and external capacitor C1 establish the maximum output frequency at pin F_{out}. The V/F device provides a linear relationship between pin V_{in} and pin F_{out}. The minimum output frequency is dependent on the offset voltage of the V/F device; in this design, no external offset voltage adjustment is provided by the meter manufacturer. The IC device under test is biased from ± 15 Vdc and the tester has a zero insertion force socket for easy removal of the device.

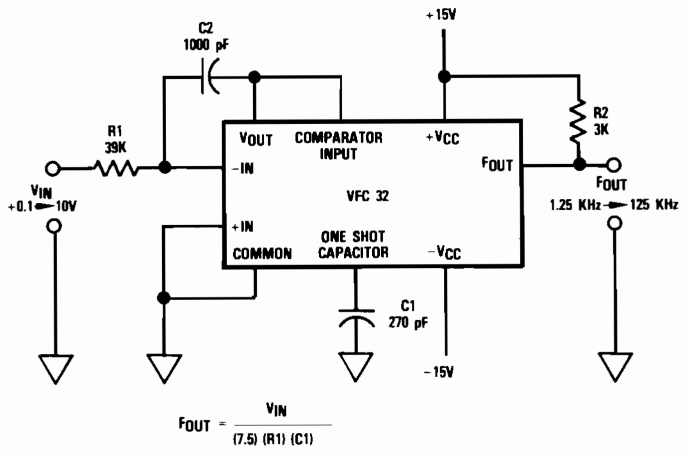


Figure 6-1. Circuit Diagram of Voltage-to-Frequency Converter

Figure 6-2 shows the hardware block diagram for an automated V/F Converter tester. The test is easily automated using a PC Instruments 61012A Dual Voltage DAC and 61015A Universal Counter connected to the PC (personal computer). One output of the Dual Voltage DAC is the stimulus device, programmed from +100 millivolts to +10 volts. The other output of the 61012A can be used to control a programmable power supply for the V/F device's bias voltages.

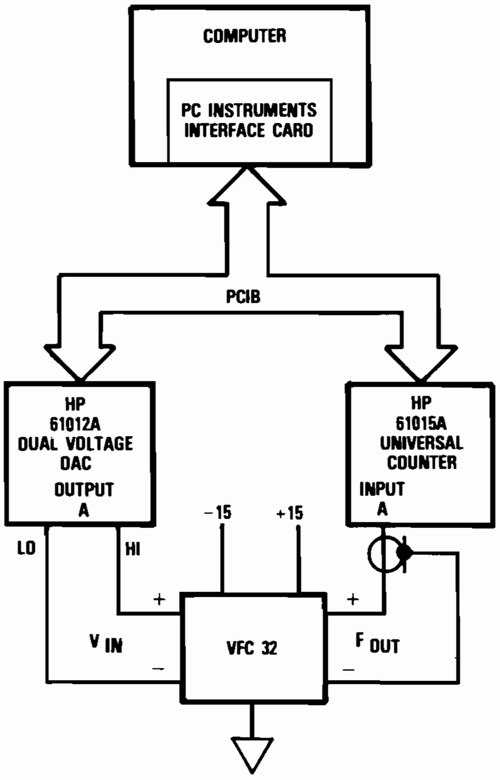

COMPUTER
PC INSTRUMENTS INTERFACE CARD
PCIB
HP 61012A DUAL VOLTAGE DAC
OUTPUT A
HP 61015A UNIVERSAL COUNTER
INPUT A
LO
HI
−15
+15
+
V IN
−
VFC 32
+
F OUT
−

Figure 6-2. Block Diagram of Incoming Inspection Test

The Universal Counter is the response measuring device, programmed to measure the output frequency of the V/F device up to approximately 125 kilohertz. The results for each tested device will be plotted on the computer screen. A second plot trace will be used to indicate any frequencies where the device fails within a ±2% tolerance band.

Getting Ready to Program the V/F Converter Tester

If you have followed the procedures given in Chapter 3, then you should have a bootable work diskette (or fixed disk) with the BASICA and PC Instruments Software files already installed. In order to follow this example completely, you will need a second work diskette that contains the graphics utility files. Copy these files from the optional Data Acquisition Software Master diskette (refer to Table 6-1 for a list of these files). If you have a fixed-disk system, insert the second work diskette in drive B. If you have a dual-diskette system, insert the first work diskette in drive A and the second work diskette in drive B. The second work diskette will ultimately hold the stimulus/response BASIC program, the PC Instruments state file, and the graphics template files for the V/F Converter test. For a dual diskette system, both these work diskettes must remain in their respective drives while the program is being developed and run.

Running the V/F Converter Test Soft Front Panel

From DOS, enter the Soft Front Panel. Bring the PC Instruments Dual Voltage DAC into the Interactive Instrument Window. Press the **REAR PANEL** softkey. Change the DAC A label from "DAC.A.01" to "V.IN" to indicate the stimulus input voltage. Press the **FRONT PANEL** softkey and set up the DAC for the ±10 Volt range with its output enabled and set for zero volts, as shown in Figure 6-3.

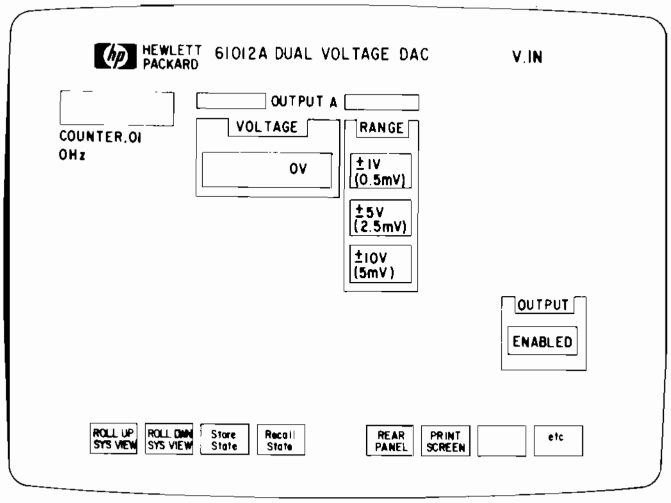


Figure 6-3. Soft Front Panel for Dual Voltage DAC (V.IN)

Next, bring the PC Instruments Universal Counter into the Interactive Instrument Window. Press **REAR PANEL** and change the counter's label from ''COUNTER.01'' to ''F.OUT'' to represent the response output frequency. Press **FRONT PANEL** . Set the counter Function to ''FREQUENCY'', the Range to ''10 Hz to 10 MHz'', the Gate Time to ''0.1 SEC'', the Trigger to ''INTERNAL'', and Slope A ''POS'' as shown in Figure 6-4.

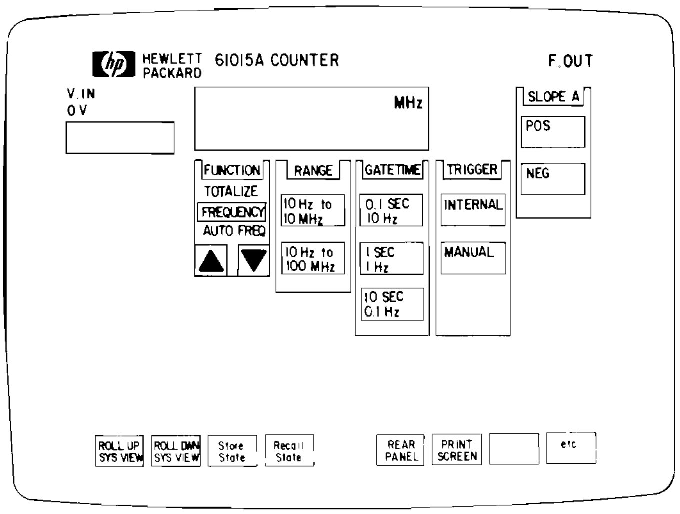


Figure 6-4. Soft Front Panel for Universal Counter (F.OUT)

Storing the State File and Program Shell File

You are now ready to store the state file for the V/F Converter test program and exit. Press the **Store State** softkey and you will be prompted for an .HPC file name. We will name the instrument state file "B:VFSTATE." Note that you are storing it on drive B.

Next, exit the Soft Front Panel by storing a BASIC application Program Shell file to disk. The Program Shell links the DAC labeled "V.IN" and the Counter labeled "F.OUT" with their respective states and with the V/F Converter test program. Press the **etc.** softkey to get to the second row of Main Menu softkeys. Then, press **Store Program** . You will be prompted for a .BAS file name. We will name the application Program Shell file "B:VFPROG" and this file will also be stored on drive B. Finally, press **EXIT** to return to DOS.

Writing the V/F Converter Test Program in BASIC

The following steps outline how to write the test program.

- Start by loading BASICA from DOS by typing:

  ```
  A> PCIBAS
  ```
  and pressing Enter

- From BASIC, load the Program Shell file (from the work diskette in the secondary drive) into the BASIC workspace by typing:

  ```
  LOAD "B:VFPROG"
  ```
  and pressing Enter

- Merge the PC Instruments Data Acquisition Software graphics utility (on the work diskette in the secondary drive) with the Program Shell already in the BASIC workspace by typing:

  ```
  MERGE "B:GRAFSHEL"
  ```
  and pressing Enter

- Keystroke the program segment from line 2000 through line 2310 (see Figure 6-5) into the BASIC workspace. (See Chapter 5 of this manual for additional details.)

- Edit the program segment to the proper array dimension and program file name with the following lines:

  ```
  1500 MAX.DIM% = 100: DIM
       DATA.VAL(MAX.DIM%,4)
  1520 PROG.NAM$ = "VFPROG"
  10010 PROG.NAM$ = "VFPROG"
  ```

- Save the total V/F Converter Test program on the work diskette in drive B (under the same name used to store the Program Shell file from the Soft Front Panel) by typing:

  ```
  SAVE "B:VFPROG",A
  ```
  and pressing Enter

```
1     DEF SEG:CLEAR ,&HFE00:GOTO 4  'BEGIN PCIB PROGRAM SHELL
999   'END PCIB PROGRAM SHELL
1000  '********************** GRAPHICS SHELL INITIALIZATION *****************
1500  MAX.DIM% = 100: DIM DATA.VAL(MAX.DIM%,4)
1510  DEF SEG=LIB.ADDR
1520  PROG.NAM$="VFPROG"
1530  PROG.LINE%=1550
1540  CHAIN "GRAFCONF",,ALL
1550  DEF SEG=PCIB.SEG: ON ERROR GOTO 900
1560  CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB.GLBERR)
1990  '********************** USER PROGRAM STARTS HERE **********************
2000  '******** INITIALIZE PC INSTRUMENTS FROM STATE FILE ******************
2010  FILE$="VFSTATE"
2020  CALL INITIALIZE.SYSTEM(FILE$)
2030      IF PCIB.ERR <>0 THEN ERROR PCIB.BASERR
2040  CALL DISABLE.INT.TRIGGER(F.OUT)
2050      IF PCIB.ERR <>0 THEN ERROR PCIB.BASERR
2060  '
2070  '******** DEFINES EXPECTED TEST RESULTS *******************************
2080  RESISTOR1=39000!
2090  CAPACITOR1=2.7E-10
2100  DEF FNR(V)=V/(7.5*RESISTOR1*CAPACITOR1)
2110  '
2120  '******** PERFORM THE STIMULUS/ RESPONSE TEST LOOP *******************
2130  MAX.VOLTS=10
2140  DELTA.V=MAX.VOLTS/ MAX.DIM%
2150  FOR LOOP%=1 TO MAX.DIM%
2160      STIMULUS=LOOP%*DELTA.V
2170      CALL OUTPUT(V.IN,STIMULUS)
2180          IF PCIB.ERR <>0 THEN ERROR PCIB.BASERR
2190      CALL MEASURE(F.OUT,RESPONSE)
2200          IF PCIB.ERR <>0 THEN ERROR PCIB.BASERR
2210  '
2220  '******** CHECK THE RESULTS AND LOAD THE ARRAY ***********************
2230      DATA.VAL(LOOP%, 1)=STIMULUS
2240      DATA.VAL(LOOP%, 2)=RESPONSE
2250          IF RESPONSE > 1.02*FNR(STIMULUS) THEN DATA.VAL(LOOP%, 3)=2
2260          IF RESPONSE < .98*FNR(STIMULUS) THEN DATA.VAL(LOOP%, 3)=1
2270  NEXT LOOP%
2280  '
2290  '******** DISABLE PC INSTRUMENTS *************************************
2300  CALL DISABLE.SYSTEM
2310      IF PCIB.ERR <>0 THEN ERROR PCIB.BASERR
2320  '
9990  '******** USER PROGRAM ENDS HERE *************************************
10000 DEF SEG=LIB.ADDR
10010 PROG.NAM$="VFPROG"
10020 PROG.LINE%=10040
10030 CHAIN "GRAFPLOT",,ALL
10040 DEF SEG=PCIB.SEG: ON ERROR GOTO 900
10050 CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB.GLBERR)
10060 END
```

Figure 6-5. Test Program Listing

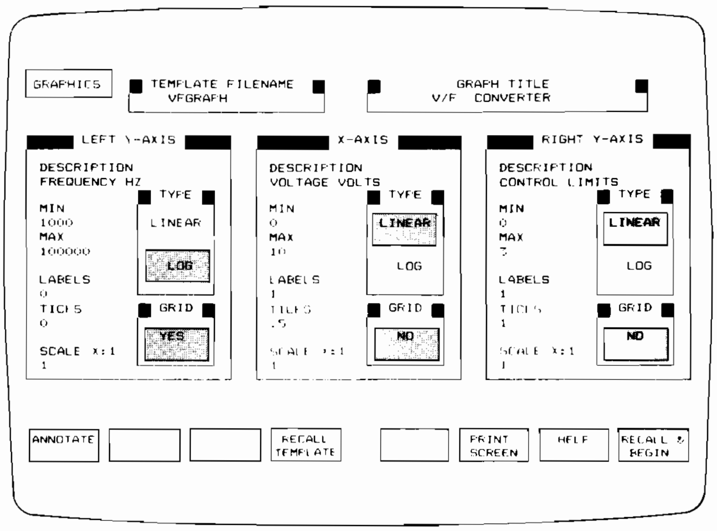
The total program now includes the PC Instruments Program Shell, plus an edited form of the Data Acquisition Software graphics utility Program Shell and a stimulus/response test built around the Dual Voltage DAC "V.IN" and the Universal Counter "F.OUT".

Running the V/F Converter Test Program

Insert a V/F Converter IC into the test socket and apply the ±15 Vdc bias. From the BASIC prompt, with the program file VFPROG still in the BASIC workspace, start the test:

```
RUN "B:VFPROG"
```

The program begins with the Program Shell segments initializing linkages to the PC Instruments. The first screen to appear on your PC will be the graphics utility Axes Menu (see Figure 6-6). Use this menu to configure a unique graphics template called "VFGRAPH" for this test. Later in this chapter, you are shown how to complete the Axes Menu.



```
GRAPHICS    TEMPLATE FILENAME       GRAPH TITLE
            VFGRAPH                 V/F  CONVERTER

LEFT Y-AXIS                X-AXIS                     RIGHT Y-AXIS
DESCRIPTION                DESCRIPTION                DESCRIPTION
FREQUENCY HZ               VOLTAGE VOLTS              CONTROL LIMITS
MIN              TYPE      MIN              TYPE      MIN              TYPE
1000             LINEAR    0                [LINEAR]  0                [LINEAR]
MAX              [LOG]     MAX              LOG       MAX              LOG
100000                     10                         3
LABELS                     LABELS                     LABELS
0                GRID      1                GRID      1                GRID
TICKS            [YES]     TICKS            [NO]      TICKS            [NO]
0                          .5                         1
SCALE X:1                  SCALE X:1                  SCALE X:1
1                          1                          1

ANNOTATE  RECALL TEMPLATE  PRINT SCREEN  HELP  RECALL & BEGIN
```

Figure 6-6. Graphics Utility Axes Menu

Press the **ANNOTATE** softkey to continue with your description of the graphics template for the V/F Converter test. This brings up the Annotate Menu (see Figure 6-7). Graphics template "VFGRAPH" will present a semilogarithmic plot of input voltage vs output frequencies for the V/F IC under test. The input voltage, or stimulus, is plotted linearly on the X-axis. The output frequency, or response, is plotted logarithmically on the left Y-axis. The right Y-axis indicates any input voltages for which the V/F IC produces output frequencies that are either under tolerance (value = 1) or over tolerance (value = 2).

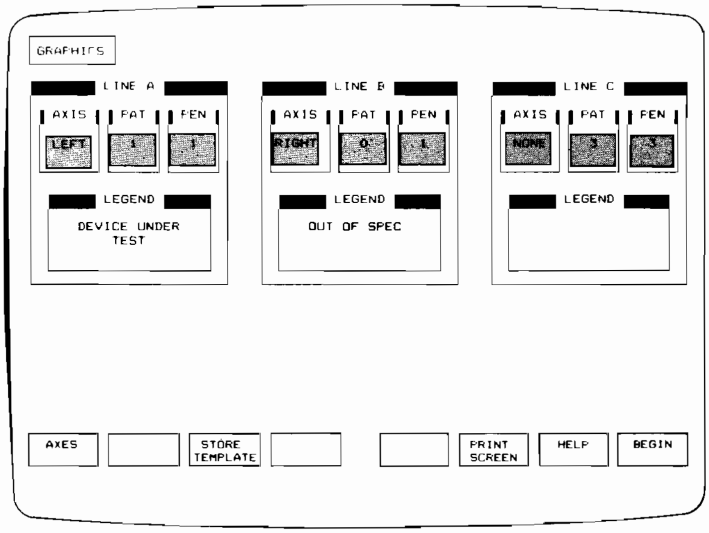


Figure 6-7. Graphics Utility Annotate Menu

Use the **STORE TEMPLATE** softkey to store the combination of the Axes Menu and the Annotate Menu with the file name "B:VFGRAPH" to the diskette in the secondary drive. After that, press the **BEGIN** softkey to continue in the program.

Plotting the Results of the V/F Converter Test

The test program next instructs the Dual Voltage DAC and the Universal Counter to loop through 100 test values. With each new input voltage, the V/F Converter's output frequency is measured and checked against limits. Each stimulus/response pair is loaded into the data array "DATA.VAL" for later plotting. It takes tens of seconds for this portion of the program to execute. As written, our example program does not automatically store the data array.

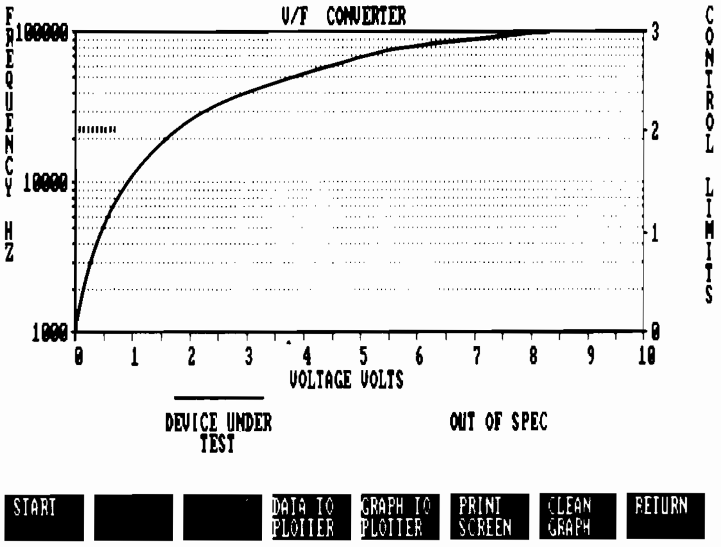


Figure 6-8. Results of Incoming Inspection Test

When the graphics template appears on the screen of your PC, press the **START** softkey to plot the test data from array "DATA.VAL" onto graphics template "VFGRAPH." Results of one such plot are shown in Figure 6-8. In our example, a V/F Converter IC fails the test if any extra pixel points are plotted at "1" or "2" with respect to the right Y-axis. For a hard copy of the plot on your graphics line printer or plotter, press the **PRINT SCREEN** or **GRAPH TO PLOTTER** softkey. The example program ends when you press the **RETURN** softkey, but you could write the program to loop back to ''pass or fail'' the next V/F Converter IC.

Techniques for Stimulus/ Response Testing

Your PC Instruments System is ideally suited for automating stimulus/response tests. In stimulus/response testing, your personal computer commands a PC Instruments output device to generate (OUTPUT) a new stimulus value. For example, with a PC Instruments Function Generator, you might set the next frequency to your test application. Then, your personal computer commands a PC Instruments input device to MEASURE the new value of the response. You could use a PC Instruments Digitizing Oscilloscope to measure the amplitude resulting from the test application at the new frequency.

Programming such a test is straightforward in BASIC. You can use a FOR...NEXT loop to OUTPUT the stimulus and MEASURE the response over a range of values. You assign the stimulus/response data pairs to an array variable. That array can then be stored to diskette for later processing. Or, it can be read by the graphics utility (included in the PC Instruments Data Acquisition Software) and plotted with the same program.

Instruments for Stimulus/Response Testing

There are three general classes of instruments you can use to automate a stimulus/response test. The first class is output instruments that provide a programmable stimulus value. Examples of this class include the Dual Voltage DAC, the Function Generator, and the Digital Output instruments. Each output channel may go directly to the test application as a form of electronic stimulus or may first be transformed by a transducer into a physical stimulus (see Figure 6-9). Notice that it is very convenient in your programming for you to label output devices and scale your variables to reflect the physical properties of the test application.

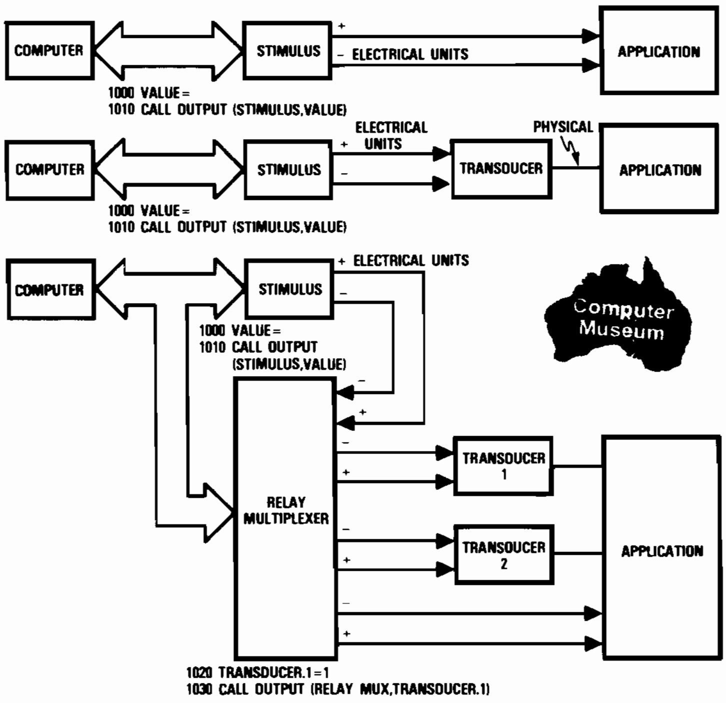


Figure 6-9. Examples of Single-Channel and Mulitple-Channel Electrical & Physical Stimulus

The second class of instruments include those input instruments that measure test response under computer control. Examples of this class include the Digital Multimeter, the Universal Counter, the Digitizing Oscilloscope, and the Digital Input instruments. Each input channel may come directly from the test application in the form of an electronic response, or may first be transformed from a physical response to an electronic signal by a transducer (see Figure 6-10). Again, it helps to document your program by labelling input devices and scaling your variables to reflect the physical properties of the test.

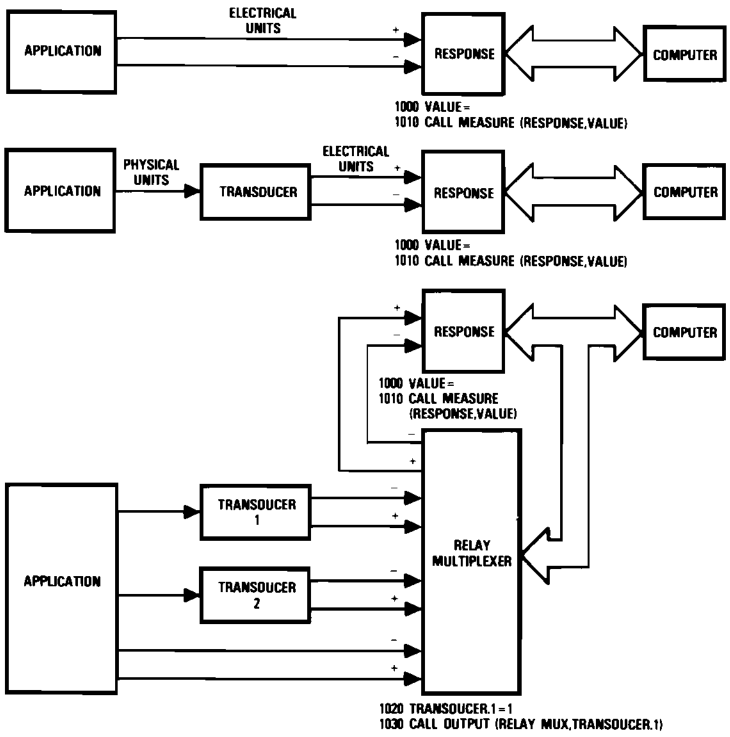


Figure 6-10. Examples of Single-Channel and Mulitple-Channel Electrical & Physical Response

There is a third class of programmable instruments that aid in configuring a stimulus/response test. Examples in this class include relay actuators, multiplexers and demultiplexers, and bias power supplies. These devices control the switching of sources to loads and power up the application under test. The PC Instruments Relay Multiplexer, Relay Actuator, Digital I/O, and Dual Voltage DAC can often be used for such tasks.

IEEE 488 (HP-IB) Instrument Control

In some applications, you may need to program an additional instrument that must be operated under IEEE 488, or HP-IB (Hewlett-Packard Interface Bus) control. You can do this if you have the optional PC Instruments HP-IB I/O Library software product. While programming with HP-IB is beyond the scope of this manual, certain design features of the HP-IB I/O Library are important to note. Its programming syntax is designed to be totally compatible with any BASIC application program written with the PC Instruments System and/or the graphics utility of the PC Instruments Data Acquisition Software. For more details about the HP-IB I/O Library, please refer to the supplement for that product that is included with this manual.

NOTE

IEEE instruments are physically connected to an IEEE 488 bus, which has no electrical connection to any PC Instruments. Although an IEEE 488 instrument is easily programmed from BASIC with the HP-IB I/O Library software, it cannot be controlled from the PC Instruments Soft Front Panel.

Stimulus/Response Loop Tasks

Introduction. Figure 6-11 helps to illustrate the fundamental loop tasks you must consider when programming a stimulus/response test. One key point to consider in your stimulus/response program is the need for scaling numerical quantities between their actual physical values and the corresponding electrical values used by PC Instruments. The incremental stimulus applied with each loop iteration must be directly related to the loop index variable. Therefore, convert this loop index variable into physical units and save it in an array (DATA.VAL). This array will then contain all the stimulus values in physical unit. Next, convert the array variable into electrical units for the PC Instruments OUTPUT statement so that actual physical units are supplied by your transducer. You may also want the value variable returned by the PC Instruments MEASURE statement to

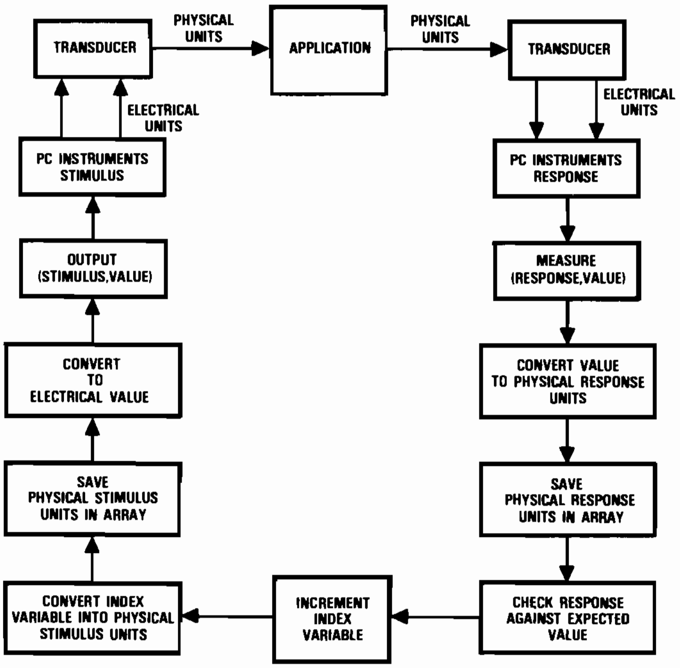

TRANSDUCER
PHYSICAL UNITS
APPLICATION
PHYSICAL UNITS
TRANSDUCER
ELECTRICAL UNITS
ELECTRICAL UNITS
PC INSTRUMENTS STIMULUS
PC INSTRUMENTS RESPONSE
OUTPUT (STIMULUS,VALUE)
MEASURE (RESPONSE,VALUE)
CONVERT TO ELECTRICAL VALUE
CONVERT VALUE TO PHYSICAL RESPONSE UNITS
SAVE PHYSICAL STIMULUS UNITS IN ARRAY
SAVE PHYSICAL RESPONSE UNITS IN ARRAY
CONVERT INDEX VARIABLE INTO PHYSICAL STIMULUS UNITS
INCREMENT INDEX VARIABLE
CHECK RESPONSE AGAINST EXPECTED VALUE

Figure 6-11. General Stimulus/Response Test Loop Tasks

be scaled to reflect the actual physical units returned by your transducer. This response variable can also be saved in an array (DATA.VAL) for later use; i.e., plotting a stimulus/response curve.

Another key point is that the maximum dimension (MAX.DIM%) of the data array (DATA.VAL) will set the number of iterations through the loop. MAX.DIM% also determines the maximum number of points plotted on your graph. There is a tradeoff here. If the array dimension is large, you can measure many data pairs at the expense of taking significantly longer to execute the loop and consuming more of the BASIC work space. If you use a smaller array dimension, then the loop will execute faster. However, there will be fewer data points plotted, resulting in a less detailed, lower-resolution graph.

Example of Programming a Loop. Earlier in this chapter, a specific example of a programming loop was given in the V/F Converter test program. The listing in Figure 6-12 is a more general example that illustrates how to scale values used in a stimulus/response programming loop. The following steps ''walk'' you through the programming loop.

1. Relate the loop index to the array index in order to load the array. The loop index will be within a FOR... NEXT construct with a linear, integer step size. It is convenient to keep the number of steps in the loop equal to the number of rows in the array DIM statement. Make the minimum value of loop index equal to the minimum value of array index, i.e. either 0 or 1. If this is not the case, use the OPTION BASE statement to make them equal. Within the loop, load the stimulus/response pair into the array (DATA.VAL) in terms of physical units. For example:

```
2150 FOR LOOP.INDEX% = 1 TO MAX.DIM%
    STEP 1
    .
2170 DATA.VAL(LOOP.INDEX%,1) = STIMULUS
    .
2260 DATA.VAL(LOOP.INDEX%,2) = RESPONSE
2270 NEXT LOOP.INDEX%
```

2. Scale the desired value of physical stimulus values to the programmed range of the loop index. In many applications, this will be a linear relationship, such as:

```
2160 STIMULUS = SLOPE.1*LOOP.INDEX%
    +OFFSET.1
```

   For example, assume you want a transducer to apply a linear force stimulus over the range of 33 to 330 newtons while your programmed loop index increments from 1 to 100. From the equation of a straight line:

```
SLOPE = [Y(end)-Y(start)]/[X(end)-X(start)]
      = 330-33/100-1 = 3.00
OFFSET = Y(start)-SLOPE X(start) = 33-(3.00)(1)
       = 30.0
STIMULUS = 3.00*LOOP.INDEX% + 30.0
```

In a similar manner, scale the electrical units so that your PC Instruments OUTPUT statement provides the desired physical stimulus value at the application. In the case of a linear relationship:

```
2180 VALUE.OUT = SLOPE.2*STIMULUS +
     OFFSET.2
```

NOTE

In some applications (such as the V/F Converter test example in this chapter), one or both of the data conversions detailed in this step may not be necessary. Also, depending on your application, the scaling of VALUE.OUT to STIMULUS and/or STIMULUS to LOOP.INDEX% may not be linear, as depicted here.

3. For sensitive loads, it is a good idea to test the computed stimulus value against "soft limits" before invoking the OUTPUT statement. Soft limits are software (not hardware) constraints that limit the stimulus output. If a limit is exceeded, then skip the program lines that generate the stimulus.

For example:

```
2190 IF VALUE.OUT > LIMIT.1 THEN 2270
2200 IF VALUE.OUT < LIMIT.1 THEN 2270
2210 CALL OUTPUT
    (STIMULUS.DEVICE.LABEL,VALUE.OUT)
2270 NEXT LOOP.INDEX%
```

4. Measure the response value from the application. (Make sure that the PC Instruments response device has been properly triggered to take a reading at this time.) For example:

```
2220 CALL MEASURE
    (RESPONSE.DEVICE.LABEL,VALUE.IN)
```

5. Scale the value of VALUE.IN (in electrical units) returned from the PC Instruments to the desired application physical response units (refer to Step 2 for details). Note that the stimulus/response pair loaded in the array DATA.VAL in Step 1 is in physical units. An example in which the electrical response units are linearly related to the actual physical response is:

```
2230 RESPONSE = SLOPE.3 * VALUE.IN +
    OFFSET.3
```

6. In many cases you will want to test each response against upper and lower acceptable control limits and program an alarm if a response exceeds a limit. For example:

```
1990 REM DEFINE RESPONSE TO BE AN
    EXPECTED FUNCTION OF THE STIMULUS
2000 DEF FNR(S) = (user's function)
    .
2240 IF RESPONSE > LIMIT.3*FNR(STIMULUS)
    THEN ALARM FUNCTION
2250 IF RESPONSE < LIMIT.4*FNR(STIMULUS)
    THEN ALARM FUNCTION
```

```
2000  DEF FNR(S) = (DEFINE THE RESPONSE AS A FUNCTION OF THE STIMULUS)
2010  '
2020  REM  for a linear relationship
2030  REM  Xaxis - loop.index and Yaxis - transducer physical units
2040  SLOPE.1  = (Refer to Step 2)
2050  OFFSET.1 =(Refer to Step 2)
2060  REM  for a linear relationship
2070  REM  Xaxis - physical units and Yaxis -  PC Instruments electrical units
2080  SLOPE.2  = (Refer to Step 2)
2090  OFFSET.2 =(Refer to Step 2)
2100  REM for a linear relationship
2110  REM Xaxis - PC Instruments electrical units and Yaxis - physical units
2120  SLOPE.3  = (Refer to Step 2)
2130  OFFSET.3  = (Refer to Step 2)
2140  '
2150  FOR LOOP.INDEX% = 1 TO MAX.DIM% STEP 1
2160            STIMULUS = SLOPE.1 * LOOP.INDEX% + OFFSET.1
2170       DATA.VAL(LOOP.INDEX%, 1) = STIMULUS
2180            VALUE.OUT = SLOPE.2 * STIMULUS + OFFSET.2
2190                 IF VALUE.OUT > LIMIT.1 THEN 2270
2200                 IF VALUE.OUT < LIMIT.2 THEN 2270
2210       CALL OUTPUT(STIMULUS.DEVICE.LABEL, VALUE.OUT)
2220       CALL MEASURE(RESPONSE.DEVICE.LABEL, VALUE.IN)
2230            RESPONSE = SLOPE.3 * VALUE.IN + OFFSET.3
2240                 IF RESPONSE > LIMIT.3 * FNR(STIMULUS) THEN ALARM.FUNCTION
2250                 IF RESPONSE < LIMIT.4 * FNR(STIMULUS) THEN ALARM.FUNCTION
2260       DATA.VAL(LOOP.INDEX%, 2) = RESPONSE
2270  NEXT LOOP.INDEX%
```

Figure 6-12. Example of Stimulus/Response Program Loop

Stimulus/Response Loop Housekeeping

Certain other housekeeping chores must be programmed before and after the FOR...NEXT loop. These tasks have to do with initializing the PC Instruments to a known state before the stimulus/response test and then returning it a known state after the data acquisition (but before plotting the graph).

Some things to do *before* the loop are:

- If you are *not* using the Data Acquisition Software graphics utility, then be sure to DIMension the data array variable. (If you are using the graphics utility, its Program Shell will automatically do this for you):

  1010 DIM DATA.VAL(100,2)

- Initialize the PC Instruments from their state file.
- Initialize the PC Instruments error handling variables.
- Turn on any bias power supplies needed by the application.
- Set the triggering mode of all response devices.
- Enable the outputs of all stimulus devices.

Some things to do *after* the loop are:

- Perform any error handling functions.
- Disable all PC Instrument stimulus devices.
- Turn off any bias power supplies for the application.

- EITHER CHAIN to the PC Instruments graphics utility, OR save the contents of the array DATA.VAL in a data file.

- END the program. (If you are using the graphics utility, its Program Shell will automatically do this for you).

Software Timing Considerations. The time required by your program to loop through its data acquisition and the graphical presentation of the data is directly related to the dimension of the data array. The smaller the array's row dimension, the faster the program will run. You can also reduce the time by limiting the number of BASIC statement lines within the FOR...NEXT loop. Use only statements needed by the application for scaling (stimulus and/or response), stimulus soft limits, and response alarms. Keep constants and initialization statements outside the FOR...NEXT loop. Remember, the fewer the number of programming statements within its loops, the faster a program will run.

Hardware Timing Considerations. A fundamental timing consideration with the OUTPUT command to a stimulus device is: "when does the newly programmed value become valid?" It takes time for the hardware to respond to range mode changes. Any PC Instruments SET.XXXX or

OUTPUT statement allows time for the hardware to completely respond before the next statement becomes effective. However, watch for capacitive loading effects from cabling to the application; stray capacitance will delay the leading edge of a newly programmed stimulus value.

There is a similar fundamental timing consideration with the MEASURE command to a response device. That is: ''when is the device actually triggered?'' When executed, by the computer, the PC Instruments MEASURE statement generates a single trigger. It is important to your application program that you understand exactly where the trigger occurs on the response waveform. Noise and ringing effects on input signals can cause unexpected results, such as measurement of the noise instead of the input signal. Noise can also completely mask very low amplitude signals.

Plotting Test Results with the PC Instruments Graphics Utility

The optional PC Instruments Data Acquisition Software product has a graphics utility designed to be easily integrated with BASIC programs that utilize PC Instruments System Software. This graphics utility allows you to conveniently plot the engineering or scientific data, acquired by the PC Instruments software, on your personal computer screen.

What You Can Do With the Graphics Utility

The graphics utility can accommodate up to three concurrent response lines. The plot relationships may be linear, semilogarithmic, or logarithmic. Different sets of test data can be plotted on a common graphics template. And, the graphics utility can be used to obtain a hard copy of the graph from a graphics line printer or a HP plotter.

Structure of the Graphics Utility's Program Files

Figure 6-13 shows the structure of the Data Acquisition Software product's graphic utility. Information stored in a graphics template file is combined with data in array variable DATA.VAL, which is resident in the BASIC workspace. It does not matter how DATA.VAL is loaded. Consequently, it can be loaded from a saved data file, from your BASIC stimulus/response program, from the PC Instruments Data Acquisition Software, or from some other BASIC program that you write.

You will MERGE the graphics utility Program Shell ''GRAFSHEL'' with the BASIC workspace by a keyboard command. The Program Shell will:

- DIMension an array called ''DATA.VAL''
- CHAIN to the graphics template program ''GRAFCONF''
- After you have loaded array DATA.VAL, CHAIN to the graphics plotting program ''GRAFPLOT''

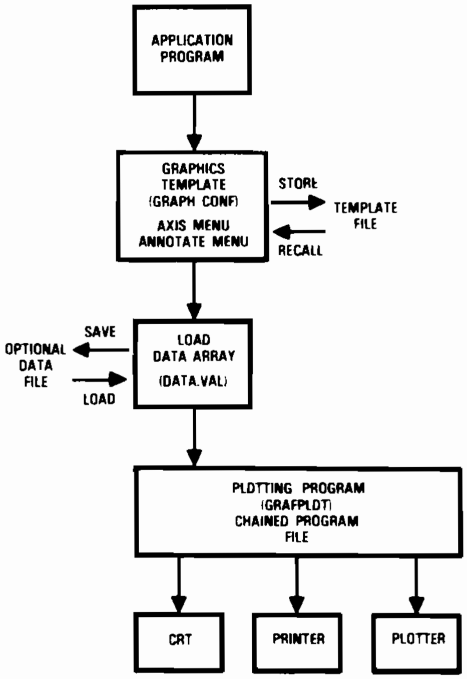


Figure 6-13. Structure of the Graphics Utility

The graphics utility is physically partitioned into files (refer to Table 6-1) that must be resident on a diskette drive while you use these programs. In the process of running the graphics utility you will create one or more graphics template files. If you want these files on a diskette drive other than the one currently holding the graphics utility files, then you must specify a complete drive and directory path when you name the template file. However, if you only want to put the .APP and .EXE utility files in a subdirectory, then use the DOS SET command, such as:

```
>A SET APP = (drive:) \subdirectory\
```

Table 6-1. Diskette-Resident Graphics Utility Files

No.	Name	Contents
1	GRAFSHEL.BAS	graphics Program Shell
2	GRAFÇONF.BAS	graphics template program
3	GRAFPLOT.BAS	graphics plotting program
4	GRAFCONF.APP	ASCII text file
5	SOFTKEYS.APP	softkey label file
6	STATUS.APP	status message file
7	HELP.APP	help message file
8	APPLIC.EXE	assembly utilities
9	*filename	graphics template file

* Any DOS filename may be used.

How to Integrate the Graphics Utility with a BASIC Program

All the graphics utility files from Table 6-1 must be either on your fixed-disk drive or copied to a work diskette that is resident in your diskette drive while you are using the graphics utility programs.

If you have a dual flexible diskette system, copy the graphics utility files to the work diskette resident in the secondary drive (usually drive B). The work diskette in the secondary drive will also contain the PC Instruments state file and the PC Instruments Program Shell file. The work diskette resident in the default drive (usually A) must have a copy of the PC Instruments System Software files and the BASICA language files.

Once you have set up your files as described, it is easy to integrate the graphics utility with your BASIC stimulus/response test program. The following steps show you how:

1. From BASIC, load the PC Instruments Program Shell file (that you named and saved when you exited the Soft Front Panel) by typing:

   LOAD "Filename"
   and pressing Enter

If, for some reason, the PC Instruments Program Shell file is resident on the alternate drive, you can still access the file by specifying its drive designator and directory path. For example, you can type:

LOAD "B: \ Filename"

Once loaded into the BASIC workspace, the PC Instruments Program Shell becomes the starting point for your final stimulus/response application program as shown in Figure 6-14.

2. Merge the graphics utility Program Shell with the PC Instruments Program Shell in the BASIC workspace by typing:

   MERGE "B:GRAFSHEL"
   and pressing Enter

3. Now edit the three graphics utility Program Shell lines that dimension the data array (DATA.VAL) and that specifically reference your stimulus/response

program filename (''Filename''). The maximum value (MAX.DIM%) for your data array is limited only by the available memory size. Throughout our examples, we will assume that there are 100 data points. When you finish writing your stimulus/response test program, you will save it under some filename. Use that same filename when you edit the following three graphics utility Program Shell lines:

```
1500 MAX.DIM% = 100: DIM
    DATA.VAL(MAX.DIM%,4)
1520 PROG.NAM$ = ''Filename''
10000 PROG.NAM$ = ''Filename''
```

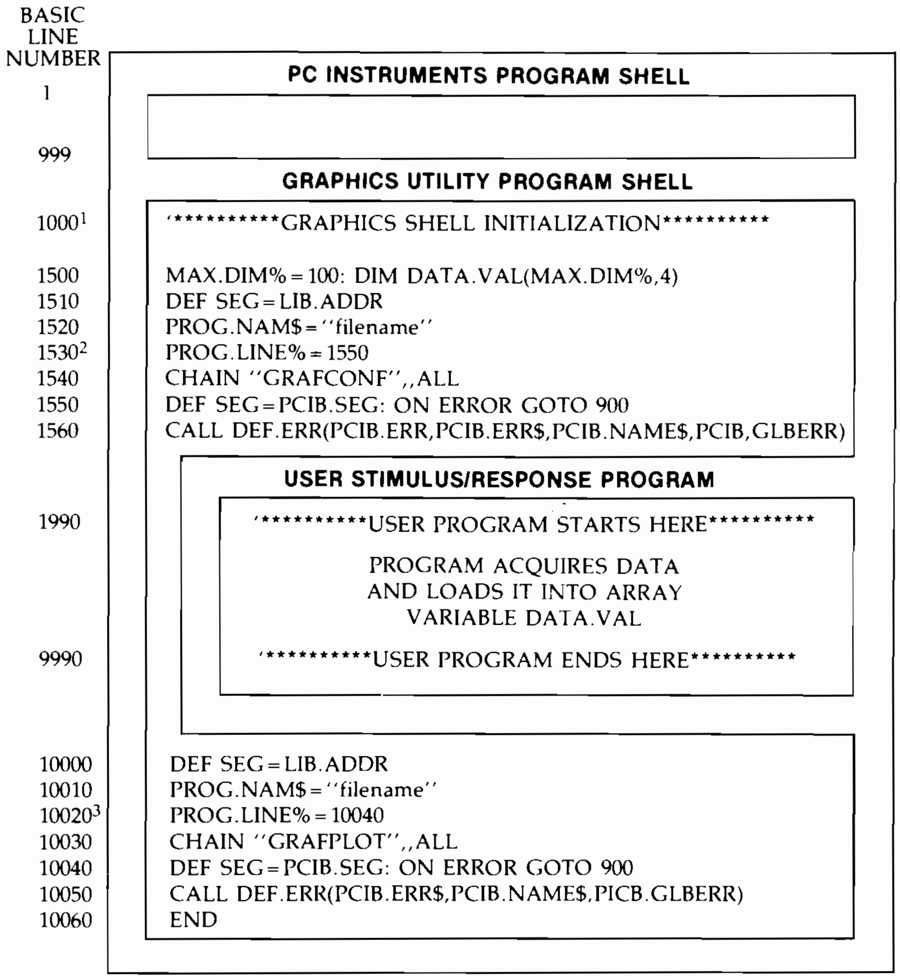
4. Now enter the BASIC commands for the stimulus/response loop that uses the PC Instruments to acquire data and load it into data array DATA.VAL. Please consider the following important points when you write these commands:

   - Your program segment *must use line numbers between 2000 and 9990*. If you use the AUTO command, begin numbering at line 2000 and increment by 10:

     ```
     AUTO 2000,2000,10
     ```

   - If you use the RENUM(ber) command, always start from line 2000 and work up. Do not renumber either the PC Instruments Program Shell or the graphics utility Program Shell.
   - You may also use syntax from the PC Instruments HP-IB I/O Library software in your program segment.

5. Link to the data array. You *must* use the array variable name DATA.VAL. Tables 6-2 and 6-3 show two ways to load the data variable. Use the format of Table 6-2 if you have three (or two) response values for each stimulus value. If you are plotting only a single response against its stimulus, you can get better graphics resolution by reversing the X and Y-axes, as shown in Table 6-3.


BASIC LINE NUMBER
PC INSTRUMENTS PROGRAM SHELL
GRAPHICS UTILITY PROGRAM SHELL
USER STIMULUS/RESPONSE PROGRAM

```
1
999
1000[1]   '**********GRAPHICS SHELL INITIALIZATION**********

1500      MAX.DIM%=100: DIM DATA.VAL(MAX.DIM%,4)
1510      DEF SEG=LIB.ADDR
1520      PROG.NAM$=''filename''
1530[2]   PROG.LINE%=1550
1540      CHAIN ''GRAFCONF'',,ALL
1550      DEF SEG=PCIB.SEG: ON ERROR GOTO 900
1560      CALL DEF.ERR(PCIB.ERR,PCIB.ERR$,PCIB.NAME$,PCIB,GLBERR)

1990        '**********USER PROGRAM STARTS HERE**********

                    PROGRAM ACQUIRES DATA
                    AND LOADS IT INTO ARRAY
                      VARIABLE DATA.VAL

9990        '**********USER PROGRAM ENDS HERE**********

10000     DEF SEG=LIB.ADDR
10010     PROG.NAM$=''filename''
10020[3]  PROG.LINE%=10040
10030     CHAIN ''GRAFPLOT'',,ALL
10040     DEF SEG=PCIB.SEG: ON ERROR GOTO 900
10050     CALL DEF.ERR(PCIB.ERR$,PCIB.NAME$,PICB.GLBERR)
10060     END
```

[1] First line following PC Instruments Program Shell *must* be line 1000.

[2] PROG.LINE% in 1530 *must* reference line immediately following CHAIN ''GRAFCONF'' command.

[3] PROG.LINE% in 10020 *must* reference line immediately following CHAIN ''GRAFPLOT'' command.

Figure 6-14. BASIC Workspace Map for Stimulus/Response Application Program

6. Save the entire stimulus/response test program *to the same drive* that has the graphics utilities files. Using the same filename that you specified in Step 3, type:

SAVE ''B: Filename''
and press Enter

Unless you specify the full drive designator and directory path, the file will be stored to the default drive. The BASIC workspace you stored now includes the PC Instruments Program Shell, an edited version of the graphics utility Program Shell, and the actual stimulus/response program statements.

Table 6-2. Structure of DATA.VAL Containing Three Response Values

DATA.VAL Column	Variable	Where Plotted
DATA.VAL(loop.index%,1)	STIMULUS	X-axis
DATA.VAL(loop.index%,2)	RESPONSE1	Either Y-axes
DATA.VAL(loop.index%,3)	RESPONSE2	Either Y-axes
DATA.VAL(loop.index%,4)	RESPONSE3	Either Y-axes

Table 6-3. Structure of Single-Response DATA.VAL with Reversed Axes

DATA.VAL Column	Variable	Where Plotted
DATA.VAL(loop.index%,1)	RESPONSE	X-axis
DATA.VAL(loop.index%,2	STIMULUS	Either Y-axes
DATA.VAL(loop.index%,3)	(Null)	(Not used)
DATA.VAL(loop.index%,4)	(Null)	(Not used)

Storing DATA.VAL as a Data File

You can store the contents of the data array DATA.VAL as a data file for future retrieval and plotting. Do this by including your own BASIC programming statements within the application program. Insert the needed program lines after DATA.VAL has been loaded with the appropriate data, but before the program CHAINs to

''GRAFPLOT'' in the graphics utility Program Shell (as described under ''Structure of the Graphics Utility's Program Files''). Chapter 7 gives details for making your BASIC program write data to, and read data from, data files.

NOTE

The program lines that write data from DATA.VAL to your data file and read data from the file into DATA.VAL must include identical file formats. Data must be stored in sets of {X,Y1,Y2,Y3} even if you are not using Y2 and Y3.

Running a BASIC Program with the Graphics Utility

Before running your BASIC program that now includes CHAIN(ing) to the PC Instruments graphics utility, check the following:

- If you will be using a graphics line printer, specify it as the default system line printer. Make sure the printer is turned on and has paper properly installed.
- If you will be using a HP7470A or HP7475A Plotter, with an RS-232 interface then connect it to RS-232 communications port No. 1 on your Computer. Make sure that the port is configured for 9600 baud without parity. Be sure the plotter is turned on with properly installed paper and color pens.
- If you will be using the PC Instruments, then make sure the power connections, bus connections, and application connections are properly made and that every device has power applied.
- BASICA, the PC Instruments System Software, the PC Instruments graphics utility software, your PC Instruments state file, and your BASIC application program file must all be resident on either a fixed disk or on the two work diskettes in the dual diskette drives.

or on the two work diskettes in the dual diskette drives.

You are now ready to run the program from BASIC. With your program file loaded into the BASIC workspace type:

RUN
and press Enter

When your program chains to ''GRAFCONF,'' it will display the menus that are required to define your graphics templates. The rest of this chapter gives an introduction to these menus. For more detailed explanations, consult the documentation supplied with your Data Aquisition Software product.

The Graphics Template

The graphics utility program of the Data Acquisition Software plots the data contents of the array variable, DATA.VAL, onto a graphics template that you create from the keyboard. You can define any number of graphics templates through the use of two friendly menus, the Axes Menu and the Annotate Menu. Once stored, a graphics template may be retrieved at any time to display new sets of data from DATA.VAL. The program to build the graphics template resides in the ''GRAFCONF'' program file.

A Softkey Label Map for the Graphics Template Menus. Figure 6-15 is the softkey label map for the Axes and Annotate Menus. Use these two menus to create and store your graphics templates. Table 6-4 briefly describes the softkey labels in alphabetical order.

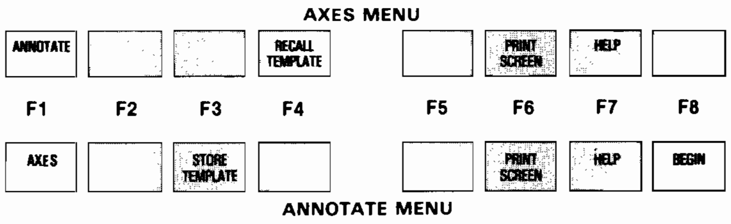


Figure 6-15. Softkey Label Map for Graphics Template

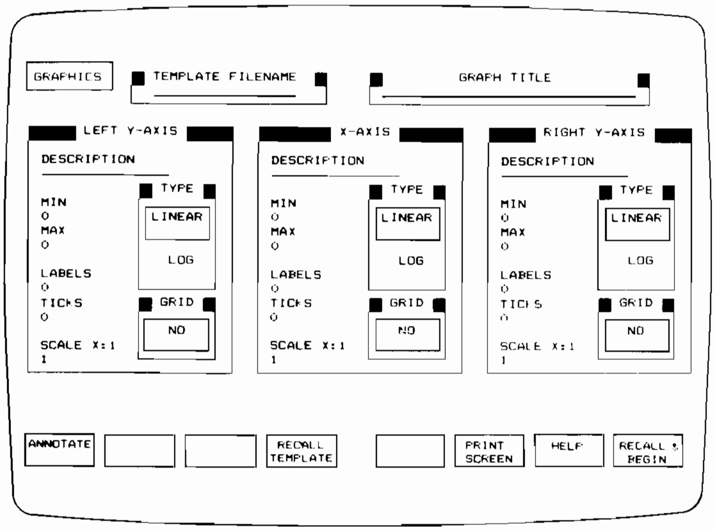
Table 6-4. Softkeys for the Axes and Annotate Menus

Softkey	Function
ANNOTATE (F1)	Changes display from Axes Menu to Annotate Menu.
AXES (F1)	Changes display from Annotate Menu to Axes Menu.
BEGIN (F8)	Continues the program in the BASIC workspace. Press **BEGIN** when you are ready to exit from the graphics template program ''GRAFCONF'' and display the template.
HELP (F7)	Displays a single-line help message that concerns the field where the cursor is currently positioned. Repeatedly pressing **HELP** will scroll, one line at a time, through the entire message for that field. To terminate the help message, press the [Enter] key.
PRINT SCREEN (F6)	Dumps the entire screen contents of your personal computer to your system graphics line printer. Press **PRINT SCREEN** to obtain a hardcopy of either the Axes or Annotate Menu.
RECALL TEMPLATE (F4)	This softkey appears only on the Axes Menu. Use the softkey to load all entry fields for both the Axes and Annotate Menus from the file specified in the TEMPLATE FILENAME field.
STORE TEMPLATE (F3)	This softkey appears only on the Annotate Menu. Use the softkey to store all entry fields for both the Axes and Annotate Menus to the file specified in the TEMPLATE FILENAME field.

The Axes Menu. When your BASIC program CHAIN(s) to ''GRAFCONF'', the Axes Menu (see Figure 6-16) is the first to appear on the screen of your personal computer. This menu enables you to name the graphics template file, title your graph, and fully describe each axis. Besides the X-axis, you can elect to plot one or two Y-axes along the left and right sides of your graph. The selection of Y-axes depends on your application.

Once it has been stored, it is easy to recall a graphics template from the Axes Menu. This occurs whenever you rerun your program. When the Axes Menu appears with its cursor positioned in the graphics template file name field, type your template file name and press the **RECALL TEMPLATE** softkey. This retrieves the graphics template file and loads every field from the Axes and Annotate menus.

Table 6-5 shows the individual entry fields for the Axes Menu. These entry fields are given in the order that you will use them. Use the [Enter] key to enter your selection and move the cursor to the next group of fields. For further details about these entries, refer to the Data Acquisition Software Reference Guide.


GRAPHICS
TEMPLATE FILENAME
GRAPH TITLE

LEFT Y-AXIS
DESCRIPTION
MIN
0
MAX
0
LABELS
0
TICKS
0
SCALE X:1
1
TYPE: LINEAR / LOG
GRID: NO

X-AXIS
DESCRIPTION
MIN
0
MAX
0
LABELS
0
TICKS
0
SCALE X:1
1
TYPE: LINEAR / LOG
GRID: NO

RIGHT Y-AXIS
DESCRIPTION
MIN
0
MAX
0
LABELS
0
TICKS
0
SCALE X:1
1
TYPE: LINEAR / LOG
GRID: NO

Figure 6-16. Example of Axes Template for the Graphics Utility.

Table 6-5. Entry Fields for The Axes Menu

Field	Function
TEMPLATE FILENAME:	The DOS file name under which the graphics template is stored. This file name must follow the rules for all DOS files as given in your Disc Operating System manual. Unless a drive designator and directory path are specified, this file will be stored to the default drive. Exit this field by pressing Enter .
GRAPH TITLE:	The title of the plotted graph. It often is useful to include a date in your title. The title will appear centered above the actual graph on the screen. Exit by pressing Enter .
(Axes Fields)	Because the entry fields for the Left Y-axis, X-axis, and Right Y-axis are identical, each field is described only once. You need use only the X-axis and one Y-axis. The use of the other Y-axis is optional (it may be eliminated by leaving its MIN and MAX entry fields at zero).
DESCRIPTION:	This field lets you enter a descriptive name for each axis. Y-axis descriptions are printed vertically along either the left or right sides of your graph. The X-axis description is printed horizontally and centered under the axis. This field is often useful for entering the physical units of measurement. Exit by pressing Enter .
MIN:	The minimum value plotted on the axis. Exit this field by pressing Enter . For a unipolar axis, value can be zero. For a bipolar axis, value is negative. For expanded-scale axis, value is negative or positive, depending on the application. (See TYPE for logarithmic axis application.)
MAX:	The maximum value plotted on the axis. If both MIN and MAX values are zero, then the axis is deleted from the graph. Exit by pressing Enter . For expanded-scale axis, value is negative or positive, depending on the application. (See TYPE for logarithmic axis application.)

Table 6-5. Entry Fields for The Axes Menu (Cont)

Field	Function
LABELS:	Defines the intervals where labels appear on the axis. A label will be centered around its TICKS value (see Figure 6-17). Labels cannot be specified for a logarithmic axis. Exit by pressing the [Enter] key.
TICKS:	Defines where tick marks appear on the axis. Marks will appear at the value and at integral multiples of the value (see Figure 6-17). Tick marks cannot be put on a logarithmic axis. Exit this field by pressing [Enter].
SCALE:	Lets you plot scale measurement units for plotting (this is described in the PC Instruments Data Acquisition Software Reference Guide). Use the default value (1:1). Exit by pressing [Enter] .
TYPE:	Lets you select either a linear or logarithmic scale for any axis. Pressing [Tab] toggles the field between these two choices. If you specify LOG, then any previous positive entries for MAX and MIN will be rounded (up or down) to the power of 10. Any previous TICKS and LABEL entries will be ignored. The axes will be labeled with powers of 10 and tick marks scaled between the labeled points. Exit by pressing [Enter] .
GRID:	Lets you select either a clear graph or one with grid marks drawn at each tick mark. Pressing [Tab] toggles the field between these two choices. For better results, use a grid on only one axis. Exit by pressing [Enter] .

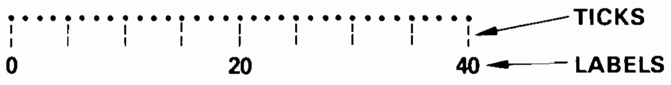


Figure 6-17. Example for TICKS = 5 and LABEL = 20

The Annotate Menu. Once you have completed the Axes Menu, press the **ANNOTATE** softkey to display the Annotate Menu (see Figure 6-18). This menu enables you to enter the specifications for up to three lines plotted on the graphics template. Each line can be plotted with respect to either of the Y-axes (Left or Right). Table 6-6 shows the relationship between the plotted variables and the data elements in DATA.VAL for each plotted line.

Table 6-7 shows the individual entry fields for the Annotate Menu. These entry fields are given in the order that you will use them. Since the fields under "Line #1", "Line #2", and "Line #3" operate identically, each field is described only once. For further details about these entries, refer to the Data Acquisition Software Reference Guide.

Table 6-6. Relationship of Line Plot Variables to DATA.VAL

Line No.	Variable Plotted	Data Array Elements	
		Y Variable Data	X Variable Data
1	Y1 vs. X	DATA.VAL(loop. index%,2)	DATA.VAL(loop. index%,1)
2	Y2 vs. X	DATA.VAL(loop. index%,3)	DATA.VAL(loop. index%,1)
3	Y3 vs. X	DATA.VAL(loop. index%,4)	DATA.VAL(loop. index%,1)

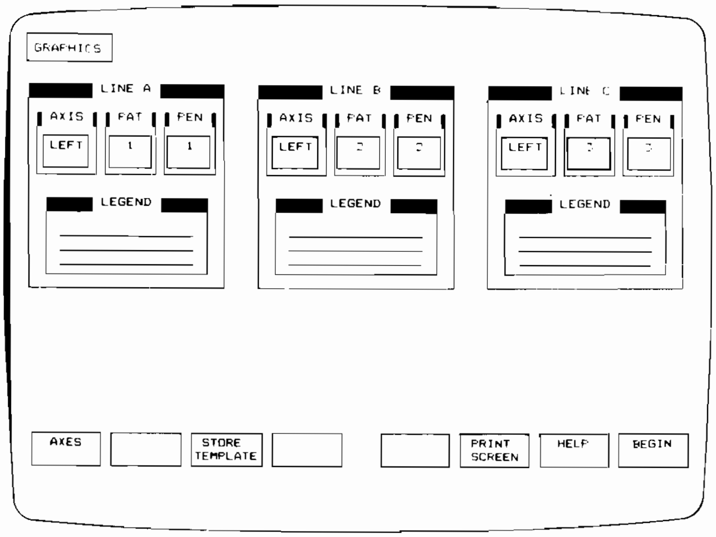


Figure 6-18. Example of Annotate Template for the Graphics Utility

How to Exit the Graphics Template Menus. When you have completed both the Axes Menu and the Annotate Menu to your satisfaction, store the template by pressing **STORE TEMPLATE** softkey. If you fail to store the template, then you will have to re-enter each field the next time you run this program.

You can now continue your BASIC program to acquire, load the DATA.VAL array, and plot the data. To do this, press the **BEGIN** softkey.

Plotting with the Graphics Utility

Your BASIC application program file will be brought into the BASIC workspace and array DATA.VAL will be loaded. Ultimately, your program will CHAIN to the program ''GRAFPLOT''. ''GRAFPLOT'' loops through the data sets X,Y1,Y2,Y3 in the data array (DATA.VAL) and plots each data set onto the graphics template currently entered through ''GRAFCONF''.

Table 6-7. Entry Fields for the Annotate Menu

Field	Function
AXIS:	This field lets you select the Y-axes (LEFT or RIGHT) against which data is plotted. If you select NONE, then this line is eliminated from your graph. Step through the three choices by pressing the [Tab] key. Exit the field by pressing [Enter] .
PAT:	This field allows you to select one of four different patterns for the plotted line. Step through the selections by pressing [Tab]. Exit by pressing [Enter] . 0 - Plot only the data points with no connecting line. 1 - Connect data points with a solid line. 2 - Connect data points with a dashed line. 3 - Connect data points with a dotted line.
PEN:	Use this field to specify which pen (out of 1 to 6) will draw the line. It is up to you to ensure that each number represents the proper pen color or line thickness. Step through the selections by pressing [Tab] . If you specify a number for which no pen exists, then the default pen will be used. Pen No. 1 is used for the graphics template. Exit by pressing [Enter] . **NOTE** *The HP Model 7470A has two pens; the HP 7475A has six.*
LEGEND:	Use this field to label each of your plotted lines. The legend appears below the graph. You can use LEGEND to subtitle each of the plots within the context of the GRAPH TITLE that you entered in the Axes Menu. For example, if your graph is titled ''Engine Temperature vs RPM'' then your LEGENDS might read: (Line #1) ''Engine Block Temperature'', (Line #2) ''Engine Water Temperature'', (Line #3) ''Engine Oil Temperature''. Exit by pressing [Enter] .

''GRAFPLOT'' supports three output devices. The program first plots your graph on the screen of your personal computer. You then have the option of dumping the graph to your system graphics line printer or to a HP Model 7470A or HP7475A Plotter for hard copy. The plotter option gives the best graphics resolution and can plot each line with a different color. Figure 6-8 shows a sample graph from the Data Acquisition Software graphics utility.

Plotting to the Personal Computer Screen. The ''GRAFPLOT'' program first plots your graph on the screen of your personal computer. From the Annotate Menu, press the **BEGIN** softkey. After a short delay, the computer screen will clear and the specified graphics template will be drawn. The following step-by-step procedures refer to softkey labels that appear with the graphics template drawn on the screen.

- Press the **START** softkey to plot the current contents of the data array (DATA.VAL) on the visible graphics template.
- To terminate the plotting at any time, press **STOP** . The softkey legend will change to **START** . Touch it to start the plot again from the beginning.
- To exit the program ''GRAFPLOT'' and continue your BASIC application program to its end, press **RETURN** . You may also halt your BASIC program with a soft reset [Ctrl] [Break] , but this will place the BASIC prompt within the graphics template.

Hardcopy with the System Graphics Line Printer. Make sure the printer is turned on and has paper. With the graph plotted on the screen of your personal computer:

- Press the **PRINT SCREEN** softkey to dump the contents of the screen to the system graphics line printer.

- When the dump is completed, you can end your application program by pressing **RETURN** .

Hardcopy with a Plotter. Make sure your printer is turned on and has the proper paper and pens. With the graph plotted on the screen of your personal computer:

- Press the **GRAPH TO PLOTTER** softkey to dump just the template to the plotter.
- Press the **DATA TO PLOTTER** sotkey to overlay the data on the template.
- When the dump is completed, you can end your application program by pressing **RETURN** .

7 Using HP PC Instruments with Other Applications Software

How this Chapter is Organized

This Chapter explains how you can integrate the data acquired from your HP PC Instruments System with "third party" (ISV) software applications packages (e.g Lotus 1-2-3) to accomplish specific tasks. You are given a detailed description of how file compatibility is achieved through three data formats— Stripped ASCII, BASIC and Data Interchange Format (DIF). This is followed by a complete example illustrating the use of such applications as Lotus 1-2-3 and NWA STATPAK (referred to as STATPAK throughout this chapter). Before you read this chapter, you should be familiar with how to use your third-party application package (refer to its manual for complete operating details).

General Information

In general, there is a strong overlap between the file formats given in this chapter and the formats discussed with each individual application package. STATPAK, for example, has its own Convert Utility that strips out the commas between the data fields and the quotation marks around strings. As explained further in this chapter, this is the same as creating a Stripped ASCII formatted file from a BASIC formatted file. You have purposely been provided with this overlap of features because of the differences in style between each application package. The overlap gives you an even greater amount of file formatting flexibility and helps you choose the method that best suits your needs.

File Compatibility

The data you have acquired from your PC Instruments System can be transported between application software packages by using common data formats. To do this, you should have first gone through program development (as explained in Chapter 5) and done the following:

- developed an application program for your test system
- run your program and recorded and stored the resultant data

The data acquired and stored must be formatted to be directly compatible with such useful application packages as Lotus 1-2-3, STATPAK and VisiCalc. The three common formats that assure file compatibility are BASIC, Stripped ASCII and Data Interchange Format (DIF). A sample program for writing your data in the BASIC format and the Stripped ASCII format is given in this chapter, together with complete details of each. The DIF format is more complex to generate. The PC Instruments System software, therefore, provides you with a utility program that you invoke from DOS to create a DIF formatted file. Each of these data formats support the integrated software solution. That is, they provide you with a smooth, easy method of porting data between many different application packages for the complete solution.

The following paragraphs will review the Stripped ASCII and BASIC file structures and show how you produce compatible diskette data files from your BASIC program.

NOTE

*In general, when creating a formatted data file, each column **must** have the same number of data values.*

Stripped ASCII Format

A Stripped ASCII formatted file can be transported to many applications including Lotus 1-2-3 (Table 7-1 lists some applications packages that use the Stripped ASCII format). This formatted file is a row-column array, where each row is a record and each column is made up of data fields. The data fields are separated by spaces and each record is separated by a carriage return/line feed (<CR><LF>). You cannot, therefore, have blank fields. There must be some character in each column position.

A Stripped ASCII formatted file is quite simple to produce. It is a standard format with only spaces separating each data field. This means that you do not have to insert any other delimiter, such as a comma, between each item in a row. In a Stripped ASCII format, there is no distinction between strings and numeric values. That is, there are no quotes around strings. In summary, the format has:

- strings are *not* enclosed in quotation marks ('')
- fields are separated by spaces
- records are separated by <CR><LF>

To illustrate how to create a Stripped ASCII formatted file, assume you have previously run your program and stored your numeric data in three equally DIMensioned data arrays— DATA1, DATA2 and DATA3. Each array becomes a column in the stripped ASCII formatted data file. To show how string values are handled, each column is labeled with a string expression.

```
10 OPEN ''O'',#3,''DATAFILE''
20 A$=''Measurement'' : B$=''Temp'' : C$=''Volts''
30 PRINT #3, A$,B$,C$
40 ROW =10
50 FOR I=1 TO ROW
60 PRINT #3, DATA1(I),DATA2(I),DATA3(I)
70 NEXT I
80 CLOSE #3
```

This program code creates the Stripped ASCII formatted output file, DATAFILE. For this example, the output of the file is:

```
Measurement    Temp    Volts    *
     1          22       5
     2          32      11
     3          10       6
     4          45       3
     5          55      12
     6          53      20
     7          33       9
     8          50       8
     9          16      17
    10          40      31
    EOF
```

* Although not visible, there is a <CR><LF> after each line.

SUMMARY:

ROW -	Record (case, dataset, etc.)
COLUMN -	Field (variable, datapoint, etc.)
SPACE -	Field delimiter
<CR><LF> -	Record delimiter (carriage return/line feed)
EOF -	End of File (Control-Z). Indicates end of file to the program. Inserted by operating system when it writes a file to diskette.

BASIC Format

A BASIC formatted file is similar to a Stripped ASCII formatted file (as previously described). Both are row-column arrays, where each row is a record and each column is a data field. Each column of data values corresponds to a BASIC array.

A BASIC formatted file is slightly more difficult to produce than a Stripped ASCII formatted file. When writing data in the BASIC format, you must include a comma as the delimiter between data fields. Also, in the BASIC format, strings are distinguished from numeric values by quotation marks. In summary, you *must* explicitly specify the following when creating a BASIC formatted file:

- strings are enclosed in quotation marks ('')
- fields are separated by commas (,)
- records are separated by <CR><LF>

To illustate how to create a BASIC formatted file, assume you have previously run your program and stored the acquired numerical data in three equally DIMensioned data arrays (DATA1, DATA2, and DATA3). Since numeric values and strings are written differently in the BASIC format, this example has been divided into two parts. The first produces a BASIC formatted file containing numeric values only. The second produces a BASIC formatted file containing both numeric values and strings.

1. **Example of Numeric Values Only.** The following program code produces the BASIC formatted numeric output file DATAFILE.

```
10 OPEN ''O'',#3,''DATAFILE''
20 ROW = 10
30 FOR I=1 TO ROW·
40 PRINT #3,DATA1(I);'','';DATA2(I);'','';DATA3(I)
50 NEXT I
60 CLOSE #3
```

The commas within the quotation marks serve as delimiters that separate each data value in a row and will appear in the actual output file. For this example, the output file is:

```
1 , 12 , 2      *
2 , 54 , 7
3 , 11 , 6
4 , 34 , 1
5 , 22 , 12
6 , 14 , 10
7 , 12 , 7
8 , 55 , 24
9 , 23 , 26
10 , 9 , 3
EOF
```

* Although not visible, there is a <CR><LF> after each line.

SUMMARY:

ROW -	Record (case, dataset, etc.)
COLUMN -	Field (variable, datapoint, etc.)
COMMAS -	Field delimiter
<CR><LF> -	Record delimiter (carriage return/line feed)
EOF -	End of File (Control-Z). Indicates end of file to the program. Inserted by operating system when it writes a file to diskette.

2. Example of Numeric and String Values. The following program code productes the BASIC formatted numeric and string output file DATAFILE. CHR$(34) prints the double quote('').

```
10 OPEN ''O'',#3,''DATAFILE''
20 A$=''Measurement'': B$=''Temp'' : C$=''Volts''
30 ROW = 10
40 PRINT#3,CHR$(34);A$;CHR$(34);'','';CHR$(34);B$;
    CHR$(34);'','';CHR$(34); C$;CHR$(34)
50 FOR I=1 TO ROW
60 PRINT #3,DATA1(I);'','';DATA2(I);'','';DATA3(I)
70 NEXT I
80 CLOSE #3
```

When strings are added, the following data file (DATAFILE) is produced:

```
''Measurement'',''Temp'',''Volts''
1 , 12 , 7
2 , 18 , 8
3 , 7 , 9
4 , 22 , 17
5 , 45 , 22
6 , 54 , 17
7 , 33 , 6
8 , 65 , 44
9 , 0 , 2
10 , 10 , 9
```

Note the differences between the Stripped ASCII and BASIC formats. Once you understand both Stripped ASCII and BASIC file formats, it is easy to change your data from one format to the other for specific applications. BASIC formatted data can be transported to such application packages as Lotus 1-2-3 and Condor for spreadsheet and database capabilities. Table 7-1 lists some application packages that use the BASIC format.

DIF Format

The DIF format is widely used for the exchange of data. It is a well defined data file structure that is more complex than either Stripped ASCII or BASIC. The DIF format is widely used by many application packages. VisiCalc, for example, only reads a DIF formatted data file and requires the .DIF file extension. Table 7-1 identifies some application packages that use the DIF format.

DIF puts the data into a table format where fields are called ''vectors'' (often treated as columns) and records are called ''tuples'' (often treated as rows). A DIF formatted file consists of a header section, which defines the table headings, and a data section that contains the actual values.

Using the Convert Utility

You do not have to be concerned about the details of converting a BASIC or Stripped ASCII file to DIF. PC Instruments has a Convert Utility program to do this for you. To use the Convert Utility, you *must* have created a data file in Stripped ASCII or BASIC format (as previously illustrated in this chapter), or already have a DIF file. Invoke the Convert Utility from DOS, by typing:

A> CONVERT *input file/input format output file/output format*

Where:

input file	is the filename containing the data to be formatted.

input format	is the current format of the data in the input file. Specify the format by a code letter as follows:
	A = Stripped ASCII
	B = BASIC
	D = DIF

output file	is the filename to contain the newly formatted data (specify a filename of your choice).

output format	is the format to which the data is to be converted. Specify the format by a code letter as follows:
	A= Stripped ASCII
	B = BASIC
	D = DIF

As examples, this first command takes data from the Stripped ASCII file named INFILE, converts it to the DIF format, and stores the newly formatted file under OUTFILE.

```
A> CONVERT INFILE/A OUTFILE/D
```

This next command converts the DIF formatted data in DATFILE to a BASIC format stored in STORFILE.

```
A> CONVERT DATAFILE/D STORFILE/B
```

Summary of Conversion Types

The following information applies to all conversion types:

1. Each input file row must have the same number of fields.
2. All remark lines in an input file will be ignored. You may use them to document your input file or to provide data that is meaningful for other programs but of no significance to the Convert Utility.
3. The only nonprinting characters permitted in an input file are tabs and line terminator pairs (<CR><LF>). The presence of any other nonprinting character may produce an error.

4. Any errors that occur while CONVERT is running are not recoverable. Once an error is detected, an error message is displayed on the screen, the partially written output file is deleted, and the utility exits to the operating system (DOS).

Here is a summary of what occurs when each of the six possible types of conversion is made:

BASIC-to-Stripped ASCII. The following occurs during this conversion:

1. Numeric fields in the input file are copied unchanged to the output file.
2. Commas separating fields in an input file are changed to single spaces separating fields in the output file.
3. Fields within quotes are stripped of enclosed quotes and delimiters (spaces, tab, and single quotes).
4. An input file having an empty field between quotes or between commas produces an error.
5. Any field that is neither numeric nor a string within quotes produces an error.

Stripped ASCII-to-BASIC. The following occurs during this conversion:

1. Numeric fields in the input file are copied unchanged to the output file.
2. Non-numeric fields are enclosed within quotes and copied to the output file.
3. The tabs or spaces separating fields in an input file will be stripped and commas will separate fields in the output file.

DIF-to-BASIC. In this conversion, the input file header must have the four required items (TABLE, VECTORS, TUPLES, and DATA), which are the only ones used by the Convert Utility. Any other header items (such as LABEL, SIZE, COMMENT, etc.) will be ignored and not create any data in the output file.

DIF-to-Stripped ASCII. This conversion proceeds the same as the DIF-to-BASIC conversion, except that an empty data field in the input file will produce an error.

BASIC-to-DIF. In this conversion, only the four required header items (TABLE, VECTORS, TUPLES, and DATA) are generated and written, in that order, to the output file. An input file with an empty field between commas produces an error.

Stripped ASCII-to-DIF. This conversion proceeds the same as the BASIC-to-DIF conversion.

Example 1A - Using PC Instruments with Lotus 1-2-3

To illustrate the usefulness of integrating your PC Instruments data with Lotus 1-2-3, we will expand the example of Programmed Instrument Control (given in Chapter 5). We will acquire data from that example, store it, transfer it to a Lotus 1-2-3 spreadsheet, and then plot it.

Acquiring and Storing the Test Data

Figure 7-1 shows the program for stepping a Dual Voltage DAC through ten voltage values, reading those values with a Digital Multimeter, and storing them in an array variable. The data is then written to a diskette file in Stripped ASCII format so that it can be directly transported to Lotus 1-2-3.

For this example, the data created and stored on the diskette file (TESTRSLT) is:

1	9.32	6	58.68
2	22.10	7	72.68
3	29.03	8	81.08
4	41.28	9	87.67
5	51.37	10	98.67

```
1       <Program Shell>
999     '
1000    ' User application starts at this line
1010    DIM VALS(10)
1020    FILE$="STATE1"
1030    ' Initialize all the instruments in the system to
1040    ' the previously stored state specified by
        'FILE$. The outputs of TESTV__IN and
1050    ' TEMPSET are enabled because they were
        stored ENABLED.
1060    CALL INITIALIZE.SYSTEM(FILE$)
1070    ' Wait for the oven temperature to stabilize
1080    PRINT "Press any key to continue"
1090    A$ = INKEY$ : IF A$ = ""THEN 1090
1100    ' Step the voltage of the DAC and take a
        reading
1110    FOR VOLTS = 1 TO 10
1120    CALL OUTPUT(TESTV.IN,VOLTS)
1130    CALL MEASURE(TESTV.OUT,VALS(VOLTS))
1140    NEXT VOLTS
1150    ' Create a Stripped ASCII formatted file for
        Lotus 1-2-3
1160    OPEN "O",#3,"TESTRSLT.PRN"
1170    ROW = 10
1180    FOR I = 1 TO ROW
1190    PRINT #3,I,VALS(I)
2000    NEXT I
2010    CLOSE #3
2020    END
```

Figure 7-1. Program for Acquiring Data and Storing it in Stripped ASCII Format.

Using Lotus 1-2-3 to Display the Test Data

Assuming that you are now in Lotus 1-2-3, you can use its menu commands to transfer the data to a spreadsheet, create a graph from the spreadsheet, store the graph, and plot the result.

1. Transfer the data with the Import command (Figure 7-2 shows the resultant spreadsheet):

```
F(ile)    I(mport)    N(umbers)
  Enter name of file to import: TESTRSLT
```

	A	B	C	D	E	F	G
1	1	9.32					
2	2	22.10					
3	3	29.03					
4	4	41.28					
5	5	51.37					
6	6	58.68					
7	7	72.68					
8	8	81.08					
9	9	87.67					
10	10	98.67					
11							
12							
13							
14							
15							
16							
17							
18							
19							

Figure 7-2. Spreadsheet of Example 1A Test Data.

2. Specify the graph parameters:

```
G(raph)  T(ype)  L(ine)  X
  Enter X-axis range: A1.A10
    A
  Enter first data range: B1.B10
```

3. Label the graph:

   O(ptions) T(itles) F(irst)
     Enter graph title, top line: Example 1A - Lotus 1-2-3

   T(itles) X(-Axis)
     Enter X-axis title: TEST READING

   T(itles) Y(-axis)
     Enter Y-axis title: VOLTS

   Q(uit)

4. Save the graph in a file named "TESTPLOT"

   S(ave)
     Enter graph file name: B:TESTPLOT

5. Save the spreadsheet in a file (WORKSHEET)

   F(ile) S(ave)
     Enter save filel name: B:WORKSHEET

6. Return to the Access Menu (or DOS):

   Q(uit) Y(es)

7. After quitting 1-2-3, you will be in the Access System Menu. Follow the prompts to insert the PrintGraph diskette in the correct drive and obtain the PrintGraph menu.

NOTE

If you are in the operating system, then obtain the PrintGraph menu by typing:

A> GRAPH
and pressing [Enter]

---

8. Make the following entries from the PrintGraph menu:

   S(elect)
   TESTPLOT
   G(o)

Figure 7-3 shows the resultant plot.

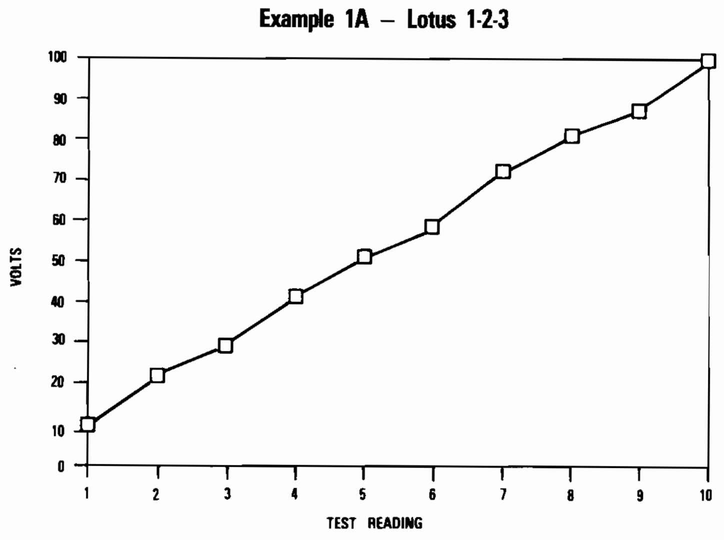


Figure 7-3. Graph of Example 1A Test Data.

Example 1B - Using PC Instruments with STATPAK and Lotus 1-2-3

This example is an expansion of Example 1A to illustrate the power of integrating one set of data with two application packages. STATPAK is first used to fit the acquired data along a straight line. The data is then imported into Lotus 1-2-3 for graphing (refer to Example 1A for complete details about using Lotus 1-2-3 for graphing).

NOTE

*You **cannot** use BASICA with STATPAK. Load standard BASIC before using STATPAK.*

Sequence of STAKPACK Commands

From BASIC type:

```
RUN "ONEVREG"
 and press [Enter]
```

To fit the data along a straight line, respond to the STATPAK menu as follows:

```
Maximum - of data pairs? 10
Linear(Y/N)? Y
Exponential(Y/N) N
Logarithmic(Y/N) N
Power Curve(Y/N) N

Input file : B:TESTRSLT.PRN
Field assignments(s): [Enter]
Output: F(ile)
Output: B:CURVFIT.PRN
Output: Q(uit)
```

Return to DOS by typing:

```
S(ystem)
```

Now you can integrate this data with Lotus 1-2-3 for graphing. Exit STATPAK, run Lotus 1-2-3, and import the data from STATPAK by typing:

```
F(ile)   I(mport)   N(umbers)

Enter name of file to import: B:CURVFIT.PRN
```

After the data is imported, the Lotus 1-2-3 spreadsheet appears as shown in Figure 7-4.

	A	B	C	D	E	F	G	H
1	1	10.87254	0	0	0	9.32	-1.55253	0
2	2	20.72042	0	0	0	22.1	1.379583	0
3	3	30.5683	0	0	0	29.03	-1.53829	0
4	4	40.41618	0	0	0	41.28	0.863822	0
5	5	50.26406	0	0	0	51.37	1.105946	0
6	6	60.11194	0	0	0	58.68	-1.43193	0
7	7	69.95982	0	0	0	72.68	2.720184	0
8	8	79.8077	0	0	0	81.08	1.272308	0
9	9	89.65557	0	0	0	87.67	-1.98557	0
10	10	99.50345	0	0	0	98.67	-0.83345	0
11								
12								
13								
14								
15								
16								
17								
18								
19								

Figure 7-4. Spreadsheet of Example 1B Test Data

NOTE

Column A is the number of the data point (X-axis location).

Column F is the actual data value imported from the test program and plotted in Example 1A (Y-axis value for the data plot).

Column B is the STATPAK value calculated for a point closest to a straight line (Y-axis value for the fitted line).

Column G This value is not needed for the plot shown in this example.

Creating the Graph

You can now graph the spreadsheet data by using the same process previously illustrated in Example 1A . Change the response for the graph title to: Example 1B — STATPAK and Lotus 1-2-3. When you enter the graph parameters, be sure to specify three columns (A, B, and F) for graphing. Figure 7-5 is the plot for this example.

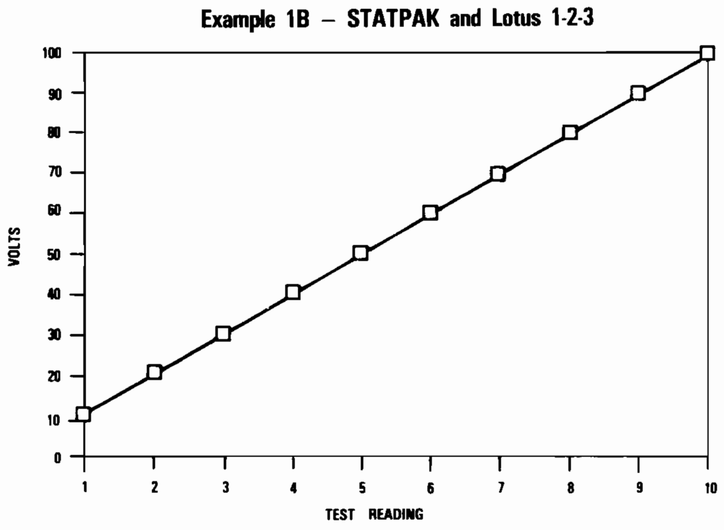


Figure 7-5. Straight Line Graph of Example 1B Test Data.

Summary of Applications Packages

Table 7-1 lists some software packages that you may possibly use to accomplish specific data applications tasks. Due to the numerous software packages available, it is not possible to list all of them. Instead, Table 7-1 gives a sample of some popular application packages and some of the file formats accepted by each.

Table 7-1.* Applications Packages Summary

Application Package	Capability	Stripped ASCII	BASIC	DIF
VisiCalc	Spreadsheet			x
Lotus 1-2-3	Spreadsheet	x (1)(2)	x (2)	x (3)
Picture Perfect	Graphing	x	x	x
Condor	Database		x	
dBASEII	Database	x	x	
STATPAK	Statistics	x	x (4)	
Wordstar	Wordprocessing	x	x	x

1. Lotus 1-2-3 will ignore any string of text not enclosed in double quotes.
2. Lotus 1-2-3 requires that the file has the extension .PRN.
3. Requires use of TRANSLATE utility provided by Lotus 1-2-3 to convert the DIF format to Lotus 1-2-3.
4. Requires use of the conversion utility provided with either STATPAK or with the PC Instruments Data Acquisition Software.

* Table 7-1 is based on information available at the time of this publication.

A Using the HP PC Instruments System with the IBM PCs

System Configuration Switch Settings

When you receive your HP PC Instruments System, you may have to check the configuration switch settings on the computer system board to ensure that they match the computer hardware configuration you are using. The computer system board is the large one mounted on the bottom of the computer. With the unit cover removed (refer to the Options section of your IBM *Guide to Operations*), you will be able to locate the system switch assembly (or assemblies). The *Guide to Operations* has charts that show you how to set the switches for various system configurations.

System Memory Switch Settings

The amount of RAM memory needed for HP PC Instruments System is given in Table 1-1 of Chapter 1. Because the system board does not have enough memory, one or more memory modules or memory expansion cards will have to be installed. Install the cards in any available accessory slots. You must always have the full amount of memory on the system board before you add memory expansion cards. The system board configuration switch must be set to indicate that the full complement of memory is on the system board.

It is not enough to just install memory cards into the computer. Each memory card has a switch that must be set so the computer will ''know'' the starting address for the memory on that card. Refer to your IBM documentation and the manual accompanying the memory card for charts that show how to set memory switches.

Accessory Slot Considerations

The accessory slots that are available to you can vary depending on the type of computer you have. Table 1-1 lists the minimum number of slots as well as the minimum number and type of accessory cards required for the IBM PC, PC/XT, and PC/AT to support a HP PC Instruments System. The IBM PC/XT and PC/AT have enough accessory slots available so that you can satisfy the requirements of the PC Instruments System with various hardware configurations. However, the IBM PC has only four available accessory slots. Therefore, you must use the 384 K Quadram Quadboard, which includes sufficient memory as well as a serial adapter for the mouse, to ensure that there will be enough slots remaining to support the minimum PC Instruments System requirements.

Avoiding Address Conflict

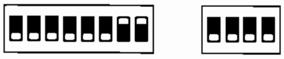
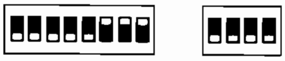
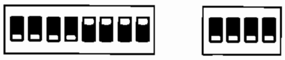
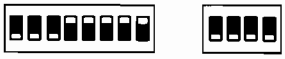
The PC Instruments System Interface Cards have a four digit hexadecimal address. The least-significant digit is hardwired for zero, leaving three digits for a unique address (see Table A-1). The interface card is shipped with its address switch set to 0300H. If you already have some other card at that address, then you will have to change the address on one of the cards. Also, if you install a second interface card, you will have to change its address setting.

Since the interface card's address can be set for any hexadecimal number up to FFF0, it would seem that there should be no problem with conflicting addresses.

However, some option cards designed for IBM PCs do not have a four-digit hexadecimal range. For example, assume you already have a card at 0300H and it has only a three-digit hexadecimal address range. Then, you must not set the address of the PC Instruments Interface Card at any multiple of that address (1300H, A300H, etc) because the other card will also respond to those addresses. This can be regarded as "imaging", because the card responds not only to 300H, but any image (X300H) of the address. Figure A-1 shows a general procedure for avoiding address conflicts when installing a PC Instruments Interface Card.

There is a similar situation when you install more than one PC Instruments Interface Card. When you change an address from 300H to a higher number, be sure that the address is not an image for some other option card with a lower address. If you are able to use 300H for the first card, then subsequent cards may be set to any of the addresses listed in Table A-1

Table A-1. Standard and Three Other Available Addresses

Address	Switch Position
0300H **	 *
0700H	 *
0F00H	 *
1F00H	 *
	16^3 16^2 16^1 16^0

* Least-significant hexadecimal digit is hardwired for 0H.

** Standard (default) switch setting.

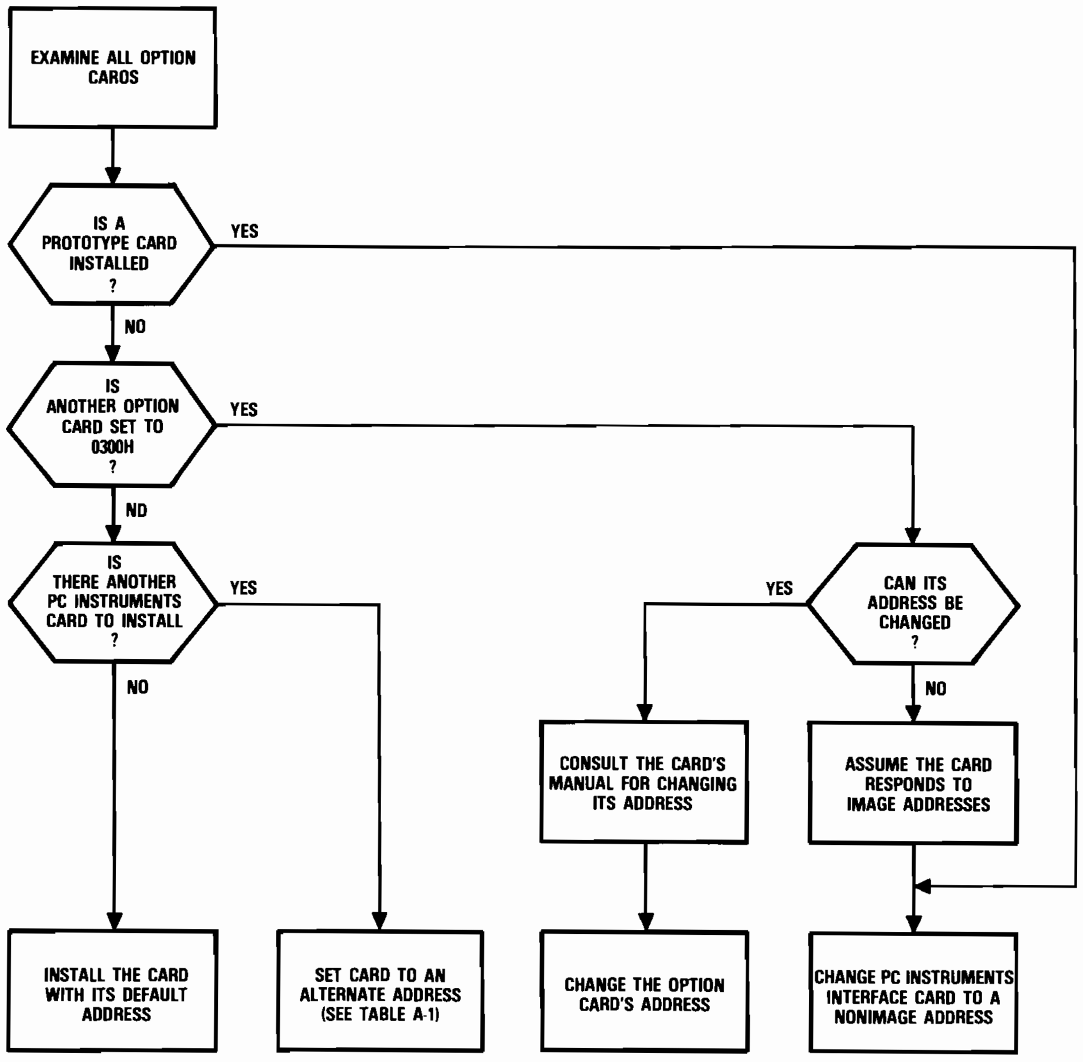

EXAMINE ALL OPTION CAROS
IS A PROTOTYPE CARD INSTALLED ?
YES
NO
IS ANOTHER OPTION CARD SET TO 0300H ?
YES
ND
IS THERE ANOTHER PC INSTRUMENTS CARD TO INSTALL ?
YES
NO
CAN ITS ADDRESS BE CHANGED ?
YES
NO
CONSULT THE CARD'S MANUAL FOR CHANGING ITS ADDRESS
ASSUME THE CARD RESPONDS TO IMAGE ADDRESSES
INSTALL THE CARD WITH ITS DEFAULT ADDRESS
SET CARD TO AN ALTERNATE ADDRESS (SEE TABLE A-1)
CHANGE THE OPTION CARD'S ADDRESS
CHANGE PC INSTRUMENTS INTERFACE CARD TO A NONIMAGE ADDRESS

Figure A-1. General Procedure for Avoiding Address Conflict

Reallocating Memory Space for PC Instrument Data Structures

The Configure Menu (refer to Chapter 3) has an entry for PC Instrument data structure module memory space. This is the total space needed by all the modules connected to the PC Instruments Interface bus and is determined by the types of modules and the quantity of each type. The Digitizing Oscilloscope requires far more memory than any other instrument and a system that has no oscilloscope modules requires far less than the default space.

If the 64 K default space is *not* available in your RAM, then the Soft Front Panel will not load and you will see the operating system error message ''DOS Memory Expansion Fail.'' To reduce the allocated memory below 64 K, enter the Configure Menu and proceed as follows:

1. Answer the first three menu questions the same way as the last time you used this menu.
2. To the memory space question, enter: 48. The menu interprets this as 48 K bytes.
3. Exit the menu and load the Soft Front Panel. If the Soft Front Panel loads, you now have sufficient RAM space. If you still get the original DOS Memory Expansion Fail error, repeat the above procedure in steps of 16 K (enter 32 and last, if required, 16).

NOTE

If you should reduce the memory space below what is actually needed for your instruments data structures, then you will get a PC Instruments System error message (no. 70).

Using the Mouse Device

Figure A-2 shows the two types of mouse device that are supported by the PC Instruments Software. The Microsoft unit is operated on any flat surface and either of its two control buttons may be used to make a selection. The Mouse Systems unit is operated on a graphics pad and has three control buttons. For PC Instruments, the center button is disabled and either of the other two may be used to make a selection. You can continue to point by moving either mouse while holding down the button. The actual selection is not made until you release the button while the cursor arrow is on the desired field.

Some versions of the Mouse Systems device must be calibrated the first time they are used. Refer to the PC Mouse Set Up Guide supplied with the device.

If your have the IBM PC/AT, remember that it requires a special cable for any serial mouse device (such as the Microsoft Mouse). Refer to your IBM and Microsoft documentation.

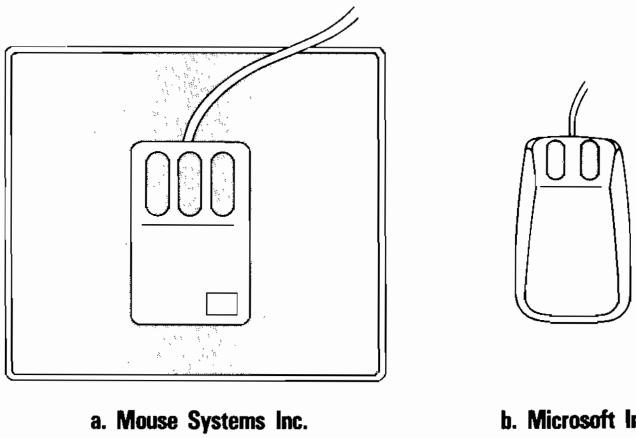


Figure A-2. Typical Mouse Devices Used with PC Instruments

B HP PC Instruments System Verification

Introduction

The HP PC Instruments System Utilities and Verification diskette includes software that can be used to test and calibrate your PC Instruments System. This verification software is a menu driven program. When you run the program, step-by-step instructions appear on your computer's screen to guide you through each part of the verification procedures. These verification procedures give you a high degree of confidence that your PC Instruments System is operating correctly. There are two parts, or levels, to the verification program:

- system verification
- instrument verification (including calibration, if applicable)

This appendix only explains how to use the system-level verification portion of the program. While instrument verification and calibration procedures are also included on this diskette, the details for using that portion of the program are supplied in Appendix B of the applicable Instrument Owner's Guide. You must read this appendix and run the system-level of verification before performing the instrument-level tests.

Equipment Required

The only equipment needed for system-level verification is a properly installed system consisting of a PC Instruments Interface Card, the PC Instruments Bus (PCIB), and at least one instrument module. If you have more than one instrument in your system, install all of them before beginning the system verification. Make sure that all instruments connected to the PCIB are receiving power before you run the verification program.

NOTE

*The instruments used in your test must **not** be connected to any application. Remove any front panel connections before beginning verification.*

Copying and Running the Verification Program

NOTE

Before copying the verification software, you must have already configured the PC Instruments System Software as outlined in Chapter 3 of this manual. If you have not already done this, please turn to Chapter 3 and complete those procedures before continuing with verification.

The PC Instruments verification software is located on the PC Instruments System Utilities and Verification diskette. This software must be copied before performing verification of your system for the first time. If you have already copied the verification software, proceed to ''Running the Verification Program''.

The procedures that you use to copy and run the program depend on the type of diskette drive that you have connected to your computer. Follow the procedures that apply to your situation.

Procedures for Dual Diskette Drives

Copying the Verification Software. Copy the verification software as follows:

1. Format a blank double-sided diskette.
2. Copy the subdirectory \PCIB_VER\ from the PC Instruments System Utilities and Verification diskette to the blank diskette.

3. Copy the file SYS__VER.BAT from the Utilities and Verification diskette to the blank diskette.
4. Copy the BASICA.COM file from the DOS diskette to the same diskette. You can now proceed to run the verification program.

Running the Verification Program. Run the verification program as follows:

1. Insert the system work diskette that you created in Chapter 3 of this manual into the boot drive of your computer.
2. Insert the verification work diskette that you just created into the other drive.
3. Reboot your system by pressing the Alt Ctrl Del keys simultaneously.
4. To begin the system-level verification,

   Type: SYS__VER
   and press Enter

Procedures for One Fixed Disk and One Diskette Drive

Copying the Verification Software. Copy the verification software as follows:

1. Insert the PC Instruments System Utilities and Verification diskette into the diskette drive.
2. Copy the subdirectory \PCIB.VER\ and the file SYS__VER.BAT from the PC Instruments System Utilities and Verification diskette into the \PCIB\ subdirectory on your fixed disk.

Running the Verification Program. Run the verification program as follows:

1. Remove the System Utilities and Verification diskette from the diskette drive.
2. Reboot your system by pressing the Alt Ctrl Del keys simultaneously.
3. To begin the system-level verification,

   Type: SYS__VER
   and press Enter

What System-Level Verification Does

When you run the system-level verification program, the software first reminds you about the installation procedures that you should already have completed (see Table B-1). The program then performs the following three tests:

1. Addresses and checks the PC Instruments Interface Card that you specified in the configuration program.

NOTE

The verification program can only check one PC Instruments Interface card and its associated instruments at a time. If you have a second interface card in your system, you must rerun the verification program to verify the second interface card and the instruments connected to it.

2. Checks the eight instrument addresses available at the interface card that you selected and identifies the instruments connected at each address. This lets you determine if the actual system configuration agrees with what you wanted when you installed the hardware.
3. Sends commands and data to each of the instruments to check its control circuitry and the PC Instruments Bus (PCIB). If the instrument responds correctly to this preliminary test, the program will inform you that the instrument has ''passed'' the system-level verification.

If you experience problems with any of these tests, proceed to ''In Case of Trouble''.

Proceeding to Instrument-Level Verification

The system-level verification tests are completed when the program presents a list of all the instruments it found connected to the interface card that you specified along with their test results. Pressing the **CONTINUE** softkey inititates the instrument-level verification portion of the program. You can now select an instrument that has passed the system-level test for further verification from a menu that is displayed on the left of your computer's screen. Instructions for performing the instrument-level tests are included in Appendix B of the applicable PC Instrument Owner's Guide. At the end of the instrument-level tests, you will be returned to the menu where you can either select another instrument for verification or end the program.

Rerunning the Verification Program

You will need to rerun the verification program if you have more than one interface card in your system or if you perform any of the troubleshooting procedures that are discussed in the following section. To rerun the program you can either:

- press the Ctrl and Break keys simultaneously, type RUN , and press Enter , or
- press the **STOP** softkey in the system program to return to DOS, and type SYS_VER and press Enter .

In Case of Trouble

It is important that you complete the system-level as well as the instrument-level verification tests. If your system does not pass the system-level tests, first consult the troubleshooting procedures that follow. Then, if your system is still not functioning properly, consult your *HP PC Instruments Support Guide* for assistance.

Troubleshooting Procedures

There are three error conditions that can occur as you run the system-level verification. First, the verification program may not be able to locate your PC Instruments Interface

Card. If this is the case, refer to Table B-2 for troubleshooting instructions.

Secondly, the verification program may not properly locate and identify all of the instruments in your system. If this is the case, turn off your system and repeat the System Hardware Check given in Table B-1. Turn your system on and run the verification program again. If you still have an error, see Figure B-1 for additional troubleshooting instructions.

Finally, an instrument may fail the system-level verification tests. If this is the case, turn off your system and repeat the System Hardware Check given in Table B-1. Turn your system on and run the verification program again. If your instrument still fails the test, see Figure B-2 for additional troubleshooting instructions.

Additional Troubleshooting Information

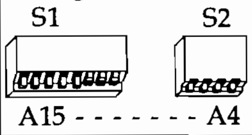
1. If an instrument passes the system-level tests but either fails the instrument-level tests or appears not to work at all, turn off the instrument, connect a different Power Pack to it, and rerun the verification.
2. If an instrument that is connected to the interface cable is not receiving a-c power, unpredictable system operation may occur. If any instrument is not receiving power, remove it from the PCIB.

Table B-1. System Hardware Check.*

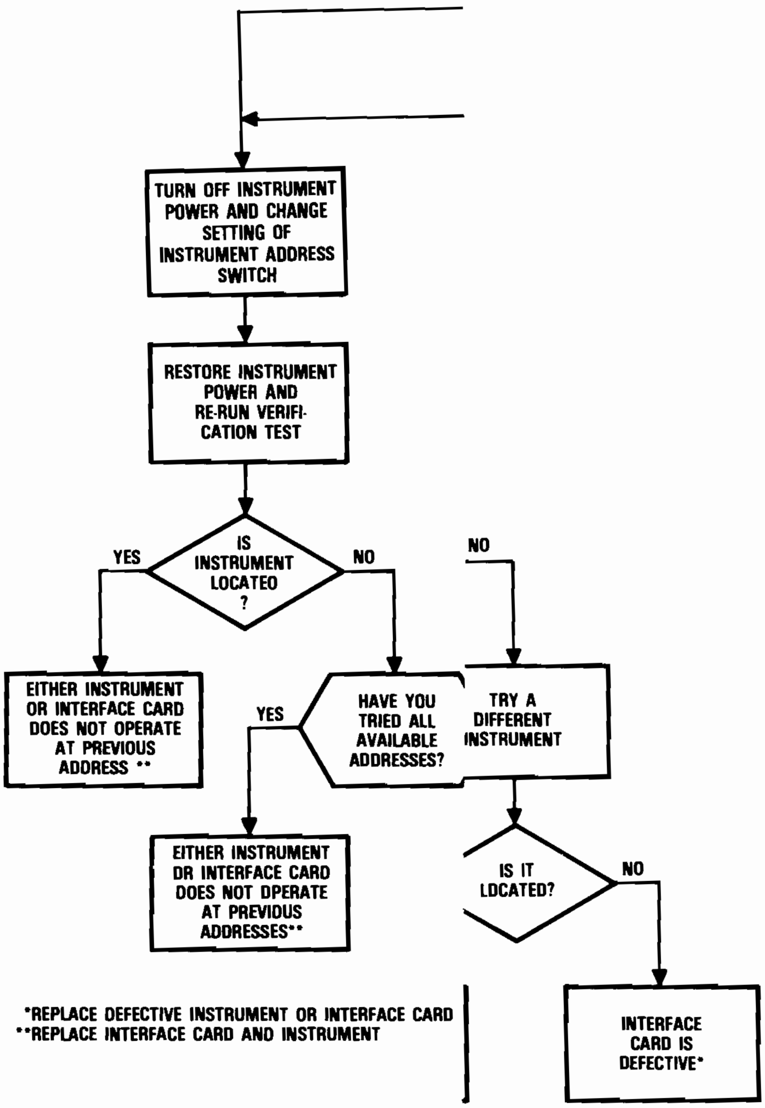
1. Note the address of your PC Instruments Interface Card(s).
2. Check that each instrument has a unique address.
3. Be sure that each instrument is receiving power.
4. Secure all interface and instrument cables.

*Refer to Chapter 2 of this guide for further details.

Table B-2. System Interface Card Troubleshooting

Step	Action	Result
1	Turn off system power. Remove the interface card and examine its address switch.	a. If switch is set to address 0300H, which also must be specified in the Configuration Program, proceed to step 2. b. If switch is not set to 0300H, refer to Chapter 2, Figure 2-4, and set the switch accordingly. Reinstall the interface card, turn on system power and repeat the verification.
2	Set the interface card to address 0700H (see below). You must also specify this new address in the Configur Menu Program 	a. If the interface card is still not located, it is defective. Replace the card. b. If the interface card is located, then it will operate at some but not all addresses.The card should be replaced.*

*If the card address is the same as that for some other device, the resulting bus contention can cause the test to fail or to continue with unpredictable results.



TURN OFF INSTRUMENT POWER AND CHANGE SETTING OF INSTRUMENT ADDRESS SWITCH
RESTORE INSTRUMENT POWER AND RE-RUN VERIFI-CATION TEST
IS INSTRUMENT LOCATED ?
YES
NO
NO
EITHER INSTRUMENT OR INTERFACE CARD DOES NOT OPERATE AT PREVIOUS ADDRESS **
YES
HAVE YOU TRIED ALL AVAILABLE ADDRESSES?
TRY A DIFFERENT INSTRUMENT
EITHER INSTRUMENT OR INTERFACE CARD DOES NOT OPERATE AT PREVIOUS ADDRESSES**
IS IT LOCATED?
NO
INTERFACE CARD IS DEFECTIVE*

*REPLACE DEFECTIVE INSTRUMENT OR INTERFACE CARD
**REPLACE INTERFACE CARD AND INSTRUMENT

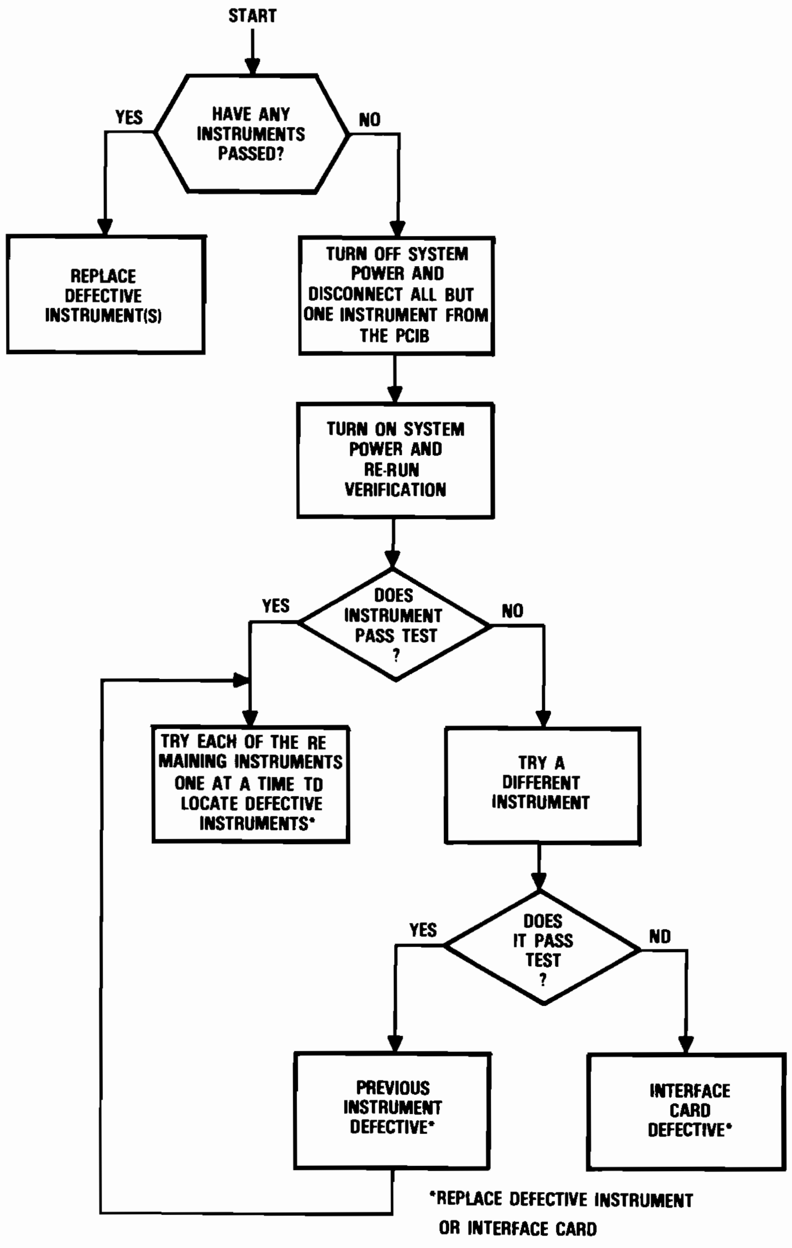

START
HAVE ANY INSTRUMENTS PASSED?
YES
NO
REPLACE DEFECTIVE INSTRUMENT(S)
TURN OFF SYSTEM POWER AND DISCONNECT ALL BUT ONE INSTRUMENT FROM THE PCIB
TURN ON SYSTEM POWER AND RE-RUN VERIFICATION
DOES INSTRUMENT PASS TEST ?
YES
NO
TRY EACH OF THE RE MAINING INSTRUMENTS ONE AT A TIME TO LOCATE DEFECTIVE INSTRUMENTS*
TRY A DIFFERENT INSTRUMENT
DOES IT PASS TEST ?
YES
NO
PREVIOUS INSTRUMENT DEFECTIVE*
INTERFACE CARD DEFECTIVE*
*REPLACE DEFECTIVE INSTRUMENT OR INTERFACE CARD

Figure B-2. System Verification Troubleshooting

C Error Messages

Introduction

This appendix lists the HP PC Instruments System error messages that apply when you operate the PC Instruments System from a BASIC program. These error messages are only returned when you use the error handling routines that are described in Chapter 5 of this guide. All error messages and their corresponding numbers are listed in the PCIBILC.HPE and HPERR.HPE file on your PC Instruments Software Master diskette. The numbers that precede each message are coded to indicate either a system error or a specific instrument that the error message applies to.

System Error Messages

The PC Instruments System error messages are assigned numbers between 0 and 99. Because BASIC error messages also use these same values, a distinction must be made between PC Instruments errors and BASIC errors. If you use the error handling routine in line 900 of your Program Shell, this distinction is made automatically. If you are writing your own error handling routine, refer to Chapter 5 of this guide for the procedure to detect PC Instruments errors. Refer to your BASIC manual for the procedure to detect BASIC errors.

The PC Instruments System error messages and numbers are:

01 Break detected
02 Undefined function — System must be reset
03 PCIBILC.BLD not at paragraph boundary
04 Out of memory
05 File not found
06 Incorrect file format
07 Unable to open the file
08 Unable to close the file
09 End of file while reading
10 Error while reading
11 Disk not ready
12 Primary initialization error
13 ''Panels'' not loaded
14 Input string too big
15 Invalid value
29 Unknown error
30 Can not open error message file, HPERR.HPE
31 Illegal syntax in error message file, HPERR.HPE
32 Can not open hardware configuration file, HPPI.HPH
33 Illegal syntax in hardware configuration file, HPPI.HPH
35 Illegal interface type in file HPPI.HPH
36 Illegal interface address in file HPPI.HPH
40 The instrument being defined is not present on the PCIB
41 Two instruments have been set to the same address
50 Use of undefined label
51 Instrument not found in State file.
52 No match for any module in State file.
60 Cannot close file
61 Cannot create file
62 Cannot write to file
63 Cannot read file
64 Cannot open file
70 Instrument memory space exhausted. Reconfigure PC Instruments
71 PCIB timeout
72 PCIB parity error
73 Illegal reserved constant

NOTE

If your error handling routine returns an error number with the following message: SYSTEM ERROR (Range)— where range can be any number from 0 to 65,535—, an error has occurred that should not have happened during normal operation. Contact your local HP Applications Engineering Organization. Your HP PC Instruments Support Guide lists the HP Applications Engineering Organization nearest you.

Instrument Error Messages

PC Instruments error messages numbered from 100 and up apply to specific instruments. These messages are organized as follows:

100-199	=	Universal Counter
200-299	=	Relay Multiplexer
300-399	=	Digital Multimeter
400-499	=	Digital I/O
500-599	=	Dual Voltage DAC
600-699	=	Function Generator
700-799	=	Digitizing Oscilloscope
800-899	=	Relay Actuator

You can either refer to the applicable Instrument Owner's Guide for these messages or examine the files that were referenced in the introduction.

Glossary

Many terms are listed here to help those new to computer-controlled instrumentation. Other terms are listed because they are used in a way that is specific to HP PC Instruments.

Accessory slot · One of the two covered openings on the rear of the computer designed to accommodate plug-in accessory cards. The PC Instruments Interface Card is installed in an accessory slot.

Address · A number, usually given in hexadecimal, that identifies the instrument on the System Interface Bus. The address is determined by a switch on the instrument. The system instruments are ordered, or arranged in the System View Window, in the same sequence as their bus addresses.

Application program · Those programs that can be started directly from DOS. Also, a BASIC program that controls one or more instruments for a specific application.

Backup copy · A copy of a master diskette that has been duplicated on a diskette or on the fixed disk.

BASIC · Beginner's All-Purpose Symbolic Instruction Code. The programming language used to operate the system in the program mode is an advanced version of BASIC (see BASICA).

BASICA · An advanced version of Microsoft BASIC designed to operate under DOS. PC Instruments applications programs designed for the IBM PC, PC/XT, or PC/AT must be written in BASICA.

Boot · Or boot up. The action that automatically loads the operating system into memory when you turn the computer on. This ''starts'' the computer. You can also boot the system via a ''hard'' reset.

Bootable diskette · Any diskette that includes the operating system program.

Bus address · See ''Address.''

Configure · To specify the hardware parameters for the PC Instruments System.

Cursor · A graphics indicator that shows you where your entry or selection is being made.

DAC · Digital-to-Analog Converter instrument.

Default · A value or a file that is used by the system unless you specify a particular value or filename. The default values for instrument parameters are determined by the instrument hardware.

Default drive · The drive that the computer uses unless instructed otherwise. This normally is ''A'' for a dual-diskette system or ''C'' for a fixed-disk system.

Directory · A listing, or table of contents, of files on a fixed disk or a diskette. See also ''root directory'', ''default directory'', and ''subdirectory.''

Disable · To disconnect (by program command) an instrument's output signal from the system. Instruments that do not generate an output (e.g., the DMM) cannot be disabled.

Disk · Same as ''fixed disk.''

Diskette · A removable, 5¼ inch magnetic mass storage medium that stores programs and data. The standard double-sided diskettes store over 300 Kb of information. The high capacity diskettes store over 1 Mb of information.

Display - The output on the computer screen.

DMM - Digital Multimeter instrument.

DOS - Disk operating system. The master program that must always be loaded into the computer in order for it to operate. (See PC-DOS).

Double-sided diskette - A diskette that stores information on both sides. It requires a double-sided drive.

Drive specifier - A letter followed by a colon (''A:, B:'', etc.) that indicates which drive is to be used.

Enable - To connect (by program command) an instrument's output signal to the system.

Field - An area on the screen reserved to display an instrument name, settings, or a data value.

File - A collection of information stored, under a specific name, on a fixed disk or a diskette.

File extension -A period followed by three letters, added to the filename, to further identify the file. DOS and BASIC have certain reserved file extensions that cannot be used.

Fixed disk - A rigid, large-capacity (two or more megabytes) mass storage device permanently enclosed in a dust-free case.

Flexible diskette - A diskette.

Floppy disk - A diskette.

Function key - A keyboard softkey. One of the keys labled ''F1'' through ''F8'' that correspond to the softkeys at the bottom of the Soft Front Panel display.

Graphics pointer - The arrow that serves as a cursor on the Soft Front Panel display.

Hard disk · See "Fixed disk".

Hard reset · A reset [Alt] [CTRL] [DEL] that effectively "reboots" the operating system. This takes you out of your current application and returns you to DOS. In the process, anything stored in RAM is lost.

HP-IB · Hewlett-Packard Interface Bus. A non-PC Instruments bus used for computer control of instruments. Also known as "IEEE-488."

HPSTATE.HPC · A file that is stored automatically whenever you exit the Soft Front Panel. This file stores the configurations of all system instruments and can be used to recall those configurations.

Initialize · To establish the starting operating conditions for the Soft Front Panel or instruments.

Input instrument · A PC Instruments module (such as a DMM) that performs some operation on an input signal.

Install · To connect a hardware device into the system. Also, to put an application on a diskette other than its original master diskette.

Instrument-specific statement · A PC Instruments library statement that performs some function specific to an instrument.

Instrument state · The condition and values of all the instrument's fields as they appear in the Interactive Instrument Window.

Interactive field · A Soft Front Panel field that can be used for either entering a value or for executing some function.

Interactive Instrument Window · That portion of the Soft Front Panel display reserved for showing the front and rear panels of the currently operable instrument.

Interface number - The number (not address) of the PC Instruments Interface Card. This number is normally 01, unless there are two PC Instruments Interface Cards in the computer.

Kb - Kilobyte. Approximately one thousand bytes, or characters.

Keywork - See ''Reserved word.''

Label - A user-defined or default instrument name. Each label appears in the right-hand corner of the rear panel view and in the System View Window.

Load - To bring data or program information into the computer memory.

Main directory - See ''Root directory.''

Manual mode - The mode of system operation in which you control the instruments from the Soft Front Panel.

Mass storage - A diskette or a fixed disk device that provides large, nonvolatile memory.

Master diskette - Any original program diskette.

Mb - Megabyte. Approximately one million bytes, or characters.

Memory - Any device that stores information. See ''RAM'' and ''Mass storage.''

Menu - A screen display that guides your through a program by prompting you to choose from a list of operations.

Model number - The official HP number given to an instrument or component.

Module · An instrument for use in the PC Instruments System.

MS-DOS · Microsoft Disk Operating System. (See PC-DOS).

Noninteractive field · See ''passive field.''

Nonvolatile memory · A permananent storage device, such as a diskette. Information stored in nonvolatile memory remains there until erased or overwritten.

Operable instrument · The instrument that appears in the Interactive Instrument Window and is under your immediate control.

Operating system · The computer's master program that allows you to use other programs, such as PC Instruments, BASIC, etc.

Order · The sequence in which the system instruments are arranged in the System View Window. See ''Address.''

Overwrite · To replace information in memory or mass storage with new information.

Page · One CRT screen. When a screen cannot hoid all the information that must be displayed, a second screen, or page, is created. You go from one page to another by pressing a softkey.

Pathname · The name that completely defines the location of a file.

Passive field · A Soft Front Panel field that displays information but does not respond when selected.

PC-DOS · A slightly modified version of MS-DOS that is used as the operating system for the IBM PC, PC-XT, and PC-AT. Generally referred to in this manual as ''DOS.''

PCIB - PC Instruments Interface Bus. The bus that connects the PC instruments to each other and to the interface card.

PC Instruments Interface Card - The hardware device installed into one of the computer's accessory slots. It connects the PCIB with the computer's internal bus.

PC Instrument Statement Library - A set of system statements and instrument-specific statements used to control your instruments.

Point - To choose a field on the screen for selection. This can be done either by moving the mouse or by moving the graphics pointer with the cursor control keys.

Point to and select - To choose and select a field on the screen. This can be done by moving the mouse to point to it (choosing) and then pressing the mouse button (selecting). It can also be done by moving the graphics pointer to the field (choosing) and then pressing the $\boxed{\text{Enter}}$ key (selecting).

Program mode - The mode of system operation in which the computer controls the instruments via a BASIC program.

Program Shell - BASIC program code generated from the Store Program Menu of the Soft Front Panel. The Program Shell serves as the framework from which you build an application program to control your instruments.

RAM - Random access memory. The volatile memory inside the computer.

Read - To bring information into the computer memory.

Reserved variable - A variable name, used exclusively by PC Instruments Program Shell, that must not be tampered with.

Reserved word · A word that is used for a special purpose and may not be used for any other purpose within a program. The PC Instruments and Data Aquisition software have reserved words, as do BASIC and DOS.

Reset · See "Hard reset" and "Soft reset."

Root directory · The directory that is automatically created when a diskette (or the fixed disk) is formatted.

ROM · Read only memory. A nonvolatile memory device that can be read, but not written to or erased. A program in ROM allows the computer to "boot" the operating system.

Screen · The computer display.

Secondary address · An identifying name or mnemonic (not an number) that distinquishes a port of a multiport instrument.

Select · To select an interactive field after it has been chosen. This is done by pressing either the mouse button or the keyboard Enter key.

Soft Front Panel · The PC Instruments display that permits operation in the manual mode. Also, the software that generates the display.

Softkey · An operating key that changes its function according to the PC Instruments program. Keyboard keys F1 through F8 are PC Instruments softkeys that duplicate those in the softkey area of the screen.

Soft reset · A reset Ctrl Break that interrupts your current operation without taking you back to the operating system. Whatever is stored in RAM normally is not lost.

Status Window - That portion of the Soft Front Panel display reserved for messages generated by the system or instrument programs, or for file entries by the user.

Subdirectory - A particular subgroup of fixed disk or diskette files. You can create a subdirectory of a root directory and subdirectory of a subdirectory.

System statement - A PC Instruments library statement that performs a function pertaining to all instruments or to a group of instruments.

System View Window - That portion of the Soft Front Panel display reserved for showing all the system instruments.

Volatile memory - Temporary storage that holds information only as long as the computer is turned on. RAM.

Winchester - See "Fixed disk".

Window - An area of the PC Instruments Soft Front Panel display that provides either graphical or textual information.

Work diskette - The diskette used for daily operations. Essential software is copied to this diskette from master diskettes, which are put away in a safe place. With fixed-disk systems, the fixed disk becomes the work disk.

Write - To take information from computer memory.

Write protect - To protect the information on a diskette from being erased or altered. Protection is done by covering a notch on the diskette.

Using HP-IB Instruments with PC Instruments

Introduction

You can combine *HP-IB compatible instruments with HP PC Instruments in the same application and control them from your IBM PC, PC/XT, or PC/AT personal computer. In order to do this, you must have purchased the HP 61062A HP-IB I/O Library package and the applicable HP-IB compatible instrument(s) in addition to your HP PC Instruments System. The HP 61062A HP-IB I/O Library includes a diskette and an HP-IB Interface card (HP Part No. 27209A) which must be installed in the computer. The diskette and interface card allow an IBM PC, PC/AT, or PC/XT computer to control HP-IB compatible instruments.

The HP-IB is separate from the PC Instruments Bus (PCIB) and has no electrical connections to any of the PC Instruments. Files from the HP 61062A Library must be copied onto the same media as the PC Instrument software so that the HP-IB instruments as well as the PC Instruments can be controlled from the same BASIC application program. Note, however, that while an HP-IB compatible instrument is easily programmed from BASIC with the HP-IB commands, it cannot be controlled ''manually'' using the Soft Front Panel feature of the PC Instruments software. Only the PC Instruments have Soft Front Panel representation on the computer's display.

The following paragraphs describe a programming example that uses both HP-IB and PC Instruments. First, the hardware configuration is described; then software setup instructions are given; and finally, a programming example is given with a description of each line in the program.

* The HP-IB, a versatile interconnect system for instruments and controllers, is Hewlett-Packard's implementation of IEEE Standard 488, ''Standard Digital Interface for Programmable Instrumentation''.

The BASIC programming example includes HP-IB commands from the HP 61062A Library as well as programming statements from the PC Instruments Library. Refer to the HP-IB I/O Library Guide (HP Part No. 5957-6306) for complete details on programming HP-IB compatible instruments.

Programming Example

The programming example that is given at the end of this discussion uses an HP 61014A PC Instruments Function Generator and an HP 3456A Digital Voltmeter (HP-IB compatible) to test operational amplifiers. The program tests the response of each op amp to a 2V peak-to-peak signal, swept from 1 kHz to 10 kHz.

Hardware Configuration

Figure S-1 illustrates the hardware configuration required for the programming example. The HP 61014A PC Instruments Function Generator is connected to the IBM PC, PC/AT, or PC/XT computer via the PC Instruments Interface Card which is installed in the rear of the computer. Up to eight PC Instruments can be connected to the PC Instruments Bus (PCIB). Complete installation instructions for PC Instruments including power connections, PC Instruments bus connections, and address switch settings are given in Chapter 2 of this guide.

The HP 3456A Digital Voltmeter (DVM) is connected to the IBM PC, PC/AT, or PC/XT computer via the HP-IB Interface card which is installed in the rear of the computer. One HP-IB Interface card can accomodate up to fourteen HP-IB compatible instruments. Refer to Appendix B in the HP-IB I/O Library Guide (HP Part No. 5957-6306) for instructions on configuring and installing the HP-IB Interface Card. Refer to the Operating Manual for the HP 3456A for instructions on installing and configuring the DVM.

NOTE

To simulate the fixture/op amp combination, use a coaxial cable with meter lead adapters to connect the signal (OUT 50Ω) from the Function Generator to the DVM's VOLTS input.

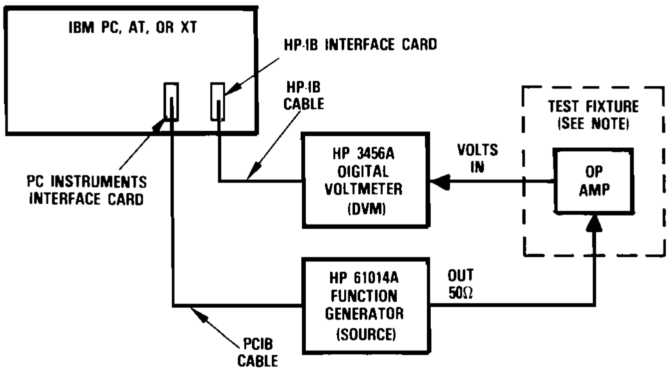


Figure S-1. Hardware Configuration

The programming example that is given later uses an HP-IB select code of 7 (default for the HP-IB Interface Card) and an HP-IB device address of 22 for the DVM. To ensure that the hardware address settings agree with the programming example, you can either; (1) make sure that the HP-IB address switches on the HP-IB Interface Card as well as those on the rear of the DVM are set as specified above or (2), change lines 1080 and 1090 in the programming example to agree with your HP-IB address switch settings.

Copying the BASIC HP-IB Library

Before you can write a BASIC program for this PC Instrument/ HP-IB Instrument configuration, you must copy two files (HPIB.SYN and HPIB.PLD) from the HP 61062A Library diskette onto your PC Instruments work diskette or your fixed disk. The work diskette or the fixed disk must already contain the PCIB subdirectory (Refer to Chapter 3 in this manual). To copy the HP-IB files onto the disc, proceed as follows:

NOTE

If you have dual diskette drives, it is assumed that drive A contains your HP 61062A Library diskette and drive B contains your PC Instruments diskette. If you have a fixed disk, it is assumed that drive A contains your HP 61062A Library diskette and your fixed disk is designated drive C.

---

1. Boot up the system.
2. Insert the HP 61062A diskette into drive A and if you have a PC Instrument work diskette, insert it into drive B.
3. Copy the HPIB.SYN file form the HP 61062A diskette into the PCIB subdirectory on the diskette in drive B (or the fixed disk, drive C) by typing:

```
A> COPY A:HPIB.SYN B:\PCIB
              or
A> COPY A:HPIB.SYN C:\PCIB
```

4. Copy the HPIB.PLD file from the HP 61062A diskette into the PCIB subdirectory on the diskette in drive B (or the fixed disk, drive C) by typing:

```
A> COPY A:HPIB.PLD B:\PCIB
              or
A> COPY A:HPIB.PLD C:\PCIB
```

Creating the Program Shell

After you have copied the HP-IB commands onto your PC Instruments work diskette or fixed disk, you must create a Program Shell that will accommodate your HP-IB instrument (DVM) as well as your PC Instrument

(Function Generator). In order to do this you must follow these steps:

1. Run the PC Instruments Configure program as described in Chapter 3. This causes the PC Instruments System Configuration menu to appear on the computer screen.
2. Answer the questions on the menu by making the appropriate entries from the keyboard. In addition to answering the question about your computer and PC Instruments System, you must answer "Y" (yes) to the HP- IB Option question.
3. Run the PANELS program and set up your PC Instrument using the Soft Front Panel and Soft Rear Panel displays on the computer screen as described in Chapters 3 and 4 in this guide and Chapter 3 in the Function Generator Owner's Guide. The programming example that is given later, assumes that you have assigned the label "SOURCE" on the Soft Rear Panel and did not change any of the Function Generator's default settings (Function = Sine, Frequency = 1.0 Khz, Offset = 0 V, Amplitude = 8 mV, Symmetry = 50%, Mode = Continuous, and Output = Disabled) on the Soft Front Panel (see Chapter 3 in the Function Generator's manual).
4. Using Soft Front Panel system softkeys as described in Chapter 4 of this manual, save the Program Shell under the filename "OPAMP" and then exit the PANELS program.

Writing the BASIC Program

Before you write your program, you must first load the Program Shell (filename "OPAMP") into the BASIC workspace as described in Chapter 5 of this manual. You can now write your BASIC program by inserting PC Instruments programming statements, HP-IB commands, and BASIC statements as shown in the example program below. Be sure to preface each PC Instrument statement and HP-IB command with CALL.

You begin by inserting your program starting at line 1000. Remember that the label "SOURCE" was assigned to the Function Generator and must be used in your PC Instrument programming statements.

NOTE

If the PC Instrument programming statements or HP-IB commands fail to execute, you may have a program error. Refer to Chapter 5 in this guide, and Chapter 2 in the HP-IB I/O Library Guide which discuss error handling methods.

First, you initialize some of the variables. Numeric variables must be single-precision real.

```
1000  'User program starts at this line.
1010  '
1020  CODES$=SPACE$(100)
1030  MIN.FREQ=1000
1040  MAX.FREQ=10000
1050  STEP.FREQ=1000
1060  AMPL=2
1070  '
```

line 1020 - Initialize the string (CODES$) to hold a variable number of HP-IB instrument commands.

line 1030 - Set the minimum (or start) frequency to 1 kHz.

line 1040 - Set the maximum (or stop) frequency to 10 kHz

line 1050 - Define frequency increment (or step frequency) as 1 kHz.

line 1060 - Define the SOURCE output amplitude of 2 V peak-to-peak.

Next, define the HP-IB address variables.

```
1080  ISC=7
1090  DVM=722
1100  '
```

line 1080 - Set the HP-IB interface select code variable ISC to 7.

line 1090 - Set the Digital Voltmeter HP-IB address variable DVM to 722.

Now, define some HP-IB initialization characteristics

```
1110  CALL IORESET(ISC)
1120  TIMEOUT=5
1130  CALL IOTIMEOUT(ISC,TIMEOUT)
1140  ENABLE=1
1150  CALL IOEOI(ISC,ENABLE)
1160  CALL IOREMOTE(ISC)
1170  CALL IOCLEAR(ISC)
1180  '
```

line 1110 - Reset the HP-IB interface bus to clear all previous activity.

line 1120/1130 - Establish a system timeout of 5 seconds.

line 1140/1150 - Execute the IOEOI command with ENABLE of 1. When OUTPUTing, this enables the controller to set the EOI line true on the last byte of the write cycle. When ENTERing, it enables the read cycle to terminate when the EOI line is sensed true.

line 1160 - Set the Interface Remote Enable line (REN); this places any device on the HP-IB bus in the remote mode as soon as it's addressed to listen.

line 1170 - Sets all HP-IB instruments into a known device-dependent state.

The following lines program the PC Instruments signal source.

```
1190  CALL SET.MODE(SOURCE,CONTINUOUS)
1200  CALL SET.FUNCTION(SOURCE,SINE)
1210  CALL SET.AMPLITUDE(SOURCE,AMPL)
1220  '
```

line 1190 - Set the SOURCE to output a waveform continuously.

line 1200 - Set the output wave form to be a sine wave.

line 1210 - Set the amplitude of the output sine wave to be 2 V peak-to-peak.

Next program the HP-IB Voltmeter.

```
1230  CODES$="H F2 R4 Z0 T3"
1240  LENGTH=LEN(CODES$)
1250  CALL IOOUTPUTS(DVM,CODES$,LENGTH)
1260  '
```

line 1230 - Define the programming codes string CODES$ to hold:

H	- software reset the voltmeter
F2	- select the AC volts function
R4	- select the 10 volt range
Z0	- turn off auto zero
T3	- select single trigger

line 1240 - Determines the length of the programming codes string. This is needed for the output command in the next line.

line 1250 - Sends the programming codes string to the voltmeter.

Now that the instruments have been programmed to the required settings, the actual test can be performed.

```
1270  CALL ENABLE.OUTPUT(SOURCE)
1280  '
1290  FOR FREQ=MIN.FREQ TO MAX.FREQ STEP
        STEP.FREQ
1300  CALL SET.FREQUENCY(SOURCE,FREQ)
1310  CALL IOTRIGGER(DVM)
1320  CALL IOENTER(DVM,READING)
1330  LPRINT FREQ,READING
1340  NEXT FREQ
1350  '
1360  END
```

line 1270 - The output sine wave from the SOURCE is started.

line 1290 - A FOR/NEXT loop starts to increment the SOURCE output frequency from 1 kHz to 10 kHz in 1 kHz steps.

line 1300 - Sets the output frequency of the SOURCE.

line 1310 - Triggers the DVM to take a reading.

line 1320 - Gets the reading from the DVM.

line 1330 - Prints each frequency and the measured voltage.

line 1340 - Completes the FOR/NEXT loop.

line 1360 - End the program.

Now that you have finished writing the program, you can save it on a work diskette and run it as described in Chapter 5 of this guide.